Table des matiéres

Introduction générale

L'évolution croissante du marché aéronautique au monde, a poussé les entreprises du domaine à chercher d'autres alternatives pour améliorer les performances de leurs équipements et offrir les meilleurs services avec plus d'options surtout en ce qui concerne la sureté de fonctionnement de ces derniers.

Pour proposer des produits novateurs et à la pointe de la technologie, le groupe ZODIAC AEROSPACE se recentre sur son cœur de métier entraînant ainsi une redéfinition globale des relations avec les équipementiers qui se spécialisent à leur tour pour répondre efficacement à l'intégration nécessaire de l'électronique et de l'informatique dans les différents équipements aéronautiques et spatiaux.

A cela s'ajoute des enjeux de protection de l'environnement, du prix du pétrole et du renforcement de la sûreté des appareils. Et du fait des exigences de sécurité imposées par les autorités de certification, les opérations de validation constituent aujourd'hui une des phases les plus délicates du développement de logiciels embarqués sur avion. La sécurité des aéronefs est intimement liée à la qualité des logiciels embarqués. Plusieurs activités sont réalisées par Zodiac Aerospace.

Le Maroc est en cours d'une progression industrielle, le domaine aéronautique ne fait pas de l'exception, la preuve l'installation de ZAM depuis janvier 2009. C'est un secteur, qui grâce à son excellent niveau d'activité, crée une chaine diversifiée de métiers partant de la conception à la fabrication des aéronefs. ZAM pour sa part s'organise autour des activités principales du groupe ; à savoir équipement, sécurité et systèmes embarqués.

Dans cette perspective s'inscrit le projet SPDS (Secondary Power Distribution System) du département logiciel du pôle ingénierie. Ce système relie, via un bus CAN, la carte CPS à :

- 8 cartes SSPC (Solid State Power Controller)
- 2 DU (Display Unit)
- Un ordinateur de maintenance MTC (Maintenance Computer Tools)

La mission définie par le sujet consiste à développer un simulateur permettant de tester le bon fonctionnement du logiciel implanté dans la carte CPS. Le programme de test permet de générer certains cas particuliers auxquels le logiciel doit répondre de manière déterministe. La phase des tests informels permet de repérer les bugs et d'y remédier avant même de franchir la phase de test du cycle de vie du logiciel.

Ce mémoire, organisé en 5 chapitres principaux, regroupe l'ensemble des informations relatives au simulateur permettant de tester le bon fonctionnement d'un calculateur. Le premier de ces chapitres est consacré à la présentation de la société et à la présentation des différentes théories qui ont été utilisées lors de la réalisation du projet. Le second donne une présentation générale de la problématique, le troisième à effectuer une analyse du cahier de charge du projet et à développer les différents besoins en tâches bien précises. Le quatrième présente les différents outils logiciels utilisés, le cinquième à l'implémentation du simulateur et au développement de l'interface graphique.

Université Sidi Mohammed Ben Abdellah
Faculté Des Sciences et Techniques Fès
Département de Génie Electrique

Université Sidi Mohammed Ben Abdellah
Faculté Des Sciences et Techniques Fès
Département de Génie Electrique

Chapitre I

Organisme d'accueil
ZODIAC AEROSPACE MAROC
&
Développement Logiciel

Ce chapitre sera consacré à la présentation de Zodiac Aerospace Maroc de ces activités, ces clients et au développement logiciel au sein de cet organisme.

PRESENTATION DE ZODIAC AEROSPACE

Le groupe ZODIAC AEROSPACE est un équipementier aéronautique. La vocation de Zodiac pour l'aéronautique s'est faite jour en 1896 avec le développement des ballons dirigeables et des aéroplanes. Dans les années 1930, c'est l'invention du concept du bateau pneumatique qui permet à la société de prendre son essor et qui impose l'image de marque du Groupe après la seconde guerre mondiale.

Le Groupe est aujourd'hui recentré sur l'aéronautique, après la cession de ses activités Marine en septembre 2007.

Zodiac Aerospace poursuit son développement pour assurer une croissance constante de ses activités. Avec une rigueur industrielle et de gestion, les savoir-faire technologiques de Zodiac sont organisés en trois métiers principaux : équipements de sécurité, systèmes embarqués, et intérieurs de cabines, et se déclinent en six branches d'activité.

L'ensemble des activités de Zodiac regroupe près de 20 000 personnes dans le monde.

Organisation du groupe

En bonne logique avec son approche de développement sur des créneaux spécifiques et porteurs, les implantations industrielles du Groupe couvrent les zones géographiques où se concentrent les principaux constructeurs aéronautiques, notamment l'Amérique du Nord, l'Amérique du Sud et l'Europe.

Ainsi le Groupe peut offrir le meilleur service aux grands donneurs d'ordre tels que les avionneurs, les compagnies aériennes, les administrations, les armées, ...

Le Groupe compte 77 sites de production répartis dans le monde et une branche d'activité entièrement dédiée au service et au support clients avec 16 centres d'opération répartis entre l'Europe, les Etats-Unis et l'Asie.

Métier du groupe Zodiac Aerospace

Le groupe Zodiac Aerospace exerce son métier dans six branches d'activités qui sont :

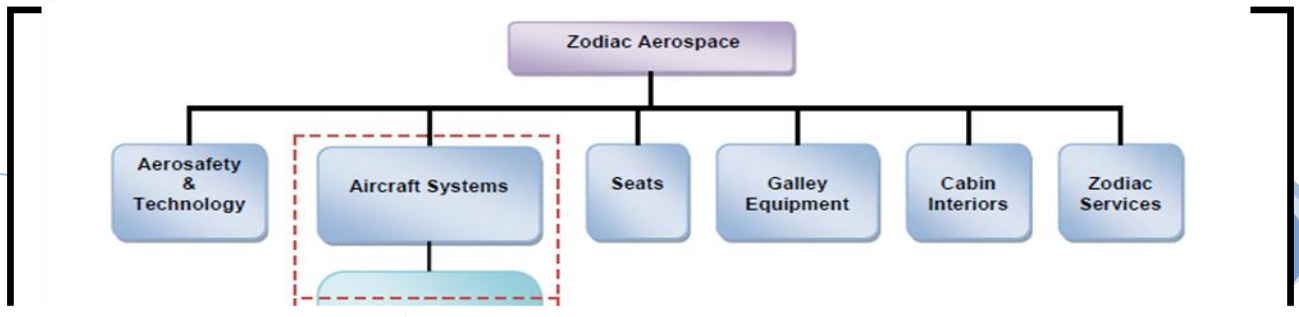

Figure 1 : Organisation en branches du groupe Zodiac Aerospace

✈ Aerosafety & Technology :

La branche Aerosafety & Technology conçoit et fournit des équipements, des produits, des systèmes complets dédiés au sauvetage et à la protection ainsi que des solutions technologiques pour des domaines d'applications très variés.

Sa mission a aussi pour vocation d'innover dans le domaine de la télémesure et de la télécommunication avec des produits et systèmes à forte valeur ajoutée pour les secteurs aéronautique et spatial tout comme pour certains marchés par exemple l'automobile avec ses solutions pour les airbags.

✈ Aircraft Systems :

La branche Aircraft Systems opère dans les parties citées ci-dessous:

→ Circulation carburant
→ Surveillance et gestion de systèmes
→ Oxygène et protection physiologique
→ Gestion de la puissance électrique
→ Commandes et signalisation dans le cockpit
→ Eclairage et actionneurs
→ Hydraulique et régulation
→ Actionneurs, capteurs et moteurs

Les sociétés de la Branche fournissent des équipements fiables et performants aux constructeurs d'avions et de moteurs et sont reconnues au plan mondial comme les spécialistes des équipements et des systèmes de haute technologie au service de fonctions essentielles tant en vol qu'au sol.

La branche développe des compétences de pointe dans les domaines suivants :

- ***Intertechnique :*** circulation carburant (circulation, gestion du carburant), surveillance et gestion de systèmes ; calculateurs embarqués ; surveillance et mesure de la température et de l'humidité.
- ***Intertechnique & Avox Systems*** : systèmes oxygène
- ***ECE*** : gestion de la puissance électrique ; commandes et signalisation dans le cockpit ; éclairage et actionneurs.
- ***IN-LHC & IN-FLEX*** : hydraulique et régulation.
- ***PRECILEC*** : actionneurs, capteurs & moteurs.

✈ Seats :

Cette branche est spécialisée dans la fabrication et la commercialisation de sièges pour les avions civils et les hélicoptères.

Cette branche permet à Zodiac Aerospace d'occuper la première place mondiale dans ce secteur, au travers de deux divisions, Sièges Europe et Sièges US, qui se caractérise chacune par une offre personnalisée et des gammes très spécifiques suivant les marchés.

✈ Galley and Equipment :

L'activité Galley & Cargo Equipment assure la conception, la production et le service après-vente des galleys, des équipements de galley, en particulier des trolleys, et des équipements Cargo (containers) destinés aux avions commerciaux.

- Cabin Interiors :

La branche Cabin Interiors intervient dans deux grands domaines d'activité, la conception d'équipements embarqués et l'aménagement d'intérieurs de cabine d'avions civils.

- Zodiac Services :

Afin d'améliorer et d'optimiser son savoir-faire tout en restant à l'écoute de ses clients, le groupe a développé une activité service offrant un service clients **ZODIAC SERVICES** centralisé pour l'ensemble des produits du Groupe.

Zodiac Aerospace Maroc (ZAM)

Pourquoi choisir le Maroc ?

Les leaders mondiaux du secteur aéronaval envisagent une extension massive vers l'étranger. Ils ciblent les pays en cours de développement pour l'implantation de nouvelles unités industrielles. La concurrence est donc de plus en plus sévère et seuls les meilleurs peuvent subsister. Dans un marché qui est totalement mondial, le Maroc profite de son contexte socio-économique favorable, pour devancer des concurrents régionaux très tenaces.

Plus de 30 ans d'expérience industrielle en aéronautique, proximité géographique et culturelle, disponibilité de compétences et mains d'œuvre hautement qualifiée sont tous des atouts qui font que le Maroc tient bon sur le terrain de compétition.

De nos jours l'enjeu est de taille ; le tissu industriel local continue à s'élargir. Plusieurs projets de développement soutiennent l'attractivité du pays ; on site entre autres :

→ Le développement de la sous-traitance

→ L'ouverture sur les marchés internationaux

→ L'engagement des établissements de formation dans la qualification des cadres et techniciens du secteur.

Tous ces facteurs font de la création d'une filiale de ZODIAC AEROSPACE au Maroc une vision à des perspectives stratégiques.

Présentation du ZAM

La société ZODIAC AEROSPACE Maroc, filiale du groupe ZODIAC AEROSPACE fait partie de la branche AIRCRAFT SYSTEMS. Elle s'organise en 5 entités à savoir :

- Service Qualité et Sécurité.
- Administration Finances/Comptabilité
- Production Mécanique.
- Production Electronique.
- Pôle ingénierie.

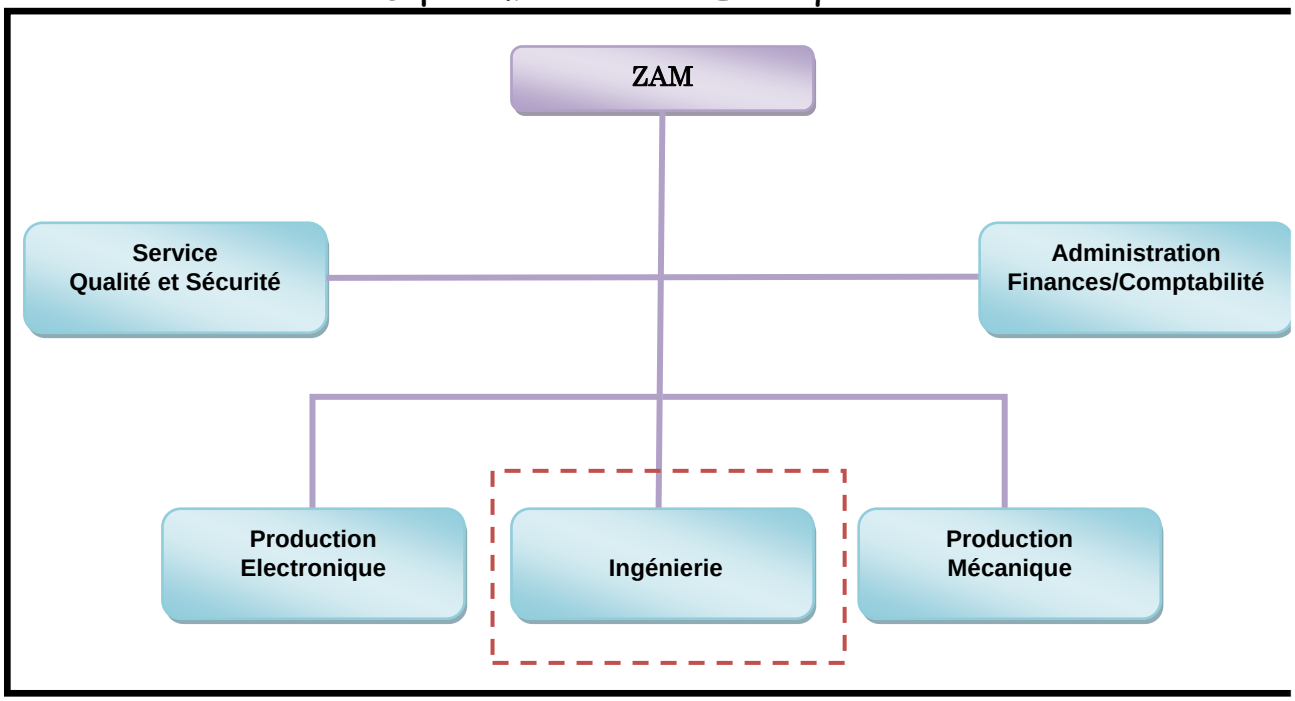

Figure 2 : départements de Zodiac Aerospace Maroc

Présentation du pôle ingénierie

Le département ingénierie est organisé en 2 Pôles :

- ***Pôle logiciel & FPGA*** : inclus toutes les équipes qui développent des logiciels.
- ***Pôle bureau d'études*** : inclus toutes les équipes qui font le design des parties mécaniques ou électroniques.

Le pôle logiciel & FPGA est constitué principalement de 4 équipes*:*

→ D7 SW Team**.**
→ ECE SW Team.
→ ECE FPGA Team.
→ ZAM Engineering Quality team.

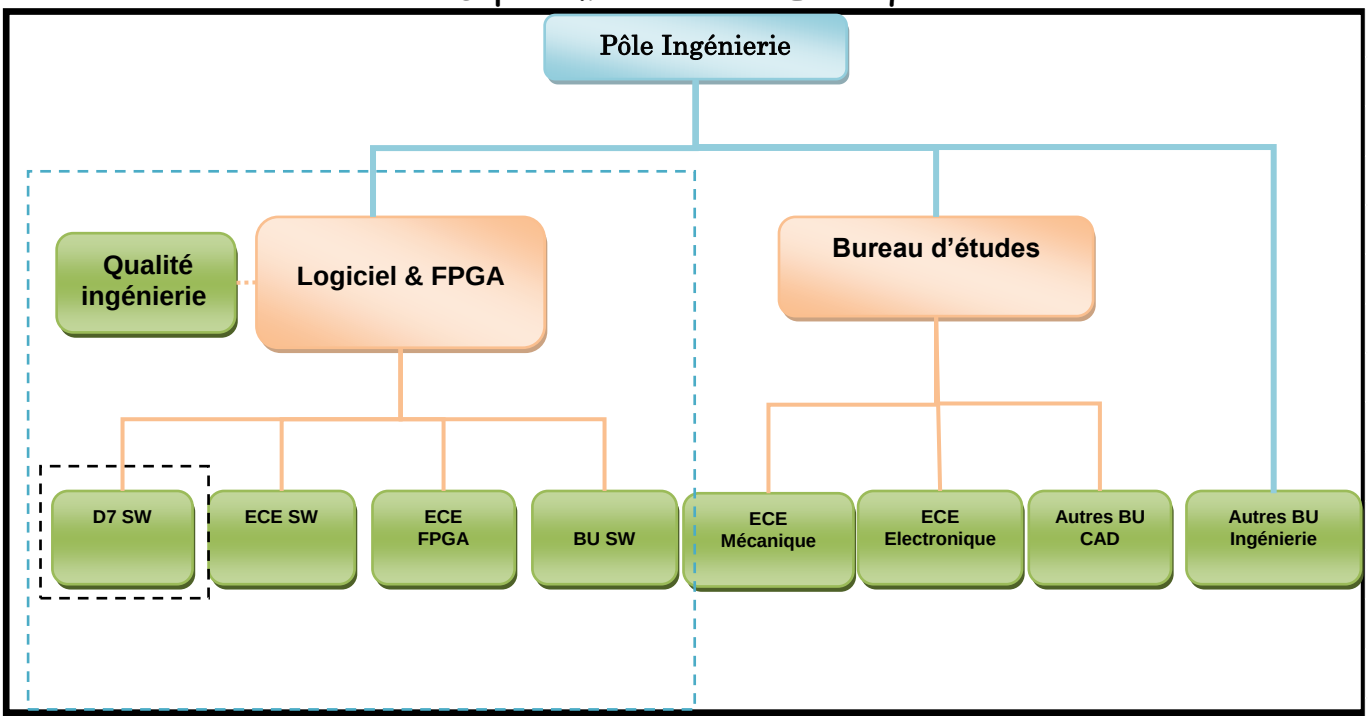

Figure 3 : équipes de Pôle ingénierie

Présentation de l'équipe de travail

Le stage se déroule au sein de l'équipe ***D7***. Cette équipe , qui se compose de 32 personnes, est divisée en quatre équipes :

- Tests unitaires.
- Validation et vérification.
- Développement.
- Qualité et assurance des processus.

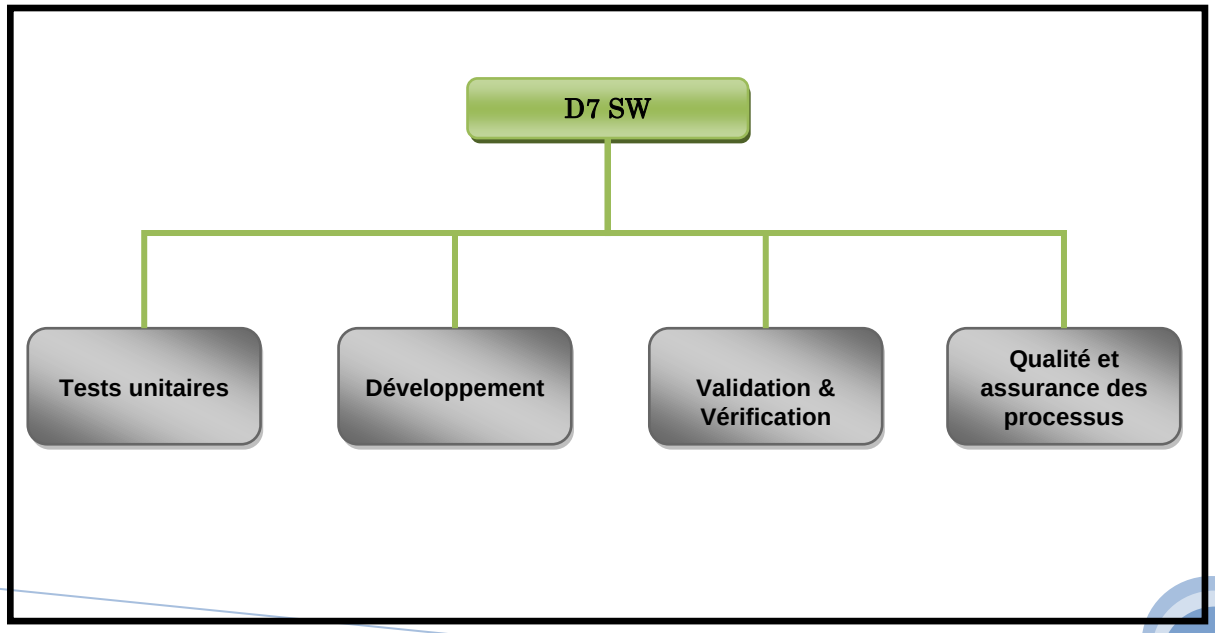

Figure 4 : équipes D7

L'équipe ***D7*** de ZAM prend en charge les projets de la société INTERTECHNIQUE.

Ce département a en charge l'étude et le développement des composants suivants :

- ***Capteurs :*** Capteur de jaugeage de carburant (jauges de réservoir), jauge d'huile du moteur Ces capteurs peuvent être équipés de têtes intégrant des circuits électroniques de traitement du signal. Capteurs de détection de niveau à capacitance ou à thermistance, Capteur de mesure de température et d'humidité
- ***Actionneurs :*** Actionneurs linéaires à moteurs électriques pas à pas, actionneurs rotatifs de vannes à moteurs électriques continu et pas à pas.
- ***Indicateurs :*** Indicateurs à LCD et à aiguilles pour l'affichage de nombreux paramètres par exemple, la quantité de carburant et débit de carburant, les paramètres électriques, la température, les paramètres moteurs… etc.
- ***Calculateurs :*** ce sont essentiellement des calculateurs numériques temps réels qui assurent de nombreuses fonctions telles que (voir exemple figure 5) :
  - ✓ Mesure de paramètres
  - ✓ Régulation
  - ✓ Surveillance de paramètres de mesure et d'informations discrètes
  - ✓ Couplage à de nombreux types de Bus : CAN, ARINC 429, RS 232ou 485, AFDX…
  - ✓ Mémorisation d'informations pour l'aide à la maintenance

Figure 5 : Calculateur

❖ Clients d'Intertechnique

Les principaux clients d'Intertechnique sont :

AIRBUS ; **BOEING** ; **EMBRAER** ; ALENIA ; ANTONOV ; ATR ; BAE ; SYSTEM ;…..

