

THÈSE

En vue de l'obtention du

DOCTORAT DE L'UNIVERSITÉ DE TOULOUSE

Délivré par *l'Institut National Polytechnique de Toulouse*
Discipline ou spécialité : *Systèmes Informatiques et Systèmes Embarqués*

Présentée et soutenue par *Manel SGHAIRI HAOUATI*
Le *27 mai 2010*

Titre :

Architectures innovantes de systèmes de commandes de vol

JURY
Françoise Simonot-Lion – Présidente
Mireille Bayart – Rapporteur
Laurent Pautet - Rapporteur
Yvon Trinquet – Examinateur
Jean-Jacques Aubert - Examinateur
Patrice Brot - Examinateur
Agnan de Bonneval – Directeur de thèse
Yves Crouzet – Directeur de thèse

Ecole doctorale : *Systèmes*
Unité de recherche : *LAAS-CNRS*
Directeurs de Thèse : *Agnan de Bonneval et Yves Crouzet*

à toi mon cher papa,

à toi ma chère maman,

à mon chéri et notre petit prince R*ayan*

AVANT-PROPOS

Pour débuter ce mémoire de thèse, je tiens à remercier toutes les personnes qui ont contribué à des degrés divers au bon déroulement de ce travail de thèse.

Le travail présenté dans ce mémoire résulte de la collaboration entre la société Airbus France et le Laboratoire d'Analyse et d'Architecture des Systèmes du Centre National de la Recherche Scientifique (LAAS-CNRS) à Toulouse.

J'exprime tous mes remerciements à Monsieur François Polchi, responsable du département Commandes de vol électrique et pilote automatique (EDYCC) et Messieurs Malik Ghallab et Raja Chatila, directeurs successifs du LAAS-CNRS, pour m'avoir accueilli dans leur établissement respectif.

Je remercie également Jean Arlat et Karama Kanoun, les responsables successifs du groupe de recherche Tolérance aux fautes et Sûreté de Fonctionnement informatique (TSF).

Je tiens à exprimer une profonde reconnaissance à mes encadrants académiques Yves Crouzet, Chargé de recherche au CNRS, et Agnan de Bonneval, Maître de conférence à l'Université Paul Sabatier de Toulouse, pour leur soutien permanent au cours de ces trois dernières années, pour leur conseils et leur grande patience, surtout dans les moments de doute, où je suis difficile à convaincre.

Je tiens à exprimer ma gratitude envers mes encadrants industriels, Jean-Jacques Aubert, Responsable de la Recherche pour les Systèmes de contrôle de l'avion au sein du département EDYCC et Patrice Brot, Ingénieur Etudes Avancées Commandes de Vol Airbus dans le département EDYCC, d'abord pour tout le savoir technique qu'ils m'ont enseigné. Par leur approche industrielle, stratégique, et leur suivi hebdomadaire, ils m'ont permis de ne pas me perdre dans mes recherches théoriques pour construire à temps des solutions industrialisables adaptées au monde aéronautique. J'ai également beaucoup apprécié leur écoute, leur disponibilité et leur gentillesse.

Je remercie sincèrement Madame Françoise Simonot, Professeur à l'Institut National Polytechnique de Lorraine qui m'a fait le très grand honneur de présider mon jury et de participer à l'amélioration de mes travaux par ses conseils judicieux. Je remercie également Monsieur Laurent Pautet, Professeur à Télécom ParisTech (ENST) et Madame Mireille Bayart, Professeur à l'Université de Lille 1 pour avoir soigneusement rapporté sur mon manuscrit et Monsieur Yvon Trinquet, professeur à l'Université de Nantes pour avoir accepté avec beaucoup d'amabilité d'examiner mon travail.

Merci à tous les personnes d'Airbus France et du LAAS-CNRS que j'ai pu côtoyer et qui m'ont permis de travailler dans une ambiance chaleureuse.

Merci enfin à mon mari, et à tous les membres de ma famille, qui m'ont toujours apporté leur interminable soutien durant toutes ces années, et qui n'ont jamais cessé d'être à mes cotés, malgré les barrières géographiques.

Bonne lecture !

ACRONYMES

AAA, Adéquation Algorithme Architecture

ACE, *(Actuator Control Electronics)*

AFDX, *(Avionics Full DupleX Ethernet)*

ALIC, *(Application Level Integrity Checking)*

AMDE, Analyse des Modes de Défaillance et de leurs Effets

AMDEC, Analyse des Modes de Défaillance, de leurs Effets et de leur Criticité

BCM, *(Backup Control Module)*

CAN, *(Controller Area Network)*

CCA, *(Common Cause Analysis)*

CDV, Commandes De Vol

CDVE, Commandes De Vol Electrique

COM, voie commande des calculateurs Airbus

COTS, *(Components Off The Shelf)*

FAR, *(Federal Aviation Regulations)*

FC, *(Failure Condition),* condition de panne

FCCP, calculateur primaire dans les nouvelles architectures

FCCS, calculateur secondaire dans les nouvelles architectures

FCRM, *(Flight Control Remote Module)*

FHA, *(Functional Hazard Assessment)*

FMEA, *(Failure Mode and Effect Analysis)*

FMES, *(Failure Mode and Effect Summary)*

GALS, Globalement Asynchrone Localement Synchrone

IMA, *(Integrated Modular Avionics)*

JAR, *(Joint Aviation Requirements)*

MMEL, *(Master Minimum Equipment List),* liste principale d'équipement minimal

MFCC, (*Main Flight Control Computer*), calculateur primaire architecture Falcon

MON, voie Surveillance des calculateurs Airbus

MTBF, *(Mean Time Between Failures),* temps moyen entre défaillances

MTTF, *(Mean Time To Failure),* temps moyen de bon fonctionnement avant défaillance

MTTR, *(Mean Time To Repair),* temps moyen de réparation

PA, Pilote Automatique

PFC, (*Primary Flight Control*)

PHR, plan horizontal réglable

PRIM, *(PRIMary Computer),* calculateur primaire dans l'architecture actuelle Airbus

RCCB, *(Remote Control Circuit Breaker)*

SEC, *(SECondary Computer),* calculateur secondaire dans l'architecture actuelle Airbus

SFCC, *(Secondary Flight Control Computer),* calculateur secondaire architecture Boeing

SSA, *(Safety System Assessment)*

SAO, Spécification Assistée par Ordinateur. (L'atelier SAO est utilisé par l'avionneur pour notamment concevoir les spécifications fonctionnelles des calculateurs).

SCADE, Il a la même fonction que l'atelier SAO, et est aujourd'hui utilisé uniquement sur le FCSC A340-500/600.

TMR, *(Triple Modular Redundancy)*

TTA, *(Time Triggered Architecture)*

TTP, *(Time Triggered Protocol)*

TABLE DES MATIERES

TABLE DES MATIERES III

INTRODUCTION GENERALE 1

CHAPITRE I - COMMANDES DE VOL ELECTRIQUES : ETAT ACTUEL, TENDANCES ET ORIENTATIONS 5

I.1 COMMANDES DE VOL : ROLE ET EXIGENCES 6
I.1.1 Rôle, évolutions et définitions de base 6
I.1.1.1 Rôle des systèmes de commandes de vol 6
I.1.1.2 Évolutions : du tout mécanique au Fly-By-Wire et tendances futures 6
I.1.1.3 Définitions de base 7
I.1.2 Ensemble des exigences à satisfaire par les systèmes de CDVE 8
I.1.2.1 Sûreté de fonctionnement 8
1) Concepts de base 8
2) Les attributs 8
3) Moyens 9
4) Entraves : classification des pannes dans les systèmes de commandes de vol 10
5) Techniques de tolérance aux fautes pour les systèmes de commandes de vol 11
I.1.2.2 Exigences règlementaires 13
I.1.2.3 Exigences supplémentaires : expérience en service et précautions forfaitaires 13
I.1.2.4 Exigences économiques 14
I.1.2.5 Exigences embarqués et temps réel 15
I.1.3 Conclusion 15
I.2 SOLUTIONS ARCHITECTURALES TOLERANTES AUX FAUTES : ETAT DES LIEUX DES ARCHITECTURES ACTUELLES 16
I.2.1 L'architecture du système de CDVE Airbus 16
I.2.1.1 Les calculateurs : architecture et fonctionnement 16
I.2.1.2 Les communications 17
I.2.1.3 Les actionneurs 17
I.2.1.4 Logique de priorité de reconfiguration en cas de défaillance 17
I.2.2 L'architecture du système de CDVE Boeing 18
I.2.2.1 Les calculateurs : architecture et fonctionnement 18
I.2.2.2 Les communications 18
I.2.2.3 Les actionneurs 19
I.2.3 Autre architecture : Falcon 7X 20
I.2.4 Synthèse 21
I.3 TENDANCE FUTURE : LES ENJEUX NOUVEAUX LIES A LA CONCEPTION DES CDVE 22
I.3.1 Utilisation d'actionneurs et capteurs intelligents 22
I.3.2 Utilisation des COTS 23
I.3.2.1 Avionique Modulaire Intégrée : une nouvelle architecture Avion 24
1) L'origine 24
2) Architecture IMA (Integrated Modular Avionics) : le principe 24
3) L'IMA chez AIRBUS 24
I.3.2.2 AFDX : une nouvelle technologie réseau Avion 25
1) Les nouveaux besoins et dernières évolutions 25
2) AFDX (Avionics Full DupleX switched ethernet) 26
I.3.3 Panorama des bus avioniques actuels 27
I.3.3.1 Étude comparative 27
I.3.3.2 Intégrité des communications – cas de la solution ALIC (Application Level Integrity Checking) 29
I.3.4 Orientations : vers des architectures distribuées 30
I.4 CONCLUSION 32

CHAPITRE II - UNE DEMARCHE INCREMENTALE DE CONCEPTION D'ARCHITECTURE 33

II.1 HYPOTHESES ET DEFINITION DE LA DEMARCHE INCREMENTALE 34
II.1.1 Objectifs et hypothèses de travail.....34
II.1.2 Orientations et démarches possibles.....35
II.1.3 Démarche de conception « amont » pour les CDVE.....37
II.1.4 Processus de conception architecturale.....38
II.2 APPLICATION DE LA DEMARCHE INCREMENTALE.....40
II.2.1 Analyse préliminaire (étape 1).....40
II.2.2 Distribution de l'intelligence (étape 2).....40
II.2.3 Briques de base et architecture primaire (étape 3).....41
II.2.4 Analyse des exigences non fonctionnelles (étape 4).....43
II.2.4.1. Exigences globales (pour tout le système) 43
II.2.4.2. Exigences calculateur.....43
II.2.4.3. Exigences actionneur 44
II.2.4.4. Exigences réseau.....45
II.2.4.5. Exigences d'installation 45
II.2.5 Injection des exigences et optimisation (étape 5).....45
II.2.6 Architecture finale ou optimale (étape 6).....47
II.3 ANALYSE QUANTITATIVE DE SURETE.....48
II.3.1 Rappel de définitions et de propriétés générales 48
II.3.1.1. Fiabilité, taux de défaillance et MTBF.....48
II.3.1.2. Calcul des probabilités des défaillances.....48
II.3.2 Exemple de calcul pour le sous-système calculateurs.....49
II.4 NOUVEAUX CONCEPTS ARCHITECTURAUX ISSUS DE LA DEMARCHE : PRINCIPES DE FONCTIONNEMENT....52
II.4.1 Option 1 : Architecture à vote massif.....52
II.4.2 Option 2 : Architecture à priorité 53
II.5 CONCLUSION 54

CHAPITRE III - NOUVELLE ARCHITECTURE A VOTE MASSIF.....55

III.1 DESCRIPTION GENERALE : PRINCIPES.....56
III.2 DESCRIPTION DETAILLEE DES DIFFERENTS COMPOSANTS DE L'ARCHITECTURE 58
III.2.1 Calculateurs CDVE.....58
III.2.1.1. Architecture matérielle et logicielle 58
III.2.1.2. Description fonctionnelle.....59
1) Interface système 59
2) Architecture fonctionnelle de base.....60
3) Principe de fonctionnement et logiques spécifiques (niveau calculateur).....61
III.2.2 Les électroniques locales FCRM.....63
III.2.2.1. Architecture matérielle et logicielle 63
III.2.2.2. Description fonctionnelle.....64
1) Interface système 64
2) Architecture fonctionnelle de base.....64
3) Principe de fonctionnement et logiques spécifiques (niveau FCRM) 65
a) Les logiques classiques.....65
b) Nouvelle logique de choix : traitement optimisé hybride.....67
c) Reconfiguration 70
d) Logique de validation 70
e) Commande et surveillance des gouvernes 70
III.2.3 Les relais d'alimentation : les RCCB.....70
III.2.3.1. Architecture matérielle et logicielle 70
III.2.3.2. Description fonctionnelle.....71
1) Interface système 71
2) Architecture fonctionnelle 71
3) Principe de fonctionnement et logiques spécifiques (niveau RCCB).....72
III.2.4 Réseaux de communication 72
III.3 PREMIERE ÉVALUATION : MODES DE DEFAILLANCES ET MECANISMES DE DETECTION 73
III.4 CONCLUSION 75

CHAPITRE IV - MODELISATION ET VALIDATION DE L'ARCHITECTURE A VOTE MASSIF....77

IV.1 PRINCIPES DE VALIDATION ORIENTEE « SECURITE » 78
IV.1.1 Processus actuel d'analyse de la sûreté de fonctionnement....78
IV.1.1.1 Démarche....78
IV.1.1.2 Techniques....79
IV.1.2 Nouvelle approche d'analyse de sécurité fondée sur les modèles....79
IV.1.2.1 AltaRica et plate-forme associée....80
1) Langage....80
2) Syntaxe du langage....80
3) Illustration....81
4) Sémantique formelle....82
5) Simulation et Analyse....82
a) Simulation interactive....82
b) Génération des modèles de sûreté de fonctionnement....82
c) Analyse quantitative....82
IV.2 VALIDATION ORIENTEE SECURITE SOUS L'ENVIRONNEMENT OCAS/ALTARICA....83
IV.2.1 Cas d'étude : commande en tangage....83
IV.2.2 Principe de fonctionnement : rappel....84
IV.2.3 Exigences de sécurité....84
IV.2.4 Modélisation en Altarica....84
IV.2.5 Simulations et Analyse de sécurité....87
IV.2.5.1 Analyse qualitative....87
IV.2.5.2 Analyse quantitative....89
IV.2.5.3 Bilan....92
IV.3 VALIDATION ORIENTE ROBUSTESSE SOUS L'ENVIRONNEMENT MATLAB/SIMULINK 93
IV.3.1 Robustesse : comportement vis-à-vis des asynchronismes....93
IV.3.2 Outil de simulation retenu : Matlab/Simulink....93
IV.3.3 Modélisation sous l'environnement Matlab/Simulink....94
IV.3.3.1 Exemple de modules de base....94
IV.3.3.2 Exemple de modules développés....95
IV.3.3.3 Modèle final de simulation....96
IV.3.3.4 Simulation....97
1) Simulation hors asynchronisme....97
2) Simulation et validation de l'architecture avec asynchronisme et cas de panne....99
IV.4 CONCLUSION....103

CHAPITRE V - ARCHITECTURE À PRIORITÉS....105

V.1 DESCRIPTION GENERALE : PRINCIPES....106
V.2 DESCRIPTION DETAILLEE DES DIFFERENTS COMPOSANTS DE L'ARCHITECTURE....109
V.2.1 Calculateur CDVE....109
V.2.1.1. Architecture matérielle et logicielle....109
V.2.1.2. Description fonctionnelle....109
1) Interface système....109
2) Architecture fonctionnelle de base....110
3) Principe de fonctionnement et logiques spécifiques (niveau calculateur)....111
V.2.2 Les électroniques locales FCRM....111
V.2.2.1. Architecture matérielle et logicielle....111
V.2.2.2. Description fonctionnelle....111
1) Interface système....111
2) Architecture fonctionnelle de base....112
3) Principe de fonctionnement et logiques spécifiques (niveau FCRM)....113
V.3 ARCHITECTURE A PRIORITES : VALIDATION....114
V.4 CONCLUSION....117

CONCLUSIONS ET PERSPECTIVES....119

REFERENCES BIBLIOGRAPHIQUES....125

ANNEXE 1 - ARCHITECTURES RESEAUX DE COMMUNICATION ET TOPOLOGIES POSSIBLES....133

ANNEXE 2 - PROCESSUS DE SECURITE POUR LES SYSTEMES AERONAUTIQUES....137

INTRODUCTION GENERALE

Depuis quelques années, les progrès technologiques sans cesse croissants des actionneurs/capteurs intelligents et des communications numériques ont facilité le développement, dans certains domaines, des systèmes de commande-contrôle de plus en plus complexes et intelligents. Mais tous les systèmes de ce type ne bénéficient pas encore pleinement de ces progrès. C'est notamment le cas des systèmes de *Commandes De Vol (CDV)* dans le domaine de l'aviation civile. Ce type de système de commande-contrôle est embarqué, réparti, temps réel et critique, de par le service qu'il doit rendre : il gère la trajectoire de l'avion en agissant sur ses gouvernes à partir des consignes données par le pilote

Principalement, la dernière génération des capteurs et des actionneurs dits intelligents intègrent, sous la forme d'un microcontrôleur ou de processeur, des capacités importantes de stockage et de traitement, et des dispositifs évolués de communications numériques, allant jusqu'à l'intégration de technologies sans fil (GSM, wifi, bluetooth, etc.). De plus, ces composants sont aujourd'hui de taille très réduite, légers et économes en énergie, et la production en masse de tels composants, du fait d'une utilisation multidomaines très large, permet de rentabiliser les coûts de développement et conduit à un prix unitaire très faible. Enfin, si le besoin de disposer d'un composant davantage « taillé sur mesure » se fait sentir pour le système de commandes de vol, il serait également possible aujourd'hui de développer son propre SOC (System On Chip) en faisant appel à des conceptions « sur étagère », plus connues sous le nom de IP (Intellectual Property). Il est donc finalement possible d'intégrer, sur une même puce, les dispositifs nécessaires taillés au plus juste en termes de capacités de stockage de traitement et de communication, tout en y intégrant des capteurs physiques.

Mais aussi, les récentes technologies de communication, en offrant des niveaux de robustesse (intégrité et déterminisme) et de débits de plus en plus élevés, facilitent la généralisation des médias numériques et l'optimisation des architectures réseaux dans les systèmes critiques.

Et donc, aujourd'hui, la maturité de ces deux technologies, largement utilisées dans des domaines autre que l'avionique, ouvre de nouvelles opportunités d'évolutions dans le développement de nouvelles architectures distribuées pour les systèmes avioniques critiques.

C'est dans ce contexte que mon travail de thèse CIFRE avec AIRBUS s'intéresse à la modernisation des systèmes de CDV Électriques (CDVE). Ainsi, le Concorde a vu l'apparition du premier système de *CDVE*, à base de calculateurs analogiques. La génération suivante est née avec l'Airbus A320, et le passage à des calculateurs désormais numériques, mais toujours reliés par des liaisons directes et analogique aux capteurs et actionneurs. Le programme en cours (A350) va voir l'introduction de la première génération d'actionneurs et de capteurs intelligents pour l'avionique, c'est-à-dire dotés d'une électronique locale évoluée.

Pionner sur ces différentes évolutions, Airbus prévoit, dans ses programmes futurs, de passer à du « tout numérique » pour les communications et à davantage de capteurs et actionneurs encore plus intelligents, dont l'utilisation, justement en réseau, accroîtra fortement les possibilités de distribution de l'intelligence du système, intelligence jusqu'à alors très fortement centralisée dans les calculateurs.

Quoi qu'il en soit, les nouvelles évolutions sont telles, qu'il est maintenant incontournable de repenser la conception des architectures de CDVE.

L'objectif principal de ma thèse a été de définir comment concevoir de nouvelles architectures pour le système de commandes de vol mettant à profit les évolutions précédemment citées, et de définir de telles architectures. Bien entendu, la recherche de nouveaux systèmes plus économes en ressources matérielles ne peut se faire qu'en veillant à ce qu'ils satisfassent au moins le même niveau de sécurité que les systèmes actuels. Et même en fait, des exigences encore plus fortes de fiabilité opérationnelle sont à considérer afin de répondre aux demandes croissantes des compagnies aériennes.

Mener à bien un tel travail exige de commencer par une analyse des architectures actuelles des systèmes de CDVE chez les plus grands avionneurs mondiaux, pour bien appréhender les moyens de sûreté de fonctionnement utilisés et bien identifier les différentes exigences qui guident le développement d'un tel système. Il faut également identifier et analyser les différentes technologies potentiellement candidates pour la définition de nouvelles architectures, en privilégiant celles déjà éprouvées, et déjà intégrées en particulier dans les systèmes avioniques actuels autres que les systèmes de CDVE, telles que l'AFDX ou l'IMA. Enfin, cette première phase a conduit à fixer les hypothèses fondamentales de travail suivantes :

1) les calculateurs sont désormais des unités « simplex » (c'est-à-dire à une seule voie),
2) les actionneurs, eux, ont désormais une architecture de type COM/MON,
3) les communications entre tous les composants sont intégralement numériques, en « full broadcast » (tous les composants peuvent tous communiquer entre eux).

Les résultats de cette première phase ont mis en évidence la nécessité d'utiliser une démarche plus formalisée qu'actuellement en phase « amont » de la conception des architectures, et destinée à dimensionner le système en définissant au mieux le nombre d'éléments redondants nécessaires. D'où la deuxième phase de mon travail. Un bref tour d'horizon des démarches existantes montre qu'aucune n'est bien adaptée au contexte des systèmes de commandes vol. Il s'avère donc nécessaire de définir une nouvelle démarche, plus adaptée à ce contexte. La nouvelle démarche qui a été définie, permet, en partant d'une configuration architectural purement fonctionnelle, donc « minimale », d'aboutir à une configuration « optimale » en ressources matérielles et logicielles tout en satisfaisant l'ensemble des exigences. Cette démarche est dite incrémentale, parce qu'à partir de briques de base *calculateur*, *actionneur* et *réseau*, on injecte progressivement les différentes exigences, et l'on détermine à chaque étape le niveau de redondance à ajouter selon l'exigence traitée.

Une fois cette démarche établie, une troisième phase consiste à proposer plus finement des architectures optimales en définissant précisément les moyens et les mécanismes pour mettre en œuvre la sûreté de fonctionnement. C'est principalement, l'introduction de communications « tout numérique » et d'actionneurs intelligents qui permet de déporter certaines fonctions de surveillance au niveau des actionneurs et d'abandonner ainsi le principe du COM/MON (Airbus) ou TMR (Boeing) au niveau des calculateurs actuels. Deux types d'architectures sont proposés : une dite « *à vote massif* » et une dite « *à priorités* », qui se distinguent par le niveau d'intelligence des équipements concernés et des mécanismes de tolérance aux fautes pour la consolidation des ordres à appliquer aux gouvernes.

Mais la pertinence et les intérêts des architectures proposées ne peuvent prendre toute leur dimension que si elles sont soumises à des vérifications de leurs conformités avec les exigences de sécurité et de robustesse des systèmes CDVE Airbus. Ainsi, la quatrième et dernière phase du travail a été consacrée à la modélisation et à la validation.

*
* *

Ce mémoire de thèse est organisé en cinq chapitres qui sont précédés par une liste des principaux acronymes utilisés.

Après un bref rappel des concepts de base de la sûreté de fonctionnement et plus particulièrement des mécanismes de tolérance aux fautes dans le contexte avionique, ***le premier chapitre*** dresse un bilan des exigences à satisfaire : exigences règlementaires, supplémentaires (précautions forfaitaires et expériences en service), économiques (dont la fiabilité opérationnelle). S'en suit une étude, sous l'angle de la sûreté de fonctionnement, des architectures conçues par différents constructeurs d'avions civils : Airbus, Boeing et Dassault Aviation. Le bilan de cette étude fait apparaître en point commun une apparente « sur-redondance » au niveau des calculateurs, qui est cependant justifiée par le souci d'une démonstration de sécurité dictée par les réglementations de certification et par la préoccupation des constructeurs de ne pas proposer un produit taillé au plus juste. Enfin, ce chapitre présente les nouvelles technologies dont l'utilisation est recommandée pour les nouvelles architectures.

Le deuxième chapitre présente la démarche proposée pour définir la conception de nouvelles architectures, en phase amont, et dans un objectif de dimensionnement du système. En effet, suite à une analyse des approches existantes pour le partitionnement Matériel/Logiciel et l'Adéquation Algorithme Architecture (AAA) du domaine des systèmes embarqués, il s'avère que ces approches sont mal adaptées aux besoins du cadre de nos travaux. Et il est donc nécessaire de développer une nouvelle démarche. Il s'agit d'une méthode incrémentale où les différentes exigences, identifiées dans le premier chapitre, sont injectées par étapes successives de manière à prévoir au mieux les redondances nécessaires pour les couvrir. Cette nouvelle démarche, en sept étapes, est d'abord présentée, puis appliquée à un cas d'étude pour montrer sa faisabilité, son mode d'emploi et ses intérêts. Au final, si on devait résumer ici en une phrase, le cas d'étude montre qu'on peut obtenir une architecture à 6 calculateurs à comparer avec l'architecture actuelle qui en comporte 12 (mais attention, la comparaison complète ne se limite pas à ce simple aspect « comptable »). Dans tous les cas, à l'issue de ce deuxième chapitre, deux pistes d'architectures sont engagées, selon les mécanismes de gestion de la redondance envisagés : une architecture « *à vote massif* » et une « *à priorités* ».

Le troisième chapitre est entièrement consacré à la description de l'architecture « *à vote massif* », qui est multi-maîtres, tolérante aux fautes, et qui est basée sur trois types de briques de base : calculateur simplex, actionneur intelligent de type COM/MON et switch ou micro-switch AFDX. Les fonctions remplies par l'unité MON des calculateurs à structure duplex de l'art antérieur sont mises en œuvre par les actionneurs conjointement avec les autres calculateurs. En effet, le niveau d'intelligence considéré pour les actionneurs leur permet de déterminer par eux-mêmes, à partir des ordres calculateurs qui leur sont transmis, sur quels ordres se baser pour définir l'action à exécuter. En fait, ce chapitre est un des cœurs du travail réalisé, avec la description de la pleine exploitation de la répartition de l'intelligence.

Le quatrième chapitre est consacré à la validation de l'architecture « *à vote massif* » proposée au chapitre III en se basant sur les outils de modélisation OCAS avec le langage Altarica, et l'environnement Matlab/Simulink. L'objectif est la vérification de la conformité de l'architecture avec les exigences de sécurité et la vérification de sa robustesse. Ce chapitre présente les besoins de validation et les outils de modélisation retenus dans le cadre de la thèse, et présente les prototypes d'architectures que nous avons réalisés, les scénarios de simulation sur lesquels nous nous sommes appuyés pour valider l'architecture à vote massif.

Le dernier chapitre décrit le second type d'architecture, dit « *à priorités* ». La description sera fortement réduite par rapport à l'architecture « *à vote massif* », car, pour éviter trop de redites, on se limitera à ne présenter que les caractéristiques qui diffèrent du premier type d'architecture et des architectures de l'état de l'art actuel. Et pour les mêmes raisons, les aspects de modélisation et de simulation pour validation sont intégrées dans ce chapitre, et non dissociés dans un sixième chapitre. Le concept de base de cette architecture, qui est de type maître/esclaves ou maître/valideurs, est qu'un calculateur « *valideur* » ne décide pas lui seul si l'ordre du calculateur « *maître* » doit être transmis à l'actionneur : l'ordre du maître est systématiquement transmis à l'actionneur et c'est l'actionneur lui-même qui, grâce aux moyens logiques qu'il comporte, décide, en fonction du résultat renvoyé par le (ou les) calculateur(s) valideur(s) du calculateur maître, d'exécuter ou non l'ordre du calculateur maître.

Chapitre I - COMMANDES DE VOL ELECTRIQUES : ETAT ACTUEL, TENDANCES ET ORIENTATIONS

Ce chapitre introduit le contexte dans lequel les travaux de thèse se sont déroulés, la problématique abordée et les hypothèses de travail qui ont été prises dans ce cadre. Le sujet de la thèse est large et touche plusieurs domaines : systèmes critiques et distribués, commandes de vol, sûreté de fonctionnent, réseau de communication, etc. Il est donc très difficile de dresser un état de l'art à la fois complet et succinct de tous les domaines touchés. L'objectif de ce chapitre, qui comporte trois parties, est d'introduire les principales notions utiles pour la suite de la thèse.

La première partie commence par une brève description de l'historique et du rôle des systèmes de commandes de vol électrique « Fly-By-Wire ». Sont ensuite détaillés les exigences à considérer dans la suite du travail réalisé : les exigences règlementaires du domaine de l'aviation civile, les exigences liées aux précautions forfaitaires et aux retours d'expériences des avions en service, les exigences liées au caractère embarqué et temps-réel des systèmes visés, et enfin, les exigences économiques. La majorité de ces exigences relevant de la sûreté de fonctionnement, leur description sera précédée d'un rappel des définitions générales de la sûreté de fonctionnement, néanmoins restreint à ce qui est appliqué dans les domaines visés.

La deuxième partie présente les architectures et les technologies qui assurent le fonctionnement actuel des commandes de vol électriques chez trois des plus grands constructeurs d'avions civils : Airbus, Boeing et Dassault Aviation. On mettra en évidence la manière dont les exigences précédemment décrites sont satisfaites par les trois avionneurs, avec en point de mire une analyse des niveaux et types de redondance actuellement appliqués.

La troisième partie présente les nouvelles tendances technologiques envisagées pour l'avionique du futur, à court et à long terme dans le domaine de l'aviation civile, avec principalement : a) la distribution de l'intelligence avec comme support l'emploi de capteurs et actionneurs intelligents ; b) l'utilisation d'un réseau de communication intégralement numérique.

I.1 COMMANDES DE VOL : ROLE ET EXIGENCES

I.1.1 Rôle, évolutions et définitions de base

I.1.1.1 Rôle des systèmes de commandes de vol

En aéronautique, le système de Commandes De Vol (CDV) est le système embarqué qui contrôle la trajectoire de l'avion en agissant sur ses gouvernes à partir des consignes données par le pilote : il fait le lien entre le pilote et les gouvernes qui sont des surfaces aérodynamiques mobiles. Il s'agit de l'ensemble des éléments compris entre les organes de pilotages en cockpit et les gouvernes destinés à gérer l'attitude, la trajectoire et la vitesse de l'avion en mode de pilotage manuel et automatique. Les références [Traverse *et al.* 2004, Lin 1993, Lower *et al.* 2005] présentent des exemples de systèmes de commandes de vol.

I.1.1.2 Évolutions : du tout mécanique au Fly-By-Wire et tendances futures

L'architecture du système de commandes de vol a évolué très significativement au cours des précédentes décennies. Dès les débuts de l'aviation, des surfaces articulées (gouvernes), et donc mobiles, ont été introduites pour le contrôle basique de l'aéronef. Ces gouvernes sont manipulées par le pilote via un système de câbles mécaniques. Cette technique a survécu pendant des décennies et est encore utilisée pour les avions de petite taille comme les avions de loisir. Avec l'introduction des avions de plus grande taille, les capacités musculaires du pilote n'étaient plus suffisantes pour contrer les forces aérodynamiques qui s'opposent au braquage des gouvernes. Ainsi sont apparus les systèmes servo-commandés pour fournir aux actionneurs la puissance nécessaire pour commander les différentes surfaces mobiles qui contrôlent l'avion. Ce principe est illustré sur la Figure I.1.a.

Mais l'évolution, encore plus marquante, a été dans les années 1970, l'introduction de la technique des Commandes De Vol Électriques, notée CDVE, aussi dénommée « *Fly-By-Wire* » ou « *FBW* ») [Briere & Traverse 1993, Langer *et al.* 1992, Thomas & Ormsby 1994]. Fondée sur le traitement, par un calculateur, des signaux de la demande en déflexion du pilote avant de les convertir en courant vers les actionneurs, cette technique a conduit à une architecture qui est schématiquement décrite par la Figure I.1.b.

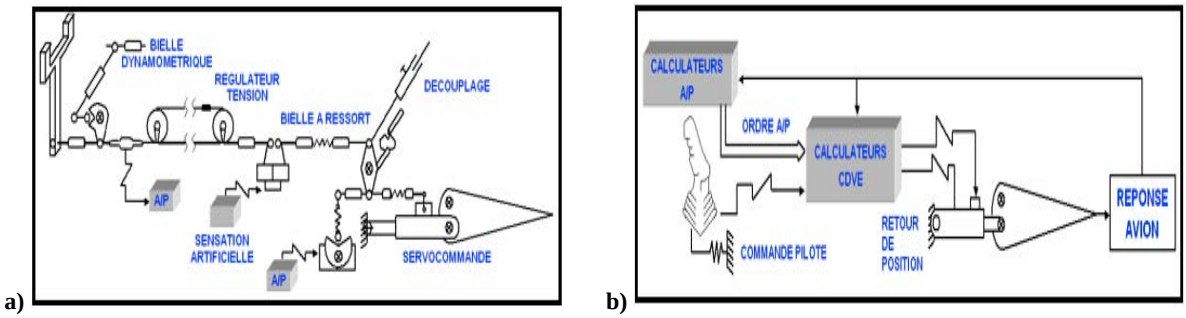


Figure I.1 - a) Commandes de vol mécanique avec servocommande des actionneurs b) Commandes de vol totalement électriques

Le pilote dispose dans le cockpit de moyens de commande (mini manches, palonnier, levier des aérofreins) pour agir sur l'attitude et la trajectoire de l'avion. La consigne du pilote est transformée en un ensemble de signaux électriques vers des calculateurs inter-communicants mais indépendants, qui traitent cette consigne pour envoyer, d'une manière asservie à l'état du vol, les ordres adéquats aux actionneurs des surfaces mobiles. Les calculateurs

échantillonnent les consignes du pilote, mais également d'autres données sur les conditions de vol, telles que les données inertielles, les positions des actionneurs.

A ses débuts, dans les années 70, l'architecture Fly-By-Wire a été développée comme une technologie analogique (cartes analogiques), le Concorde étant le premier avion civil à en être équipé Elle a ensuite évoluée vers des technologies numériques, avec d'abord un développement pour les avions militaires (F-8), puis une utilisation dans les années 80 par Airbus dans le domaine civil, au début avec l'A320, suivi par les autres programmes.

Le système de CDVE actuel est constitué d'équipements géographiquement répartis dans l'avion : des capteurs d'ordres au niveau du cockpit, divers capteurs (anémomètres, gyromètres, accéléromètres…) en des points précis de l'avion, des actionneurs (hydrauliques ou électriques) au niveau des diverses surfaces mobiles, et enfin des calculateurs en soute électronique qui concentrent pratiquement toute l'intelligence du système : toutes les fonctions logicielles embarquées de commandes de vol sont gérées par les calculateurs. Ces calculateurs communiquent entre eux et avec le reste de l'avionique par voie numérique, mais, par contre, leurs communications avec la plupart des capteurs et actionneurs sont à base de liaisons directes analogiques.

Le système entier présente une haute redondance dans le but d'avoir au moins le même niveau de sûreté qu'un système mécanique ou hydraulique. Cette redondance se présente sous de multiples formes (duplex, triplex, quadruplex) ou voies parallèles et indépendantes qui génèrent et transmettent les signaux, vers et à partir des calculateurs qui traitent ces signaux.

Les bénéfices apportés par le système « Fly-By-Wire » et ses évolutions futures sont multiples : protection du domaine de vol, réduction de la masse, maintenance, installation, allégement de la charge de travail du personnel navigant.

Aujourd'hui, on doit s'attendre à des nouvelles évolutions architecturales des systèmes de CDVE. D'une part, certaines évolutions technologiques permettent d'envisager très sérieusement une distribution beaucoup plus importante de l'intelligence. Et d'autre part, de toute manière, il faut tenir compte de l'augmentation générale de la complexité du système avionique. Ainsi, entre 1983 (A310) et 1993 (A340), le nombre d'équipements embarqués a augmenté de 50% (de 77 à 115 unités), alors que la puissance embarquée a été multipliée par 4, passant de 60 à 250 MIPS (Million d'Instructions Par Seconde). De manière générale, à l'instar de la loi de Moore, l'augmentation du nombre de transistors intégrés sur une même puce, il a été établi que la complexité des systèmes avioniques double tous les 5 ans.

I.1.1.3 Définitions de base

Nous définissons ici des expressions de base qui seront utilisées tout au long de ce mémoire.

a) *Loi de pilotage* : le calcul des lois correspond au calcul des ordres de commande ou d'asservissement des gouvernes, à partir des informations provenant de capteurs ou des ordres pilotes. C'est la fonction principale que réalise un calculateur commandes de vol.
b) *Asservissement ou commande de gouverne* : la commande des gouvernes correspond à la transmission de l'ordre d'asservissement du calculateur vers les gouvernes.
c) *Loi normale* : loi de pilotage valable sur tout le domaine de vol. Les calculateurs commandes de vol vont rejeter et/ou ajuster des consignes de pilote qui risquent de dépasser les maximums de facteurs de charge supportable par la structure de l'avion dans le but de garder l'avion dans un domaine d'opération ou de vol sûr et sécurisé). L'avion est totalement protégé contre les sur-vitesses, le décrochage, les excès de facteurs de charge...
d) *Loi directe (ou simple)* : loi de pilotage valable sur une partie de domaine de vol.

I.1.2 Ensemble des exigences à satisfaire par les systèmes de CDVE

Le système de commandes de vol est un système temps-réel embarqué. Il doit donc satisfaire non seulement des exigences temps-réel très strictes, mais surtout des exigences de sécurité très élevées. C'est en effet le système le plus critique à bord d'un avion : ses défaillances éventuelles peuvent induire des conséquences catastrophiques, à savoir la perte de vies humaines et/ou de l'appareil. C'est pourquoi le taux d'occurrence des évènements dits catastrophiques pour le système de CDVE doit être inférieur à 10^{-9} par heure de vol.

En fait, la conception et le développement d'un système « Fly-By-Wire » doit impérativement remplir tout un ensemble d'exigences. Si les exigences temporelles et plus largement celles liées à la démonstration de sûreté de fonctionnement restent les plus essentielles, il ne faut pas oublier que le système doit notamment remplir des exigences économiques pour que l'avion reste compétitif. Donc, en général, on classe ces exigences en quatre catégories : *techniques, réglementaires, supplémentaires,* et *économiques*.

Cependant, une partie importante de ces exigences relèvent de la sûreté de fonctionnement. Nous commençons donc par un tour d'horizon des principaux concepts et moyens de la sûreté de fonctionnement, mais limité aux besoins du cadre avionique du travail présenté.

I.1.2.1 Sûreté de fonctionnement

1) Concepts de base

Comme tout système critique, le système de commandes de vol requiert une sûreté de fonctionnement élevée. La *S*ûreté *d*e *F*onctionnement (SdF) d'un système peut être définie comme étant : *« la propriété qui permet à ses utilisateurs de placer une confiance justifiée dans le service qu'il leur délivre ; le service délivré par un système est son comportement tel que perçu ou requis par ses utilisateurs »*. Nous définissons les concepts de base en utilisant la terminologie de J.C Laprie [Avizienis *et al.* 2004, Laprie & Randell 2001, Laprie *et al.* 1996]. La notion de confiance est fondamentale puisque tout système matériel/logiciel peut contenir des fautes, comme les fautes du matériel ou de conception du logiciel. Cette définition très générique peut se décliner en trois composantes :

- Les « *attributs* » de la SdF : selon les applications auxquelles le système est destiné, la sûreté de fonctionnement peut être vue selon des attributs différents, mais complémentaires, qui permettent de la caractériser. Ce sont des *aptitudes* (ex. : fiabilité, disponibilité) que le système doit présenter ou vérifier avec un certain niveau.
- Les *« entraves »* à la SdF : ce sont les phénomènes qui nuisent à la SdF, ou dit autrement, qui réduisent/limitent les aptitudes évoquées ci-dessus (ex. : rayonnements altérant les mémoires ou les communications, fautes dans la conception d'un logiciel).
- Les *« moyens »* de la SdF : ce sont les méthodes et les techniques mises en place pour limiter, contourner ou simplement évaluer les effets des entraves (ex. : tripler une fonction et effectuer un vote sur les 3 résultats pour « voter » la bonne valeur à retenir).

2) Les attributs

Les attributs de la sûreté de fonctionnement peuvent être nombreux. Nous nous limitons ici à définir ceux qui nous intéressent, à savoir :

a) <u>*La fiabilité*</u>, si l'on se place du point de vue de la continuité du service que le système doit accomplir. Dans ce cas, la sévérité des défaillances n'est pas prise en compte. Un exemple de mesure de fiabilité (temps de bon fonctionnement) est le MTTF (*Mean Time To Failure*), temps moyen jusqu'à l'occurrence de la première défaillance qui est l'inverse du taux de défaillance.

b) *La maintenabilité*, si l'on s'intéresse à l'aptitude aux réparations et aux évolutions du système, et lorsque la maintenance doit être accomplie dans des conditions données avec des procédures et des moyens prescrits. Un exemple de mesure de maintenabilité est le MTTR (*Mean Time To Repair*), le temps moyen de réparation ou de restauration du système dans l'état de bon fonctionnement.
c) *La disponibilité*, par rapport au fait d'être prêt à l'utilisation. Ce qui importe, c'est que le service correct puisse être fourni au moment où l'utilisateur en a besoin. La disponibilité est une mesure sans unité. Elle correspond à la proportion du temps de bon fonctionnement sur le temps total d'exécution du système. La disponibilité est tributaire à la fois de la fiabilité et de la maintenabilité.
d) *La sécurité–innocuité (safety)*, si l'on s'intéresse à la non occurrence de défaillances catastrophiques, c'est–à–dire celles pour lesquelles les conséquences sont inacceptables vis–à–vis du risque encouru par les utilisateurs du système. Accroître ce type de sécurité d'un système peut parfois se faire au détriment de sa disponibilité. Dans le cas d'une centrale nucléaire, par exemple : arrêt d'un réacteur pour éviter la fusion du cœur.
e) *La sécurité–confidentialité (security)*, par rapport à la prévention d'accès ou de manipulations non autorisées de l'information, ou de façon plus générale, si la préoccupation essentielle est de lutter contre les fautes intentionnelles (virus).

Dans le cadre de mon travail, certaines hypothèses ont été prises en considérant que l'évaluation de ces différentes mesures de sûreté de fonctionnement fait souvent appel à des calculs de probabilités, en particulier pour les défaillances survenant de façon aléatoire. Et donc, sans entrer dans le détail, on retiendra que c'est la loi exponentielle qui est la plus fréquemment utilisée en fiabilité, car elle se prête bien aux calculs. Cela conduit à considérer que le taux de défaillance est une constante, hypothèse réaliste ou du moins acceptable, dans la plupart des cas.

3) Moyens

Il existe toute une panoplie de moyens pour assurer ou évaluer la sûreté de fonctionnement d'un système. Une classification usuelle de ces moyens conduit à quatre catégories.

a) *L'évitement des fautes :* l'action d'éviter les fautes consiste à réduire autant que possible leur nombre en se basant sur un processus de développement rigoureux.

   Le processus de développement des CDV est soumis à des règlements. Les règles sont formulées par les *autorités de certification*. Pour l'Europe, l'autorité de certification est l'EASA. Cette agence est garante de la qualité et de la sécurité des avions construits. Cette notion de qualité étant non quantifiable sur l'avion, l'EASA s'assure de la qualité du processus de développement, en ayant établi des règles que l'avionneur doit suivre durant toute la phase de développement : JAR et CRI. Les ingénieurs d'Airbus se sont basés sur les recommandations qui sont définies dans les documents ARP4754 5 et DO-178B/ED-12 [DO178-B 1992] pour définir des règles de développement en concordance avec les exigences des autorités de certification.

b) *L'élimination des fautes* : l'élimination de faute se fait pendant les phases de validation ou lors des premières utilisations du système (simulations, essais en laboratoire, essais au sol, essais en vol).

c) *La prévision des fautes :* la prévision des fautes anticipe ces dernières pour ensuite pouvoir appliquer des moyens de l'élimination ou de la tolérance aux des fautes. Il s'agit d'une estimation et évaluation de la présence des fautes et de leurs conséquences.

d) *La tolérance aux fautes,* qui consiste à fournir, principalement par redondance, un service conforme aux spécifications du système initial en dépit de fautes. Elle se présente comme des techniques supplémentaires aux actions permettant d'éviter les fautes, de les prévoir et de les éliminer. Le recours à ces méthodes dépend des étapes de conception et de développement de système.

4) Entraves : classification des pannes dans les systèmes de commandes de vol

La sûreté de fonctionnement est la science des *défaillances* ou des *pannes*. Une défaillance survient lorsque le service délivré par le système dévie de l'accomplissement de la fonction du système. Les défaillances sont des *entraves* à la sûreté de fonctionnement.

Mais, les défaillances ont toujours une cause, ayant elle-même une cause, etc. Une analyse plus fine de cette relation de causalité a conduit à définir trois notions : les *fautes*, qui peuvent provoquer des *erreurs*, qui a leur tour peuvent provoquer des *défaillances,* qui sont l'effet perceptible (au niveau du service rendu) des fautes (comme le symptôme chez un malade, dont il faut déterminer la cause pour pouvoir le guérir). Et ces définitions sont récursives : la défaillance d'un système, si elle est exploitée par un utilisateur, peut conduire cet utilisateur à « défaillir » à son tour, et elle devient alors une faute chez l'utilisateur. Avec ces définitions, le terme *entrave* ne couvre pas seulement les défaillances, mais toute la chaîne de causalité jusqu'aux fautes. Des classifications très précises ont été dressées pour inventorier toutes les entraves, simples et leurs combinaisons. Toutes ces définitions ont le mérite d'avoir disséqué au plus fin toute la chaîne de causalité, et de lui avoir donné une terminologie générique.

Mais il faut bien avoir à l'esprit que selon les domaines d'application (automobile, industrie, logiciel, etc.) la terminologie usitée peut varier, même si les phénomènes restent les mêmes. Ainsi, on parle souvent d'une panne matérielle, mais jamais d'une panne logicielle.

Et ainsi, dans le domaine des systèmes de CDVE, bien souvent, on parle de panne pour désigner l'état du système résultant de défaillances.

Quoi qu'il en soit, nous ne rappellerons pas ici l'ensemble de ces définitions et classifications, mais seulement ce qu'il est nécessaire dans connaître pour la suite des travaux présentés.

a) *Les modes de défaillance* traduisent le fait qu'un système ne défaille pas toujours de la même façon, et peuvent être caractérisés selon trois points de vue :
   - le domaine de défaillance,
   - la perception des défaillances par les utilisateurs du système,
   - les conséquences des défaillances sur l'environnement du système.

b) Concernant le *domaine de défaillance,* il est divisé en deux catégories :
   - les défaillances *en valeur* : la valeur du service délivré ne permet plus l'accomplissement de la fonction du système (la valeur est en dehors de la plage de valeur considérée comme « valide »),
   - les défaillances *temporelles* : les conditions temporelles de délivrance du service ne permettent pas l'accomplissement de la fonction (ex. : trop tôt ou trop tard).

c) Concernant *la perception* des défaillances, on distingue, lorsqu'un système a plusieurs utilisateurs, deux types de défaillances :
   - les défaillances *cohérentes* : tous les utilisateurs ont la même perception de la défaillance,
   - les défaillances *incohérentes* (souvent qualifiées de byzantines) [Lamport *et al.* 1982] : tous les utilisateurs n'ont pas la même perception de la défaillance, ce qui accroît la difficulté de son traitement.

Pour les commandes de vol, il y a essentiellement deux critères pour distinguer le type d'une faute : ses conséquences et sa persistance temporelle.

a) En termes de conséquences ou de répercussion, on distingue :
   - o les pannes dites *actives* ou *franches*, si les répercussions sont immédiates,
   - o les pannes dites *passives* ou *cachées* dans le cas contraire, c'est-à-dire s'il est nécessaire de faire un contrôle régulier pour vérifier l'état du système.

b) Pour la persistance temporelle, on parle de faute *permanente* pour celle qui persiste dans le temps et de faute *transitoire* dans le cas contraire.

5) Techniques de tolérance aux fautes pour les systèmes de commandes de vol

Les techniques de tolérance aux fautes occupent une place privilégiée dans les systèmes de CDVE. D'où la description maintenant, dans ce cadre, des bases des principales techniques, puisque nous nous focaliserons principalement sur ce type moyen dans la suite des travaux.

a) La *redondance* au sens large

Rappelons d'abord qu'il ne peut pas y avoir de tolérance sans, à la base, une forme ou une autre de « redondance », que l'on peut définir, en première approche, comme consistant à disposer de plusieurs « exemplaires » d'un même élément (équipement ou processus, ou tout autre élément matériel ou logiciel) [Schor *et al.* 1989]. Et on distingue deux formes primaires :

- la redondance *symétrique* : les éléments sont semblables en tout point,
- la redondance *asymétrique* ou *différentielle* : les éléments sont diversifiés, différents.

b) La *diversification*

La diversification consiste à utiliser des éléments matériels et/ou logiciels qui ne sont pas issus d'un même processus de développement. Par exemple, la diversification logicielle [Mitra & McCluskey 2001, Deswarte *et al.* 1998] consiste à concevoir, à partir d'une seule spécification fonctionnelle, plusieurs variantes d'une même fonction dans le but de rendre indépendantes les fautes de conception logicielles.

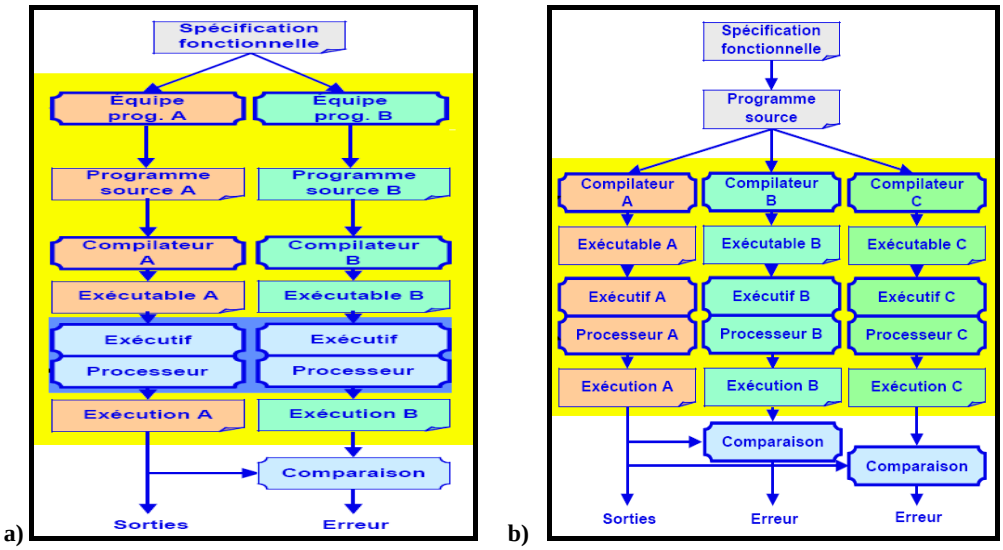


Figure I.2. - Principe de développement d'un composant logiciel diversifié : a) N-autotestable ; b) N-Versions

Citons les deux principes les plus courants :

- la programmation *N-autotestable* [Laprie *et al.* 1990] où au moins deux logiciels sont exécutés en parallèle. Par exemple, un composant 2-autotestable associe, soit deux variantes et un algorithme de comparaison (cf. Figure I.2.a), soit une variante et un test d'acceptation.
- la programmation *en N-versions* [Avizienis 1985] où les variantes de mise en œuvre d'une même fonction sont appelées des *versions* (la Figure I.2.b illustre le cas 3-versions) et un décideur effectue un vote sur les résultats de toutes les versions.

La diversification logicielle est utilisée dans le cadre du logiciel de commandes de vol pour la tolérance des fautes de conception ; le terme « conception » est à prendre au sens large et recouvre en fait les différentes étapes de la génération d'un logiciel.

c) La *ségrégation physique*

La ségrégation physique est une technique qui s'applique que la redondance soit symétrique ou non. Elle consiste à faire en sorte que tous les équipements ne soient pas localisés au même endroit et ne soient pas alimentés par une seule et même source, en termes d'informations et en termes d'énergie, ceci pour qu'une panne simple ne mette pas en danger la sécurité (évitement des fautes de mode commun) du système.

Ainsi, dans un avion, tous les équipements ne sont ni alimentés par une seule ressource électrique ou hydraulique, ni placés dans la même zone géographique. Les calculateurs sont installés dans trois zones différentes en soute pour que, suite à une destruction d'une partie de l'avion, tous les calculateurs et tous les actionneurs ne soient pas perdus. Les circuits électriques et hydrauliques (bien sûr redondés) circulent sur des chemins physiques différents, tous comme les bus de communication [Andrade & Tenning 1992].

d) La gestion de la redondance

La gestion de la redondance est indispensable dans les systèmes critiques pour garantir l'intégrité de l'architecture globale. Classiquement, on distingue deux grandes classes de gestion : le *masquage* et la *détection/recouvrement.*

Le *masquage* a pour effet de rendre le système insensible aux fautes (dans une limite donnée en fonction de ses exigences). Un exemple largement utilisé est le vote majoritaire qui extrait une donnée correcte d'un ensemble de données redondantes comportant une minorité de résultats erronés. On parle de « masquage » car on ne détecte même pas s'il y a des défaillances. Le vote est effectué systématiquement dans tous les cas. On cite aussi parfois dans cette classe, les codes correcteurs d'erreur dont l'effet est de fournir un message juste à partir d'un message perturbé ou erroné. Cependant, cette technique est aussi parfois dans la catégorie ci-dessous.

La *détection/recouvrement* (ou *remplacement*) consiste à détecter des défaillances et de reprendre les traitements avec des parties correctes. C'est le cas des codes détecteurs d'erreur qui permettent de rejeter les messages perturbés ou erronés, et de demander et attendre des retransmissions, ou encore de réutiliser les dernières données jugées valides. Un autre exemple est celui de la comparaison qui permet de constater des erreurs en cas de non-cohérence, de lever des alarmes et d'empêcher l'utilisation des données erronées, puis de faire reprendre le traitement par d'autres unités initialement actives ou en attente (stand-by).

Dans les systèmes de commandes de vol, la redondance est utilisée pour tolérer principalement les pannes matérielles. Souvent ces deux approches sont combinées dans des stratégies dites hybrides pour les systèmes de commandes de vol Airbus : on masque et on détecte en même temps.

Après cette présentation générale des aspects de sûreté de fonctionnement dans les systèmes de CDVE, les quatre sous-sections suivantes vont décrire plus précisément les différentes catégories d'exigences auxquelles sont soumis ces systèmes.

I.1.2.2 Exigences règlementaires

Ces exigences sont directement et principalement en rapport avec la sûreté de fonctionnement. Il s'agit de montrer que les systèmes de CDVE sont robustes à toute panne (ou combinaison de pannes) critique. Ces systèmes ont classiquement deux types d'objectifs de sûreté de fonctionnement :

- ***l'intégrité*** : les calculateurs ne doivent pas envoyer de sorties erronées vers les actionneurs,
- ***la disponibilité*** : le système doit présenter un haut niveau de disponibilité.

Par exemple, parmi les exigences établies lors de l'analyse de sécurité, la probabilité d'une condition de panne « catastrophique » par heure de vol doit être extrêmement improbable. Satisfaire ces exigences, induit la mise en jeu d'un certain nombre de techniques de sûreté de fonctionnement, dont les principales actuellement ont été développées au paragraphe I.1.2.1.

La tenue des exigences règlementaires est impérativement à analyser ou évaluer. Ces analyses de sûreté de fonctionnement seront largement détaillées dans le chapitre 4. On dit aussi « ***analyses de sûreté*** », ou « ***analyses de sécurité*** », ou « ***SSA (Safety System Assesment)*** », ces termes ne sont pas totalement en adéquation avec les définitions académiques de référence [Blanc *et al.* 2009]. Mais ce sont les termes consacrés dans le domaine qui a été le cadre de ma thèse. C'est pourquoi, par la suite, nous utiliserons indifféremment ces termes, en particulier « sécurité ».

I.1.2.3 Exigences supplémentaires : expérience en service et précautions forfaitaires

Les avionneurs prennent désormais classiquement en compte des exigences supplémentaires, pour produire des avions avec un niveau de sécurité encore plus élevé que le niveau requis par les autorités de certification. Le but est de renforcer la sécurité et d'avoir plus de marge pour la mise à jour du système suite à la détection des anomalies en service. Ces exigences sont définies, soit à partir de l'expérience en service, soit à titre forfaitaire. Citons les cas suivants :

- Le cas d'une panne qui ne peut pas faire l'objet d'une analyse statistique, ou qui est extrêmement improbable, mais que l'on veut traiter forfaitairement. Par exemple, perdre tous les calculateurs de commandes de vol de même technologie est extrêmement improbable, et pourtant, on veut couvrir ce cas.
- La qualité des lots de composants : les choix technologiques pour les composants, les règles de conception des équipements, etc. sont d'autres facteurs importants de maîtrise de la fiabilité. Or, malgré les précautions prises, il peut arriver qu'une production baisse de qualité et que cela ne soit détecté qu'après la mise en service de plusieurs composants moins fiables.

Grâce à la prise en compte de ces exigences supplémentaires, les systèmes de CDVE offrent des marges suffisantes permettant d'éviter, par exemple, qu'à cause d'un mauvais lot de composants, une flotte d'avions ne soit momentanément interdite de vol en attendant l'application des mesures correctives.

En conclusion, pour ces exigences, une des difficultés majeures consiste à les formaliser. Et leur prise en compte va dans le sens d'augmenter encore plus les redondances.

I.1.2.4 Exigences économiques

Les systèmes avioniques sont fortement contraints par les exigences économiques, et dans ce cadre, la notion de coût est très importante. Aujourd'hui, le système avionique correspond à plus de 1/3 du coût total d'un appareil civil, et ce chiffre ne cessera de croître. Pour le futur, les systèmes de CDVE sont une des cibles prioritaires dans la recherche de la réduction des coûts, notamment en termes de développement et de maintenance.

Bien que « tout ait un coût », il y a des aspects sur lesquels il plus facile de travailler que sur d'autres, comme typiquement la « sécurité » qui ne peut pas être réduite.

a) *Fiabilité opérationnelle*

Les attentes légitimes des passagers, l'encombrement du trafic aérien, la gestion des opérations sont des éléments qui conduisent les compagnies aériennes à avoir des exigences supplémentaires en termes de fiabilité. Typiquement, ces exigences sont de l'ordre de « *99,5% de vols ayant moins de 15 minutes de retard pour des raisons liées à l'avion* » [Wu 2005]. Donc, afin de permettre aux compagnies aériennes d'organiser plus facilement la gestion de leur flotte, il convient d'exiger que le système de CDVE reste encore utilisable avec le niveau de sécurité attendu, même si un équipement en panne ne pourra être réparé qu'après plusieurs jours (retour de l'avion au site de maintenance).

