La conception orientée objet au secours de la programmation de microcontrôleur ou inversement…

G. Auriol[1,2], V. Mahout[1,2], T. Rocacher[2], P. Acco[1,2], PE. Hladik[1,2]
{guillaume.auriol@insa-toulouse.fr, vincent.mahout@insa-toulouse.fr }
1 CNRS, LAAS, 7 avenue du colonel Roche, F-31400 Toulouse, France
2 Univ de Toulouse, **INSA**, F-31400 Toulouse, France

RESUME : Cet article présente une maquette pédagogique ayant pour objectif l'apprentissage de compétences relatives à deux domaines à priori dissociés : la conception orientée objet (COO) et la programmation de périphériques d'un microcontrôleur. Le contexte et les pré-requis sont tout d'abord décrits. La seconde partie s'attache à décrire les réalisations matérielles qui ont été nécessaires pour la mise en œuvre de cette maquette. Le déroulement des séquences pédagogiques ainsi que les attendus au niveau étudiants sont ensuite détaillés. Le retour de l'équipe enseignante et des étudiants font l'objet de la conclusion de cette expérience.

Mots clés : dispositif pédagogique, COO, programmation de périphériques, microcontrôleur

1 INTRODUCTION

Ce papier décrit une maquette pédagogique qui cible l'apprentissage de la conception orientée objet (COO) et de la programmation des périphériques d'un microcontrôleur. La conception orientée objet est apparue en même temps que la programmation orientée objet dans les années 1980-1990. De nos jours, beaucoup de formations, notamment en informatique, associent la COO à la programmation, essentiellement sur PC, en langage JAVA. La programmation de périphérique, appelée parfois programmation de microcontrôleur, électronique numérique, informatique industrielle est une discipline qui présente les concepts et la pratique de développement de logiciel embarqué sur des microcontrôleurs. Nous avons choisi dans cet article de présenter notre expérience de fusion de ces deux disciplines pour démontrer leur complémentarité dans un cursus automatique et électronique.

Dans la partie (2), le contexte de l'enseignement ainsi que son objectif sont présentés. La partie (3) détaille les réalisations demandées alors que la partie (4) décrit la séquence pédagogique. La conclusion et les adaptations possibles de cette expérience sont données dans la dernière partie.

2 CONTEXTE ET OBJECTIFS DES ENSEIGNEMENTS

2.1 Contexte

Les étudiants à qui s'adresse cette formation sont en 4ème année dans une spécialité Automatique et Electronique (AE) de la formation initiale d'ingénieur de l'Institut National des Sciences Appliquées de Toulouse. Depuis de nombreuses années, les deux enseignements, la COO et la programmation de périphériques étaient dispensés de manière disjointe, chacune de ces matières ayant leurs propres objectifs. Pour ces étudiants de la spécialité AE, l'enseignement de COO leur paraissait relativement éloigné du cœur de leur formation. D'un autre côté, l'enseignement de périphérique se fait sous la forme d'un projet conséquent (une quarantaine d'heures) dans laquelle manquait une compréhension globale du cahier des charges.

2.2 Objectifs du regroupement des deux enseignements

Les objectifs du regroupement de ces deux disciplines sont d'une part d'améliorer la perception des étudiants de l'importance de passer par une étape de modélisation orientée objet lors de projet conséquent de développement et d'autre part de profiter de cette modélisation pour pouvoir aller plus loin dans la réalisation du projet de programmation des périphériques.

La maquette que nous avons construite autour d'un voilier de modélisme correspond complètement à nos attentes. En effet, d'une part vues les fonctions logicielles que nous demandons pour l'application (et les contraintes que nous imposons pour leur développement), il devient opportun de travailler en adoptant une méthodologie de conception et un formalisme orientés objet habituellement employés pour des projets conséquents (en terme de quantité de code embarqué notamment et de partage du développement entre plusieurs équipes). D'autre part, il est possible d'instrumenter cette maquette avec suffisamment de capteurs et d'actionneurs pour illustrer les principales unités périphériques d'un microcontrôleur.

