Tuteur : Sébastien RICHE
Encadrant : Isabelle FERRANÉ

Développement, amélioration et intégration d'outils de génération de code

pour une plateforme de prototypage de fonctions de contrôle moteur

Mathieu SOUM

Stage du 3 novembre 2014 au 31 juillet 2015
Chez Aboard Engineering

Master 2 – Développement Logiciel / 2014 - 2015

Tuteur : Sébastien RICHE
Encadrant : Isabelle FERRANÉ

Développement, amélioration et intégration d'outils de génération de code

pour une plateforme de prototypage de fonctions de contrôle moteur

Mathieu SOUM

Stage du 3 novembre 2014 au 31 juillet 2015
Chez Aboard Engineering

Master 2 – Développement Logiciel / 2014 - 2015

Je souhaite remercier ici l'équipe pédagogique de la formation pour nous proposer une telle opportunité. Contrairement aux autres stages effectués lors de notre cursus, celui-ci s'étale sur toute une année et nous permet, non seulement de travailler sur un projet de longue haleine, mais également de montrer notre adaptation à une équipe sur le long terme.

Je tiens aussi à remercier Paul Martin pour m'avoir permis de réaliser ce stage chez Aboard Engineering ainsi que Vincent Huc et Sébastien Riche pour m'avoir épaulé tout au long de cette expérience. Ils ont su me faire une place et me faire sentir utile et considéré et pour cela je leur en suis reconnaissant.
J'aimerais également remercier toute l'équipe d'Aboard Engineering pour l'accueil qu'ils m'ont accordé et leur bonne humeur quotidienne qui fait du bien et qui rend encore plus agréable l'environnement de travail.

Enfin, je souhaite remercier aussi ceux qui m'ont accompagné durant ce stage, certes plus moralement que techniquement, je veux parler de Clément, Ophélie, (petit) Clément, Antoine et d'autres que j'oublie sûrement. Ces gens l'ombre sans qui les journées n'auraient pas le même parfum.

Sommaire

Chapitre 1

Introduction

Contexte du stage

Ce stage s'inscrit dans le cadre de mon Master 2 Développement Logiciel à l'Unversité Paul Sabatier de Toulouse. Cette formation a la particularité d'offrir un stage en alternance dès le début de l'année universitaire. C'est une très bonne expérience qui nous tient en haleine tout au long de notre dernière année d'étude et nous permet – pour la première fois dans notre cursus – de réaliser des projets d'envergure en entreprise et de pouvoir nous y impliquer pleinement sur une durée plus longue que lors des expériences de stage précédentes.

Ce stage s'est donc déroulé en alternance au rythme de 3 jours par semaine entreprise (Lundi, Mardi, Mercredi) et 2 jours à l'université (Jeudi, Vendredi) sur une durée totale de 9 mois (3 Novembre 2014 – 31 Juillet 2015).

Structure du rapport

Ce rapport va rendre compte de mon expérience de stage. Il sera découpé selon 3 parties principales :

Le contexte dans lequel j'ai évolué durant les 9 mois de stage. Dans ce chapitre, je fera une présentation de l'entreprise qui m'a accueillie ainsi que l'environnement de travail dans lequel j'ai évolué.

Le sujet sur lequel j'ai travaillé. Je ferai une description en détail de la problématique realtive à mon stage et du travail à réaliser.

Le travail réalisé durant ce stage. Ce chapitre abordera les détails technique de mon stage. J'y développerai les différentes tâches que j'ai réalisé et en quoi elles aboutissent à une solution viable à la probl'ématique.

Conseils de lecture

Ce document est à destination de mon maître de stage, mon encadrant universitaire ainsi que le jury de la soutenance que l'aura à disposition. Pour faciliter sa lecture, voici quelques précisions quant aux détails typographiques utilisés dans ce document.

Les citations relatives à du code source telles que les noms de classes ou les éléments du système de fichier seront écrit avec une `police à chasse fixe`.