Là encore, sans cette exigence, un avion hors de son site de maintenance, avec un équipement en panne, pourrait être interdit de vol dans l'attente de son équipement de rechange. Et là encore, la prise en compte de cette exigence va dans le sens d'augmenter les redondances.

b) *Évolutivité et Modularité*

La durée de vie des avions peut être considérée comme un point vital lors de la conception du système avionique. En effet, actuellement, les avionneurs exigent une durée de vie d'au moins vingt ans pour tout aéronef civil ou militaire (c'est l'âge de l'Airbus A300 ou du Boeing 747, tous deux encore en production). Ceci signifie que la configuration de ces aéronefs peut changer et être continuellement améliorée au cours du temps. Sur une telle durée, les évolutions du trafic aérien, des services aux passagers, des possibilités offertes en matière de navigation (etc.) génèrent le besoin de mettre en place de nouvelles fonctionnalités insoupçonnées lors de sa conception initiale. L'architecture avionique doit donc s'accommoder de cette nécessité. Ainsi, toute architecture proposée doit également supporter l'addition, le déplacement et l'entretien de tous les composants. De plus, il doit être modulaire pour permettre une évolution facile.

c) *Masse et volume*

Le volume et la masse sont évidemment ennemis de base d'un avion. Le volume de chaque équipement a un impact sur le volume global de l'avion. En général, la masse des câblages dépasse souvent largement celle des calculateurs et la marge d'optimisation de masse est plus importante sur les architectures des réseaux de communication. Un autre paramètre, plus insidieux, est la consommation électrique. Elle induit un premier effet sur le dimensionnement du système de génération électrique, et un second sur le système de refroidissement.

I.1.2.5 Exigences systèmes embarqués et temps-réel

Le logiciel des systèmes de CDVE est généralement de grande taille, mais de complexité moyenne, et ils sont développés pour des traitements cycliques dont la période d'échantillonnage de base est de l'ordre de 10 ms.

a) Comportement temps réel et comportement déterministe

Les systèmes de CDVE pour avion civil sont des systèmes à dynamique lente : la durée d'une évolution significative de déplacement de gouvernes est supérieure à la durée du cycle de base de commande de ces gouvernes. Néanmoins, ils obéissent à des contraintes de temps-réel, et se doivent absolument d'être déterministes pour assurer correctement le contrôle de l'avion (et être certifiés). Pour ces systèmes temps-réel, les messages urgents (ex. : en cas de rafale) doivent être transmis à temps pour garantir leurs échéances et ceci même en présence de messages non urgents. Ces systèmes doivent aussi se comporter d'une manière prévisible de telle façon que le délai minimal et le délai maximal subis puissent être calculés, ceci pour tout type de trafic. De plus, ces délais doivent respecter les contraintes temporelles du type de trafic. Ainsi, le système doit pouvoir délivrer une information correcte en un temps fini et connu.

b) Débit et puissance de calcul

Les systèmes avioniques embarqués se basent sur des fonctions de plus en plus nombreuses et complexes. Cela induit un accroissement des besoins en capacité de calcul et de communications. En effet, comme déjà dit, la complexité des systèmes avioniques double tous les 5 ans, et la durée de vie d'un avion est de 20 ans en moyenne. Il faut donc prévoir assez de débit pour les composants de communication et assez de puissance de calcul pour les composants de calcul pour être à même de supporter toute évolution pendant de nombreuses années.

c) Résistance

Les composants avioniques doivent fonctionner de manière nominale dans des conditions difficiles. Certaines des technologies utilisées sont particulièrement adaptées aux applications embarquées soumises à des contraintes environnementales sévères comme la très haute température (jusqu'à 200°C), les chocs thermiques et les forts niveaux de vibration. Il faut ainsi utiliser des composants très résistants physiquement, particulièrement au niveau des connecteurs pour les composants de communication et des mémoires pour les composants de calcul. Les câbles de communication peuvent être à proximité de câbles électriques de puissance et doivent donc être très résistants vis-à-vis des interférences possibles.

I.1.3 Conclusion

Dans cette première section, nous avons d'abord présenté le rôle des systèmes des CDVE et de ses divers constituants de base actuels (calculateurs, actionneurs, capteurs et réseau de communication). Ensuite, ont été présenté les nombreuses exigences à satisfaire, avec bien entendu les exigences fonctionnelles, mais surtout les exigences non fonctionnelles (règlementaires, supplémentaires et économiques), qui sont au cœur de nos préoccupations.

Dans la prochaine section, nous allons développer les réponses concrètes apportées par trois principaux avionneurs pour satisfaire ces exigences, sachant que ces réponses sont très largement significatives. En effet, d'après les sources DGAC, les statistiques montrent que l'avion est un moyen de transport très sûr (en comparaison avec les autres moyens de transports publics ou privés) : 1 mort pour 2 milliards de kilomètres parcourus pour un passager, soit 50000 fois le tour de la Terre …). Ce niveau de sécurité est le résultat d'études rigoureuses menées lors de la conception de tout nouvel appareil, qui doit nécessairement obtenir une certification des autorités pour pouvoir être livré aux compagnies et mis en exploitation.

I.2 SOLUTIONS ARCHITECTURALES TOLERANTES AUX FAUTES : ETAT DES LIEUX DES ARCHITECTURES ACTUELLES

Airbus et Boeing sont les deux plus grands constructeurs mondiaux d'avions civils, avec la production de centaines d'unités par an et la maintenance de milliers d'unités. L'étude des solutions architecturales qu'ils ont mises en place pour remplir les exigences décrites dans la section précédente est donc totalement incontournable, et très largement illustrative. Pour être encore plus large, nous introduisons aussi brièvement un grand constructeur d'avions d'affaires, Dassault Aviation, avec des solutions hybrides entres celles d'Airbus et de Boeing.

I.2.1 L'architecture du système de CDVE Airbus

I.2.1.1 Les calculateurs : architecture et fonctionnement

Chez Airbus, le système CDVE utilise deux types de calculateurs numériques *autotestables* : 3 primaires (PRIM, ou P) et 3 secondaires (SEC, ou S) dans une configuration maître-esclave. Un calculateur SEC agit comme une redondance asymétrique (cf. I.1.2.15) d'un calculateur PRIM dans la mesure où il utilise un matériel différent et des lois de pilotages simples (cf. I.1.1.3). Les calculateurs de même type (P ou S) ont des logiciels différents [Traverse *et al.* 2004]. La Figure I.3 présente le schéma de principe d'un calculateur de commandes de vol.

Le processus de développement de l'applicatif de ces calculateurs est basé sur la programmation *N-autotestable* avec *N=2*. Chaque calculateur est composé de deux unités (ou voies) matérielles de calcul quasiment identiques, mais ségréguées (les cartes sont dans des boitiers distincts, mais accolés). Le mode de fonctionnement général est le suivant :

- Une voie *COM* (*commande* ou command) élabore les consignes de commande des actionneurs et les émet vers, les actionneurs et vers sa voie MON qui la surveille, mais aussi vers les autres calculateurs.
- Une voie *MON* (*moniteur* ou monitor) élabore les mêmes consignes que COM mais ne fait que surveiller COM, et lève des alarmes vers d'autres systèmes de l'avion.

A la sortie, les valeurs obtenues par chaque unité sont comparées. S'il y a un écart qui dépasse le seuil de tolérance autorisé par les concepteurs système, le calculateur perd la fonction système concernée et passe la main à un autre calculateur pour traiter la fonction perdue.

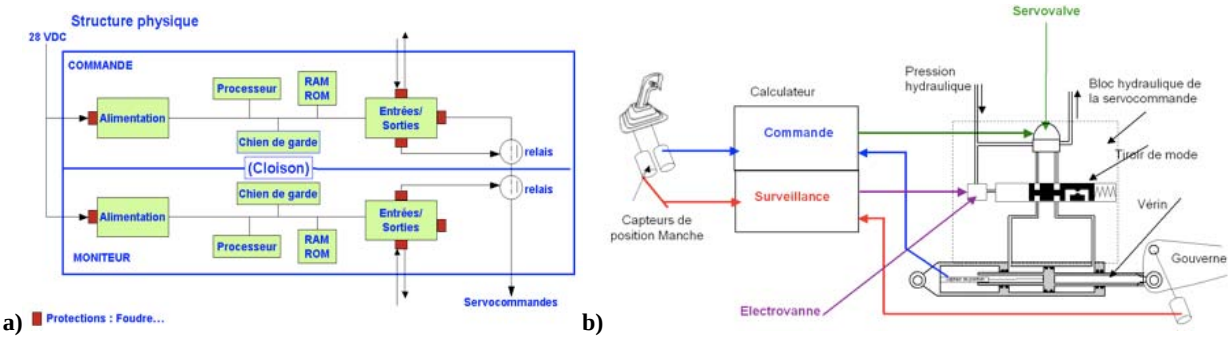


Figure I.3. – a) Schéma simplifié du matériel d'un calculateur CDVE AIRBUS
b) Schéma fonctionnel simplifié d'un calculateur de CDVE AIRBUS

I.2.1.2 Les communications

Les communications inter-voies d'un même calculateur (COM-MON) et les communications inter-calculateurs s'effectuent essentiellement via des bus ARINC 429. Dans cette architecture, tous les calculateurs fonctionnent en parallèle indépendamment les uns des autres. C'est-à-dire que chaque calculateur est indépendant des autres et possède sa propre horloge temps-réel et sa propre alimentation. La communication inter-calculateur n'est pas synchronisée et se fait par échantillonnage périodique. Le système global est un système de type *GALS* : Globalement Asynchrone (inter-calculateurs), Localement Synchrone (intra-calculateur) [Benveniste & Berry 1991, Hemani *et al.* 1999].

I.2.1.3 Les actionneurs

Rappelons que la commande des gouvernes correspond à la transmission de l'ordre d'asservissement (ou de commande) du calculateur vers les gouvernes. D'une part, la commande de chacune des gouvernes est assurée, par un ou plusieurs actionneurs. D'autre part, chaque actionneur est pilotable par un ou plusieurs calculateurs : à un instant donné, leur contrôle est assuré par un seul calculateur (en mode actif), mais en cas de défaillance de ce dernier, le système est reconfiguré pour permettre aux calculateurs en secours de prendre le relais. C'est le principe de fonctionnement utilisé pour les gouvernes de type ailerons, profondeur et direction. Cependant, pour des gouvernes telles que les spoilers, la commande est dédiée à un actionneur unique. En cas de mise en défaut de ce dernier, la gouverne correspondante n'est plus asservie.

La Figure I.4 décrit la distribution des différents calculateurs PRIM (P1, P2, P2) et SEC (S1, S2) sur les différentes gouvernes. Elle montre aussi qu'il existe au total trois circuits hydrauliques indépendants (noté en bleu, vert et jaune), physiquement ségrégés, et que tous les actionneurs ne dépendent pas du même circuit hydraulique.

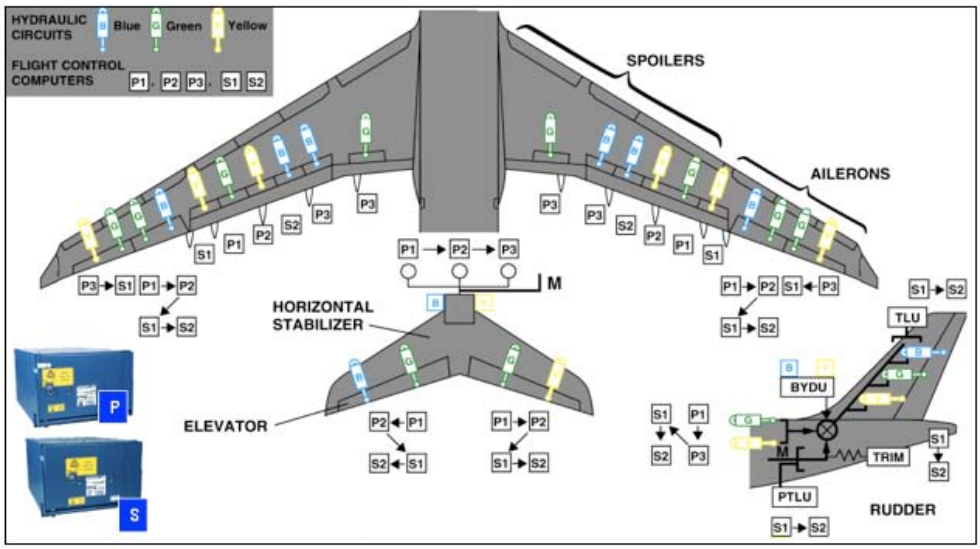


Figure I.4. - Distribution des calculateurs PRIM et SEC sur les gouvernes

I.2.1.4 Logique de priorité de reconfiguration en cas de défaillance

En fonction de sa criticité, une surface mobile est contrôlable à la fois par les calculateurs primaires et secondaires, et est actionnable par un, deux ou trois des systèmes hydrauliques (gouverne de direction et de profondeur) avec des logiques de priorité et de reconfiguration définis à l'avance. Un exemple de priorité sur la Figure I.4 est celle de la gouverne de

profondeur gauche (left elevator). Cette gouverne est commandée par les calculateurs P1, avec P2 en « secours », et si P2 est en panne, c'est alors le calculateur S1 qui prend la relève et la chaîne se poursuit avec le calculateur S2. Cette manière de faire permet au système d'avoir un haut niveau de disponibilité. De plus, cette même gouverne est actionnée par deux sources de puissance qui sont les deux circuits hydrauliques bleu et vert. Un seul circuit hydraulique active l'actionneur alors que l'autre reste passif et en fonctionnement amorti.

I.2.2 L'architecture du système de CDVE Boeing

Chez Boeing, un des fondements de la tolérance aux fautes est l'utilisation de la solution de type TMR *(Triple Modular Redundancy)*, aussi appelée *triplex*, qui consiste à avoir 3 exemplaires pour chaque élément. Cette technique est appliquée pour tout ce qui est matériel. On la retrouve donc au niveau des calculateurs, des circuits électriques et hydrauliques, et des bus de communication. Mais ce n'est pas la seule technique utilisée [Yeh 1998, Andrade & Tenning 1992].

I.2.2.1 Les calculateurs : architecture et fonctionnement

Le système CDVE utilise des calculateurs comme suit :

- trois calculateurs numériques centraux *PFC* (*Primary Flight Control*), aussi appelés « *chaîne* », pour le calcul des lois (on les désigne aussi par *left*, *center* et *right* PFC).
- quatre calculateurs analogiques *ACE* (*Actuator Control Electronics*) pour l'asservissement des actionneurs et la réalisation des conversions numériques-analogiques.

Les PFC sont identiques sur le plan matériel (tout comme les ACE), alors qu'au plan logiciel, de la diversification intervient au niveau des compilateurs pour la génération de l'applicatif pour les PFC, et elle est basée sur la programmation *N-versions avec N=3* (cf. I.1.2.15).

La Figure I.5 décrit succinctement la composition matérielle et le schéma fonctionnel d'un seul calculateur PFC. Un calculateur PFC est composé de 3 voies (lanes) de calcul redondantes. Les 3 voies sont *matériellement* (processeurs différents) et *fonctionnellement dissimilaires* aussi, puisque les voies sont organisées en une configuration fonctionnelle *command(voie1)-monitor(voie2)-standby(voie3)*. La voie *command* élabore les consignes de commande et les écrit sur un bus (ARINC 629), tandis que les deux autres voies surveillent son fonctionnement. Quand la voie *command* est déclarée en panne, elle se désengage, et une des voies supplémentaires est promue dans le rôle de voie *command*.

En considérant maintenant l'ensemble des 3 PFC, ceux-ci échangent, via un bus, leurs consignes (générées par les voies *command* de chacun) et sélectionnent la valeur médiane par un simple vote et déclarent la valeur sélectionnée comme la commande pour les ACE.

I.2.2.2 Les communications

Les calculateurs sont connectés à trois bus (du fait de l'application du TMR) de données commandes de vol (des bus ARINC 629) qui servent à échanger les informations entre les composants du système. Chaque PFC transmet sur un bus de donnée dédié et reçoit sur tous les deux autres bus. Les échanges se font d'une manière synchrone [Tripakis *et al.* 2008].

L'architecture Boeing est fondée sur le paradigme tout synchrone avec une architecture TTA (Time Trigger Architecture) à base de bus ARINC 659 [Carpenter et al. 1994, Natale & Stankovic 2000] à la différence de la solution GALS d'Airbus.

La Figure I.6 présente le schéma de principe simplifié de l'ensemble de l'architecture d'un calculateur commandes de vol d'un Boeing 777.

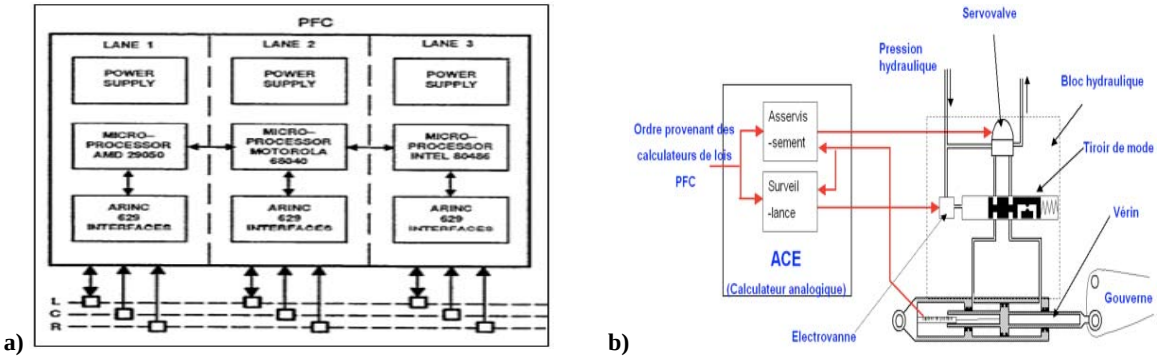


Figure I.5. - a) Schéma simplifié du matériel d'un calculateur PFC de CDVE BOEING b) Schéma fonctionnel simplifié d'un calculateur PFC de CDVE BOEING

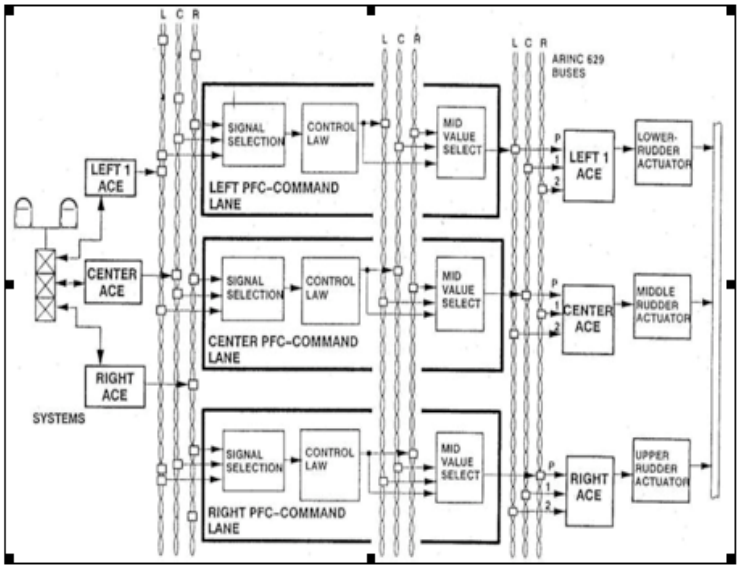


Figure I.6. - Architecture commandes de vol d'un Boeing 777

Pour ce qui est de la distribution des trois chaînes (de PFC) sur les gouvernes de l'avion, celle-ci se fait suivant une approche similaire à celle d'Airbus pour assurer une disponibilité du système global en cas de pannes. On n'en donne donc pas de représentation ici.

I.2.2.3 Les actionneurs

L'asservissement des actionneurs est réalisé par quatre calculateurs analogiques, les ACE. Chaque ACE est responsable d'un quart de l'ensemble des gouvernes, et chaque surface est contrôlée par un et un seul ACE. Les ACE jouent aussi le rôle d'intermédiaire entre les organes de pilotage et les actionneurs pour les conversions numérique/analogique et de secours en cas de panne des PFC.

Note : les Figure I.4 et Figure I.6 ne sont que des schémas de principe, et le nombre réel de calculateurs dépend d'autres facteurs (par exemple, l'efficacité de chaque gouverne).

I.2.3 Autre architecture : Falcon 7X

Pour compléter l'étude des solutions architecturales courantes pour les systèmes de CDVE, nous présentons brièvement celle d'un Falcon 7X (cf. Figure I.7), de Dassault Aviation, grand constructeur d'avions d'affaire, qui est le premier avion d'affaires à commandes de vol électriques. Schématiquement, l'architecture est organisée autour de six calculateurs centraux pour le calcul des lois et de quatre calculateurs pour l'asservissement des actionneurs :

- trois calculateurs primaires *MFCC* (*Main Flight Control Computer*) double chaîne pour le contrôle,
- trois calculateurs secondaires *SFCC* (*Secondary Flight Control Computer*) simple chaîne pour le contrôle,
- quatre calculateurs *ACMU* (*Actuator Control & Monitoring Units*) simple chaîne pour l'asservissement des actionneurs,
- et à cela s'ajoutent cinq calculateurs supplémentaires *FDC* (*Flight Data Concentrator*), utilisés pour sauvegarder les données de vol.

L'avion est contrôlé par les trois calculateurs primaires (MFCC) en mode normal, et par les trois calculateurs secondaires (SFCC) en cas de défaillance des primaires. Les calculateurs primaires sont basés sur du matériel différent de celui des secondaires, et l'ensemble des calculateurs est alimenté électriquement par quatre sources électriques indépendantes.

L'architecture Falcon peut être perçue comme étant « hybride » entre celle du Boeing B777, et celle d'AIRBUS. En effet, les ACMU assurent le même rôle que les ACE dans l'architecture Boeing décrite précédemment (chaque ACMU est responsable de l'asservissement d'un quart des actionneurs), alors que les MFCC utilisent un principe proche du COM/MON des calculateurs chez AIRBUS.

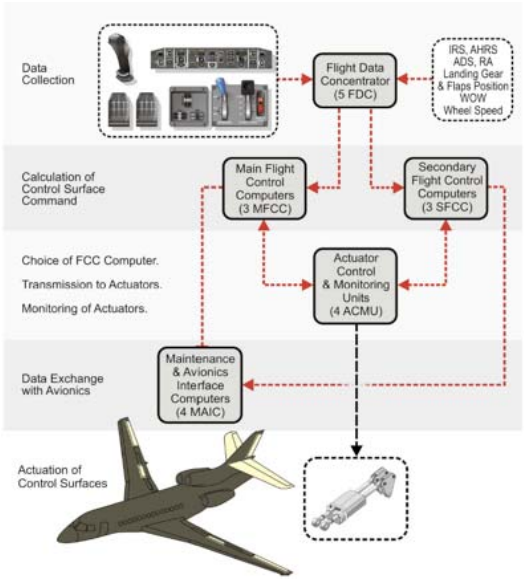


Figure I.7. - Système de commandes de vol d'un Falcon 7X

Enfin, un point commun aux trois types d'architectures présentées est l'existence d'un « secours » mécanique ou électrique (A380, A400M), grâce auquel, en cas de panne totale ou transitoire des calculateurs, le pilote peut contrôler la direction et la profondeur puisque la capacité de l'avion à voler en sécurité a été démontrée avec un contrôle limité au roulis et au tangage pour les avions Airbus, et sur une paire de spoilers et le plan horizontal pour les avions Boeing.

I.2.4 Synthèse

Dans cette section, nous avons présenté un tour d'horizon des différentes architectures tolérantes aux fautes pour le système de commandes de vol. La synthèse n'est certes pas complète, mais nous nous sommes limités aux avions civils des grands avionneurs.

Les architectures CDVE actuelles ont montré leurs mérites depuis l'A320, avec l'introduction du contrôle électrique en pleine autorité sur les axes de l'avion. L'analyse que nous avons menée montre que ce sont des architectures centralisées et tolérantes aux fautes. En effet, la conception et la réalisation d'un tel système sûr de fonctionnement passe par l'utilisation combinée de redondances matérielles et logicielles pour rendre indépendantes les fautes et pour minimiser la probabilité de défaillances de mode commun entre les unités redondantes.

Chaque avionneur utilise des techniques et propose des architectures différentes. Ainsi, la solution Boeing utilise un principe de vote majoritaire avec trois calculateurs triplex pour atteindre le niveau de tolérance requis pour les fautes. La solution Airbus utilise un principe de comparaison COM/MON et remplacement, avec six calculateurs duplex.

Une comparaison, en première approche, entre les deux solutions architecturales Airbus et Boeing, est résumée dans le Tableau I.1 ci-dessous. Bien entendu, la comparaison ne peut absolument pas se réduire à un simple aspect comptable du nombre de calculateurs : chaque solution présente des avantages non négligeables

	Airbus	**Boeing**
Type de redondance	COM/MON + remplacement architecture GALS (Globalement Asynchrone Localement Synchrone)	vote majoritaire architecture TTA (Time Triggered Architecture)
Logiciel fonctionnel	4 (2 PRIM + 2 SEC)	3 PFC
Nombre total d'équipements	12 unités numériques de 2 types 1 ultime secours	9 unités numériques de 3 types 4 unités analogiques (ASIC) 1 ultime secours

Tableau I.1 - *Comparaison architecturale*

Enfin, et surtout, notre analyse fait ressortir que, bien qu'ayant des différences notables, les deux solutions ont au moins un grand point commun : **un niveau de redondance très élevé.**

Ce niveau est justifié par le souci d'une démonstration de sécurité, guidée à la fois, par les réglementations de certification, et par la préoccupation des constructeurs de ne pas proposer un produit taillé au plus juste, et qui serait alors davantage susceptible de devenir temporairement interdit de vol, si les hypothèses initiales de fiabilité des composants devaient momentanément être revues à la baisse. Ces architectures et le niveau actuel de redondances sont tout simplement le fruit d'un grand savoir faire et d'une longue expérience des concepteurs de ces systèmes.

Malgré tout, l'introduction de la tolérance aux fautes dans les systèmes critiques dans le but d'augmenter la sécurité n'est pas sans coût. Et vu le nombre de redondances introduites dans l'architecture, ce coût est non négligeable. La problématique est alors de déterminer le niveau de redondance adéquat en fonction des besoins réels de l'application ou du système : il ne s'agit pas de faire de compromis entre coût et risque, mais plutôt d'optimiser l'utilisation des ressources redondantes et de profiter de toutes les possibilités offertes par les nouvelles technologies disponibles sur le marché. Durant les décennies passées, les technologies avioniques n'ont pas cessé d'évoluer et les architectures des systèmes de CDVE doivent s'aligner sur ces progrès.

I.3 TENDANCE FUTURE : LES ENJEUX NOUVEAUX LIES A LA CONCEPTION DES CDVE

Les conclusions de la section précédente conduisent naturellement à présenter maintenant les axes de progrès qui vont orienter les prochaines générations des systèmes de CDVE, ceci pour répondre à l'augmentation du nombre et de la complexité des fonctions embarquées, ainsi qu'à celle de la répartition de ces fonctions, le tout impliquant un accroissement important des données échangées et du nombre d'interconnexions nécessaires. Nous explorerons donc les notions de capteurs/actionneurs intelligents [Bayart *et al*. 2005], celles des COTS [Iyer *et al.* 1999, Murdock & Koenig 2001, Arlat *et al*. 2000] (avec l'IMA [Obermaisser 2008] et l'AFDX [Brajou & Ricco 2005]), pour finir avec un tour d'horizon des bus de terrains « avionables ».

I.3.1 Utilisation d'actionneurs et capteurs intelligents

Les capteurs et les actionneurs ont évolué de manière spectaculaire grâce à plusieurs facteurs techniques et économiques. Ces équipements ont bénéficié des innovations technologiques dans les domaines de la microélectronique et de l'informatique, qui ont permis notamment :

- d'augmenter significativement l'intelligence (puissance de traitement et de mémorisation) d'équipements jadis passifs,
- de réduire très fortement leur taille, consommation, et prix de revient unitaire.

La notion capteur intelligent (ou *smart sensor*) est apparue dans les années 80 motivée en grande partie par le développement des réseaux industriels. L'objectif à l'époque était de disposer de capteurs dotés d'une capacité de calcul leur permettant de prendre en compte certaines grandeurs pour générer un signal corrigé, et d'une interface de communication leur permettant de transmettre ce signal à un système centralisé via un réseau informatique.

Aujourd'hui, cette notion a énormément évolué. Pour être qualifié d'intelligent, un capteur ne doit plus se contenter de délivrer un signal électrique correct ou de communiquer dans un format numérique, il doit apporter une certaine valeur ajoutée aux données afin d'être un support, ou un acteur pour la prise de décision et le traitement réparti. En effet, d'après la norme IEEE 1451.2-1997, *« un capteur intelligent est un capteur qui fournit des fonctions allant au-delà de celles nécessaires pour générer une représentation correcte d'une quantité mesurée ou contrôlée. Ces fonctionnalités simplifient typiquement l'intégration du capteur dans une application appartenant à un environnement réseau »*.

D'un point de vue technique, un capteur intelligent intègre désormais des capacités de calcul interne et de mémorisation, et une interface de communication bidirectionnelle. De manière plus détaillée, un capteur intelligent est généralement constitué d'un (ou plusieurs) capteur(s), d'une alimentation, d'un organe de calcul interne (microcontrôleur, processeur de signaux ou microprocesseur), d'une mémoire locale et d'une interface de communication. La mémoire est utilisée pour stocker notamment des programmes et leurs paramètres de configuration, ou encore pour sauvegarder des données en cours de fonctionnement du système. L'organe de calcul permet d'exécuter les programmes qui sont stockés en mémoire. Enfin, l'interface de communication permet au capteur d'échanger des données sur un réseau.

En outre, les technologies, le niveau de miniaturisation et de production de masse proposés pour les actionneurs et les capteurs intelligents offrent la possibilité d'associer gains, intégration, fiabilité et optimisation des coûts.

Cependant, ces évolutions soulèvent encore des problèmes :

- elles ne sont pas encore totalement matures dans tous les domaines d'application, notamment au point de vue fiabilité,
- la répartition de l'intelligence ainsi rendue possible augmente la complexité de la gestion globale des informations (par rapport à une gestion centralisée).

Et justement, pour notre domaine de l'avionique, la maturité n'est pas encore toujours suffisante. Le fait d'envisager de mettre plus d'électronique au niveau des actionneurs, pour y intégrer une partie de l'applicatif (fonctions intelligentes), nous amène à nous poser la question de la fiabilité des électroniques dans les zones non pressurisées de l'avion, parce que pour de telles technologies, les MTBF prévisionnels fournis par les fournisseurs d'équipement ne sont pas toujours respectés. Le calcul théorique a montré qu'il y a un impact sur la fiabilité des électroniques déportées en zone non pressurisée, mais cet impact est limité ; il conduit au pire à un MTBF divisé par 2.

Et en pratique, les actionneurs actuels sur les avions Airbus à partir du programme A 350 et les Boeing à partir du B787 vont bénéficier de cette technologie. Et donc, toute nouvelle architecture des systèmes de CDVE devra prendre en compte ce type d'équipement, avec les nouvelles possibilités qu'ils offrent, mais aussi les risques associés.

I.3.2 Utilisation des COTS

Actuellement, il y a deux philosophies que les avionneurs essaient d'adopter pour minimiser les coûts de leurs systèmes avioniques. La première est basée sur les « *Open Systems* », c'est-à-dire, des systèmes ouverts aux utilisateurs pour qu'ils puissent y a apporter leurs modifications et adaptations [Bérard *et al*. 2003]. La seconde consiste à utiliser les technologies « grand public » aussi appelées *COTS*, qui à l'origine signifiait « Commercial Off The Shelf » mais doit s'entendre en fait comme « *Components Off The Shelf* » [Arlat *et al*. 2000].

En effet, l'utilisation de ces technologies, largement répandues dans certains domaines, permet de garantir une plus grande disponibilité des composants, augmente le nombre possible de fournisseurs et permet de réduire considérablement le coût de développement et de faciliter la maintenance et l'évolution du système. Cependant, les systèmes de CDVE (comme toutes les applications critiques) ne font pas encore massivement appel à ces technologies, car leur utilisation nécessite une analyse profonde pour s'assurer que soient garanties les exigences de sûreté, bien plus fortes que dans d'autres domaines. A cela s'ajoute le problème de la durée de vie ou de maintenance de tels composants, souvent plus courts que pour des composants plus « classiques ». Néanmoins, les COTS sont à l'origine de deux évolutions très notables dans les systèmes de CDVE :

- des modules « *IMA* » (*Integrated Modular Avionics*), qui sont des modules standardisés dans leurs composants essentiels, destinés à abriter les divers applicatifs, et à acquérir et échanger des données ;
- un réseau de communication à haut débit, l'*AFDX* (Avionics Full DupleX Ethernet).

La suite de cette section est consacrée aux principales orientations et raisons de l'utilisation de composants COTS dans les CDVE.

I.3.2.1 Avionique Modulaire Intégrée : une nouvelle architecture Avion

1) L'origine

Pour assurer leurs fonctions, les systèmes avioniques disposent de capteurs, de calculateurs, d'actionneurs, d'alimentation en énergie, etc. Chacun de ces systèmes doit donc, en soi, être capable de gérer, d'échanger et de traiter des données. Ceci avait historiquement été réalisé par une électronique dédiée à chacun de ces systèmes, voire à chacune des fonctions, qui avait été conçus indépendamment des autres systèmes, dans une architecture fédérative.

Mais l'augmentation continuelle des fonctionnalités demandées a conduit progressivement à une forte inflation du nombre de calculateurs embarqués dans un avion. Cela présente un nombre important d'inconvénients : une masse importante de câblage et calculateurs, une importante consommation d'énergie, un coût élevé (faibles séries par rapport à d'autres systèmes comme l'automobile), une gestion complexe de la diversité des composants électroniques, l'augmentation des types de métiers requis, etc.

2) Architecture IMA (Integrated Modular Avionics) : le principe

Le principe de l'avionique modulaire est de tendre à faire disparaître la notion d'équipement propriétaire pour lui substituer celle de composant standard inter-opérable (tant matériel que logiciel). Elle vise le développement de systèmes pour lesquels le choix des ressources matérielles et le placement des applications sont beaucoup plus flexibles que les systèmes avioniques récents. Le principe de l'architecture IMA [Morgan 1991] est de répartir dans tout le volume de l'appareil un ensemble de ressources (calcul, mémoire, communication), qui seront partagées par plusieurs applications, qui accompliront les différentes fonctions du système avionique. De manière pratique, l'architecture physique est décrite par la norme ARINC 651. Les ressources sont regroupées dans des modules génériques appelés *LRM* (*LineReplaceable Module*), qui sont à leur tour regroupés dans des étagères, la communication au sein de ces étagères étant réalisée avec des bus spéciaux, généralement de type ARINC 659. Les modules peuvent être de trois types :

- des modules *cœurs* qui sont ceux qui se chargent de l'exécution des applications,
- des modules *d'entrée/sortie* permettant la communication avec des éléments ne respectant pas l'architecture IMA,
- des modules *passerelles* servant à la communication entre étagères (l'architecture IMA n'impose pas de moyen de communication spécifique entre ses différents composants).

3) L'IMA chez AIRBUS

L'approche retenue par Airbus a consisté à développer 8 composants de base seulement (ex. : boîtier, alimentation, carte CPU, plusieurs cartes d'entrées-sorties), qui peuvent être assemblés de façon modulaire, pour couvrir les besoins de chaque système ou groupe de systèmes particuliers. La conception de chaque composant est calculée pour que chacun des modules assemblés ait un *MTBF* de 50 000 heures pour des raisons de fiabilité.

Dans le cas de l'A380, ces 8 composants de base permettent d'assembler les 30 modules nécessaires à chaque avion ; le nombre total élevé est dû aux nécessaires redondances. Le fournisseur de chacun des systèmes devra donc élaborer ses applicatifs de manière à les implanter dans ces modules. Et chacun de ces modules peut ainsi abriter plusieurs applications (freinage, gestion de carburant, …) provenant éventuellement de fournisseurs différents. Cette stratégie permet une meilleure utilisation des ressources. A partir des composants décrits plus hauts, Airbus s'est attaché à définir une architecture de réseau permettant une optimisation de la masse ainsi que des flux de données, tout en respectant

évidemment les contraintes de redondance et de ségrégation, indispensables à la sécurité. La réponse apportée par l'avionique modulaire intégrée a permis de faire face aux défis de l'A380 et de proposer un nouveau standard pour l'aéronautique, déjà adopté par l'A400M (avion européen de transport militaire) et qui doit être adapté pour tous les nouveaux programmes Airbus.

On peut en conclure que l'intégration des commandes de vol dans l'architecture IMA globale de l'avion est un nouveau défi, et cela constitue une exigence technique supplémentaire qui devra être traitée, et ne pourra donc pas être exclue dans le cadre de la thèse.

I.3.2.2 AFDX : une nouvelle technologie réseau Avion

En ce qui concerne les aspects de communication, ce sont sur des bus avioniques de conception assez ancienne que reposent les architectures avioniques civiles récentes. Parmi ceux-ci, on peut citer le bus ARINC 429 (communication unidirectionnelle, câble paire torsadée blindée, message sur 32 bits, adressage par label, bande passante 12 Kbit/s ou 100 Kbits/s), qui est un bus mono émetteur très répandu, fiable et d'une grande simplicité d'utilisation mais aux performances limitées (débit maximal de 100 kbits/s). Quant aux réseaux avioniques militaires, eux reposent essentiellement sur des bus avioniques multiplexés, notamment sur le standard MIL-STD 1553B. Ce dernier est un bus maître/esclave qui utilise un mécanisme de commande/réponse pour gérer le contrôle d'accès.

1) Les nouveaux besoins et dernières évolutions

Le nombre et la complexité des fonctions embarquées n'ont cessé de croître, et la répartition de ces fonctions sur différents équipements va elle aussi croître. Cela implique un très important *accroissement de la nature et surtout du volume de données échangées*, mais également *du nombre d'interconnexions nécessaires*, puisque les échanges ne suivent plus un simple schéma producteur/consommateur, mais ont lieu entre de nombreux intervenants.

Les réponses à ces évolutions se sont faites selon deux grandes tendances, qui ont fini par converger ces dernières années. La première tendance visait à répondre à l'augmentation du besoin d'interconnexion en remplaçant les bus avioniques mono-émetteurs classiquement utilisés (qui brident très clairement la croissance du système avionique) par des bus multi-émetteurs (ARINC 629 ou MIL-STD 1553B). Ainsi, par exemple, Boeing a choisi d'embarquer dans le cockpit du 777 des bus ARINC 629, bus multi-émetteurs autorisant de bien meilleures performances (2 Mbps et jusqu'à 120 utilisateurs). Il apparaît cependant que cette amélioration des bus avioniques implique des coûts de développement de composants spécifiques très importants. Et au final, ces différents facteurs font que la technologie des bus mono-émetteurs n'est plus considérée comme une solution satisfaisante.

La seconde tendance, pour les aéronefs civils qui sont au centre de nos préoccupations, a consisté à réutiliser, principalement pour des raisons économiques, du matériel développé pour des réseaux locaux classiques. Les standards *ARINC 636* (*Onboard Local Area Network*) et *ARINC 646* (*ELAN : Ethernet Local Area Network*) constituent de bons exemples de standards visant à modifier des technologies existantes afin de les rendre avionables.

Mais plus particulièrement, c'est l'Ethernet Commuté qui a suscité beaucoup d'intérêt de la part des scientifiques et des industriels en raison de ses multiples avantages. Récemment, le standard ARINC 664, qui est la normalisation du réseau *AFDX*, a été intégré avec succès dans des avions Airbus (A380 et A400M) pour remplacer les bus traditionnels ARINC 429.

2) AFDX (Avionics Full DupleX switched ethernet)

L'explication des termes est la suivante :

- Avionics : réseau adapté aux contraintes avioniques,
- Full DupleX : échanges bidirectionnels entre abonnés via des liens à 100Mbits/s,
- Switched : connexions entre abonnés assurés par des commutateurs (switchs),
- Ethernet : conformité aux normes Ethernet 802.3.

L'AFDX est une adaptation aux besoins de l'avionique, d'une technologie réseau Ethernet commuté, qui présente les avantages suivants :

- Il s'agit d'un standard IEEE (donc non-propriétaire) bien connu. Ceci implique notamment des coûts d'utilisation réduits, car les investissements peuvent être amortis sur de plus grands volumes.
- C'est une technologie mûre, donc avec moins de surprises lors de sa mise en pratique. Et la longue expérience industrielle de ce standard permet de réutiliser des outils de maintenance déjà développés pour d'autres domaines, et permet d'avoir une bonne confiance en la fiabilité du matériel et sur la facilité de sa maintenance.
- Le fait que ce soient les commutateurs qui réalisent les duplications des trames permet de diminuer le nombre de câbles embarqués, par rapport à des bus mono-émetteurs.
- Plusieurs débits sont possibles et ils peuvent aller de 10 Mbps à 1Gbps. Cette variété assure une grande flexibilité d'utilisation. Et cela permet de remplacer plusieurs bus avioniques de débits différents et hétérogènes par de l'Ethernet Commuté, pour tendre vers un réseau homogène avec une utilisation optimale de la bande passante.
- Ce remplacement éliminera de plus les interfaces multiples de communication de certains équipements et diminuera le nombre de liens d'interconnexion.

L'AFDX est un réseau déterministe et redondé. Le déterminisme de l'AFDX signifie que le temps de transfert des données est borné : dans cette architecture, chaque équipement est seulement relié à un seul commutateur, par le biais d'un câble Full Duplex. De cette manière, il ne peut plus y avoir de collision sur le support physique. L'interconnexion des éléments repose donc sur des commutateurs Ethernet ou des micro-commutateurs, reliés entre eux par la même technologie par le biais d'un câble Full Duplex.

La Figure I.8 montre une vue « synthétique » du réseau AFDX à bord de l'A380, dans le seul but d'en donner une idée. Une explication serait sans apport significatif.

L'AFDX est donc une technologie bien adaptée aux besoins de l'avionique mais, comme toute technologie, elle présente également un certain nombre d'inconvénients :

- l'ajout de nouveaux équipements (switchs et/ou micro-switchs) induit une augmentation de la masse, l'ajout d'alimentations et de ventilations, et une occupation des baies avioniques pour des équipements supplémentaires,
- de nouvelles pannes à gérer (celles des switchs), liées à ce nouveau système,
- une latence (temps de traversée réseau) plus importante que sur l'ARINC 429.

Cependant, dans l'état actuel, l'AFDX ne remplace pas encore la totalité des réseaux de communications, et il reste et restera encore un certain nombre d'autres réseaux, principalement des réseaux de terrain.

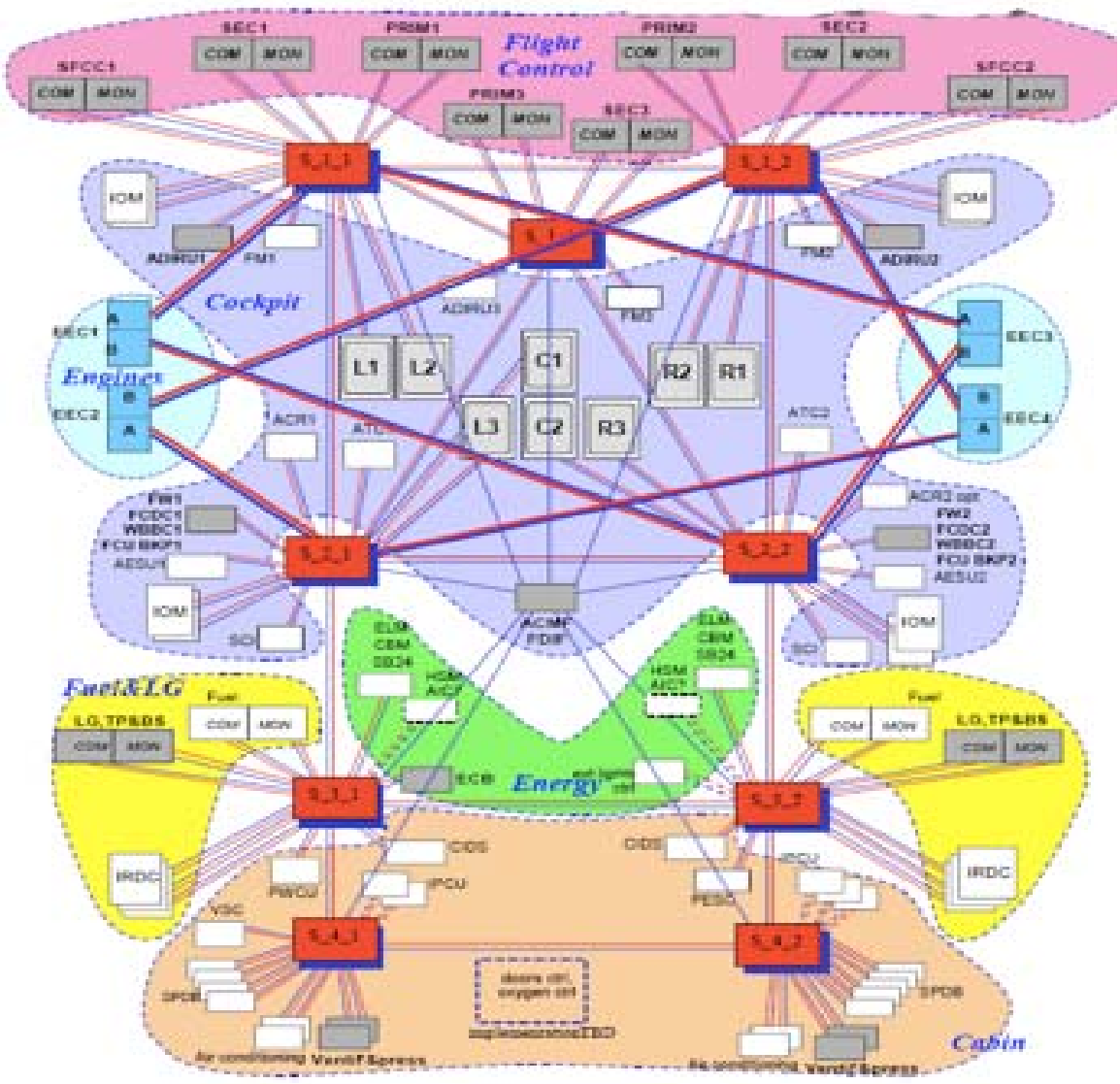


Figure I.8. - Réseau AFDX A380

I.3.3 Panorama des bus avioniques actuels

I.3.3.1 Étude comparative

Le choix du réseau de communication est un élément clef de toute architecture pour les systèmes de CDVE. L'AFDX ne peut pas encore être utilisé pour la totalité des communications, en particulier pour une partie de la communication entre les actionneurs et les calculateurs. Nous avons donc mené une étude comparative des différents bus et technologies déjà largement utilisés dans les systèmes industriels embarqués. Nous nous sommes limités à ceux qui offrent des perspectives intéressantes pour les systèmes de CDVE, pour la partie des communications contrôle-actionneurs, même s'il existe aussi une partie capteurs-pilotes. Toutefois, notre étude se base sur l'utilisation d'une même technologie de communication pour tous ces éléments. Les caractéristiques des bus que nous avons étudiés sont résumées dans le Tableau I.2.

Critères de comparaison	*CAN (Controller Area Network)*	*ARINC 429*	*MIL-STD-1553*	*TTP (Time-Triggered Protocol)*	*Flexray*
Description	Bus série avec accès CSMA/C Adressage par message. Domaine : automobile, industrie, …	Communication unidirectionnelle. Câble paire torsadée blindée. Message sur 32 bits. Adressage par label Domaine : avionique	Bus série, avec accès TDMA (Time-Division Multiple Access) Câble paire torsadée blindée. Domaine : avionique militaire, et spatial.	Bus avec accès TDMA (Time-Division Multiple Access)	Bus avec accès TDMA Domaine : automobile
Débit	1 Mbit/s max sur 30 m	100 Kbit/s	1 Mbit/s	5 Mbit/s	10 Mbit/s max sur 26 m.
Topologie	Bus Nombre d'abonnés limité par la bande passante	Point à point	Bus avec stub long	Etoile duplex	Etoile duplex
Isolation aux fautes	Pas d'isolation	Pas d'isolation	Isolation	Pas d'isolation	Pas d'isolation
Déterminisme	Bus multi-maître. Pas totalement déterministe.	Bus mono émetteur	Protocole nativement déterministe	Protocole nativement déterministe	Protocole nativement déterministe
Maturité dans le domaine avionique	Besoin de bâtir un sur-protocole à développer	Produit mature et pérennité assurée	Produit mature et pérennité assurée	Pas de couche physique dans la norme. Produit peu utilisé	Pas encore utilisé par son marché cible (automobile)
Installation	Nombreux câbles (point à point)	Nombreux câbles (point à point)	Peu de câbles. Installation des coupleurs à prévoir.	Installation des hubs. Câblage important entre hub et actionneurs (point à point)	Installation des hubs. Câblage important entre hub et actionneurs (point à point)
Intégrité	CRC 16 bit	bit de parité	bit de parité	CRC 16 bits	CRC 16 bits

Tableau I.2. - Comparatif des bus de terrain

L'analyse détaillée des différents candidats révèle que c'est le bus MIL-STD-1553 [MIL-STD-1553B 1978] qui répond le mieux aux besoins du système CDVE distribué sur avion carbone, pour les raisons suivantes.

- *Déterminisme* : les communications entre abonnés sont organisées par un maître du bus unique (le Bus Controller (BC)) sous la forme d'un protocole requête/réponse. Le BC communique cycliquement avec les autres abonnés (les Remote Terminal (RT)), soit pour transmettre des données, soit pour en recevoir. Les RT ne peuvent communiquer que lorsqu'on les interroge, il n'y a donc pas de risque de collision.
- *Topologie* : la couche physique se présente sous la forme d'une ligne principale sur laquelle les abonnés sont connectés par « *stub* » via des coupleurs de bus. Ces derniers réalisent deux fonctions : l'isolation et la non-propagation des défauts d'impédance du stub (court circuit, fils coupés, etc.). De plus, la topologie en « bus » (par opposition aux topologies en étoile) de la couche physique convient très bien aux commandes de vol : de nombreux abonnés répartis le long des ailes.
- *Robustesse aux agressions électromagnétiques* : la structure des avions du futur sera principalement composée de fibre de carbone (CFRP). Cette mutation a déjà commencé sur l'A350 et pose déjà de nouveaux défis : la cage de Faraday formée par la structure aluminium dans les programmes précédents et qui protégeait les câblages n'existe plus

vraiment en technologie CFRP. La couche physique 1553 est, de par la présence des transformateurs d'isolation, totalement flottante. Elle présente donc une très bonne qualité intrinsèque de rejet des agressions.

– *Coût* : le bus 1553 est souvent présenté comme une solution a priori onéreuse. Ceci est vrai pour les programmes militaires et spatiaux où les volumes de production sont faibles. Cela l'est beaucoup moins dans une perspective de production d'avions futurs mono-couloir : avec 350 avions produits par an pour environ 70 têtes de bus et 20 coupleurs par avion, on arrive à des volumes annuels significatifs permettant de baisser très sensiblement les coûts.

Les éléments décisifs ayant conduit à ne pas sélectionner une liaison optique, ou des bus prouvés pour l'automobile comme les réseaux CAN et TTP, sont les suivants :

– *Couche physique incompatible sur avion carbone* : communication non robuste aux agressions foudre (le différentiel est référencé masse), pas de ségrégation électrique des abonnés (propagation des courts-circuits sur le réseau).
– *Protocole standard non-déterministe* : « sur-protocole » maître-esclave spécifique à développer avec gestion des modes de panne réseau.
– *Bande passante faible* : nombre d'abonnés limité à 3 actionneurs.
– *Hub à développer* : pas de produit ni de spécification existante compatible CDVE.
– *Installation lourde et complexe* : câblage important entre hub et actionneurs (point à point voilure très pénalisant), hubs à installer dans le fuselage et l'empennage.

Concernant la fibre optique, cette technologie offre l'avantage d'un débit d'informations important et l'immunité aux perturbations EMI/foudre. Mais elle est moins adaptée à des liaisons avec des dérivations (liaisons par bus), qu'à des liaisons point à point.

De plus, cette technologie n'est pas encore éprouvée dans le domaine de l'avionique. En effet, la fiabilité des liaisons optiques dans les conditions d'environnement et de maintenance des avions commence tout juste à être observée au travers d'une application sur A340.500/600 : la liaison cockpit↔caméra en haut de la surface « rudder », utilisée pour transmettre au pilote une vue privilégiée de l'avion lors des manœuvres au sol (les images sont même proposées aux passagers sur leurs écrans personnels, pendant le vol). Les bus de terrain offrent, par rapport à la fibre optique, l'avantage de la souplesse (bi-directionnels, multi-abonnés). Ils reposent sur une technologie plus classique (liaisons électriques). Pour recourir aux bus de terrain ou à l'optique dans les systèmes de commandes de vol, il faudrait changer d'approche et être moins exigeant sur l'aspect « éprouvé » de la technologie utilisée.

I.3.3.2 Intégrité des communications – cas de la solution ALIC (Application Level Integrity Checking)

De manière générale, le réseau de communication pour l'avionique doit assurer un confinement des erreurs, c'est-à-dire qu'une erreur locale ne doit pas se propager et détériorer le comportement d'autres éléments. Ceci implique par exemple un filtrage des trames dans les éléments chargés sur le réseau avant leur utilisation par l'applicatif. Ce filtrage permet de supprimer les trames de longueur incorrecte, ou corrompues, ou dont la source ou la destination n'est pas identifiée.

Cette intégrité est assurée notamment grâce à l'utilisation de *FCS* (*Frame Check Sequence*) ou contrôle par *CRC*. Le principe de base, pour détecter une erreur de transmission, est d'ajouter des bits de contrôle aux bits qui constituent le champ de données. L'idée est donc de transmettre $n = k+p$ bits au lieu des k bits de données utiles, et d'utiliser les p bits

supplémentaires pour associer à l'émission un code de contrôle aux données utiles, et vérifier la validité de ce code à la réception.

Les bus avioniques ARINC 429, ARINC 629 et MIL-STD-1553, offrent une protection qui est très faible pour les systèmes avioniques parce qu'ils utilisent juste un seul bit de parité contrairement aux bus classiques de terrain (CAN, TTCAN, TTP/C) qui offrent la possibilité d'utiliser des codes CRC variant entre 16 et 24 bits.

Dans ce cadre, Airbus, pour une meilleure détection d'erreur sur les trames AFDX, a opté pour un code CRC sur 32 bits, qualifié de « *certificat applicatif* » [Youssef 2005, Cabaret *et al.* 2008]. Cette solution, appelée *ALIC*, a été développée essentiellement pour le réseau AFDX avec des fonctions de hachage complexes, mais peut s'appliquer à d'autres bus comme le 1553. La mise en œuvre est basée sur une approche cryptographique, et principalement sur un code d'authentification de message ou certificat, dont le principe est illustré sur la Figure I.9.

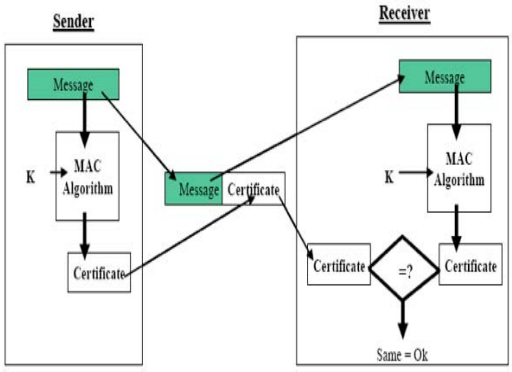


Figure I.9. - Principe d'implémentation ALIC

Ce principe utilise un algorithme *H* qui crée un certificat *C*, basé sur le message *m* et une clé secrète *K*. On note alors *C = H (m)*.

- A l'émission, un message *m* sera converti en un message *M* qui est composé de la concaténation du message *m* et du certificat *H (m)*.
- A la réception du message *M*, le même algorithme que celui utilisé par l'émetteur va extraire les données applicatives *m*, et connaissant la clé, va recalculer *H (m)* et le comparer à celui reçu. L'intégrité du message reçu sera ainsi vérifiée par le récepteur, et si le certificat calculé par le récepteur n'est pas le même que celui envoyé par l'émetteur, le message sera rejeté.

I.3.4 Orientations : vers des architectures distribuées

Après l'introduction des gouvernes servo-commandées, l'introduction des commandes de vol électriques a marqué une nouvelle étape fondamentale dans l'avancée technologique des avions. Et les commandes de vol vont continuer à évoluer en respectant les mêmes principes : recherche d'amélioration des performances tout en appliquant des solutions techniques validées.

Or aujourd'hui encore, les systèmes de commandes de vol électriques sont mis en œuvre par des architectures centralisées et basées sur des calculateurs tolérants aux fautes, alors que, de nos jours, la taille de logiciel calculateur, le nombre de fonctions et leur complexité ainsi que l'élémentaire raison de coût rendent cette architecture mal adaptée à l'utilisation dans les

avions commerciaux. Les commandes de vol se doivent donc d'évoluer pour proposer un produit moins consommateur de ressources matérielles pour un même service rendu.

Plus précisément, les architectures actuelles de commandes de vol ne bénéficient pas tout à fait de toutes les nouvelles technologies offertes pour les systèmes avioniques pour des raisons de sécurité et de maturité de technologies.

- Les fonctions commandes de vol sont mises en œuvre par des calculateurs dédiés aux commandes de vol qui ne font pas partie de l'architecture IMA actuelle proposée pour l'A380 ou l'A400M.
- Ces calculateurs sont connectés au réseau AFDX de l'avion, mais seulement pour les échanges d'informations non critiques.
- Les électroniques locales des actionneurs et des capteurs ont des capacités de traitement limitées et leurs intelligences sont limitées à une interface de communication qui leur permet de communiquer avec le système central (les calculateurs) via un réseau numérique, mais seulement depuis l'A350 ; il s'agit instruments intelligents de niveau 2 selon la classification présentée dans [Mkhida 2008]. Les possibilités de distribution de traitement offertes par la microélectronique permettent aux capteurs et actionneurs d'élargir leur fonction initiale (mesurer pour un capteur, agir pour un actionneur) jusqu'à la participation à certaines fonctions auparavant effectuées par les calculateurs centraux, cela correspond à des instruments intelligents de niveau 3 selon la classification présentée dans [Mkhida 2008].

Les architectures actuelles sont physiquement distribuées vu la localisation géographique de leurs composants physiques, mais, pour des raisons de sécurité, elles ont été centralisées au niveau logique. Ce sont des architectures de type maître-esclave. Les architectures actuelles sont robustes et offrent un grand degré de sécurité mais, pour le futur, les nouvelles technologies offrent aux industriels de nouvelles possibilités de conception d'architecture de commande pour améliorer les architectures actuelles. Et de plus, la réduction du coût du traitement pousse à déplacer le traitement le plus près possible de sources de données et des consommateurs, c'est-à-dire les capteurs et les actionneurs.