3 RÉALISATIONS

3.1 La maquette du voilier

Cette maquette a été entièrement conçue par l'équipe. Nous sommes certes partis d'une base de modèle réduit « bas de gamme » dont nous n'avons conservé au final que le gréement (avec son servomo-

Article publié par EDP Sciences et disponible sur le site http://www.j3ea.org ou http://dx.doi.org/10.1051/j3ea/2013010

teur) et la quille. La coque et le gouvernail, inutile par la suite ont donc été supprimés.

L'équipe, en relation avec les personnels techniques en électronique et en mécanique, a donc été amenée à réfléchir au support (berceau en rotation sur plateforme tournante) ainsi qu'à l'ensemble des dispositifs électroniques (figure 1).

3.1.1 Aspect fonctionnel

Le bateau est tout d'abord lié à un berceau par un axe en rotation (⑥). Cette rotation permet de simuler une gîte due au vent. Le berceau est lui-même posé sur une plateforme sur roulette équipée d'un moteur (⑤) permettant son orientation. Ce degré de liberté permet de simuler un changement du cap du voilier dans un vent engendré par un ventilateur fixe (①). Le sens du vent est capturé par une girouette *maison* (④) et l'ordre de rotation du moteur est envoyé à la carte contrôleur par une télécommande de modélisme (⑨).

Les étudiants doivent concevoir le programme pour la carte microcontrôleur (⑦).

Le but de l'application est donc en premier lieu de border les voiles avec le servomoteur (⑤) suite à une rotation de la plateforme et en fonction de l'orientation du vent. Une unité d'émission/réception HF (⑧) permet de recevoir le signal de la télécommande et d'envoyer des messages de détresse vers une console de télésurveillance (simple PC non visible sur la photo)

Le bordage des voiles doit également tenir compte de la gîte qui est mesurée par un accéléromètre (③) placé sous le pont.

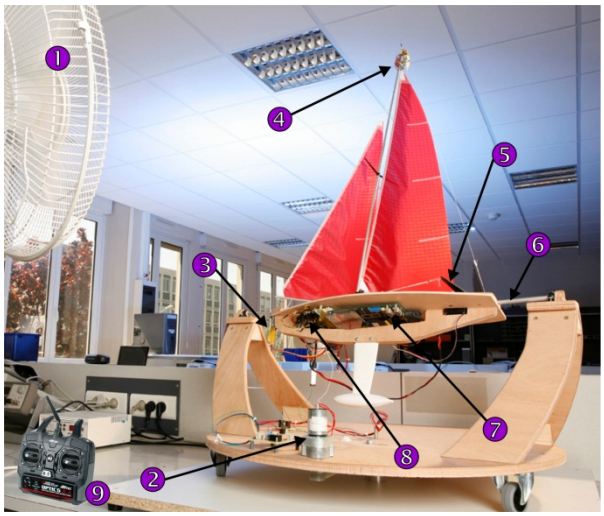

Figure 1 : Maquette de voilier de modélisme

3.1.2 Aspect électronique / mécanique

La figure 2 montre le schéma fonctionnel de la partie électronique du bateau.

Les cartes électroniques réalisées pour ce bureau d'étude sont relativement simples. En effet, la plupart des capteurs fournissent des tensions directement compatibles avec les niveaux requis du STM32F103, le microcontrôleur utilisé. Le seul actionneur qui requiert une adaptation est le moteur cc de 12V. En effet, le microcontrôleur n'est pas capable de fournir la puissance nécessaire. Un petit hacheur simple quadrant a été conçu et réalisé pour interfacer le moteur.