Un glossaire à la fin du document répertorie certains termes techniques qui nécessiteraient une définition plus précise. les mots apparaissant dans ce glossaire seront écris *en écriture penchée* dans le reste du document.

Chapitre 2

Contexte

2.1 L'entreprise : Aboard Engineering

Aboard Engineering est un bureau d'études en automatique, électronique et informatique industrielle. Composé d'experts en contrôle et instrumentation de systèmes embarqués temps réel, ses ingénieurs développent des solutions pour des applications civiles et militaires. De la R&D à la série, Aboard Engineering travail dans le domaine des transports (automobile, aéronotique, marin), de l'off-road (agriculture, engins de chantier, ...) et de l'industrie. Les figures 2.1 et 2.2 sont tirées du site web d'Aboard Engineering et récapitulent les domaines et métiers de la société.

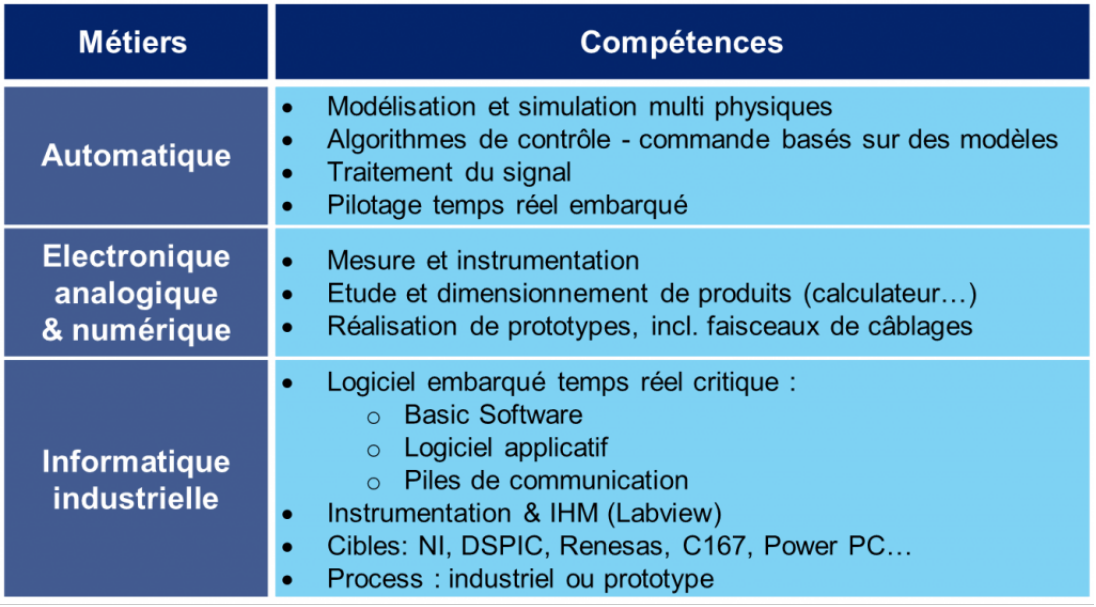
Domaines	Connaissances & expérience
Moteurs thermiques (essence , Diesel)	• Combustion • Stratégies de contrôle moteur, incl. dépollution • Technologie des capteurs et actionneurs • Problématiques de production série (diagnostic, adaptatif…) • Mise au point moteur / véhicule
Machines électriques	• Modélisation et simulation électronique de puissance et machine électrique • Commande vectorielle • Pilotage de machines électriques • Mise au point et calibration
Hybrides	• Contrôle & supervision d'énergie
Outils	• Développement d'outils génériques et spécifiques client • Instrumentation et essais pour tous les domaines

FIGURE 2.1 – Domaines – Source : Abord Engineering

2.2 La plateforme Orianne

Aboard Engineering fournie différents types de produits et services. Je ne vais ici développer que le cas de la plateforme *Orianne* sur laquelle j'ai travaillé.