On peut en conclure qu'une plus grande intégration de la microélectronique permettra des architectures distribuées et que les capacités de traitement ne seront plus un facteur limitant, mais c'est plutôt la gestion de la communication inter-nœud et le nombre des informations échangées qui devra faire l'objet d'une plus grande attention.

Et justement, les communications numériques multiplexées offrent des perspectives intéressantes pour les commandes de vol (fiabilité, bande passante, immunité aux perturbations EMI/foudre). Leur grand avantage indirect est la réduction de la quantité de câblage et de la taille des connecteurs pour les composants du réseau.

En définitive donc, la prévision de composants intelligents qui seront de plus en plus disponibles et fiables, ainsi que l'utilisation de communications numériques, constituent deux des principaux fondements du raisonnement et de l'analyse présentés dans cette thèse. Et plus précisément, pour la suite des travaux, nous allons nous baser sur les hypothèses suivantes :

- standardisation de l'avionique centrale : utilisation de modules IMA pour les fonctions commandes de vol,
- utilisation des technologies numériques de communication AFDX et bus 1553 pour la communication inter-calculateurs et la communication calculateurs-actionneurs,
- utilisation d'actionneurs intelligents et distribution de l'intelligence système entre l'avionique centrale et les actionneurs,
- utilisation du CRC applicatif ALIC.

I.4 CONCLUSION

Ce chapitre a été consacré à la présentation des systèmes de commandes de vol électriques qui sont au cœur du travail réalisé. Un état de l'art des principales solutions architecturales actuelles a été dressé, et les exigences à satisfaire pour concevoir une architecture ont été identifiées : exigences règlementaires du domaine de l'aviation civile, exigences liées aux précautions forfaitaires et aux retours d'expériences des avions en service, exigences liées au caractère embarqué et temps-réel des systèmes visés, et enfin, exigences économiques. Pour finir, une introduction aux nouvelles technologies candidates à l'utilisation pour les nouvelles générations des systèmes de CDVE a été présentée.

L'analyse des architectures actuelles présentées dans ce chapitre a montré qu'elles étaient centralisées et faisaient appel à l'utilisation combinée d'importantes redondances matérielles et logicielles pour satisfaire toutes les exigences d'un tel système critique. Et même si chaque avionneur utilise des techniques et propose des architectures différentes, il n'en reste pas moins que les différentes architectures ont un important point commun : une apparente « sur-redondance ». Au final, elles sont coûteuses, rigides et nécessitent beaucoup de matériel. Cependant, ces architectures et le niveau actuel de redondances ne sont pas le fruit du hasard, mais celui d'un grand savoir faire et d'une longue expérience des concepteurs de ces systèmes pour répondre au besoin d'une démonstration de sécurité et de fiabilité, guidée à la fois, par les réglementations de certification, et par la préoccupation des constructeurs de ne pas répondre au plus juste à la demande des clients. Et toute évolution ne peut se faire qu'après une longue maturation de la technologie et par étapes successives.

Mais aujourd'hui, certaines avancées technologiques dans les domaines des réseaux numériques de communication et des capteurs et actionneurs intelligents sont arrivées à un niveau de maturation qui rend ces technologies (par ailleurs déjà bien implantées dans d'autres domaines) candidates à l'intégration dans la conception des systèmes de commandes de vol.

Et il devient donc possible de remettre en cause le niveau de redondance constaté jusqu'à ce jour, puisque ces avancées rendent pertinente la recherche d'architectures alternatives tout aussi robustes et plus avantageuses en ressources nécessaires. Ces nouvelles architectures doivent donc intégrer les nouvelles technologies tout en conservant le savoir-faire et l'expérience des concepteurs. Elles doivent également être justifiées par les avantages qu'elles apporteront en termes de sécurité, masse, coût, fiabilité, etc.

Chapitre II - UNE DEMARCHE INCREMENTALE DE CONCEPTION D'ARCHITECTURE

Le chapitre précédent a montré que l'aboutissement aux Commandes De Vol Électriques (CDVE) des avions civils actuels s'est fait par étapes, après une longue maturation des différentes technologies mises en place, et en se basant sur des retours d'expérience réguliers.

Mais aujourd'hui, une nouvelle étape s'ouvre vers des architectures distribuées avec l'emploi de réseaux numériques (à la place des liaisons analogiques) entre les calculateurs CDVE et les capteurs/actionneurs, ainsi qu'une distribution de l'intelligence vers ces derniers dont le niveau « d'intelligence » ne cesse de croître. Et de nouvelles architectures de systèmes de CDVE sont donc envisageables [Godo 2002], et avec pour défis de répondre, bien entendu, aux mêmes exigences de sécurité et de disponibilité qu'actuellement, mais aussi de répondre aux exigences croissantes des compagnies aériennes en termes de fiabilité opérationnelle.

Le système CDVE étant un système complexe, une des difficultés est d'avoir une vue globale de toute son architecture, ainsi que du comportement de ses composants et de leurs interactions.

Par conséquent, pour définir une nouvelle architecture, il est plus naturel de procéder de manière progressive en construisant et validant l'architecture par étapes : c'est l'objectif de la démarche de conception que l'on propose et qui est présentée dans ce chapitre. La démarche incrémentale définit des lignes directrices et des principes de conception afin de répondre aux défis, et gérer ainsi la complexité associée au système et à ses exigences.

La première section commence par apporter des précisions sur les hypothèses de travail, avant de définir les objectifs et le processus général de la démarche incrémentale proposée pour construire de nouvelles architectures. L'application de cette démarche est ensuite illustrée dans une deuxième section, pas sur la globalité du système, mais seulement sur les calculateurs. La troisième section détaille alors plus avant les principes et les calculs pour les analyses de sûreté. Enfin, la dernière section introduit deux nouveaux concepts d'architecture distribuée, qui résultent de l'application complète de la démarche incrémentale dans un périmètre restreint aux calculateurs, réseaux de communication et actionneurs. Ces deux architectures et leurs caractéristiques seront développées dans les chapitres suivants.

II.1 HYPOTHESES ET DEFINITION DE LA DEMARCHE INCREMENTALE

II.1.1 Objectifs et hypothèses de travail

La tendance pour de nouvelles architectures est de remplacer les architectures actuelles centralisées (coûteuses, rigides et gourmandes en matériel si les composants nécessaires sont peu fiables en regard des exigences de sécurité pour un système critique) en distribuant les traitements et les contrôles entre l'ensemble des composants intelligents du système. Pour les CDVE du futur [Fields & McKendree 2005, Blake & Liguori 2001], la répartition des fonctions sera optimisée selon un minimum de ressources matérielles, logicielles et de communication.

Pour ce faire, les hypothèses que nous avons prises s'appuient sur les considérations suivantes :

1) utilisation de calculateurs simplex indépendants les uns des autres, et supportant des opérations de reconfiguration logicielle dynamique, ce qui permettra une amélioration de la disponibilité des calculateurs de commandes de vol (actuellement, dans le principe COM/MON, le calculateur est perdu dès la perte de l'unité COM (respectivement MON), même si l'unité MON (respectivement COM) est encore fonctionnelle),
2) réduction du nombre de calculateurs et de leurs dépendances,
3) utilisation d'actionneurs intelligents,
4) utilisation de communication numérique,
5) utilisation de nouvelles répartitions des fonctions intelligentes sur les divers supports matériels : les électroniques des capteurs et actionneurs, les calculateurs, et des éventuels équipements intermédiaires du réseau,
6) exploitation des avantages des communications numériques multiplexées pour définir de nouveaux protocoles de communication entre calculateurs et actionneurs qui doivent être utilisés pour supporter ces architectures.

Mais pour proposer de nouvelles architectures intégrant de nouvelles technologies, et qui soient aussi robustes que les architectures actuelles avec un minimum de ressources matérielles et logicielles, il faut trouver des réponses à un certain nombre de questions :

1) Comment savoir au mieux à quel niveau de redondance il faut s'arrêter pour satisfaire toutes les exigences « commandes de vol » ?
2) Quels mécanismes de gestion de la redondance faut-il utiliser ?
3) Comment considérer les possibilités offertes par les différentes technologies envisageables à moyen et long termes pour les équipements CDVE ?
4) Quel découpage fonctionnel est le plus pertinent, avec quelle répartition des fonctions de commande et des surveillances sur les divers équipements CDVE ?
5) Quelle architecture de communication, avec quels moyens de préserver l'intégrité des données échangées, devront être utilisés pour supporter ces architectures dans la démonstration de sûreté de fonctionnement ?

II.1.2 Orientations et démarches possibles

Le développement des commandes de vol chez Airbus peut se représenter sous la forme d'un cycle en V (Figure II.1), qui diffère cependant légèrement du cycle en V classique. Il résulte de l'imbrication d'un cycle de développement classique et d'un cycle de sûreté de fonctionnement. On peut déjà constater que les activités d'évaluation liées à la sûreté de fonctionnement occupent une place prépondérante dans la validation des architectures CDVE.

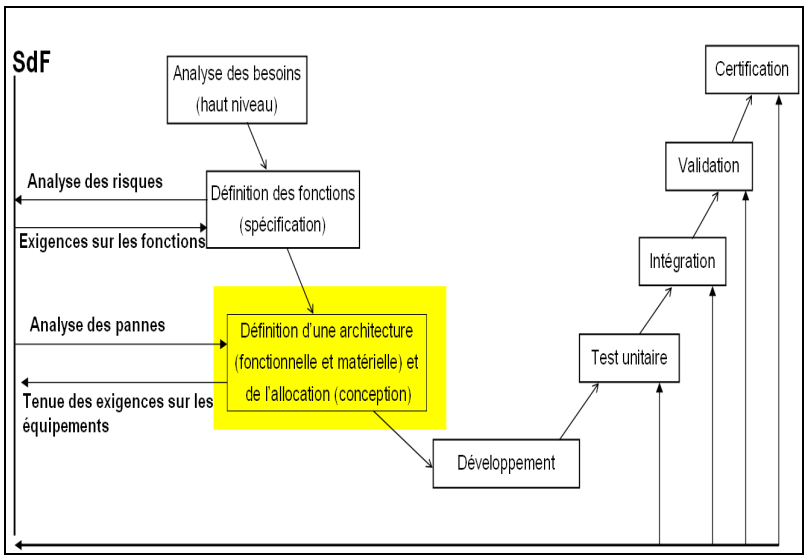


Figure II.1. - Cycle de développement Airbus

Dans le cadre de la thèse, on se placera en phase amont de conception et du point de vue d'un concepteur système commandes de vol dont l'action intervient dans la partie du cycle repérée en jaune sur la Figure II.1. Le concepteur, pour un système donné et à partir d' « *exigences de haut niveau* » (il s'agit d'une spécification textuelle), établit une architecture système (architecture matérielle et logicielle des calculateurs, actionneurs/capteurs avion et réseaux).

Le concepteur système propose un découpage fonctionnel (appelé architecture fonctionnelle) et une architecture support préliminaire (appelé architecture matérielle). L'ensemble forme une architecture préliminaire du système. Cette phase est primordiale, puisque c'est à partir de ses sorties que les développements des logiciels et des équipements seront lancés. Si l'architecture et l'allocation proposées ne peuvent pas répondre aux exigences, alors elles sont rejetées. Les concepteurs doivent alors rechercher de nouvelles solutions compatibles avec l'ensemble des exigences du système.

Une des phases primordiales et décisives du développement d'un système hautement critique comme celui des CDVE est donc la phase de définition des architectures fonctionnelles et matérielles, et celle de l'allocation associée respectant les exigences du système. Chez Airbus, cette phase est historiquement basée sur un savoir empirique, accumulé au fil des décennies, qui donne d'excellents résultats, mais qui n'est pas assez formalisé ni (ou mal) partagé et traçable. Cela ne permet pas de justifier pleinement la sur-redondance des architectures actuelles, et de prévoir au mieux les redondances nécessaires pour dimensionner des nouvelles architectures.

D'où un besoin de mise en place d'une phase d'optimisation ou de dimensionnement dans le processus de conception Airbus. Pour cela, il faut disposer de méthodologies de conception et d'algorithmes d'optimisation.

Au début de la thèse, nous avons voulu nous inspirer des approches utilisées pour le partitionnement Matériel/Logiciel et l'Adéquation Algorithme Architecture (AAA) du domaine des systèmes embarqués et l'utiliser pour l'optimisation des architectures CDVE.

1) La méthodologie de partitionnement Matériel/Logiciel décrite en détail dans [Diguet *et al.* 2006, Ben Chehida & Auguin 2002, Ben Chehida *et al.* 2002, Ben Chehida 2004] a pour objectif de trouver des solutions optimisant l'implémentation d'une architecture en jouant sur un compromis Logiciel/Matériel (calculateur / actionneur dans notre cas) et de partitionner l'application en respectant les contraintes et les caractéristiques du système de calcul utilisé.
2) La méthodologie AAA définie dans [Raulet 2006] et l'environnement SynDEx [Syndex] vise à développer des méthodes systématiques d'une meilleure mise en correspondance entre l'algorithme d'une part et l'architecture d'autre part.

Ce problème d'optimisation et d'allocation existe déjà depuis un certain nombre d'années, et il reste toujours ouvert. D'ailleurs, plusieurs travaux ont été menés, tous basés sur des principes différents pour résoudre cette problématique. Sans prétendre être exhaustif, on peut citer les algorithmes d'optimisation proposés dans [Kalavade & Lee 1994, Kalavade & Subrahmanyam 1997, Chatha & Vemuri 2001, Dave *et al.* 1997, Manikas & Cain 1996].

Au final, après une analyse approfondie des systèmes de CDVE, nous avons constaté que les pratiques et les méthodologies actuelles en termes d'optimisation et de partitionnement ne sont pas bien adaptées aux systèmes de commandes de vol. En effet, ces méthodes demandent un important travail de modélisation et de formalisation d'exigences, sachant que de plus les exigences à considérer sont hétérogènes, et certaines orientations et choix industriels sont déjà bien engagés. Donc, pour toutes ces raisons, ces méthodes ne sont pas bien adaptées à notre problématique de proposer de nouvelles architectures basées sur des calculateurs simplex.

Nous avons ainsi choisi de développer une nouvelle démarche simple, orientée et adaptée aux exigences CDVE, qui soit compatible avec les méthodes de conception actuelles chez Airbus et qui permette une évaluation rapide étape par étape [Sghairi *et al.* 2008c, Sghairi *et al.* 2009d].

Les analyses du processus actuel de conception chez Airbus montrent que les architectures sont évaluées d'un point de vue des risques, performances et coûts associés par différentes équipes dissociées. Et cela produit un certain nombre d'exigences que devraient satisfaire les architectures pour répondre à des objectifs fonctionnels et non fonctionnels. La conception des architectures sont donc plus ou moins guidées et validées par les exigences.

Vu l'importance de ces dernières, nous avons décidé de les prendre en compte en phase amont de conception, ceci afin de gérer très tôt le meilleur compromis entre, d'une part, les exigences auxquelles sont soumis les systèmes, et d'autre part un dimensionnement de l'architecture.

En définitive, la sous-section suivante propose, à l'intention des concepteurs de commandes de vol, une démarche amont, étape par étape destinée, à les aider au dimensionnement et l'optimisation des architectures.

II.1.3 Démarche de conception « amont » pour les CDVE

On ne peut pas trouver directement une solution architecturale pour le système global tant il est complexe. Aussi le décomposera-t-on d'abord en sous-systèmes (typiquement : calculateurs, actionneur, capteur et réseau de communication), auxquels on intégrera ensuite, itérativement, leurs exigences jusqu'à ce que ces sous-systèmes soient suffisamment optimisés en termes de ressources pour leur trouver des solutions architecturales potentielles.

Plus précisément, la démarche proposée consiste à définir, dans un premier temps, des architectures dites « minimales », c'est-à-dire en termes de briques de base purement fonctionnelles pour chaque sous-système, et donc non tolérantes aux fautes à ce stade. Puis, dans un second temps, chacune de ces briques de base est enrichie au fur et à mesure en analysant les besoins réels de redondance, ce qui permet de justifier chaque surcoût matériel ou logiciel. L'assemblage de ces briques de base enrichies conduit à une architecture finale, qui décrit la structuration d'un système en termes de composants et d'organisation de ses fonctions. L'aspect « *incrémental* » de la démarche est dû à l'injection incrémentale de l'ensemble des exigences « *non fonctionnelles* » en phase amont de conception sur la Figure II.2, qui permet d'aboutir, à partir d'une architecture de base purement fonctionnelle, à une (ou plusieurs) architecture(s) optimale(s) satisfaisant les exigences considérées.

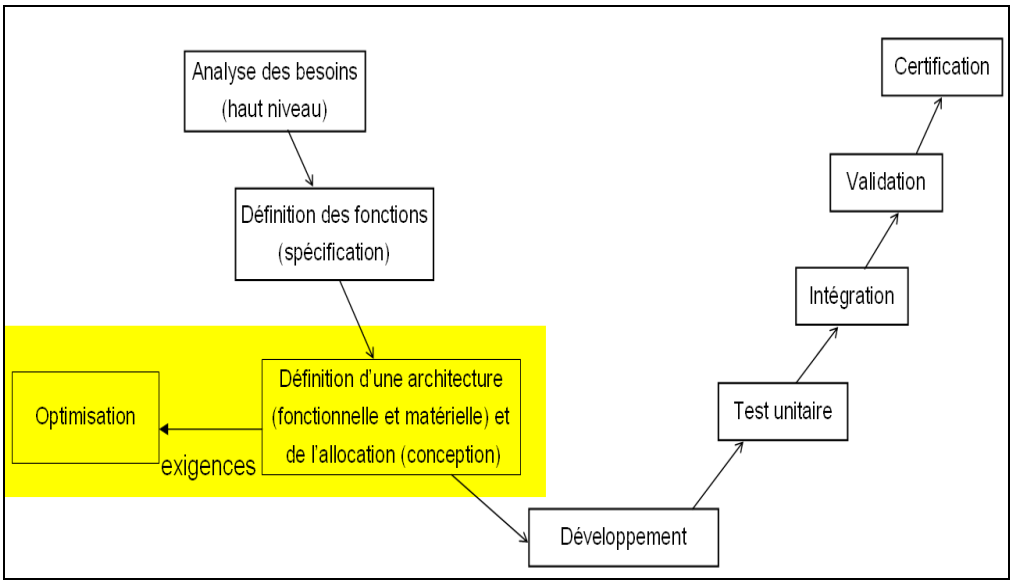


Figure II.2. - Phases d'optimisation

Par ailleurs, il est important de prendre en compte les possibilités d'évolutions technologiques dès les premières étapes de conception. En effet, dans l'intervalle de temps entre la phase de conception et la phase de production d'un programme avion, des évolutions importantes au niveau des composants utilisés peuvent se produire.

C'est pourquoi, dans le cadre de notre travail, nous supposons que, pour les prochaines générations de systèmes de CDVE, les électroniques locales des actionneurs et des capteurs contiendront des micro-contrôleurs dont les processeurs pourront être programmés par le concepteur. Ces capteurs et ces actionneurs auront donc la capacité d'héberger des fonctions CDVE, et ils seront de nouveaux nœuds dans des architectures distribuées en plus des calculateurs centraux CDVE. D'où le besoin d'un nouveau découpage fonctionnel.

II.1.4 Processus de conception architecturale

La redondance matérielle, au niveau des calculateurs, n'est faite que pour tolérer des défauts permanents, alors que la redondance logicielle est ajoutée pour tolérer les fautes de conception.

Pour cela, dans notre démarche, nous avons choisi de traiter :

- les fautes matérielles (ex. : perte calculateur) ou permanentes par la redondance,
- les fautes logicielles ou transitoires (ordre calculateur erroné) par une redistribution de l'intelligence et une reconfiguration logicielle dynamique (à chaud).

Le principe de la démarche permettant de passer de l'expression des exigences et des possibilités de redistribution de l'intelligence à la définition de la solution architecturale la plus optimale est décrit par la Figure II.3.

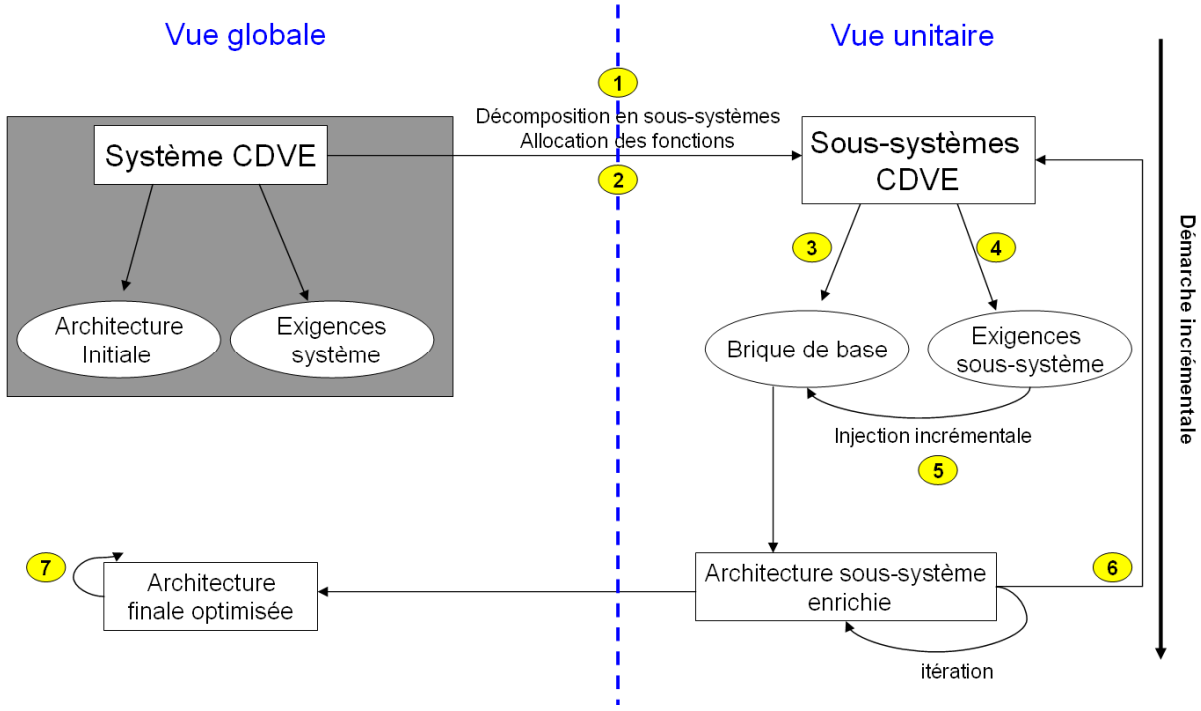


Figure II.3. – Principe de la démarche incrémentale

Nous décrivons ici le processus général qui sera illustré par la suite par une application à la section II.2. Le processus est composé de 7 étapes :

Étape 1 : Analyse architecturale (*fonctions système* et équipements matériels)

Cette première étape du processus produit une décomposition en sous-systèmes physiques, avec leur(s) fonction(s) de base. Elle consiste en l'identification des :

- fonctions principales (tâches fonctionnelles des commandes de vol électriques),
- équipements physiques avec leurs localisations géographiques.

Étape 2 : Allocation des fonctions « non fonctionnelles »

Il s'agit d'une redistribution de l'intelligence ou des tâches non fonctionnelles (actuellement centralisées sur les calculateurs). Cette étape a pour but de considérer les possibilités offertes par les différentes technologies envisageables (niveau d'intelligence par équipement matériel)

pour simplifier l'architecture interne des calculateurs centraux en optimisant l'utilisation des électroniques locales des actionneurs et des composants réseau.

A la fin de cette étape, on aura, par exemple, alloué aux actionneurs la vérification de l'ordre qu'ils ont à appliquer.

Étape 3 : Définition des briques de base, pour aboutir à une architecture « primaire » (ou minimale)

A partir des exigences fonctionnelles, on peut définir des briques de base pour chaque sous-système (calculateurs, actionneurs, composants de communication), et proposer, à partir de ces briques de base, une architecture « *primaire* » non redondante pour le système global. A ce stade, seul l'aspect fonctionnel est pris en compte et les briques de base sont non tolérantes aux fautes.

Étape 4 : Analyse des exigences non fonctionnelles

A cette étape, nous procédons à l'identification et à la classification (par ordre de criticité) des exigences non fonctionnelles (exigences de sûreté de fonctionnement, exigences économiques et temporelles, …) que doit satisfaire chaque sous-système, mais aussi bien sûr, le système global.

Étape 5 : Modélisation à base des exigences non fonctionnelles

On prend un sous-système et on procède à une injection incrémentale des exigences non fonctionnelles afin d'enrichir ses briques de base et de les rendre tolérantes aux pannes. L'injection des exigences est faite jusqu'à ce que toutes les exigences aient été, non seulement intégrées, mais surtout satisfaites, ce qui exige de faire des évaluations, dont en particulier les « *analyses de sécurité* », qui portent sur toutes les exigences (qu'elles soient qualitatives ou quantitatives) dont le respect doit être démontré aux autorités de certification.

Certaines exigences peuvent se traduire par des objectifs probabilistes (ex. : la perte de l'ensemble des gouvernes de roulis doit être extrêmement improbable, soit 10^{-9}/hv), alors que d'autres, non. Et donc, on peut avoir recours à des évaluations quantitative ou qualitative des exigences (cf. chapitre 2, section II.3).

Étape 6 : Itérer l'étape 5 sur l'ensemble des sous-systèmes

L'architecture finale sera la connexion de l'ensemble des architectures des sous-systèmes optimisés.

Étape 7 : Évaluation de l'architecture

L'architecture finale doit être analysée et évaluée pour fournir l'assurance nécessaire que toutes les configurations de panne (ou *FC* pour *Failure Condition*) appropriées aient été vérifiées en considérant toutes les combinaisons significatives de défaillances qui pourraient causer celles-ci.

Du processus que nous venons de proposer pour la conception de nouvelles architectures, nous n'avons décrit ici que les grandes lignes. La compréhension de ce processus nécessite de présenter son application concrète. C'est l'objet de la section suivante.

II.2 APPLICATION DE LA DEMARCHE INCREMENTALE

Cette section est entièrement consacrée à une description détaillée, quasiment étape par étape, de l'application du processus de construction de nouvelles architectures. Cependant, la présentation se limitera à un seul des sous-systèmes à considérer : les calculateurs.

II.2.1 Analyse préliminaire (étape 1)

Pour le passage d'une architecture centralisée à une architecture distribuée, la connaissance des fonctions, de leurs interactions et de leurs caractéristiques (capacité de traitement, taille, localisation géographique…) fournie pour chaque composant matériel est indispensable afin de pouvoir prendre des décisions de redistribution de l'intelligence. Le modèle fonctionnel CDVE est analysé et les fonctions du système sont divisées en sous-fonctions pour qu'elles puissent être allouées ou mises en correspondance avec l'un des équipements du système.

En termes d'équipement matériel, l'analyse est simple et conduit à 4 types d'équipements : les calculateurs, les actionneurs, les capteurs, et le réseau.

Par contre, en termes de fonctions, l'analyse que j'ai menée a été nettement plus lourde. Les principales fonctions ou sous-fonctions CDVE qui ont été identifiées portent sur :

- l'acquisition des ordres des pilotes, des valeurs des capteurs avion et des capteurs de surveillance des actionneurs,
- l'élaboration des ordres de pilotage qui correspond au calcul des lois de pilotage : loi normale pour le mode normal, et loi directe pour les modes dégradés,
- l'élaboration de l'asservissement des actionneurs,
- la validation des informations et la signalisation des pannes des capteurs,
- la surveillance de l'embarquement,
- la détection et la signalisation des pannes des actionneurs et de l'état du système (pilote et maintenance),
- la gestion de la redondance des calculateurs : choix de l'ordre (vote ou logiques de sélection des calculateurs valides),
- les logiques de priorité et de reconfiguration du système.

II.2.2 Distribution de l'intelligence (étape 2)

Cette sous-section discute de choix faits pour la décision d'allocation des fonctions intelligentes et des différentes options possibles de leur répartition. En effet, avec le principe de répartition des décisions et de contrôle, et grâce aux systèmes distribués, il y a plusieurs choix pour savoir où placer les fonctions intelligentes.

Rappelons qu'actuellement les architectures sont physiquement distribuées, mais logiquement centralisées, et les calculateurs centraux CDVE concentrent pratiquement toute l'intelligence du système. Il s'agit d'architectures maître/esclave classiques. Nous qualifierons cette configuration de « *solution A*». L'autre extrême consisterait à supprimer tous les calculateurs centraux et à distribuer l'intelligence à des actionneurs intelligents qui coopèrent entre eux pour contrôler l'avion. Nous qualifierons cette configuration de « *solution B* »

Nous avons opté pour des solutions « *hybrides* » où le contrôle et les surveillances sont partagés entre les calculateurs et les actionneurs qui se surveillent mutuellement. Ce type de solution a pour avantages d'être plus économique que la *solution A*, et plus robuste/sécurisée que la *solution B.*

Et plus précisément, nous avons opté pour la solution hybride qui consiste à :

- d'une part, garder les fonctions de contrôle (lois de pilotage) dans une configuration centralisée dans laquelle les calculateurs conservent la fonction « lois de pilotage » pour de raisons de simplification et de sécurité,
- d'autre part, distribuer les fonctions de surveillance sur des actionneurs intelligents afin de parvenir à une détection de pannes avec moins de matériel.

Il est important de préciser ici que, bien que l'architecture distribuée pour les commandes de vol comprenne deux parties quasi indépendantes : contrôle des actionneurs et traitement des capteurs des commandes des pilotes, notre étude se limite à la partie actionneurs et calculateurs.

Rappelons qu'une raison importante pour l'utilisation d'actionneurs intelligents dans une architecture CDVE distribuée est de parvenir à la détection de pannes avec moins de matériel et de réduire le nombre des calculateurs CDVE centraux en les déchargeant de toute ou partie de ces fonctions de surveillance. Avec plusieurs actionneurs intelligents, un haut degré de détection des erreurs et de localisation des erreurs peut être atteint. De tels nœuds redondants, avec détection embarquée d'erreur, sont extrêmement précieux dans la conception de systèmes distribués tolérants aux pannes où par construction plusieurs surfaces jouent parfois le même rôle, et ont chacune au moins un actionneur.

Par conséquent, la répartition fonctionnelle a été faite en ayant pour objectif d'équilibrer les fonctions réalisées en mode centralisé et en déporté.

Bien entendu, les actionneurs conservent une boucle d'asservissement locale en raison de la contrainte forte sur la latence entre l'acquisition et la réalisation des commandes.

Nous sommes donc arrivés à la répartition fonctionnelle décrite dans le Tableau II.1.

	Calculateurs	**Actionneurs**
Calcul des lois de pilotage (élaboration des ordres de pilotage)	X	
Surveillance de l'embarquement	X	X
Détection et signalisation des pannes des actionneurs et du système	X	X
Choix de l'ordre : vote ou logiques de sélection des calculateurs valides	X	X
Logique de priorité et de reconfiguration du système	X	X
Application des consignes aux actionneurs (élaboration des asservissements)		X

Tableau II.1. - Répartition fonctionnelle

II.2.3 Briques de base et architecture primaire (étape 3)

Cette sous-section présente et justifie les trois types de briques de base qui ont été définies pour construire ensuite une architecture minimale : calculateur, actionneur, réseau.

a) Brique de base calculateur

Rappelons que, dans le cadre de la recherche de nouvelles architectures, nous sommes repartis de l'hypothèse de calculateurs simplex (et non pas COM/MON) basés sur un même type de matériel (*H1*), indépendants les uns des autres, tous avec une même variante logicielle (*A*), tout en prévoyant de rendre possible des opérations de reconfiguration logicielle dynamique. Ces choix sont motivés par les raisons suivantes :

- amélioration de la disponibilité des calculateurs de commandes de vol : actuellement, chez Airbus, la perte de l'unité COM (respectivement MON) d'un calculateur de commandes de vol entraîne systématiquement la perte de tout le calculateur même si l'unité MON (respectivement COM) est encore fonctionnelle ;

- d'un point de vue purement fonctionnel, un seul calculateur simplex suffit pour contrôler l'avion ; aujourd'hui, un DSP 21160N de chez Analog Device ou un Power PC 755 cadencé à 30 MHz est suffisant pour calculer l'ensemble des lois de pilotage et des asservissements ;
- de plus, puisque ces asservissements seront déportés sur les actionneurs, les calculateurs peuvent être plus simples et moins nombreux.

b) Brique de base actionneur

Comme déjà dit, la détermination de l'action à exécuter par un actionneur est déportée au niveau de cet actionneur (selon l'étape 2). Cela nécessite une certaine intelligence locale à l'actionneur. Et donc, pour répondre aux exigences fonctionnelles d'après notre répartition fonctionnelle, nous avons besoin au moins d'une électronique locale (mémoire et capacité de traitement) par actionneur. Le nombre des actionneurs dépend de la configuration avion qui varie d'un modèle d'avion à un autre.

c) Brique de base réseau

Les échanges d'information entre calculateurs et actionneurs doivent désormais s'appuyer sur des communications intégralement numériques. Afin de supporter la distribution de l'intelligence, la configuration sera de type *Full Broadcast* pour que tous les calculateurs puissent communiquer avec tous les actionneurs, et que chaque actionneur puisse communiquer avec tous les calculateurs.

Dans l'étude que nous avons menée sur les réseaux (cf. chapitre I, paragraphe I.3.3), les bus de communication numériques multiplexés sont recommandés en raison de leur fiabilité, disponibilité, maturité, et de leurs faibles coûts. Les possibilités pour les analyses temporelles et la fiabilité sont mieux servies par une topologie de type bus. Mais, pour répondre au mieux aux exigences fonctionnelles (adressage en broadcast), et pour optimiser le câblage, nous avons choisi des topologies en étoile avec la technologie AFDX à base de switchs et micro-switchs AFDX. Les micro-switchs seront placés près des actionneurs alors que les switchs seront plus proches des calculateurs. Cette utilisation permettra de réduire les longueurs de bus sur les grandes longueurs existant entre deux zones telles que : cockpit ↔ voilure gauche, cockpit ↔ voilure droite, cockpit ↔ empennage/fuselage.

Cependant, dans les topologies à bus, les contraintes de ségrégation physique entre bus sont difficiles à tenir dans les zones de voilure et d'empennage. De plus, contrairement aux topologies en étoile, où les liaisons avec les abonnés sont réalisées en point à point de façon similaire au routage des liaisons analogiques actuellement utilisées dans les CDVE, les bus présentent l'inconvénient de partager le même media, ce qui constitue un point commun de panne (perte du bus = perte de toutes les données).

Au final, on retiendra en priorité par la suite, une topologie réseau de type étoile.

Enfin, en termes d'exigences sur les capacités de transmission de données et de débit, celles-ci ne sont pas très importantes pour les commandes de vol : en moyenne, mot de 70 bits toutes les 10 ms qui est le cycle de base des échanges (pour le « rafraichissement » des commandes). Les débits offerts par les nouvelles technologies sont largement suffisants pour ces besoins.

d) Architecture primaire ou minimale

La connexion de différentes briques de base définies pour les principaux sous-systèmes donne une image de l'architecture primaire (ou minimale) globale. La connexion se fait très simplement du fait du faible nombre de briques à relier. Elle est décrite par la Figure II.4.

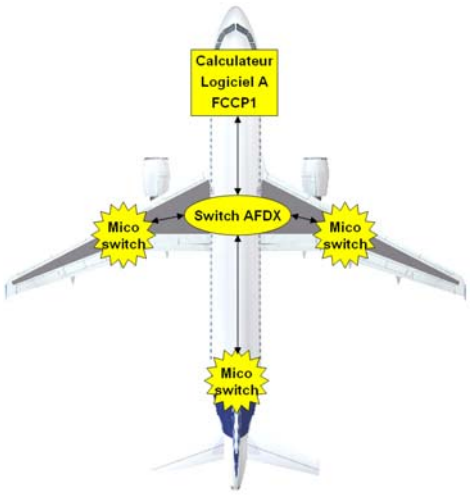


Figure II.4. - Architecture primaire (ou minimale)

II.2.4 Analyse des exigences non fonctionnelles (étape 4)

Pour connaître les équipements matériels supplémentaires nécessaires pour enrichir l'architecture primaire et, par la suite, définir l'architecture finale (optimisée), les exigences non fonctionnelles doivent être bien identifiées et précisées pour tous les sous-systèmes CDVE. Dans cette sous-section, nous allons préciser ces exigences pour les deux principaux sous-systèmes CDVE : les calculateurs CDVE et les actionneurs.

II.2.4.1. Exigences globales (pour tout le système)

Le système CDVE est susceptible de satisfaire un certain nombre d'exigences qualitatives ou quantitatives. Ce système a schématiquement deux types d'objectifs de sécurité :

a) la probabilité d'une panne catastrophique doit être inférieure à 10^{-9} par heure de vol (on qualifie aussi ce niveau de « extrêmement improbable »),
b) il ne doit pas se produire de panne simple catastrophique.

En fait, ce qu'il est important, c'est de comprendre que, de manière générale, toutes les exigences définies au niveau global sont à décliner sur chacun des différents sous-systèmes CDVE. Cela conduit à définir de nouvelles exigences spécifiques à chacun d'entre eux.

II.2.4.2. Exigences calculateur

Au niveau d'un calculateur, nous distinguons les exigences de sécurité, les exigences supplémentaires et économiques. Chacune des exigences développées sera notée *Ei*, car elles seront reprises au II.2.5, lors de la description de l'injection des exigences à l'étape 5.

a) Exigences de sécurité (*E1*) : ce sont les exigences réglementaires dont les deux principales sont :

- l'intégrité : éviter les altérations pouvant conduire à des dysfonctionnements,
- la disponibilité : éviter de perdre une fonction.

b) Exigences économiques :

- Exigence de fiabilité opérationnelle (dispatch possible avec une panne préexistante) (*E3*): afin de permettre aux compagnies aériennes d'organiser plus facilement la maintenance de leur flotte, il convient d'exiger que le système de CDVE reste encore utilisable avec le niveau de sécurité attendu, même si un équipement en panne ne pourra être réparé qu'après plusieurs jours (retour de l'avion à la base). Là encore, sans cette exigence, un avion hors de sa base, ayant un équipement en panne, pourrait être interdit de vol dans l'attente de son équipement de rechange. La prise en compte de cette exigence va dans le sens d'augmenter les redondances pour éviter que la planification des vols de l'avion ne soit pas remise en cause ; on prendra un facteur 10 sur le MTBF comme marge.

c) Exigences supplémentaires : les risques à couvrir sont les suivants :

- Mauvais lots de composants (*E2*) : une dégradation d'un facteur 10 du MTBF peut être constatée à l'essai en vol ou en service sur un ou plusieurs avions.
- Erreur de conception logicielle (spécification incorrectement implémentée) (*E4*): risque de dysfonctionnement cohérent des diverses unités d'accueil (ex. : symbole filtre mal codé)
- Erreur de spécification (*E5*) : risque associé aux spécifications complexes et pouvant amener au déclenchement intempestif d'une surveillance. Ces risques sont à considérer pour les lois de pilotage automatique (PA) et la loi manuelle normale, mais pas pour la loi directe (loi simple et élémentaire)
- Erreur de conception matérielle (*E6*) : risque de dysfonctionnement (non cohérent) des diverses unités d'un même type matériel (ex. : sensibilité de composants à une température élevée, à une alimentation électrique anormale, à une perturbation foudre, etc.).

II.2.4.3. Exigences actionneur

Au niveau d'un actionneur, on distingue des exigences de sécurité et des exigences de disponibilité (ou fiabilité opérationnelle).

a) Exigences de sécurité :

- Perte de la commande de roulis : extrêmement improbable (10^{-9}/hv)
- Perte de la commande de tangage : extrêmement improbable (10^{-9}/hv)
- Perte des deux gouvernes de profondeur : extrêmement improbable (10^{-9}/hv)
- Embarquement du Plan Horizontal Réglable : extrêmement improbable (10^{-9}/hv)
- Intégrité : éviter les altérations (data, perte routage, etc.) pouvant conduire à des dysfonctionnements.

b) Exigences de disponibilité :

Le décollage doit être possible avec un actionneur en panne. La perte d'un actionneur est mineure (10^{-4}/hv).

Un choix important qui a été fait dans le cadre de la thèse est que, pour une meilleure détection d'erreur en local, et pour augmenter la fiabilité des retours des actionneurs vers les calculateurs, nous avons proposé d'utiliser une structure COM/MON pour les actionneurs avec une logique de choix ou de vote. Et, vu le rapport entre la fiabilité des électroniques locales et celle des actionneurs, on estime qu'il est inutile d'utiliser plus de deux électroniques locales par actionneur.

II.2.4.4. Exigences réseau

Concernant la partie communication, il faut donc s'assurer d'une distribution des connexions (switch et micro-switch) entre les actionneurs et les calculateurs autorisant un fonctionnement du système après une perte commune de plusieurs bus en un point de connexion sur un équipement. Dans le cas des architectures que l'on propose, plusieurs topologies sont possibles (voir annexe 1).

II.2.4.5. Exigences d'installation

Il est nécessaire d'assurer une ségrégation électrique générale entre le coté 1 et le coté 2 (parties droite et gauche de l'avion), et l'installation doit couvrir les risques particuliers (éclatement moteurs).

II.2.5 Injection des exigences et optimisation (étape 5)

Nous allons maintenant voir comment les différentes exigences sont injectées lors de la conception du système. Pour cela, nous allons nous baser sur une étude de cas : le sous-système des calculateurs CDVE (le cas des actionneurs et du réseau ne seront pas développés).

Les systèmes de commandes de vol sont actuellement mis en œuvre avec des calculateurs centraux, fiables, tolérants aux pannes, mais qui sont complexes et coûteux. Les calculateurs proposés sur le marché sont si peu fiables (MTBF = 50000 h) pour les applications considérées ici que le recours à la redondance est indispensable afin de permettre au système de calculateurs de répondre aux exigences CDVE.

Comme mentionné précédemment, la capacité de calcul de microcontrôleurs va augmenter au fil des ans en même temps que le fonctionnel demandé. Dans un système distribué, la capacité de calcul dans les calculateurs n'est pas considérée comme un problème dans le travail présenté ici. Cet aspect n'est donc pas étudié.

Mais bien entendu, le recours à la redondance reste indispensable pour définir l'architecture finale de sous-systèmes calculateurs (définie à l'étape 3) dans le processus d'injection des contraintes calculateurs. Mais il ne doit pas être systématique car la redondance a aussi un coût. On propose, dans la mesure du possible, de la remplacer par des reconfigurations ou des fonctions logicielles, ceci avec le niveau de sécurité requis pour les CDVE.

La Figure II.5 résume le processus d'injection des exigences calculateur, étape par étape, avec comme légende pour simplifier la représentation :

- H1 : matériel de type 1 (calculateurs en jaune)
- H2 : matériel de type 2 (calculateurs en vert)
- A : logiciel de type 1 pour les lois normale et directe
- B : logiciel de type 2 pour les lois normale et directe
- C : logiciel de type 3 pour la loi directe
- TaF : Tolérance aux Fautes

La première colonne indique la nouvelle exigence satisfaite à l'étape considérée (les exigences précédemment satisfaites, le restent, mais elles ne sont pas rappelées). La deuxième colonne précise le nombre et le type de calculateurs et logiciels utilisés qui ont été nécessaires pour satisfaire l'exigence. Sous ces considérations, la colonne 3 indique la prochaine exigence à satisfaire, avec l'objectif probabiliste alors visé (colonne 4), les risques ou pannes à considérer (colonne 5) et la technique de tolérance aux fautes à considérer (colonne 6) pour tenter de satisfaire l'objectif. Les architectures obtenues à chaque étape ne sont pas montrées ;

seules les architectures minimale et finale, ainsi qu'une architecture intermédiaire sont décrites dans la figure. Enfin, sur l'architecture minimale (de départ), on voit 3 actionneurs. Là, il ne s'agit pas de redondance, mais c'est simplement que le calculateur doit commander 3 actionneurs. De ce fait, ceux-ci ne sont plus représentés dans le reste de la figure.

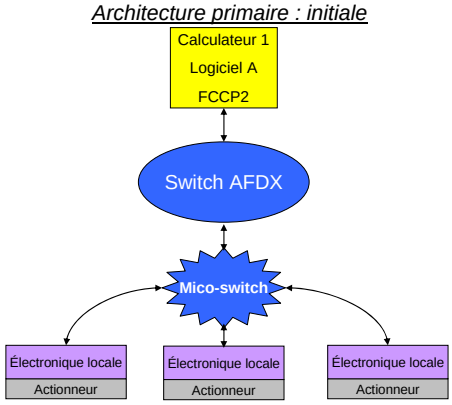
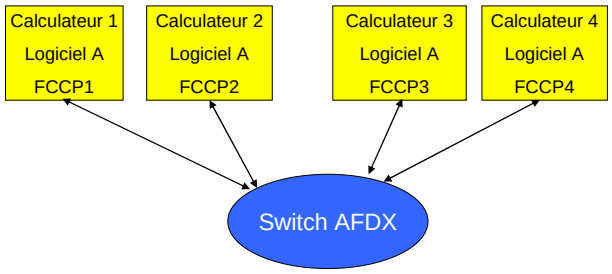
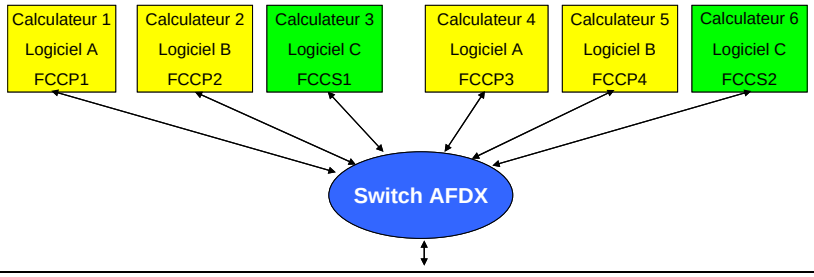
Exigence satisfaite	*calculateur et logiciel : nombre et type*	*Exigences à satisfaire*	*Objectif probabiliste*	*Risques ou pannes à considérer*	*Technique de TaF appliquée*
			Architecture primaire : initiale 			
Aucune	1 : (H1, A)	E1	10^{-9}/ hv	Perte ou information erronée d'un calculateur	Redondance matérielle
E1	2 : (H1, A)	E2	10^{-9}/ hv	Dégradation du MTBF calculateur	Redondance matérielle
E2	3 : (H1, A)	E3	10^{-9}/ hv	Panne préexistante d'un calculateur	Redondance matérielle
			Architecture intermédiaire 			
E3	4 : (H1, A)	E4	10^{-5}/ hv	Erreur de conception logicielle	Diversification logicielle
E3	4 : 2(H1, A)+ 2(H1, B)	E4		Perte de deux calculateurs similaires	**Reconfiguration logicielle**
E4	4 : 2(H1, A) + 2(H1, B)	E5	10^{-5}/ hv	Erreur de conception matérielle	Diversification matérielle
E5	5 : 2(H1, A) + 2(H1, B) + 1(H2, C)	E6		Erreur matérielle	Redondance matérielle
E6	6 : 2(H1, A) + 2(H1, B) + 2(H2, C)	aucune			
			Architecture finale : enrichie 			

Figure II.5. – Exemple d'injection des exigences non fonctionnelles sur le sous-système calculateur

II.2.6 Architecture finale ou optimale (étape 6)

L'étape 6 du processus d'élaboration d'une architecture est l'itération de l'étape 5 sur l'ensemble des sous-systèmes. A l'issu de ces itérations, on aboutit à une architecture finale, aussi dite optimale. Pour l'exemple développé, cette architecture est décrite ci-après.

En termes de matériel, l'architecture optimale est à base de 6 calculateurs « simplex », au lieu d'un minimum de 8 (par référence à l'A400M). Ces calculateurs sont configurables à chaud en fonction des défaillances (logicielle ou matérielle). Ils se répartissent comme suit :

- 4 unités sont dédiées aux fonctions évoluées (FCCP – primaire) pour la loi normale et la loi directe,
- 2 unités sont dédiées aux fonctions simples (FCCS – secondaire) pour la loi directe seulement.

En outre, les 6 calculateurs CDVE sont répartis en deux groupes de 3 (2 calculateurs primaires et 1 secondaire) alimentés chacun par un système électrique ségrégué, chacun des systèmes électriques assurant l'alimentation électrique de trois calculateurs. Enfin, ces calculateurs sont localisés en baie avionique.

En termes de logiciel, l'architecture utilise 3 logiciels CDV au lieu de 4 (cf. Tableau I.1. du chapitre I) :

- 2 variantes dissimilaires, logiciel A et logiciel B, pour le logiciel des lois normale (basique complexe) et directe (simple) des calculateurs primaires FFCP,
- une seule variante, logiciel C, pour le logiciel de la loi directe des calculateurs secondaires FCCS.

Pour chacun de ces logiciels fonctionnels, on suppose qu'il existe deux partitions indépendantes :

- une partition loi normale sur les calculateurs primaires,
- une partition loi directe sur les calculateurs primaires et secondaires.

Enfin, en termes d'actionneurs, chaque actionneur est connecté aux deux côtés, droit et gauche (ce qui correspond aux 2 groupes de calculateurs définis ci-dessus). De ce fait, les actionneurs communiquent avec tous les calculateurs. De manière similaire à la logique actuelle des calculateurs PRIM ou SEC, un actionneur comporte deux unités (ou voies) matérielles quasiment identiques : l'unité commande (COM) et l'unité moniteur (MON).

L'unité COM réalise les fonctions de choix de l'ordre à appliquer, ou, selon le cas, du calculateur valide (ce point sera précisé brièvement dans la section II.4, puis dans les chapitres consacrées aux architectures proposées). L'unité MON (surveillance) effectue, de son côté, les mêmes types d'opérations. À la sortie, les valeurs obtenues par chaque unité sont comparées et, s'il y a un désaccord, l'actionneur est désactivé.

II.3 ANALYSE QUANTITATIVE DE SURETE

Les exigences non fonctionnelles qui sont injectées de manière incrémentale à l'étape 5, le sont jusqu'à obtention de la satisfaction de ces exigences. La détermination de cette satisfaction se fait par des analyses de sûreté. C'est un point clé dans la démarche, puisqu'il en découle le niveau de redondance, que l'on cherche justement à cerner au plus juste.

Nous donnons ici, quelques exemples purement illustratifs des calculs à la base de ces évaluations, surtout utilisés pour faire des premières estimations de satisfaction. Il s'agit seulement de donner une idée des calculs menés. De ce fait, il faut bien comprendre que tous les calculs ne sont pas totalement détaillés, faits à partir de formules simplifiées, et que les principes de ces calculs ne sont pas spécifiques à Airbus.

II.3.1 Rappel de définitions et de propriétés générales

II.3.1.1. Fiabilité, taux de défaillance et MTBF

En termes simples, chez Airbus, la fiabilité d'un équipement est son aptitude à ne pas tomber en panne en cours d'utilisation. C'est plus précisément la probabilité de bon fonctionnement ou de non-défaillance :

- dans des conditions d'utilisation données,
- pendant une certaine période d'utilisation, tout en considérant qu'il était en état de bon fonctionnement au début de cette période.

La fiabilité est exprimée en fonction du temps d'utilisation, qui, pour les systèmes de CDVE est soit la durée d'un vol, soit la durée de vie du système, qui peut être extrêmement importante, et donc pas toujours très « parlant ». C'est pourquoi, on effectue classiquement un changement d'échelle, en passant du temps « mission/vie » à un temps exprimé en « heure de vol » (noté « hv »). C'est ainsi qu'est classiquement défini un autre indice : λ, *le taux de défaillance*, qui est le nombre de défaillance par unité de temps de vol (l'unité est l'heure).

L'inverse du taux de défaillance définit le temps *moyen avant défaillance*, aussi noté MTBF (Mean Time Before Failure).

Pour un intervalle de temps [t, t + dt], on considère habituellement deux probabilités :

- $P(t)$: probabilité de panne durant [0, t], l'équipement fonctionnant à l'instant t_0 (t= 0)
- $\lambda(t)$: taux de panne durant [t, t+dt], l'équipement fonctionnant à l'instant t.

$$P(t) = 1 - exp(-\lambda.t)$$

Pour un équipement électronique, on considère généralement que le taux de défaillance est constant pendant la vie normale de l'équipement (on ne considère que les défaillances aléatoires), et que pour t ≤ 10 % du MTBF (soit $\lambda.t \leq 0.1$) on a :

$$P(t) = \lambda.t$$

II.3.1.2. Calcul des probabilités des défaillances

Dans le cadre de la thèse, ces calculs se font sous les règles suivantes :

- Pour tout événement φ, on a : $0 \leq P(\varphi) \leq 1$;
- Pour les analyses de sûreté, on considère que P (événement sûr) = 1 (ex. : perte d'un calculateur) ;

- P(A et B) = P(A) x P (B), A et B étant 2 équipements ou événements indépendants ;
- P(A ou B) = P(A) + P (B).

A partir de là, nous allons maintenant définir les probabilités, pour un équipement, de tomber en panne selon que la panne est active ou passive (cf. chapitre I, section I.1.2.1.4), et cela, pour un système S, d'abord simple, puis formé de plusieurs sous-systèmes.

Alors, pour tout système S avec un taux de défaillance λ, on a :

- Si on considère que la panne est active, alors la probabilité qu'à la fin du vol le système S soit en panne correspond à la probabilité qu'il y ait eu une défaillance entre l'instant 0 (t_0), début du vol, et T0 (la durée moyenne d'un vol), soit :
$$P(t) = \lambda.T0$$
- Si on considère que la panne est passive ou cachée, alors S est contrôlé toutes les périodes de T1 heures de vol. T1 est le délai maximal pour effectuer une réparation, et T1 est un multiple entier N de vols (ou missions) T1 = N.T0. Le calcul se fait pour un vol V_i quelconque de cette période T1, avec $1 \leq i \leq N$ où V_i est le i ème vol depuis le dernier contrôle de S. Soit au final :
$$P(t) = \lambda.T1 = \lambda.N.T0$$

Si S est composé de plusieurs sous-systèmes avec un taux de défaillance individuel λ, alors :

- La probabilité de la panne totale est calculée à partir de la probabilité de ses coupes minimales. Une coupe minimale représente la plus petite combinaison d'évènements (ou de panne de sous systèmes) pouvant conduire à la perte de système global. On parle parfois de « chemin critique ».
- Le temps de vol moyen est négligeable devant les temps entre contrôles et donc, pour simplifier les calculs, on considère que les sous-systèmes qui ont des pannes passives sont déjà en panne au début du vol, et que les sous-systèmes qui ont des pannes actives tombent en panne au cours du vol.

II.3.2 Exemple de calcul pour le sous-système calculateurs

Dans toutes les analyses de sûreté, la durée du vol est prise égale à 10 heures (long courrier) et un délai maximum pour réparation de 200 heures. C'est un choix conservateur.

L'exemple développé ici est largement simplifié par rapport au cas réel. Son but est seulement d'illustrer simplement les principes de calcul utilisés. Pour ce faire, nous allons les appliquer à l'étape 5 du processus de la démarche incrémentale à un des sous-systèmes des CDVE : celui des calculateurs (en rappelant que la perte au cours d'un même vol de tous les calculateurs de vols redondants conduit à la perte du système).

Donc, pour le système considéré on a, dans notre exemple (avec les simplifications retenues) :

- composition : calculateurs simplex et de leurs 2 barres d'alimentation électriques,
- fonction assurée : calcul de la loi normale
- exigence sur le système : probabilité de perte de la loi normale < 10^{-7}/hv
- pannes considérées : perte d'une barre électrique (PP1 ou PP2) ou perte du calculateur lui-même
- types de pannes : actives
- valeurs numériques :
  - $\lambda e = 2.10^{-6}$ /hv : taux de défaillance de la barre électrique 1PP ou 2PP (une panne au niveau d'une barre électrique affecte tous les calculateurs alimentés par cette barre)

- $\lambda 1 = 10^{-5}$ /hv : taux de défaillance d'un calculateur simplex
- d = 10 : coefficient de dégradation du MTBF (ex. : mauvais lot de composants)
- T0 = 10 heures : temps moyen d'un vol
- T1 = 200 heures : délai maximum pour réparation
- P(t) : Probabilité moyenne (sur l'ensemble de tous les vols) de panne par heure de vol

A partir de là, on doit montrer que toute combinaison de pannes de calculateurs ou de perte des barres électriques ne devra pas provoquer la perte du sous-système de calculateurs.

Rappelons enfin une hypothèse forte : dans les architectures à base de calculateurs simplex, pour éviter des cas triviaux d'états non sûrs, un calculateur seul n'est pas autorisé à assurer le contrôle, car on ne sait pas si l'information qu'il délivre est correcte ou non : cette information doit être « surveillée » (on parle aussi de « contrôle non surveillé d'un seul calculateur »). Donc, sur un ensemble de calculateurs, si tous sont perdus, sauf un seul, celui-ci sera alors aussi considéré comme perdu (puisqu'il n'aura pas l'autorité pour assurer le contrôle).

Dans notre exemple, on a pris deux phases intermédiaires du processus de calcul qui est illustré par la Figure II.6. On part d'un seul calculateur simplex et d'une seule barre d'alimentation (ce qui, au terme de l'étape 3 de la démarche incrémentale, est suffisant pour assurer les aspects fonctionnels du sous-système considéré).

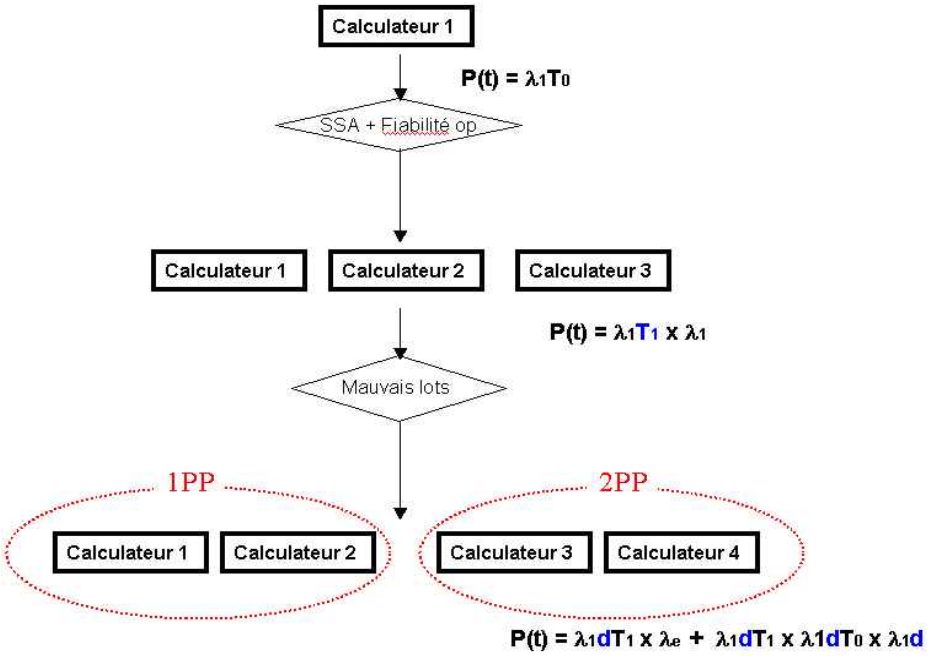


Figure II.6 - Exemple de calcul pour l'étape 5 du processus démarche incrémentale : injection des exigences non fonctionnelles dans le sous-système calculateur.