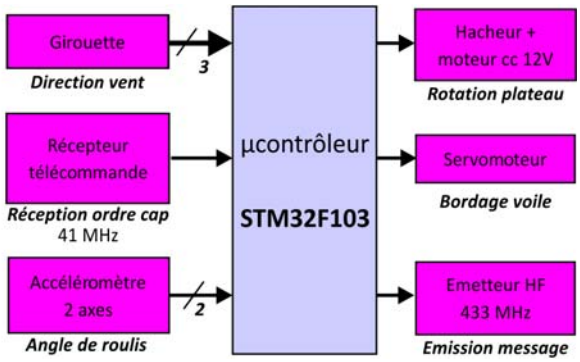

Figure 2 : Schéma fonctionnel de la partie électronique

Du point de vue mécanique, la difficulté majeure a été la fabrication de la girouette. Elle devait être légère et de faible encombrement. Nous sommes partis sur un codeur incrémental de référence AEDB-9140 de chez Agilent.. Il offre 360 périodes sur un tour avec deux voies A et B en quadrature, et une sortie I, l'index, qui donne une seule impulsion par tour.

L'alignement de l'électronique et du disque doivent être relativement précis. Le disque doit être solidaire d'un axe supportant l'aileron qui prend le vent. Il doit être monté sur roulement à billes si on veut espérer un mouvement de rotation dû au souffle du ventilateur. Le moindre frottement coincerait l'axe.

Pour la réalisation en aluminium du support mécanique (roulement, fixation,…) de la girouette, la considération de poids est critique, car si elle est trop lourde, la girouette peut créer un déséquilibre important de l'ensemble du bateau. La version définitive du support correspond en tout point au dessin de conception, sauf que la pièce la plus massive a été évidée pour gagner en poids. Cette pièce a conduit à un bel exercice de tournage (tolérance des alésages précis, usinage délicat de l'axe qui est relativement étroit de diamètre). De plus, l'exercice a dû être reproduit 6 fois, puisque nous disposons d'une « flotte » de 6 voiliers.

3.2 Programmation de périphériques

Un des objectifs initiaux de cette unité de formation est de montrer aux étudiants les concepts et les méthodes qui leur permettront d'être opérationnels dans la mise en œuvre d'une application logicielle embarquée et ce quel que soit le microcontrôleur utilisé. Le travail se déroule cependant sur un processeur imposé, tant il était difficile (matériellement parlant) de faire fonctionner les salles avec des processeurs divers et variés. Celui-ci est un STM32 (version 103RB)[1] .

Ce processeur 32 bits est en partie déjà connu par les étudiants puisque c'est sur ce processeur (et plus précisément son cœur) qu'est basé l'apprentissage en 3ème année de la programmation en langage d'assemblage [2]. Ils connaissent donc l'architecture générale de ce processeur et surtout ils ont déjà pratiqué l'environnement de développement (Keil microVision), ce qui se traduit par un gain de temps appréciable dans la prise en main du travail à faire.

Si très rapidement il nous est apparu que les objectifs visés ne pouvaient être atteints que si les étudiants passaient par une phase pratique conséquente, d'autres réflexions ont animé l'équipe pour trouver le format le plus adéquat. Nous souhaitions en effet illustrer au mieux les différences et les difficultés de la programmation d'un microcontrôleur (et donc la configuration de ses unités périphériques) par rapport à un programme informatique classique.

3.2.1 Périphériques étudiés

Dans cette partie, nous allons détailler les différentes unités périphériques utilisées par le projet en indiquant les éventuelles difficultés de mise en œuvre auxquelles sont exposés les étudiants.

Les ports d'entrées/sorties. Nommées GPIO (General Purpose Input Ouput) dans le monde STM32 ils constituent le point d'accroche idéal pour débuter sur un microcontrôleur. Ces ports réalisent le lien physique entre le processeur et le dispositif physique à piloter. Sur le voilier aucune entrées/sortie de type tout ou rien n'est utilisée. Par contre comme ces ports sont les portes d'accès pour les autres unités, il est nécessaire de les configurer correctement. Cette dualité entre fonction logique binaire et fonction alternative n'est pas forcément immédiate à comprendre.

Les Timers. Ce sont évidemment des unités essentielles dans l'organisation logicielle d'un tel projet. Lorsque les étudiants ont besoin d'en programmer un pour mettre en route un séquenceur qui va rythmer le déroulement des différentes tâches, ils seront déjà expérimentés dans la programmation de ceux-ci. En effet, ils en auront eu besoin pour mesurer la largeur des impulsions reçues par voie hertzienne en provenance d'une télécommande. La girouette, technologiquement parlant, est un codeur incrémental, ils auront donc appris à configurer un timer pour qu'il puisse lire un tel codeur.