DEVELOPPEMENT LOGICIEL

Logiciel embarqué

Définitions

- **Logiciel :**

Programme informatique et, éventuellement, documentation et données associées concernant le fonctionnement d'un système informatique.

- **Logiciel embarqué :**

Logiciel contenu dans un équipement ayant une vocation principale autre qu'informatique

→ Téléphones portables, appareils ménagers, automobiles, aéronefs…

- **Logiciel embarqué aéronautique :**

Logiciel embarqué dans un système monté sur un aéronef (avion, hélicoptère, drone)

→ Systèmes de navigation, pilote automatique, régulation moteur.

Caractéristiques

L'informatique embarquée a des impératifs différents de l'informatique personnelle (les micro-ordinateurs). Ce sont principalement :

- La criticité : Les systèmes embarqués sont souvent critiques, et les systèmes critiques sont presque toujours embarqués. En effet, comme un tel système agit sur un environnement physique, les actions qu'il effectue sont irrémédiables. Le degré de criticité est fonction des conséquences des déviations par rapport à un comportement nominal, conséquences qui peuvent concerner la sûreté des personnes et des biens, la sécurité, l'accomplissement des missions, la rentabilité économique,
- La réactivité : ces systèmes doivent interagir avec leur environnement à une vitesse qui est imposée par ce dernier. Ceci induit donc des impératifs de temps de réponses. C'est pour cette raison que l'informatique embarquée est souvent basée sur un système temps réel
- L'autonomie : Les systèmes embarqués doivent en général être autonomes, c'est-à-dire remplir leur mission pendant de longues périodes sans intervention humaine. Cette autonomie est nécessaire lorsque l'intervention humaine est impossible, mais aussi lorsque la réaction humaine est trop lente ou insuffisamment fiable.
- La robustesse, sécurité et fiabilité : L'environnement est souvent hostile, pour des raisons physiques (chocs, variations de température, impact d'ions lourds dans les systèmes spatiaux, ...) ou humaines (malveillance). C'est pour cela que la sécurité - au sens de la résistance aux malveillances - et la fiabilité - au sens continuité de service - sont souvent rattachées à la problématique des systèmes embarqués.
- Et enfin des contraintes non fonctionnelles, comme par exemple l'occupation mémoire, la consommation d'énergie, ...

En résumé, le logiciel embarqué aéronautique n'est pas que le code source qui permet de programmer le microprocesseur, mais toute la documentation qui permet :

- De définir ce que fait le logiciel**,**
- De décrire comment le logiciel est conçu,
- De décrire comment le logiciel a été vérifié,
- Plus généralement, toutes les preuves qui permettent de démontrer que le logiciel réalise correctement les fonctions prévues, et que la sécurité est assurée.

→ *Quelques chiffres :*

- Entre 1983(Airbus A310) et 1993(Airbus A340) le nombre d'équipements embarqués a augmenté de 50%(de 77 à 115).
- La puissance embarquée a été multiplié par 4(60 à 250 Mips)
- Le coût de l'avionique correspond à 30% du coût total d'un appareil civil, et jusqu'à 50% du coût d'un appareil militaire.
- Le développement de l'avionique a démarré en 1960 par le remplacement progressif d'équipements analogiques par des équipements numériques.

Document de référence DO-178B

Lors du développement d'un logiciel embarqué aéronautique, trois entités sont présentes :

- ✈ Le systémier ou équipementier (ZODIAC AEROSPACE)
- ✈ L'avionneur (Le client : AIRBUS, EMBRAER…)
- ✈ Les autorités de certification (EASA, FAA…)

La norme DO-178B est utilisée en aéronautique civile et militaire, pour les logiciels intégrés dans les systèmes et équipements de bord.

Certification : DO-178B

Reconnaissance légale par l'autorité de certification du fait qu'un produit, un service, une organisation ou une personne est conforme aux exigences. Une telle certification comprend l'activité de contrôle technique du produit, du service, de l'organisation ou de la personne et la reconnaissance formelle de la conformité avec les exigences applicables par émission d'un certificat, d'une autorisation, d'une approbation ou de tout autre document en fonction des lois, approbations et procédures nationales.

❖ *Conditions de défaillance*

La DO-178B utilise sur l'échelle de gravité suivante :

- ***Catastrophique*** **:** Conditions de panne susceptibles d'empêcher la poursuite en toute sécurité d'un vol et d'un atterrissage.
- ***Dangereuse*** : Conditions de panne susceptible de réduire les possibilités de l'aéronef ou la capacité de l'équipage à faire face à des conditions hostiles.
- ***Majeure*** **:** Conditions de panne susceptibles de réduire les possibilités de l'aéronef ou la capacité de l'équipage à faire face à des graves conditions hostiles.
- ***Mineure*** : Conditions de panne n'engendrant pas de réduction significative de la sécurité de l'aéronef, et susceptibles d'entrainer pour l'équipage des actions se situant tout à fait dans le domaine de leurs capacités
- ***Sans effet***.

CYCLE DE VIE D'UN LOGICIEL

L'objet de cette norme DO-178B est de préciser quels sont les différents processus, dits processus du cycle de vie du logiciel, à mettre en œuvre au cours de l'élaboration de logiciels destinés aux systèmes et équipements de bord.

Le cycle de vie d'un logiciel est un ensemble ordonné des processus déterminés par la société comme étant suffisants et adéquats pour réaliser un produit logiciel.
Il signifie également la période de temps qui commence avec la décision de réaliser ou de modifier un produit logiciel et qui s'achève quand le produit est retiré du service.
En adéquation avec la norme DO-178Bl, le développement est découpé en plusieurs processus :

- ✈ Spécification
- ✈ Conception
- ✈ Codage
- ✈ Intégration

Au sein de ZODIAC AEROSPACE, on opte pour l'utilisation du modèle en V pour tout projet de développement. Le modèle du cycle en V décrit par la figure 6 est un modèle conceptuel de gestion de projet. Il permet, en cas d'anomalie, de limiter un retour aux étapes précédentes. Les phases de la partie montante doivent renvoyer de l'information sur les phases en vis-à-vis lorsque des défauts sont détectés, afin d'améliorer le logiciel.

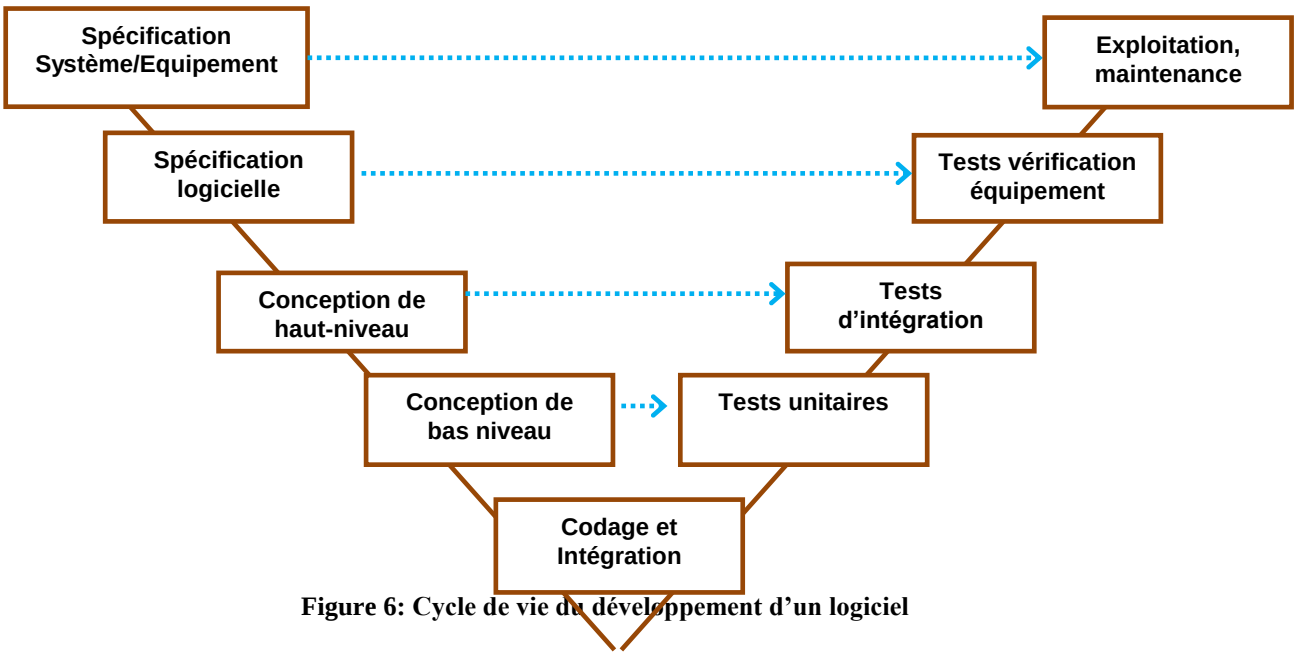

Figure 6: Cycle de vie du développement d'un logiciel

Chacune des étapes est une phase transitoire dans la vie du logiciel ; elle permet d'effectuer une tâche bien déterminée :

Spécification Système/Equipement	⇒	Cahier des charge livré par le client
Spécification logicielle	⇒	Software Requirements Document
Conception de haut-niveau	⇒	Software Design Document
Conception de bas niveau	⇒	Low Level Requirement

Remarque :

Le projet de la réalisation du simulateur permet d'effectuer les tests informels liés à la phase d'intégration du code.

CONCLUSION :

Dans cette partie du rapport, on a présenté les différentes activités de Zodiac Aerospace Maroc, ces clients et le développement logiciel au sein de cet organisme.

Chapitre II

Contexte général

Ce chapitre est une description globale du contexte du projet ; il donne une vue générale de sujet de travail ainsi que le plan de d'avancement ayant encadré le déroulement des différentes tâches.

PRESENTATION DU PROJET

Cadre général :

Les logiciels embarqués sur avion sont des logiciels de niveau catastrophique et dangereux, donc des conditions de panne susceptibles d'empêcher la poursuite en toute sécurité d'un vol et d'un atterrissage.

C'est dans ce cadre que les opérations de vérification et de validation sont des phases les plus importantes pour les équipementiers pour assurer la sureté de fonctionnement de ces équipements

Dans cette optique s'inscrit le projet SPDS (Secondary Power Distribution System) du département logiciel du pôle ingénierie. Ce système relie, via bus CAN, la carte CPS à :

- 8 cartes SSPC (Solid State Power Controller)
- 2 DU (Display Unit)
- Un ordinateur de maintenance MCT (Maintenance Computer Tools)

Les 2 DU (Interface Homme Machine) sont deux équipements manquant du système de distribution électrique SPDS de l'avion Legacy450/500, œuvre de l'avionneur brésilien EMBRAER.

La mission définit par le sujet consiste à développer un simulateur qui simule le comportement nominal des deux DU.

L'objectif c'est de faire un test d'intégration entre les deux entités (les 2 DU et la carte CPS) permettant de tester le bon fonctionnement du logiciel implanté dans la carte CPS.

La phase des tests informels permet de repérer les bugs et d'y remédier avant même de franchir la phase de test du cycle de vie du logiciel.

Le sujet a pour mission :

- la conception et la configuration d'un simulateur à base de l'outil CAPL sous CANalyzer capable de simuler et modéliser les 2 DU permettant d'effectuer un ensemble de test de transfert des trames entre le simulateur qui représente les deux DU et le logiciel embarqué dans la carte CPS.

- le développement d'une couche VB capable de communique à la couche CAPL et de piloter la configuration CANalyzer Permettant d'intervenir et d'agir sur les données circulant, le choix du bus à utiliser ainsi que le type de trames transférées (voir figure 7).

- le développement d'une interface graphique permettant de faciliter et d'accélérer les procédures de tests. Et d'avoir la possibilité d'agir facilement sur les donnés transférées vers la carte CPS et également la possibilité d'afficher la totalité des donnés reçus, pour vérifier que les données envoyées par la carte CPS sont les même reçu sur le simulateur.

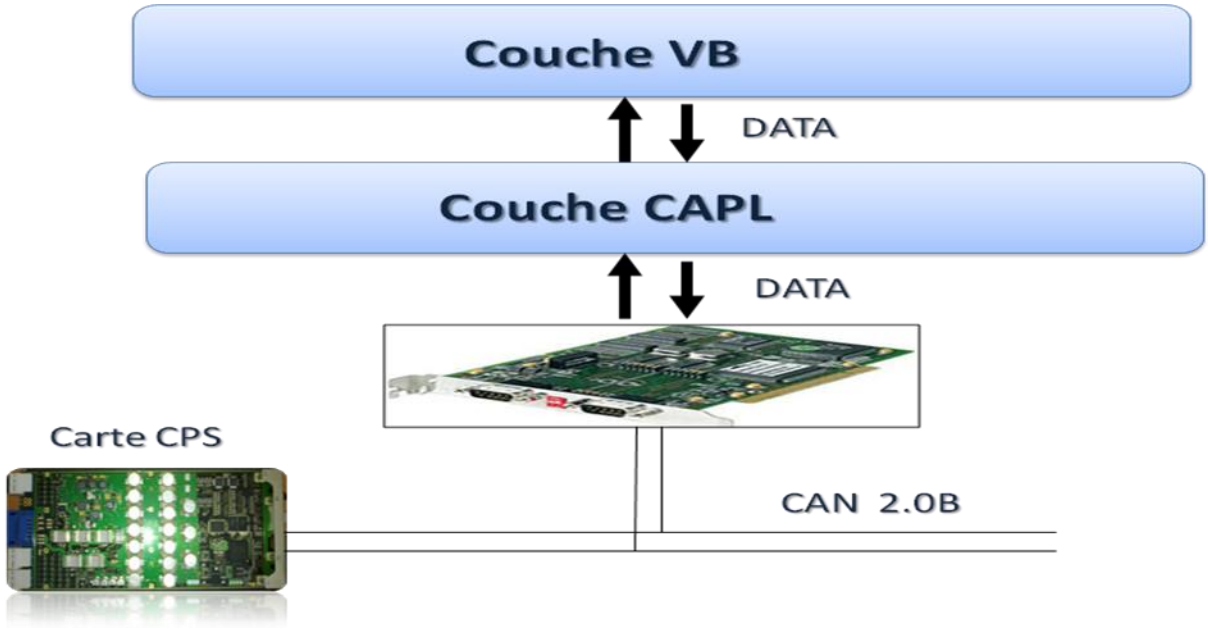

Figure 7: Schéma descriptif du simulateur

Vue globale sur le système SPDS « Secondary Power Distribution System»

L'unité de distribution secondaire SDU (Secondary Distribution Unit) est une partie du système SPDS de distribution de puissance secondaire de l'avion Legacy 450/500 (embrayer). Ce système est constitué de 4 unités SDUs et de deux unités de visualisation DUs.

La figure en dessous présente la localisation de ces équipements dans l'avion:

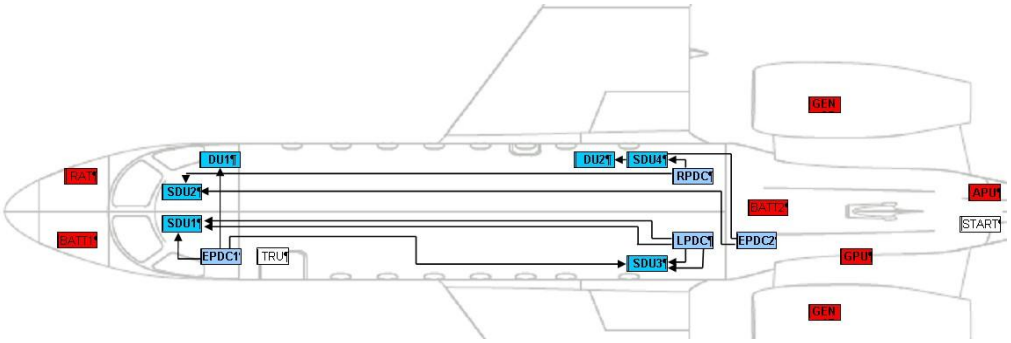

Figure 8: Localisation de l'équipement dans l'avion

Présentation du CPS system :

La figure en dessous présente le système CPS :

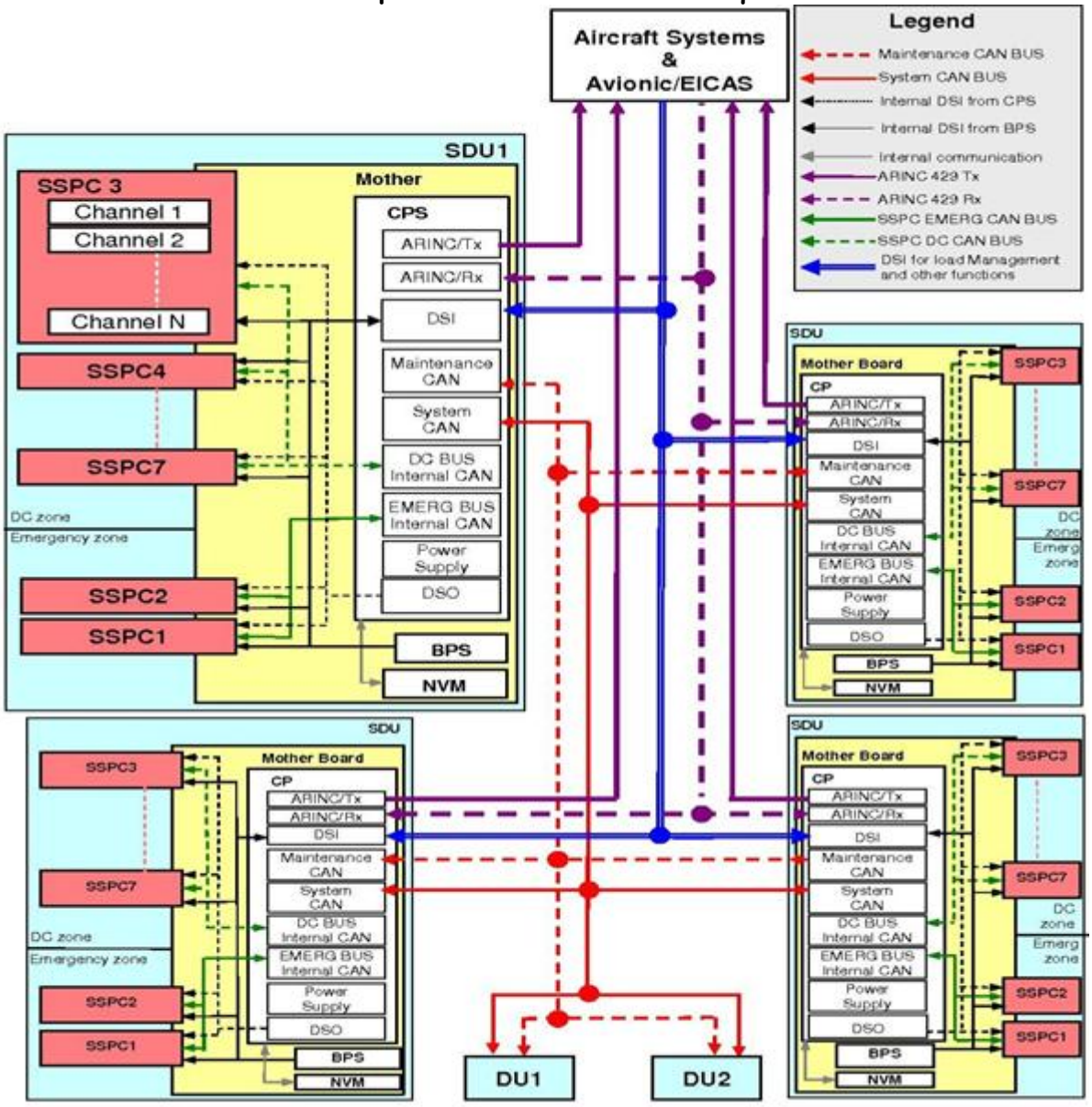

Figure 9 : CPS System

Un réseau CAN externe à deux bus est utilisé pour permettre la communication entre la carte CPS (à travers le SDU), les unités d'affichage (DUs) et l'ordinateur de maintenance (MCT):

- Le Bus Système (bus prioritaire) assure la communication avec les 2 unités d'affichage (DUs).
- Le Bus Maintenance assure la communication entre les 2 unités d'affichage (DUs) et l'ordinateur de Maintenance (MCT).

Pour la communication interne, l'unité de distribution secondaire inclut :

- Bus ARINC 429(Bus unidirectionnel)
- Des entrées discrètes (DSI) pour gérer le téléchargement
- Des entrées discrètes (DSI) pour gérer d'autres modes

Un réseau CAN interne (2bus) est utilisé pour permettre la communication entre la carte CPS et les canaux SSPC (Solid State Power Controller).