Dans un premier temps, on injecte les exigences de sécurité et celle de fiabilité opérationnelle. Ce qui conduit à ajouter 2 autres calculateurs. Alors, l'analyse de sûreté quantitative donne :

$$P(t) = P \text{ (coupe minimale = perte de deux calculateurs)} = \lambda 1.T1.\lambda 1$$

La perte combinée des 2 calculateurs fait perdre le système. Sa probabilité est de $2.10^{-8} < 10^{-7}$ et satisfait donc l'objectif de sécurité et de fiabilité opérationnelle.

Dans un deuxième temps, on injecte les exigences sur la qualité des lots de composants, c'est-à-dire, tenir compte de la dégradation de MTBF. Avec notre hypothèse d'un coefficient de dégradation de 10, la probabilité devient $\lambda 1.d.T1 + \lambda 1.d$, qui vaut $2.10^{-6} > 10^{-7}$. D'où le besoin d'ajouter un quatrième calculateur. Alors, l'analyse de sûreté quantitative donne :

P1 (t) = P (coupe minimale = perte de trois calculateurs) = $\lambda 1.d.T1$ x $\lambda 1.d.T0$ x $\lambda 1.d$
P2 (t) = P (coupe minimale = perte d'un calculateur et de la barre 1PP) = $\lambda 1.d.T1$ x λe

Les circuits électriques sont au nombre de 2 (puisque une barre seule est à 2.10^{-6}). La perte totale d'électricité est extrêmement improbable. Si on considère les barres électriques dans nos analyses de sûreté, la perte combinée des 2 calculateurs liés à une barre électrique et d'un calculateur ou de trois calculateurs dans un ordre quelconque fait perdre le système. Sa probabilité est $> 10^{-7}$.

II.4 NOUVEAUX CONCEPTS ARCHITECTURAUX ISSUS DE LA DEMARCHE : PRINCIPES DE FONCTIONNEMENT

Si la section II.2 a permis de jeter les bases méthodologiques pour construire de nouvelles architectures, il reste encore, pour finaliser de telles architectures, à définir les logiques de priorité, de reconfiguration système et la logique de sélection du calculateur valide.

Nous présentons donc maintenant dans cette section deux architectures issues de la démarche développée à la section précédente :

- une architecture dénommée *« architecture à vote massif »*
- une architecture dénommée *« architecture à priorité »*

Ce sont des architectures optimales, et satisfaisant un ensemble de contraintes (objectifs de sécurité et disponibilité), pour relier les calculateurs aux actionneurs des commandes de vol.

Dans ces nouvelles architectures, la comparaison et la validation des ordres des calculateurs ne s'effectuent plus au niveau de ces calculateurs, mais au niveau de chaque actionneur selon deux options. Chaque actionneur est ainsi en mesure par lui-même :

- de déterminer, à partir des ordres calculateurs qui lui sont transmis, l'action à exécuter sur la surface mobile de vol ;
- et de déterminer, grâce aux moyens logiques qu'il comporte, sur quels ordres se baser pour définir l'action à exécuter et de rejeter les ordres dont la comparaison avec les autres ordres montre qu'ils présentent une anomalie.

Rappelons que l'étape de détermination de l'action à exécuter étant déportée au niveau de l'actionneur, ceci permet également de réaliser des calculateurs plus simples et moins coûteux tout en offrant une plus grande flexibilité pour l'agencement du système.

Ceci permet en particulier de s'affranchir des architectures « COM/MON » et « maître/attente » de l'art antérieur (ce qui minimise significativement le nombre total de calculateurs) tout en conservant un haut niveau de sécurité.

Les deux sous-sections suivantes présentent brièvement le principe de fonctionnement des deux architectures à vote massif et à priorité.

II.4.1 Option 1 : Architecture à vote massif

Dans cette première option d'architecture, dite à vote massif, les fonctions remplies par l'unité MON des calculateurs à structure duplex de l'art antérieur sont ainsi mises en œuvre par les actionneurs conjointement avec les autres calculateurs. Tous les calculateurs simplex calculent les lois de pilotages et l'ensemble des ordres de gouverne pour tous les actionneurs.

Ils transmettent leurs ordres gouvernes via les réseaux de communication vers toutes les unités COM et MON de tous les actionneurs, ceci indépendamment de la validité des ordres calculés par chaque calculateur.

Ce sont les deux unités COM et MON de chaque actionneur qui effectuent des opérations de vote sur l'ensemble des ordres reçus de tous les calculateurs afin d'obtenir un ordre valide, (qui est ainsi dit « consolidé » à partir de l'ensemble des ordres indépendants que ces deux unités ont reçus). Les calculateurs reçoivent à leur tour des retours de tous les actionneurs sur la validité de leur ordre.

II.4.2 Option 2 : Architecture à priorité

L'option 2 est une architecture maître/esclave, ou plutôt « *maître/valideurs* ». Tous les calculateurs calculent les lois de pilotage et l'ensemble des ordres gouvernes. Mais, chaque calculateur joue le rôle de *maître* pour un groupe d'actionneurs et le rôle de *valideur* pour les autres actionneurs. Chaque calculateur transmet ses ordres gouvernes vers les actionneurs dont il est maître, et vers les autres calculateurs. Le rôle de *valideur*, du calculateur, consiste à comparer son ordre avec l'ordre du calculateur maître, ceci pour les actionneurs dont il n'est pas maître lui-même (mais seulement valideur). Si le résultat de la comparaison est positif (ordre du maître validé), le calculateur valideur transmet cette information (maître valide) aux actionneurs concernés.

Donc, contrairement à l'architecture COM/MON de l'art antérieur, ici le calculateur valideur ne décide pas lui-même si l'ordre du maître doit être transmis à l'actionneur ; l'ordre du maître est systématiquement transmis à l'actionneur et c'est l'actionneur lui-même qui, grâce aux moyens logiques qu'il comporte, décide, en fonction du résultat renvoyé par le ou les calculateurs valideurs du calculateur maître, d'exécuter ou non l'ordre du calculateur maître.

L'étape de décision étant déportée au niveau de l'actionneur, il est ainsi possible d'associer un calculateur maître à plusieurs calculateurs valideurs, ou bien encore d'associer un actionneur à plusieurs couples maître/valideur de calculateurs pour s'assurer avec une sécurité renforcée de la fiabilité de l'ordre transmis.

Ceci permet de rendre chaque calculateur polyvalent, les rôles entre maître et valideur pouvant être échangés à tout moment en fonction des défaillances des calculateurs ; cela contribue à rendre le système plus flexible et à réduire le nombre total de calculateurs nécessaires.

Chaque actionneur (unité COM et unité MON) reçoit donc un ou plusieurs ordres gouverne (autant que de maîtres potentiels), ainsi que les validités correspondantes (provenant des calculateurs valideurs). Il choisit l'ordre à appliquer en fonction d'une logique de priorité. La logique de priorité sera détaillée avec des exemples dans le chapitre V.

II.5 CONCLUSION

Avec l'évolution des réseaux numériques et des composants électroniques pour l'avionique, et compte tenu du besoin de réduction des coûts dans le contexte de l'évolution des commandes de vol électriques, de nouvelles démarches de conception et des nouvelles architectures pour le système de CDVE ont été proposées dans ce chapitre.

En préalable à toute solution, nous avons d'abord dû réaliser un important travail d'étude, d'analyse et de bilan sur les systèmes de CDVE actuels chez deux avionneurs (Airbus et Boeing), ceci pour apprécier les nouvelles solutions. C'est principalement ce travail que résume ce chapitre, pour terminer sur l'ébauche d'une approche rigoureuse pour la définition de nouvelles architectures.

Les commandes de vol électriques ont marqué une nouvelle étape dans l'avancée technologique des avions. Vraisemblablement, un des maîtres-mots qui vont guider la conception des aéronefs de future génération sera « l'éco-efficience ». Dans cette optique, les commandes de vol se doivent d'évoluer pour proposer un produit moins consommateur de ressources matérielles pour un même service rendu. Dans cette perspective, nous avons proposé une démarche incrémentale de construction de ces architectures de CDVE du futur. Cette démarche permet de garantir que l'on continuera à satisfaire les exigences du système de CDVE actuel : exigences de sécurité réglementaires, précautions supplémentaires, exigences économiques tout en permettant de construire de nouvelles architectures moins consommatrices de ressources.

Pour le futur, de nouveaux systèmes de CDVE sont nécessaires non seulement à contrôler l'avion, mais aussi pour réduire les coûts de développement et de maintenance.

Les deux chapitres suivants exposent en détail les architectures à vote massif et à priorité.

Chapitre III - NOUVELLE ARCHITECTURE A VOTE MASSIF

Nous présentons maintenant dans ce chapitre les caractéristiques de base et le mode de fonctionnement de la première architecture proposée, issue de la démarche développée au chapitre précédent : l'architecture *à vote massif*. C'est une architecture optimale (avec notamment moins de matériel mis en jeu) et satisfaisant un ensemble de contraintes (objectifs de sécurité et disponibilité), pour relier les calculateurs aux actionneurs des commandes de vol.

L'architecture proposée est une architecture distribuée, multi-maîtres et tolérante aux fautes. Elle est basée sur des calculateurs simplex, des actionneurs intelligents et des technologies de communications numériques multiplexées pour l'échange d'information entre l'avionique centrale (calculateurs de commandes de vol) et l'avionique déportée (électroniques locales des actionneurs). L'architecture *à vote massif* présente donc, par rapport à l'existant (architecture matérielle et fonctionnelle), une nouvelle répartition des fonctions intelligentes du système de commandes de vol entre les différentes unités de traitement avec de nouveaux composants (différents, ou simplement modifiés). L'intérêt est de répartir les traitements entre les calculateurs et les électroniques locales des actionneurs avec de nouvelles structures de fonctionnement pour les composants du système (calculateur, actionneurs, …) et des nouveaux protocoles de communication définissant le type et le séquencement des informations échangées entre les calculateurs et les actionneurs.

S'il fallait résumer les idées novatrices de la conception de cette architecture, ce serait que contrairement à l'existant :

1) les calculateurs sont désormais des unités « simplex » (c'est-à-dire à une seule voie), tolérantes aux fautes grâce à leurs surveillances internes,
2) les actionneurs eux, ont désormais une architecture de type COM/MON,
3) les communications sont intégralement numériques entre tous les composants, en broadcast généralisé (tous les calculateurs peuvent communiquer avec tous les actionneurs et vice-versa),
4) la comparaison et la sélection des ordres des calculateurs ne s'effectuent plus au niveau de ces calculateurs, mais au niveau de l'électronique locale de chaque actionneur.

Cela est rendu possible grâce à l'utilisation des électroniques locales des actionneurs aux capacités de traitement accrues et à l'utilisation de communications numériques de type « broadcast » qui permettent à chaque calculateur de communiquer avec tous les actionneurs, et à chaque actionneur de communiquer avec tous les calculateurs.

A noter que, dans la suite, on parlera indifféremment de *« coté »* ou de *« site »* ou *« ensemble »* pour distinguer les deux ensembles de calculateurs alimentés électriquement de manière séparée. Le coté (ou site) gauche sera le numéro 1, et le coté droit sera le 2.

III.1 DESCRIPTION GENERALE : PRINCIPES

La première solution architecturale proposée pour optimiser les architectures actuelles, tout en satisfaisant les mêmes exigences de sécurité et de disponibilité, repose sur un principe de vote massif, qui s'appuie sur plusieurs ordres et utilise différentes logiques de sélection (qui seront précisées dans la section III.2.2.2.3)), et surtout, qui est effectué au niveau des actionneurs. Pour cela, les actionneurs seront dotés d'une intelligence locale (capacité à mémoriser des informations numériques, à les traiter et à communiquer) grâce à des électroniques locales. Ils sont ainsi en mesure : 1) de déterminer par eux-mêmes, à partir des ordres calculateurs qui leur sont transmis, et, grâce aux nouveaux moyens logiques qu'ils comportent, sur quels ordres se baser pour définir l'action à exécuter et 2) d'ignorer les ordres dont la comparaison avec les autres ordres montrent qu'ils présentent une anomalie. Depuis l'A350 et le B787, les actionneurs sont dotés d'une intelligence limitée à la communication en numérique et à la réalisation de la boucle locale d'asservissement (c'est-à-dire contrôler l'amplitude du déplacement de cette surface) à partir de l'ordre validé au niveau des calculateurs centraux.

La Figure III.1 décrit les bases de l'architecture du système de commandes de vol pour le vote massif, et le fonctionnement global de cette architecture est décrit à la suite.

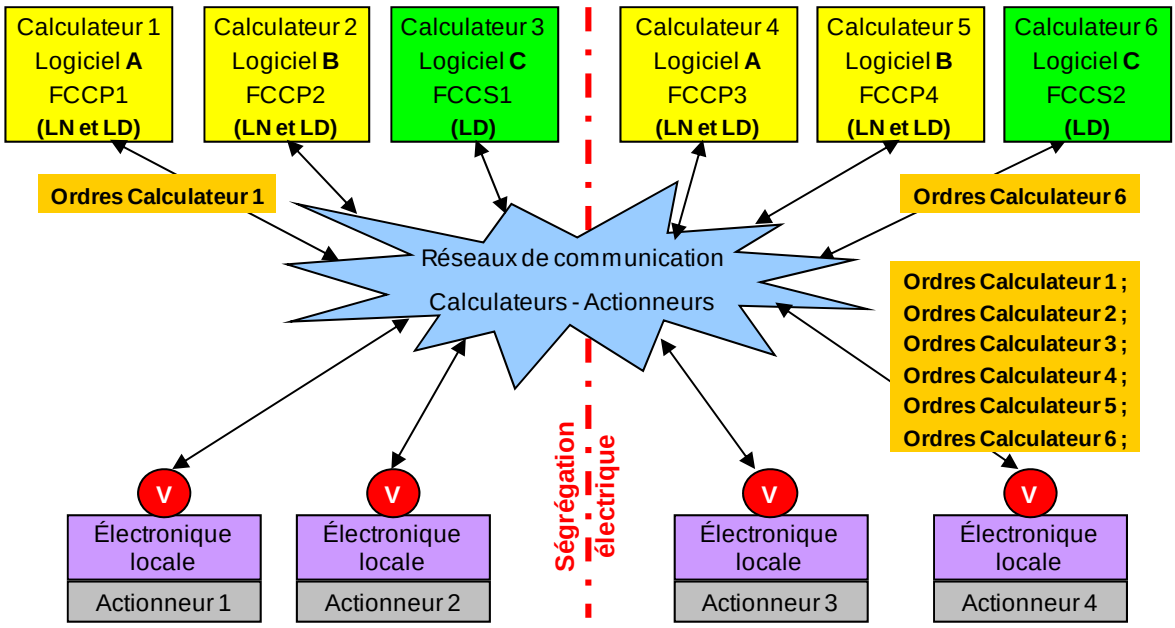


Figure III.1 – Principe de base de l'architecture à vote massif

Tous les calculateurs du côté gauche (respectivement droit) calculent les lois de pilotage et l'ensemble des ordres des gouvernes pour tous les actionneurs valides, que ceux-ci soient en mode actif ou amorti (ou Stand-by). Plus précisément, tous les calculateurs primaires calculent les lois normales (LN) et directes (LD), et tous les secondaires calculent les lois directes (LD). En l'absence de panne, tous les calculateurs transmettent leurs ordres (indépendamment de la validité de ces ordres) via le réseau de communication vers toutes les unités COM et MON de tous les actionneurs, dont ils reçoivent aussi tous les retours.

Ainsi, avant la première panne, chaque actionneur (unité COM et unité MON) reçoit 4 consignes pour la loi normale et 6 pour la loi directe, étant donné qu'il reçoit pour chacun des deux côtés (droite et gauche) 2 ordres pour la loi normale et 3 ordres pour la loi directe.

Les deux unités COM et MON de chaque actionneur effectuent des opérations de vote afin d'obtenir un ordre valide, consolidé à partir de l'ensemble des ordres indépendants qu'elles ont reçus de l'ensemble des calculateurs primaires et secondaires.

1) L'unité COM transmet le résultat de son vote vers son unité MON, et vers tous les calculateurs, en indiquant : 1) la consigne, 2) la loi qu'elle a sélectionné (LN ou LD), 3) les acquittements positifs ou négatifs des calculateurs qui ont calculé un ordre considéré égal ou différent à la consigne sélectionnée (égalité relative avec une marge d'erreur correspondant à un seuil défini par les concepteurs du système de CDVE).
2) Si l'unité MON est d'accord sur les valeurs votées avec l'unité COM, elle transmet les mêmes informations que l'unité COM vers tous les calculateurs. Sinon, en cas de désaccord, l'unité MON bloque la sortie de l'unité COM, l'actionneur est considéré « en panne » et l'information de « **non validité** » de l'actionneur est transmise à tous les calculateurs par la voie MON.

Le fait que chaque calculateur reçoive des retours de toutes les unités COM et MON de tous les actionneurs, rend possible les deux cas suivants de gestion de dysfonctionnement :

1) Le cas d'un calculateur qui détecte un désaccord entre 2 unités COM et MON d'un actionneur ou un comportement erroné de cet actionneur (au vu des retours reçus des autres actionneurs) : alors le calculateur considère cet actionneur en panne et lui envoie une consigne **de non-opération (mode amorti),** jusqu'à ce que ce dernier subisse une opération de maintenance. Mais cet actionneur n'appliquera cette consigne que si elle est consolidée (confirmée) par les autres calculateurs (mécanisme de vote majoritaire).
2) Le cas d'un calculateur dont un ordre envoyé à un « groupe d'actionneurs[1] » est invalidé par une majorité des actionneurs de ce groupe de façon cohérente (COM et MON invalident l'ordre). Le calculateur invalide alors de façon permanente son ordre pour ce groupe d'actionneurs, et ce calculateur est **déclaré en panne,** jusqu'à l'exécution d'une opération de maintenance sur le calculateur. Ajoutons qu'un calculateur peut également lui-même se déclarer en panne via des surveillances internes (aspects développés au III.2.1.2.3).

Dans tous les cas, un calculateur déclaré en panne est ignoré par tous les actionneurs et les autres FCC. Un actionneur (voies COM et MON) qui détecte un calculateur au comportement erroné, l'écarte de son processus de vote de façon permanente (**verrouillage de panne**). Et lorsqu'une majorité d'actionneurs a détecté la perte de deux calculateurs primaires exécutant la même variante logicielle (ex. : A), il y a une **requête de reconfiguration logicielle** vers l'un des deux calculateurs primaires restants valides, pour assurer une dissimilarité logicielle pour la loi normale avec les deux calculateurs primaires restants valides.

De plus, les unités COM et MON de chaque actionneur peuvent partager un même média pour communiquer avec les calculateurs, ceci grâce à l'utilisation de CRC « applicatifs » pour signer leurs messages, ce qui réduit le nombre de liens entre calculateurs et actionneurs.

Enfin, on notera que, dans ce type d'architecture, aucune interconnexion directe entre les calculateurs n'est nécessaire, puisque c'est par l'intermédiaire des actionneurs (comportant la logique de comparaison des calculateurs entre eux) que l'invalidation d'un calculateur est déterminée. De plus, aucune gestion d'engagement des calculateurs, que ce soit en commande ou en asservissement (qui est délocalisé dans l'actionneur maintenant), n'est nécessaire, puisqu'on travaille dans le cadre d'une architecture multi-maîtres.

[1] On appelle groupe d'actionneurs l'ensemble des actionneurs exécutant le même ordre de gouverne. Comme exemples d'actionneurs de surfaces critiques, on peut citer les 4 actionneurs de profondeur et les 3 actionneurs de direction.

III.2 DESCRIPTION DETAILLEE DES DIFFERENTS COMPOSANTS DE L'ARCHITECTURE

Cette section présente dans le détail l'architecture et le fonctionnement des principaux composants de la nouvelle architecture à vote massif proposée : les calculateurs CDVE (FCCP et FCCS), les actionneurs et leur électronique locale (FCRM), mais également, quelques bases sur les relais d'alimentation et les éléments du réseau de communication.

III.2.1 Calculateurs CDVE

Rappelons que les calculateurs de CDVE sont principalement conçus pour calculer les lois de pilotage (en mode manuel et en mode pilotage automatique), dont l'objectif est d'élaborer les consignes de braquage destinées aux actionneurs des gouvernes de l'avion à partir des consignes issues des pilotes et des données issues des capteurs de l'avion (position des organes de commande en cockpit, paramètres anémo-inertiels de l'avion) et suivant des algorithmes de contrôle-commande.

III.2.1.1. Architecture matérielle et logicielle

En termes de matériel, l'architecture matérielle des calculateurs CDVE comprend une seule unité indépendante (unité simplex) qui possède sa propre horloge temps-réel et sa propre alimentation électrique. Elle est composée d'une ou plusieurs cartes CPU supportant les ressources de calcul, de stockage et de communication, ainsi que d'une ou plusieurs cartes d'entrées/sorties numériques et d'une alimentation, etc. De plus, on admet que les calculateurs primaires auront deux partitions indépendantes (une pour la loi normale et une pour la loi directe), et les calculateurs secondaires auront une seule partition (pour la loi directe).

En termes de logiciel, les calculateurs primaires fonctionnent avec une variante de programmes *A* ou une seconde variante de programme *B* pour le calcul des lois normale et directe, tandis que les calculateurs secondaires fonctionnent pour le calcul de la loi directe avec une troisième variante de programme *C*.

Une variante, ou version logicielle, dépend non seulement du type de calculateur (primaire ou secondaire) mais aussi de la reconfiguration de l'ensemble des calculateurs (avant ou après panne). Les calculateurs primaires sont reconfigurables en cours de vol : changement de variantes de programmes embarquées sur le calculateur de *A* en *B*, ou de *B* en *A*. Cette reconfiguration se fait sous certaines conditions, qui seront définies ultérieurement dans ce chapitre. Et pendant la durée de la reconfiguration, seule la loi directe est disponible.

Pour une meilleure (plus sûre) gestion de la reconfiguration des calculateurs, seuls les deux calculateurs primaires (FCCP1 et FCCP2) d'un même côté (le droit a été choisi) effectuent des opérations de reconfiguration. Lors de l'installation des boîtiers calculateurs sur l'avion, on affecte à chacun d'eux un numéro d'identification qui va correspondre à un identifiant physique déclaré à l'ensemble des actionneurs et des calculateurs. Cet identifiant physique permet de spécifier la version logicielle initiale (*A* ou *B*) du calculateur ainsi que sa localisation géographique (côté gauche ou côté droit).

III.2.1.2. Description fonctionnelle

1) Interface système

Pour fonctionner, les calculateurs commandes de vol consomment et produisent des flux de données, précisés dans le Tableau III-1. Ces flux sont reçus et transmis via le même réseau de communication dans une ou plusieurs trames de données selon les technologies et les protocoles de communication qui seront retenus. Ces calculateurs ont également besoin d'alimentation électrique. Celle-ci est fournie par l'un des deux systèmes électriques de l'avion, en fonction de la localisation géographique du calculateur (côté droit ou côté gauche).

Flux de données	*Description*	*Type*	*Flux entrant (source)*	*Flux sortant (destinataire)*
Paramètres lois	Les fonctions lois de pilotage consomment des paramètres « lois » : position envoyée par les organes de commande en cockpit, paramètres anémo-inertiels de l'avion, …		Système avion cockpit, capteurs, …	
Retour_FCRM	Le calculateur reçoit des retours (de toutes les unités COM et MON de tous les actionneurs) sur la validité des traitements qu'il a effectué et par rapport aux résultats de vote effectué par chaque FCRM (actionneur).	Trame FCRM	FCRM	
Ordre actionneurs	Ordres de braquage pour les actionneurs.	Trame FCC		FCRM
Mode_FCRM	Mode d'activation des électroniques locales (FCRM).	Booléens		FCRM
Ordre_RCCB	Mode d'activation des électroniques locales, via les relais électriques.	Booléens		RCCB

Tableau III-1 - Flux de données entrants et sortants des calculateurs

Nous décrivons maintenant un peu plus en détail le format des 2 types de trames nécessaires, tout en précisant que ce format n'est pas encore exactement figé à ce stade de l'étude (notamment le nombre de bits des champs).

a) Trame_FCC : ce sont les trames ou les blocs d'information envoyés par les calculateurs vers les actionneurs. Elles ont la forme suivante :

Adresse destinataire	Adresse source	Type de la trame	Variante logicielle	Ordre Loi Normale	Ordre Loi Directe	CRC

Les données véhiculées par la trame sont :

- *Adresse destinataire* : adresse physique ou identifiant des composants destinataires de la trame (actionneurs). Dans le cas de ces trames FCC, il s'agit en fait d'une adresse unique, de multidiffusion (ou broadcast) correspondant à tous les actionneurs du réseau.
- *Adresse source* : adresse physique ou identifiant du composant émetteur de la trame (calculateur).
- *Type de la trame* : indique le type des données véhiculées par la trame (ordres).
- *Variante logicielle* : indique la version logicielle embarquée sur le calculateur au moment de transmission de la trame (A, B ou C).
- *CRC (Cyclic Redundancy Code)* : champ de contrôle pour l'intégrité de la trame reçue.

Exemple de trame envoyée par le calculateur primaire FCCP1 avant reconfiguration :

FCRMs	FCCP1	Ordres	A	16°	18°	CRC

b) Trame_FCRM : ce sont les trames ou les blocs d'information envoyés par les FRCM (voie COM et voie MON) vers les calculateurs. Elles ont la forme suivante :

Adresse destinataire	Adresse source	Type de la trame	Ordre voté en Loi Normale	Ordre voté en Loi Directe	Loi disponible (Normale ou directe)	Acquittement FCC	CRC

ACK FCCP1 Loi Normale	ACK FCCP1 Loi Directe	ACK FCCP2 Loi Normale	ACK FCCP2 Loi Directe	ACK FCCS1 Loi Directe	ACK FCCP3 Loi Normale	ACK FCCP3 Loi Directe	ACK FCCP4 Loi Normale	ACK FCCP4 Loi Directe	ACK FCCS2 Loi Directe

Les valeurs booléennes utilisées dans le champ « Acquittement » ont pour signification :

- la valeur 1 : pour un acquittement positif (ordre valide)
- la valeur 0 : pour un acquittement négatif (ordre non valide) et pour les trames vides (si on n'a rien reçu des calculateurs),
- un champ vide dans la trame : 0

Par exemple, avec calculateur FCCP1 sous MMEL et FCCP2 au comportement erroné, les trames envoyées par tout FCRM valide ont un champ « Acquittement » codé comme suit :

		ACK	ACK	ACK	ACK	ACK	ACK	ACK	ACK
FCCP1=0	FCCP1=0	FCCP2=0	FCCP2=0	FCCS1=1	FCCP3=1	FCCP3=1	FCCP4=1	FCCP4=1	FCCS2=1

Remarque : un équipement est dit « sous **MMEL** (*Master Minimum Element List*) » s'il est déclaré en panne avant le début du vol, mais peut être autorisé à faire le vol quand même (cas d'un équipement dit GO).

2) Architecture fonctionnelle de base

Le logiciel fonctionnel de chacun des calculateurs peut schématiquement être réparti en deux catégories : le calcul des lois de pilotage (déjà rappelé) et les surveillances. La Figure III.2 représente la structure fonctionnelle de base d'un tel calculateur simplex tolérant aux fautes.

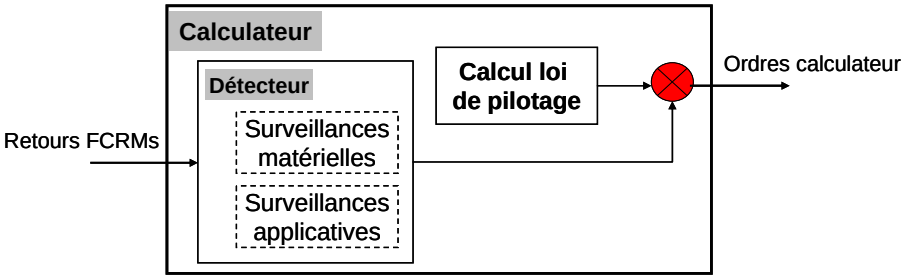


Figure III.2 - Schéma fonctionnel de base d'un calculateur simplex tolérant aux fautes

Pour les **surveillances,** on distingue les surveillances classiques (qui existent déjà dans les calculateurs actuels) et des nouvelles surveillances spécifiques à l'architecture à vote massif. Les calculateurs effectuent des opérations de surveillance sur eux-mêmes, propres aux services qu'ils fournissent aux autres composants du système, pour la vérification de leurs défaillances internes. Ces vérifications visent à s'assurer du bon fonctionnement de chaque calculateur vis-à-vis des autres calculateurs et des actionneurs. On distingue deux types de surveillances : les *surveillances matérielles* et les *surveillances applicatives logicielles*.

1) Les **surveillances matérielles** ou « core software » (logiciel interne) classiques assurent la vérification des fonctionnalités matérielles du boîtier au démarrage du calculateur, ou en phase opérationnelle, comme les surveillances des cartes CPU (watchdog) ou la vérification cyclique des checksums mémoires.

2) Les **surveillances applicatives logicielles** sont de nouvelles surveillances, spécifiques à l'architecture à vote massif pour la vérification de l'intégrité des informations fournies par le calculateur (consignes, modes d'activation…) par rapport aux informations fournies par les autres calculateurs, et en fonction des retours des unités COM et MON de chaque actionneur faits selon des logiques spécifiques à l'architecture à vote massif mises en œuvre au niveau des électroniques locales.

Chaque calculateur est ainsi conçu pour détecter ses propres pannes et inhiber les sorties affectées par ces pannes, tout en signalant son état aux autres composants du système. La surveillance applicative est un détecteur logiciel qui vérifie le fonctionnement du calculateur, (et éventuellement son propre fonctionnement), et émet un signal indiquant que le système global fonctionne correctement. Lorsqu'un signal `stop` (sortie de détecteur) apparaît, l'information (sorties calculateur) est bloquée.

3) Principe de fonctionnement et logiques spécifiques (niveau calculateur)

La Figure III.3 donne le schéma fonctionnel détaillé d'un calculateur simplex.

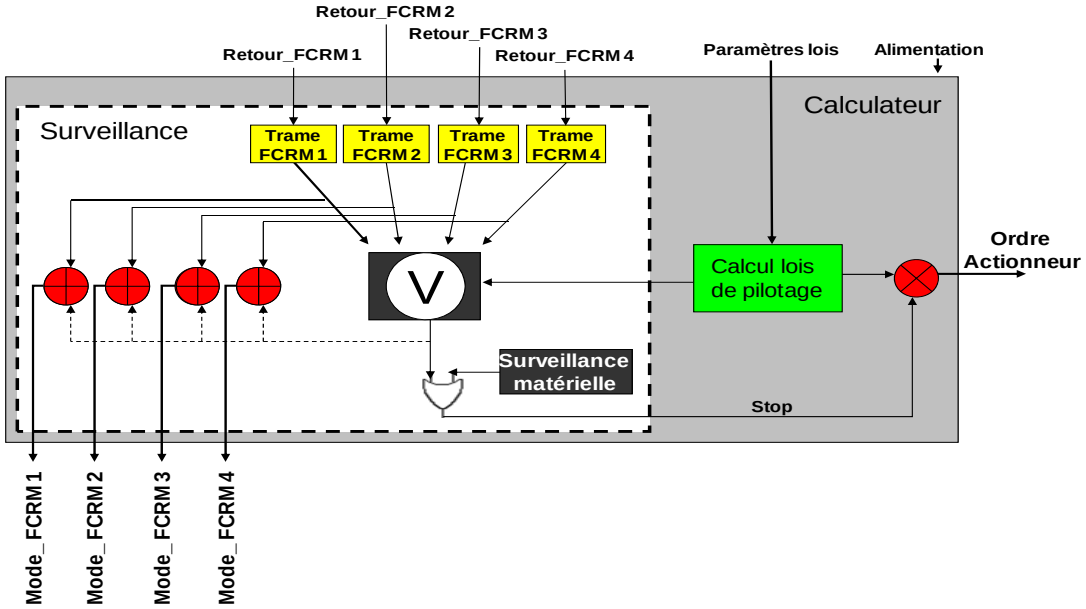


Figure III.3 - Schéma fonctionnel détaillé d'un calculateur simplex

Les calculateurs émettent les résultats de leurs traitements, indépendamment de leurs validités, vers tous les actionneurs via des Trame_FCC (cf. plus haut) sur le réseau de communication. Et les actionneurs sont capables de comparer les ordres des calculateurs entre eux, et sont donc capables de détecter, par rapport à leur vote interne, quels calculateurs sont défaillants et de renvoyer aux calculateurs un message pour les en informer.

Chaque calculateur reçoit donc des retours de toutes les unités COM et MON de tous les actionneurs (via des Trame_FCRM). En se basant sur ces retours des FCRM, un calculateur peut valider ou invalider ses ordres selon sa logique de surveillance interne à l'aide de vote majoritaire et de comparateurs (voir plus loin). Le calculateur invalide de façon permanente son ordre pour un groupe d'actionneurs (envoi d'un signal Stop) lorsque au moins une majorité des actionneurs de ce groupe a invalidé cet ordre de façon cohérente (COM et MON invalident l'ordre du calculateur) en envoyant des acquittements négatifs pour la loi normale ou la loi directe. Et le calculateur est déclaré en panne.

Un calculateur déclaré en panne, soit par lui-même (déclenchement de la surveillance interne classique), soit sur la base des retours des actionneurs (déclenchement de la surveillance applicative interne), est ignoré de tous les actionneurs et des autres calculateurs.

En outre, l'architecture à vote massif permet également un contrôle supplémentaire des actionneurs à partir des calculateurs. En effet, un calculateur qui détecte un désaccord entre deux unités COM et MON d'un actionneur, ou un comportement erroné d'un actionneur (par rapport au résultat du vote que fait ce calculateur sur l'ensemble des retours des actionneurs) considère cet actionneur en panne et lui envoie un ordre de non-opération (mode amorti). Le mode amorti est pilotable par la trame Mode_FCRM envoyée directement au FCRM ou indirectement via son RCCB (Trame_RCCB). Ce dernier contrôle n'est pas obligatoire mais il permet une surveillance optionnelle et une sécurité supplémentaire pour l'architecture globale.

Concernant la logique que chaque calculateur implémente pour s'assurer de la validité de son ordre et pour valider les FCRM, nous allons, pour l'expliquer, nous appuyer sur l'exemple de la gouverne de profondeur qui a quatre actionneurs (deux pour l'elevator droit et deux pour l'elevator gauche), et donc quatre FCRM (deux actifs pour les actionneurs actifs A, et deux en mode amorti pour les actionneurs en mode stand-by S) (cf. Figure III.4).

a) Le calculateur envoie sa consigne (trame_FCC) aux quatre FCRM (dont chacune des voies COM et MON reçoit un exemplaire).

b) Le calculateur reçoit donc, en retour, huit trames_FCRM. Chaque trame indique : les valeurs votées par chaque FCRM (voie COM et voie MON) pour la loi normale et pour la loi directe, la loi finalement disponible pour l'actionneur et les validations (acquittements) calculateurs. Les huit trames sont les suivantes :
   1) trame_FCRM 1 envoyée par le FCRM_COM de l'elevator gauche
   2) trame_FCRM 1 envoyée par le FCRM_MON de l'elevator gauche
   3) trame_FCRM 2 envoyée par le FCRM_COM de l'elevator droit
   4) trame_FCRM 2 envoyée par le FCRM_MON de l'elevator droit
   5) trame_FCRM 3 envoyée par le FCRM_COM de l'elevator gauche
   6) trame_FCRM 3 envoyée par le FCRM_MON de l'elevator gauche
   7) trame_FCRM 4 envoyée par le FCRM_COM de l'elevator droit
   8) trame_FCRM 4 envoyée par le FCRM_MON de l'elevator droit

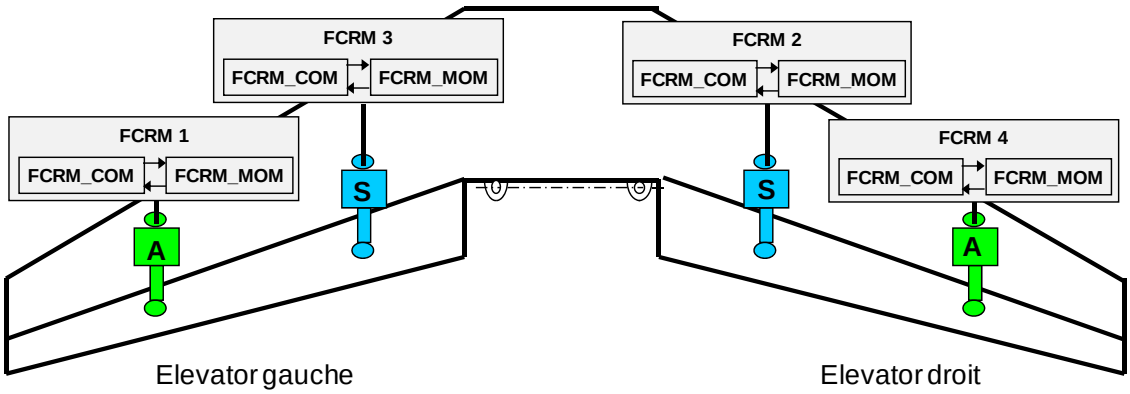


Figure III.4 - Gouverne de profondeur

Remarque : on pourrait envisager de regrouper les quatre trames des FCRM pour n'envoyer au calculateur qu'une seule trame finale avec toutes les informations issues de tout le FCRM. Cela peut faciliter le contrôle de cohérence temporelle des messages [Cauffriez *et al.* 2004]. Mais ce point ne sera pas développé plus avant ici.

Le calculateur effectue alors des opérations de vote majoritaire sur les quatre trames.

- Si le résultat du vote donne un acquittement négatif (calculateur invalidé par une majorité), la surveillance interne logicielle active le signal Stop et le calculateur inhibe ses sorties profondeur (dans l'exemple pris ici).
- Dans le cas contraire, le calculateur vérifie le résultat du vote pour s'assurer qu'il a bien calculé des consignes égales à celles votées par la majorité des FCRM.

Le symbole ⊕ représente un comparateur pour détecter la minorité (si elle existe) des FCRM défaillants qui ont invalidé le calculateur malgré son bon fonctionnement. Chaque comparaison négative sera traduite par un arrêt d'émission de l'ordre RCCB (ce sont des relais électriques intelligents, présenté dans ce chapitre au III.2.3).

Chaque actionneur peut envoyer des requêtes de reconfiguration vers les calculateurs en fonction des défaillances logicielles et matérielles qu'il détecte. Un calculateur appliquera cette consigne de reconfiguration, si et seulement si, elle est consolidée avec les retours des autres actionneurs (mécanisme de vote majoritaire), et si elle lui est bien destinée en fonction de son identifiant physique.

III.2.2 Les électroniques locales FCRM

III.2.2.1. Architecture matérielle et logicielle

Chaque actionneur est équipé d'au moins une électronique locale avec des capacités pour : communiquer sur le réseau, réaliser des opérations de vote et piloter les actionneurs. La Figure III.5 donne un exemple d'architecture matérielle pour un des Elevator de la gouverne de profondeur (déjà présenté au complet) : soit deux actionneurs, donc deux FCRM (un actif pour l'actionneur actif A, un en mode amorti pour l'actionneur en mode stand-by S).

En fait, l'architecture et le mode de fonctionnement s'inspire des principes actuels des calculateurs duplex (COM/MON). Une électronique locale ou FCRM comporte deux unités (ou voies) fonctionnellement équivalentes : l'unité commande (COM) qui pilote les sorties actionneur et l'unité moniteur (MON). Les deux voies ont des matériels et des logiciels identiques ou différents, et sont aussi alimentées électriquement par un équipement dénommé RCCB (Remote Control Circuit Breaker), présenté en détail dans ce chapitre au III.2.3.

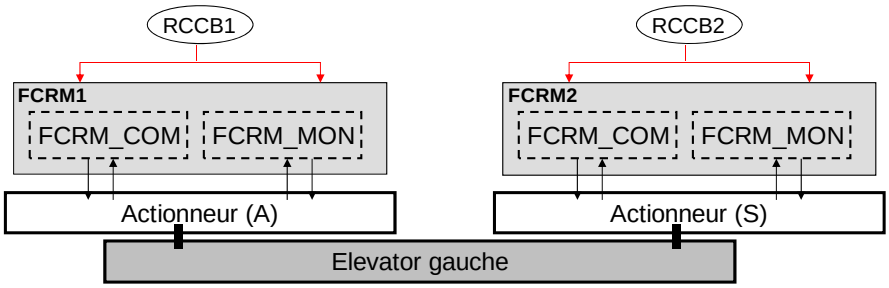


Figure III.5 – Structure matérielle pour le contrôle d'un Elevator

L'unité COM réalise les traitements nécessaires à la réalisation des fonctions du FCRM (vote, requête de reconfiguration…). L'unité MON effectue, de son côté, les mêmes types d'opérations. Les valeurs obtenues sont comparées au niveau de l'unité MON, et avec un seuil de tolérance autorisé, à celle de l'autre unité. S'il y a un écart qui dépasse ce seuil, il y a un désaccord. En cas de désaccord détecté, les sorties FCRM sont automatiquement désactivées, et un autre actionneur (initialement en mode stand-by) prend la main. Avec cette structure, l'émission d'ordres erronés non détectés vers l'actionneur est extrêmement improbable.

III.2.2.2. Description fonctionnelle

1) Interface système

Les électroniques locales de chaque actionneur produisent et consomment des flux de données via le même réseau de communication, dans une ou plusieurs trames de données (selon les protocoles de communication retenus). Le Tableau III-2 décrit ces flux et indique aussi des échanges internes : ceux entre les voies COM et MON du FCRM, via des liens directes inter-unités. Précisons de plus que le FCRM a une alimentation électrique, qui est contrôlée par un relais d'alimentation (RCCB, défini au III.2.3), lui-même contrôlé par les calculateurs.

Flux de données	*Description*	*Flux entrant (Source)*	*Flux sortant (destinataire)*
Ordres_Calculateurs	Ordres des calculateurs (consignes)	Calculateurs	
Ordre_Actionneur	Le FCRM (voie COM) réalise l'asservissement local de l'actionneur.		Actionneurs
Retour _FCRM	Résultat du vote et les acquittements FCRM vers les calculateurs (celui du COM et celui du MON).		Calculateurs
Request_reconfiguration	Requête de reconfiguration pour demander un changement de la variante logicielle embarquée sur le calculateur concerné par la reconfiguration.		Calculateurs
Signal d'autorisation_MON	Un signal d'autorisation de fonctionnement pour la voie COM (selon le résultat de la comparaison).	de MON vers COM	
Traitements_COM	Résultats des traitements de la voie COM vers la voie MON pour la surveillance COM/MON.	de COM vers MON	

Tableau III-2 - Flux de données externes et internes du FCRM

2) Architecture fonctionnelle de base

Les ordres sont calculés par tous les calculateurs. Le choix de l'ordre à appliquer à l'actionneur est le résultat d'un vote réalisé par les électroniques locales de l'actionneur et portant sur tous les ordres qu'il a reçus. Dit autrement, les électroniques locales « sélectionnent » les ordres qui seront appliqués par la voie COM de chaque actionneur. La voie MON surveille la voie COM, et elle a le pouvoir de désactiver le COM et/ou l'actionneur en cas de désaccord. La Figure III.6 donne le schéma fonctionnel de base d'un FCRM.

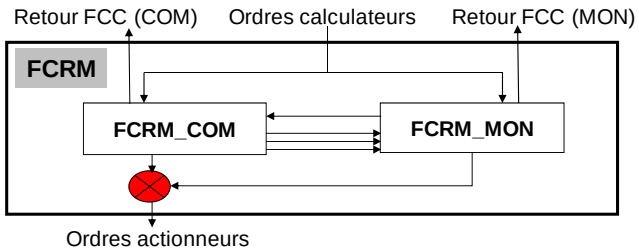


Figure III.6 - Schéma fonctionnel d'une électronique locale

Les traitements (algorithmes de vote) sont basés sur les ordres calculateurs. Le logiciel fonctionnel des FCRM peut schématiquement être réparti en quatre catégories :

a) logique de choix (ou de consolidation),
b) logique de reconfiguration,
c) logique de validation,
d) contrôle et surveillance des gouvernes.

En précisant que la partie commande-surveillance des gouvernes ne sera pas détaillée, la Figure III.7 précise le schéma fonctionnel de la partie COM d'un FCRM vu à la Figure III.6.

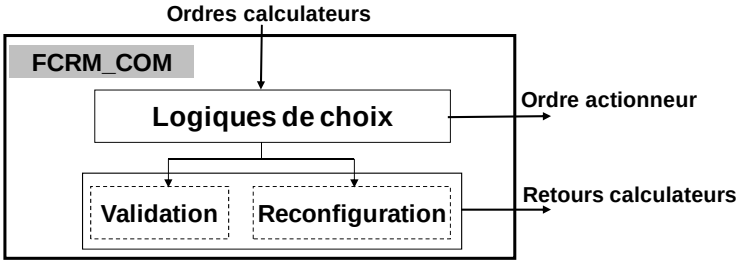


Figure III.7 - Schéma fonctionnel de l'unité COM

3) Principe de fonctionnement et logiques spécifiques (niveau FCRM)

La tolérance aux fautes permet de répliquer des ordres et ainsi de tolérer la défaillance ou la perte de composants comme les calculateurs ou les actionneurs dans une architecture distribuée. Pour la gestion de la redondance et notamment la consolidation des données, il est nécessaire de disposer de protocoles de choix ou d'accord [Parhami 1994] : une logique de choix a pour but de déterminer/sélectionner la valeur d'une grandeur (physique ici) parmi un lot de valeurs de cette grandeur. Dans [Barbaria 2008] est décrit en détail le problème général de consensus, qui est à la base de tous les algorithmes d'accord [Schneider & Lamport 1985]. Et, dans [Namazi & Nourami 2007], [Hardekopf *et al.* 2001a], [Hardekopf *et al.* 2001b], [Boyer & Moore 1991] et [Hesselink 2005], on trouve des descriptions d'algorithmes de vote majoritaire pour la gestion de la redondance dans les systèmes distribués.

Cette partie s'attache à décrire en détail les mécanismes que nous avons mis en place dans ce cadre, avec principalement un long exposé sur les logiques pour consolider les ordres de gouvernes, puis les logiques de reconfiguration logicielle et de validation. Les logiques de choix sont un des fondements des solutions architecturales proposées, et sont l'expression de la distribution de l'intelligence. Sont d'abord présentées les logiques classiques chez Airbus, puis les nouvelles logiques que nous proposons.

a) Les logiques classiques

Chez Airbus, les types de traitements envisagés pour l'obtention d'une grandeur physique à partir de 3 mesures, puis de 2 mesures après une première panne de l'équipement fournissant la valeur, sont les mécanismes classiques de vote suivants : **la moyenne, la médiane, le passiveur**. L'élément réalisant le vote sera appelé « **voteur** ».

Certains de ces traitements sont en particulier utilisés pour l'élaboration des paramètres inertiels ou anémométriques nécessaires aux lois de contrôle. Ces traitements sont également utilisés pour certaines fonctions CDV ou le pilote automatique.

L'efficacité de ces 3 mécanismes est jugée sur la base de critères de **passivation des pannes** et de **précision**. Précisons que « passiver » une panne affectant une mesure consiste à limiter l'effet de cette panne sur la sortie du traitement (ou la fonction calculateur qui utilise la mesure sélectionnée) avant que les surveillances dédiées à ces mesures ne se déclenchent.

- **Moyenne** : le voteur prend trois entrées *E1*, *E2* et *E3* et la valeur choisie est la moyenne *M* des trois mesures, ce qui revient simplement à réaliser $M = (E1+E2+E3) / 3$. La moyenne n'est actuellement pas utilisée chez Airbus. La raison essentielle est qu'elle ne passive pas, par construction, l'embarquement d'une mesure.

- **Médiane** : le voteur prend trois entrées *E1*, *E2* et *E3* et produit sur sa sortie *S* la médiane de ces trois valeurs, c'est-à-dire la mesure comprise entre les 2 autres. Si une des entrées n'est pas valide, alors le voteur opère sur les deux entrées valides restantes et calcule leur demi-somme. La médiane est largement utilisée dans le système de commandes de vol depuis l'A320. Elle permet une passivation naturelle (ou directe) en cas d'embarquement d'une mesure ou de panne oscillatoire.

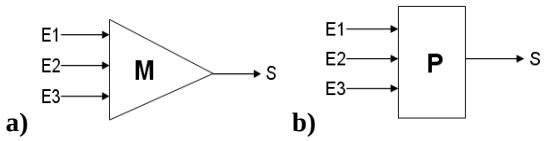


Figure III.8 - Schéma fonctionnel d'un :
a) voteur de la médiane ; b) passiveur

- **Passiveur** : le passiveur prend trois entrées *E1*, *E2* et *E3*, et produit sur sa sortie *S* un résultat selon le principe qui consiste à privilégier une des mesures, *E1*, mais qui est bornée par des limites supérieure et inférieure en fonction des deux autres mesures et de deux seuils de tolérance *C2* et *C3* sur *E2* et *E3* respectivement. L'algorithme est le suivant :

```
Si  E1 > Max   Alors S = Max
  Sinon si  E1 < Min   Alors S = Min
                       Sinon S = E1

avec :
        Si E2+C2 < E3+C3  Alors Max = E2+C2
                                Sinon Max = E3+C3
        Si E2-C2 < E3-C3  Alors Min = E2-C2
                                Sinon Min = E3-C3
```

La comparaison de ces 3 mécanismes montre que la technique de la médiane des trois mesures donne un très bon compromis vis-à-vis des deux critères retenus chez Airbus (passivation des pannes et précision) étant donné qu'elle assure une bonne passivation et une précision très honorable (proche de la vraie valeur de la grandeur physique).

Cependant, dans l'architecture proposée, le nombre de mesures pouvant être supérieure à 3 (6 typiquement), il faut trouver des nouveaux moyens pour combiner plus que 3 mesures d'une même grandeur physique qui réduisent le plus possible l'erreur finale sur la vraie valeur et qui passivent le maximum de pannes. Les mécanismes utilisés chez Airbus ne permettent pas de faire un vote exact (bit à bit) qui demande : une synchronisation des horloges des calculateurs et une synchronisation des valeurs des capteurs reçues par les calculateurs. Mais ces mécanismes assurent une tolérance aux fautes acceptables dans les limites des seuils utilisés.

b) Nouvelle logique de choix : traitement optimisé hybride

L'architecture à vote massif met en jeu de nouveaux types de logique de choix pour sélectionner l'ordre final à transmettre aux actionneurs, plus adaptés au nombre de calculateurs impliqués, à leurs architectures matérielle et logicielle et à leur localisation géographique, sans obligation de synchronisation des horloges des calculateurs [Kossentini & Caspi 2004, Caspi & Salem 2000]. La logique que nous avons développée est composée comme suit, et son principe général est représenté sur la Figure III.9.

- Une première phase de chaque coté « électrique », avec deux types de **votes locaux** (à ces côtés) : un vote » **explicite** » et, si besoin est, un vote « **sélectif** », menés sur des mesures issues des calculateurs d'un même côté électrique (site gauche ou droit),
- Une seconde phase, la sélection finale, avec une logique de comparaison et de choix entre les deux côtés électriques.

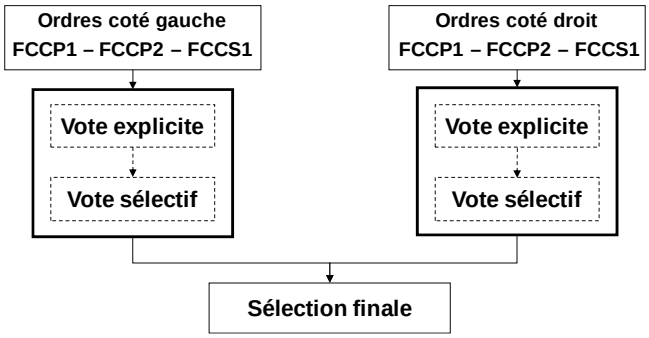


Figure III.9 – Diagramme de l'algorithme global de vote massif

L'algorithme global de vote massif met en œuvre un premier vote au niveau de chaque coté électrique. Ce vote, dit « **vote explicite** », détermine pour les calculateurs d'un même côté, à partir des ordres qu'ils ont fournis (3 pour la loi directe et 2 pour la loi normale), quels sont les ordres valides (acceptables) pour chaque côté. Ce nombre d'ordres acceptables est défini avec un poids *P* (ou facteur de confiance) qui indique le nombre d'unités qui sont en accord.

Toujours au niveau de chaque coté, si le vote explicite d'un ensemble de calculateurs n'est plus capable de fournir un résultat avec un poids *P* supérieur ou égal à 2, l'algorithme engage un second vote, dit **« vote sélectif »**, pour faire intervenir d'une façon indirecte dans le processus de sélection les calculateurs de l'autre ensemble permettant d'avoir le maximum de variantes de logiciel distinctes (*A*, *B* ou *C*), soit 2 ou 3 variantes distinctes. L'objectif est de voir s'il est possible d'obtenir au final un poids *P* au moins égal à deux.

Ainsi, par exemple, si le voteur explicite du côté droit produit un résultat avec un poids *P* inférieur strictement à 2, alors le résultat du vote explicite de l'ensemble 2 (droit) n'est pas utilisable. Et si, du coté gauche, l'un des calculateurs est indisponible (ex. : le calculateur FCCP3), alors l'information utilisée par le processus de choix final est celle émanant du calculateur de l'ensemble 2 (droit) qui présente la même configuration logicielle que le calculateur indisponible de l'ensemble 1 (gauche), donc, ici, le calculateur FCCP3. De même, si les calculateurs FCCP1 et FCCP2 sont indisponibles, ce seront les calculateurs FCCP3 et FCCP4 qui seront pris en compte dans le processus de choix final, au titre de l'ensemble 2.

Enfin, on combine les résultats fournis par les deux ensembles de calculateurs (droit et gauche) pour la sélection finale de l'ordre à envoyer à l'actionneur. Rappelons que l'algorithme de choix est implémenté dans toutes les électroniques locales des actionneurs.

Décrivons maintenant plus en détail encore ces trois types de mécanismes : les deux votes et la sélection.

- **Vote explicite** (voir Figure III.10) : il consiste à déterminer si une majorité de résultats fournis par les entités redondantes sont tels que leurs écarts relatifs des uns par rapport aux autres sont acceptables. Il s'agit d'un vote majoritaire avec seuil *C* classique, qui fournit un couple *R = (M, P)*, où *M* est le résultat voté et *P* le nombre des unités validées par le voteur. *P* désigne un poids ou un facteur de confiance qui indique le nombre d'unités qui sont en accord.

  Exemple : soient, un seuil *C = 1*, et des ordres Ordre1 de valeur *v1 = 16*, Ordre 2 de valeur *v2 = 15*, Ordre 3 de valeur *v3 = 17*. On a alors *R = (16, 3)*.

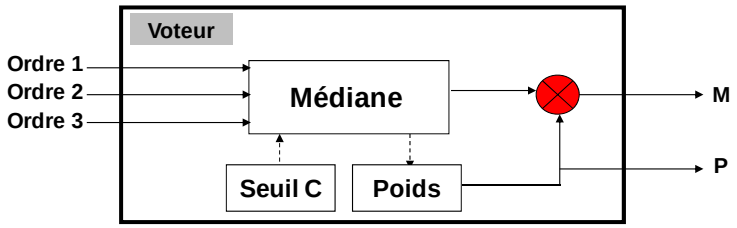


Figure III.10 – Structure du voteur explicite

  Dans l'architecture à vote massif, pour l'étape du vote explicite, la valeur *M* est égale :

  – dans le cas de la loi directe, à la médiane des 3 ordres reçus, qui correspondent aux ordres des 3 calculateurs d'un même côté électrique,
  – dans le cas de la loi normale, il y a 2 ordres reçus (d'un même côté électrique) :
    - lorsqu'on reçoit 2 valeurs valides, la valeur votée est la ½ somme,
    - lorsqu'on reçoit une seule valeur valide, la valeur votée est celle-ci.

  Enfin, le vote explicite est réalisé pour chacun des deux côtés électriques, pour les ordres de la loi normale, et pour les ordres de la loi directe.

- **Vote sélectif** (voir Figure III.11**)** : il est engagé uniquement sous certaines conditions, notamment en fonction du poids *P*. En fait, le vote sélectif doit être engagé dans le cas où le vote explicite d'un côté (ou site) n'est plus capable de fournir un résultat avec un poids *P* supérieur ou égal à 2. Le vote sélectif élimine les résultats calculés par des calculateurs qui fonctionnent avec des logiciels ayant des signatures valides (qui sont disponibles) pour le voteur explicite du côté opposé selon une logique bien précise.

  Le Tableau III-3, donne un exemple de cette logique dans le cas où P < 2 du côté droit.

Disponibilité logicielle (A, B, C) site gauche	**Sortie du voteur du site droit vers la sélection finale (envoi du complément de ce qui est indisponible de l'autre site)**
(1, 1, 1)	Ordre vide
(1, 1, 0)	Ordre simple (C)
(1, 0, 1)	Ordre simple (B)
(0, 1, 1)	Ordre simple (A)
(0, 0, 1)	Ordre double (A, B)
(0, 1, 0)	Ordre double (A, C)
(1, 0, 0)	Ordre double (B, C)

Tableau III-3 – Exemple de logique d'établissement des validités logicielles

Une entrée est dite valide lorsqu'elle est disponible (mais pas forcément correcte). La signification des valeurs booléennes est :

– 1 est utilisée pour un logiciel disponible,
– 0 est utilisée pour un logiciel indisponible.

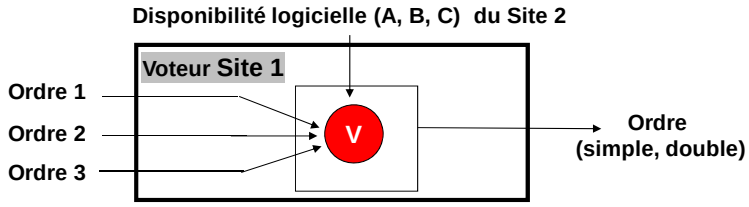


Figure III.11 – Structure du voteur sélectif

- **Sélection finale :** enfin, la dernière étape de vote massif consiste à prendre en compte les valeurs votées provenant des deux ensembles de la façon suivante :

  1) Si pour **chaque ensemble**, l'ordre n'est transmis que par un seul calculateur (donc deux ordres au total) avec un poids *P* supérieur au égal à 2, et si les ordres des deux ensembles ont le même poids, alors l'ordre final est la demi-somme des ordres issus de chaque ensemble.
     Sinon l'ordre final est celui de l'ensemble ayant produit l'ordre avec *P* le plus élevé.
  2) Si pour **un des ensembles**, l'ordre est transmis par deux calculateurs, et si pour l'autre ensemble l'ordre n'est transmis que par un seul calculateur, alors :
     – si le poids *P* des deux ensembles est strictement inférieur à 2, le résultat final est la médiane des trois valeurs avec au moins deux valeurs devant être cohérentes,
     – si le poids *P* d'un des deux ensembles est déjà égal à 2, l'autre ensemble est ignoré,
  3) Sinon l'ordre est indisponible (la sélection finale ne fournit pas d'ordre).