Orianne est une plateforme de prototypage rapide de fonctions de contrôle moteurs. Elle permet de regrouper différents modules d'un calculateur moteurs (transmission, ABS, injection, couple moteur, etc.) afin de générer une unique application qui sera déployée dans un calculateur moteur. Pour cela, une interface graphique permet de configurer quelles fonctions sont à intégrer dans l'application (elles-mêmes configurables), les caractéristiques du moteur cible, la configuration du *RTOS* (définition des récurrence des tâches). Une fois l'application configurée, la plateforme va chercher les codes sources des modules sélectionnés, génère les fichiers sources manquant (non relatifs à des fonctions moteurs)



Métiers	Compétences
Automatique	• Modélisation et simulation multi physiques • Algorithmes de contrôle - commande basés sur des modèles • Traitement du signal • Pilotage temps réel embarqué
Electronique analogique & numérique	• Mesure et instrumentation • Etude et dimensionnement de produits (calculateur…) • Réalisation de prototypes, incl. faisceaux de câblages

Informatique industrielle	• Logiciel embarqué temps réel critique :
	  ○ Basic Software
	  ○ Logiciel applicatif
	  ○ Piles de communication
	• Instrumentation & IHM (Labview)
	• Cibles: NI, DSPIC, Renesas, C167, Power PC…
	• Process : industriel ou prototype

FIGURE 2.2 – Métiers et Compétences – Source : Abord Engineering

et effectue la compilation. La sortie produite est un fichier binaire représentant le code source compilé ainsi qu'un fichier « dictionnaire » associant chaque élément à l'adresse à laquelle il sera stocké dans la mémoire du calculateur. Ce fichier contient les adresses des constantes, des calibrations, des différentes fonctions appelées, etc. Les données non modifiées durant l'exécution de l'application. L'intérêt d'avoir des adresses fixes et connues pour ces éléments est qu'il est possible via un logiciel tiers de récupérer ces informations et de les modifiées directement dans la mémoire sans avoir à déployer une nouvelle application. Pour rappel, nous sommes dans un cadre R&D et beaucoup de tests sont effectués sur banc de test et le besoin d'adapter les calibrations pour obtenir le meilleur comportement est primordial.

La plateforme en elle-même est composée de plusieurs outils :

— Configuration du moteur cible
— Configuration des fonctions moteur
— Configuration du *RTOS*
— Générations des fichiers sources
— Compilation de l'application

2.3 L'environnement de travail //TODO

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Ut purus elit, vestibulum ut, placerat ac, adipiscing vitae, felis. Curabitur dictum gravida mauris. Nam arcu libero, nonummy eget, consectetuer id, vulputate a, magna. Donec vehicula augue eu neque. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Mauris ut leo. Cras viverra metus rhoncus sem. Nulla et lectus vestibulum urna fringilla ultrices. Phasellus eu tellus sit amet tortor gravida placerat. Integer sapien est, iaculis in, pretium quis, viverra ac, nunc. Praesent eget sem vel leo ultrices bibendum. Aenean faucibus. Morbi dolor nulla, malesuada eu, pulvinar at, mollis ac, nulla. Curabitur auctor semper nulla. Donec varius orci eget risus. Duis nibh mi, congue eu, accumsan eleifend, sagittis quis, diam. Duis eget orci sit amet orci dignissim rutrum.

Nam dui ligula, fringilla a, euismod sodales, sollicitudin vel, wisi. Morbi auctor lorem non justo. Nam lacus libero, pretium at, lobortis vitae, ultricies et, tellus. Donec aliquet, tortor sed accumsan bibendum, erat ligula aliquet magna, vitae ornare odio metus a mi. Morbi ac orci et nisl hendrerit mollis. Suspendisse ut massa. Cras nec ante. Pellentesque a nulla. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Aliquam tincidunt urna. Nulla ullamcorper vestibulum turpis. Pellentesque cursus luctus mauris.