Architecture électronique

Toute unité de distribution secondaire est constituée de 3 sortes de cartes :

- **Carte de communication :** Communication Power Supply board (CPS)
- **Carte de Protection :** Solid state Power Controller (SSPC)
- **Alimentation de secours :** Back-up Power Supply board (BPS)

Le schéma suivant explique la répartition des différents composants du système :

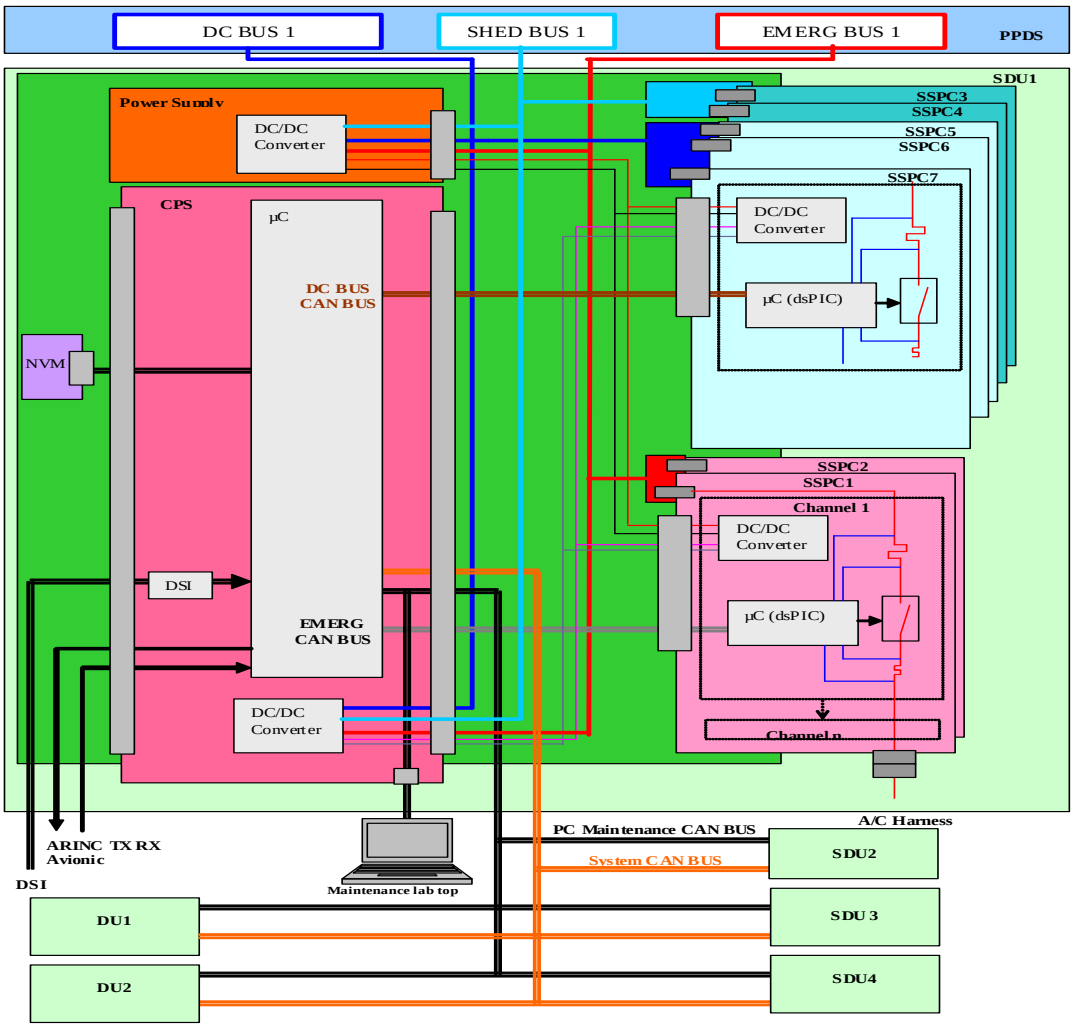

Figure 8 : Vue d'ensemble de l'architecture SPDS

Planification du projet

La planification est un outil indispensable de management de projet, un moyen essentiel de prise de décisions pour le chef de projet : pour définir les travaux à réaliser, fixer des objectifs, coordonner diverses

actions, maîtriser les moyens requis, minimiser les risques rencontrés, enfin mettre en place une stratégie de management, notamment en suivant les activités en cours et en rendant compte de l'état d'avancement du projet.

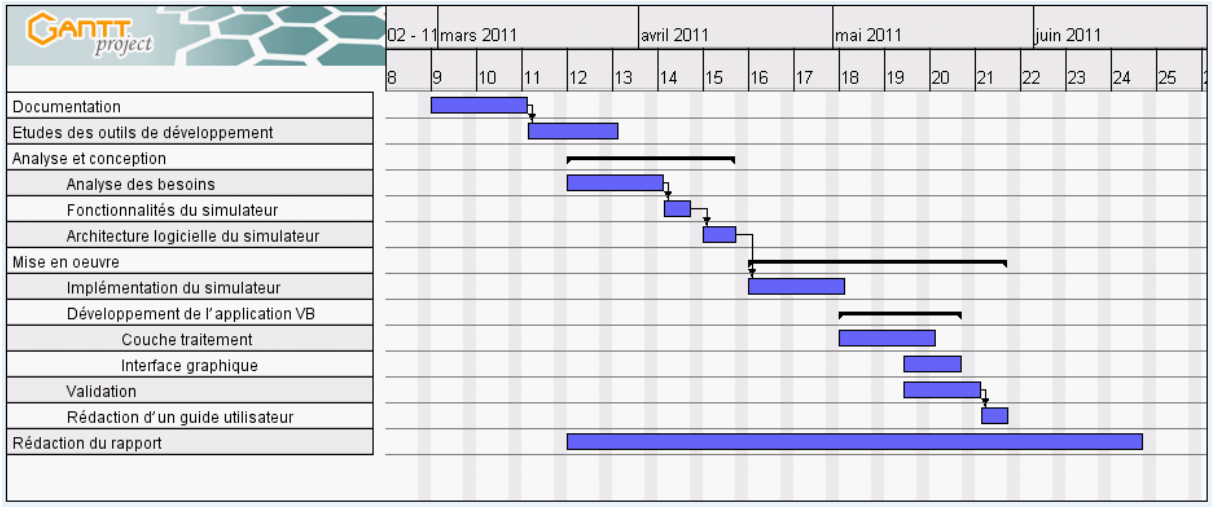

Figure 9 : plan d'avancement de stage

CONCLUSION

Dans cette partie du rapport, on a présenté le contexte du projet en décrivant les différents systèmes ainsi que leurs objectifs, et le planning du déroulement du projet.

Université Sidi Mohammed Ben Abdellah
Faculté Des Sciences et Techniques Fès
Département de Génie Electrique

Chapitre III

Cahier de charge et Problématique

Après avoir fait l'étude de contexte générale de projet et son environnement ce chapitre donne une description de cahier des charges et des fonctionnalités du simulateur.

CAHIER DES CHARGES

Le but du stage est de réaliser un simulateur pour tester l'interface de communication CAN d'un logiciel embarqué.

Le rôle de ce simulateur est de tester et de vérifier, la conformité des différentes fonctionnalités du logiciel avec celles attendues.

Le stage s'articule autour du cahier des charges suivant :

- **Définir l'intérêt et les fonctionnalités du simulateur :**

L'équipe logicielle de ZAM a en charge le développement des logiciels embarqués sur les calculateurs qui permettent la gestion de la quantité de carburant, la gestion du système électrique pour des avions.

Ce développement se fait en assurant les contraintes imposées par la norme DO-178B comme la définition des spécifications du logiciel, la définition de son architecture, son codage et sa validation, et la vérification du système sur un banc d'essai et sur un avion.

Le test de ces logiciels nécessite la présence d'une cible matérielle pour s'assurer du bon fonctionnement. C'est dans ce cadre qu'on a pensé à faire une version logicielle d'un nœud manquant du system. les deux DU représentent les deux nœuds manquant du system.

A partir des besoins identifiés, les fonctionnalités du simulateur seront définies. Celles-ci seront à la base de la conception de l'environnement du test.

- **Définir l'architecture logicielle du simulateur**

Cette architecture permettra de satisfaire les fonctionnalités du simulateur et de faire évoluer facilement la simulation des données entrées et sorties du calculateur.

- **Implémenter le simulateur :**

Cette implémentation sert à vérifier le bon fonctionnement du simulateur. Pour ce faire on passera par les étapes suivantes :

- Création de la configuration
- Chargement des programmes CAPL
- Création d'une interface Graphique, cette interface est l'intermédiaire entre l'utilisateur et le logiciel embarqué, il lui permet de saisir les valeurs des trames à transférer ainsi la récupération des trames provenant de la carte CPS.

- **Test du fonctionnement et Validation :**

Pour valider le fonctionnement correct du simulateur, des tests seront réalisés en parallèle de l'implémentation. Pour ce faire, le logiciel embarqué sur la carte CPS sera utilisé. Il y est chargé à l'aide d'un émulateur.

- **Rédaction d'un guide utilisateur**

Un tutorial relatif à la configuration et à la manipulation du simulateur est produit. Celui-ci indiquera à l'utilisateur la manière dont fonctionne le simulateur, ainsi que la façon d'interagir avec le logiciel embarqué.

ANALYSE DES BESOINS

On précise dans ce qui suit les différentes spécifications du processus logiciel prévu.

Plusieurs fonctionnalités devront être satisfaites par le simulateur. Ces fonctionnalités sont classifiées sur 3 niveaux :

Niveau de la transmission :

Dans chacune des trames transmises, le DU doit envoyer l'une des requêtes suivantes :

- Requête d'activation d'un canal SSPC
- Requête de commande d'un sectionneur
- Demande de téléchargement
- Requête de changement de mode

Le DU doit également préciser le ou les équipements cibles. Il est important que la transmission soit gérer sur les deux canaux, de manière cyclique suivant un ordre bien défini (voir figure 10 suivante).

Il est également prévu de permettre à l'utilisateur de manipuler les données transportées par les trames. L'utilisateur peut saisir les trames à transmettre sur le bus CAN. Il pourra ainsi modifier ces trames à tout moment de la simulation.

Au démarrage de la simulation, le processus nominal de fonctionnement des deux DUs est lancé. Ceci comprend l'envoi cyclique des trames vers les autres équipements du système, la réception des données reçues. Le choix d'un mode de fonctionnement, l'affectation du canal de transmission approprié à chacune des trames, sont aussi des fonctionnalités à prendre en considération pendant la phase initiale.

Trame DU→All	**Période**	**Window (Cycle)**																			
		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
SPDS Number	10Te	x										x									
StdIdent	10Te		x										x								
OptIdent	10Te			x										x							
DUPN	10Te				x										x						
Additional	5Te					x					x					x					x
BITEResult	10Te						x										x				
Broad -Casting1	10Te							x										x			
Broad -Casting2	10Te								x										x		

Broad -Casting3	10Te								x									x	

Figure 10 : Description dynamique des trames DU→All

Nous appelons « Window », le cycle élémentaire pendant lequel une trame est transmise sur chacun des canaux du Bus CAN. La durée de chaque cycle est de Te=10ms.

Il est nécessaire que l'utilisation du réseau soit asynchrone entre la carte CPS, les deux DUs ainsi que l'ordinateur de Maintenance s'il est connecté.

En mode Normal et mode maintenance, le calendrier pour la transmission de message doit respecter le tableau ci dessus

Niveau de la réception

Le simulateur permet de visualiser la totalité des valeurs des différents signaux reçus.

Test de robustesse

Afin de vérifier quelques fonctionnalités du calculateur en réponse à des comportements inattendus des DUs, un test de robustesse doit être effectué. Il est ainsi demandé de permettre à l'utilisateur d'activer des actions particulières à n'importe quel moment de la simulation.

Ce mode permet 3 tests :

- Test par envoi des trames de données invalides dites NCD (No Computed Data) : ce qui permet de vérifier si le calculateur rejette ces trames.
- Test de la double requêtes : ce test envoie 2 requêtes successives en moins de 1s ; le calculateur doit respecter la priorité en ne considérant que la première requête arrivée.
- Test de la requête de connexion d'un ordinateur de maintenance : à la réception d'une trame de demande de connexion avec le MTC le calculateur doit adopter le mode maintenance et donc lire sur le Bus Maintenance.

ETUDE DE L'ANCIENNE APPLICATION

Une ancienne application a été déjà développée par la société, l'étude de l'ancienne application était nécessaire afin de déceler les points forts et les point faibles et afin de proposer une nouvelle solution qui permet d'accélérer et de faciliter les procédures de test

Points faibles

Malgré que l'ancienne application à répondu strictement à cet objectif, elle présente des inconvénients suivants :

- difficulté de Manipulation

- Utilisation d'une console Write pour le test
- Manque d'une interface graphique
- Latence des procédures de test
- Difficulté d'afficher la totalité des données reçues.
- Complexité de programme utilisé
- Utilisation de nombreuse fonction CAPL(Communication Access Programming Language).

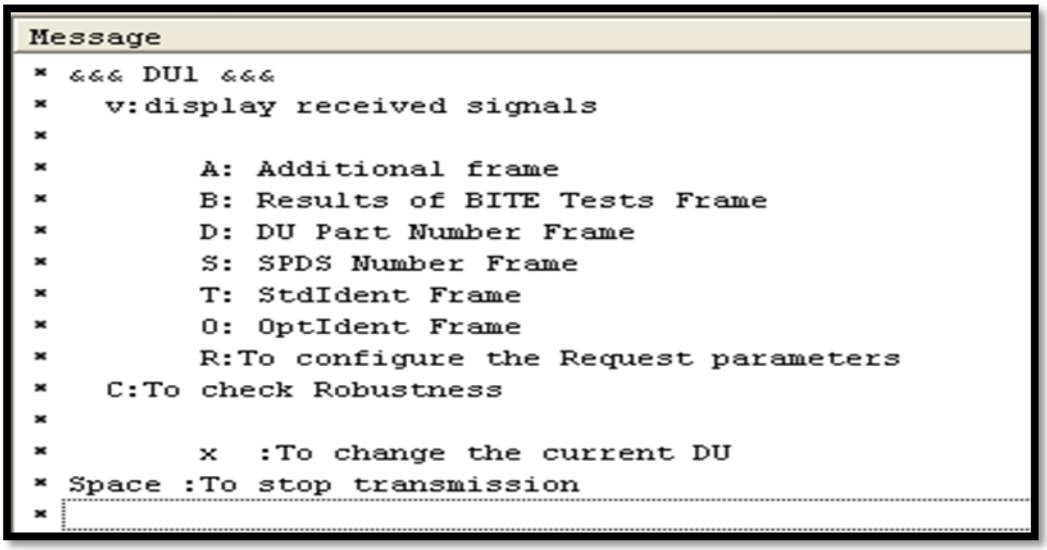

Figure 11 : console Write

APPORT DU PROJET :

L'objectif consiste à concevoir une nouvelle version logicielle des deux Unités d'Affichage DU. Pour pouvoir tester la gestion de la communication entre la carte CPS et les deux DU, en offrant :

- ➔ Meilleur flexibilité.
- ➔ Plus de rapidité.

Grâce à notre nouvelle version logicielle, ces inconvénients n'existent plus. Notre nouvelle version logicielle offre :

- ☺ *flexibilité de Manipulation (test)*
- ☺ *Création d'une interface graphique*
- ☺ *Plus de rapidité des procédures de test*
- ☺ *Facilité d'afficher la totalité des données reçues.*
- ☺ *Clarté du programme CAPL.*

ARCHITECTURE DU SIMULATEUR

Le simulateur est développé avec deux langages différents VB.NET et CAPL, donc une application VB capable de piloter une configuration sous le logiciel CANalyzer à travers un COM server.

L'interaction avec le logiciel de simulation implémenté sur la carte CPS se fait sur deux niveaux.

- niveau de transmission ; il permet d'agir sur les données sortantes du simulateur via l'interface graphique

- niveau de récupération : caractérisé par sa fonction principale qui est l'affichage et la visualisation de la totalité des valeurs des différents signaux reçus.

La figure suivante explique l'interaction du simulateur avec le logiciel embarqué :

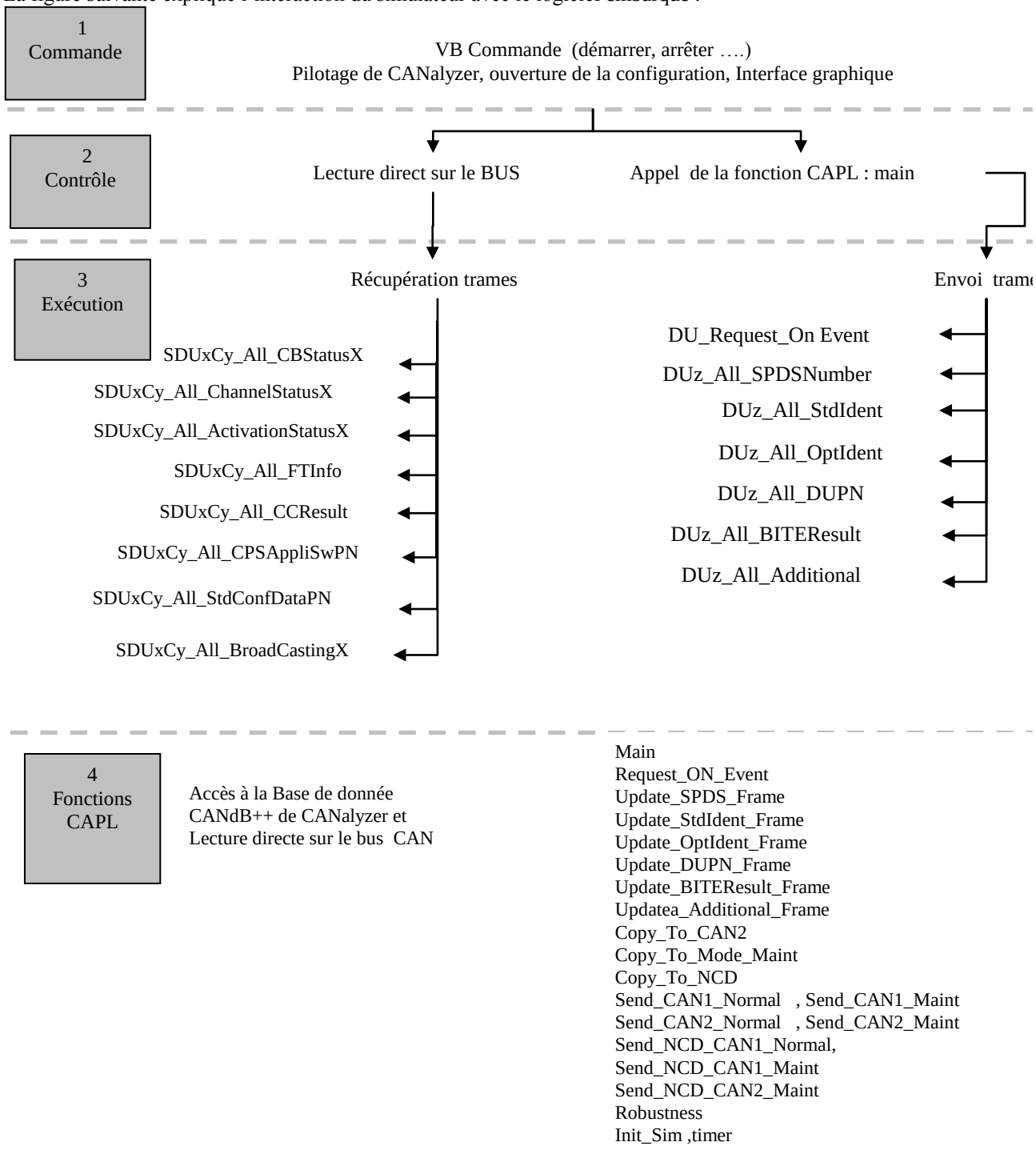

Figure 12 : Architecture fonctionnelle de simulateur

La figure ci-dessus représente l'interaction du simulateur avec le logiciel embarqué :

- La première étape représente la partie commande, elle permet le démarrage et l'arrêt du CANalyzer via l'interface graphique.

- La deuxième étape : une fois la communication est établit l'application VB communique avec la fonction CAPL **main** pour l'envoi des trames (exécution). L'application VB permet en plus de l'envoi, la récupération des trames par une lecture directe sur le bus CAN sans la nécessite d'une fonction CAPL ou d'une procédure événementielle.

Les fonctions CAPL permet l'envoi des trames provenant des deux nœuds DU sur le même bus CAN. Ces fonctions permettent d'agir sur les signaux qui forment ces trames (ex : **Update_SPDS_Frame**) et la transmission cyclique de ces trames à l'aide de la fonction **Send_CAN1_Normal.**

Les autres fonctions sont développées pour produire des fonctionnalités particulières, générer des comportements erronés définis par l'utilisateur, pour tester la réaction du logiciel vis-à-vis ce genre de comportements.

CONCLUSION

Dans cette partie du rapport, on a présenté le cahier des charges et les fonctionnalités assurées par le simulateur.