L'ordre, final et unique, ainsi déterminé va être appliqué et exécuté par l'actionneur pour faire déplacer la surface mobile de la quantité désirée, l'actionneur choisissant la loi normale si l'ordre pour la loi normale est disponible, ou la loi directe si l'ordre pour la loi normale est indisponible mais que l'ordre pour la loi directe est disponible. Si aucun ordre n'est disponible, l'actionneur se désactive (mode amorti).

Le Tableau III-4 résume les situations possibles pour la logique de sélection finale.

Coté 2 / *Coté 1*	**Ordre simple**	**Ordre double**
Ordre simple	- 1/2 somme si poids de l'ordre 1 = poids de l'ordre 2 - Sélection de l'ordre avec le poids le plus élevé	- Si poids (ordre simple) >= 2, alors ignorer ordre double - Sinon vote médiane des 3 valeurs avec au moins 2 valeurs cohérentes parmi les 3
Ordre double	- Si poids (ordre simple) >= 2, alors ignorer ordre double - Sinon vote médiane des 3 valeurs avec au moins 2 valeurs cohérentes parmi les 3	- Ordre indisponible

Tableau III-4 - Logique de sélection finale

c) Reconfiguration

Le principe général et les règles de la reconfiguration logicielle ont été évoqués dans le chapitre I, au I.2.1.4. Dans l'architecture à vote massif, la reconfiguration est pilotée par les actionneurs : chaque actionneur qui a détecté la panne de 2 calculateurs primaires exécutant le même logiciel envoie une requête de reconfiguration logicielle à l'un des 2 autres calculateurs primaires restant valides (les deux cotés confondus). Le choix du calculateur est prédéfini et arbitraire, de telle sorte que tous les actionneurs ont la même requête.

Le calculateur à reconfigurer procède effectivement à sa reconfiguration lorsqu'il reçoit une requête en reconfiguration cohérente de la part d'un certain nombre d'actionneurs.

Par exemple, si les FCCP1 et FCCP3, fonctionnant avec le programme *A*, tombent en panne, une requête de reconfiguration est envoyée au FCCP2 (fonctionnant avec le programme *B*) pour que celui-ci soit reconfiguré et fonctionne avec le programme *A*, afin que la loi normale soit toujours obtenue par 2 calculateurs respectant le principe de la dissimiliarité logicielle.

De même, si les FCCP2 et FCCP4, fonctionnant avec le programme *B*, tombent en panne, une requête de reconfiguration est envoyée au FCCP1 (fonctionnant avec le programme *A*) pour que celui-ci soit reconfiguré et fonctionne avec le programme *B*.

d) Logique de validation

Chaque FCRM peut valider ou invalider les calculateurs en comparant la valeur qu'il sélectionne avec celle envoyée par chaque calculateur. Le FCRM écarte tout calculateur invalide de son prochain processus de vote et il lui envoie un acquittement négatif.

e) Commande et surveillance des gouvernes

Rappelons que la commande des gouvernes correspond à la transmission de l'ordre d'asservissement du calculateur vers les gouvernes. L'asservissement de l'actionneur est piloté et contrôlé par les FCRM (boucle d'asservissement en locale). Certains actionneurs peuvent être pilotés par deux FCRM : FCRM actif en mode actif, et FCRM passif en mode amorti. Au démarrage, les deux FCRM s'initialisent en mode amorti. Les calculateurs peuvent alors commander le passage en mode actif d'un FCRM via son RCCB. Le FCRM en secours surveille le FCRM actif et il passe en mode actif dès que ce dernier se désengage.

III.2.3 Les relais d'alimentation : les RCCB

Toute électronique locale des actionneurs doit nécessairement être alimentée électriquement. Pour des raisons de sûreté, cette alimentation est elle aussi gérée par un équipement, un module « relais d'alimentation », lui-même piloté par les calculateurs FCCP et FCCS.

Ce module, nommé *RCCB* (*Remote Control Circuit Breaker*)[2], pourrait être un composant optionnel dans les architectures que nous proposons, mais il permet une sécurité supplémentaire au delà des exigences de sécurité exigées pour les commandes de vol.

III.2.3.1. Architecture matérielle et logicielle

L'alimentation des électroniques locales est pilotée par les calculateurs (primaires et secondaires) via leur RCCB, qui permet de contrôler l'alimentation électrique des voies COM et MON de chaque actionneur. Chaque FCRM est équipé d'un seul RCCB. Les RCCB sont eux-mêmes alimentés par les systèmes électriques de l'avion (coté gauche ou coté droit).

[2] Ce nom RCCB est issu du nom du relais actuel pour l'A380, alors que pour l'A350, on parle de *SSPC* (*Solide System Power Control*).

Ce relais d'alimentation est plus intelligent que les relais classiques actuels, car il implémente une fonction logicielle : il peut répondre automatiquement aux alertes provenant des calculateurs via le réseau de communication pour couper l'alimentation de son FCRM selon une logique de vote minimale élémentaire.

À la mise sous tension, les RCCB alimentent les FCRM (voies COM et MON). En fonctionnement normal, tous les calculateurs (primaires et secondaires) demandent à tous les RCCB d'alimenter tous les FCRM. Le RCCB, lorsqu'il est alimenté en électricité (sauf panne électrique totale) et qu'il reçoit des ordres d'activation, envoie au FCRM un signal représentatif de l'énergie électrique qu'il lui fournit.

III.2.3.2. Description fonctionnelle

1) Interface système

Les calculateurs utilisent le réseau de communication pour communiquer avec les RCCB, alors que les RCCB sont reliés à leurs FCRM via des liaisons directes. Les RCCB consomment et produisent des flux de données décrits dans le Tableau III-5. Quant à l'alimentation électrique des RCCB, elle est fournie directement par l'un des systèmes électriques de l'avion en fonction de leur localisation géographique.

Flux de données	*Description*	*Type*	*Flux entrant (source)*	*Flux sortant (destinataire)*
Ordres calculateurs	Booléens d'activation ou de désactivation du FCRM venant des calculateurs.	Trame de 6 booléens	Calculateurs	
Alimentation_FCRM	En fonction des ordres d'activation envoyés par les calculateurs et sa logique fonctionnelle, le RCCB alimente ou coupe d'alimentation de son FCRM.	Tension		FCRM

Tableau III-5 - Flux de données d'un RCCB intelligent

2) Architecture fonctionnelle

Par défaut, un calculateur qui veut activer un FCRM envoie un booléen positif (1) au RCCB de ce FCRM. Pour activer un FCRM, il suffit d'avoir deux booléens positifs de n'importe quels calculateurs actifs. Cette logique de vote minimale permet d'éviter la perte de tous les actionneurs suite aux défaillances communes qui peuvent affecter simultanément tous les calculateurs primaires (ex. : problèmes matériels). Dans ce cas, un vote majoritaire n'est donc plus fiable.

Chaque calculateur détectant un désaccord entre deux unités COM et MON d'un actionneur ou un comportement erroné de cet actionneur, considère celui-ci en panne et lui envoie une consigne de non-opération (coupure d'alimentation électrique) via son RCCB auquel il envoie un booléen nul (0). Cependant, et selon un mécanisme de décision élémentaire, le RCCB n'appliquera cette consigne de désactivation que lorsqu'elle est consolidée par les autres calculateurs.

Enfin, dans le cas où un calculateur est désengagé ou qu'il ne demande pas d'activation, ce calculateur n'émet rien. Le RCCB interprète cette absence de l'ordre d'activation en provenance d'un calculateur, comme une requête de désactivation de son FCRM.

La Figure III.12 donne le schéma fonctionnel d'un RCCB.

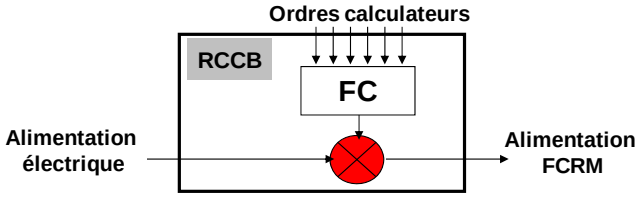


Figure III.12 - Schéma fonctionnel d'un relais électrique intelligent

3) Principe de fonctionnement et logiques spécifiques (niveau RCCB)

La logique de vote minimale élémentaire est une fonction de comptage des ordres de désactivation envoyés par les calculateurs. En fonctionnement normal, avant la première panne de calculateur, on dispose de 6 calculateurs. Il faut donc 5 ordres de non-activation pour couper l'alimentation d'un FCRM.

Chaque RCCB maintient un compteur ou une fonction de comptage *FC* à 6 entrées *fci*, chacune correspondant à un calculateur *i*. Initialement, ce compteur est mis à 6 (nombre initial de calculateurs actifs). Ce qui correspond à un état initial actif pour le FCRM. Chaque fois qu'un calculateur demande à désactiver un actionneur, son RCCB décrémente son compteur, et quand la valeur du compteur est égale à 1, il coupe l'alimentation de son FCRM (voie COM et voie MON). La description de l'algorithme de la fonction *FC* est comme suit :

```
FC = 6                        / Initialisation du compteur /
Pour i de 1 à 6 faire         /Calcul /
  Si  fci = 0 (entrée calculateur i) /
        Alors FC = FC – 1
Si  FC =1 Alors alimentation_FCRM = ko
```

On note :

État calculateur i	Entrée *fci*
Calculateur en panne	0
Calculateur actif avec demande d'activation	1
Calculateur actif avec demande de désactivation	0

III.2.4 Réseaux de communication

Concernant l'architecture réseau, la thèse se limite à l'utilisation des moyens de communication avionables qui seront utilisés à court terme ou à moyen terme par Airbus (AFDX, Micro-AFDX, bus 1553). Dans ce chapitre, le réseau de communication a été considéré comme une boîte noire au travers de laquelle circulent toutes les informations des calculateurs vers les actionneurs et des actionneurs vers les calculateurs. La description de quelques pistes d'architectures réseaux support des nouvelles architectures qu'on propose dans le cadre de la thèse, sont présentées en annexe 1. Il y sera question des architectures réseaux minimales satisfaisant un ensemble de contraintes (objectifs de sécurité, ségrégation des routes physiques) pour l'échange d'information entre différents composants du système.

III.3 PREMIERE ÉVALUATION : MODES DE DEFAILLANCES ET MECANISMES DE DETECTION

En comparaison avec une architecture classique avec deux calculateurs PRIM et deux calculateurs SEC (cf. chapitre I), l'architecture à vote massif proposée, avec ses 6 calculateurs simplex, semble théoriquement encore plus robuste et plus disponible pour trois raisons.

1) Grâce au principe de communication de type « broadcast », chaque calculateur communique avec tous les actionneurs, et donc tous les actionneurs disposent des ordres de tous les calculateurs. Ainsi, la perte des calculateurs du côté gauche ou droit reste transparente par rapport aux actionneurs.
2) Grâce au mécanisme de vote massif au niveau des actionneurs, le fonctionnement erroné d'un calculateur primaire, ou la perte totale des calculateurs primaires (sous MMEL), ou le fonctionnement erroné de plus de trois calculateurs (extrêmement improbable), ou la perte d'un système électrique (gauche ou droit) ne perturbe pas le fonctionnement des actionneurs et ne conduit à aucune perte des performances de l'avion.
3) Enfin, la reconfiguration logicielle augmente la disponibilité de la loi normale et du système global.

Les cas et le risque de déconnexion intempestive des calculateurs sont minimisés grâce à l'utilisation des calculateurs simplex à la place des calculateurs de type COM/MON qui faisaient appel à des surveillances par comparaison et des logiques d'auto-déconnexion en cas de désaccord. Cette utilisation confirme l'intérêt d'une architecture de type vote massif dans laquelle de telles logiques sont exclues.

Précisons que les cas d'embarquement des surfaces ne sont pas considérés, car on les suppose couverts par l'architecture COM/MON mise en place dans les FCRM, en plus des surveillances supplémentaires réalisées par les relais d'alimentation intelligents (RCCB).

Mais la perte ou le fonctionnement erroné des composants de l'architecture sont à considérer.

Au final, il y a plusieurs cas intéressants pour vérifier la robustesse d'une architecture en présence de fautes ou de pertes. Ces cas sont résumés dans le Tableau III-6 qui suit.

Modes de défaillances	Mécanismes de détection	Effets/risques
Composant : CALCULATEUR PRIMAIRE		
Fonctionnement erroné non détecté par les surveillances internes du calculateur	Masqué par le vote massif au niveau des actionneurs et détecté par les FCRM	Le fonctionnement erroné de plus de 2 calculateurs (extrêmement improbable) implique la perte de la loi normale. Sinon, aucun effet sur le comportement global du système
Perte	Surveillances internes du calculateur ou perte d'émission sur le réseau	La perte de plus de 2 calculateurs implique la perte de la loi normale. Sinon, aucun effet sur le comportement global du système
Émission de donnés erronées sur le réseau de communication	Contrôle d'intégrité (CRC applicatif) au niveau du réseau ou des actionneurs	Aucun effet

Modes de défaillances	Mécanismes de détection	Effets/risques
Composant : CALCULATEUR SECONDAIRE		
Acquisition erronée (retour FCRM)	Détecté par le calculateur et masqué par le vote majoritaire (au niveau du calculateur)	Aucun effet
Fonctionnement erroné non détecté par les surveillances internes du calculateur	Détecté par les FCRM et masqué par le vote massif (au niveau du FCRM)	Aucun effet
Perte	Surveillances internes du calculateur ou perte d'émission sur le réseau	Aucun effet
Émission de donnés erronées sur le réseau de communication	Contrôle d'intégrité (CRC applicatif) au niveau du réseau ou des actionneurs	Aucun effet
Composant : RELAIS D'ALIMENTATION (RCCB)		
Perte	Détecté par les calculateurs (perte des retours FCRM)	Perte de l'actionneur
Composant : ÉLECTRONIQUE LOCALE (FCRM)		
Fonctionnement erroné non détecté par la surveillance COM/MON du FCRM	Vote majoritaire au niveau des actionneurs	Extrêmement improbable Aucun effet
Perte		Aucun effet
Fonctionnement erroné de la voie COM	Détecté par la surveillance de la voie MON	Perte de contrôle de l'actionneur
Composant : RÉSEAUX DE COMMUNICATION		
Perturbation électrique sur les fils	Contrôle d'intégrité (CRC applicatif) au niveau du calculateur ou des actionneurs	Aucun effet
Perte	Utilisation des données du réseau secondaire	Aucun effet (réseau redondant)

Tableau III-6 – Récapitulatif des modes de défaillances considérés, des mécanismes de détection, et des risques et effets associés

III.4 CONCLUSION

Ce chapitre nous a tout d'abord permis de décrire l'architecture vote massif. Il s'agit d'une architecture distribuée, multi-maîtres et tolérante aux fautes, avec des différences fondamentales par rapport à l'existant présenté dans l'état de l'art. Les différences viennent de nouvelles répartitions des fonctions intelligentes entre l'avionique centrale (calculateurs CDV) et l'avionique déportée (électroniques locales des actionneurs), avec pour avantage de consommer une moins grande quantité de ressources par rapport aux architectures conventionnelles tout en satisfaisant un même objectif de sécurité/disponibilité.

On a montré que le fait de considérer toutes les possibilités offertes par des actionneurs intelligents avec des capacités à mémoriser des informations et de les traiter, ainsi que par des communications numériques, permet d'optimiser l'architecture globale du système de commandes de vol.

Le chapitre suivant sera consacré à la modélisation de l'architecture proposée en vue d'une validation par simulation en termes de satisfaction des besoins et de respect des contraintes de robustesse et de sûreté de fonctionnement.

On propose une première modélisation de l'architecture sous Simulink pour montrer la robustesse de l'architecture et une deuxième modélisation basée sur le langage AltaRica pour la sûreté de fonctionnement.

Chapitre IV - MODELISATION ET VALIDATION DE L'ARCHITECTURE A VOTE MASSIF

La modélisation et la simulation sont des activités de plus en plus répandues lors de la conception de systèmes complexes et critiques tels que ceux embarqués dans des avions, et en particulier pour le système de commandes de vol. La vérification et la validation des architectures lors des phases préliminaires de conception passent nécessairement par une phase de modélisation. Cette modélisation nous permet de mieux décrire, analyser et de simuler l'architecture système en fonction des exigences du système et des objectifs de la validation.

Parmi les exigences du système de commandes de vol, les exigences de robustesse et celles liées à la démonstration de sûreté de fonctionnement sont essentielles. Rappelons que le système de CDVE a principalement deux types d'objectifs de sûreté de fonctionnement :

1) l'intégrité (les calculateurs ne doivent pas envoyer de commandes erronées non détectées vers les actionneurs)
2) la disponibilité (le système doit présenter un haut niveau de disponibilité).

L'architecture Airbus trouve ses origines dans la suite de l'évolution architecturale des systèmes analogiques vers les systèmes numériques, puis vers les calculateurs numériques.

Rappelons aussi que le système de CDVE Airbus est un système *GALS (Globalement Asynchrone Localement Asynchrone)* [Tran 2008]. En effet, les calculateurs et les FCRM sont des sous-systèmes complètement autonomes et synchrones, mais ils fonctionnent par échantillonnage dans un environnement asynchrone (absence d'échelle globale de temps).

La communication par échantillonnage en présence de dérives d'horloges des composants du système peut introduire des déphasages temporels entre les différents composants redondants ou coopérants de l'architecture, et donc poser des problèmes de robustesse [Caspi 2003].

L'objectif de ce chapitre est de présenter les travaux réalisés sur la vérification de la conformité de l'architecture à vote massif (décrite au chapitre précédent) avec les exigences de sécurité et de robustesse fixées pour les systèmes de CDVE Airbus. Après avoir présenté les besoins de validation et les outils de modélisation retenus, nous décrivons les modèles d'architectures que nous avons réalisés, et les scénarios à vérifier, sur lesquels nous nous sommes appuyés pour valider l'architecture à vote massif.

IV.1 PRINCIPES DE VALIDATION ORIENTEE « SECURITE »

IV.1.1 Processus actuel d'analyse de la sûreté de fonctionnement

IV.1.1.1 Démarche

Durant la conception, en vue d'obtenir la certification, des analyses sont menées afin de chercher tout ce qui pourrait conduire l'avion dans une situation dégradée pouvant éventuellement porter atteinte à l'équipage, aux passagers ou à toute personne dans le périmètre de l'appareil. Ces analyses sont des ***analyses de sûreté de fonctionnement***, aussi dénommées ***« analyses de sécurité »***, formulation retenue dans la suite de cette partie.

Le processus d'évaluation de la sécurité d'un avion a pour but de vérifier la conformité de la conception avec les exigences de sécurité spécifiées par les autorités [Bernard 2009]. L'évaluation de la sécurité peut être qualitative et quantitative, et se décompose en quatre types définis ci-après. La Figure IV.1 montre les relations entre ces types, ainsi que leur positionnement dans le processus de développement des systèmes aéronautiques.

1) *L'évaluation des risques fonctionnels* ou *Functional Hazard Assessment* (***FHA***) : elle étudie les fonctions avion et système pour déterminer les défaillances fonctionnelles possibles et classifie les risques associés à des configurations de panne spécifiques, ou ***Failure Condition (FC)***. Les résultats de la *FHA* sont les données initiales de la *SSA*.
2) *L'évaluation de la sécurité des systèmes* ou *System Safety Assessment* (***SSA***) : elle vérifie que le système développé satisfait les exigences formulées dans la FHA.
3) L'analyse des *Modes de Défaillance et de leurs Effets* (***AMDE***) [Jordan & Marshall 1972], ou *Failure Mode and Effect Analysis* (***FMEA***) : c'est une démarche classique en sûreté de fonctionnement qui consiste à identifier les modes de défaillance de chaque composant d'un système et à analyser leurs effets sur les autres composants du système. D'un niveau de détail très fin, cette analyse est généralement associée dans l'aéronautique à un document de synthèse appelé *Failure Mode and Effect Summary* (***FMES***). Elle permet de lister les pannes des composants du système et donne leur taux de défaillance.
4) *L'analyse des causes communes* ou *Common Cause Analysis* (***CCA***) : complément des *SSA*, elle vise à réduire les risques liés à des causes communes au système ou à plusieurs systèmes. Elle traite des aspects d'installation des systèmes, de facteurs humains, etc.

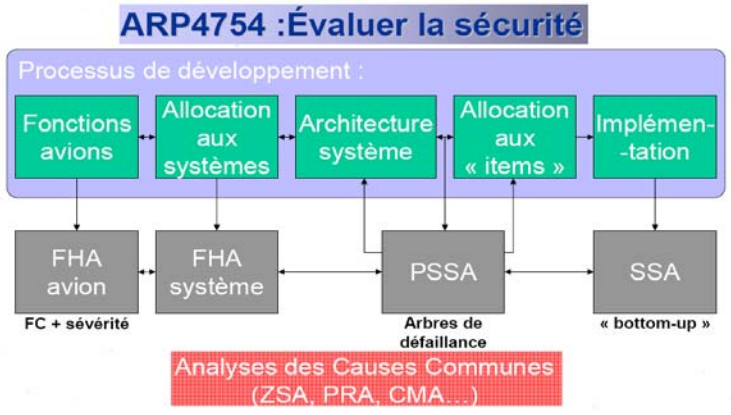


Figure IV.1 - Processus d'évaluation de la sécurité utilisé dans l'aéronautique

IV.1.1.2 Techniques

Afin d'étudier les pannes simples ou les combinaisons de défaillances conduisant à chaque « Failure condition » (FC) identifiée dans la FHA, l'ARP 4761 propose trois techniques d'analyse descendante :

1) l'arbre de défaillances ou *Fault Tree (**FT**)*,
2) le diagramme de dépendance ou *Dependence Diagram (**DD**)*,
3) l'analyse markovienne ou *Markov Analysis (**MA**)*.

Afin de garantir une analyse indépendante de la conception (c'est-à-dire, sans influence du concepteur), l'analyse de sécurité est menée par un ingénieur spécialisé qui réalise le **FT/DD**. Chaque FT/DD doit ensuite être validé par le concepteur afin de s'assurer que le fonctionnement du système pris en compte par l'ingénieur spécialiste est correct : chaque « Failure condition » nécessite donc un FT/DD.

Les arbres de défaillances et les diagrammes de dépendance sont très répandus dans l'aéronautique car ils sont particulièrement lisibles à la fois pour les ingénieurs et par les autorités de certification. Néanmoins, ils présentent un certain nombre de limitations :

1) Dans le cas d'un système très complexe, la taille d'un arbre de défaillance peut être particulièrement grande : au-delà d'une certaine profondeur, l'arbre devient difficilement lisible.
2) La représentation du comportement d'un système est statique : la chronologie des défaillances dans une coupe minimale n'est pas prise en compte.
3) Toute modification de l'architecture d'un système nécessite la vérification et, éventuellement, la mise à jour manuelle de tous ses FT/DD.

Pour ce qui est des graphes de Markov [Bouissou & Bon 2003], ils nous semblent moins répandus dans l'aéronautique. Cela s'explique probablement par le fait que réaliser un graphe de Markov manuellement n'est réaliste que pour de très petits systèmes. La taille d'un tel graphe augmente très vite.

IV.1.2 Nouvelle approche d'analyse de sécurité fondée sur les modèles

La section précédente a présenté un rapide tour d'horizon des méthodes classiques utilisées pour les analyses de sûreté de fonctionnement pour le système des CDVE chez Airbus. Ces techniques permettent de réaliser des analyses qualitatives et quantitatives. Malheureusement, bien souvent, chaque type d'analyse nécessite souvent sa propre forme de modélisation du système étudié, ce qui donne autant de modélisations différentes que de types d'analyses.

Le langage AltaRica [Altarica], présenté dans la sous-section suivante, est une forme de réponse à ce problème. AltaRica est un langage formel de modélisation dont le but est de pouvoir servir de base à divers outils performants de l'atelier AltaRica. Ces outils permettent de réaliser une grande partie des analyses de sûreté de fonctionnement (que nous avons présentées) à partir du seul modèle AltaRica du système à valider. La société Dassault Aviation a, en 2005, démontré que les commandes de vol de son Falcon 7X étaient conformes aux exigences des autorités à l'aide de modèles réalisés en langage AltaRica. Ce langage, développé pour supporter les analyses de sûreté de fonctionnement des systèmes, fait l'objet d'expérimentations internes par la société Airbus depuis 4 ans, suite à l'implication d'Airbus dans plusieurs projets [Akerlund *et al.* 2006, Bieber *et al.* 2004, Pagetti 2004, Kehren *et al.*2004, Kehren 2005, Bernard 2009] ; l'objectif était de proposer un environnement de spécification /modélisation unifié pour les concepteurs, et de simplifier les étapes de modélisation de la sûreté de fonctionnement.

Les capacités intéressantes d'AltaRica nous ont poussés à l'utiliser pour la modélisation et la validation des architectures que nous avons proposées dans le cadre de la thèse.

Néanmoins, AltaRica n'est pas le seul langage dédié à la sûreté de fonctionnement pour les systèmes critiques. Bien qu'il soit au cœur de ce chapitre, il existe d'autres langages et environnements qui sont orientés sur la modélisation fonctionnelle et dysfonctionnelle d'un système, et qui sont très proches de la problématique décrite dans l'introduction. Sans prétendre être exhaustif, on peut citer les travaux de [Rugina 2007, Feiler & Rugina 2007] et AADL, le langage Figaro [Bouissou & Seguin 2006] et l'environnement SyRelAn [Raksch *et al.* 2007].

IV.1.2.1 AltaRica et plate-forme associée

1) Langage

Le langage AltaRica est un langage formel développé au LaBRI (Laboratoire Bordelais de Recherche en Informatique), conjointement avec des industriels (dont Dassault Aviation, Elf Aquitaine) dans l'objectif de réaliser des modélisations à la fois fonctionnelles et dysfonctionnelles, utiles pour les analyses de sûreté de fonctionnement de systèmes complexes. La vue dysfonctionnelle est permise grâce à l'introduction d'événements modélisant les défaillances de chaque composant de système définies dans la *FMES* (*Failure Mode and Effect Summary).*

AltaRica est un langage à la fois formel et graphique grâce aux ateliers **OCAS** (Outil de Conception et d'Analyse Système) développé par Dassault, et **SIMFIA** développé par EADS APSYS, ateliers qui facilitent la programmation, la simulation ainsi que l'analyse des modèles. D'un point de vue pratique, l'outil **OCAS** offre une prise en main aisée en guidant l'utilisateur lors de la création d'un composant.

Pour toutes ces raisons, l'outil OCAS et Altarica ont été retenus pour effectuer des analyses de sécurité dans le cadre de la thèse.

2) Syntaxe du langage

Un composant AltaRica est appelé nœud. Un modèle est un ensemble de nœuds interconnectés. La description d'un nœud AltaRica est composée de deux parties :

<u>*Une partie déclarations :*</u>

- *State* : variables internes dites « d'état »,
- *Flow* : variables dites « de flux » (entrées, sorties variables locales),
- *Event* : événements impactant les variables d'état (erreur, perte, reconfiguration).

<u>*Une partie définitions :*</u>

- *Transitions* : pour chaque événement déclaré, on définit des gardes (formule booléenne précisant dans quelle situation un événement peut se produire) et les conséquences internes (modifications des affectations des variables d'état),
- *Assertions* : formules de calcul de chaque variable de sortie en fonction de l'état courant et des variables d'entrée.

3) Illustration

Afin d'illustrer la syntaxe présentée précédemment, considérons un calculateur CPU qui contrôle une servo-commande et surveille son bon fonctionnement. Tout d'abord, pour traiter les différents modes de panne à considérer, le type `StatusType` est défini comme l'ensemble énuméré {`correct`, `erroné`, `perdu`}. Pour modéliser l'état interne du composant, la variable d'état `Status` est définie, ayant pour valeur nominale `Correct`. Le composant dispose de deux entrées (l'alimentation et le retour de capteur), et d'une sortie. Ces éléments sont donnés dans la Figure IV.2.

State	Type	Valeur Initiale
Status	StatusType	Correct

a)

Flow	Type	Direction
Resource	Boolean	In
Feedback	StatusType	In
Ordre	StatusType	Out

b)

Figure IV.2 - Variables du composant CPU : a) variable d'état ; b) variables de flux

On peut définir des modes de défaillances génériques, comme la *Perte* et *l'Erreur* du calculateur. Pour des analyses quantitatives, la probabilité d'occurrence de chaque événement doit être fixée, comme le montre la Figure IV.3.a). En termes de transitions, l'événement `Perte` peut se produire lorsque le composant est dans un état correct ou erroné, tandis que l'événement `Erreur` ne peut se produire qu'en situation nominale correcte (cf. Figure IV.3.b).

Event	Loi	Paramètres
Perte	Constant	1e-4
Erreur	Constant	1e-5

a)

Event	Garde	Affectations
		Status
Perte	Status != perdu	perdu
Erreur	Status = correct	erroné

b)

Figure IV.3 - a) Événements du composant CPU ; b) Transitions du composant CPU

Enfin, pour les assertions, le composant doit transmettre :

1) un ordre correct (***valeur = correct***) lorsque : il est alimenté (***Resource***) et non défaillant (***Status = correct***), et que le retour de capteur lui indique que la servocommande fonctionne correctement (***Feedback = correct***),
2) un ordre erroné lorsque : il est alimenté (***Resource***) et en cas d'erreur interne ou de retour de capteur erroné,
3) sinon il ne transmet pas d'ordre.

Et en faisant le choix d'assimiler l'absence d'ordre à la valeur `perdue`, le calcul de la variable `Ordre` est présenté dans la Figure IV.4 :

Assertion	Case	Valeur
Ordre	Resource and (Status = correct and Feedback = correct)	correct
	Resource and (Status = erroné or Feedback = erroné)	erroné
	Else	perdu

Figure IV.4 - Assertions du composant CPU

4) Sémantique formelle

Un modèle AltaRica est représenté par un automate (ou système de transitions) de tous les scénarios possibles. Chaque nœud de l'automate est un état, composé d'une valeur pour chaque variable d'état et de flux. L'état initial est défini d'abord par les valeurs initiales des variables d'état des composants, puis par les assertions qui fixent les valeurs des variables de sortie. Ces valeurs sont propagées aux composants connectés pour définir les valeurs de leurs variables d'entrée. Un scénario d'un modèle est donc un chemin dans l'automate qui débute à l'état initial puis évolue d'état en état.

5) Simulation et Analyse

a) Simulation interactive

L'outil OCAS permet de réaliser mais aussi d'exploiter des modèles en langage AltaRica et propose une animation de la représentation graphique faisant évoluer, après chaque tirage d'événement, les icônes des composants et la couleur de chaque lien. La simulation interactive permet de jouer manuellement des scénarios de pannes, afin de valider/comprendre les réactions du système et les reconfigurations éventuelles.

b) Génération des modèles de sûreté de fonctionnement

Pour une simulation exhaustive, l'outil OCAS permet la génération automatique des modèles de sûreté de fonctionnement (arbre de défaillance, modèles stochastiques, séquences,…) pour des évaluations (fiabilité, disponibilité,...) quantitatives ou qualitatives.

- La génération automatique d'arbres de fautes s'appuie sur un calcul exhaustif de toutes les combinaisons de défaillances conduisant le modèle de son état initial nominal à la situation à observer spécifiée. Les combinaisons obtenues sont présentées sous forme d'arbre de fautes.
- La génération automatique de coupes/séquences minimales se fait en procédant à une analyse exhaustive du modèle, similaire à la génération d'arbres. Les combinaisons de pannes peuvent être présentées sous forme textuelle :
  - coupes minimales : les défaillances sont présentées dans un ordre arbitraire à l'intérieur de chaque combinaison,
  - séquences minimales : chaque scénario présente les défaillances par ordre chronologique.

c) Analyse quantitative

À partir d'un arbre de fautes ou des coupes minimales associées à une « Failure Condition » et en précisant la probabilité correspondant à chaque défaillance, il est possible de procéder à une analyse quantitative d'un modèle en calculant la probabilité de chaque coupe et/ou la probabilité totale de la « Failure Condition ».

IV.2 VALIDATION ORIENTEE SECURITE SOUS L'ENVIRONNEMENT OCAS/ALTARICA

IV.2.1 Cas d'étude : commande en tangage

Parmi les cas d'étude possibles pour valider l'architecture à vote massif, nous avons choisi le système de commande en tangage, en se basant sur une description simplifiée. D'autres systèmes Airbus ont fait l'objet d'un prototypage en AltaRica comme dans [Bernard 2009].

Le contrôle de l'avion suivant l'axe de tangage s'effectue en contrôlant trois surfaces mobiles horizontales à l'arrière de l'appareil : 2 *gouvernes de profondeur* et 1 *plan horizontal réglable* (***PHR*** ou ***THSA*** : Trimmable Horizontal Stabilizer) (voir Figure IV.5). Les gouvernes de profondeur assurent la commande longitudinale à court terme, alors que le PHR (THSA) assure la commande longitudinale à long terme permettant de conserver un braquage de la profondeur en moyenne nulle, malgré les variations de centrage et les variations de vitesse. En résumé, le PHR équilibre l'avion, et les gouvernes de profondeur permettant de le commander autour de cette position d'équilibre.



Figure IV.5 - Gouvernes de contrôle en tangage

Dans le cadre de la thèse, nous considérerons une configuration avec deux actionneurs en parallèle pour chacune des surfaces (elevator gauche, elevator droit et PHR). Chaque actionneur a deux modes de fonctionnement : un mode normal (servocommande active) et un mode amorti (servocommande passive). En conditions normales, une servocommande est en mode normal, l'autre en mode amorti. Chaque actionneur est doté d'une électronique locale ou FCRM comportant deux unités (ou voies) fonctionnellement équivalentes telles que décrites dans le chapitre III. On considère aussi deux cotés électriques (BusBar_1 et BusBar_2) avec la distribution décrite par la Figure IV.6.

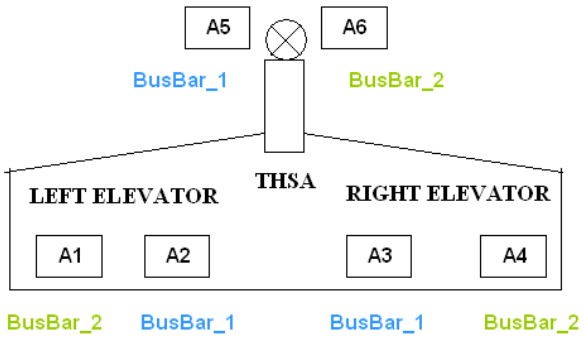


Figure IV.6 - Alimentation électrique des gouvernes de contrôle en tangage

IV.2.2 Principe de fonctionnement : rappel

Selon l'architecture à vote massif, les calculateurs sont désormais des unités « simplex » (c'est-à-dire à une seule voie), tolérantes aux fautes grâce à leurs surveillances internes. Et la comparaison et la sélection des ordres des calculateurs ne s'effectuent plus au niveau des calculateurs, mais au niveau de l'électronique locale de chaque actionneur.

Le principe de contrôle des actionneurs est le suivant : tous les calculateurs calculent les lois de pilotage ainsi que l'ensemble des ordres gouverne pour tous les actionneurs. Ils transmettent leurs ordres gouverne via le réseau de communication vers tous les actionneurs, ceci indépendamment de la validité des ordres calculés par chaque calculateur. Les deux unités COM et MON de chaque actionneur effectuent des opérations de vote afin d'obtenir un ordre valide, consolidé à partir de l'ensemble des ordres indépendants qu'elles ont reçus. Dans cette partie du mémoire, le réseau de communication a été considéré globalement comme une boîte noire. Les travaux de modélisation et de validation ont porté principalement sur les calculateurs, les FCRM et les actionneurs.

IV.2.3 Exigences de sécurité

Ces exigences pour le système de CDVE sont généralement de la forme :

1) la Failure Condition <FC> est classée <sévérité>,
2) la probabilité doit être inférieure à <objectif quantitatif>,
3) la FC ne doit pas être atteinte en moins de <objectif qualitatif> défaillances ».

Les correspondances entre « sévérité » et « objectif » pour le contrôle en tangage sont illustrées dans le Tableau IV.1:

Evénement redouté (FC)	*Objectif qualitatif*	*Sévérité*	*Objectif quantitatif*
Embarquement PHR ou profondeur (elevator gauche ou droit)	2	Catastrophique	10-9/hv
Perte de la commande de tangage (perte PHR et 1 profondeur)	2	Catastrophique	10-9/hv
Perte de la commande de tangage (perte de deux gouvernes de profondeur)	2	Catastrophique	10-9/hv
Passage en loi directe	1	Majeur	10-7/hv

Tableau IV.1 - Sévérité des configurations de panne pour le contrôle en tangage

Rappelons que notre objectif, en termes de sûreté de fonctionnent, est la vérification de la conformité de l'architecture à vote massif avec les exigences de sécurité et de disponibilité des architectures actuelles, et ce malgré la réduction du nombre des calculateurs et la redistribution de l'intelligence entre les calculateurs et les actionneurs.

IV.2.4 Modélisation en Altarica

Avant d'obtenir les séquences utiles pour l'analyse de sécurité, il faut élaborer un modèle Altarica du système et de ses exigences de sécurité selon l'architecture à vote massif. La première étape consiste à recenser les composants à modéliser, ce qui revient à lister les composants nécessaires au fonctionnement du système. Chaque type d'élément correspondra à un nœud Altarica.

La seconde étape consiste, pour chaque composant, à lister les défaillances potentielles ainsi que les informations échangées avec l'extérieur : ordres reçus/émis, informations transmises et énergie reçue. Toutes ces informations vont définir les transitions d'une part, et les entrées/sorties d'autre part.

Une fois les composants définis, il faut s'intéresser à l'implémentation de leur spécification fonctionnelle en Altarica. Quand tous les composants sont modélisés, vient la phase de modélisation globale du système. Grâce aux capacités de l'éditeur graphique de modèle de l'environnement OCAS, le modèle de l'architecture est obtenu rapidement en interconnectant les composants élémentaires.

D'un point de vue visuel, chaque composant est représenté par une icône permettant d'élaborer une architecture multi-composants claire. De plus, il est possible d'associer une icône différente à chaque état du composant afin de mieux mettre en avant, lors des phases de simulation, l'état de chaque composant au sein d'un système.

À titre d'illustration, la Figure IV.7 présente les différentes icônes pour un calculateur primaire : état initial, configuration en logiciel A (noté S1), configuration en logiciel B (noté S2) et l'état de panne.



Figure IV.7 - Icônes du composant calculateur primaire

La sortie principale d'un calculateur commandes de vol est l'ordre d'asservissement (`OrderComputerd`). Pour cela, il a besoin de deux entrées principales : les paramètres lois pour le calcul des lois de pilotage (`DataReceived`) et l'énergie électrique (`PowerReceived`), d'où le tableau de la Figure IV.8.

Name	Type	Type name	Direction
CheckOk	Boolean		In
PowerReceived	Boolean		In
DataReceived	Recorded	Data_3values	In
OrderComputed	Recorded	Data_3values	Out

Figure IV.8 - Variables de flux du composant calculateur primaire

Pour un composant calculateur (primaire ou secondaire), nous avons défini deux modes de défaillances : la perte totale (le calculateur ne fournit aucune information) et le fonctionnement erroné (le calculateur peut fournir tout type d'informations erroné par rapport à sa spécification fonctionnelle), comme indiqué sur la Figure IV.9.

Name	Law	Law parameters
TotalLoss	Exponential law	1e-5
MalfunctionOfCPU	Exponential law	1e-6

Figure IV.9 - Événements du composant calculateur primaire

Pour qu'un calculateur propage une défaillance, il doit produire une donnée symbolisant cette défaillance et la transmettre à d'autres composants. Dans le cas d'un composant AltaRica, les variables de flux sortant jouent ce rôle. La valeur de chaque variable de flux sortant (champ `flow`) du calculateur est définie à l'aide d'une assertion (champ `assert`) en fonction des flux entrants et des états du composant. À titre d'illustration, la Figure IV.10 présente un extrait de code en ALTARICA pour un calculateur primaire dans notre architecture.

```
node main
  flow
    CheckOk : bool : in;
    PowerReceived : bool : in;
    DataReceived : T516_Data_3values : in;
    OrderComputed : T516_Data_3values : out;
    Icon : {I3500FCPC, I3499FCPC_Ko} : out;
  state
    Status : T516_Data_3values;
  event
    TotalLoss;
    MalfunctionOfCPU;

  trans
    Status = ok |- TotalLoss -> Status := lost;
    Status = ok |- MalfunctionOfCPU -> Status := err;
  assert
    OrderComputed = case{(PowerReceived  and Status = ok ) : ok,
    (PowerReceived  and Status = err) : err,
      else lost
      };

    Icon = case{(Status=ok) : I3500FCPC,
      else I3499FCPC_Ko
      };
  init
    Status := ok;
  extern
      law <event TotalLoss> = exponential(1e-5);
      law <event MalfunctionOfCPU> = exponential(1e-6);
edon
```

Figure IV.10 - Extrait du code Altarica du composant calculateur primaire

La modélisation AltaRica est hiérarchique. Un modèle de système est un nœud principal qui est composé de sous-nœuds. Chaque sous-nœud peut à son tour contenir des sous-nœuds. À titre d'exemple, un FCRM peut être modélisé comme un composant unique ; cependant on peut considérer qu'un FCRM est composé d'une voie de commande et d'une voie de surveillance, auquel cas le noeud FCRM peut être composé de deux sous-nœuds : commande (`REU_Com`) et surveillance (`REU_Mon`), comme le montre la Figure IV.11.

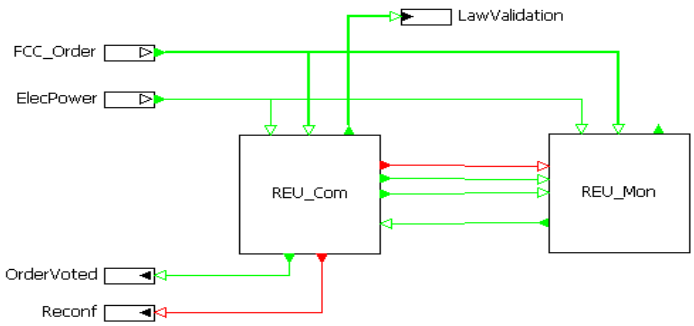


Figure IV.11 - Modèle Altarica du composant FCRM

À l'issue de la première phase de modélisation de l'architecture à vote massif, nous disposons d'un modèle abstrait complet. Les fonctions et les mécanismes principaux de chaque composant sont traités. Ainsi, les fonctions de vote et de choix des ordres FCC, les logiques de désengagement et d'activation des calculateurs, ainsi que les défaillances et la prise en compte des signaux dans le comportement des composants ont été modélisées.

Il est maintenant possible de réaliser des simulations sur le modèle final de l'architecture, simuler les enchaînements de défaillances et observer les conséquences de la perte ou l'erreur de chaque équipement du système.

La Figure IV.12 présente la vue de plus haut niveau du modèle que nous avons réalisé pour l'architecture à vote massif. Dans ce modèle, le contrôle en tangage est assuré par six actionneurs (A1, A2, A3, A4, A5 et A6). Chaque actionneur est contrôlable par un FCRM et chaque FCRM est commandé par l'ensemble des calculateurs (FCCP1, FCCP2, FCCP3, FCCP4, FCCS1 et FCCS2).

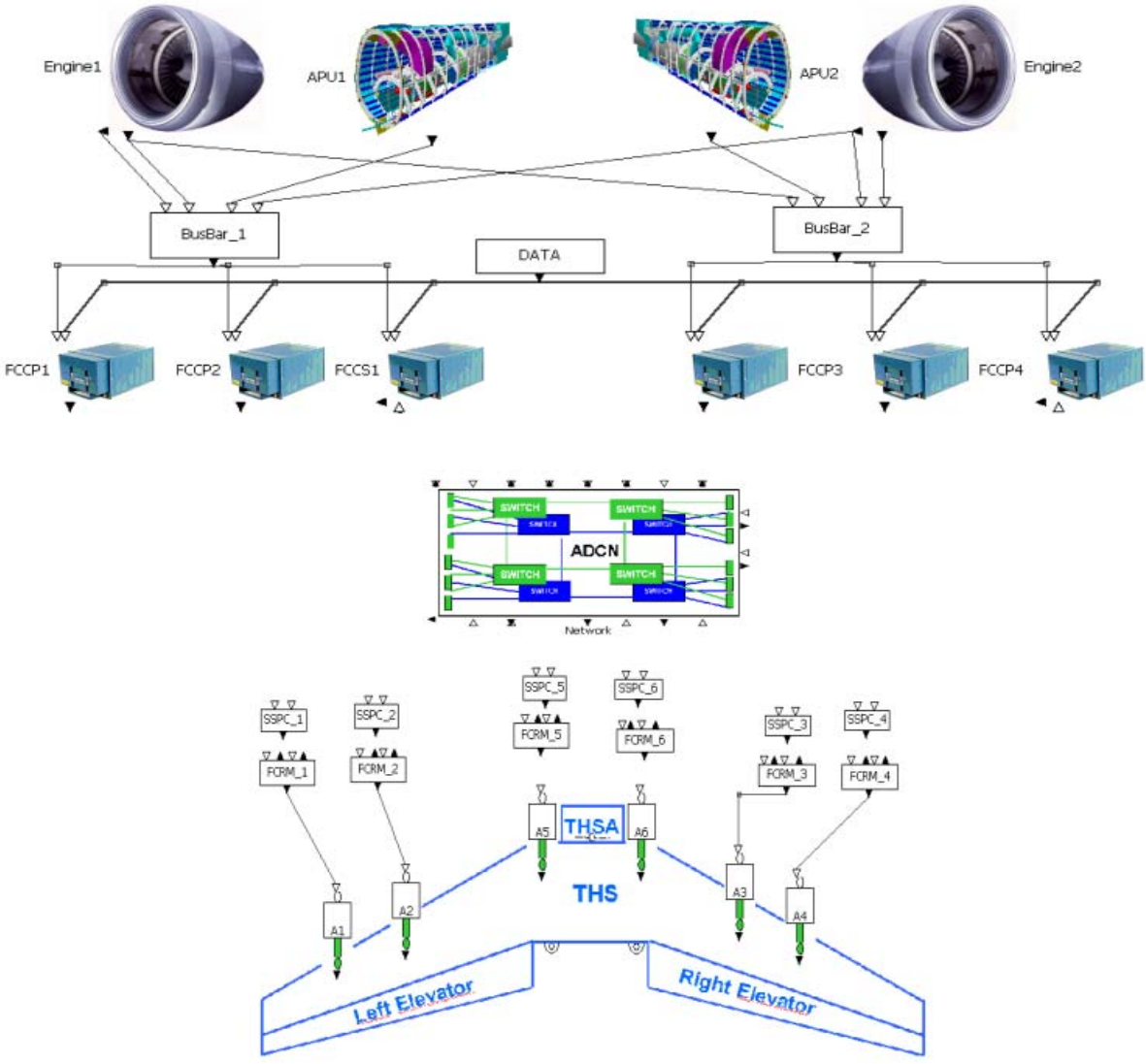


Figure IV.12 - Modèle Altarica de haut niveau de l'architecture à vote massif

IV.2.5 Simulations et Analyse de sécurité

IV.2.5.1 Analyse qualitative

Le passage en mode simulation interactive sous l'environnement OCAS nous donne accès à tous les modes de défaillance de l'ensemble des composants du modèle. Ainsi, on peut exploiter le modèle réalisé en langage AltaRica.

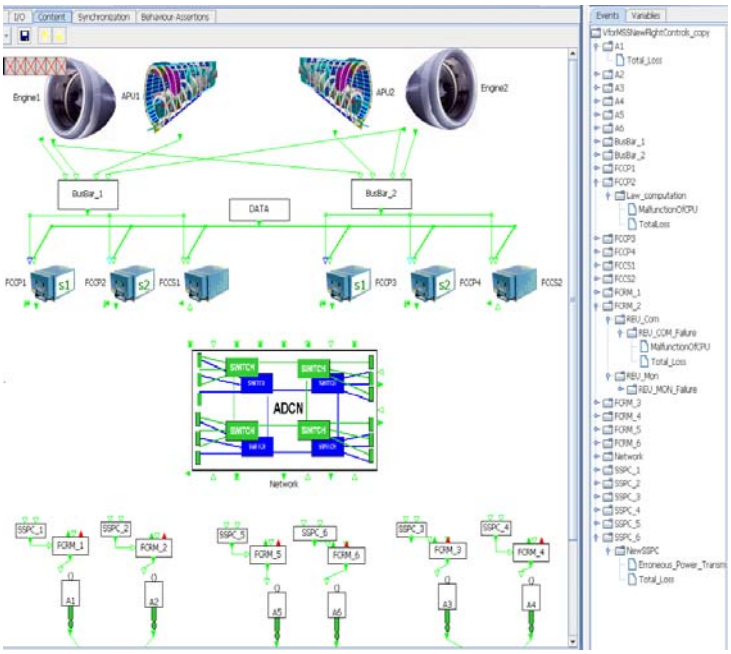


Figure IV.13 - Modèle Altarica en mode simulation interactive hors pannes

La simulation interactive permet de jouer manuellement des scénarios de pannes, pour valider ou comprendre les réactions du système et les reconfigurations éventuelles. Les simulations interactives que nous avons menées (via l'interface présentée par la Figure IV.13) ont permis de vérifier les exigences qualitatives les plus significatives du système : aucune panne simple (logicielle ou matérielle), ni aucune panne double, n'a conduit à un scénario catastrophique.

1) *Exemple de panne simple :* perte d'une barre électrique sous MMEL d'un calculateur (c'est-à-dire qu'il est en panne, mais que le décollage est autorisé)

Le scénario de la perte d'une barre électrique BusBar_1 sous MMEL du calculateur FCCP3, qui est alimenté par la deuxième barre électrique (BusBar_2) dans notre modèle, est bien couvert dans l'architecture à vote massif. Grâce au principe actuel de ségrégation électrique côté 1/côté 2, seuls les calculateurs associés au côté électrique 1 sont touchés, et le contrôle en tangage est assuré en loi directe par la moitié des actionneurs alimentés par la barre électrique BusBar_2. Le résultat est illustré par la Figure IV.14.

2) *Exemple de panne double :* perte de 2 calculateurs et cas de reconfiguration logicielle

Pour le risque de perte de deux calculateurs primaires (panne de deux calculateurs ou panne simple d'un calculateur sous MMEL d'un autre calculateur) qui exécutent le même logiciel, le principe de reconfiguration implémenté dans les FCRM permet de garder le contrôle en loi normale en reconfigurant l'un des deux calculateurs primaires restants avec le logiciel perdu. Par exemple, sous MMEL de FCCP1, exécutant le logiciel A (ou S1), si le calculateur FCCP3 exécutant aussi le logiciel A (ou S1) tombe en panne, alors le calculateur FCCP2 exécutant le logiciel B (ou S2) est reconfiguré avec le logiciel A, ceci pour garder la dissimilarité logicielle pour la loi normale. Au moment de la reconfiguration, seule la loi directe est donc disponible, ce qui est a priori acceptable. Le résultat est illustré par la Figure IV.14.

Pour ce qui est du risque de la perte de deux calculateurs primaires d'un même coté électrique, il est couvert par les stratégies de choix de l'ordre des FCRM, et donc on garde un contrôle en loi normale sans aucune reconfiguration système.

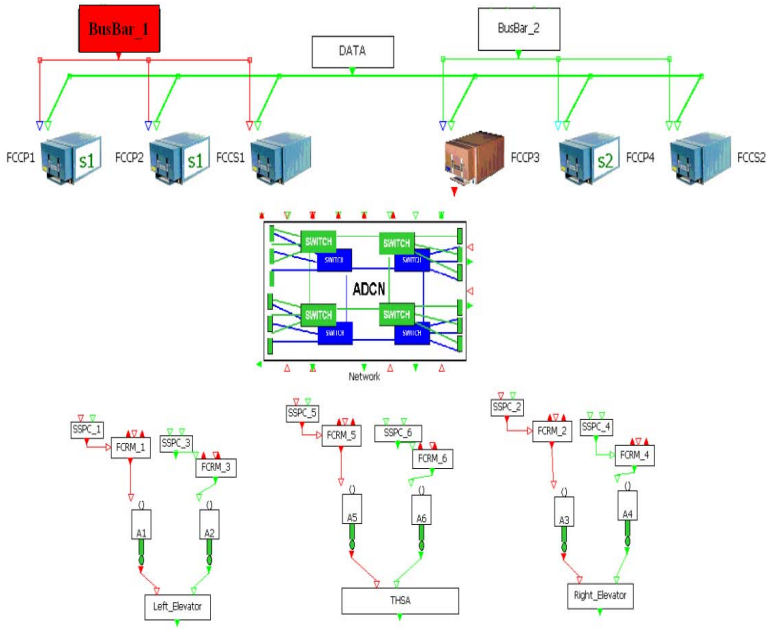


Figure IV.14 - Résultat de simulation pour la panne simple de la barre 1 sous MMEL de FCCP3

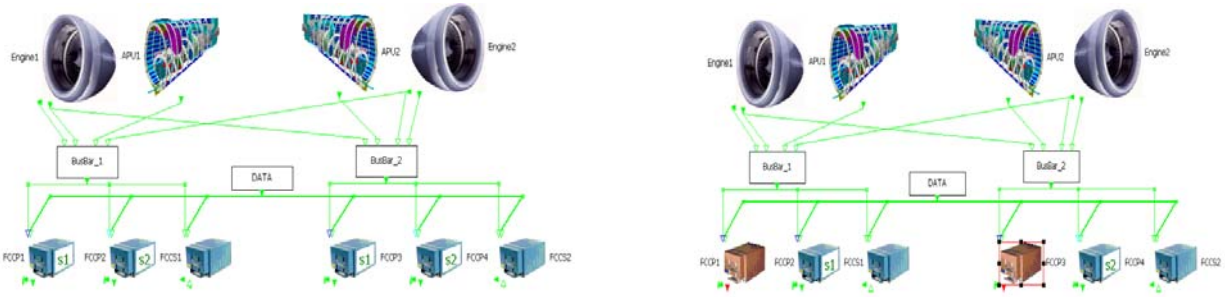


Figure IV.15 - Panne double des calculateurs FCCP1 et FCCP3 et reconfiguration logicielle a) configuration modèle avant panne ; b) configuration modèle après panne.

IV.2.5.2 Analyse quantitative

Pour des analyses quantitatives, il faut modéliser les situations redoutées à observer dans notre modèle AltaRica. Une situation redoutée peut généralement être exprimée comme une combinaison de sorties des composants du système. Pour ne pas impacter les composants (pour leur garder un modèle générique), nous avons créé des composants supplémentaires, dit ***observateurs***, reliés aux composants pris en compte dans les « Failure Condition » à traiter.

Un observateur dispose donc d'une entrée par composant contribuant à une situation redoutée et d'une sortie booléenne par situation redoutée : la sortie a pour valeur `vrai` lorsque la situation est atteinte, `faux` dans le cas contraire.

Les Figure IV.16, Figure IV.17 et Figure IV.18 montrent les résultats des générations de séquences par l'outil OCAS pour les 3 cas suivants : l'embarquement d'une surface (un des Elevator ou le PHR/THSA), une de perte du contrôle en tangage, et un passage en loi directe.

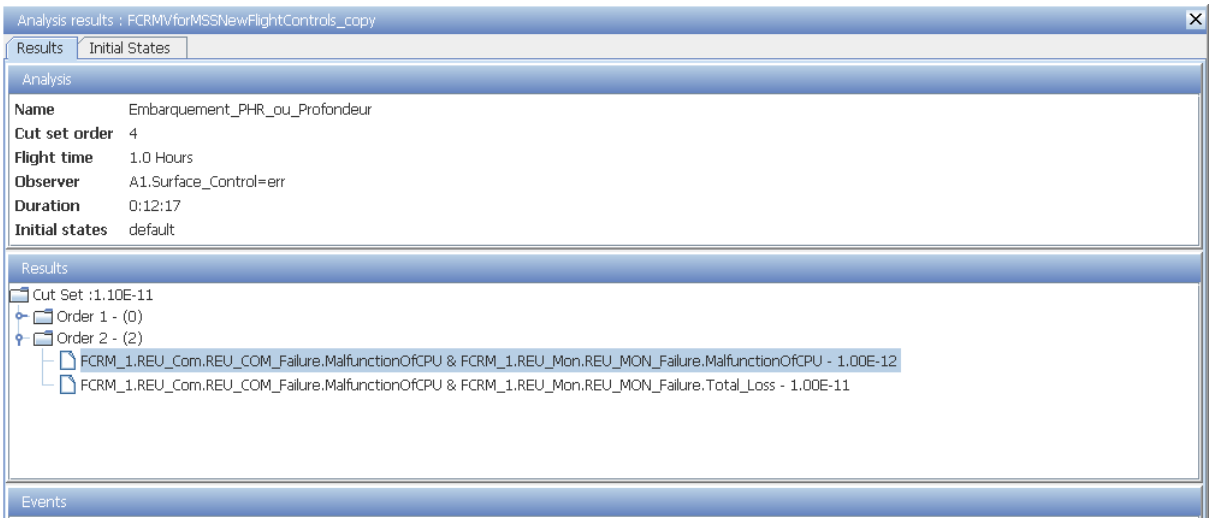


Figure IV.16 - FC1 : Embarquement PHR ou profondeur (elevator gauche ou droit)

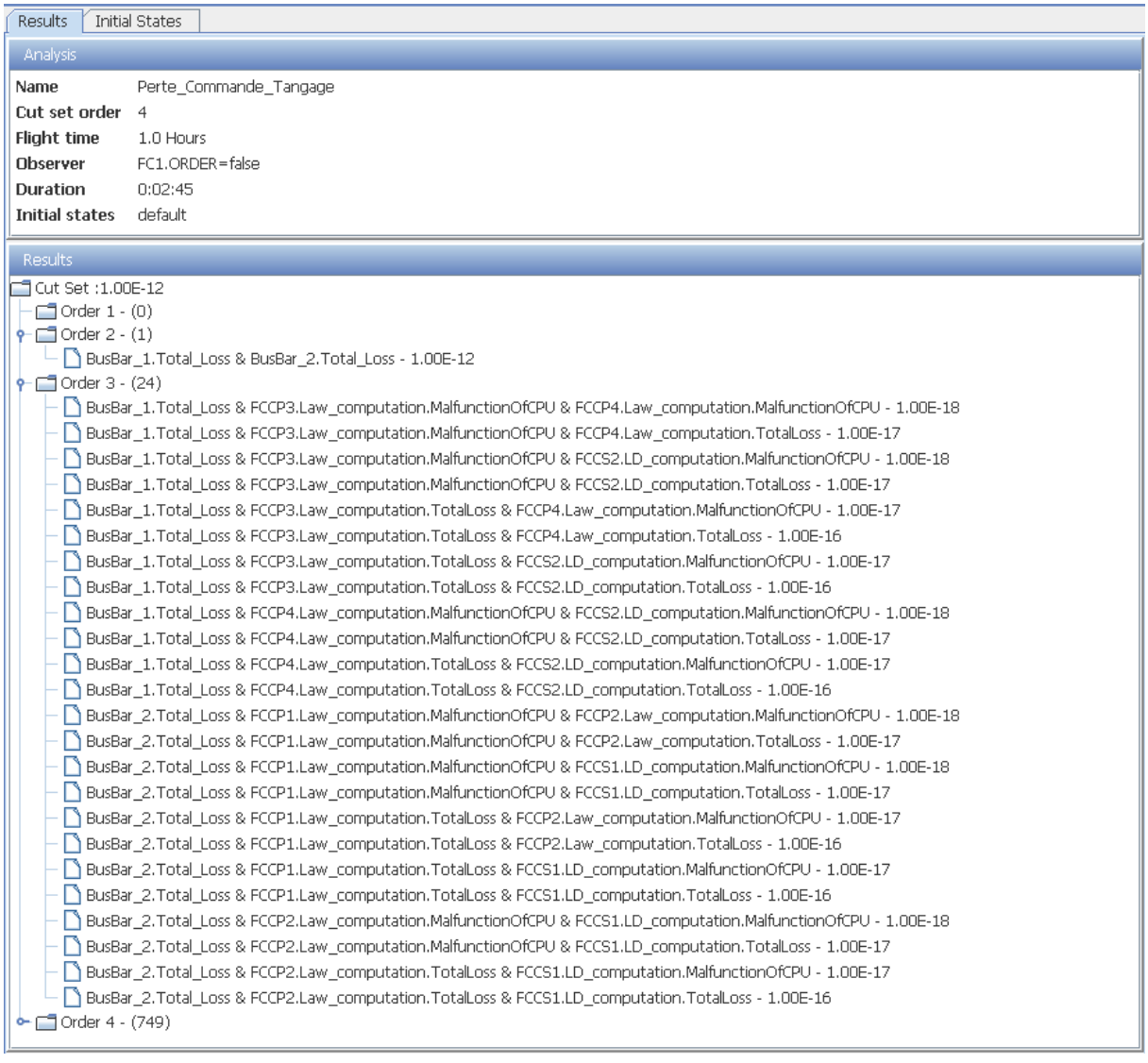


Figure IV.17 - FC 2 : Perte de la commande de tangage (perte PHR et profondeur)

Pour le cas d'un embarquement, le seul cas d'embarquement non détecté est l'embarquement de la voie COM du FCRM, non détecté par la voie MON du même FCRM. Ce résultat est en fait similaire au cas non détecté dans les architectures actuelles.