Nulla malesuada porttitor diam. Donec felis erat, congue non, volutpat at, tincidunt tristique, libero. Vivamus viverra fermentum felis. Donec nonummy pellentesque ante. Phasellus adipiscing semper elit. Proin fermentum massa ac quam. Sed diam turpis, molestie vitae, placerat a, molestie nec, leo. Maecenas lacinia. Nam ipsum ligula, eleifend at, accumsan nec, suscipit a, ipsum. Morbi

blandit ligula feugiat magna. Nunc eleifend consequat lorem. Sed lacinia nulla vitae enim. Pellentesque tincidunt purus vel magna. Integer non enim. Praesent euismod nunc eu purus. Donec bibendum quam in tellus. Nullam cursus pulvinar lectus. Donec et mi. Nam vulputate metus eu enim. Vestibulum pellentesque felis eu massa.

Quisque ullamcorper placerat ipsum. Cras nibh. Morbi vel justo vitae lacus tincidunt ultrices. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. In hac habitasse platea dictumst. Integer tempus convallis augue. Etiam facilisis. Nunc elementum fermentum wisi. Aenean placerat. Ut imperdiet, enim sed gravida sollicitudin, felis odio placerat quam, ac pulvinar elit purus eget enim. Nunc vitae tortor. Proin tempus nibh sit amet nisl. Vivamus quis tortor vitae risus porta vehicula.

Fusce mauris. Vestibulum luctus nibh at lectus. Sed bibendum, nulla a faucibus semper, leo velit ultricies tellus, ac venenatis arcu wisi vel nisl. Vestibulum diam. Aliquam pellentesque, augue quis sagittis posuere, turpis lacus congue quam, in hendrerit risus eros eget felis. Maecenas eget erat in sapien mattis porttitor. Vestibulum porttitor. Nulla facilisi. Sed a turpis eu lacus commodo facilisis. Morbi fringilla, wisi in dignissim interdum, justo lectus sagittis dui, et vehicula libero dui cursus dui. Mauris tempor ligula sed lacus. Duis cursus enim ut augue. Cras ac magna. Cras nulla. Nulla egestas. Curabitur a leo. Quisque egestas wisi eget nunc. Nam feugiat lacus vel est. Curabitur consectetuer.

Suspendisse vel felis. Ut lorem lorem, interdum eu, tincidunt sit amet, laoreet vitae, arcu. Aenean faucibus pede eu ante. Praesent enim elit, rutrum at, molestie non, nonummy vel, nisl. Ut lectus eros, malesuada sit amet, fermentum eu, sodales cursus, magna. Donec eu purus. Quisque vehicula, urna sed ultricies auctor, pede lorem egestas dui, et convallis elit erat sed nulla. Donec luctus. Curabitur et nunc. Aliquam dolor odio, commodo pretium, ultricies non, pharetra in, velit. Integer arcu est, nonummy in, fermentum faucibus, egestas vel, odio.

Sed commodo posuere pede. Mauris ut est. Ut quis purus. Sed ac odio. Sed vehicula hendrerit sem. Duis non odio. Morbi ut dui. Sed accumsan risus eget odio. In hac habitasse platea dictumst. Pellentesque non elit. Fusce sed justo eu urna porta tincidunt. Mauris felis odio, sollicitudin sed, volutpat a, ornare ac, erat. Morbi quis dolor. Donec pellentesque, erat ac sagittis semper, nunc dui lobortis purus, quis congue purus metus ultricies tellus. Proin et quam. Class aptent taciti sociosqu ad litora torquent per conubia nostra, per inceptos hymenaeos. Praesent sapien turpis, fermentum vel, eleifend faucibus, vehicula eu, lacus.