Université Sidi Mohammed Ben Abdellah
Faculté Des Sciences et Techniques Fès
Département de Génie Electrique

Chapitre VI

Outils de développement

Cette phase du projet consiste à étudier les outils de réalisation du projet. Donc on a consacré ce chapitre pour faire une description de différents outils logiciels utilisés.

OUTILS LOGICIELS

Afin de franchir les différentes étapes de la réalisation du simulateur, plusieurs outils ont été utilisés. Le tableau ci-dessous liste l'ensemble de ces logiciels

Logiciel	Utilisation	Constructeur	Référence	Version
CANalyzer	Création de la configuration et implémentation du simulateur.	Vector	CANalyzer	5.2
CAPL Browser	Développement et débogage du code CAPL.	Vector	CANalyzer	5.2
CANdB++ Editor	Création de la base de données.	Vector	CANalyzer	5.2
COM server	contrôleur de CANalyzer.	Vector	CANalyzer	5.2
VB.NET	Développement de l'interface graphique	Microsoft	_	10

Figure 13 : Table des outils logiciels

Le logiciel « CANalyzer »

CANalyzer est l'outil d'analyse universel pour les réseaux et les systèmes répartis. Ce produit facilite l'observation, l'analyse et la simulation du trafic de données dans le réseau CAN. Le logiciel est contrôlé à partir d'un tableau de bord sous forme de diagramme, qui permet d'observer le trafic du bus à travers diverses fenêtres d'analyse et de mesure. L'utilisateur peut en outre placer des blocs fonctionnels sur le schéma avant de les configurer, par exemple, des blocs tels que 'Filtre', 'Nœud CAPL'.

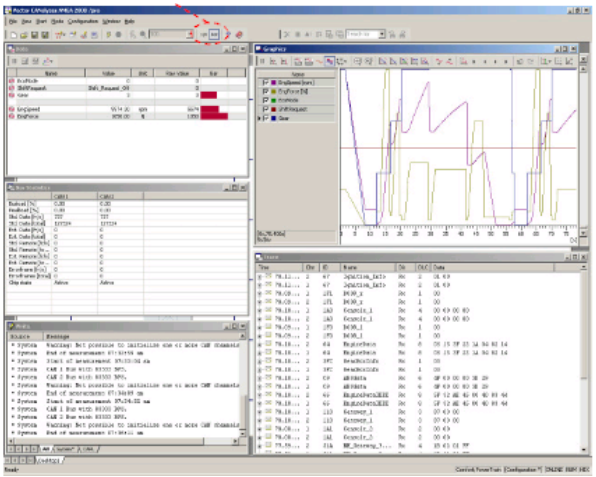

Figure 14 : block d'analyse

✈ **La fenêtre trace affiche le trafic du bus durant la mesure**

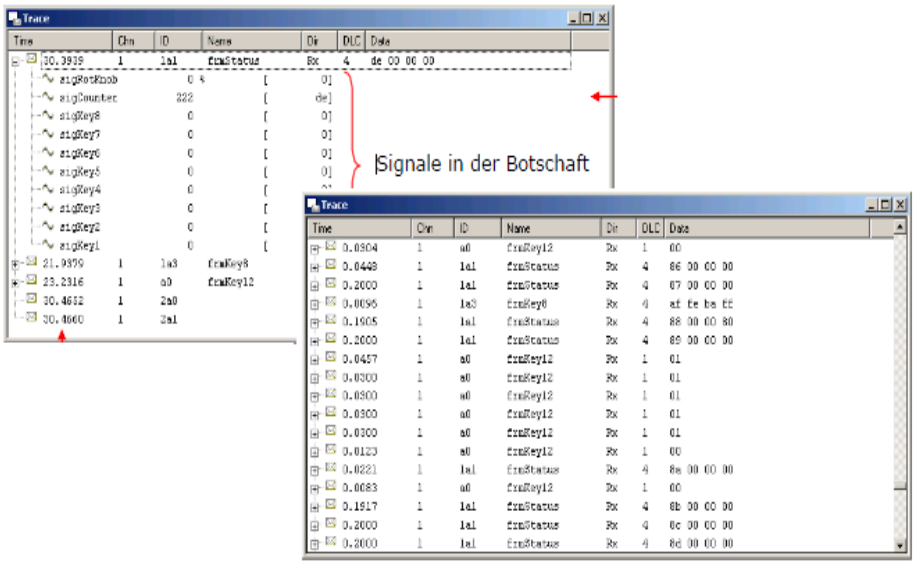

Figure 15 : Fenêtre trace

Les fonctions de base intégrées dans le CANalyzer proposent une multitude de possibilités de mise en œuvre, notamment :

- ✈ Affichage du trafic de données de bus (trace)
- ✈ Affichages graphique et textuel des valeurs de signal
- ✈ Envoi interactif de messages prédéfinis
- ✈ Statistiques concernant les messages
- ✈ Acquisition et analyses du trafic bus
- ✈ Sert comme environnement de test (messages et erreurs)

Les deux outils CAPL Browser et CANdB++ Editor sont complémentaires à la configuration de base. Ils permettent respectivement de générer et déboguer le code CAPL, et de créer la base de données comportant les différentes trames utilisées.

Configuration CANalyzer

Cette configuration permet de visualiser le trafic sur le bus CAN.

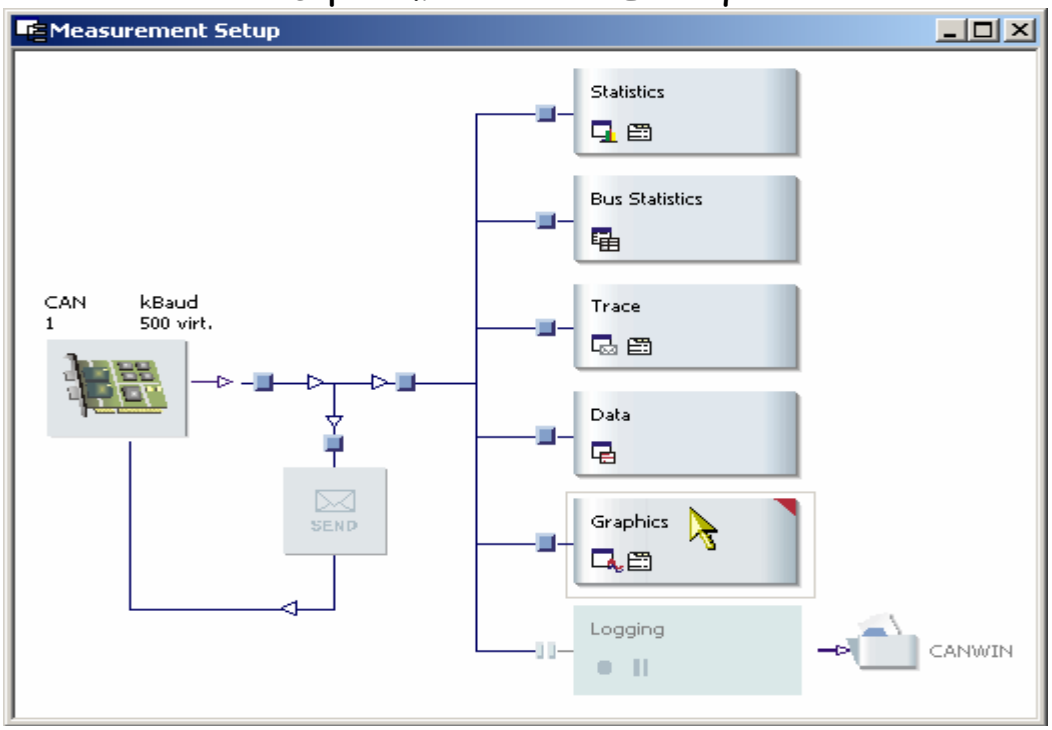

Figure 16 : Configuration CANalyzer

Ci-dessus est la description de chaque élément de la configuration CANalyzer :

- **CAPL nodes « P » :** Les nœuds CAPL sont insérés sous forme de blocks fonctionnels. Ils sont configurés afin que les données puissent être reconnues à la réception. Chaque nœud CAPL émule le comportement d'un DU.

- **Pass Filters « PF » :** Les filtres sont utilisés dans l'optique de gérer le passage des trames dans les fenêtres d'affichage.

- **Trace windows :** Les fenêtres visualisent le flux de données échangées avec les autres nœuds.

Construction des bases de données : CANdb

CANdb++ permet de décrire l'ensemble des objets utilisés dans un projet (un réseau CAN ou un système de réseaux CAN), Ces objets sont les objets de communication (messages, signaux).

L'ensemble des trames gérant la communication au sein du système se partagent en 4 grandes familles. Chaque famille de trames adopte un système d'identification (Nomenclature, Identifiant) approprié. La répartition des trames se fait suivant le mode de fonctionnement. On distingue donc, des trames du mode Normal celles du mode Maintenance et les trames de données invalides correspondantes à chaque mode. De ce fait, il y a 4 bases de données différentes.

Pour mieux expliquer le concept, l'exemple suivant illustre les spécifications relatives à une trame type du système :

Frame Mapping

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Byte 0	SPDS Interface Number MSB							
Byte 1	SPDS Interface Number LSB							
Byte 2	CPS-DU Download Compatibility Number MSB							
Byte 3	CPS-DU Download Compatibility Number LSB							
Byte 4	Spare							
Byte 5	DU Request On Event (Refer to 4.2.2.1.2)							
Byte 6								
Byte 7								

Data Description

Name	Type	Value	Role
SPDS Interface Number	16 bits	Unsigned integer	Compatibility Check number between CPS and DU (cf IDD_CAN_SPDS_GLOBAL_006)
CPS-DU Download Compatibility Number	16 bits	Unsigned integer	Download Compatibility Check number between CPS and DU (cf [2])
DU Request On Event	NA	NA	Refer to 4.2.2.1.3

Frame Id Values

CAN1			CAN2		
Frame Name	Normal	Mainten-ance	Frame Name	Normal	Mainten-ance
DU1_All_SPDSNumber_CAN1	0x145181D3	0x155181D3	DU1_All_SPDSNumber_CAN2	0x145181E3	0x155181E3
DU2_All_SPDSNumber_CAN1	0x145182D3	0x155182D3	DU2_All_SPDSNumber_CAN2	0x145182E3	0x155182E3

Figure 17 : Spécifications de la trame « DUy_All_SPDSnumber_CANw »

Introduction au langage CAPL

Le langage de programmation CAPL (Communication Access Programming Language) permet d'écrire des programmes pour des applications individuelles. Notamment, dans des cas de simulation de l'environnement d'un système qui n'est pas disponible ou encore le trafic des données d'un ensemble de nœuds physiquement non-existants. Il est également possible de programmer des éléments de liaison afin que les données puissent être échangées entre les différents bus. Les fonctions CAPL sont utilisées, entre autres, pour automatiser un fonctionnement répétitif ou même pour former des interfaces pour notre cas des interfaces VB.NET.

La syntaxe du langage CAPL est similaire au langage C, mais il existe certaine fonctionnalités supplémentaires qui ne sont pas incluses en C, nous les listons ci-dessous :

- Un type de résultat manquant est interprété « void »
- Une liste de paramètres vide est autorisée (Comme le cas du C++)
- Le contrôle des paramètres est effectué comme en C++
- Des tableaux de dimension arbitraire peuvent passer

Un programme CAPL est constitué de 2 parties ; à savoir la déclaration des variables globales d'une part et la déclaration des fonctions définies par l'utilisateur en plus des procédures événementielles d'autre part.

DECLARATION DES VARIABLES GLOBALES :

En plus des types standards disponibles pour le C, le CAPL compte d'autres types de variables globales:

- **CAN Messages**, on accorde un identifiant symbolique à chaque message de la base de données déclaré comme variable globale.

✓ **Message** *TTT_RRR_xxx_CANw (ou bien l'ID du message)* message

✈ **Timer**, les variables de type timer sont appelées par la fonction:

SetTimer (**msTimer (ms)/ timer(s)** *t*, long *duration*)

✈ PROCEDURES EVENEMENTIELLES

CAPL est un langage procédural dans lequel l'exécution des blocs de programme est contrôlée par des événements. Ces blocs de programmes sont appelés des procédures événementielles. Dans une procédure événementielle il peut y avoir des réactions aux événements suivants

Réception d'un message CAN	On Message {}
Appui sur une touche clavier	On key {}
Initialisation de la	On preStart {}
configuration	On Start {}
Départ du programme	On StopMeasurement {}
Arrêt du programme	On Timer
Ecoulement du Timer	

Figure 18: liste des procédures événementielles

Le code du programme qu'on définit dans les procédures est exécuté lorsque l'événement se produit. Par exemple, on peut envoyer un message sur le bus en réponse à un appui sur une touche clavier, suivre l'apparition des messages ou encore exécuter des actions définies cycliques.

La figure suivante présente la fenêtre CAPL, qui se constitue de deux parties, la déclaration des variables globales d'une part et la déclaration des fonctions CAPL d'autre part

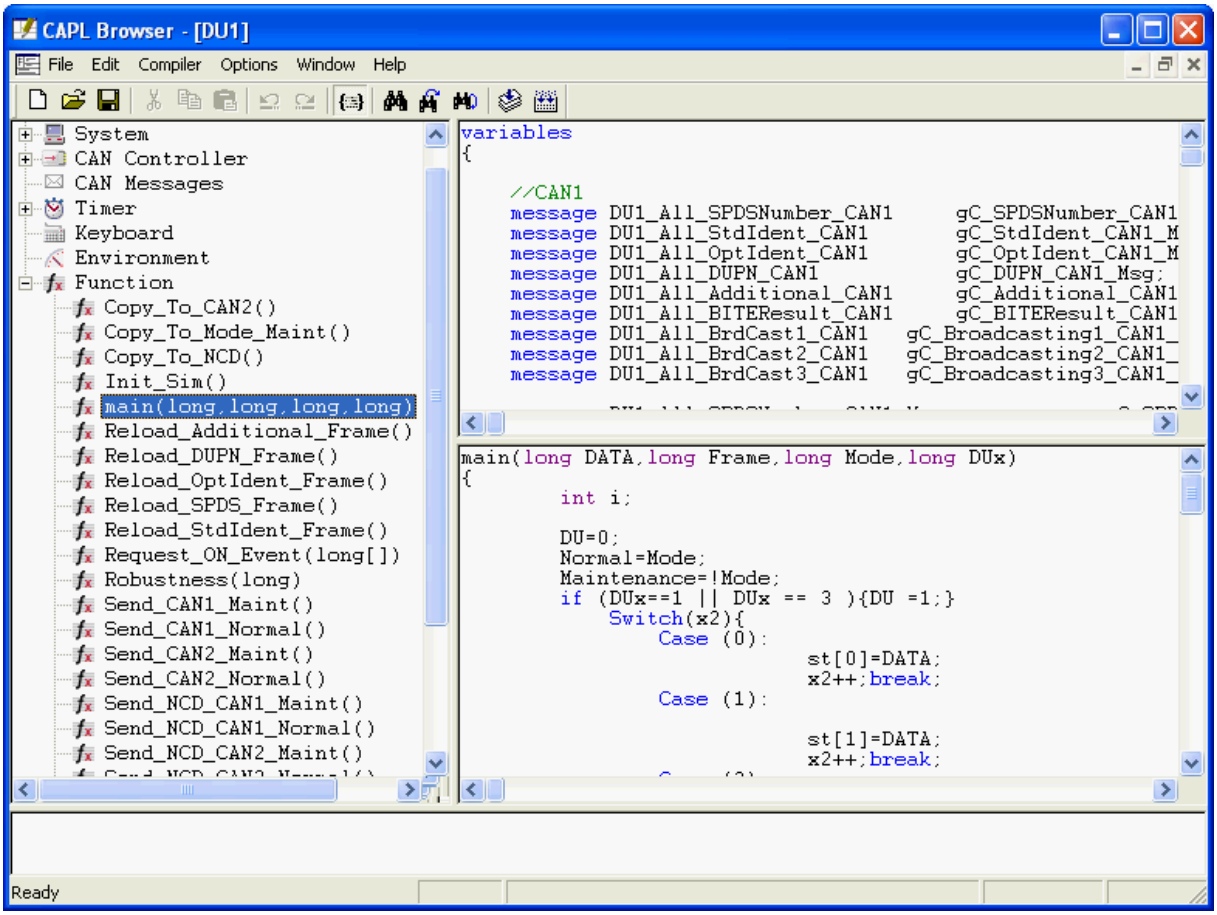

Figure 19: Localisation de la fonction main dans CAPL Browser

COM Server

COM Server permet à des applications ou à des Scripts d'avoir accès aux fonctionnalités d'autres applications.

Avec COM Server, un programme ou un Script peut contrôler le CANalyzer.

Le CANalyzer offre donc la possibilité à des programmes externes en C/C++, Visual Basic et à des Script en VBScript, JScript, Perl, VBA d'accéder à bon nombre de fonctionnalités telles que le chargement d'une configuration, le réglage du débit, l'accès à une fonction CAPL ou encore la capture de signaux, …

C'est à l'aide de ce server COM qu'on peut créer l'interface graphique en VB.NET et agir sur le code CAPL pour envoyer les trames de DU vers la carte CPS.

✈ COM SERVER -LES OBJETS COM DU CANALYZER

- ✓ Application : L'application CANalyzer
- ✓ Bus : Le bus
- ✓ BusStatistics : Les statistiques du bus
- ✓ CANBusStatistic : Les Statistiques d'un seul canal CAN
- ✓ CANController : Un contrôleur CAN

- ✓ CAPL : Une fonction CAPL qui est disponible (déclaration et affectation d'une fonction CAPL à un objet de l'application externe
- ✓ CAPLFunction : Une fonction CAPL spécifique (déclarée et affectée)
- ✓ Configuration : La configuration chargée
- ✓ Databases : Les bases de données d'une application
- ✓ Database : Une base de données d'une application
- ✓ DatabaseSetup : Les bases de données assignées à la configuration courante
- ✓ GeneralSetup : Les paramètres d'une configuration
- ✓ Logging : Un bloc logging
- ✓ LoggingCollection : Tous les blocs logging d'une configuration de mesure
- ✓ Measurement : Les fonctions d'une mesure
- ✓ MeasurementSetup : Les paramètres d'une mesure
- ✓ Signal : Un signal sur un bus CAN
- ✓ UI : L'interface utilisateur
- ✓ Write : La fenêtre write

Utiliser VB.NET pour accéder au COM server

VB.NET est un développement rapide d'application, outil qui permet à l'ingénieur pour créer des applications impressionnantes pour la plateforme Microsoft Windows.

Le serveur COM est constitué de plusieurs objets qui sont liés les uns aux autres, en utilisant une hiérarchie d'objets de certains. Chaque objet serveur COM est défini par trois éléments, qui donnent l'accès à la fonctionnalité du serveur COM:

- ➔ Méthodes des objets par exemple: **Measurement.Start**
- ➔ Objet événements exemple: **Application.OnQuit**

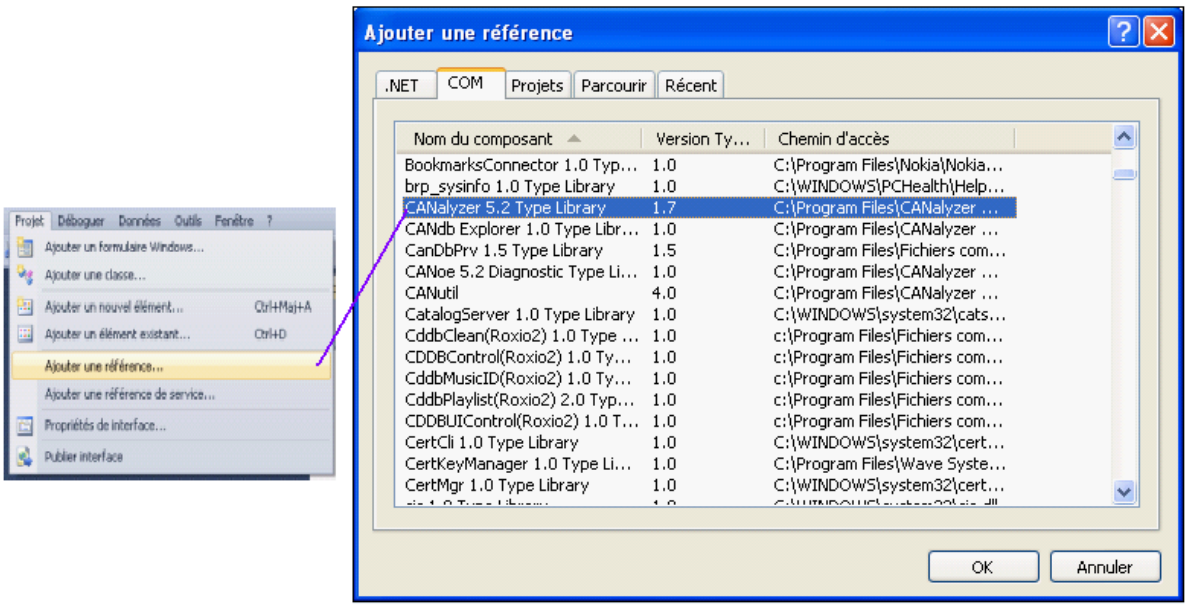

Figure 20: utiliser CANalyzer comme référence

Afin d'avoir accès au serveur COM à partir de VB.NET, le projet VB.NET doit être configuré pour utiliser la bibliothèque CANalyzer. En VB.NET ceci est réalisé à l'aide de référence du projet comme on le voit dans la figure 20.