Pour le cas du système de contrôle en tangage, la perte totale de la commande en tangage se traduit par la perte de contrôle des deux gouvernes de profondeur, ou de deux moteurs de THSA et d'une gouverne de profondeur, tandis que l'embarquement se traduit par l'embarquement d'un ou de plusieurs actionneurs.

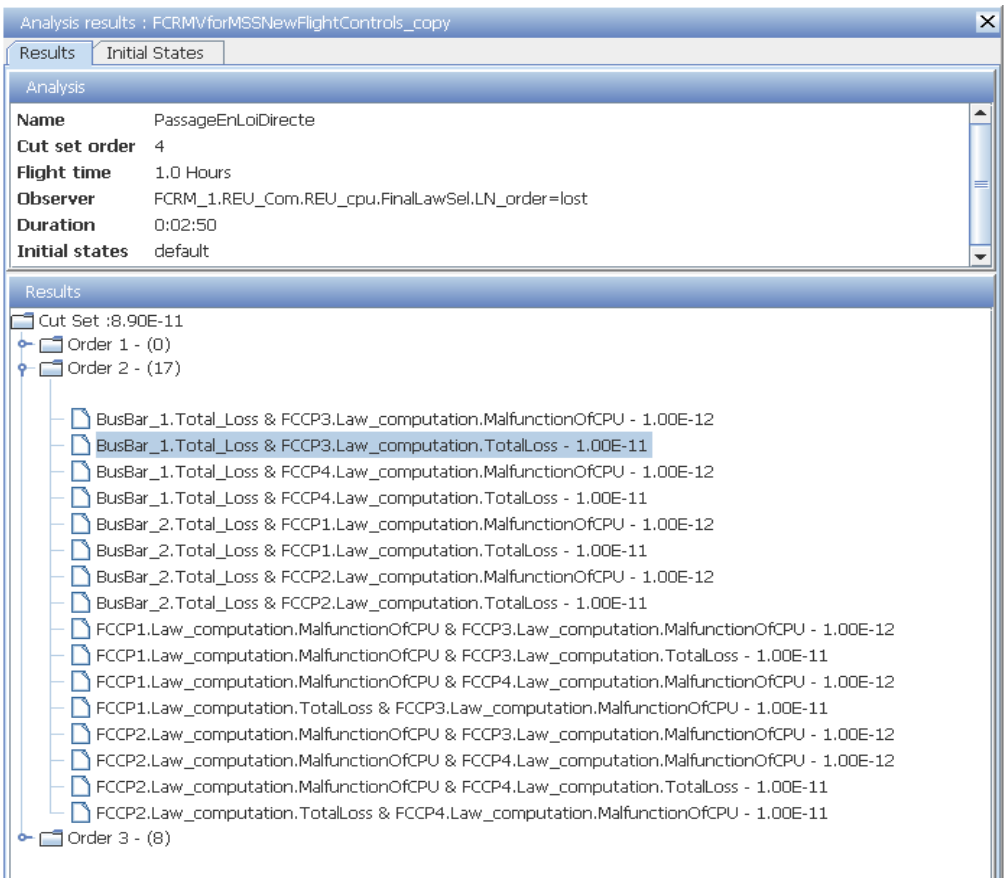


Figure IV.18 - FC2 : Passage en loi directe

Les résultats de simulation du modèle de l'architecture à vote massif sous l'environnement OCAS/ALTARICA sont en accord avec les exigences de sécurité du système des CDVE : la probabilité d'occurrence de chaque FC est inférieure ou égale à l'objectif déduit de la criticité du Tableau IV.1, et on peut donc estimer que les attentes ont été satisfaites. Les résultats des générations de séquences par l'outil OCAS sont décrits par le Tableau IV.2.

Ordre	***Nombre de séquences à tester***	***Nombre de séquences conduisant à la perte de contrôle***	***Temps de calcul***
2	1	1	1 min
3	24	12	2 min
4	749	360	2 min
5		(résultat non disponible car temps de calcul trop long)	> 2 jours

Tableau IV.2 – Génération des séquences par l'outil OCAS

Le tableau montre que les performances de l'outil sont acceptables jusqu'à l'ordre 4. Cet ordre semble suffisant pour dimensionner une architecture commandes de vol en fonction des exigences qualitatives et quantitatives du domaine aéronautique civil.

IV.2.5.3 Bilan

Bien que détaillé fonctionnellement, le modèle réalisé ne permet pas encore de réaliser une étude détaillée comme celle réalisée de manière traditionnelle (SSA). Cependant, le modèle proposé semblant correct, le travail restant devrait se limiter à introduire toutes les défaillances des composants de notre modèle par les défaillances recensées dans la FMES. Cela permettrait ensuite d'observer les résultats fournis par l'outil et de les comparer avec les coupes présentes dans la SSA.

L'étude de modélisation et de simulations menée a permis de valider l'architecture à vote massif par rapport à nos abstractions des dysfonctionnements, et de mettre en avant les qualités de l'approche analyse de sécurité basée sur les modèles Altarica :

- La lisibilité : le fort côté visuel d'un modèle permet de mieux comprendre la logique du système traité, et donc de faciliter la compréhension du système et des scénarios conduisant à un événement redouté ;
- La mise à jour de l'étude de sécurité en cas d'évolution de l'architecture d'un système : un changement d'architecture peut impacter plusieurs événements redoutés et la mise à jour des diagrammes de dépendance peut être contraignante (risque d'oubli). Grâce à cette nouvelle approche, la prise en compte des évolutions revient à mettre à jour le modèle et à générer les coupes ou séquences conduisant aux événements redoutés.
- La réutilisation : l'outil OCAS dispose d'une gestion des composants sous forme de bibliothèque. Tout composant créé et validé est stocké, ce qui permet une réutilisation ultérieure et laisse à penser que le temps passé sur les premiers modèles permettra de réaliser les modèles suivants plus rapidement.
- Le modèle commun d'échange et de travail : un modèle fonctionnel intégrant les modes de défaillances permet aux analystes de sécurité dans un premier temps de se familiariser avec le système pour ensuite travailler en coopération avec les concepteurs. Ce modèle permet d'évaluer l'architecture lors des phases initiales de conception de l'architecture.

IV.3 VALIDATION ORIENTE ROBUSTESSE SOUS L'ENVIRONNEMENT MATLAB/SIMULINK

IV.3.1 Robustesse : comportement vis-à-vis des asynchronismes

La robustesse est un terme qui qualifie la sensibilité du système à des phénomènes comme les perturbations, bruits de mesure, etc. Ces phénomènes doivent être considérés dans la conception et le système doit être capable de résister à ces phénomènes dans la limite de ses exigences fonctionnelles.

Le logiciel embarqué des avions Airbus à partir de l'A320 a été conçu et développé sous la plate-forme SAO [Brière *et al.* 1995], puis sous l'atelier SCADE/LUSTRE depuis l'A340-600 basé les travaux de [Caspi *et al.* 2001]. Cette conception engendre du code synchrone. Ce code est ensuite réparti et intégré à la plate-forme GALS de l'architecture matérielle. Dans cette architecture, chaque calculateur se comporte d'une manière synchrone (pilotée par sa propre horloge temps-réel). Par contre, les communications s'effectuent d'une manière asynchrone. Cette structure se caractérise par l'absence d'une échelle de temps global, qui n'est pas non plus supplée par la notion de synchronisation d'horloge. Les calculateurs de commandes de vol Airbus fonctionnent par échantillonnage périodique. L'échantillonnage concerne aussi bien l'acquisition des entrées capteurs que la communication entre calculateurs. L'hypothèse consistant à considérer que les unités de calcul ont des horloges parfaites n'est pas réaliste. Pour s'approcher de la réalité, nous allons prendre en compte les dérives d'horloges et les asynchronismes entre calculateurs.

Les évolutions des architectures Airbus sont caractérisées par l'effort de garder l'indépendance des différents agents de calcul (cartes ou calculateurs) qui est considéré chez Airbus comme une condition minimale de sûreté de fonctionnement. Cette notion d'indépendance est devenue une exigence de conception chez Airbus. De ce fait, toutes les nouvelles architectures ou évolutions doivent respecter cette exigence d'indépendance, ceci malgré trois types d'entraves :

1) tous les calculateurs fonctionnent en parallèle indépendamment les uns des autres,
2) chaque calculateur est complet en incluant sa propre horloge et, souvent, sa propre alimentation en énergie,
3) la communication entre les différentes unités est non-bloquante, basée sur des lectures et écritures périodiques.

L'architecture GALS et le fonctionnement par échantillonnage introduisent, en conséquence, des motifs de comportement particulier dû aux aléas et délais introduits par la communication asynchrone entre les différentes unités. L'objectif de cette partie est l'étude de ses comportements et la vérification de la robustesse de l'architecture à vote massif.

IV.3.2 Outil de simulation retenu : Matlab/Simulink

Altarica est un langage événementiel où l'évolution du modèle est conditionnée par les événements qui se produisent et non par l'évolution continue du temps, d'où le besoin d'un deuxième outil pour la vérification de l'aspect robustesse dans le cadre de la thèse.

Simulink et Matlab sont commercialisés par la société MathWorks [Mathworks]. Simulink est un environnement pour la simulation multi-domaines et le développement basé sur les modèles pour les systèmes embarqués dynamiques. Il fournit un environnement graphique interactif et des opérateurs de base qui permettent de modéliser, simuler et tester plusieurs types de systèmes, discrets ou continus. Simulink est intégré à Matlab, fournissant un accès à un éventail de fonctionnalités. Le choix de l'environnement Matlab/Simulink s'explique par les points suivants :

- capacités de modélisation similaires à celles de SCADE ;
- présence d'un ensemble de blocs ou opérateurs de base bien adaptés pour la modélisation et la simulation des phénomènes d'échantillonnage et des horloges physiques dont on a besoin pour modéliser le problème d'asynchronisme ;
- présence de solutions de prototypage rapide et modulaire ;
- possibilité d'ajouter des modules (programmation en C ou en ADA) qui n'existent pas, par défaut, dans l'outil de base et qui sont mieux adaptés à nos besoins de modélisation.

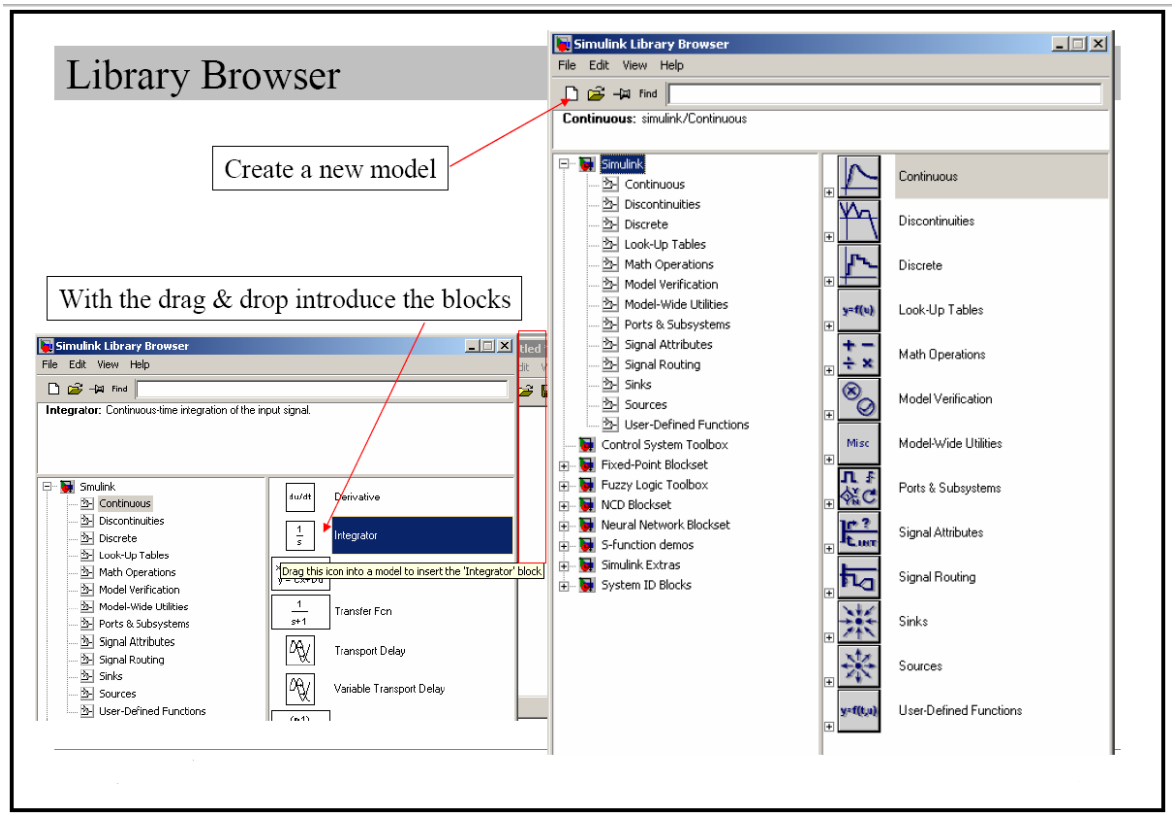


Figure IV.19 – Environnement Matlab/Simulink

IV.3.3 Modélisation sous l'environnement Matlab/Simulink

IV.3.3.1 Exemple de modules de base

Nous présentons ici une description des blocs de base offerts par l'outil Matlab/Simulink qui ont servi à l'élaboration des modèles présentés par la suite.

- Les ordres des calculateurs en lois normale et directe sont modélisés à l'aide du bloc `Data WorkSpace` de la *Library sources de données* qui permet d'introduire les signaux (récupérés lors des essais laboratoires, cf. Figure IV.22) dans notre modèle.
- Pour simuler les retards, nous avons utilisé le bloc `UnitDelay` qui permet de copier sur sa sortie le signal en entrée à l'instant d'échantillonnage précédent.
- Le bloc `Pulse Generator` est utilisé pour modéliser les horloges internes des calculateurs et des FCRM.
- Pour implémenter les fonctions de vote et de surveillance, nous avons utilisé des blocs `Embedded-Function` de la *Library User Defined Functions*. Ces blocs permettent un prototypage rapide et simple pour les personnes qui maîtrisent des langages de programmation comme le C ou ADA sans avoir une grande maîtrise de tous les blocs Simulink.

IV.3.3.2 Exemple de modules développés

Les modules supplémentaires que nous avons développés concernent les fonctions de choix et de vote pour les FCRM, les surveillances internes pour les FCC et les logiques de reconfiguration en cas de pannes. Dans le cadre de notre cas d'étude, les fonctions de choix et de surveillance ont été implémentées en langage C à l'aide des blocs `User Defined Functions`.

Pour simplifier la modélisation et les traitements, les trames réseau ont été modélisées à l'aide de multiplexeurs. Cela permet de regrouper les ordres de tous les calculateurs d'un même coté électrique dans une seule trame avant de les envoyer aux FCRM.

Le modèle détaillé d'un calculateur primaire est présenté sur la Figure IV.20 ; il comporte un bloc calcul des lois de pilotage, un bloc de surveillance et une partie non fonctionnelle pour modéliser des injections de panne conformément à l'architecture système décrite au chapitre III. Le fonctionnement interne de chaque calculateur est piloté par une horloge interne.

Nous avons ajouté trois blocs `Unit Delay` sur les retours FCRM vers les calculateurs. Cela permet de forcer le bloc surveillance du calculateur à prendre en compte les retours FCRM du cycle de calcul précédent pour le calcul de l'état courant.

Les blocs `PanneFCCP1` et `PerteLoiNormaleFCCP1` représentent respectivement un booléen de disponibilité du calculateur et un booléen de disponibilité de la loi normale. On peut ainsi simuler la panne du calculateur (PanneFCCP1 = 0) et la perte de la loi normale (PerteLoiNormaleFCCP1 = 0).

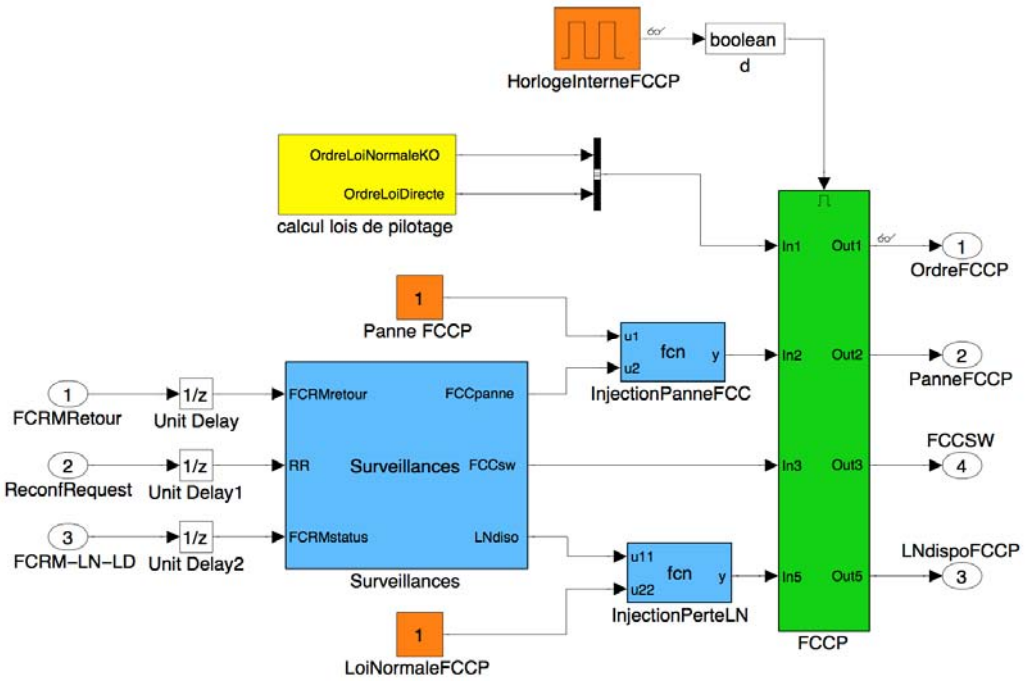


Figure IV.20 - Modèle détaillé d'un calculateur primaire

IV.3.3.3 Modèle final de simulation

Nous introduisons maintenant le modèle Simulink de l'architecture vote massif (Figure IV.21) qui nous servira d'illustration tout au long de cette partie.

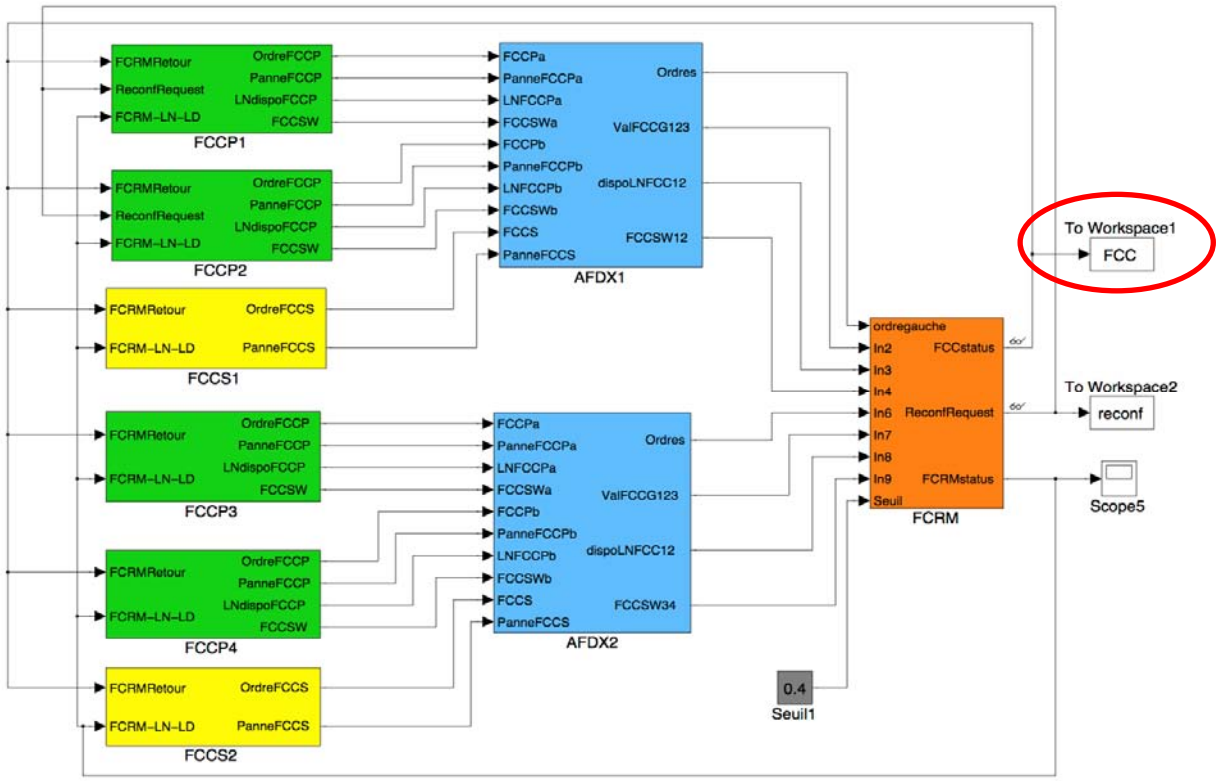


Figure IV.21 - Modèle de haut niveau Simulink de l'architecture à vote massif

Le modèle est composé de trois sous-modèles : un pour les 6 calculateurs commandes de vol (primaires et secondaires), un pour les switchs AFDX et un pour le FCRM (COM-MON). La logique de fonctionnement du modèle a été implémentée conformément à la description de l'architecture à vote massif décrite au chapitre précédent.

IV.3.3.4 Simulation

1) Simulation hors asynchronisme

Le premier objectif d'une modélisation sous Matlab/Simulink était l'implémentation et la validation des nouvelles fonctions à implémenter dans les FCC et dans les FCRM avec le nouveau découpage fonctionnel que nous avons proposé aux chapitres précédents.

Pour cela, nous avons réalisé un premier modèle de l'architecture à vote massif où nous avons implémenté les fonctions de choix et de vote ainsi que les retours FCRM vers les calculateurs (FCCP et FCCS) et les surveillances internes à ces derniers (voir détails au chapitre III). Ce premier modèle a pour but la validation de l'architecture d'un point de vue fonctionnel et sans considérer les déphasages d'horloge entre les horloges des différents composants de l'architecture.

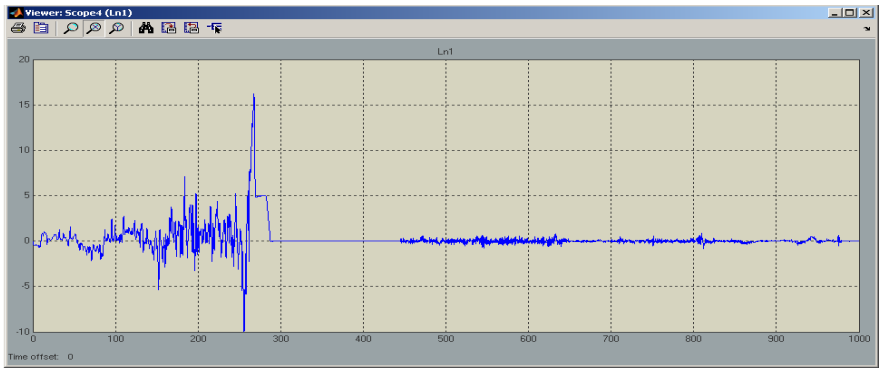


Figure IV.22 – Signal de base ordre loi normale Elevator d'un PRIM A 380

Dans un premier temps, on va utiliser des horloges et des périodes identiques pour tous les composants du modèle (10 ms pour les calculateurs et 5 ms pour les FCRM) pour simuler le comportement des FCC et des FCRM hors panne, et en présence des pannes d'une façon statique. Cette première simulation fonctionnelle nous a permis de s'assurer que les fonctions implémentées sous Matlab/Simulink correspondent bien à la spécification fonctionnelle définie au chapitre III pour l'architecture à vote massif.

Dans notre modèle, chaque FCRM maintient un compteur Ci pour chaque calculateur. Initialement, les compteurs des calculateurs primaires sont mis à deux. Ce qui correspond à un calculateur actif en loi normale et en loi directe. Les compteurs des calculateurs secondaires sont mis à 1 vu qu'ils ne peuvent être actifs qu'en loi directe. Initialement on a : C1_FCCP1 =2 ; C2_FCCP2 =2 ; C3_FCCS1 = 1 C4_FCCP3 =2 ; C5_FCCP4 =2 ; C6_FCCS2 =1. Si le FCRM ne reçoit plus d'ordre d'un calculateur secondaire, ou s'il détecte une erreur sur son ordre en fonction de ces fonctions de choix, il décrémente le compteur.

Nous allons maintenant étudier deux cas.

- *Cas d'une configuration nominale (hors panne)*

Le résultat de la simulation de la sortie de la fonction FCRM qui implémente la logique décrite précédemment est donné par la Figure IV.23. Cette simulation nous a permis de valider les fonctions de choix pour les sites gauche et droit, respectivement pour la loi normale (la valeur votée est la médiane avec un poids égal à deux) et la loi directe (la valeur votée est la médiane avec un poids égal à trois).

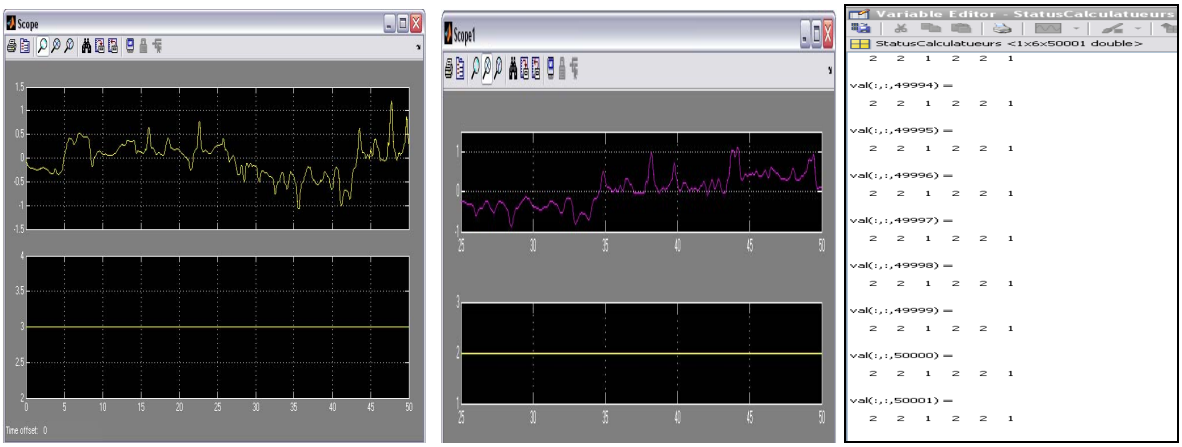


Figure IV.23 - Résultats de la simulation de la sortie fonction de vote site gauche a) ordres loi directe, b) ordres loi normale, c) valeurs finales des compteurs

- *Cas d'une configuration avec panne : perte de la loi normale du calculateur FCCP2 sous MMEL du calculateur FCCP1*

Pour simuler ce cas, il suffit d'injecter ces pannes dans le modèle. Pour cela, sur le modèle du calculateur FCCP1, on met Panne(FCC1) ==0 et PerteLoiNormale (FCCP2) ==0

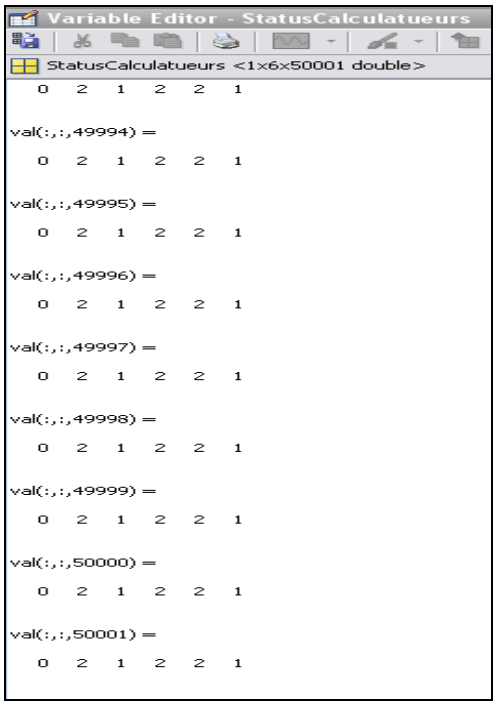


Figure IV.24 – Valeurs finales des compteurs sous MMEL de FCCP1

Une comparaison bit à bit des ordres de la loi directe des calculateurs FCCP1, FCCP2 et FCCS1 sans l'utilisation d'un seuil de tolérance, hors panne et avec un décalage des horloges ne permet pas au FCRM de déterminer une médiane avec un poids égal à 3. Les résultats des simulations de la Figure IV.25 montrent que le poids est nul et que la sortie est non disponible le plus souvent.

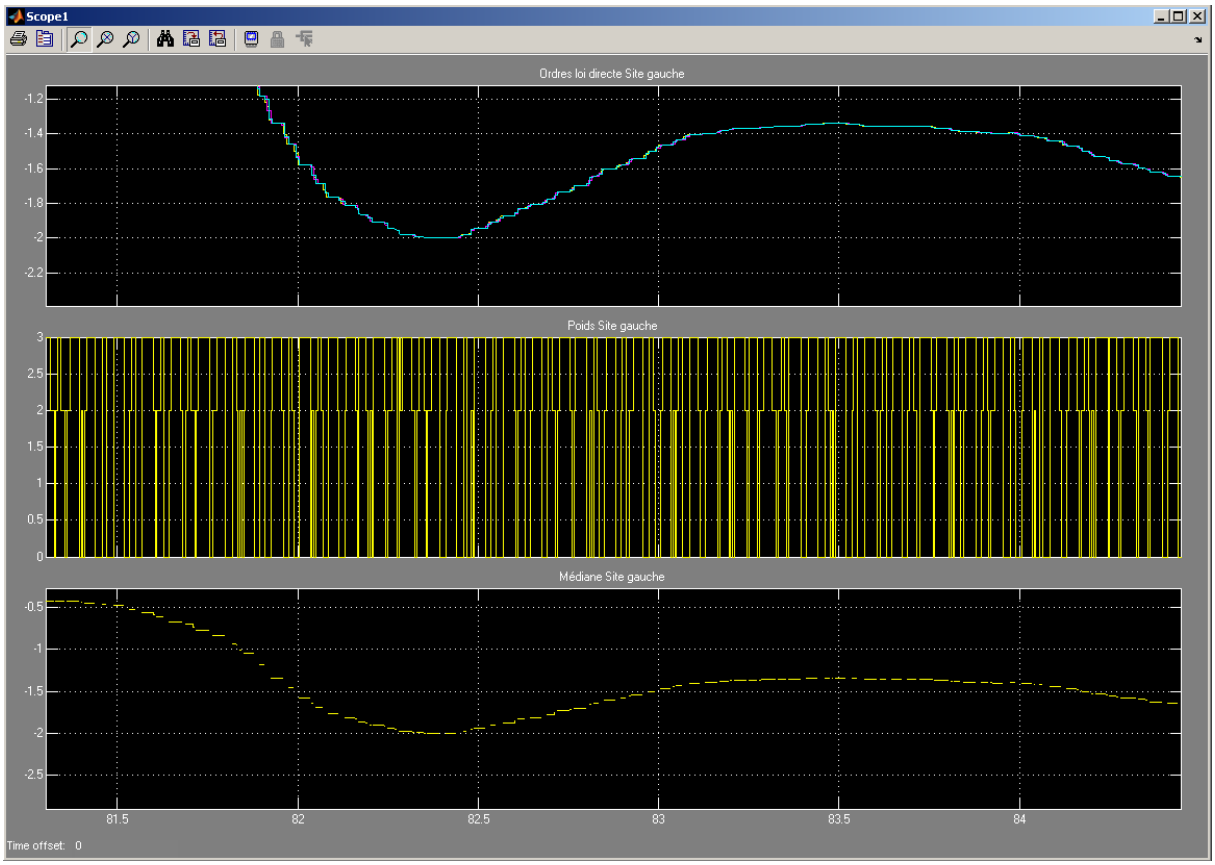


Figure IV.25 - Injection des asynchronismes hors seuil de tolérance

Ce cas de panne a été couvert par l'introduction d'un seuil de tolérance dans toutes les fonctions de vote des FCRM. Le calcul théorique donne un seuil de tolérance des dérives d'horloge de 0.4 ° pour une vitesse maximale de déploiement de 40°/s et un cycle de base de 10 ms. Les résultats de simulation ont aussi donné un seuil proche de 0.4°. Une comparaison des signaux des calculateurs point à point avec l'introduction d'un déphasage temporel, sans la considération d'un seuil de tolérance se voit prise en défaut dans une telle situation, même en l'absence de faute, et des désaccords durables sont détectés. Le poids de la médiane est rarement à 3, d'où un problème évident de robustesse.

2) Simulation et validation de l'architecture avec asynchronisme et cas de panne

Nous considérons maintenant des fréquences différentes d'échantillonnage suivantes :
Fe (FCCP1) = 10 ms ; Fe (FCCP2) = 9.9 ms ; Fe (FCCP3) = 10.5 ms ;
Fe (FCCP4) = 9 ms ; Fe (FCCP5) = 10.7 ms ; Fe (FCCP6) = 11 ms.

Le risque lié à la défaillance d'un calculateur, qui se manifeste par un embarquement des sorties en loi normale ou en loi directe, est modélisé par la fonction erreur. On distingue deux

types d'erreur : 1) la panne solide : le signal de base (signal de la loi normale) est remplacé par un autre signal, 2) la panne liquide : un autre signal vient s'ajouter au signal de base.

Nous avons utilisé une fonction décrite en langage C qui permet :

1) de remplacer l'ordre de la loi normale du FCCP1 par une rampe de pente égale à 0.3 pour modéliser la panne solide illustrée par la Figure IV.26 et la Figure IV.27 ;
2) d'ajouter une sinusoïde à la loi normale pour modéliser la panne liquide illustrée par la Figure IV.28 et la Figure IV.29.

L'injection de panne est introduite, dans les deux cas, à partir de l'instant t1 = 120 s (modélisé à l'aide de bloc `Digital CLOCK`) par rapport à un temps de simulation de 400 s avant d'être envoyé au FCRM.

Rappelons que notre objectif est d'assurer un niveau de détection de panne au niveau des FCRM équivalent à celui des architectures actuelles en présence des asynchronismes.

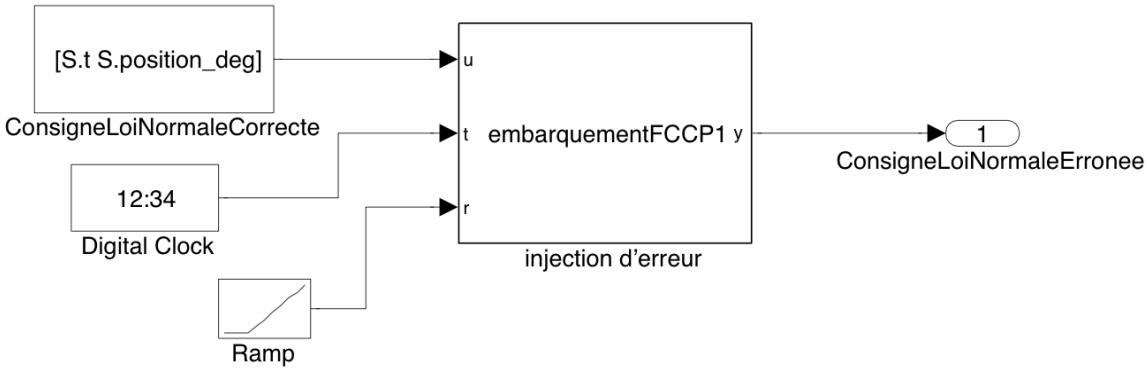


Figure IV.26 - Modèle d'injection d'erreur pour la panne solide

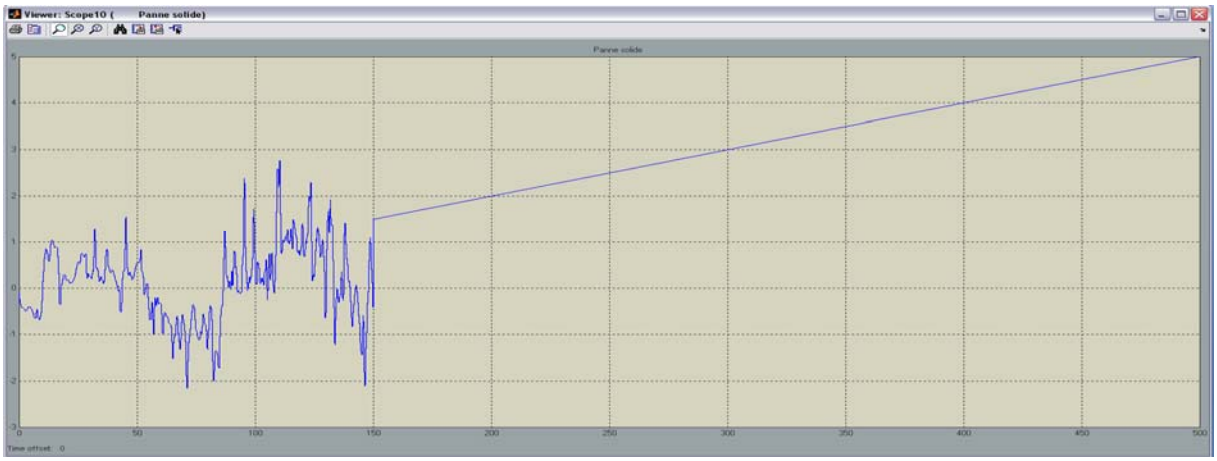


Figure IV.27 - Signal erroné – panne solide

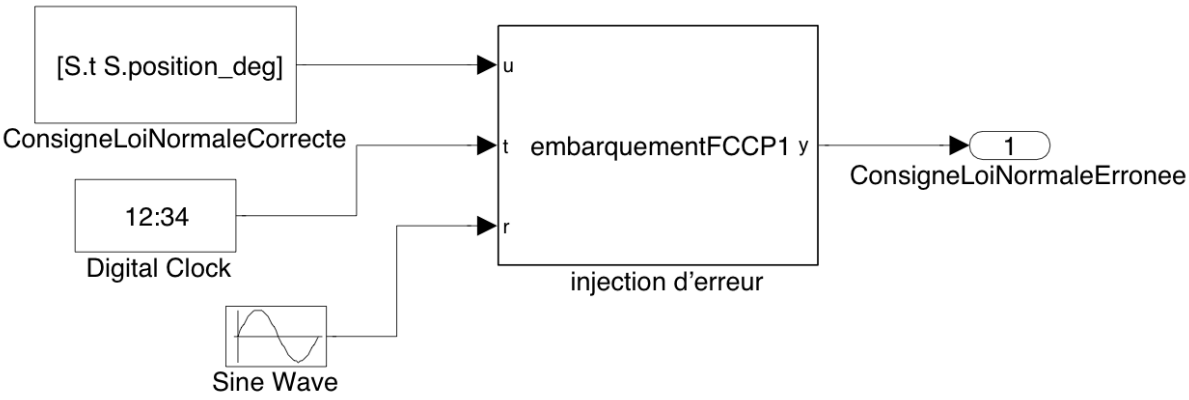


Figure IV.28 - Modèle d'injection d'erreur pour la panne liquide

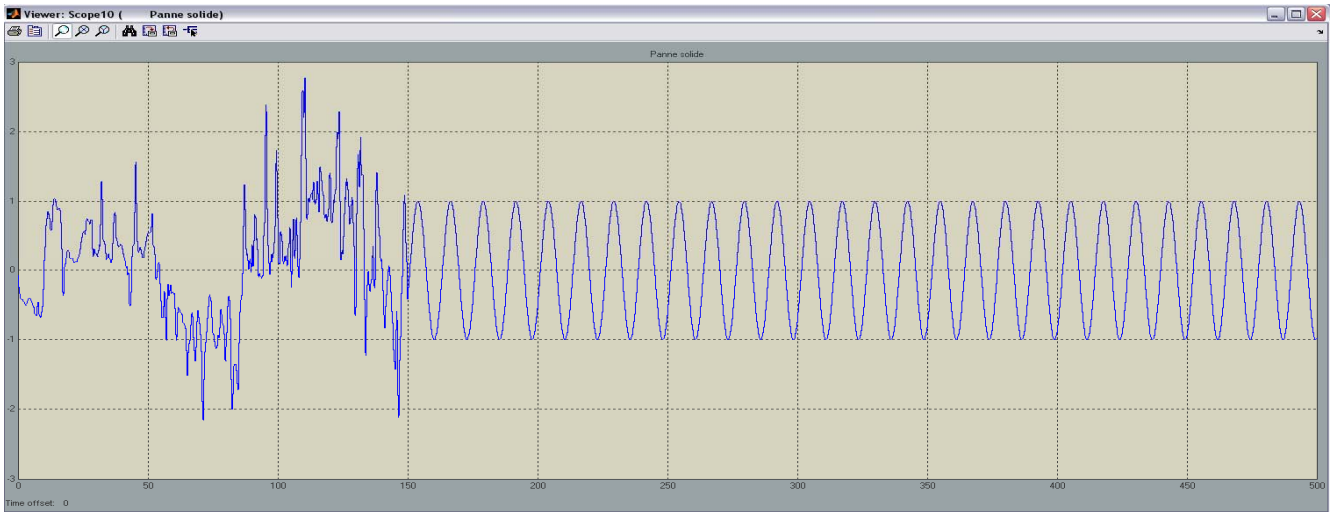


Figure IV.29 - Signal erroné – panne liquide

La Figure IV.30 montre que le FCRM a bien détecté la panne de calculateur FCCP1 et que celui-ci a été écarté du processus de vote à partir de l'instant t120. Pour réduire le temps d'exécution des simulations et la capacité de mémoire, nous avons utilisé un temps de simulation de 400 s au lieu de 4010 s (le temps réel des enregistrements de la loi normale lors des essais en vol).

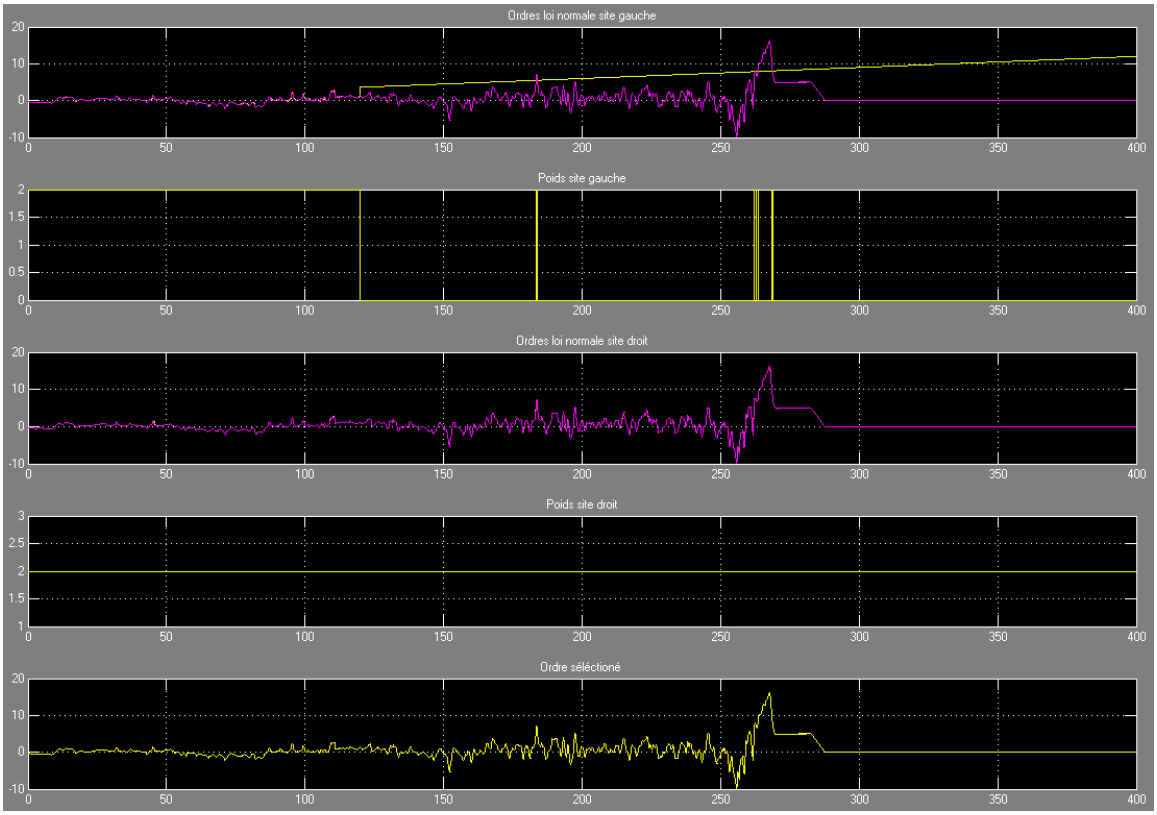


Figure IV.30 - Illustration de l'embarquement de la loi normale FCCP1

On constate sur la Figure IV.31 que la loi normale est toujours disponible grâce aux ordres des calculateurs du côté droit en cas de panne du calculateur FCCP2 sous MMEL du calculateur FCCP1.

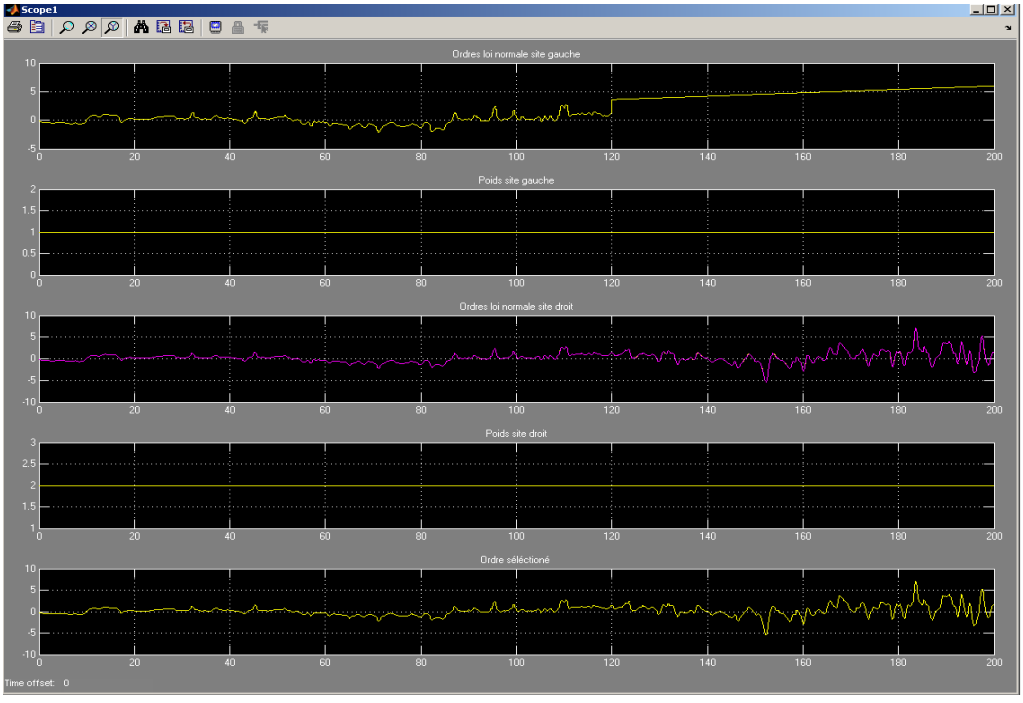


Figure IV.31 - Illustration de l'embarquement de la loi normale FCCP2 sous MMEL

IV.4 CONCLUSION

Nous avons proposé dans le chapitre précédent une nouvelle architecture avec de nouvelles répartitions des fonctions entre l'avionique centrale (calculateurs CDV) et l'avionique déportée (électroniques locales des actionneurs) ayant pour avantage de consommer une moins grande quantité de ressources par rapport aux architectures conventionnelles tout en satisfaisant un même objectif de sécurité / disponibilité.

Ce chapitre a été consacré à la validation par simulation de l'architecture à vote massif, en se basant sur les outils de modélisation OCAS avec le langage Altarica et l'environnement MatLab/Simulink. Les résultats que nous avons obtenus aux cours des différentes simulations ont confirmé la conformité de l'architecture avec l'ensemble des exigences de sécurité et de robustesse.

Il est à noter que le modèle ALTARICA réalisé, bien que détaillé, ne permet pas encore de réaliser une comparaison directe avec l'étude réalisée de manière traditionnelle (SSA).

Chapitre V - ARCHITECTURE À PRIORITÉS

Ce dernier chapitre est consacré à la présentation des caractéristiques, du mode de fonctionnement et des éléments de validation de la deuxième architecture proposée dans la thèse : l'architecture *« à priorités »*. Issue de la démarche développée au deuxième chapitre, il s'agit là aussi, comme pour l'architecture à vote massif, d'une architecture optimale (avec notamment moins de matériel mis en jeu) et satisfaisant les objectifs de sécurité et disponibilité, pour relier les calculateurs aux actionneurs des commandes de vol.

Cette architecture *« à priorités »* a pour points communs avec l'architecture *« à vote massif »* du chapitre III, de faire appel aux mêmes types de calculateurs simplex, aux mêmes types d'actionneurs intelligents, et aux mêmes technologies de communications numériques entre l'avionique centrale (calculateurs CDVE) et l'avionique déportée (électroniques locales des actionneurs). L'architecture *« à priorités »* est également basée sur la même architecture matérielle et la même architecture logicielle.

Les différences apparaissent au niveau de la répartition de l'intelligence (ou architecture fonctionnelle). Car, si bien entendu elle met en œuvre une répartition des fonctions intelligentes (entre les calculateurs et les électroniques locales des actionneurs) qui est différente par rapport aux architectures actuelles, cette répartition est aussi en partie différente de celle de l'architecture à vote massif, et ce sur les points suivants.

Les comparaisons des ordres calculateurs sont réalisées au niveau des calculateurs (un peu comme dans les solutions actuelles), mais cette fois, les calculateurs peuvent faire plusieurs comparaisons au lieu d'une seule comparaison (contrairement au principe de COM/MOM des architectures actuelles) et ils ne jouent pas tous le même rôle (ou fonction). On introduit la notion de *« calculateur maître »* et de *« calculateur valideur »*.

- Le *« calculateur maître »* contrôle l'actionneur en lui envoyant un ordre à exécuter,
- Le *« calculateur valideur »* informe l'actionneur sur la validité de l'ordre du maître.

Seule la sélection des ordres est effectuée au niveau de chaque électronique locale de chaque actionneur. Et cette sélection ne se fait plus de la même manière que dans l'architecture à vote massif. Elle se fait toujours en fonction des informations fournies par les calculateurs, mais ces informations sont désormais assorties de priorités, utilisées pour faire la sélection au niveau des actionneurs. Ces priorités, mais également les logiques de reconfiguration système, sont cette fois définies statiquement par les concepteurs. D'où, au final, le nom de « architecture *à priorités* ».

V.1 DESCRIPTION GENERALE : PRINCIPES

Comme annoncé dans l'introduction, dans cette deuxième solution architecturale, dite *« à priorités»*, contrairement à l'architecture à vote massif, les calculateurs CDVE ne jouent pas tous le même rôle. On distingue les fonctions *« calculateur maître »* et *« calculateur valideur »*

Le choix des *valideurs*, pour un *maître* donné, est défini en fonction du type du matériel, et du logiciel qu'ils exécutent, le tout par rapport au maître *considéré*, avec pour objectif de garantir une dissimilarité logicielle entre chaque calculateur maître et ses valideurs potentiels.

Le comportement global dans une architecture *« à priorités »* est le suivant. Initialement, comme pour l'architecture *« à vote massif »*, tous les calculateurs (primaires et secondaires) du côté gauche (respectivement du côté droit) (cf. chapitre II, introduction) calculent les lois de pilotage et l'ensemble des ordres des gouvernes pour tous les actionneurs valides, qu'ils soient en mode actif ou amorti (ou stand-by). Plus précisément, comme déjà dit au chapitre III, tous les calculateurs primaires calculent les lois normales (LN) et directes (LD), et tous les calculateurs secondaires calculent les lois directes (LD).

La où apparaissent les différences, c'est que cette fois, chaque calculateur joue un double rôle : celui de *maître* pour un groupe d'actionneurs, et celui de *valideur* pour les autres actionneurs. Rappelons qu'un groupe d'actionneurs désigne l'ensemble des actionneurs exécutant le même ordre gouverne, comme par exemple les 4 actionneurs de profondeur ou les 2 actionneurs de direction (cf. chapitre III, section III.1.).

Chaque calculateur *maître* est associé à un ou plusieurs calculateurs *valideurs*, et chaque actionneur est associé à un ou plusieurs maîtres auxquels sont associées des priorités différentes. En effet, un actionneur peut avoir, selon sa criticité, un ou plusieurs maîtres, avec une logique de priorité pour définir quel maître choisir à chaque cycle de calcul.

Chaque calculateur transmet donc ses ordres gouvernes vers les actionneurs dont il est maître, mais également vers tous les autres calculateurs. Ce qui est une nouveauté.

Le rôle de validation, du calculateur valideur, consiste à comparer son propre ordre avec celui du calculateur maître, pour les actionneurs dont il n'est pas maître lui-même (mais seulement valideur). Si le résultat de la comparaison est positif (ordre du maître validé par rapport à un seuil de tolérance), le calculateur valideur transmet cette information (*maître valide*) aux actionneurs concernés. La forme la plus simple est d'envoyer un booléen de validité.

Chaque actionneur (unité COM et unité MON) reçoit donc :

1) un ou plusieurs ordres (autant que de maîtres potentiels),
2) ainsi que les validités correspondantes (provenant des calculateurs valideurs).

L'unité de commande COM réalise les fonctions de choix de l'ordre à exécuter selon sa logique interne décrite plus loin. De son côté, l'unité de surveillance MON effectue les mêmes types d'opérations pour, qu'à la sortie, les valeurs obtenues par chaque unité (COM et MON) soient comparées et, en cas de désaccord, l'actionneur soit désactivé.

A n'importe quelle étape, l'unité MON (respectivement l'unité COM) a la possibilité de couper l'unité COM (respectivement l'unité MON) si elle détecte un désaccord entre les deux.

Le schéma de la Figure V.1 décrit les bases de cette architecture « à priorités ».

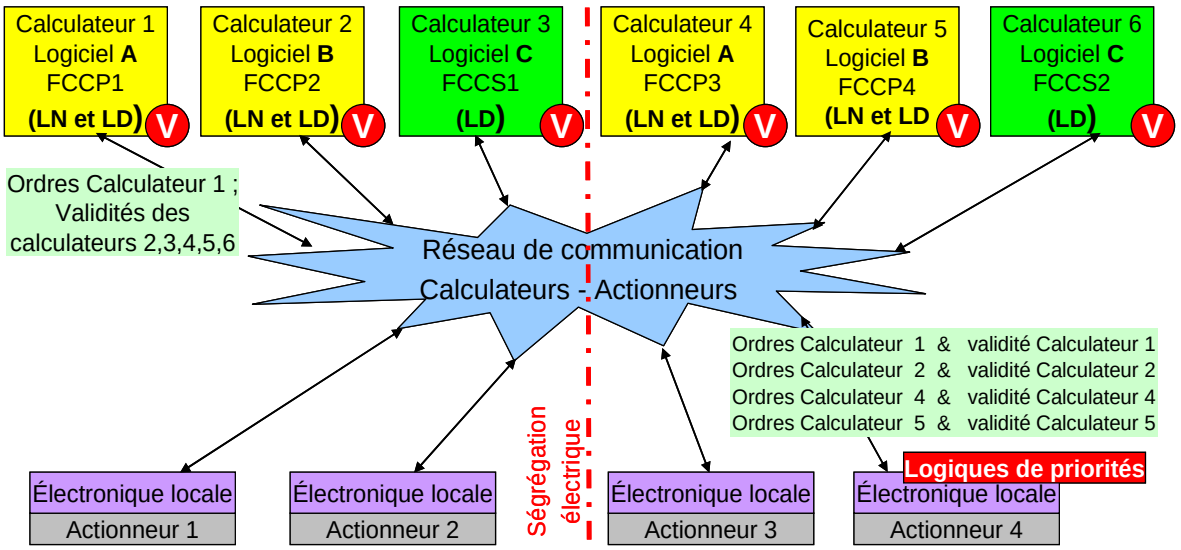


Figure V.1 - Principe de base de l'architecture à priorités

Dans cette architecture, contrairement à l'architecture COM/MON actuelle (cf. chapitre I), le calculateur *valideur* ne décide pas de lui-même si l'ordre du *maître* doit être transmis ou non à l'actionneur : l'ordre du maître est systématiquement transmis à l'actionneur, qui lui, grâce aux moyens logiques qu'il comporte, décide, en fonction du résultat renvoyé par le ou les calculateurs valideurs du calculateur maître, d'exécuter ou non l'ordre du calculateur maître. Il est possible d'associer un calculateur maître à plusieurs calculateurs valideur ou bien encore d'associer un actionneur à plusieurs couples maître/valideur de calculateurs pour s'assurer avec une sécurité renforcée de la fiabilité de l'ordre transmis. Ceci permet de rendre chaque calculateur polyvalent, les rôles entre maître et valideur pouvant être échangés à tout moment, en fonction des défaillances des calculateurs, ce qui contribue à rendre le système plus flexible, et à réduire le nombre total de calculateurs nécessaires.