La PWM. Le bateau est monté sur une plateforme tournante commandée par un moteur à courant continu, l'ordre de rotation est donné par la télécommande. Cela permet donc de simuler le changement de cap. Outre la configuration des registres d'un timer spécifique pour fabriquer le signal PWM et le besoin d'utiliser un oscilloscope pour vérifier un programme informatique, une petite difficulté théorique se fait jour : comment choisir la bonne fréquence de la PWM.

L'ADC. Un accéléromètre 2 axes est positionné sous le pont du bateau. Un autre circuit analogique permet de mesurer l'état de charge de la batterie. Ces grandeurs sont donc à mesurer avec l'unité ADC pour relâcher les voiles en cas de gîte trop importante ou d'envoyer un message en cas de batterie trop faible. L'unité ADC est assez complexe à mettre en œuvre car elle propose beaucoup de mode de fonctionnement avancé. Se pose aussi pour les étudiants la problématique du calibrage de l'information ainsi récupérée.

L'USART. Nous avons prévu une liaison HF pour transmettre des informations entre le poste de pilotage et le bateau. Cette liaison est attaquée par une simple liaison série RS-232 du côté du microcontrôleur. La mise au point est souvent assez pénible car les causes de dysfonctionnement sont fréquentes, multiples et décentralisées donc difficile à débusquer.

3.2.2 Organisation du développement

La phase de développement logicielle n'est pas uniquement une phase où il faut sortir du code pour que cela fonctionne. Nous insistons, en liaison avec ce qui est imposé en COO, sur un découpage en 3 couches de leurs programmes (figure 3) :

- la couche pilote est le plus bas niveau. C'est le seul niveau qui connait les registres et qui a donc directement accès aux unités. Elle est écrite sans présupposer une utilisation particulière par une application ;
- la couche service est un ensemble de fonctions qui masquent les périphériques matériels au niveau de l'application pour en offrir une représentation abstraite. Par exemple, le service d'acquisition d'une vitesse masque l'appel au périphérique ADC pour offrir une vision purement abstraite de la variable de vitesse ;
- la couche application ne comporte que du code dédié à une application. Elle comporte l'ensemble des fonctions de traitement et d'orchestration de l'application.

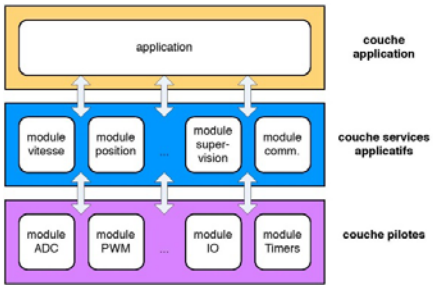

Figure 3 : découpage en couche de l'application logicielle

Au cours de ces premières séances nous attachons beaucoup d'importance à ce que cette structuration soit respectée. Outre que cela leur impose rigueur et modularité, cela permet par la suite la réutilisation directe des routines dans le projet.

3.3 COO

3.3.1 Introduction

Le développement de l'application logicielle est relativement conséquent et le travail se décompose assez naturellement en deux sous équipes : l'une sur le bloc télécommande + moteur, l'autre sur l'ensemble girouette + gîte + bordages de voiles. Cette décomposition est en effet fortement conseillée pour se donner toutes les chances d'arriver à la fin du projet.

L'intérêt de mener sérieusement cette étape de conception préliminaire par une approche COO est de diminuer le risque d'erreurs de conception qui vont apparaître au moment de l'intégration du travail des deux équipes. En effet, elles développent en parallèle du logiciel s'implantant sur un seul microcontrôleur et donc devant partager les mêmes ressources ou des ressources limitées. A titre d'exemple, il est dommageable de s'apercevoir le jour de la mise en commun que les deux sous équipes utilisent le même timer mais configuré différemment.