Bus CAN Modalités et règles de fonctionnement

Les contrôleurs CAN sont physiquement petits, peu coûteux et entièrement intégrés. Ils sont utilisables à des débits importants, en temps réel et dans des environnements difficiles.

Enfin, les transmissions ont un haut niveau de fiabilité. C'est pourquoi ils ont été utilisés dans d'autres industries que l'aéronautique et des applications utilisant le CAN sont aujourd'hui disponibles dans l'agriculture, la marine, l'automobile, etc.

1.1.1. Format de trames de messages CAN

Une trame est composée des champs suivant :

Start of Frame	Champ d'arbitrage	Champ de contrôle	Champ de données	Champ de CRC	Champ d'acquittement	End of Frame
1 bit dominant	29 bits+3 bits	6 bits	0 à 8 octets	16 bits	2 bits	7 bits récessifs

Table 1:Constitution d'une trame

Le champ d'arbitrage est structuré comme suit :

Poids forts de l'identificateur	SRR	IDE	Poids faibles de l'identificateur	RTR
11 bits	1 bit	1bit	18 bits	1 bit

Table 2:Composition du champ d'arbitrage

- **SRR :** Substitute Remote Request
- **IDE** (Identifier Extension bit) : établit la distinction entre format standard et format étendu.
- **RTR** (Remote Transmission Request): détermine s'il s'agit d'une trame de données ou d'une trame de message.

Le format du message dans notre cas est conforme au CAN 2.0B (identifiant étendu). En effet, les 29 bits sont organisés ainsi :

- ✈ Statut du message : 4bits
- ✈ Identifiant : 25 bits

Le statut du message (trame) doit être fixé suivant le tableau ci-dessous :

Message Status				Message Status Value Description
03	02	01	00	
0	0	0	0	No Data
…	…	…	…	Spare (Not Used)
0	0	1	1	Normal Operation
…	…	…	…	Spare (Not Used)
1	1	0	0	Functional Test
…	…	…	…	Spare (Not Used)
1	1	1	1	No Computed Data

Table 3:Codage du « Message Status »

Les identifiants des messages CAN sont organisés selon la structure suivante :

Message Identifier																							
28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04
Priority		Type	SPDS Mode		Function						Dir		nSDU			nCPS		nDU		nMCT		CANw	
					Class			Type																

Table 4 : Organisation de l'ID des messages CAN

Priotity: Priorité de la trame
Function: Fonction réalisée par la trame
Type: Trame local / Trame système
SPDS mode: mode normal / Maintenance

Dir: Direction (source/destination) du message
nSDU: Le numéro du SDU destination
nCPS: le numéro du CPS destination dans le SDU
nDU : le numéro du DU destination
nMCT: le numéro du MCT destination
CANw: le canal utilisé

1.1.2. Description des trames utilisées dans le projet

Dans la totalité des documents relatifs au projet CPS, l'équipe utilise une nomination standard des différents messages transmis. Cette nomination a la forme suivante :

TTT_RRR_xxx_CANw

- **TTT** indique le transmetteur
  - ✓ SDUxCy : Pour CPS numéro 'y' (y=1) dans le SDU numéro x (x∈{1,2,3,4})
  - ✓ DUz : DU numéro 'z' (z= {1,2})
  - ✓ MCT : Ordinateur de Maintenance
- **RRR** indique le récepteur
  - ✓ SDUxCy : Pour CPS numéro 'y' (y=1) dans le SDU numéro x (x∈{1,2,3,4})
  - ✓ DUz : DU numéro 'z' (z= {1,2})
  - ✓ MCT : Ordinateur de Maintenance
  - ✓ All : Pour tous les CPS de tous les SDU ou pour tous les DUs.
- **xxx** indique le contenu de la trame
- **CANw** indique le canal sur lequel les messages sont transmis (w = {1,2})

CONCLUSION

Dans cette partie du rapport, j'ai présenté les différents aspects théoriques utilisés lors de la réalisation du projet.

Université Sidi Mohammed Ben Abdellah
Faculté Des Sciences et Techniques Fès
Département de Génie Electrique

Chapitre V

Developpement de L'application

Mise en oeuvre

Dans ce chapitre, nous présentons les ressources matérielles qui ont été utilisées lors de la réalisation du projet. Ainsi le fonctionnement nominal du simulateur.

CONFIGURATION DU SIMULATEUR SUR CANALYZER

Ressources matérielles

La figure ci-dessous présente l'ensemble des équipements matériels dont on s'est servi pour mettre le simulateur en œuvre.

Figure 21 : Architecture du système de test

L'émulateur-partie de TRACE32- est responsable d'effectuer le débogage des fonctions, pour la gestion des points d'arrêt (breakpoints) et l'affichage de la mémoire .Il effectue deux opérations essentielles pour l'intégration et la manipulation du logiciel :

- ✈ Interruption et relance de l'exécution
  - Chargement du code
  - Breakpoints.
- ✈ Accès en lecture/écriture
  - Visualisation et modification du contenu de la mémoire
  - Registres internes

Les tests sont effectués sur une carte CPS, conçue par la société INTERTECHNIQUE.

Configuration

L'application VB.NET doit communiquer avec le langage CAPL qu'est implémenté dans une configuration du logiciel CANalyzer. Cette configuration permet de visualiser le trafic sur le bus CAN.

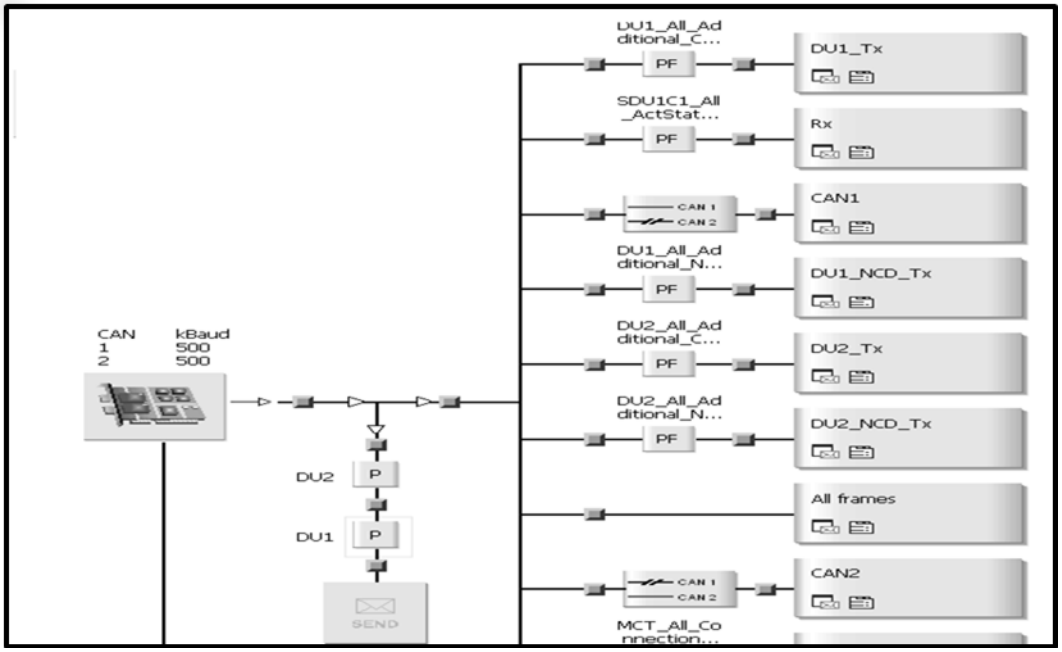

Figure 22 : Configuration CANalyzer du simulateur

Contraintes et limitations de l'application

L'objet de ce paragraphe est de préciser les contraintes et les limitations rattachées aux processus logiciels du simulateur.

- **Utilisation de VB.NET pour accéder au COM server**

COM Server permet à des applications ou à des Scripts d'avoir accès aux fonctionnalités d'autres applications. L'application VB.NET aura accès au serveur COM.

La couche VB communique avec la couche CAPL par une seule fonction <<**main**>>, on était sensé d'une boucle itérative pour transmettre tous les éléments d'un tableau en argument, puisqu'on ne peut pas passer un tableau en argument à la fois.

- **Cohabitation des deux nœuds dans une même configuration**

Puisqu'on a besoin d'implémenter les deux unités d'affichage, on est affronté à la contrainte de cohabitation de deux nœuds CAPL dans une même configuration CANalyzer. Le point principal à prendre en considération est l'emplacement des deux nœuds. Dans le programme CAPL on ajoute l'instruction qui permet de les retransmettre au nœud suivant à la réception de tous les messages. Ceci évite le blocage des trames de données dans une seule unité.

FONCTIONNEMENT NOMINAL DU SIMULATEUR

Processus de simulation

Le processus de simulation commence à partir de la page suivante, cette page est composée d'un menu, assurant le choix de test. Le test d'envoi des trames ou bien le test de la réception. L'utilisateur doit charger une configuration avant de commencer le test.

Figure 23: Menu principal de la simulation

Mode de transmission

En choisissant le test d'envoi :

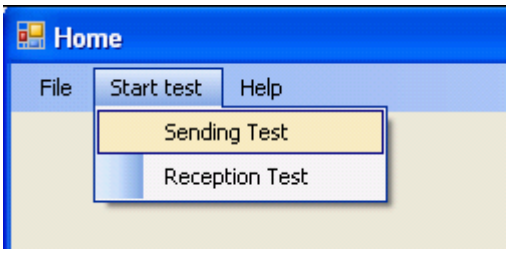

Figure 24 : choix de test d'envoi

Il nous renvoie vers la page Sending Test de notre application qui est de la forme suivante :

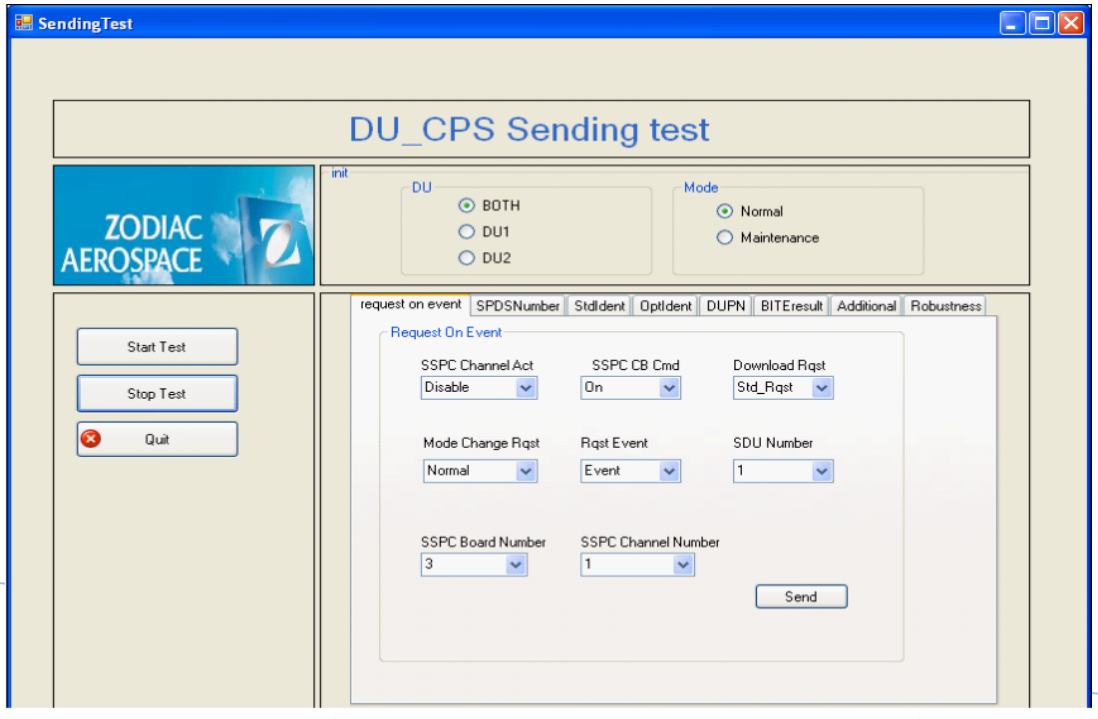

Figure 25: Mode de transmission

- Avant d'établir la communication entre le simulateur et le calculateur, un premier paramétrage est mis en place. Il permet le choix de DU (DU1, DU2 ou bien les deux), et le choix de mode (Mode normal ou maintenance).

- Une fois l'utilisateur a fait ce choix, il peut démarrer le test d'envoi (Start).

- Les trames provenant de DU1 et DU2, sont transmises sur les deux bus CAN.

- L'interaction avec le logiciel de simulation se fait sur deux niveaux. Un premier niveau de transmission, il permet d'agir sur les données sortantes du simulateur. En fait, il existe 6 types des trames de données, réparties selon la data qu'elles transportent. Le simulateur permet d'avoir la main sur les différents signaux qui forment ces trames.

- En appuyant sur les Onglets «SPDSNumber », « StdIdent », « OptIdent », « DUPN», « BITEresult», « Aditionnal » et « Robustness » l'utilisateur peut saisir les données à envoyer pour chaque trame.

- Les 3 derniers octets sont communs à toutes les trames. Ils définissent la destination et le type de la requête. Cette option est activée en cliquant sur l'Onglet « Request_On_Event »

- Un onglet « Robustness » permet de faire des tests de robustesse.

✈ Onglet « Request_On_Event »

Dans chacune des trames transmises, le DU doit envoyer l'une des requêtes suivantes :

- ✈ Requête d'activation d'un canal SSPC
- ✈ Requête de commande d'un sectionneur
- ✈ Demande de téléchargement
- ✈ Requête de changement de mode

Le DU doit également préciser le ou les équipements cibles. Il est important que la transmission soit gérée sur les deux canaux, de manière cyclique suivant un ordre bien défini. Il est également prévu de permettre à l'utilisateur de manipuler les données transportées par les trames.

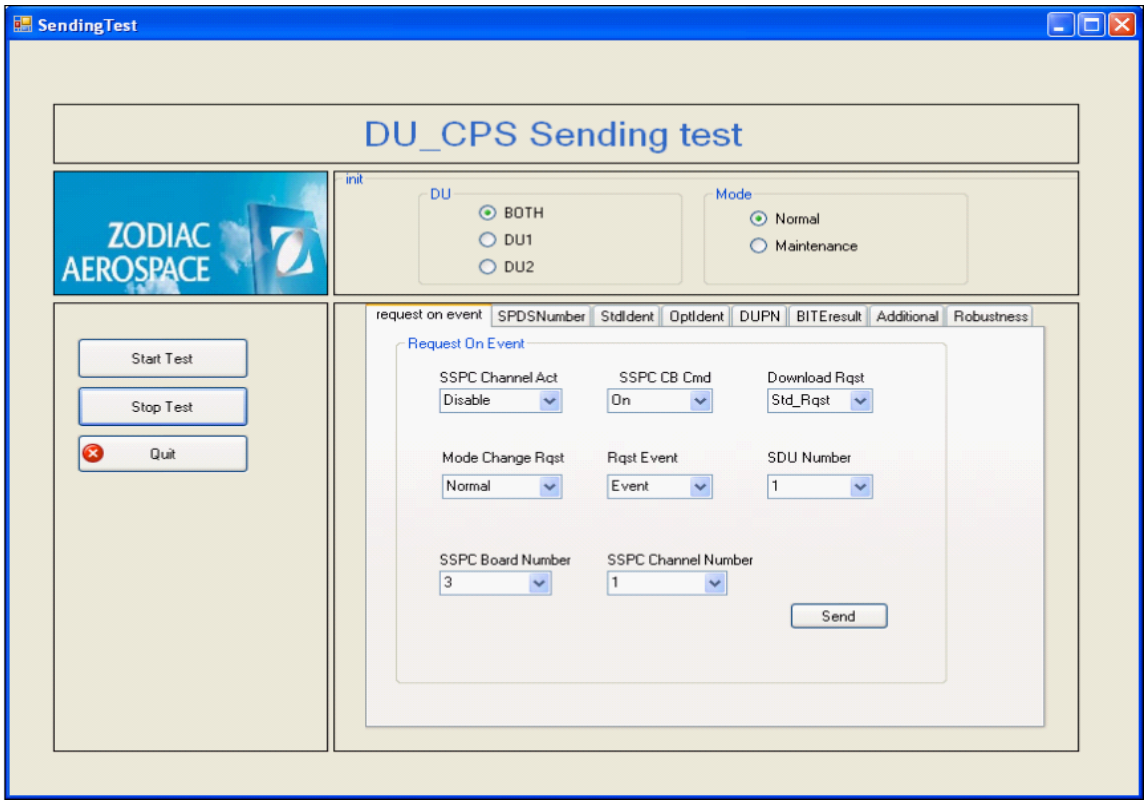

Figure 26 :Configuration de la requête

Des listes apparaissent, de sorte que vous pouvez activer la requête particulière à envoyer.
Le DU a besoin d'envoyer l'une des requêtes spécifiques suivantes:

- SSPC Channel Activation
  - 0: None
  - 1: Disable
  - 2: Enable
  - 3: Spare

- SSPC Circuit Breaker Command
  - 0: None
  - 1: Off
  - 2: On
  - 3: Lock
- Download
  - 0: None
  - 1: Std_Rqst
  - 2:Opt_Rqst
  - 3: Spare
- Mode Change
  - 0: None
  - 1: Normal
  - 2: Maintenance
  - 3: Spare

L'utilisateur détermine l'équipement à qui envoyer la requête.

- SDUx board number
  - 0 : None
  - 1 to 4 : SDU n
  - 5 to 6 : Spare
  - 7 All : SDU

- SSPCy board number
  - 0 : None
  - 1 to 7 : SSPC Board n
  - 8 to 15 : Spare
- Channel number
  - 0 : None
  - 1 to 15 : SSPC Channel n

✈ Onglet « SPDSNumber »

Des champs vides, qui vous permettent de saisir la trame<< **DUz_All_SPDSNumber_CANw** >> à envoyer.

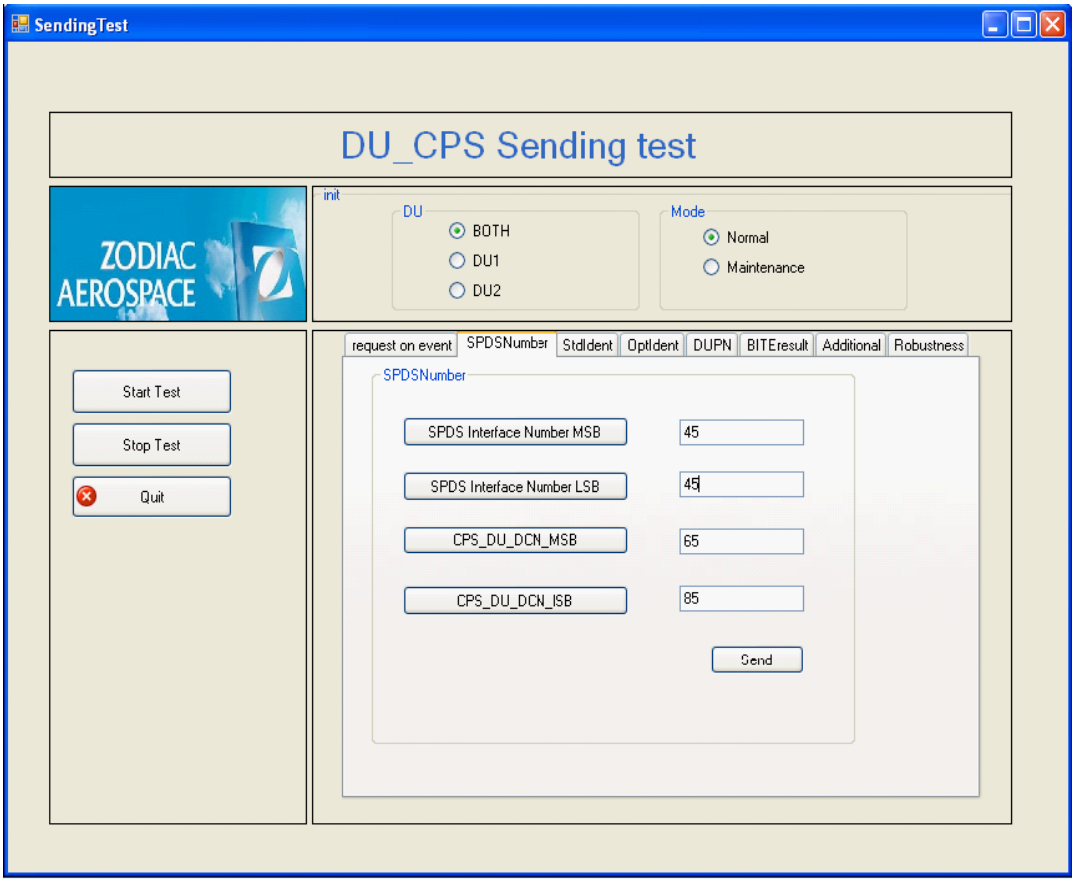

Figure 27: mise à jour des données de la trame DUz_All_SPDSNumber_CANw

Cette trame << **DUz_All_SPDSNumber_CANw** >> doit être envoyée par la DUz à tous les CPS pour fournir le SPDS Interface Number afin de vérifier la compatibilité de téléchargement entre eux.