Chapitre 3

Présentation du sujet

Aboard Engineering étant une entreprise spécialisée dans les domaines de l'informatique embarquée, les problématiques logicielles qu'elle rencontre sont liées au matériel sur lequel sera déployé le code développé. Pour facilité ce type de développement, les concepteurs de logiciels embarqués se tournent de plus en plus vers des outils de *MDB* qui permet d'utiliser des langages de haut niveau – souvent graphiques – pour la conception et la spécification des systèmes à développer. On peut citer par exemple Matlab® Simulink® qui est un outil permettant de définir graphiquement sous forme de schéma logique le fonctionnement de systèmes complexes. C'est d'ailleurs l'outil utilisé par Aboard Engineering pour définir ses fonctions de contrôle moteur. Les avantages des ces modèles sont multiples. Premièrement, les outils de *MDB* permettent de simuler l'exécution du système afin de vérifier son fonctionnement. Ensuite, ces outils embarquent généralement un générateur de code afin de générer le code correspondant à ces modèles, allégeant grandement le travail des concepteurs et évitant au maximum les erreurs faites lors de développement « manuel ». En reprenant l'exemple de Matlab® Simulink®, la suite intègre un générateur de code embarqué temps réel appelé *RTW-EC*®. Il génère aujourd'hui le code des prototypes de fonctions de contrôle moteur développés par l'équipe. J'y reviendrai dans la section **??** plus en détail.

Comme dit précédemment, la génération de code facilite le travail de conception et de réalisation de prototype. Cependant, pour pouvoir compiler le code de plusieurs modules différents en une seule application, il reste des étapes à réaliser. C'est ici que la plateforme de prototypage rapide Orianne entre en jeu. C'est son rôle de mettre en relation ces modules et de créer la « glue » entre tous les composants.
Orianne est composée de plusieurs parties que nous appellerons « composants ». Je détaillerai les composants présents lors de mon arrivée ainsi que ceux sur lesquelq j'ai travaillé dans la section **??**.

3.1 La génération de code embarqué

3.1.1 État des lieux

Le développement d'une application pour un calculateur moteur n'est pas une mince affaire. De nombreux modèles sont nécessaire à la spécification de tous les aspects du fonctionnement d'un moteur ou des différents commandes utilisateur (pédales, boîte de vitesse, etc.) par exemple. À cela peut s'ajouter le gestion d'une configuration hybride – un moteur thermique couplé à un moteur électrique – nécessitant une gestion plus pointue de l'énergie.

Une fois complets, ces modèles ont vocation à être traduit en code source qui sera compilé puis intégré dans un calculateur. C'est donc ce code compilé qui est la base du fonctionnement d'un véhicule. On comprend alors les contraintes de fiabilité et de performance sous-jacentes.

Pour générer ce code, nous utilisons Matlab® *RTW-EC*®. C'est un générateur de code embarqué temps réel puissant et reconnu pour fournir du code suivant les normes en vigueur dans l'automobile. Le code généré conttient toute la logique métier et donc l'essentiel de la complexité de l'application. Sur ce point, Matlab® *RTW-EC*® fournit des résultats très bons en terme de temps d'execution, de mémoire utilisée ou de taille de pile lors des appels aux fonctions des modules.

Cependant, une ombre plane sur ce tableau. La suite Matlab® coûte très cher. Même pour une

entreprise – d'autant plus pour une entreprise de petite taille comme Aboard Engineering –, les coût des licences Matlab® couvrant toute la suite nécessaire à l'ensemble du processus de production n'est pas négligeable et compte pour beaucoup dans les charges liées à un projet.

La problématique est alors de retrouver les même garanties de performance et de qualité du code produit par Matlab® sans le coût exorbitant qu'engendre l'achat de licences. C'est pourquoi Aboard Enginnering a décidé de se tourner vers une alternative libre : *QGen* et *Project P*.

3.1.2 QGen

A la base de *QGen*, on trouve un projet open source appelé *Project P*. Le but du projet P est de supporter le *MDB* pour des système temps réel embarqués en fournissant un framework de generation de code open source capable de :

1. vérifier l'intégrité d'un point de vue sémantique de sèstème conçus avec différents langages de modélisation (Matlab® Simuling®, Scicos, Xcos, SysML, MARTE, UML) ;
2. générer du code source optimizé pour des langages de programmation (Ada, C/C++) et des alngages de synthèse (VHDL, SystemC) ;
3. supporter des processus de certification de différents domaines (avionique, spacial, automobile).