La démarche globale est assez classique mais passe dans notre cas par une formalisation explicite. Elle se déroule en trois phases.

La phase 1 consiste à **préparer la conception** : à partir du cahier des charges elle cherche à illustrer un scénario typique d'utilisation de la maquette en identifiant les acteurs principaux. Pour chacun de ces acteurs, nous leur demandons d'expliciter les paramètres qui peuvent influencer le comportement de la maquette. Pour chacun de ces paramètres, ils doivent trouver le composant matériel embarqué sur le voilier qui permet de prendre en considération le changement d'un de ces paramètres. Enfin pour chacun de ces composants, il leur est demandé de définir les fonctions ou périphériques que le microcontrôleur embarqué pourrait utiliser pour faire l'interface entre le cœur du logiciel et ses composants.

La phase 2 vise à **préparer l'intégration** en proposant des scénarios de test. Ces test devront valider que chacun des périphériques pris isolément fonctionne selon le comportement spécifié (tests unitaires). Ils serviront aussi à valider que la plateforme présente un fonctionnement général satisfaisant (tests d'intégration).

La 3ème et dernière phase concerne **la mise en place logicielle de l'environnement de travail**. Son objectif est de mettre en place l'environnement Keil µVision en préparant toute la structure logicielle du projet. Il s'agit donc de préparer :

- la structure des fichiers de l'application : fichier *.h, *.c et éventuellement *.txt) ;
- les structures de données communes à toutes les parties logicielles.

3.3.2 Diagrammes UML

L'enseignement de la partie COO est fait sur la base de l'apprentissage du langage UML (*Unified Modeling Language*) [3] ainsi que d'une méthodologie de conception orientée objet. Seuls quelques diagrammes sont présentés durant cet enseignement au cours. Les étudiants respectent un découpage de leur modèle en accord avec les couches présentées dans la partie 3.2.2. La construction du modèle se fait de façon incrémentale.

Le cahier des charges. Ainsi dans un 1er temps, les étudiants doivent faire une étude des fonctions applicatives requises et/ou fournies par le système « PlateformeBateau ». La figure 4 présente les fonctions principales de l'application sous la forme de cas d'utilisation ainsi que les parties prenantes sous la forme d'acteurs à l'intérieur d'un diagramme des cas d'utilisations.

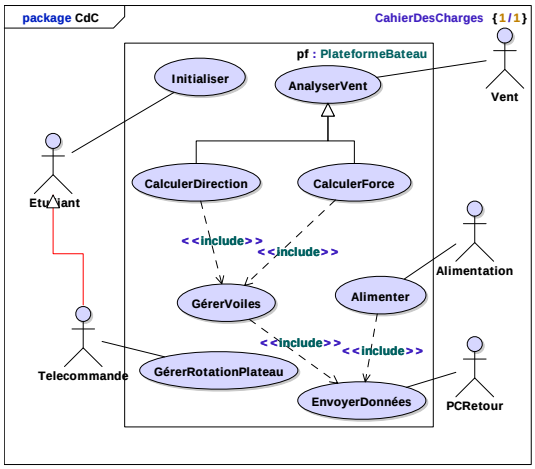

Figure 4 : cahier des charges

Le scénario principal. Le diagramme de séquence représenté dans la figure 5 permet de visualiser l'enchainement des principales fonctions. Par exemple la fonction « Initialise » qui calibre les accéléromètres et la girouette

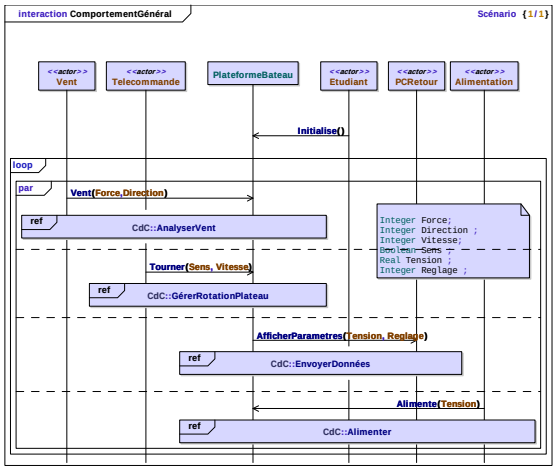

Figure 5 : scénario

Les composants et fonction détaillées. Les étudiants doivent détailler chaque fonction en impliquant les composants présents sur la plateforme. Pour cela, les

étudiants font un diagramme de classe pour représenter les composants embarqués figure 6.