✈ Onglet « StdIdent »

Des champs vides, qui vous permettent de saisir la trame<< **DUz_All_StdIdent_CANw** >> à envoyer, présenté par la figure en dessous :

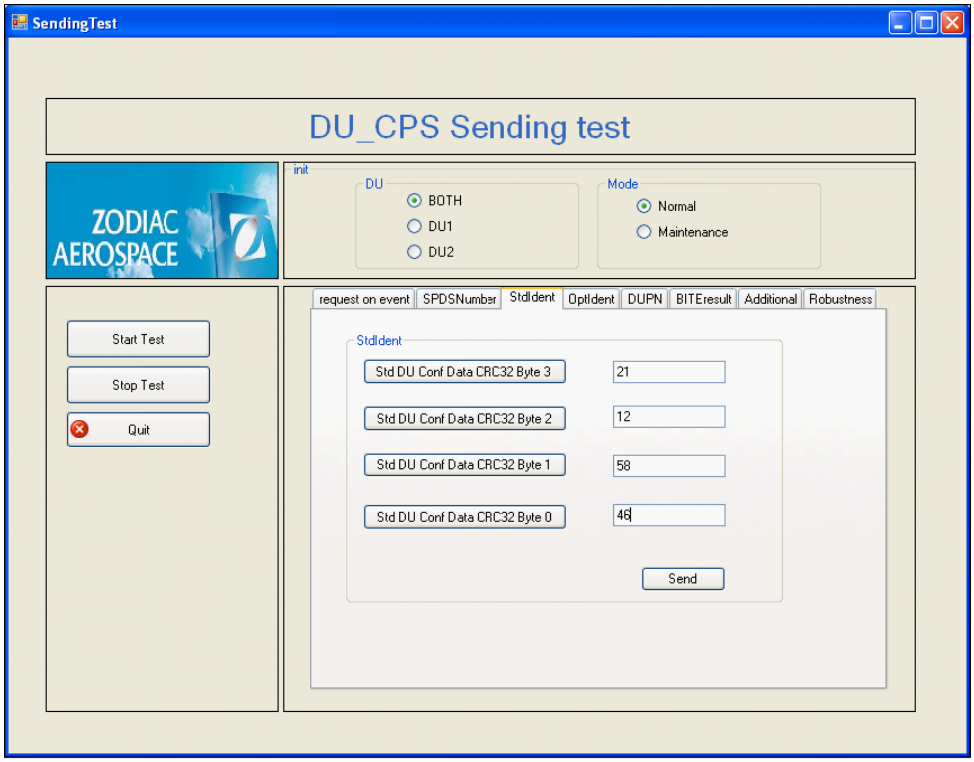

Figure 28: mise à jour des données de la trame DUz_All_StdIdent_CANw

Cette trame << **DUz_All_StdIdent_CANw** >> doit être envoyée par la DUz à tous les CPS pour l'identification de la configuration standard des chargements de données

✈ Onglet « OptIdent »

Des champs vides, qui vous permettent de saisir la trame << **DUz_All_OptIdent _CANw** >> à envoyer.

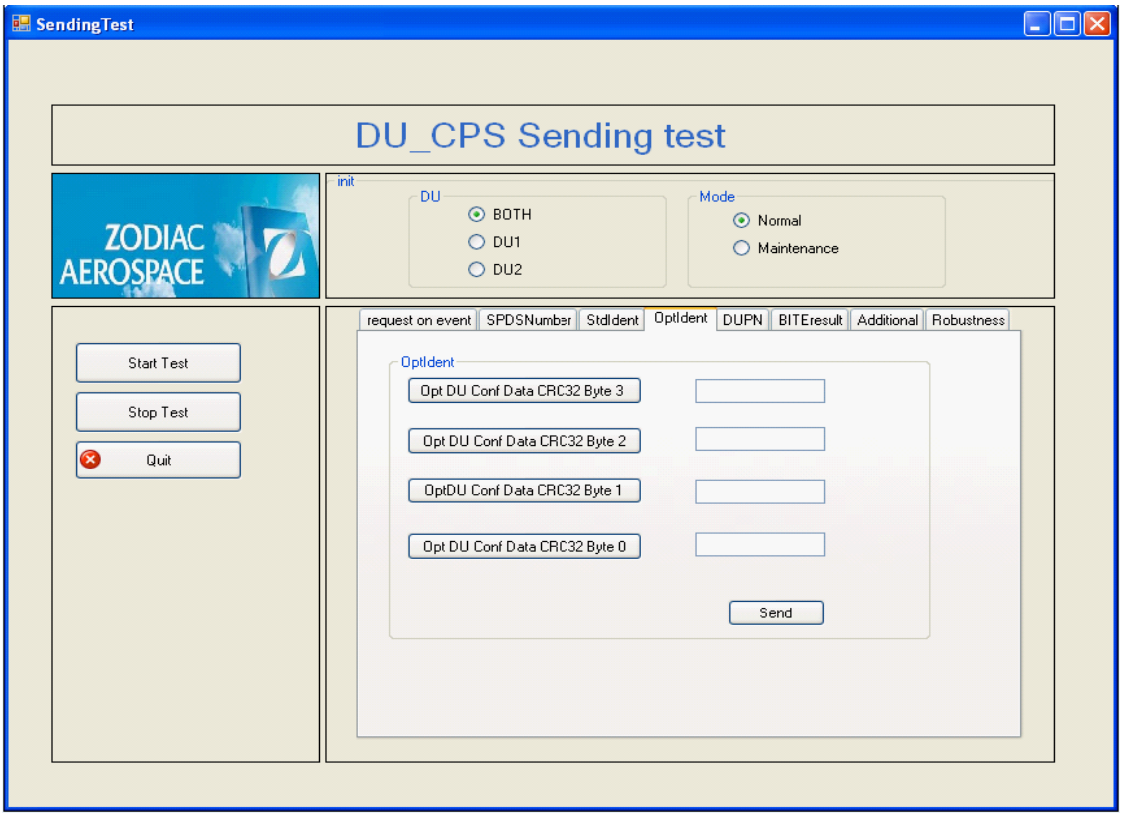

Figure 29 : mise à jour des données de la trame DUz_All_OptIdent_CANw

Cette trame << **DUz_All_OptIdent_CANw** >> doit être envoyée par la DUz à tous les CPS pour l'identification de la configuration optionnelle des chargements de données.

✈ Onglet « DUPN »

Des champs vides, qui vous permettent de saisir la trame <<**DUz_All_ DUPN _CANw** >> à envoyer.

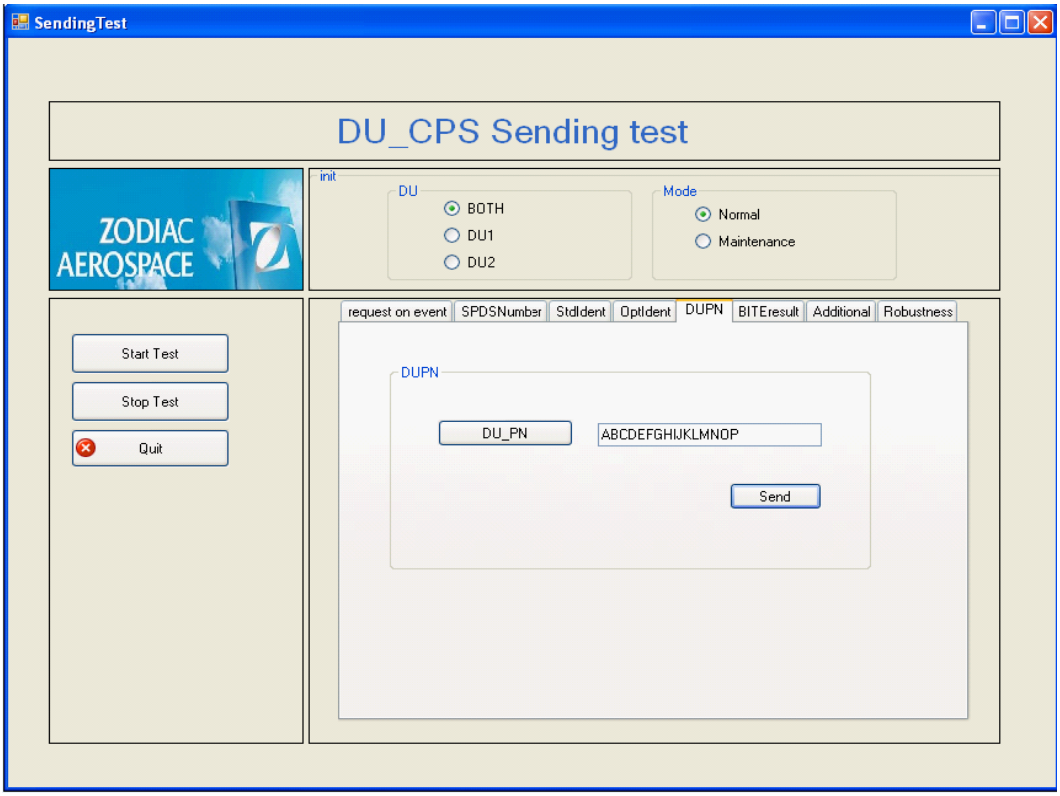

Figure 30: mise à jour des données de la trame DUz_All_DUPN_CANw

Cette trame << **DUz_All_DUPN_CANw** >> doit être envoyée par la DUz à tous les CPS pour fournir le numéro de DU .

✈ Onglet «BITEresult »

Des champs vides, qui vous permettent de saisir la trame << **DUz_All_BITEresult _CANw** >> à envoyer.

La page qui s'affiche permet d'activer/désactiver les statuts selon le choix de l'utilisateur.

Le premier click est pour la l'activation du statut (mise à 1), le deuxième désactive le statut (mise à0).

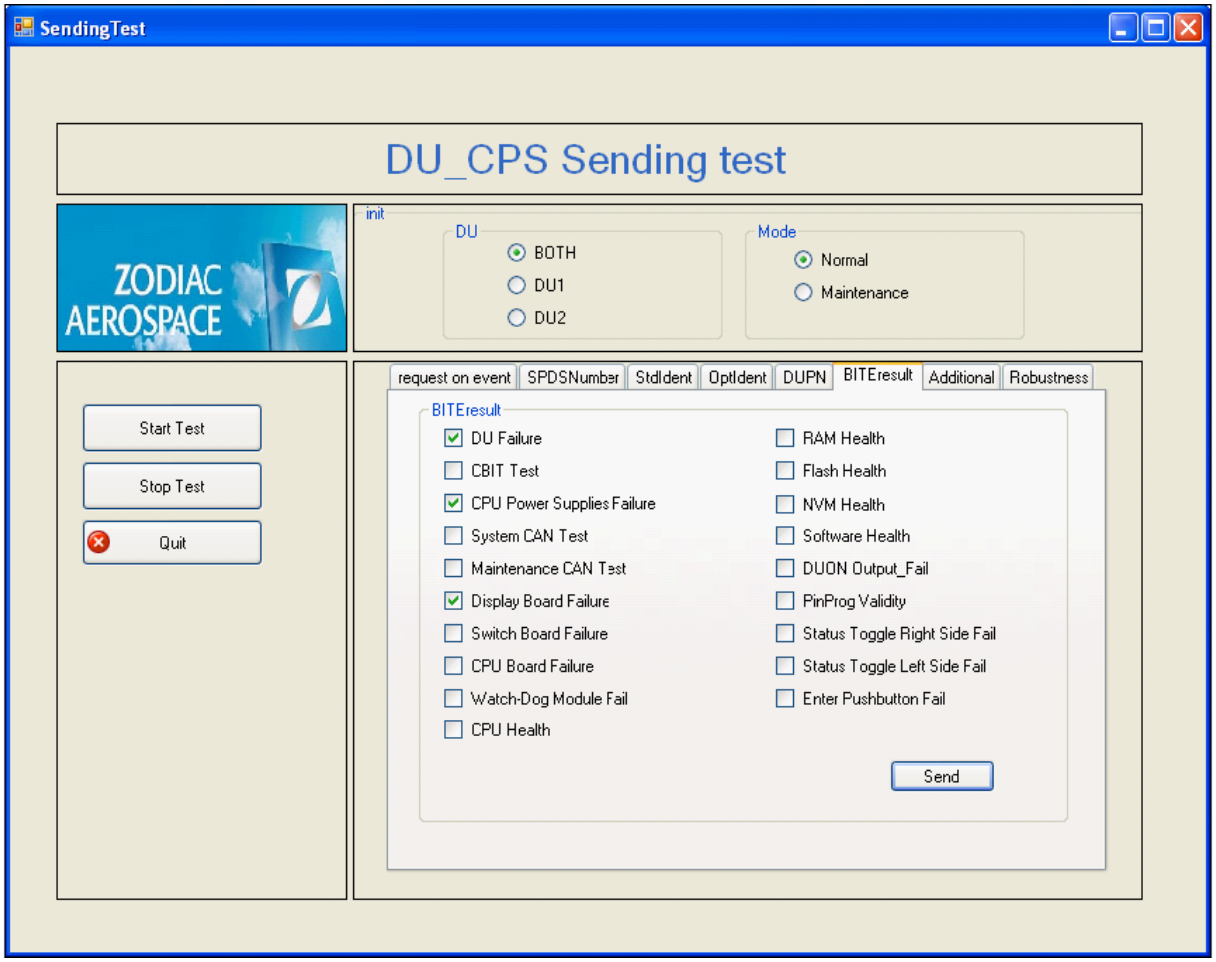

Figure 31: mise à jour des données la trame DUz_All_BITEresult_CANw

Cette trame << **DUz_All_BITEResult_CANw** >> doit être envoyée par la DUz à tous les CPS pour fournir les résultat de BITE tests (PBIT, CBIT, IBIT)

✈ Onglet « Additionnal »

Des champs vides, qui vous permettent de saisir la trame << **DUz_All_Additionnal_CANw** >> à envoyer.

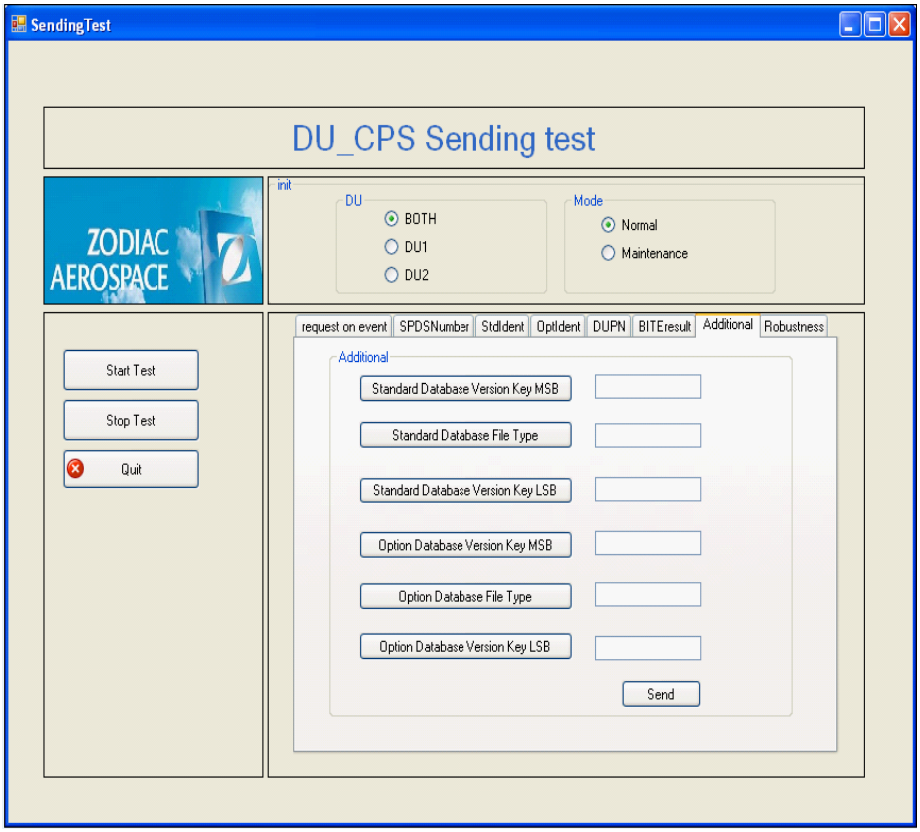

Figure 32: mise à jour des données de la trame DUz_All_Additional_CANw

Cette trame << **DUz_All_Additionnal_CANw** >> doit être envoyée par la DUz à tous les CPS pour fournir le type et la version de la base donnée standard et optionnel.

✈ Onglet « Robustnesse »

Cet onglet « **Robustness** » permet de faire 3 tests de robustesse, comme suivant : (voir figure 33)

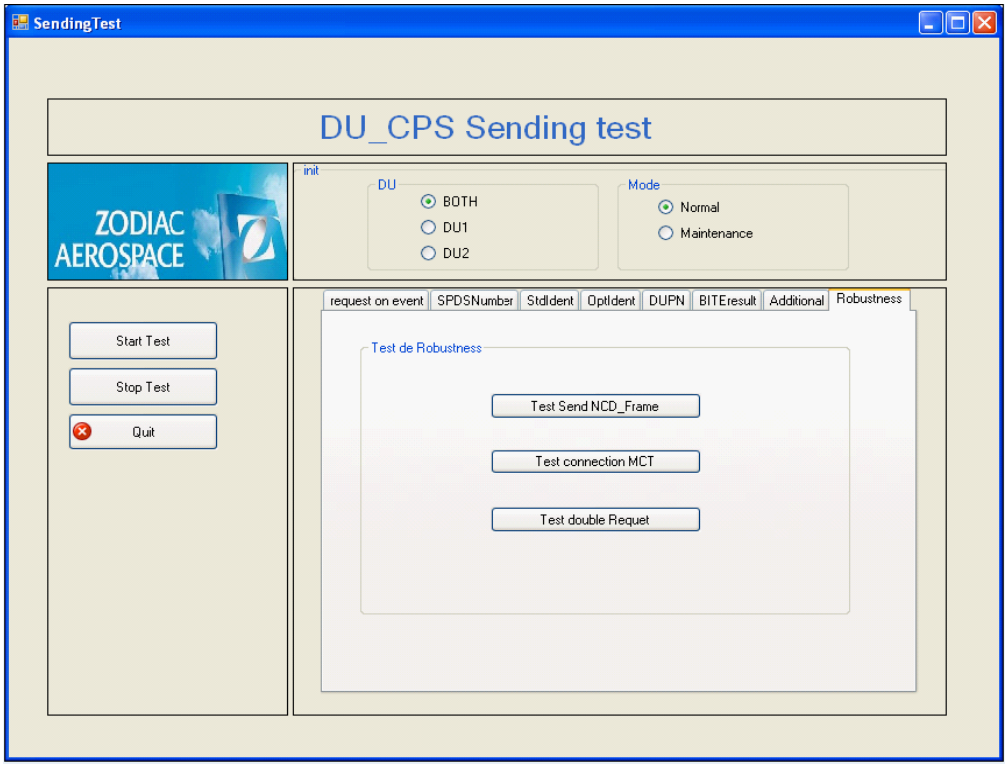

Figure 33:test de Robustesse

- Test par envoi des trames de données invalides dites NCD (No Computed Data) : ça permet de vérifier si le calculateur rejette ces trames.
- Test de la double requête : ce test envoie 2 requêtes successives séparées en moins de 1s ; le calculateur doit respecter la priorité en ne considérant que la première requête arrivée.
- Test de la requête de connexion d'un ordinateur de maintenance : à la réception d'une trame de demande de connexion avec le MTC le calculateur doit adopter le mode maintenance et donc lire sur le Bus Maintenance.

Mode de réception

Le deuxième niveau de simulation est caractérisé par sa fonction principale qui est l'affichage des valeurs des différents signaux reçus.

L'interface Graphique permet de visualiser la totalité des valeurs des différents signaux reçus.