En principe, un seul valideur est suffisant pour valider l'ordre du calculateur maître. Cependant, l'utilisation d'un deuxième valideur permet augmenter la disponibilité du maître en cas de perte ou de fonctionnement erroné du premier valideur.

L'actionneur est associé à plusieurs calculateurs maîtres pour maximiser les chances qu'un ordre provenant d'un maître soit considéré comme valide, et puisse donc être fiable. La mise en relation des unités de commande et de surveillance des actionneurs avec les calculateurs peut ainsi s'opérer directement ou indirectement (par l'intermédiaire de l'autre unité).

Ainsi, les calculateurs maîtres reliés à l'actionneur sont *« hiérarchisés » par niveau de priorité*, de sorte que c'est l'ordre du premier maître valide selon cette logique de priorité qui est exécuté. On propose 5 niveaux de priorité pour le choix du calculateur maître. Le Tableau V.1.donne un exemple pour l'établissement de ces niveaux.

Priorités	***Calculateur Maître***	***Niveaux de lois***
5	Calculateur FCCP1	loi normale
4	Calculateur FCCP3	loi normale
3	Calculateur FCCP2	loi normale
2	Calculateur FCCS1	loi directe
1	Calculateur FCCS2	loi directe

Tableau V.1 - Priorité des calculateurs

Pour une même loi de pilotage, on exige une dissimilarité logicielle pour le calcul d'un ordre entre calculateurs *maître* et *valideur* pour offrir une sécurité supplémentaire. En effet, un calculateur *maître* qui exécute un logiciel A ne doit jamais avoir des valideurs qui exécutent le même logiciel A. Par contre, il peut y avoir des valideurs qui ont le même matériel que lui.

Dans l'architecture à priorité, contrairement à l'architecture à vote massif, les calculateurs communiquent entre eux. Cela impacte le processus de reconfiguration. Lorsque le calculateur FCCP1, avec un logiciel A, voit que les calculateurs FCCP2 et FCCP4 (tous les deux avec un logiciel B) sont perdus, il se reconfigure en logiciel B afin d'assurer la dissimilarité logicielle pour la loi normale entre lui et le calculateur FCCP3 qui fonctionne avec le logiciel B.

De même, lorsque le calculateur FCCP2 voit que les calculateurs FCCP1 et FCCP3 sont perdus, il se reconfigure selon le logiciel A afin d'assurer la dissimilarité logicielle pour la loi normale entre lui et le calculateur FCCP4.

Et de façon similaire à l'architecture à vote massif, la reconfiguration logicielle permet donc de maximiser l'utilisation de chaque calculateur, ce qui contribue à minimiser le nombre total de calculateurs tout en conservant, pour une même loi de commande, une dissimilarité logicielle entre calculateurs maître et valideur.

Pour conclure, le contrôle de l'actionneur est réalisé en trois étapes majeures :

1) Etape 1 : l'unité COM (respectivement MON) reçoit les ordres gouverne de tous les calculateurs maîtres contrôlant l'actionneur considéré. L'unité COM (respectivement MON) choisit l'ordre à appliquer en fonction de sa logique de priorité.
2) Etape 2 : l'unité COM (respectivement MON) doit valider cet ordre à partir des informations des calculateurs valideurs correspondants. Si l'ordre du maître prioritaire sélectionné n'est pas correctement validé par au moins un calculateur valideur, cet ordre est ignoré, et un nouveau calculateur maître est choisi en fonction de la logique de priorité.
3) Etape 3 : les deux unités COM et MON transmettent (dans une ou plusieurs trames) à tous les calculateurs (maîtres et valideurs) l'ordre appliqué, la position actuelle de l'actionneur (ou gouverne), ainsi que les adresses (ou les identifiants) du maître et des valideurs actuels (ainsi que des maîtres rejetés, le cas échéant). Un calculateur invalidé par l'ensemble de ses calculateurs valideurs est annoncé en panne au niveau du cockpit.

Il existe un certain déséquilibre dans la thèse entre l'option à vote massif et l'option à priorités. En effet, historiquement, nous avons commencé par étudier l'option à vote massif en premier ce qui a soulevé certains problèmes (protocole d'échange, définition des surveillances, modélisation et implémentation des fonctions de choix). Les solutions apportées sont valables aussi bien dans le cas de l'option 1 que celui de l'option 2 ; cela nous a permis de gagner beaucoup de temps pour l'option 2.

V.2 DESCRIPTION DETAILLEE DES DIFFERENTS COMPOSANTS DE L'ARCHITECTURE

V.2.1 Calculateur CDVE

V.2.1.1. Architecture matérielle et logicielle

L'architecture matérielle et logicielle des calculateurs dans l'architecture à priorités est identique à celle de l'architecture à vote massif (cf. chapitre III, section III.2.1.1.).

V.2.1.2. Description fonctionnelle

1) Interface système

Pour fonctionner, les calculateurs commandes de vol produisent et consomment des flux de données. Ces flux sont reçus et transmis via le même réseau de communication dans une ou plusieurs trames de données, selon les technologies et les protocoles de communication retenus.

Certaines informations peuvent être envoyées à des cycles différents (ordres et validations). Ces calculateurs ont également besoin d'une alimentation électrique. Celle-ci est fournie par l'un des deux systèmes électriques de l'avion, en fonction de la localisation géographique du calculateur (côté droit ou côté gauche). Le Tableau V.2 décrit ces flux de données.

Flux de données	*Description*	*Type*	*Flux entrant (source)*	*Flux sortant (destinataire)*
Paramètres lois	Les fonctions lois de pilotage consomment des paramètres « lois » : position envoyée par les organes de commande en cockpit, paramètres anémo-inertiels de l'avion, …		Système avion cockpit, capteurs, …	
Ordres Maître	Le calculateur maître calcule les lois de pilotage et l'ensemble des ordres gouverne pour tous les actionneurs.	Trame Ordre		FCRMs
Validations	Le calculateur valideur compare son ordre avec les ordres des autres calculateurs maîtres pour lesquels il est valideur et il transmet le résultat (validation ou non) aux actionneurs concernés.	Trame Validation		FCRMs
Retour_FCRM	Le calculateur reçoit des retours de toutes les unités COM et MON de tous les actionneurs dont il est maître. Les retours donnent la validité de son statut de maître par rapport aux résultats de choix des priorités fait par chaque FCRM.	Trame FCRM	FCRMs	

Tableau V.2 - Flux de données entrants et sortants des calculateurs

Plusieurs remarques :

- Vu les différences entres les deux architectures, on ne retrouve pas tous les mêmes flux que pour les calculateurs de la première architecture (à vote massif).
- Les trames *Trame_Ordre* et *Trame_FCRM* ont un format semblable à celui défini pour l'architecture à vote massif (cf. III.2.2.2.a). Et nous n'avons pas repris ici les flux de la gestion électrique des FCRM qu'assurent les calculateurs.

- Les *Trame_Ordre* et *Trame_Validation*, issues de chaque calculateur, ne vont pas directement aux FCRM : elles sont en fait d'abord regroupées ou « concaténées » au niveau du réseau pour simplifier les traitements au niveau des FCRM. Ce principe de concaténation des trames a déjà été évoqué pour l'architecture à vote massif.

Au final, nous ne précisons ici que les *Trame_Validation* : chaque calculateur compare ses ordres en loi normale, et en loi directe, à ceux des ordres des autres calculateurs qu'il surveille en tant que valideur. Pour simplifier les traitements au niveau des FCRM, on considère un format unique à tous les calculateurs pour les trames de validation avec le format et les valeurs booléennes suivantes :

Adresse destinataire	Adresse source	Type de la trame	Validité loi N FCCP1	Validité loi D FCCP1	Validité loi N FCCP2	Validité loi D FCCP2	Validité loi D FCCS1	Validité loi N FCCP3	Validité loi D FCCP3	Validité loi N FCCP4	Validité loi D FCCP4	Validité loi D FCCS2

- la valeur 11 : je valide l'ordre
- la valeur 01 : je ne valide pas l'ordre
- la valeur 00 : je ne suis pas valideur de l'ordre

Voici un exemple de *Trame_Validation* envoyée par le calculateur FCCP2 dans son rôle de valideur en loi normale pour les calculateurs FCCP1 et FCCP3, et en loi directe pour le calculateur FCCS1 :

FCRM	FCCP2	Trame de validation	11	01	00	00	11	11	00	00	00	00

2) Architecture fonctionnelle de base

Le logiciel fonctionnel de chacun de ces calculateurs peut schématiquement être réparti en trois catégories (comme l'illustre la Figure V.2) :

a) le calcul des lois de pilotage pour déterminer les ordres en tant que calculateur maître,
b) le calcul des fonctions de validation pour déterminer les validations en tant que calculateur valideur (par simple comparaison),
c) les surveillances (cf. chapitre III, section 2.1.2.3).

Un point capital à relever est que le calcul des validations dans chaque calculateur se fait par rapport aux ordres du cycle précédent. En effet, à chaque cycle de calcul, chaque calculateur transmet ses ordres (loi normale et loi directe) sur le réseau de communication. Les ordres d'un même cycle de tous le calculateurs seront regroupés dans une seule trame au niveau de réseau qui la transmet ensuite vers les actionneurs et vers les calculateurs. Ainsi, un calculateur valideur calcule des validations pour des ordres (l'ordre du valideur et les ordres des autres calculateurs) d'une même trame et qui ont été calculés dans un même cycle.

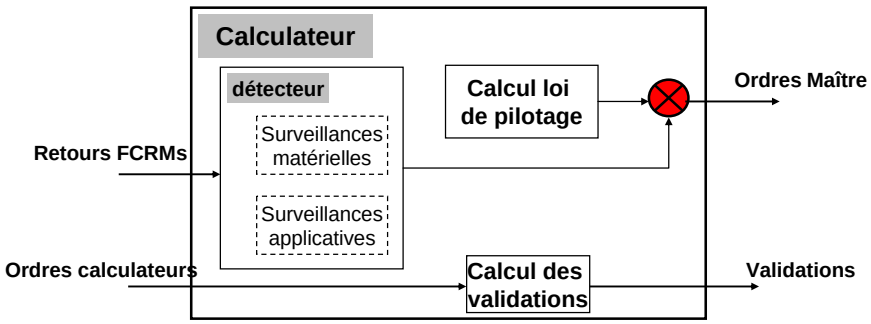


Figure V.2 Schéma fonctionnel de base d'un calculateur

3) Principe de fonctionnement et logiques spécifiques (niveau calculateur)

Chaque calculateur émet systématiquement ses ordres vers les actionneurs dont il est maître et émet des validités vers chaque calculateur dont il est valideur.

La définition (ou le choix) de « quel calculateur joue quel rôle (maître ou valideur) à quel moment » est faite de manière statique, donc prédéfinie. Par la suite, nous considérons la configuration nominale du Tableau V.3 pour les calculateurs valideurs. Chaque ligne indique pour un calculateur, son type de logiciel et de matériel, puis quel sera son ou ses calculateurs valideurs. Par exemple, pour le calculateur FCCP1, ses valideurs sont FCCP2 ou FCCP4, ou FCCP3 en cas de reconfiguration de ce dernier suite à la perte de FCCP4 et FCCP2.

Rappelons que l'architecture est composée de 6 calculateurs « simplex » dont 4 unités avec un matériel H1 exécutant des fonctions évoluées (FCCP – primaire) en loi normale et des fonctions simples en loi directe, et 2 unités avec un matériel H2 exécutant des fonctions simples (FCCS - secondaire) en la loi directe seulement. Et on considère aussi 3 logiciels différents pour l'ensemble des calculateurs : 2 variantes dissimilaires pour le logiciel des lois normale et directe des calculateurs primaires (logiciel A et logiciel B), et une seule variante pour le logiciel de la loi directe des calculateurs secondaires (logiciel C).

Calculateur Maître	*Logiciel*	*Matériel*	*Calculateur valideur*
FCCP1	logiciel A	H1	FCCP2 en loi normale ou FCCP4 en loi normale ou FCCP3 en loi normale si reconfiguration
FCCP3	logiciel B	H1	FCCP2 en loi normale ou FCCP4 en loi normale
FCCP2	logiciel C	H1	FCCP4 en loi normale si reconfiguration
FCCS1	logiciel C	H2	FCCP2 en loi directe ou FCCP1 en loi directe ou FCCS2 en loi directe
FCCS2	logiciel C	H2	FCCP4 en loi directe ou FCCP3 en loi directe ou FCCS1 en loi directe

Tableau V.3 - Configuration nominale de calculateurs valideurs

V.2.2 Les électroniques locales FCRM

V.2.2.1. Architecture matérielle et logicielle

Comme pour l'architecture à vote massif, dans l'architecture à priorités, chaque actionneur est équipé d'au moins une électronique locale avec des capacités pour : communiquer sur le réseau numérique, réaliser des opérations de choix et piloter les actionneurs. Et comme pour l'architecture à vote massif, un FCRM est composé d'une voie COM et d'une voie MON.

V.2.2.2. Description fonctionnelle

1) Interface système

Les deux voies COM et MON de chaque actionneur implémentent la même logique de priorité. Les logiques de priorité tiennent compte de la dégradation des lois (passage de loi normale en loi directe). Les électroniques locales de chaque actionneur produisent et consomment les flux de données décrits dans le Tableau V.4.

Ces flux sont reçus et transmis via le même réseau de communication (dans une ou plusieurs trames de données) selon les protocoles de communication retenus.

Flux de données	***Description***	***Type***	***Flux entrant (source)***	***Flux sortant (destinataire)***
Trame_réseau_maître	Ordres des calculateurs maîtres.	Trame_Réseau_ Maître	Calculateurs	
Ordres_valideur	Validation des calculateurs valideurs.	Trame_Réseau_ Validation	Calculateurs	
Ordre_actionneur	Le FCRM (voie COM) réalise l'asservissement local de l'actionneur.			Actionneurs
Retour_FCRM	Résultat du vote et les acquittements FCRM (celui du COM et celui du MON) vers les calculateurs.	Trame FCRM		Calculateurs

Tableau V.4 - Flux de données d'un FCRM

On se limitera ici à évoquer deux aspects significatifs révélés par ce tableau : la ***Trame_Réseau_Maître*** (respectivement ***Trame_Réseau_Validation***) reçue par chaque FCRM est une trame unique qui est le résultat de la concaténation par le réseau des *Trame_Ordre* (respectivement *Trame_Validation*) du Tableau V.3 issues de chacun des calculateurs maîtres (respectivement valideurs).

2) Architecture fonctionnelle de base

Les ordres et les validations sont calculés par tous les calculateurs. Le choix de l'ordre à appliquer à l'actionneur est le résultat d'un vote (portant sur tous les ordres reçus) réalisé par les électroniques locales. Dit autrement, les électroniques locales « sélectionnent » les ordres qui seront appliqués par les voies COM des électroniques locales associées à chaque actionneur. La voie MON surveille la voie COM et a le pouvoir de désactiver le COM et/ou l'actionneur en cas de désaccord.

Le logiciel fonctionnel des FCRM peut schématiquement être réparti en deux catégories :

1) logique de priorités,
2) logique de validation.

Le modèle de la Figure V.3 présente le schéma fonctionnel de la voie COM.

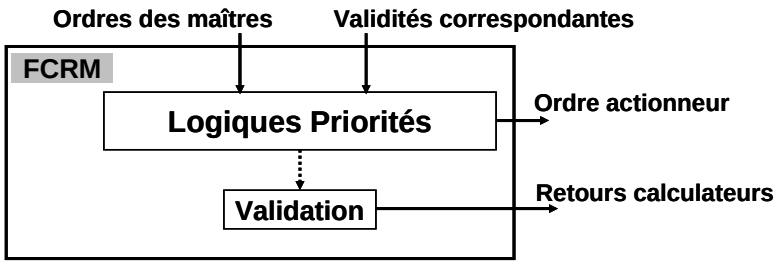


Figure V.3 - Schéma fonctionnel de la voie COM d'un FCRM

La différence avec l'architecture fonctionnelle d'un FCRM dans l'architecture à vote massif est une des plus significatives entre les deux architectures. Il s'agit donc là d'un des aspects importants au niveau des modélisations faites en vue des simulations de validation, en particulier au niveau des simulations sur la robustesse.

Or, comme nous ne présenterons pas, pour cette architecture à priorités, toutes les modélisations et simulations réalisées dans un chapitre spécifique (contrairement à ce qui avait été fait pour l'architecture à vote massif), nous présentons ici directement, sur la Figure V.4, le modèle Simulink établi pour un FRCM dans cette architecture à priorités.

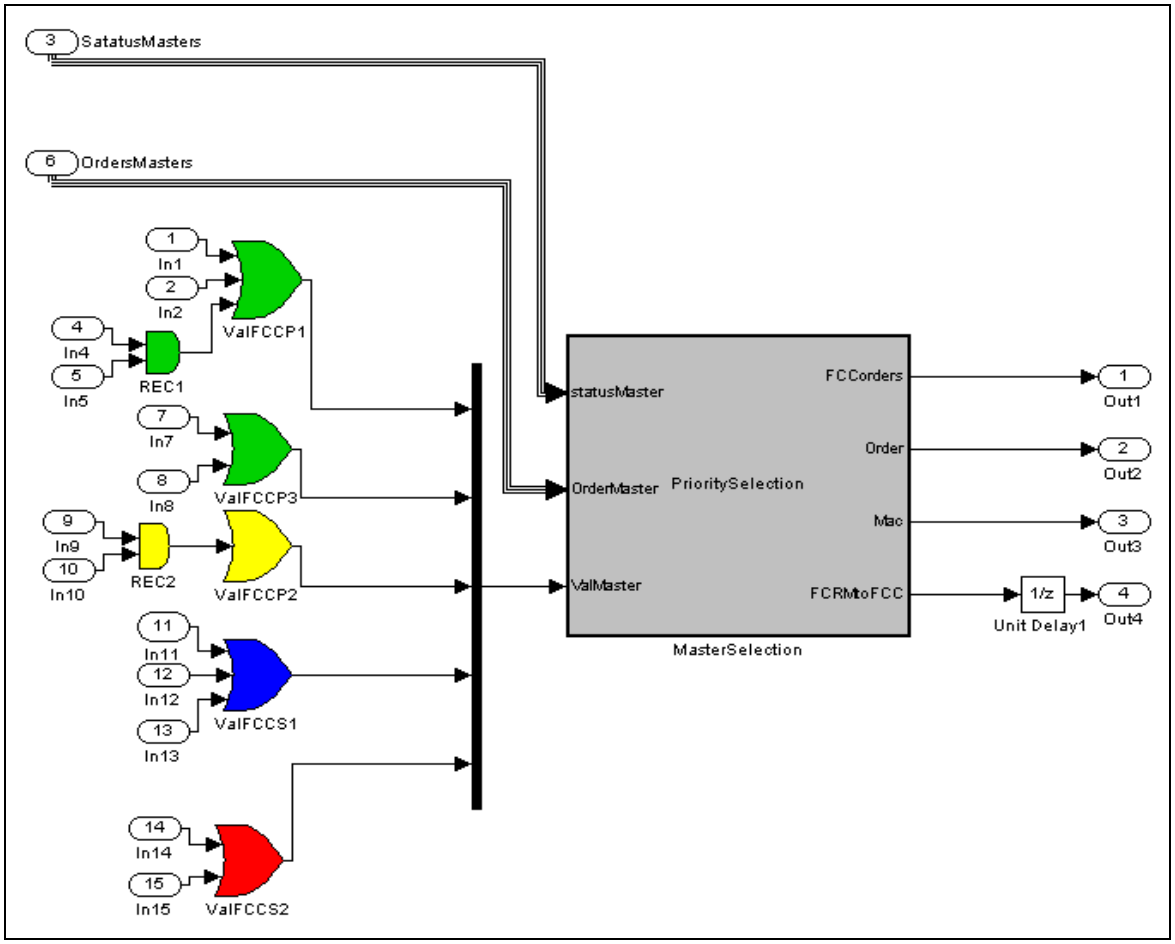


Figure V.4 - Modèle Simulink pour un FCRM

3) Principe de fonctionnement et logiques spécifiques (niveau FCRM)

Comparativement à l'architecture à vote massif, le traitement FCRM est très simple. Il suffit de lire l'ordre du calculateur FCCP1 (calculateur maître qui possède la priorité la plus élevée) de la trame (Trame_Réseau_Maître) et de vérifier sa validité sur les champs (Validité Loi N FCCP1) des calculateurs valideurs selon le Tableau V.1. Dans le cas où tous les champs (Validité Loi N FCCP1) de tous les calculateurs valideurs sont à (01), le FCRM doit refaire la même chose avec l'ordre du calculateur FCCP3 selon le Tableau V.3. Le passage d'un maître à un autre doit être échangé entre les deux voies COM et MON du FCRM.

V.3 ARCHITECTURE A PRIORITES : VALIDATION

Le chapitre IV précédent a été consacré à la présentation et la justification de nos choix d'environnement de modélisation et de simulation pour les analyses de sécurité (langage Altarica sous l'environnement OCAS) et les analyses de robustesses (sous l'environnement Matlab/Simulink). Ces éléments restent inchangés pour l'architecture à priorités. Ils ne seront donc pas repris ici pour éviter trop de redites inutiles.

Pour ce qui est des résultats des simulations des modélisations en AltaRica qui ont été réalisées pour l'architecture à priorités, les résultats obtenus sont identiques à ceux obtenus pour l'architecture à vote massif, pour les mêmes types de pannes qu'on a injecté dans les modèles des deux architectures. Et la conclusion est la même pour les analyses de robustesse.

De plus faute de temps, la mise en forme des modèles (icône sous OCAS et mise à jour des noms des variables sous Simulink) n'a pas pu être totalement achevée, pour être conforme au premier modèle qui a été établi pour l'architecture à vote massif.

Pour toutes ces raisons, nous nous limitons dans cette section à la présentation uniquement de quelques éléments sous Matlab/Simulink :

1) un modèle d'injection de panne calculateur donné sur la Figure V.5,
2) quelques résultats de simulation pour l'architecture à priorités qui montre le changement du calculateur maître et la dégradation des lois en fonction des pertes des calculateurs qu'on a injecté dans le modèle à différents instant de simulation.
3) une implémentation sous Matlab/Simulink de l'architecture à priorités, donnée sur la Figure V.7.

Ainsi, le modèle d'injection de panne de la Figure V.5 associé au modèle de chaque calculateur (Figure V.7) nous permet de simuler des pannes calculateur à différents instants et de visualiser le comportement de l'actionneur en présence de ces pannes.

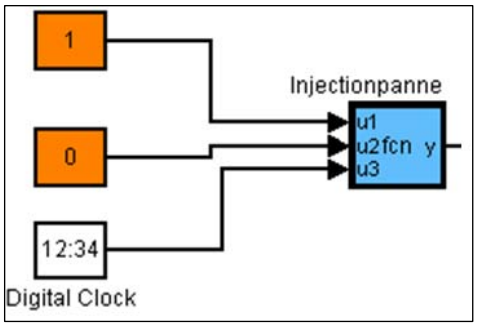


Figure V.5 - Modèle Simulink d'injection de panne calculateur

La Figure V.6 montre quelques résultats de simulation pour l'architecture à priorités associées aux sorties du FCRM suivantes :

- Ordres des calculateurs maîtres et valideurs avec un zoom.
- Lois disponibles (2 = loi normale et 1 = loi directe)
- Numéros des calculateurs maîtres avant et après l'injection des pannes, en précisant que les notations sont les suivantes : FCCP1= 1 ; FCCP2 =4 ; FCCP3 =2 ; FCCP4= 6 ;
- FCCS1=3 ; FCCS2=5

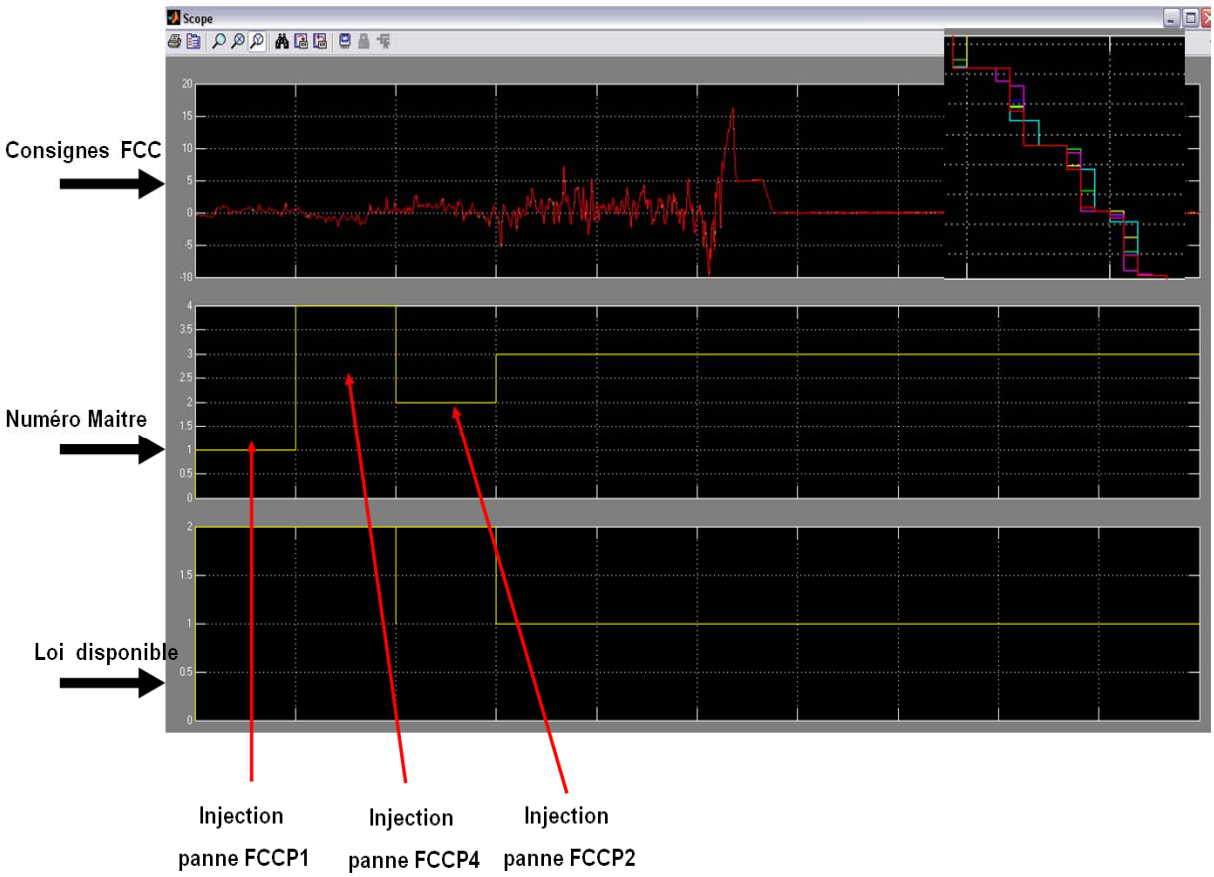


Figure V.6 - Résultats de simulation : changement du calculateur maître et la dégradation des lois

Le diagramme montre que suite, à la perte de calculateur FCCP1 à l'instant t = 50, le calculateur FCCP3 devient le nouveau maître en loi normale. Le contrôle de l'actionneur passe en loi directe à l'instant t = 100 d'une façon transitoire suite à la perte des calculateurs FCCP1 et FCCP3. Mais, après une configuration logicielle, le nouveau maître est le calculateur FCCP2 en loi normale.

Le contrôle passe en loi directe suite à la perte de tous les calculateurs primaires à partir de l'instant t = 150.

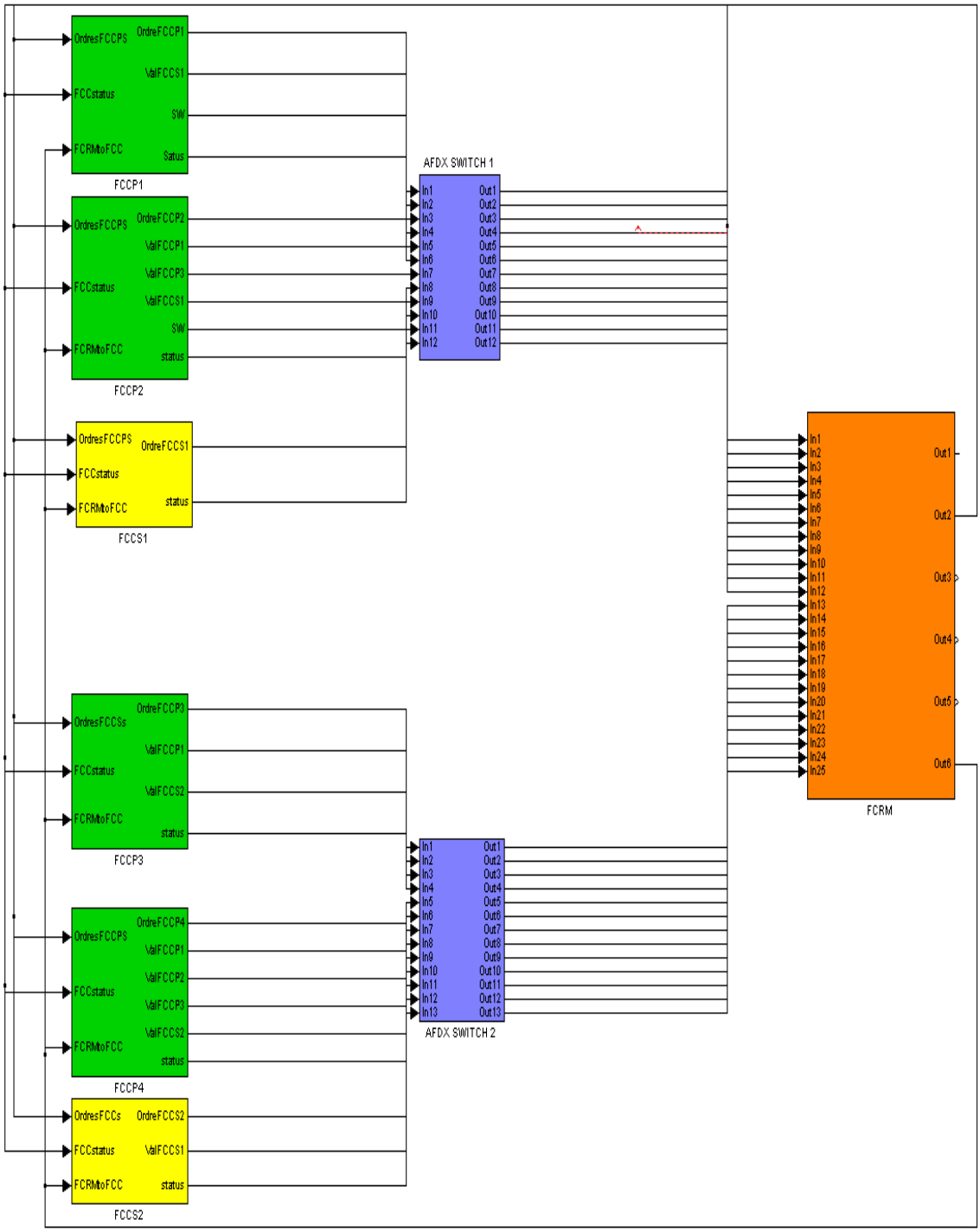


Figure V.7 - Modèle Simulink de haut niveau de l'architecture à priorités

V.4 CONCLUSION

Ce chapitre nous a permis de décrire l'architecture à priorités. Il s'agit d'une architecture distribuée, multi-maître et tolérante aux fautes, différente par rapport à l'existant présenté dans l'état de l'art, et à l'architecture à vote massif présenté dans le chapitre III. Les différences viennent de nouvelles répartitions des fonctions intelligentes entre l'avionique centrale (calculateurs CDVE) et l'avionique déportée (électroniques locales des actionneurs) ; elle a pour avantage de consommer une moins grande quantité de ressources par rapport aux architectures conventionnelles tout en satisfaisant un même objectif de sécurité/disponibilité.

Contrairement à l'architecture à vote massif, les comparaisons des ordres calculateurs ont été conservées au niveau des calculateurs ; l'étape de décision finale du choix des ordres étant déportée au niveau de l'actionneur.

Même s'il s'agit d'une redite, il est important de rappeler qu'il existe un certain déséquilibre dans la thèse entre l'option à vote massif et l'option à priorité. En effet, historiquement, nous avons commencé par étudier l'option à vote massif en premier ce qui a soulevé certains problèmes (protocole d'échange, définition de surveillances, modélisation et implémentation des fonctions de choix). Les solutions apportées sont valables aussi bien dans le cas de l'option 1 que celui de l'option 2 ; cela nous a permis de gagner beaucoup de temps pour l'option 2.

CONCLUSIONS ET PERSPECTIVES

L'objectif des travaux de thèse présentés dans ce mémoire, centrés sur les architectures du système de *Commandes De Vol Électriques (CDVE)* dans les avions civils, était d'étudier dans quelles mesures les évolutions technologiques dans les domaines des communications numériques et des capteurs/actionneurs intelligents pouvaient être prises en compte dans l'évolution de ce type de système chez Airbus, quitte à remettre totalement en cause leurs architectures actuelles.

Le système de CDVE est embarqué, réparti, temps réel et critique, puisqu'il gère la trajectoire de l'avion. Ce système (qui est tout le contraire d'un système grand public, peu onéreux et à courte durée de vie) évolue à un rythme bien plus lent que dans la majorité des domaines, car les nouvelles méthodes et technologies intégrées à chaque nouvelle évolution doivent être très bien maîtrisées, et donc bien souvent déjà largement éprouvées dans d'autres domaines.

Les architectures actuelles ont montré leurs mérites et leurs avantages depuis l'A320. Mais la recherche de nouvelles pistes pour l'optimisation et la modernisation reste une priorité chez Airbus, et donc, le système de CDVE se doit d'évoluer encore, mais toujours en respectant les mêmes principes : amélioration de la sécurité et des performances, et réduction des coûts.

Aujourd'hui, la modernisation des avions civils est étroitement liée à la généralisation du recours aux communications numériques et à des équipements de plus en plus intelligents, et à la distribution de l'intelligence système entre ces différents équipements. La maturité de ces domaines technologiques (acquises dans d'autres domaines d'application) rend aujourd'hui inconcevable de ne pas les prendre en considération pour les futurs systèmes de CDVE.

Pour parvenir à notre objectif, le travail réalisé dans cette thèse s'est articulé autour de quatre types de contributions : une analyse de l'existant, la proposition d'une nouvelle démarche de conception d'architecture, la proposition de deux nouvelles architectures optimales, distribuées et tolérantes aux fautes, et leur modélisation et la validation par simulation.

****

Naturellement, il a fallu commencer par une analyse en profondeur des solutions actuelles qui régissent ces systèmes chez Airbus et d'autres grands avionneurs. Cette analyse a permis de bien identifier et de comprendre, d'une part, les aspects fonctionnels et non fonctionnels (pour les commandes de vol) et les solutions mises en place, et d'autre part, les exigences auxquelles doivent répondre ces solutions : 1) exigences règlementaires de l'aviation civile ; 2) exigences liées aux précautions forfaitaires et aux retours d'expériences des avions en service ; 3) exigences liées au caractère embarqué et temps-réel ; 4) exigences économiques, qui ne cessent de croître.

Cette première analyse a produit plusieurs types d'apports. Au-delà du besoin premier qui était la nécessaire compréhension de ces aspects pour engager plus avant mon travail, cette analyse a conduit à identifier des exigences, puisque, bien qu'elles soient connues et maîtrisées, le niveau actuel de leur représentation et de leur synthèse devenait mal adapté à la taille de la communauté qui doit en avoir connaissance.

Mais l'apport principal de cette analyse a été de faire apparaître, dans les différentes architectures étudiées, une apparente « *sur-redondance* » des équipements. En effet, le niveau constaté de redondance était justifié par les moyens actuels de répondre au souci d'une démonstration de sécurité (dictée par les réglementations de certification) et au souci de ne pas proposer un produit taillé au plus juste (dicté par les compagnies aériennes). Mais placé en regard des possibilités offertes par les nouvelles technologies de communication et d'équipements intelligents, ce niveau de redondance semble alors potentiellement trop élevé, d'où l'expression « sur-redondance ».

Et l'analyse que nous avons menée par ailleurs sur les caractéristiques et la maturité de ces nouvelles technologies a confirmé qu'elles offraient de nouveaux axes d'amélioration et d'optimisation, confirmant donc l'importance de l'objectif initial des travaux de cette thèse.

Finalement, les résultats de cette première phase du travail ont conduit à remettre à plat les besoins de traitement et de communication du système, et à prendre pour base de la définition de nouvelles architectures, les convictions suivantes, fondamentalement nouvelles :

1) utilisation de calculateurs simplex mutuellement indépendants, et donc abandon des principes COM/MON des calculateurs Airbus ou Triplex des calculateurs Boeing,
2) utilisation d'actionneurs intelligents pour permettre de nouvelles répartitions des fonctions intelligentes entre les électroniques des actionneurs et les calculateurs, avec reprise pour ces actionneurs du concept de COM/MON, abandonné sur les calculateurs,
3) utilisation des communications intégralement numériques pour l'ensemble du système, en mode full duplex pour que tous les abonnées puissent mutuellement interagir,
4) réduction du nombre de calculateurs et de leurs dépendances.

À partir de là, il a fallu définir une nouvelle démarche de conception en phase amont pour le dimensionnement du système. La démarche, en partant d'une architecture purement fonctionnelle à un seul calculateur, est basée sur l'injection incrémentale des exigences identifiées, avec une évaluation de l'architecture à chaque étape d'injection, pour déterminer au mieux le nombre de calculateurs nécessaires pour satisfaire l'ensemble des exigences.

La démarche est nouvelle parce que l'analyse rapide que nous avons menée sur les méthodes existantes montre qu'elles sont mal adaptées aux besoins du cadre de nos travaux (ce n'est pas que ces méthodes ne pourraient pas être adaptées ou globalisées, mais ce travail là relève d'enjeux qui sortent du cadre de la thèse). Nous avons donc opté pour mettre en place une nouvelle démarche de conception simple et rigoureuse, qui peut aider les concepteurs système, en phase amont de conception, à prévoir au mieux les redondances nécessaires pour dimensionner des nouvelles architectures. Elle est générique pour tout type d'avion et pour tout type de systèmes à exigences. On peut espérer que cette démarche contribuera à améliorer l'abstraction de l'aspect empirique pour le dimensionnement des systèmes critiques en général, et celui de commandes de vol en particulier.

La démarche incrémentale proposée est un apport significatif, et est en tout cas le fondement de la définition de nos propositions de nouvelles architectures optimales (avec notamment moins de matériel et de logiciel mis en jeu) pour relier les calculateurs aux actionneurs des commandes de vol, et satisfaisant les objectifs de sécurité et disponibilité, le tout, avec les hypothèses citées plus haut : des calculateurs simplex, des actionneurs intelligents et des technologies de communications numériques.

Ainsi, à ce jour, deux nouvelles architectures de système de CDVE ont déjà pu être proposées : une architecture dite à *« vote massif »*, et une dite *« à priorités »*, avec pour différences les types de mécanismes de gestion de la redondance mise en place, pour la validation et la sélection des ordres à appliquer aux gouvernes :

1) dans l'architecture *à vote massif*, la sélection repose uniquement sur les actionneurs,
2) dans l'architecture *à priorités*, la sélection est partagée entre calculateurs et actionneurs.

Les deux architectures sont basées sur 6 calculateurs simplex. Pour respecter le principe de dissimilarité de l'état de l'art actuel, deux types de matériel sont retenus : un type pour les 4 calculateurs primaires et un autre pour les 2 calculateurs secondaires. Au niveau du logiciel, il y a deux variantes pour le logiciel primaire pour le pilotage automatique, la loi normale et la loi directe, et un logiciel secondaire pour la loi directe uniquement. Et, autre nouveauté importante, les calculateurs sont reconfigurables, pour supporter l'introduction du concept de *reconfiguration logicielle dynamique* en cours du vol pour tolérer des pannes des calculateurs, et augmenter la disponibilité du système.

Enfin, ces deux architectures ont été validées (sur un cas d'étude : la commande en tangage de l'avion), en réalisant des modélisations et des simulations dans deux contextes différents :

1) *des analyses de sécurité* (à l'aide du langage Altarica sous l'environnement OCAS) pour vérifier le comportement des architectures en présence d'erreurs (fonctionnement erroné) ou de pannes et à s'assurer que les objectifs quantitatif et qualitatif de sécurité soient bien atteint,
2) *des analyses de robustesse* (sous l'environnement Matlab/Simulink) pour, notamment, analyser les aspects temporels, plus particulièrement le comportement vis-à-vis des asynchronismes liés à l'absence d'horloge commune et de mécanismes de synchronisation des horloges.

Un effet de bord de l'utilisation du langage formel AltatRica pour valider ces architectures a été de permettre d'évaluer plus concrètement l'intérêt des méthodes basées sur des modèles pour l'évaluation de la sûreté de fonctionnement des systèmes multi-composants et critiques.

****

En proposant deux nouvelles architectures, une comparaison est inévitable et conduit à dégager plusieurs points. Pour les points communs dans ces deux architectures, tous les actionneurs possèdent une intelligence importante, tous les calculateurs communiquent en numériques avec tous les actionneurs. L'architecture à vote massif et l'architecture à priorités reposent sur la même architecture matérielle (calculateurs, actionneurs et le réseau de communication) et sur la même architecture logicielle.

Mais les deux architectures présentent des différences majeures dans la distribution de l'intelligence entre les actionneurs et les calculateurs : le niveau de l'intelligence des actionneurs et le principe de consolidation des ordres des calculateurs sont différents :

- Pour ce qui est de la passivation des pannes, les deux architectures sont équivalentes puisqu'elles tolèrent le même nombre de pannes. Par contre, en termes de rapidité de réaction aux pannes, l'architecture à priorités est légèrement moins performante que l'architecture à vote massif au niveau de détection des pannes. En effet, elle nécessite plus de temps pour la détection de pannes : un cycle de calcul de plus (de l'ordre de 10 ms).
- En termes de reconfiguration, avec l'architecture à vote massif, il y a une difficulté pour la reconfiguration logicielle (pilotée par les actionneurs), qui pourrait amener à la perte de la loi normale. Ce risque peut être limité grâce à une logique de validation des requêtes de reconfiguration robuste au niveau des calculateurs.
- En termes de comparaison des ordres calculateurs, dans l'architecture à priorités elle est réalisée au niveau des calculateurs eux-mêmes, ce qui sera plus « simple » à implémenter que des comparaisons et des consolidations au niveau des actionneurs comme cela est fait dans l'architecture à vote massif.
- En termes de complexité des logiques de choix, l'architecture à priorités est plus favorable, et cela, quelle que soit la logique de priorités à mettre en œuvre dans les actionneurs. Notamment, en identifiant en permanence un calculateur maître, elle permet de simplifier le problème de la synchronisation des objectifs au sein des diverses redondances.

Il ressort de la revue précédente qu'une analyse plus détaillée des deux architectures et une identification plus fine des critères de comparaisons sont nécessaires pour arriver à un comparatif plus évident entre les deux options d'architecture que nous avons proposées.

Au final, l'objectif des travaux a donc été largement atteint : proposer des architectures plus économes en ressources matérielles, et qui répondent aux mêmes exigences de sécurité que les architectures actuelles, et couvrent même des exigences en fiabilité opérationnelle plus élevées. Les architectures proposées ont fait l'objet de dépôts de demande de brevets : [Sghairi *et al.* 2009b] pour l'architecture à vote massif et [Sghairi *et al.* 2009a] pour celle à priorités. Et outre ces deux dépôts, les travaux réalisés ont donné lieu à : une contribution à ouvrage [Sghairi *et al.* 2008a], un article dans une revue internationale [Sghairi *et al.* 2009c], deux communications dans des conférences internationales [Sghairi *et al.* 2008b, Sghairi *et al.* 2009d], deux communications dans des conférences nationales [Sghairi 2008, Sghairi *et al.* 2008c] et un poster [Sghairi *et al*. 2008d].

****

Si l'on se projette maintenant vers le futur, au regard des perspectives liées aux travaux réalisés, plusieurs aspects sont à présenter.

En se plaçant d'abord sous le strict point de vue des résultats de nos travaux et de leur exploitation, ces résultats ont permis de mettre en lumière plusieurs nouvelles pistes d'amélioration au sein du département commandes de vol Airbus. En fait, ces résultats sont en passe d'être élargis pour la définition d'architectures préliminaires pour les prochains programmes Airbus. Pour cela, plusieurs extensions (et/ou approfondissement) sont possibles à partir du travail présenté dans cette thèse.

- Une première extension concerne la spécification de certains aspects techniques qui sont restés d'ordre général, tels que notamment : la synchronisation entre les différents calculateurs et actionneurs, le temps de reconfiguration logicielle et le verrouillage, ainsi que le temps de confirmation des pannes.
- Une deuxième extension intéressante est l'analyse des risques techniques associés aux nouvelles architectures pour une application A30X et la mise en place des mécanismes pour couvrir ces risques en accord avec les exigences de ce nouveau programme. En effet, le nouvel avion A30X doit répondre à des exigences non traitées dans le cadre de cette thèse : types et nombre de technologies réseau, installations électriques, etc.
- Une troisième extension concerne l'implémentation en SCADE des logiques spécifiques aux architectures proposées dans la thèse pour des simulations en laboratoire, et la réalisation d'un démonstrateur avec des équipements réels en banc de tests.

En se plaçant maintenant sous un point de vue plus large, et après trois ans d'une expérience personnelle à la fois académique et industrielle, j'aurai tendance à dire qu'une action pertinente à mener serait l'établissement d'un cursus de formation destinée aux chercheurs désirant travailler sur ces types de systèmes et de problématiques industrielles. Les commandes de vol électriques présentent plusieurs pistes de recherche académiques, mais aussi plusieurs aspects plus empiriques, qui ne font pas partie des formations initiales classiques et doivent faire l'objet d'un apprentissage maison. Plus particulièrement, les méthodes empiriques ou maison concernant une partie de la gestion de la redondance, la détection des pannes, et la reconfiguration de système en fonction de ses pannes utilisées dans la pratique chez Airbus donnent très de bons résultats, donc il est très intéressant d'accentuer leur formalisation pour mieux les mettre en corrélation avec les théories correspondantes.

L'acquisition de la culture commandes de vol électrique permettra une meilleure compréhension des problématiques industrielles et de les rapprocher avec le monde académique.

RÉFÉRENCES BIBLIOGRAPHIQUES

[Akerlund *et al.* 2006] O. Akerlund, P. Bieber, E. Boede, M. Bozzano, M. Bretschneider, C. Castel, A. Cavallo, M. Cifaldi, J. Gauthier, A. Griffault, O. Lisagor, A. Luedtke, S. Metge, C. Papadopoulos, T. Peikenkamp, L. Sagaspe, C. Seguin, H. Trivedi et L. Valacca, "ISAAC, a Framework for Integrated Safety Analysis of Functional, Geometrical and Human Aspects", *Proceedings of ERTS 2006*, Toulouse, 25-27 janvier 2006, 11 p.

[Altarica] http://altarica.labri.fr/

[Andrade & Tenning 1992] L. Andrade et C. Tenning, "Design of Boeing 777 Electric System", *IEEE Aerospace and Electronic Systems Magazine*, vol. 7, issue 7, juillet 1992, pp. 4-11.

[Arlat *et al*. 2000] J. Arlat, J.-P. Blanquart, T. Boyer, Y. Crouzet, M.-H. Durand, J.-C. Fabre, M. Founau, M. Kaaniche, K. Kanoun, P. Le Meur, C. Mazet, D. Powell, F. Scheerens, P. Thevenod-Fosse et H. Waeselynck, "Composants logiciels et sûreté de fonctionnement - Intégration de COTS", 158 p., Hermès Science Publications, Paris, 2000.

[Avizienis 1985] A. Avizienis, "The N-Version Approach to Fault-Tolerant Software", *IEEE Transactions on Software Engineering*, vol. SE-11, issue 12, décembre 1985, pp. 1491-1501.

[Avizienis *et al.* 2004] A. Avizienis, J. Laprie, B. Randell et C. Landwehr, "Basic Concepts and Taxonomy of Dependable and Secure Computing", *IEEE Transactions Dependable Secure Computing*, vol. 1, issue 1, janvier 2004, pp. 11-33.

[Barbaria 2008] K. Barbaria, "Conception d'une architecture et de composants prouvés pour intergiciels spécifiques aux différentes familles de systèmes critiques", doctorat de l'Ecole Nationale Supérieure des Télécommunications, 15 septembre 2008. disponible via : http://pastel.paristech.org/4308/

[Bayart *et al*. 2005] M. Bayart, B. Conrard, A. Chovin, M. Robert, "Capteurs et actionneurs intelligents", Techniques de l'Ingénieur, ISSN 1632-3831, Mars 2005, vol. S2, pp. S7520.1-S7520.16.

[Ben Chehida & Auguin 2002] K. Ben Chehida et M. Auguin, "HW/SW Partitioning Approach For Reconfigurable System Design", *Proceedings of the 2002 International Conference on Compilers, Architecture and Synthesis for Embedded Systems (CASES'02)*, Grenoble, 8-11 octobre 2002, pp. 247-251.

[Ben Chehida 2004] K. Ben Chehida, "Méthodologie de Partitionnement Logiciel/Matériel pour Plateformes Reconfigurables Dynamiquement", *Doctorat de l'Université de Nice-Sophia Antipolis*, 30 novembre 2004, 204 p.
http://hal.archives-ouvertes.fr/docs/00/04/79/23/PDF/tel-00008931.pdf

[Ben Chehida *et al.* 2002] K. Ben Chehida, P. Guitton-Ouhamou, C. Belleudy et M. Auguin, « A Multiobjective Hardware-Software Partitioner for Dynamically Reconfigurable System Design », *WSEAS International Conference on Information Security, Hardware/Software Codesign, E-Commerce and Computer Networks*, Copacabana, Rio de Janeiro, Brésil, 15-17 octobre 2002.

[Benveniste & Berry 1991] A. Benveniste et G. Berry, "The Synchronous Approach to Reactive and Real-Time Systems", *Proceedings of the IEEE*, vol. 79, n° 9, septembre 1991, pp. 1270-1282.

[Bérard *et al*. 2003] B. Bérard, P. Coupoux, Y. Crouzet, P. David, Y. Garnier, S. Goiffon, G. Mariano, V. Nicomette, L. Planche, I. Puaut, J.-M. Tanneau, H. Waeselynck, "*Logiciel libre et sûreté de fonctionnement – cas des systèmes critiques"*, (Eds P. David et H. Waeselynck), Hermès Science, ISBN 2-7462-0727-3, 2003, pp. 82-88.

[Bernard 2009] R. Bernard, "Analyses de sûreté de fonctionnement multi-systèmes", doctorat de l'Université Sciences et Technologies de Bordeaux 1, 23 novembre 2009, 168 p. disponible via : http://tel.archives-ouvertes.fr/tel-00441310/fr

[Bieber *et al.* 2004] P. Bieber, C. Bougnol, C. Castel, J.-P. Heckmann, C. Kehren, S. Metge et C. Seguin, "Safety Assessment with AltaRica - Lessons Learnt Based on Two Aircraft System Studies", *Proceedings 18th IFIP World Computer Congress, Topical Day on New Methods for Avionics Certification*, Toulouse, 26 août 2004, 6 p.

[Blake & Liguori 2001] M.B. Blake et P. Liguori, "An Autonomous Decentralized Architecture for Distributed Data Management and Dissemination", *IEICE/IEEE Joint Special Issue on Autonomous Decentralized Systems and Systems' Assurance, IEICE Transactions on Information and Systems*, vol. E84-D, n. 10, octobre 2001, pp. 1394-1398.

[Blanc *et al.* 2009] S. Blanc, A. Bonastre et P.J. Gil, "Dependability Assessment of By-Wire Control Systems using Fault Injection", *Journal of Systems Architecture*, vol. 55, issue 2, février 2009, pp. 102-113.

[Bouissou & Bon 2003] M. Bouissou et J.-L. Bon, "A New Formalism that Combines Advantages of Fault-Trees and Markov Models: Boolean Logic Driven Markov Processes", *Reliability Engineering & System Safety*, vol. 82, issue 2, novembre 2003, pp. 149-163.

[Bouissou & Seguin 2006] M. Bouissou et C. Seguin, "Comparaison des langages de modélisation Altarica et FIGARO", *Actes du 15ème colloque de fiabilité et maintenabilité (LMU'15)*, Lille, 9-12 octobre 2006, 8 p.

[Boyer & Moore 1991] R.S. Boyer et J.S. Moore, "MJRTY - A Fast Majority Vote Algorithm", *Automated Reasoning: Essays in Honor of Woody Bledsoe*, Automated Reasoning Series, Kluwer Academic Publishers, Dordrecht, The Netherlands, 1991, pp. 105-117.
disponible via : www.cs.utexas.edu/users/boyer/mjrty.ps.Z

[Brajou & Ricco 2005] F. Brajou et P. Ricco, "AFDX-Based Flight Test Computer Concept", *IEEE Instrumentation & Measurement Magazine*, vol. 8, issue 3, août 2005, pp. 55-58.

[Briere & Traverse 1993] D. Briere et P. Traverse, "AIRBUS A320/A330/A340 Electrical Flight Controls - A Family of Fault-Tolerant Systems", *Proc. 23th International Symposium on Fault-Tolerant Computing (FTCS-23)*, Toulouse, 22-24 juin 1993, pp. 616-623.

[Brière *et al.* 1995] D. Briere, D. Ribot, D. Pilaud et J.-L. Camus, « Method and Specification Tools for Airbus Onboard Systems », *Microprocessors and Microsystems*, Elsevier, vol. 19, n. 9, novembre 199 , pp. 511-515.

[Cabaret *et al.* 2008] R. Cabaret, F. David-Tupinier, R. Andreoletti, "Method of Data Integrity Control in an AFDX Network", Brevet Airbus US2008/0239973, octobre 2008, 6 p.

[Carpenter et al. 1994] T. Carpenter, K. Driscoll, K. Hoyme et J. Carciofini, "ARINC 659 Scheduling: Problem Definition", *Proceedings of the Real-Time Systems Symposium*, San Juan, Puerto Rico, 7-9 décembre 1994, pp. 165-169.

[Caspi & Salem 2000] P. Caspi et R. Salem, "Threshold and Bounded-Delay Voting in Critical Control Systems", *Proceedings of the 6th International Symposium on Formal Techniques in Real-Time and Fault-Tolerant Systems (FTRTFT 2000),* Pune, Inde, 20-22 septembre 2000, pp. 70-81.

[Caspi 2003] P. Caspi, "Automatique continue, automatique discrète, informatique industrielle : le triangle des Bermudes", *Colloque sur la modélisation des systèmes réactifs (MSR03),* conférence invitée, Metz 2003.
disponible via : http://www-verimag.imag.fr/~caspi/PAPIERS/msr03.pdf

[Caspi *et al.* 2001] P. Caspi, C. Mazuet et N. Reynaud Paligot, "About the Design of Distributed Control Systems: The Quasi-Synchronous Approach", *Proceedings of the 20th International Conference on Computer Safety, Reliability and Security (SAFECOMP 2001),* Budapest, Hungary, 25-28 septembre 2001, LNCS 2187, pp. 215-226.

[Cauffriez *et al.* 2004] L. Cauffriez, J. Ciccotelli, B. Conrard, M. Bayart and the members of the working-group CIAME, "Design of Intelligent Distributed Control Systems: a Dependability Point of View", Reliability Engineering & System Safety, vol. 84, issue 1, avril 2004, pp. 19-32.

[Chatha & Vemuri 2001] Chatha et R. Vemuri, "MAGELLAN: Multiway Hardware-Software Partitioning and Scheduling for Latency Minimization of Hierarchical Control-Dataflow Task Graphs", *Proceedings of the Ninth International Symposium on Hardware/Software Codesign (CODES 2001),* Copenhagen, Denmark, 25-27 Avril 2001, pp. 42-47.

[Dave *et al.* 1997] B.-P. Dave, G. Lakshminarayana et N.-K. Jha, "Cosyn: Hardware-Software Co-Synthesis of Embedded Systems", *Proceedings of the 34th Annual ACM IEEE Design Automation Conference (DAC'97),* Anaheim, CA, USA, 9-13 juin 1997, pp. 703-708.

[Delange *et al.* 2009] J. Delange, L. Pautet et P. Feiler "Validating Safety and Security Requirements for Partitioned Architectures", *Proceedings of the 14th Ada-Europe International Conference on Reliable Software Technologies*, Brest, 8-12 juin 2009, pp. 30-43.

[Deswarte *et al.* 1998] Y. Deswarte, K. Kanoun et J. Laprie, "Diversity against Accidental and Deliberate Faults", *Proceedings of the Conference on Computer Security, Dependability, and Assurance: From Needs to Solutions*, York - UK et Washington - DC - USA, 9-11 juillet 1998, pp. 171-181.

[Diguet *et al.* 2006] J.-P. Diguet, G. Gogniat, J.-L. Philippe, Y. Le Moullec, S. Bilavarn, C. Gamrat, K. Ben Chehida, M. Auguin, X. Fornari, P. Kajfasz, "EPICURE: A Partitioning and Co-Design Framework for Reconfigurable Computing", *Journal of Microprocessors and Microsystems*, vol. 30, n. 6, septembre 2006, pp. 367-387.

[DO178-B 1992] *Software Considerations in Airborne Systems and Equipment Certification*, Radio Technical Commission for Aeronautics (RTCA) et European Organization for Civil Aviation Electronics (EUROCAE), DO-178-B/ED-12-B, doi :10.1.1.1.6299, 1992.

[Feiler & Rugina 2007] P. Feiler et A. Rugina, "Dependability Modeling with the Architecture Analysis & Design Language (AADL)", *Technical Note CMU/SEI-2007-TN-043*, Carnegie Mellon University, juillet 2007, 256 p.