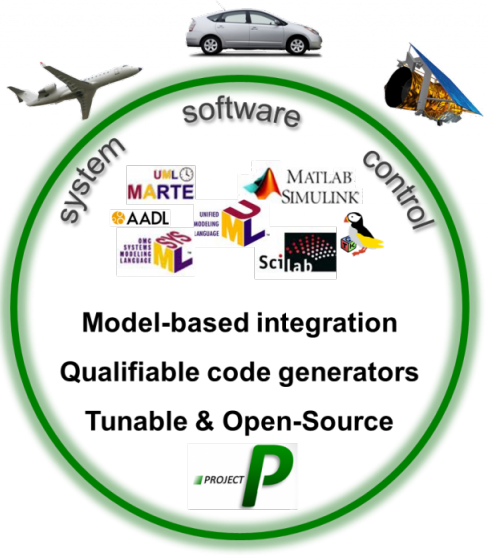


FIGURE 3.1 – Project P – Source : `http://www.open-do.org/projects/p/`

Parmis les collaborateurs à ce projet, on trouve des grands groupes industriels (Airbus, Astrium, Continental, Rockwell Collins, Safran, Thales) des PME (AdaCore, Altair, Scilab Enterprise, STI), des sociétés de services (ACG, Aboard Engineering, Atos Origins) mais aussi des centres de recherche (CNRS, ENPC, INRIA, ONERA).

À partir de ça, AdaCore a développé un produit commercial à partir des technologies développées dans *Project P* : *QGen*.

FIGURE 3.2 – QGen – Source AdaCore

3.2 Les outils de la plateforme

La plateforme Orianne est développée en Java. Le projet est composé de plusieurs modules pour séparer les différents outils. Deux principaux modules séparent la partie IHM qui sert à la configuration des outils d'une part et les générateurs de code d'autre part.

3.2.1 La partie IHM

La partie IHM s'organise sous forme d'onglets correspondant à la configuration de chaque outils (cf. figure ??).

Configuration moteur Cette partie permet de configurer les éléments relatifs à un moteur. Le type de carburant, de combustion, d'injection, les fonctions à intégrer dans l'application finale.

Configuration fonctions Chaque fonction cochée dans le premier onglet, peut être configurée dans le deuxième onglet. Les fonctions sont donc configurable via des éléments graphiques comme des cases à cocher, des champs texte ou des liste déroulantes.

Configuration RTOS Cet onglet reprend les différences fonctions triées par leur récurrences sous forme de tableau. Chaque ligne représente un point d'entrée d'une fonction de contrôle moteur. L'ordre est significatif et peut être modifié via des boutons. L'ordre des fonctions dans cette table sera répercuté dans le fichier source de configuration du RTOS.

Configuration *ECU* & *RTE* Cette section fait le lien entre les API fournies par les pilotes du calculateur moteur et les variables d'entrée/sortie des fonctions de l'application. En configurant les capteurs et actionneurs du calculateur avec leur donnée (variables), la plateforme permet de faire la correspondance entre ces variables et les données consommées par les différentes focntions de l'application.

Configuration CAN L'*ECU* n'est pas le seul calculateur dans un véhicule. La communication entre ces différentes unités se fait généralement via un bus CAN. L'onglet CAN permet de configurer les différentes cellules de communication et accepte des fichiers XML représentant les différents messages à envoyer sur une cellule définie. L'interface permet également de faire correspondre les signaux réçus ou envoyés à des variables de l'application en entrée ou en sortie des fonctions.

3.2.2 La partie génération

L'interface de la plateforme a un menu regroupant les différentes actions de génération possible via la plateforme (cf figure 3.3).

Génération CAN Cette action crée deux fichiers source `.c` et `.h` reprenant la configuration de la communication CAN.

Génération RTOS Cette action reprend les liste des points d'entrée des fonctions de l'application en fonction de leur place dans la configuration.

Génération RTE Cette action crée deux fichiers source `.c` et `.h` déclarant l'ensemble des entrées/sorties des fonctions qui seront échangées entre les différents modules. Elle génère aussi les fonctions d'accès en lecture et en écriture des données fournies par les pilotes du calculateur moteur.