En revenant au menu principal et en choisissant le test de récupération :

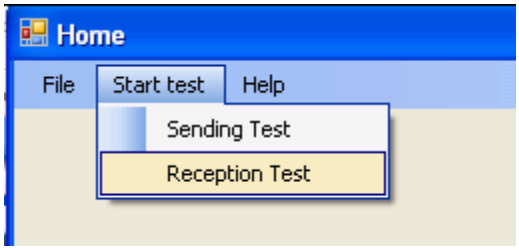

Figure 34 : choix de test de réception

Il nous renvoie vers la page **Réception trame** de notre application qui est de la forme suivante :

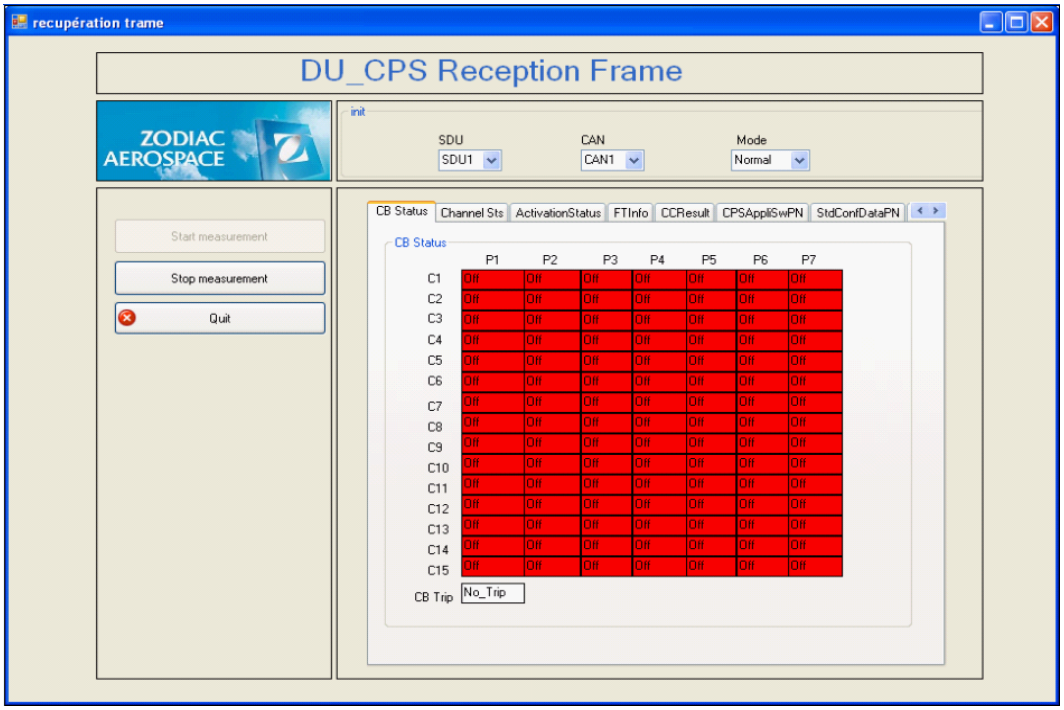

Figure 35:Mode de réception

- Avant d'établir la communication entre le simulateur et le calculateur afin de récupérer les trames renvoyées par la carte CPS, un premier paramétrage est mis en place. Il permet le choix de SDU (l'utilisateur spécifié de quel SDU il va récupérer les trames : SDU1, SDU2, SDU3, SDU4), choix de mode (Mode normal ou maintenance), choix de Bus (CAN1, CAN2).

- Une fois que l'utilisateur a fait ce choix, il peut démarrer le test de récupération (Start)

- les trames provenant de SDUy, sont transmises sur les deux bus CAN.

- Le simulateur permet d'avoir la main sur les différents signaux qui forment ces trames.

- En appuyant sur les Onglets «CBstatus », « ChannelStatus », « ActivationStatus », « FTInfo»,«CCResult»,« CPSAppliSwPN »,« StdConfDataPN» , « OptConfDataPN»et «BroadCastingX» l'utilisateur peut saisir les données à envoyer pour chaque trame

✈ Onglet «CBstatus »

Cette trame << **SDUxCy_All_CBStatusX_CANw** >> doit être envoyée par la carte Cy de SDUx à tous les DUz pour fournir les statuts des interrupteurs CB.

- CB status
  - 0: Off
  - 1: On
  - 2: Lock
  - 3: Spare

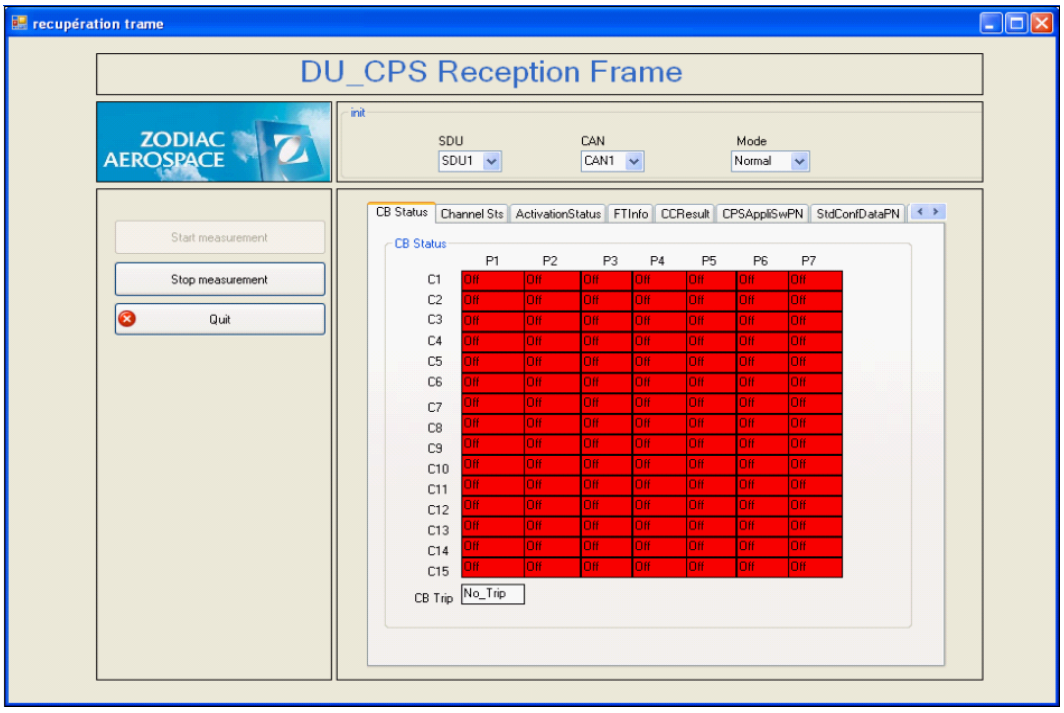

Figure 36:réception de la trame SDUz_All_CBstatus_CANw

✈ Onglet «ChannelStatus »

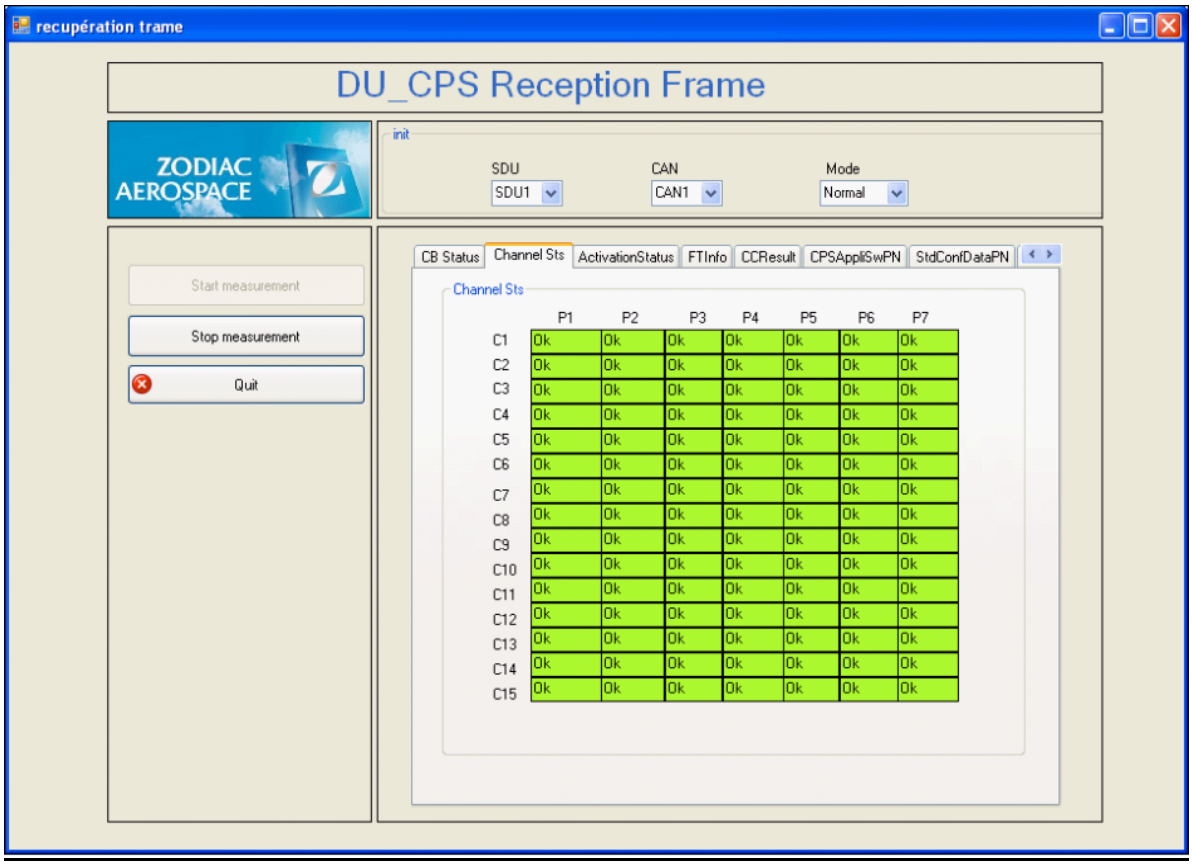

Figure 37: réception de la trame SDUz_All_ChannelStatusX_CANw

Cette trame << **SDUxCy_All_ChannelStatusX_CANw** >> doit être envoyée par la carte Cy de SDUx à tous les DUz pour fournir les statuts des Channels des cartes SSPC

- SSPC Channel Status
  - 0: Ok
  - 1: Trip
  - 2: Fail
  - 3: Not Avail

✈ Onglet «ActivationStatus»

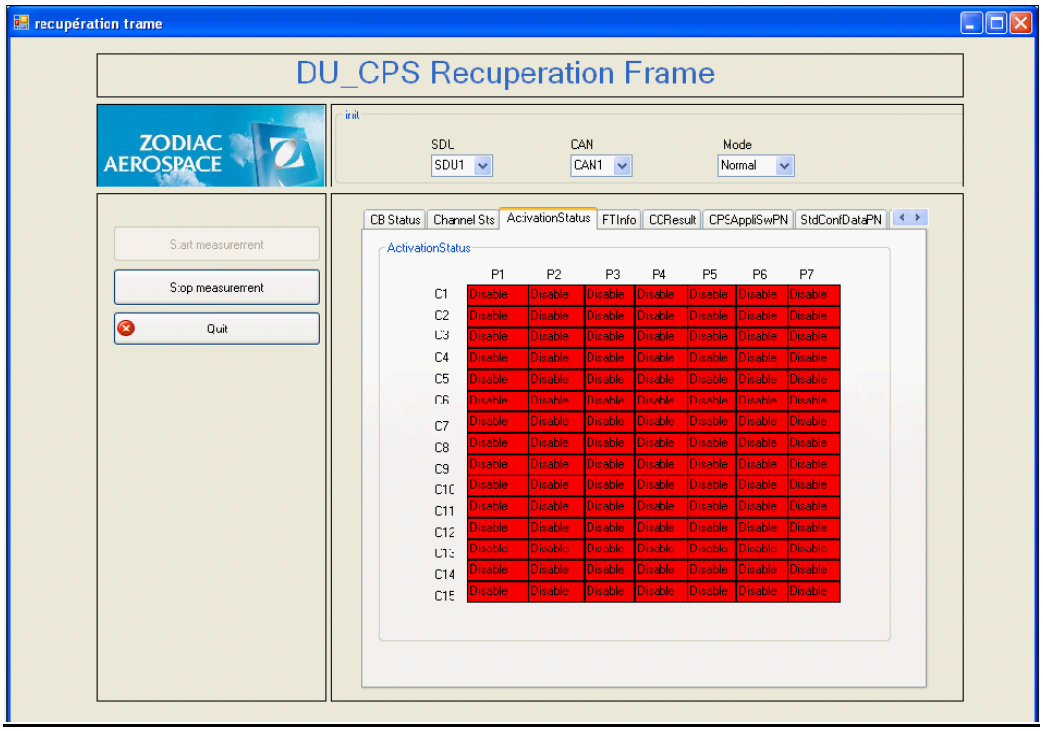

Figure 38: réception de la trame SDUz_All_ActivationStatusX_CANw

Cette trame << **SDUxCy_All_ActivationStatusX_CANw** >> doit être envoyée par la carte Cy de SDUx à tous les DUz pour fournir les statuts d'activation des Channel des cartes SSPC :

- PxCy Activation Staus
  - 0: Disable
  - 1: Enable

✈ Onglet «FTInfo »

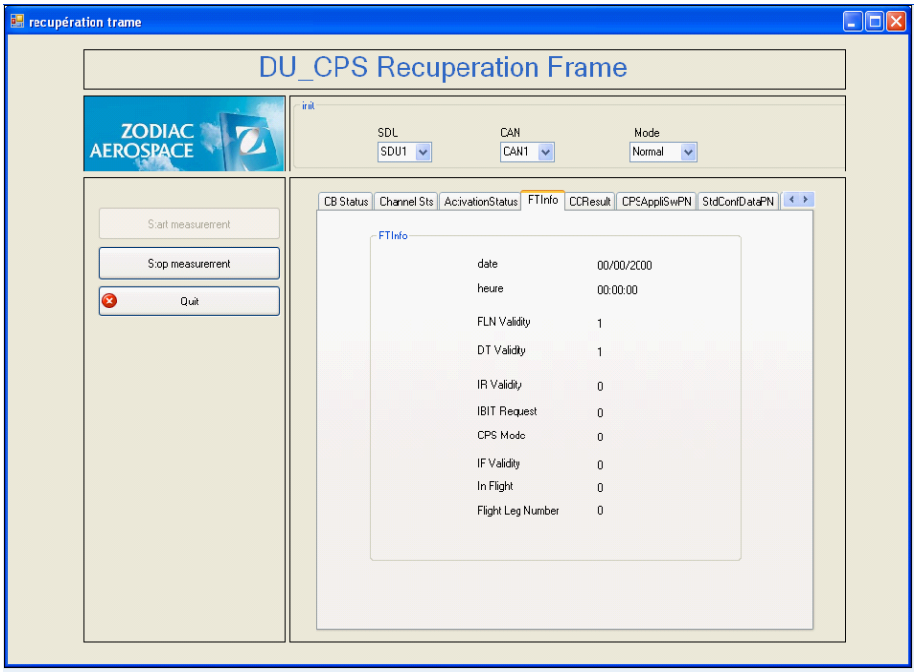

Figure 39: réception de la trame SDUz_All_FTinfo_CANw

Cette trame << **SDUxCy_All_FTinfo_CANw** >> doit être envoyée par la carte Cy de SDUx à tous les DUz pour de fournir des informations sur le vol.

✈ Onglet «CCResult »

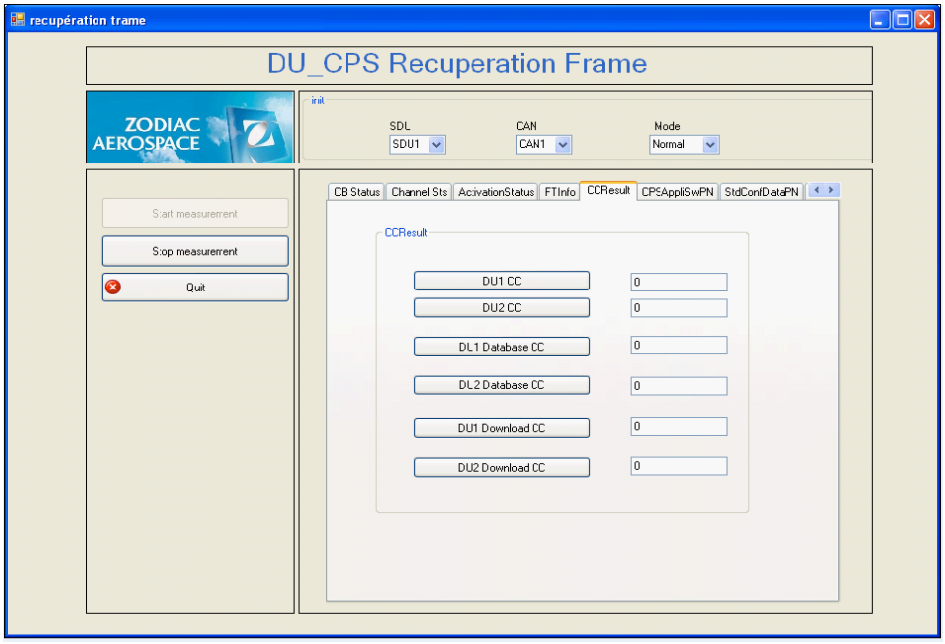

Figure 40: réception de la trame SDUz_All_CCResult_CANw

Cette trame << **SDUxCy_All_CCResult_CANw** >> doit être envoyée par la carte Cy de SDUx à tous les DUz pour de fournir le résultat de vérification de compatibilité entre CPS et DU

✈ Onglet «CPSAppliSwPN »

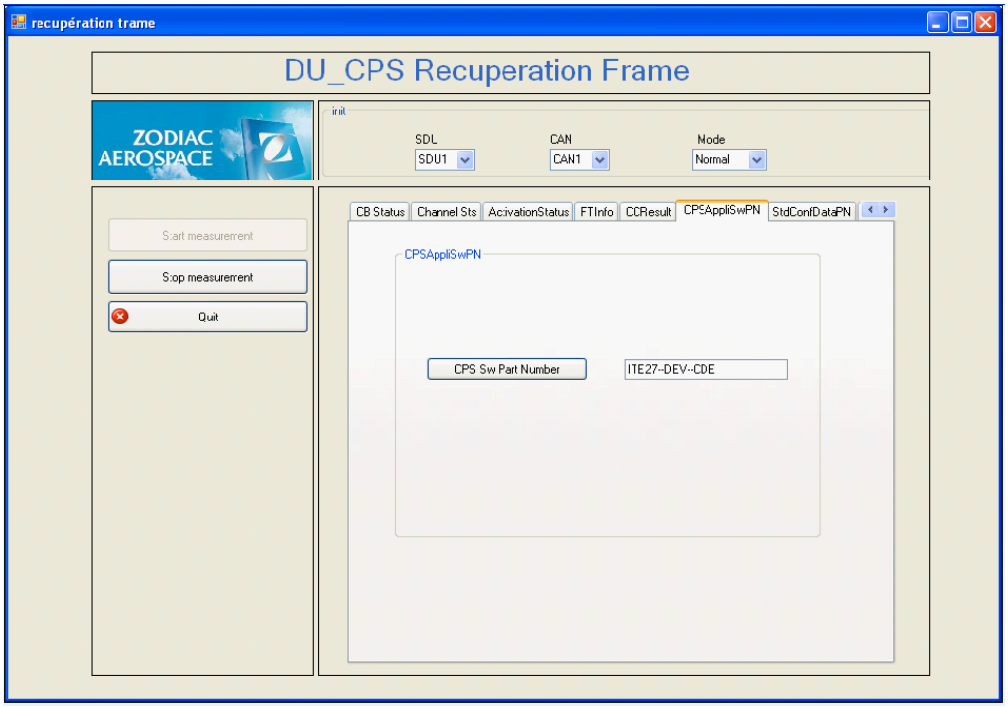

Figure 41: réception de la trame SDUz_All_CPSAppliSWPN_CANw

Cette trame << **SDUxCy_All_CPSAppliSwPN_CANw** >> doit être envoyée par la carte Cy de SDUx à tous les DUz pour fournir le numéro du logiciel applicatif

✈ Onglet «StdConfDataPN »

Cette trame << **SDUxCy_All_StdConfDataPN_CANw** >> doit être envoyée par la carte Cy de SDUx à tous les DUz pour fournir le numéro de la configuration standard de la base de données

✈ Onglet «OptConfDataPN »

Cette trame << **SDUxCy_All_OptConfDataPN_CANw** >> doit être envoyée par la carte Cy de SDUx à tous les DUz pour fournir le numéro de la configuration optionnel de la base de données

✈ Onglet «BroadCasting1 »

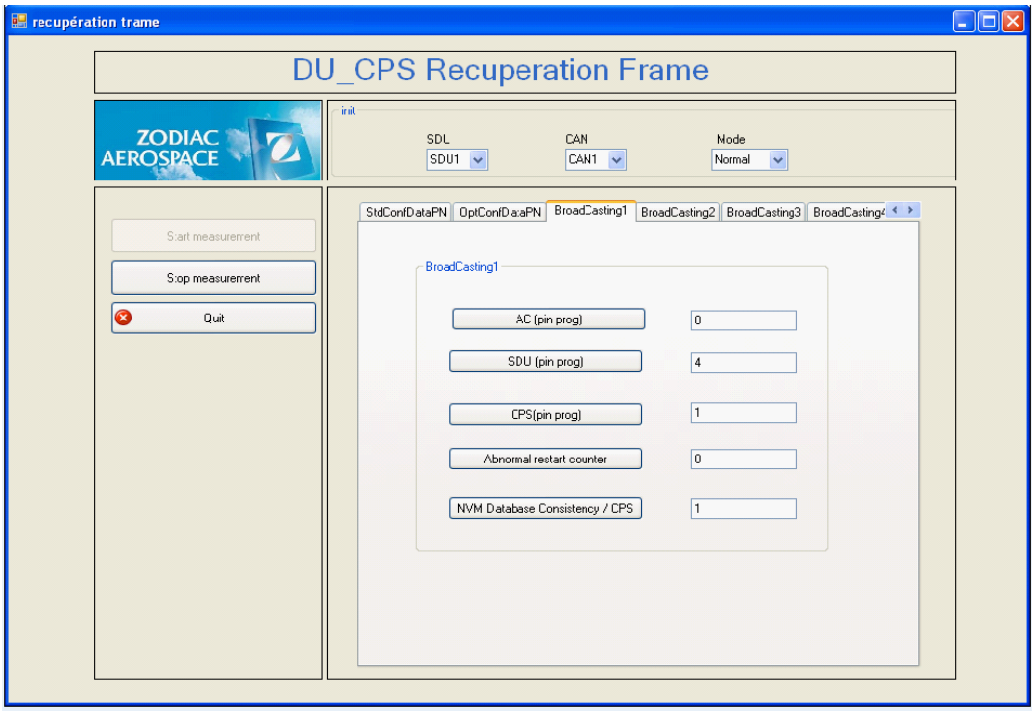

Figure 42 : réception de la trame SDUz_All_ BroadCasting1_CANw

Cette trame << **SDUxCy_All_ BroadCasting1_CANw** >> doit être envoyée par la carte Cy de SDUx à tous les DUz pour de fournir des variables internes du logiciel CPS.