[Fields & McKendree 2005] L. Fields et T. McKendree, "Airplane Digital Distributed Fly-By-Wire Flight Control Systems Architectures: Reasons for Migratory Patterns of Computational Functionality", novembre 2005, 20 p. http://www.spirit-wolf.org/education/Fields_Lanny_Research_Paper.pdf

[Godo 2002] E.-L. Godo, "Flight Control System with Remote Electronics", *Proceedings of the 21st Digital Avionics Systems Conference (DASC 2002)*, Irvine, CA, USA, 27-31 octobre 2002, vol. 2, pp. 13B1-1 - 13B1-7.

[Hardekopf *et al.* 2001a] B. Hardekopf, K. Kwiat et S. Upadhyaya, "Secure and Fault-Tolerant Voting in Distributed Systems", *Proceedings of IEEE Aerospace Conference*, vol.3, Big Sky, MT, USA, 10-17 Mars 2001, pp. 3/1117-3/1126.

[Hardekopf *et al.* 2001b] B. Hardekopf, K. Kwiat et S. Upadhyaya, "A Decentralized Voting Algorithm for Increasing Dependability in Distributed Systems", *Joint Meeting of the 5th World Multiconference on Systemics, Cybernetics and Informatics (SCI 2001) and the 7th International Conference on Information Systems Analysis and Synthesis (ISAS 2001)*, Orlando, FL, USA, 22-25 juillet 2001, pp. 1-6.

[Hemani *et al.* 1999] A. Hemani, T. Meincke, S. Kumar, A. Postula, T. Olsson, P. Nilsson, J. Oberg, P. Ellervee et D. Lundqvist, "Lowering Power Consumption in Clock by using Globally Asynchronous Locally Synchronous Design Style", *Proceedings of the 36th annual ACM/IEEE Design Automation Conference*, New Orleans, Louisiana, USA, 21-25 juin 1999, pp. 873-878.

[Hesselink 2005] W.H. Hesselink, "The Boyer-Moore Majority Vote Algorithm with a Majority of Voting Rabbits", 7 novembre 2005, 2 p. disponible via : http://www.cs.rug.nl/~wim/pub/whh348.pdf

[Iyer *et al.* 1999] R.K. Iyer, A. Avizienis, D. Barron, D. Powell, H. Levendel et J. Samson, "COTS Hardware and Software in High-Availability Systems", *Proceedings of the 29th Fault-Tolerant Computing, International Symposium*, Madison, WI, USA, 15-18 juin 1999, pp. 120.

[Jordan & Marshall 1972] W.E. Jordan et G.C. Marshall, "Failure Modes, Effects and Criticality Analysis", *Proceedings of Annual Reliability and Maintenability Symposium*, San Francisco, CA, USA, 25-27 janvier 1972.

[Kalavade & Lee 1994] A. Kalavade et E.-A. Lee, "A Global Criticality/Local Phase Driven Algorithm for the Constrained Hardware/Software Partitioning Problem", *Proceedings of the Third International Workshop on Hardware/Software Codesign*, Grenoble, 22-24 septembre 1994, pp. 42-48.

[Kalavade & Subrahmanyam 1997] A. Kalavade et P.-A. Subrahmanyam, "Hardware/Software Partitioning for Multi-Function Systems", *Digest of Technical Papers of 1997 IEEE/ACM International Conference on Computer-Aided Design*, San Jose, CA, USA, 9-13 novembre 1997, pp. 516-521.

[Kehren 2005] C. Kehren, "Motifs formels d'architectures de systèmes pour la sûreté de fonctionnement", doctorat de l'Ecole Nationale Supérieure de l'Aéronautique et de l'Espace, 20 décembre 2005, 160 p.
disponible via : http://tel.archives-ouvertes.fr/tel-00011496/fr/

[Kehren *et al.* 2004] C. Kehren, C. Seguin, P. Bieber, C. Castel, C. Bougnol, J.-P. Heckmann et S. Metge, "Advanced Multi-System Simulation Capabilities with AltaRica," *Proceedings of International System Safety Conference (ISSC'04)*, Rhodes Island, USA, août 2004, pp. 489-498.

[Kossentini & Caspi 2004] C. Kossentini et P. Caspi, "Mixed Delay and Threshold Voters in Critical Real-Time Systems", *Proceedings of the Joint International Conferences on Formal Modelling and Analysis of Timed Systems and Formal Techniques in Real-Time and Fault Tolerant System (FORMATS-FTRTFT 2004)*, Grenoble, 22-24 septembre 2004, pp 21-35.

[Lamport *et al.* 1982] L. Lamport, R. Shostak et M. Pease, "The Byzantine Generals Problem", *ACM Transactions on Programming Languages and Systems*, vol. 4, issue 3, July 1982, pp. 382-401.

[Langer *et al.* 1992] D. Langer, J. Rauch et M. Rößler, "Fly-By-Wire Systems for Military High Performance Aircraft", *Real-Time Systems Engineering and Applications*, Kluwer Academic Publishers, 1992, pp. 369-395.

[Laprie & Randell 2001] J. Laprie et B. Randell, "Fundamental Concepts of Computer Systems Dependability", *First IARP/IEEE-RAS Joint Workshop on Technical Challenge for Dependable Robots in Human Environments*, Seoul, Korea, 21-22 mai 2001.

[Laprie *et al.* 1996] J.C. Laprie, J. Arlat, J.P. Blanquart, A. Costes, Y. Crouzet, Y. Deswarte, J.C. Fabre, H. Guillermain, M. Kaaniche, K. Kanoun, C. Mazet, D. Powell, C. Rabejac, P. Thevenod-Fosse, "Guide de la sûreté de fonctionnement", *Cépadues Éditions*, N°ISBN 2-85428-382-1, 1996, 324 p.

[Laprie *et al.* 1990] J.-C. Laprie, J. Arlat, C. Béounes, K. Kanoun, "Definition and Analysis of Hardware- and Software-Fault-Tolerant Architectures", *Computer*, vol. 23, no. 7, pp. 39-51, juillet 1990.

[Lin 1993] C.-F. Lin, *Advanced Flight Control System Design*, Prentice Hall Professional Technical Reference, septembre 1993, 688 p.

[Lower *et al.* 2005] M. Lower, B. Szlachetko et D. Krol, "Fuzzy Flight Control System for Helicopter Intelligence in Hover,", *5th International Conference on Intelligent Systems Design and Applications (ISDA'05)*, Wroclaw, Pologne, 8-10 septembre 2005, pp. 370-374.

[Manikas & Cain 1996] T.-W. Manikas et J-.T. Cain, "Genetic Algorithms vs. Simulated Annealing: A Comparison of Approaches for Solving the Circuit Partitioning Problem", *Technical Report EE-TR-96-101*, Department of Electrical Engineering, University of Pittsburgh, May 1996, 15 p.
disponible via : http://lyle.smu.edu/~manikas/Pubs/gasa-TR-96-101.pdf

[Mathworks] http://www.mathworks.fr/

[MIL-STD-1553B 1978] Standard du Département de Défense Américain, Aircraft Internal Time Division Command/Response Multiplex Data Bus, septembre 1978, 66 p. disponible via : http://ams.aeroflex.com/ProductFiles/AppNotes/milstd1553B.pdf

[Mitra & McCluskey 2001] S. Mitra et E.-J. McCluskey, "Diversity Techniques for Concurrent Error Detection", *Proceedings of 2001 International Symposium on Quality Electronic Design*, San Jose, CA, USA, 26-28 mars 2001, pp 249-250.

[Mkhida 2008] A. Mkhida, "Contribution à l'évaluation de la sûreté de fonctionnement des systèmes instrumentés de sécurité intégrant de l'intelligence", *Doctorat de l'Institut National Polytechnique de Lorraine*, 14 novembre 2008, 198 p.

[Morgan 1991] M. Morgan, "Integrated Modular Avionics for Next Generation Commercial Airplanes", *IEEE Aerospace and Electronic Systems Magazine*, vol. 6, issue 8, août 1991, pp. 9-12.

[Murdock & Koenig 2001] J.-R. Murdock et J.-R. Koenig, "Open Systems Avionics Network to replace MIL-STD-1553", *IEEE Aerospace and Electronic Systems Magazine*, vol. 16, issue 8, août 2001, pp. 15-19.

[Namazi & Nourami 2007] A. Namazi et M. Nourami, "Distributed Voting for Fault-Tolerant Nanoscale Systems", *Proceedings of the 25th International Conference on Computer Design (ICCD 2007)*, Lake Tahoe, CA, USA, 7-10 octobre 2007, pp. 568-573.

[Natale & Stankovic 2000] M. Di Natale et J.-A. Stankovic, "Scheduling Distributed Real-Time Tasks with Minimum Jitter", *IEEE Transactions on Computers*, vol. 49, issue 4, avril 2000, pp. 303-316.

[Obermaisser 2008] R. Obermaisser, "Temporal Partitioning of Communication Resources in an Integrated Architecture", *IEEE Transactions on Dependable and Secure Computing*, vol. 5, issue 2, avril-juin 2008, pp. 99-114.

[Pagetti 2004] C. Pagetti, "Extension temps réel d'AltaRica », doctorat de l'Ecole Centrale de Nantes, Université de Nantes, 20 avril 2004, 167 p.
disponible via : http://tel.archives-ouvertes.fr/docs/00/04/68/96/PDF/tel-00006316.pdf

[Parhami 1994] B. Parhami, "Voting Algorithms", *IEEE Transactions on Reliability*, vol. 43, issue 4, décembre 1994, pp. 617-629.

[Raksch *et al.* 2007] C. Raksch, R. Van Maanen, D. Rehage, F. Thielecke et U. B. Carl, "Performance Degradation Analysis of Fault-Tolerant Aircraft Systems", *Proceedings of the First CEAS European Air and Space Conference Century Perspectives*, Berlin, Allemagne, 10-13 septembre 2007, 9 p, disponible via :
http://www.eurtd.com/moet/PDF/DGLR-CEAS%20congress%20-%20TUHH%20Abstract.pdf

[Raulet 2006] M. Raulet, "Optimisations Mémoire dans la Méthodologie AAA pour Code Embarqué sur Architectures Parallèles", Doctorat de l'Institut National des Sciences Appliquées de Rennes, 18 mai 2006.
disponible via : http://tel.archives-ouvertes.fr/docs/00/12/42/83/PDF/these_MRL.pdf

[Rugina 2007] A.-E. Rugina, "Modélisation et évaluation de la sûreté de fonctionnement - De AADL vers les réseaux de Pétri stochastiques", doctorat de l'Institut National Polytechnique de Toulouse, 19 novembre 2007, 197 p.
disponible via : http://tel.archives-ouvertes.fr/tel-00207502/fr/

[Schneider & Lamport 1985] F.-B. Schneider et L. Lamport, **"**Paradigms for Distributed Programs", chapitre 8 dans *Distributed Systems-Methods and Tools for Specification an Advanced Course* (Eds M. Paul et H.J. Siegert), Lecture Notes in Computer Science, Springer, vol. 190, 1985, pp. 431-480.

[Schor *et al.* 1989] A. Schor, F. Leong et P. Babcock, "Impact of Fault-Tolerant Avionics on Life-Cycle Costs", *Proceedings of the IEEE National Aerospace and Electronics Conference*, Dayton, OH, USA, 22-26 mai 1989, vol. 4, pp. 1893-1899.

[Sghairi 2008] M. Sghairi, "Nouvelles architectures pour le système de commandes de vol électriques pour avion civil", EDSYS 2008, 9ème Congrès de Doctorants, Toulouse, mai 2008, 6p. (Rapport LAAS N° 08259).

[Sghairi *et al.* 2008a] M. Sghairi, A. de Bonneval, Y. Crouzet, P. Brot., J.-J. Aubert, "Architecture Optimization based on Incremental Approach for Airplane Digital Distributed Flight Control System", *IAENG Transactions on Electrical and Electronics Engineering Volume I - Special Edition of the World Congress on Engineering and Computer Science 2008*, Publisher: IEEE Computer Society, pp. 13-20 (Rapport LAAS N° 09177).

[Sghairi *et al.* 2008b] M. Sghairi, A. de Bonneval, Y. Crouzet, P. Brot., J.-J. Aubert, "Challenges in Building Fault-Tolerant Flight Control System for a Civil Aircraft", *World Congress on Engineering and Computer Sciences*, San Francisco (USA), 22-24 novembre 2008, 5p.

[Sghairi *et al.* 2008c] M. Sghairi, A. de Bonneval, Y. Crouzet, P. Brot., J.-J. Aubert, "Tolérance aux fautes dans les systèmes de commandes de vol pour avion civil", *6ème MAnifestation des Jeunes Chercheurs en Sciences et Technologies de l'Information et de la Communication (MAJESTIC 2008)*, Marseille, 29-31 octobre 2008, 8p, (Rapport LAAS N° 08273).

[Sghairi *et al.* 2008d] M. Sghairi, A. de Bonneval, Y. Crouzet, P. Brot., J.-J. Aubert, "New Architecture for Flight Control System (FCS) for Civil Aircraft", *Airbus PhD Days 2008*, Toulouse, 10 octobre 2008, 1p. (Rapport LAAS N° 08577).

[Sghairi *et al.* 2009a] M. Sghairi, A. de Bonneval, Y. Crouzet, P. Brot., J.-J. Aubert, "Système de commande de vol et aéronef le comportant", dépôt de Brevet conjoint AIRBUS France et LAAS-CNRS, mars 2009, 13p, n° dépôt national INPI : FR20090050830, dépôt international en cours (Rapport LAAS N° 09069.

[Sghairi *et al.* 2009b] M. Sghairi, A. de Bonneval, Y. Crouzet, P. Brot., J.-J. Aubert, "Système de commande de vol et aéronef le comportant", dépôt de Brevet conjoint AIRBUS France et LAAS-CNRS, mars 2009, 13p, n° dépôt national INPI : FR20090050831, dépôt international en cours (Rapport LAAS N°09070).

[Sghairi *et al.* 2009c] M. Sghairi, A. de Bonneval, Y. Crouzet, P. Brot., J.-J. Aubert, "Challenges in Building Fault-Tolerant Flight Control System for a Civil Aircraft », *IAENG International Journal of Computer Science*, vol. 35, n. 4, pp. 495-499, janvier 2009, ISSN: 1819-9224 (online version) and 1819-656X (print version).

[Sghairi *et al.* 2009d] M. Sghairi, A. de Bonneval, Y. Crouzet, P. Brot., J.-J. Aubert, "Distributed and reconfigurable architecture for flight control system", *28th Digital Avionics Systems Conference (DASC'09)*, Orlando, USA, 25-29 octobre 2009, pp. 6.B.2-01-6.B.2-10 [Distinctions : Best student paper et Best Track paper (dans les 6 meilleurs papiers / 150 de la conférence].

[Syndex] http://www.syndex.org

[Thomas & Ormsby 1994] M. Thomas et B. Ormsby, “On the Design of Side-Stick Controllers in Fly-By-Wire Aircraft”, *ACM SIGAPP Applied Computing Review, Special Issue on Safety-Critical Software,* vol. 2, 1994, pp. 15-20.

[Tran 2008] X.T. Tran “Méthode de test et conception en vue du test pour les réseaux sur puce asynchrones : Application au réseau ANOC”, doctorat de l’Institut National Polytechnique de Grenoble, 12 février 2008, 176 p.
disponible via : http://tel.archives-ouvertes.fr/tel-00407016/

[Traverse *et al.* 2004] P. Traverse, I. Lacaze et J. Souyris, “Airbus Fly-By-Wire: A Total Approach to Dependability”, *Proceedings 18th IFIP World Computer Congress, Building the Information Society*, 22-27 août 2004, pp. 191-212.

[Tripakis *et al.* 2008] S. Tripakis, C. Pinello, A. Benveniste, A. Sangiovanni-Vincent, P. Caspi et M. Di Natale, “Implementing Synchronous Models on Loosely Time Triggered Architectures”, *IEEE Transactions on Computers*, vol. 57, issue 10, octobre 2008, pp. 1300-1314.

[Wu 2005] C. Wu, “Inherent Delays and Operational Reliability of Airline Schedules”, *Journal of Air Transport Management*, vol. 11, issue 4, juillet 2005, pp. 273-282.

[Yeh 1998] Y.-C. Yeh, “Design Considerations in Boeing 777 Fly-By-Wire Computers”, *Proceedings of the 3rd IEEE International Symposium on High-Assurance Systems Engineering*, Washington, DC, USA, 13-14 novembre 1998, pp. 64-72.

[Youssef 2005] A. Youssef, “Réseau de communication à haut niveau d’intégrité pour des systèmes de commande-contrôle critiques intégrant des nappes de microsystèmes”, *Doctorat de l’Institut National Polytechnique de Toulouse*, 22 novembre 2005, 164 p. disponible via : http://tel.archives-ouvertes.fr/tel-00111808/fr/

ANNEXE 1 - ARCHITECTURES RESEAUX DE COMMUNICATION ET TOPOLOGIES POSSIBLES

Concernant l'architecture réseau, ce mémoire se limite à l'utilisation de moyens de communication avionables qui seront utilisés à court terme ou à long terme par Airbus (AFDX, Micro-AFDX, 1553). De ce fait, plusieurs topologies sont possibles ; nous en présentons deux dans cette annexe, en précisant toutefois que le nombre réel d'actionneurs par surface (spoilers, ailerons, rudder, THS et Elevator) dépend de la configuration de l'avion.

Topologie 1

Cette topologie comprend trois réseaux de communication (rose, bleu et rouge) redondants auxquels sont reliés tous les calculateurs primaires et secondaires. Notons que, dans cette topologie, le type de composants utilisé est homogène sur un même réseau (micro-AFDX pour les réseaux rose et bleu, AFDX pour le réseau rouge.

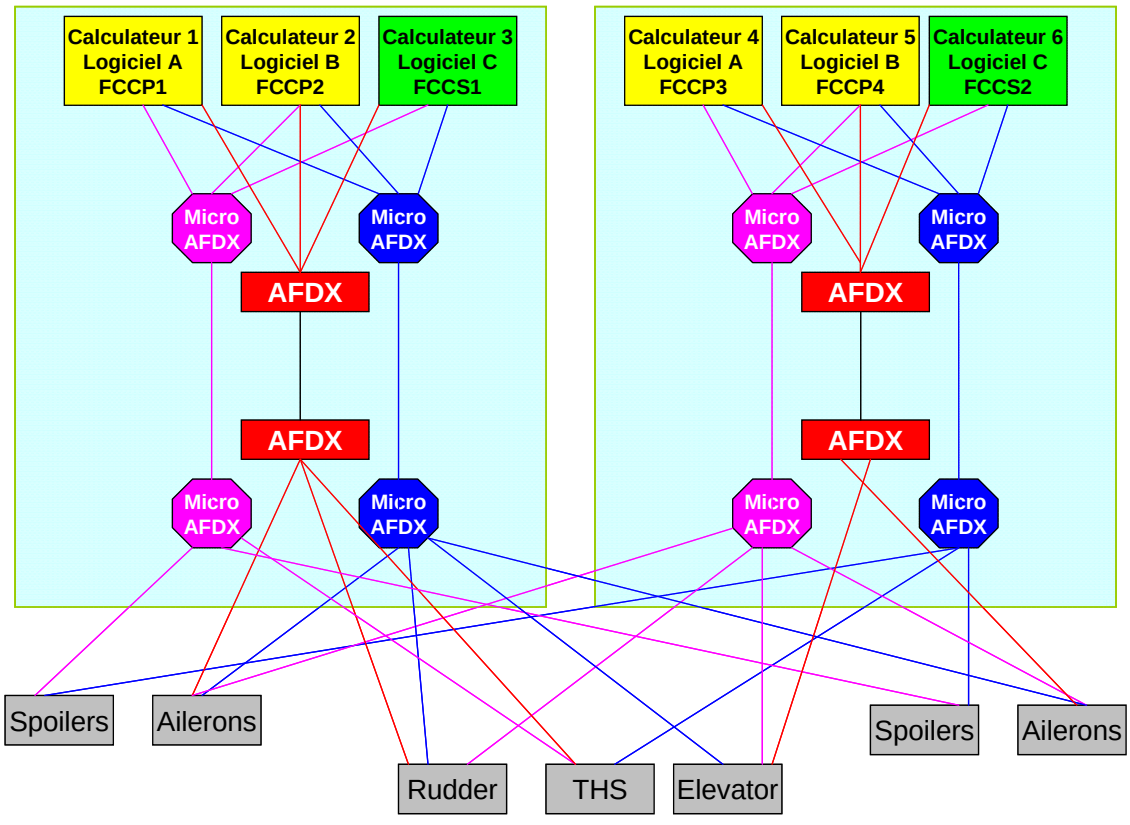


Figure A2.1 - Topologie 1

Les actionneurs, en fonction de leur criticité, peuvent être reliés à un ou plusieurs réseaux de communication. Donc, suite à la perte de deux réseaux, le contrôle de l'avion est toujours assuré par le troisième réseau.

En fonctionnement normal, les actionneurs communiquent à travers les réseaux rose et bleu à base de micro-switch AFDX. En cas de perte totale des micro-switch AFDX, la

communication entre calculateurs et actionneurs est assuré par le troisième réseau redondant (rouge) à base de switchs AFDX.

Les switchs AFDX assurent un débit de 100 Mbit/s, mais vu leurs tailles ils sont situés en général en baie avionique, ou ailleurs dans le fuselage de l'avion, alors que les micro-switchs (d'un débit de 10 Mbit/s) peuvent être situés plus proche des actionneurs.

Topologie 2

Dans cette topologie, les réseaux de communication entre les calculateurs et les actionneurs utilisent des composants hétérogènes (micro-AFDX et AFDX).

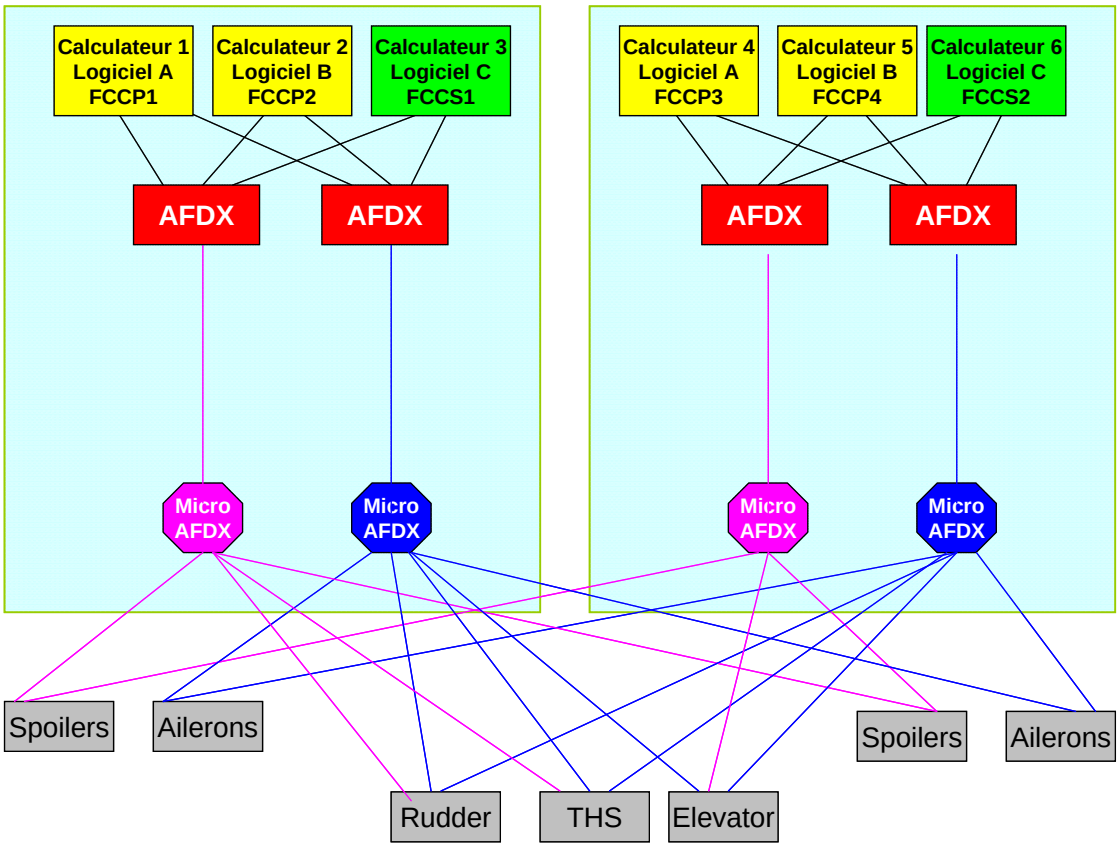


Figure A2.2 - Topologie 2

Remarque

Pour minimiser le nombre de bus de liaison actionneurs-réseaux, on propose deux topologies de communication :

- Communication directe :

  Chaque actionneur (voie COM et voie MON) est connecté aux deux côtés électriques pour pouvoir échanger des informations avec tous les calculateurs.

- Communication indirecte :

  - Les deux voies COM et MON de chaque actionneur communiquent avec l'extérieur en partageant le même média physique avec des signatures numériques.

- Chaque voie (COM ou MON) de chaque actionneur est connectée à un seul côté électrique (droit ou gauche) au lieu d'être connecté aux deux cotés électriques.

La voie COM (respectivement la voie MON) communique avec les calculateurs de site gauche (respectivement de site droit) via des liaisons directes. Elle peut être connectée à un ou plusieurs micro-switchs en fonction de la criticité de l'actionneur. Par contre, elle communique avec l'autre côté via son unité MON (respectivement COM) en utilisant des liaisons inter-unité entre les deux voies.

Exemple :

Pour les consignes FCC : l'unité COM de l'actionneur x possède les ordres des FCC numéro 1, 2 et 3, qui les échange avec MON contre les ordres des calculateurs numéro 4, 5 et 6. Au final, chaque voie peut effectuer ses traitements sur la totalité des ordres envoyés par tous les calculateurs du système.

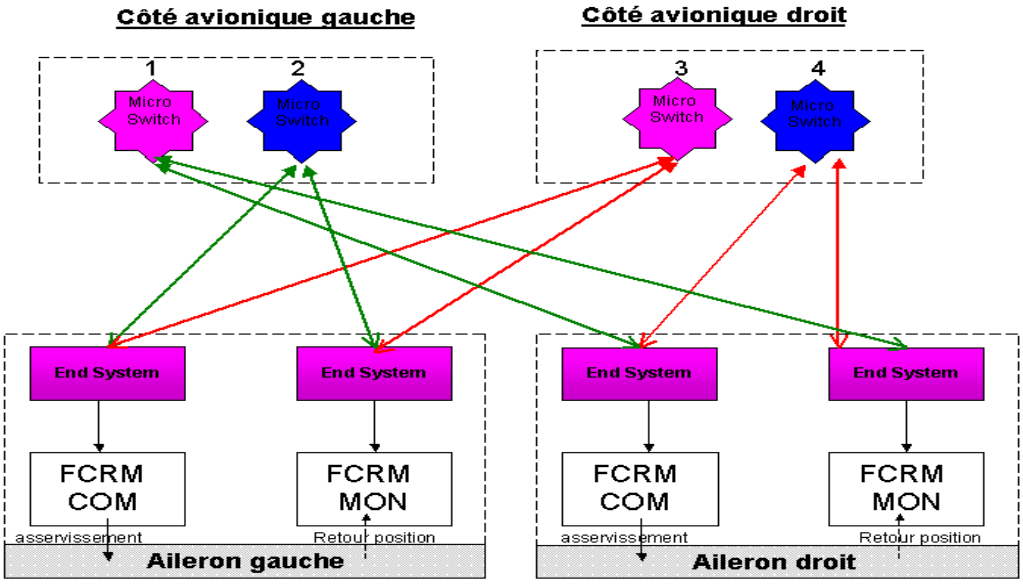


Figure A2.3 - Communication directe

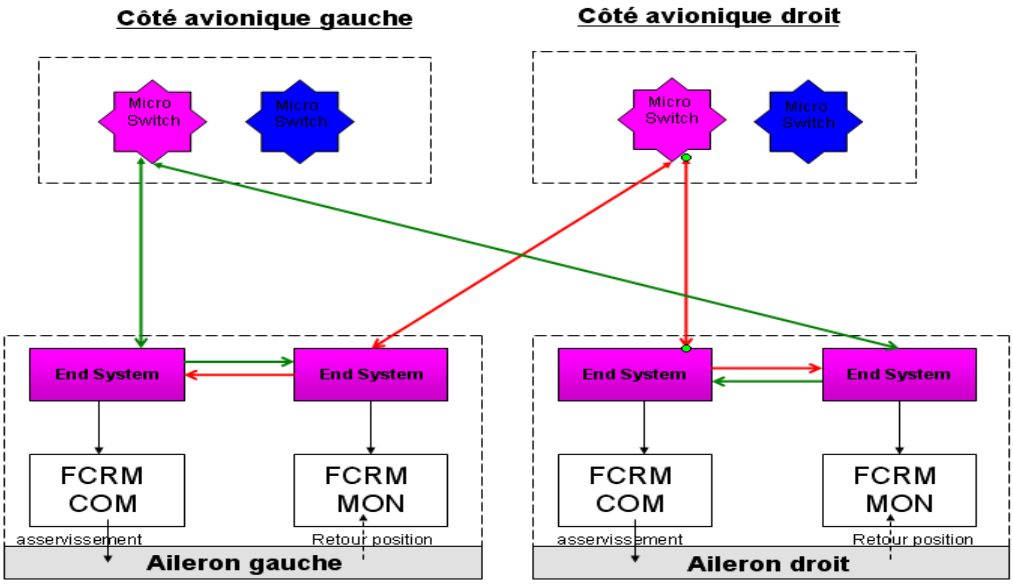


Figure A2.4 - Communication indirecte

ANNEXE 2 - PROCESSUS DE SECURITE POUR LES SYSTEMES AERONAUTIQUES

Avertissement : cette annexe correspond à la première partie du chapitre 1 de la thèse de Romain Bernard [Bernard 2009]. Elle est reproduite ici pour en faciliter l'accès à cette synthèse..

L'avion est considéré comme l'un des moyens de transport les plus sûrs. Cette réputation perdure car la sécurité est une préoccupation majeure du monde aérien :

- tout avion qui décolle d'un aéroport répond à des exigences nationales, voire internationales (suivant son périmètre d'exploitation), garantissant l'intégrité des personnes, tant à bord qu'autour de l'appareil ;
- le contrôle aérien applique des procédures précises pour guider chaque avion afin qu'il effectue un vol en évitant toute collision.

Tout règlement impose des exigences. La démonstration de la tenue de ces exigences permet en particulier d'obtenir l'autorisation de faire décoller et voler un aéronef de forte capacité à but commercial : on parle communément de *certification*.

La société Airbus, en tant qu'avionneur européen, se doit de certifier, auprès des autorités européennes, chaque avion pour que la compagnie aérienne cliente puisse en prendre possession et l'exploite sur ses lignes. La société Airbus est concernée par toutes les réglementations liées aux aéronefs, durant chaque phase du cycle de vie d'un avion :

- la conception : avant d'être produit et assemblé à grande échelle, chaque nouvel appareil doit obtenir une *certification de type* ;
- la production :
  - chaque avion produit est réalisé spécialement pour une compagnie cliente ; des écarts par rapport aux spécifications certifiées durant la conception sont possibles, mais ils doivent également répondre aux exigences des autorités européennes,
  - si la compagnie cliente réside en dehors d'Europe, il est également nécessaire de fournir des garanties aux autorités locales,
  - chaque avion obtient donc une *certification individuelle*,
- l'exploitation : chaque avion est suivi durant toute sa "vie" afin qu'il reste conforme à son état de certification :
  - chaque incident est signalé puis analysé pour démontrer la préservation de la conformité,
  - toute intervention (maintenance, réparation, modification) est enregistrée et contrôlée.

Durant la conception, en vue d'obtenir la certification de type, des analyses sont menées afin de chercher tout ce qui pourrait conduire l'avion à être dans une situation dégradée pouvant éventuellement porter atteinte à l'équipage, aux passagers ou à toute personne dans le périmètre de l'appareil : on parle d'*analyses de sûreté de fonctionnement des systèmes*.

Ce chapitre a pour but d'introduire les analyses de sûreté de fonctionnement des systèmes en présentant tout d'abord les différentes instances (autorités, groupes de travail, constructeurs) liées à la sécurité aérienne, ainsi que les différents textes dont elles ont la charge. La seconde partie de ce chapitre se focalise sur le processus actuel d'analyse de sûreté de fonctionnement des systèmes chez Airbus.

A3.1 La sûreté de fonctionnement des systèmes dans le monde aérien

La sûreté de fonctionnement des systèmes constitue une partie de la sécurité d'un avion : seuls les systèmes et les fonctions qu'ils remplissent sont analysés ; la structure de l'avion et l'intégrité des données informatiques à bord sont traitées séparément. Les acteurs de la sûreté de fonctionnement des systèmes aériens peuvent être classés en trois catégories :

- les autorités de certification qui définissent des règles et s'assurent de leur application ;
- les avionneurs, ainsi que les équipementiers et les motoristes, qui doivent appliquer ces règles et démontrer qu'elles sont correctement appliquées ;
- des groupes de travail, constitués à la fois de représentants des avionneurs et des autorités, qui s'entendent sur des pratiques permettant de satisfaire les règles et de répondre aux exigences.

Chaque catégorie d'acteurs élabore ses propres documents décrivant les pratiques permettant de satisfaire les règles.

Avant de présenter le processus actuel d'analyse de sûreté de fonctionnement des systèmes, cherchons à comprendre son élaboration en nous intéressant à chaque catégorie d'acteur et aux documents majeurs.

A3.1.1 Autorités

La sécurité aérienne est assurée par une répartition précise des rôles en matière de réglementation : des règles sont définies au niveau mondial, puis contrôlées par des autorités géographiquement liées aux avionneurs en vue d'attribuer les autorisations de vol.

L'Organisation de l'Aviation Civile Internationale (OACI), agence spécialisée des Nations Unies, établit les règles de l'air, pour l'immatriculation des aéronefs et la sécurité, ainsi que les droits et devoirs des pays signataires en matière de droit aérien relatif au transport international. L'OACI fut instaurée lors de la Convention relative à l'Aviation Internationale Civile, connue sous le nom de *Convention de Chicago*. Initialement signée le 7 décembre 1944 par 52 pays, elle compte à ce jour 190 nations signataires. Chaque pays signataire participe à l'élaboration des annexes, contenant les normes et recommandations pour le transport aérien international, et s'engage à les considérer dans ses lois nationales.

De nombreuses organisations de certification et de réglementation aéronautique civile existent dans le monde, néanmoins les deux principales agences sont :

- l'Agence Européenne de la Sécurité Aérienne (AESA) ou *European Aviation Safety Agency (EASA)* en anglais, créée en 2003 et succédant aux *Joint Aviation Authorities (JAA)*, association d'organisations européennes de réglementation aéronautique ;
- la *Federal Aviation Agency (FAA)* aux États Unis, créée en 1958.

L'EASA et la FAA définissent des exigences concernant, entre autres domaines : la conception des avions, leur exploitation commerciale ou les licences de pilotage. Chaque agence rédige ses propres documents mais les exigences étant proches, il est possible d'établir des correspondances entre documents européens et américains. Les principales exigences applicables à la sûreté de fonctionnement des systèmes sont définies dans le paragraphe numéroté 1309 à la fois dans le document baptisé *CS-25 (Certification Specification)* de

l'EASA pour l'Europe, et dans le document baptisé *FAR-25 (Federal Aviation Regulation)* de la FAA pour les États-Unis. Les paragraphes décrivant des exigences sont généralement complétés d'annexes décrivant les moyens acceptés pour répondre à ces exigences, en anglais *Acceptable Means of Compliance (AMC).*

A3.1.2 Groupe de travail

Afin d'améliorer constamment le niveau de sécurité des passagers, les principaux acteurs du monde aéronautique (avionneurs, motoristes, équipementiers) collaborent au sein de deux principaux groupes de travail :

- le groupe WG-63 *(Complex Aircraft Systems)* de la *European Organisation of Civil Aviation Equipment (EUROCAE)*, historiquement présidé par un représentant d'Airbus ;
- le groupe S-18 *(Aircraft and System Development and Safety Assessment Committee)* de la *Society of Automotive Engineers (SAE)*, historiquement présidé par un représentant de Boeing.

Ces groupes élaborent des recommandations, applicables aux processus, méthodes et outils, pour répondre aux exigences des autorités précédemment citées. La présence de représentants des autorités garantit la bonne interprétation des réglementations.

Le groupe S-18 travaille principalement à l'élaboration de deux documents dits *Aerospace Recommended Practice (ARP)* :

- l'ARP 4754 *(Certification considerations for highly-integrated or complex aircraft systems)* fournit des recommandations pour démontrer que des systèmes complexes (c-à-d dont le niveau de sécurité ne peut pas être démontré par test ou dont la compréhension du fonctionnement nécessite le support d'outils) ou hautement intégrés (c-à-d qui réalise ou contribue à plusieurs fonctions) satisfont les exigences de navigabilité ;
- l'ARP 4761 *(Guidelines and methods for conducting the safety assessment process on civil airborne systems and equipment)* propose des méthodes possibles pour évaluer la sûreté de fonctionnement des systèmes d'un avion en vue de sa certification (les différentes étapes sont présentées dans la suite de ce chapitre).

Le groupe WG-63 élabore des documents équivalents baptisés respectivement ED-79 et ED-135. Des réunions communes entre les deux groupes permettent une évolution cohérente des documents et une meilleure réflexion.

A3.1.3 Avionneurs

Les avionneurs doivent se conformer aux exigences des autorités et, pour y répondre, disposent des recommandations SAE/EUROCAE afin d'établir leurs propres processus. Si quelques représentants participent aux groupes de travail, tous les ingénieurs contribuant aux analyses de sûreté de fonctionnement des systèmes doivent connaître les règles et méthodes à appliquer, ce qui nécessite l'élaboration de documents propres à chaque avionneur.

Concernant la sûreté de fonctionnement des systèmes et la fiabilité, la société Airbus élabore deux documents :

- l'*Airbus Directive (ABD)* 100 module 1.3, destiné aux équipementiers et fournisseurs, précisant les exigences applicables aux équipements réalisés, les informations à fournir et les études à conduire ;
- l'ABD 200 module 1.3, à usage interne, détaillant les différentes analyses et le processus à respecter. Afin d'anticiper des exigences toujours plus nombreuses de la part des autorités, les avionneurs imposent généralement des contraintes supplémentaires de façon à prendre des marges de sécurité.

A3.2 Processus actuel

Conformément aux ARP 4754 et 4761, les avionneurs réalisent différentes analyses pour démontrer la tenue des exigences imposées par les réglementations internationales en matière de navigabilité aérienne.

Le processus global se décompose selon les grandes étapes suivantes :

1) identifier les fonctions souhaitées ;
2) rechercher les conditions défaillantes liées à chaque fonction pour déterminer les possibles configurations de panne, en anglais *Failure Condition (FC)* ;
3) définir une criticité pour chaque FC en fonction de ses effets ;
4) déduire des exigences (principes de conception et probabilité d'occurrence maximum tolérée) pour prévenir chaque FC ou maintenir leurs conséquences à un niveau acceptable ;
5) démontrer que l'étude est complète et que les résultats sont en accord avec les réglementations en vigueur : la probabilité d'occurrence de chaque FC est inférieure ou égale à l'objectif déduit de la criticité.

Une fonction correspond à un service (abstrait) que doit rendre un objet physique ou un ensemble d'objets physiques (concrets). Les programmes aéronautiques considèrent la décomposition physique suivante :

– un *avion* est constitué d'une structure, appelée *fuselage*, de *moteurs* et de *systèmes* ;
– un système est composé d'*équipements*.

Plusieurs systèmes peuvent fonctionner conjointement afin de remplir une fonction, ces systèmes sont alors dits *interfacés* ou *en interface*. Par exemple, le système de génération électrique est interfacé à de nombreux systèmes tels que les commandes de vol du fait que les calculateurs de vol ne peuvent fonctionner sans énergie électrique.

Cette décomposition physique est transposée en décomposition fonctionnelle hiérarchique : les fonctions de niveau supérieur sont reliées à un ensemble de fonctions de niveaux inférieurs. La relation entre niveaux signifie que les fonctions de niveau inférieur contribuent à la réalisation de la fonction de niveau supérieur. Cette décomposition est utile pour guider les analyses de sûreté de fonctionnement des systèmes et allouer des exigences aux concepteurs des fonctions. Les trois principaux niveaux de décomposition utilisés dans les programmes aéronautiques sont :

– le niveau "avion", le plus haut dans la hiérarchie, qui regroupe les fonctionnalités principales de l'avion comme "commander l'avion", "naviguer" ;
– le niveau "système", qui regroupe des fonctions au sein de systèmes tels que le système des commandes de vol ;
– le niveau "équipement", dernier niveau dans la hiérarchie, qui regroupe des fonctions réalisées par un même équipement.

Les analyses à conduire concernent initialement l'avion dans son ensemble, puis les systèmes embarqués (considérés individuellement, liés fonctionnellement ou liés par leur placement à bord) et enfin les équipements qui composent un système.

En nous référant aux ARP et ABD, nous avons résumé ci-après le processus actuel (cf. Figure A3.1) que nous décomposons en trois catégories d'analyses :

– évaluation des risques fonctionnels, en anglais *Functional Hazard Assessment (FHA)* : étude des fonctions pour déterminer les défaillances fonctionnelles possibles et classifier les risques associés à des configurations de panne spécifiques ;

– évaluation de la sécurité des systèmes, en anglais *System Safety Assessment (SSA)* : évaluation de la conformité de l'architecture d'un système avec les exigences de sécurité déduites des FHA;
– analyses des causes communes, en anglais *Common Cause Analysis (CCA)* : études complémentaires des risques liés à des causes communes à plusieurs systèmes.

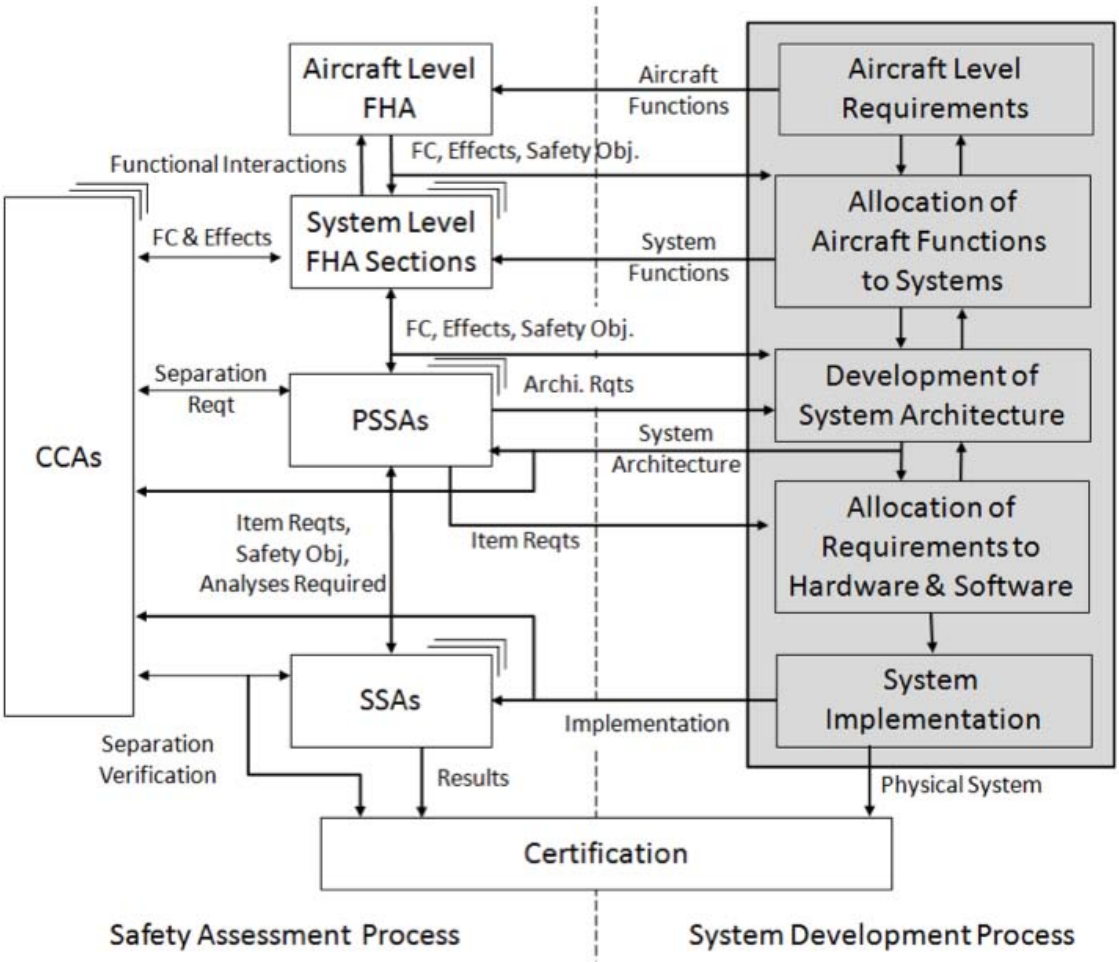


Figure A3.1 - Processus d'évaluation de la sécurité (extrait de l'ARP 4754)

Remarque : La Figure A3.1 représente un canevas de processus d'analyse de sûreté de fonctionnement des systèmes. Chaque industriel est libre de s'en inspirer ou non pour mettre en place son propre processus d'analyse de sûreté de fonctionnement des systèmes.

A3.2.1 - Functional Hazard Assessment (FHA)

Chaque nouvel avion est conçu pour une utilisation spécifique : transport de passagers, acheminement de marchandises, utilisation militaire. Chaque utilisation ajoute des fonctions spécifiques aux fonctions communes à tout avion (décoller, se mouvoir dans le ciel...). Or toute fonction peut être perdue ou remplie de façon incorrecte (partiellement, de façon intempestive, en retard...). L'examen systématique des fonctions permet de recenser toutes les Failure Conditions FC possibles. Pour chaque fonction et chaque défaillance fonctionnelle, une recherche systématique des scénarios opérationnels est réalisée. Ensuite, l'étude des répercussions de chaque scénario (sur l'avion, son équipage, ses occupants et les personnes évoluant autour de l'avion) permet d'établir, en se référant au Tableau A3.1, sa criticité. Les

scénarios sont ensuite regroupés par effet équivalent en FC. Cela permet d'associer à chaque FC :

- un objectif quantitatif : une probabilité par heure de vol à atteindre ;
- un niveau de confiance du développement, en anglais *Design Assurance Level (DAL)* : chaque catégorie de DAL impose des règles de conception tant sur les architectures physiques que sur les techniques d'implémentation logicielle.

La liste des fonctions, des FC et leur criticité constituent l'évaluation des risques fonctionnels (FHA).

Criticité	Probabilité (par heure de vol)	Niveau de confiance du développement	Effets
Mineur *(Minor, MIN)*	N/A	D	- légère réduction des marges de sécurité - légère augmentation de la charge de travail équipage - dérangements pour les occupants
Majeur *(Major, MAJ)*	10^{-5}	C	- réduction significative des marges de sécurité ou des fonctionnalités - augmentation significative de la charge de travail équipage ou conditions dégradant l'efficacité de l'équipage - gênes pour les occupants
Dangereux *(Hazardous, HAZ)*	10^{-7}	B	- réduction importante des marges de sécurité ou des fonctionnalités - charge de travail plus élevée ou détresse physique telle que l'équipage n'est pas en mesure de réaliser des tâches précisément ou complètement
Catastrophique *(Catastrophic, CAT)*	10^{-9}	A	toute condition de panne qui empêche une poursuite du vol et un atterrissage sûrs

Tableau A3.1 - Criticité des configurations de panne

Outre les exigences quantitatives, des exigences qualitatives sont associées à chaque FC en fonction de sa criticité. Ces exigences sont de la forme "la FC de criticité C ne peut se produire en moins de N défaillances". Le Tableau A3.2 présente la relation entre le nombre N de défaillances nécessaires et la criticité C.

Classification	Catastrophic	Hazardous	Major	Minor	No Safety Effect
Qualitative requirement	3	2	2	1	0

Tableau A3.2 - Criticité et exigences qualitatives associées

Durant la conception d'un avion, une première FHA de niveau avion, en anglais Aircraft level FHA(A/C FHA), considère l'avion dans sa globalité, puis différentes FHA sont réalisées pour traiter chaque système embarqué individuellement, en anglais System FHA.

A3.2.1.1 - Aircraft Level FHA

À partir de la liste des fonctions attendues pour un avion et des choix de conception initiaux (nombre de moteurs...), une évaluation qualitative de haut niveau recense les risques potentiels liés aux fonctions basiques de l'avion.

Une fonction est réalisée par un système ou par une combinaison de systèmes. La A/C FHA doit donc permettre d'identifier des FC impliquant plusieurs systèmes à la fois : la criticité déduite des effets des défaillances combinées peut être supérieure à celle qui serait définie pour chaque système étudié individuellement.

Les exigences de sécurité, qualitatives (règles de ségrégation, DAL...) et quantitatives (objectif de probabilité) déduites de la A/C FHA sont allouées aux systèmes contribuant aux différentes fonctions et doivent être prises en compte durant leur conception.

A3.2.1.2 - System FHA

La FHA système est également une analyse qualitative et itérative. Elle s'appuie sur les résultats de la A/C FHA et traite les fonctions du système concerné en étudiant les conséquences d'une défaillance ou d'une combinaison de défaillances tant sur le système que sur l'avion. Lorsque les fonctions avion ont été affectées à des systèmes par le processus de conception, chaque système fait l'objet d'une FHA système.

A3.2.2 - System Safety Assessment (SSA)

Les concepteurs de systèmes doivent élaborer une architecture pour réaliser les fonctions mentionnées précédemment. Compte tenu des FHA déjà réalisées, il est nécessaire d'évaluer chaque architecture pour s'assurer qu'elle répond aux exigences définies au préalable. Ces évaluations sont menées dans la continuité des FHA en suivant un processus itératif : chaque évaluation peut détecter une correction à effectuer qui nécessite une nouvelle évaluation.

Bien que l'évaluation d'une architecture s'intéresse principalement à un système, la forte interconnexion des systèmes dans un avion implique de considérer les systèmes en interface.

Dans un premier temps, des évaluations dites "préliminaires", en anglais *Preliminary System Safety Assessment (PSSA)*, sont réalisées. Une PSSA permet, tout d'abord, de compléter la liste des FC et donc éventuellement d'ajouter de nouvelles exigences de sécurité. D'autre part, une PSSA permet de déterminer une liste préliminaire des actions pilotes à accomplir lorsqu'une FC se produit et des opérations de maintenance à effectuer. La PSSA vérifie également que l'architecture du système est cohérente avec les modes de défaillance envisagés. Pour cela, une recherche des combinaisons minimales de défaillances, appelées *coupes minimales*, est effectuée. Par souci de concision et de lisibilité, les coupes minimales sont présentées sous la forme d'un arbre de défaillances ou d'un diagramme de dépendance par FC de criticité MAJ, HAZ ou CAT. La probabilité d'occurrence d'une FC est calculée à partir de ces coupes minimales, de la probabilité individuelle de chaque défaillance ainsi que de paramètres complémentaires (temps de vol moyen, intervalle de temps entre 2 opérations de maintenance...).

Enfin, cette évaluation permet d'allouer au niveau équipement les exigences de sécurité du niveau système. Ces exigences sont soumises aux fournisseurs d'équipements.

Après plusieurs PSSA successives, les résultats des études menées par les fournisseurs sont intégrés aux résultats des PSSA afin de vérifier que l'architecture sélectionnée est conforme aux exigences qualitatives et quantitatives définies dans les FHA et PSSA.

A3.2.3 - CCA - (PRA, ZSA, CMA)

Les exigences de sécurité peuvent imposer l'indépendance de plusieurs fonctions, systèmes ou équipements. Cette indépendance peut être une dissemblance fonctionnelle (technologies différentes pour deux composants ayant la même utilité, implémentation différentes pour deux logiciels...) ou physique (c-à-d. installation des composants redondants dans des zones différentes de l'avion). Le paragraphe 1309 du CS 25 insiste sur le besoin d'étudier les défaillances de cause commune (CCA).

Les CCA identifient les modes de défaillance individuels et les événements extérieurs potentiellement responsables de FC MAJ, HAZ ou CAT :

- dans le cas des FC CAT, de tels événements ne sont pas tolérés ;
- dans le cas des FC MAJ ou HAZ, la probabilité d'occurrence de tels événements doit respecter l'objectif de probabilité correspondant à la criticité de la FC.

Les analyses de causes communes sont constituées de trois types d'analyses : l'analyse des modes communs, en anglais *Common Mode Analysis (CMA)*, l'analyse des risques particuliers, en anglais *Particular Risk Analysis (PRA)* et l'analyse de la sécurité par zone, en anglais *Zonal Safety Analysis (ZSA).*

A3.2.3.1 - CMA

L'analyse des modes communs fournit la preuve que des défaillances supposées indépendantes le sont réellement. Cette analyse étudie les effets d'erreurs de conception, de fabrication ou de maintenance et de défaillances de composants communs. Des composants constitués des mêmes pièces ou disposant du même logiciel peuvent subir la même défaillance et donc impacter tout le système malgré leur indépendance physique.

Cette analyse peut être effectuée sur les arbres de défaillances ou les diagrammes de dépendance en vérifiant que des événements liés à une même porte logique "ET" sont réellement indépendants : deux défaillances reliées par la porte "ET" sous-entendent la présence d'un principe de redondance ; cependant, des erreurs d'implémentation, de fabrication ou de maintenance peuvent invalider ce principe.

A3.2.3.2 - ZSA

Une analyse de sécurité par zone étudie chaque zone physique de l'avion pour s'assurer que l'installation des équipements, les éventuelles interférences physiques avec les systèmes adjacents et les possibles erreurs de maintenance ne violent pas les exigences d'indépendance du système. Toute détection d'un risque potentiel pour la sécurité implique, sauf démonstration que ce risque reste acceptable, de concevoir une nouvelle architecture d'un système ou de revoir l'installation des équipements de ce système dans l'avion.

A3.2.3.3 - PRA

Une analyse des risques particuliers étudie qu'un événement extérieur au système concerné n'a pas d'influence sur les exigences d'indépendance. Cette analyse s'intéresse aux effets simultanés d'un risque ainsi qu'aux effets résultants. Il existe différents risques à prendre en compte parmi lesquels :

- le feu ;
- la glace, la grêle, la neige ;
- les fuites de liquides ;
- le péril aviaire ;
- la foudre ;
- l'éclatement pneu et moteur.

Les risques particuliers étudiés peuvent avoir un impact sur plusieurs zones alors que le périmètre d'étude d'une analyse de type ZSA est restreint à une zone.

Avertissement : pour compléter cette description du processus des analyses de sécurité, la Figure A3.2 décrit les relations entre les différents standards.

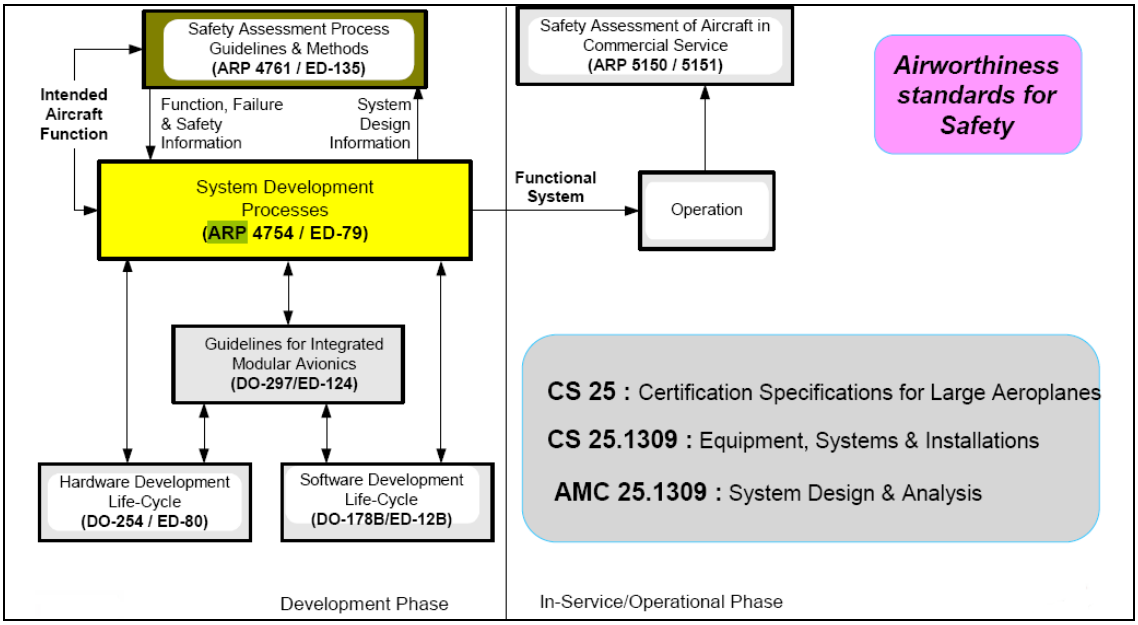


Figure A3.5 - Les relations entre les standards

(http://andrea.nfrance.com/~eq40782/files/drire/developpement-industriel/certification-together/CEAT_Evolution-standarts-aero_JC-Bonnet.pdf)

TITRE

Architectures innovantes de systèmes de commandes de vol

RESUME

L'aboutissement aux Commandes de Vol Électriques (CDVE) des avions civils actuels s'est fait par étapes, après une longue maturation des différentes technologies mises en place. La prochaine étape est l'utilisation de communications intégralement numériques et d'actionneurs intelligents.

Cette thèse propose de nouvelles architectures, en rupture avec l'état de l'art, avec de nouvelles répartitions des fonctions intelligentes entre l'avionique centrale (calculateurs de commandes de vols) et l'avionique déportée (électroniques locales des actionneurs) dont l'avantage est d'exiger moins de ressources par rapport aux architectures conventionnelles tout en satisfaisant les mêmes exigences de sécurité et de disponibilité ainsi que les exigences croissantes en fiabilité opérationnelle de la part des compagnies aériennes.

La sûreté de fonctionnement et la robustesse des nouvelles architectures proposées ont été validées respectivement sous OCAS/Altarica et Matlab/Simulink.

MOTS CLES

architectures distribuées, commandes de vol électriques, actionneurs intelligents, communications numériques, tolérance aux fautes, analyses de sécurité

TITLE

Innovative Architectures of Flight Control Systems

SUMMARY

The current civil aircraft's electrical flight control has been changed to take benefit of technical improvements. New technologies, when mature, can be incorporated in aircrafts. Evolutions are considered towards a digital communication and intelligent actuators.

This thesis is aiming at proposing alternative architectures with distribution of system functionality between flight control computers and actuators with less hardware and software resources. New architectures must meet the same safety and availability requirements with additional operational reliability (required by airlines).

Dependability and robustness of new architectures have been validated trough respectively OCAS / AltaRica and Matlab / Simulink.

KEYWORDS

distributed architectures, electrical flight control, intelligent actuators, digital communication, fault tolerance, safety analysis