Génération Orianne Cette action met en place les fichiers aux emplacements requis pour la compilation d'une application complète. Elle copie les sources générées par la plateforme et celles importées depuis la génération de code des modèles Simulink ®.

Compilation Cette action lance la compilation via le compilateur définie dans la configuration de la plateforme.

Génération A2L Cette action génère le fichier A2L servant d'index des variables dans la mémoire du calculateur.

3.2.3 La problématique

La problématique autour de cette plateforme est que tous ces outils ne sont pas encore créés.

Afin que la plateforme garde une stabilité, les générateurs non implémentés sont transformés par la copie de fichiers génériques embarqués en dur dans la plateforme. Ces fichiers doivent ensuite être modifiés à la main pour correspondre aux besoins de l'application conçue via la plateforme. Ainsi, les

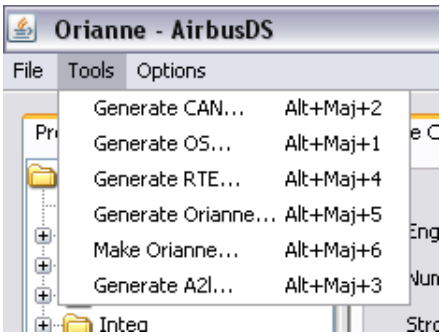


FIGURE 3.3 – Plateforme Orianne, menu Tools (capture du 17 juin 2015)

fichiers sont en place pour faciliter la compilation d'une application. L'utilisateur n'a alors plus qu'à modifier les fichiers dont il a besoin et de lancer la compilation de l'application.

Les outils manquant à la plateforme sont la génération CAN et la génération RTE. Il faudra donc développer les interfaces graphiques et les générateur de source correspondants. La prise en compte de ces nouveaux outils aura un impact sur les autres outils comme par exemple la génération RTOS qui devra prendre en compte les nouveaux point d'entrée de la communication CAN dont la structure pourra changer pas rapport à leur ancienne version.