✈ Onglet «BroadCasting2»

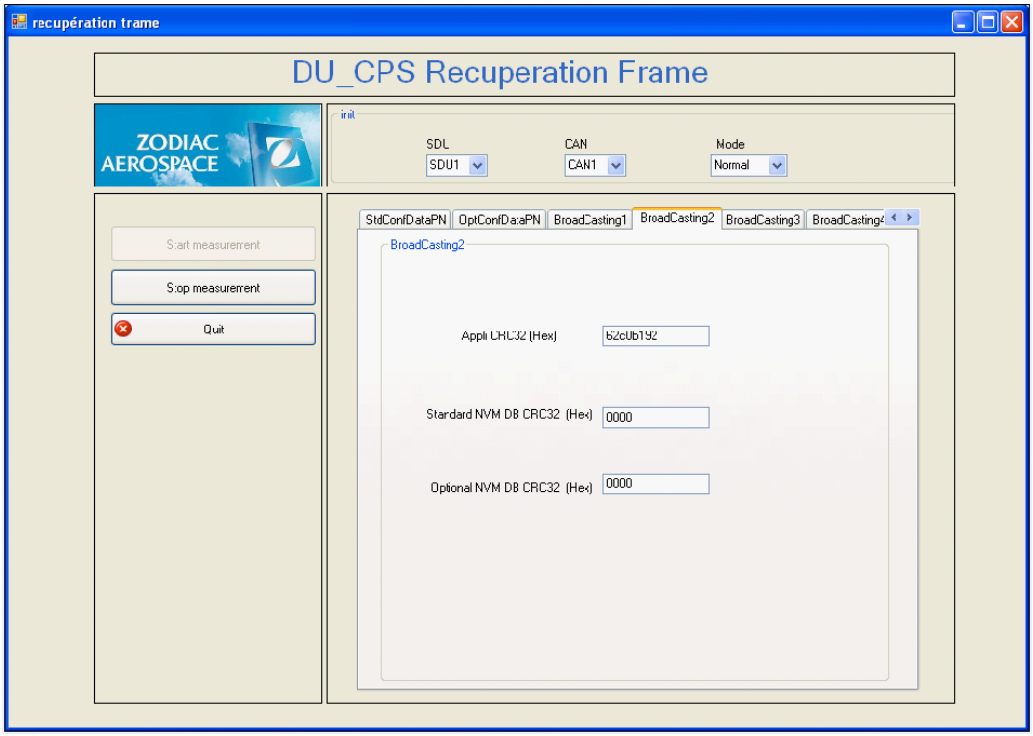

Figure 43: réception de la trame SDUz_All_BroadCasting2_CANw

Cette trame << **SDUxCy_All_ BroadCasting2_CANw** >> doit être envoyée par la carte Cy de SDUx à tous les DUz pour fournir des variables internes du logiciel CPS

✈ Onglet «BroadCasting3 »

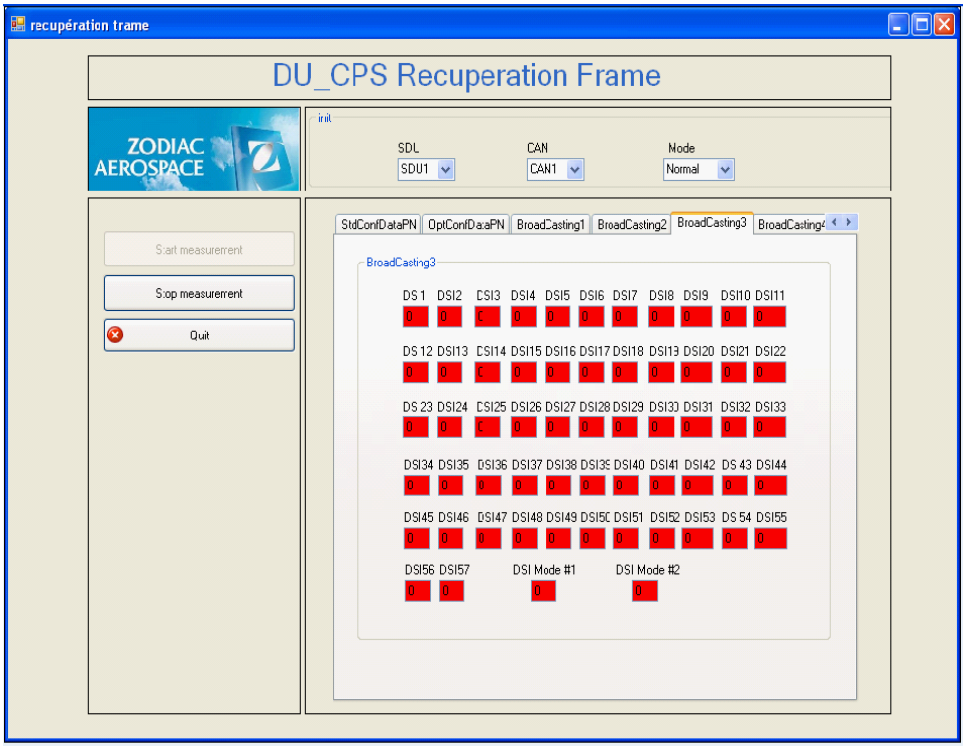

Figure 44: réception de la trame SDUz_All_BroadCasting3_CANw

Cette trame << **SDUxCy_All_ BroadCasting3_CANw** >> doit être envoyée par la carte Cy de SDUx à tous les DUz pour de fournir des variables internes du logiciel CPS

✈ Onglet «BroadCasting4»

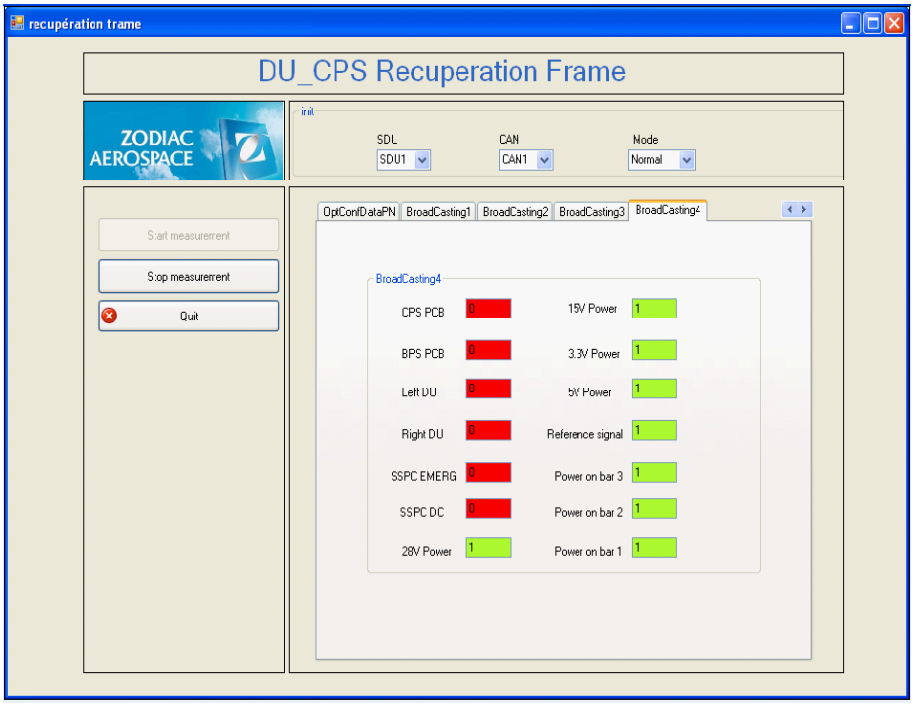

Figure 45 : réception de la trame SDUz_All_BroadCasting4_CANw

Cette trame << **SDUxCy_All_ BroadCasting4_CANw** >> doit être envoyée par la carte Cy de SDUx à tous les DUz pour de fournir des variables internes du logiciel CPS

CONCLUSION

Dans cette partie du rapport, on a présenté les ressources matérielles qui ont été utilisées lors de la réalisation du projet. Ainsi le fonctionnement nominal du simulateur.

Conclusion générale

Ce travail a été réalisé sur 3 phases:

- Une première phase de documentation et étude théorique ; elle permet en premier lieu de découvrir le métier du développement logiciel et sa réglementation. L'étude théorique porte sur les caractéristiques du protocole CAN, ainsi les différents outils logiciels.

- La deuxième phase consiste à effectuer une analyse du cahier des charges du projet et à développer les différents besoins en tâches bien précises.

- Dans la troisième phase, on a entamé la partie d'implémentation du simulateur. le développement, la configuration du simulateur, et le développement de l'interface graphique.

Après ces quatre mois de stage, on a pu accomplir le simulateur et il est maintenant prés à l'utilisation

A certaines étapes de la réalisation où on a été confronté à des blocages, au niveau du passage de la couche VB vers la couche CAPL, était de passer des données en argument de la fonction VB vers la fonction CAPL puis vers le Bus CAN.

Le simulateur représente un moyen fiable et à faible coût de validation des différentes fonctions du logiciel embarqué, cependant, le simulateur n'est pas générique pour tester tous les logiciel embarqués, il est valable pour un seul logiciel

En termes de perspectives le simulateur peut être amélioré au fur et à mesure du développement du logiciel.il peut être modifié et amélioré selon le logiciel embarqué à tester. Et Comme tous logiciels il reste toujours des améliorations à faire, pour donner plus de performance.

Ce stage répond à un réel besoin dans le domaine de la sureté de fonctionnement des équipements aéronautique.

Ce projet de fin d'études m'a permis de gérer des obstacles multiples, de prendre des initiatives, d'avoir une vision plus mûre d'un vrai travail responsable, et ainsi, de m'intégrer complètement au monde de l'entreprise.

Il m'a permis de découvrir de près le métier du développement logiciel dont je voudrais faire mon métier. Il m'a également permis de travailler dans un environnement professionnel, et d'établir des relations avec une équipe dynamique.

Université Sidi Mohammed Ben Abdellah
Faculté Des Sciences et Techniques Fès
Département de Génie Electrique

ANNEXE

Annexe: Bus CAN

1. Bus CAN :

Le bus CAN a été conçu par BOSCH en 1983.

Bien plus qu'un bus au sens électrique, le (bus/réseau de terrain) CAN est un réseau à part entière respectant le modèle d'interconnexion des systèmes ouverts OSI de l'ISO. C'est un réseau de terrain aussi car il doit fonctionner dans un environnement limité et sévère comme une usine, un atelier, une voiture, un avion…

Il commence à être utilisé en aéronautique. On le trouve sur :

→ Monde automobile : de nombreux véhicules modernes (Daimler-Benz, Audi, BMW, Renault, Peugeot, Saab, Volkswagen, Volvo, …)

→ Des avions en cours de développement : Boeing 787, Airbus A400M, Gulfstream G650

Le protocole CAN (Control Area Network) est un protocole de communication série qui supporte des systèmes temps réel avec un haut niveau de fiabilité. Ses domaines d'application s'étendent des réseaux moyens débits aux réseaux de multiplexages faibles coûts. Il est avant tout à classer dans la catégorie des réseaux de terrain utilisé dans l'industrie pour remplacer la boucle analogique 20mA.

La structure du protocole du bus CAN possède implicitement les principales propriétés suivantes :

- ➔ Hiérarchisation des messages.
- ➔ Garantie des temps de latence.
- ➔ Souplesse de configuration.
- ➔ Réception de multiples sources avec synchronisation temporelle.
- ➔ Fonctionnement multi maître.
- ➔ Détections et signalisations d'erreurs.
- ➔ Retransmission automatique des messages altérés dès que le bus est de nouveau au repos.
- ➔ Distinction d'erreurs : d'ordre temporaire ou de non-fonctionnalité permanente au niveau d'un nœud.
- ➔ Déconnexion automatique des nœuds défectueux.

2. Principe de communication:

A. Bit CAN :

a. *Codage bit:*

La méthode retenue pour le codage bit du flot des trames CAN est le codage de type No Return to Zero (NRZ).

b. *Méthode de Bit Stuffing :*

Cette méthode consiste à introduire après 5 bits de même polarité, un bit de polarité contraire. Bien évidemment, le récepteur CAN, parfaitement courant de cette technique de bourrage, procède au destuffage lors de la réception en retirant ces bits de remplissage n'ayant servi principalement qu'à fiabiliser le transport des données.

B. **Transmission des messages :**

L'information sur le bus est envoyée sous la forme de messages au format fixe. Quand le bus est libre, n'importe quel noeud peut commencer à envoyer un message.

a. *Protocoles 2.0A et 2.0B* :

Le protocole CAN 2.0 comporte deux sous-spécifications qui diffèrent uniquement au niveau de la longueur de l'ID. La version 2.0A définit des ID de 11 bits et la version 2.0B des ID de 29 bits.

b. *Types de messages :*

Le transfert des messages s'effectue à l'aide de quatre types de trames spécifiques et d'un intervalle de temps les séparent.
Il s'agit des :

- trames de données : sont utilisées pour transporter des données sur le bus.
- trames de requête : Dans le bus CAN, une station peut demander une information qu'elle souhaite en émettant une trame de requête avec le même identifiant que celui de l'objet qu'elle souhaite. Elle est constituée de la même manière qu'une trame de données sauf que le champ de données est vide.
- trames d'erreur : elles sont transmises par un noeud ayant détecté une erreur.
- trames de surcharge **:** elles sont utilisées pour produire un délai entre deux Data ou Remote Frames successives.

C. **Format des trames de données :**

Elle se décompose en 7 champs différents comme suit :

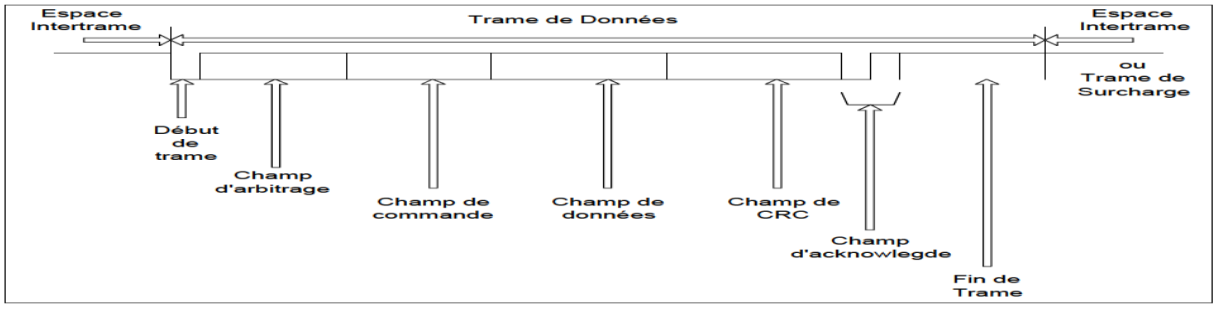

Figure 46: Trame de donnée de CAN

➔ **Début de trames (SOF) :**

N'est effectif que si le bus était précédemment au repos, tous les nœuds du réseau doivent se resynchroniser sur le bit SOF.

➔ **Champ d' arbitrage:**

Composé de 11 bit pour l'identificateur et d'un bit RTR (Remote Transmission Request) :

- ✓ Identificateur: les bits sont transmis dans l'ordre de ID_10 à ID_0, la priorité est d'autant plus élevée que la valeur de l'identificateur est faible.
- ✓ RTR: définit la nature de la trame. Il est dominant pour dataframe et récessif pour une trame de requête (remote frame).

➔ **Champ de contrôle:**

2 bits dominants sont en réserve et 4 bits indiquant le nombre d'octets contenant dans le champ de données.

➔ **Champ de données:**

Il a une longueur qui peut varier de 0 à 64 bits. Il est vide dans le cas d'une trame de requête.

➔ **Champ de CRC:**

Il se compose de séquence du CRC, qui assure la validité du message transmis, suivie d'un délimiteur de CRC.

➔ **Champ d'acquittement:**

Il est composé de 2 bit:
ACK slot

ACK Délimiter transmis à l'état récessif.

Toutes les stations ayant validé la séquence CRC envoient un état dominant dans le bit ACK Slot. Ainsi, le noeud émetteur sait que son message a été reçu au moins par une station.

➔ **Champ EOF:**

C'est une séquence de 7 bits récessifs qui ne subit pas la règle de bit ***Stuffing***

3. Architecture générale du protocole CAN

➔ **Aspect physique**

La philosophie du bus CAN impose que chaque nœud soit autonome. Ainsi trouve-t-on deux architectures liées à deux familles de circuits :

--Les contrôleurs indépendants
--Les microcontrôleurs intégrant le contrôleur du bus CAN

La première famille nécessite une unité de traitement de type microcontrôleur ou microprocesseur pour programmer et dialoguer avec le contrôleur, voir la représentation ci-dessous :

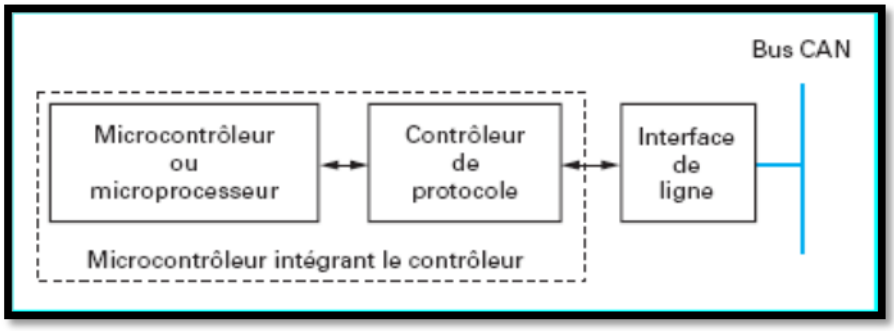

Figure 47:Architecture d'un nœud CAN

On peut subdiviser les composants CAN selon les 3 grandes classes de fonctionnalités suivantes :

- ✈ Les gestionnaires de protocole
- ✈ Les microcontrôleurs ayant à bord des gestionnaires de protocole
- ✈ Les interfaces de ligne

L'ensemble des nœuds d'un même réseau CAN est structuré suivant le couplage physique expliqué dans le schéma de la figure ci-dessus :

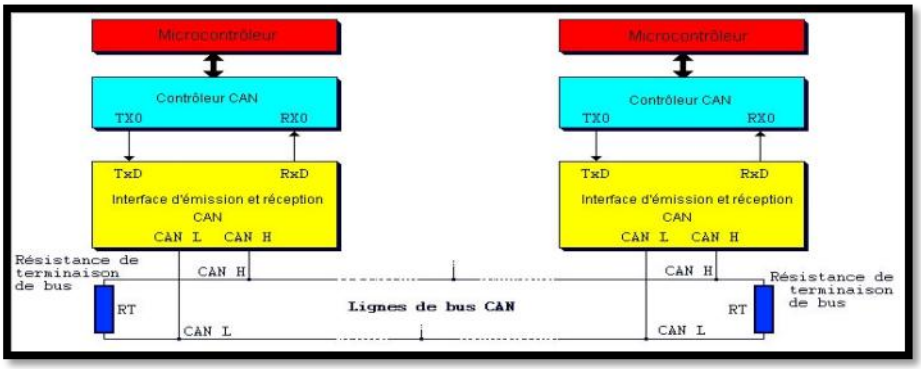

Figure 48: Constitution d'un réseau CAN

Le câble est constitué par deux paires de fils torsadés ce qui permet de limiter les parasites auxquels un bus de terrain est généralement soumis. Une première paire transporte l'alimentation électrique qui permet d'alimenter directement les appareils de faible consommation:

- CAN0V (masse)
- CAN+V (+12V)

La deuxième paire supporte les signaux de données :

- CANL (CAN LOW)
- CANH (CAN HIGH)

Liste des figures:

Liste des Tableaux

Bibliographie :

Documents ZAM :

- ✓ **SPE100241_SPDS_IDD_CAN-1D2**
  Edition: *Février 2010*
- ✓ **broadcast framemaping next 2**
  Edition: *2010*

✓ **SPE100240_SPDS_Download_Specification-1**
Edition: *Avril 2010*

Web-graphie

[4] : http://fr.wikipedia.org/wiki/DO-178B
[5] : http:// www.developpez.net
[6] : http://www.vbfrance.com/