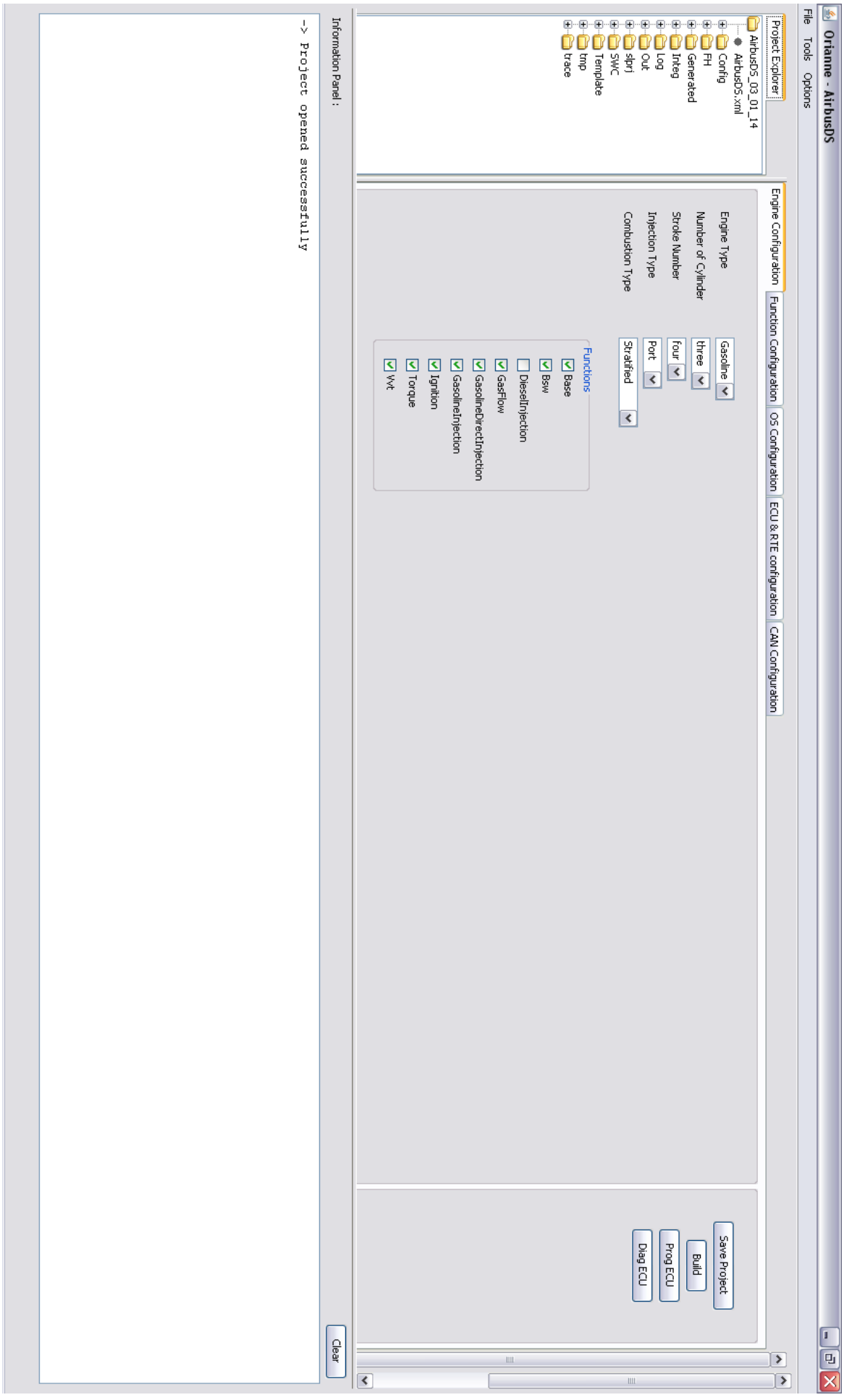


FIGURE 3.4 – Plateforme Orianne, configuration moteur (capture du 17 juin 2015)

Chapitre 4

Travail réalisé

<Introduction éventuelle>

4.1 Génération de code embarquée

4.1.1 Qu'est-ce que QGen ?

4.1.2 Pourquoi QGen (par rapport à RTW/EC) ?

4.1.3 Le processus de génération

4.1.4 Les premiers résultats

4.1.5 Les étapes d'amélioration

4.2 Outils de prototypage

4.2.1 La génération CAN

L'existant

La factorisation

L'intégration

4.2.2 Les autres outils

La partie RTE

La partie A2L

Chapitre 5

Bilan

Expérience professionnelle

Pourquoi ce stage

Les plus pour mon CV

Apports personnels

La communication en anglais avec AdaCore

Mes perspectives personnelles et professionnelles

Glossaire et acronymes

MDB **M**odel **B**ased **D**esign. Utiliser notamment dans des application automobiles, aérospatiale ou pour de l'informatique industrielle, MDB est un méthode mathématique et visuelle pour comprendre et résoudre des problème complexes de système de contrôle, de traitement du signal ou des systèmes communicants. Elle est beaucoup utilisée pour des logiciels embarqués.

Orianne //TODO <acronym>. Plateforme de prototypage rapide de fonctions de contrôle moteur. Voir 2.2 pour plus de détails.

OS temps réel (**R**eal **T**ime **O**perating **S**ystem en anglais). Système d'exploitation multi-tâches destinée aux applications temps réel. Il facilite la création d'un système temps réel via des ordonnances de tâches spécialisés afin de fournir aux développeurs des systèmes temps réel les outils et les primitives nécessaires pour produire un comportement temps réel souhaité dans le système final.

QGen Générateur open source de code embarqué temps réel en C et Ada à partir de modèles Matlab® Simulink®. Basé sur *Project P*, son développement est piloté par la société AdaCore.

RTW-EC® **R**eal-**T**ime **W**orkshop-**E**mbedded **C**oder®. Aujourd'hui renommé Simulink Coder™. Outil de génération de code C et C++ à partir de diagrammes Simulink®. Le code généré peut-être utiliser pour des applications temps réel et hors temps réel, notamment pour le prototypage rapide.

Table des figures