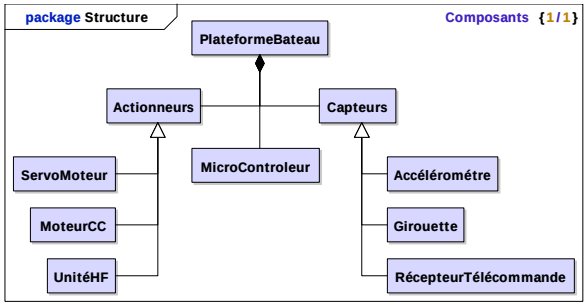

Figure 6 : composants

Puis par le biais d'un diagramme de séquence, ils détaillent chacune des fonctions. La figure 7 illustre le détail dans la fonction « GérerRotationPlateau ».

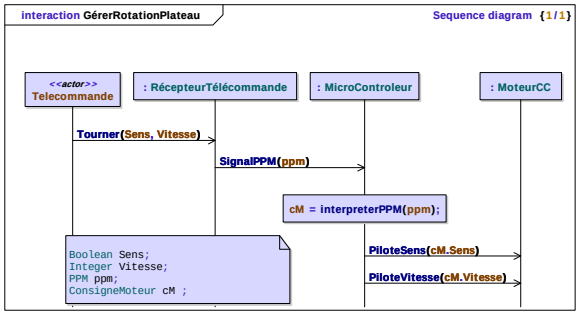

Figure 7 : détail d'une fonction

Passage aux couches services et pilotes. Les étudiants modélisent ensuite les fonctions au niveau du microcontrôleur. Ils font pour cela un cahier des charges (CdC) du microcontrôleur, figure 8.

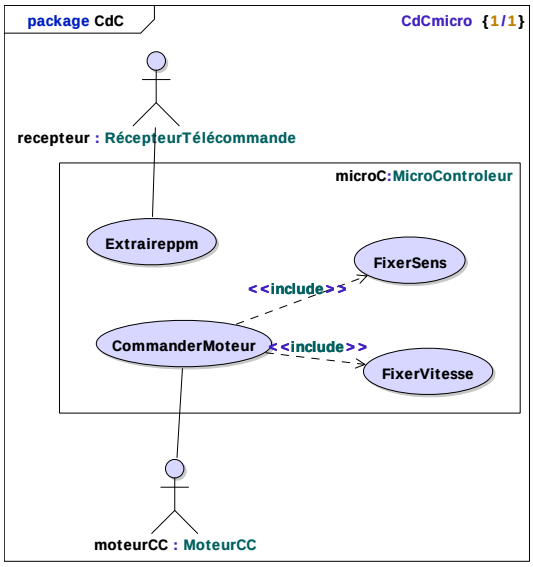

Figure 8 : CdC du microcontrôleur

Enfin, pour représenter les échanges entre les composants logiciels, les étudiants réalisent un diagramme de structures composites, figure 9 et un diagramme d'activités, figure 10.

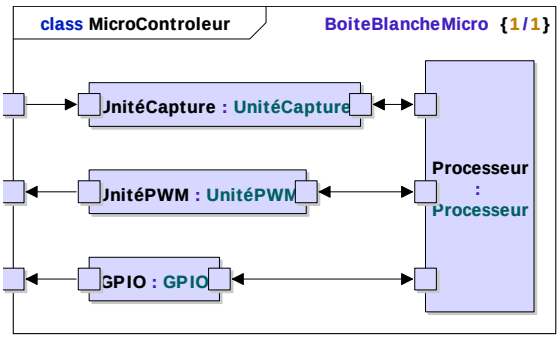

Figure 9 : Communications internes

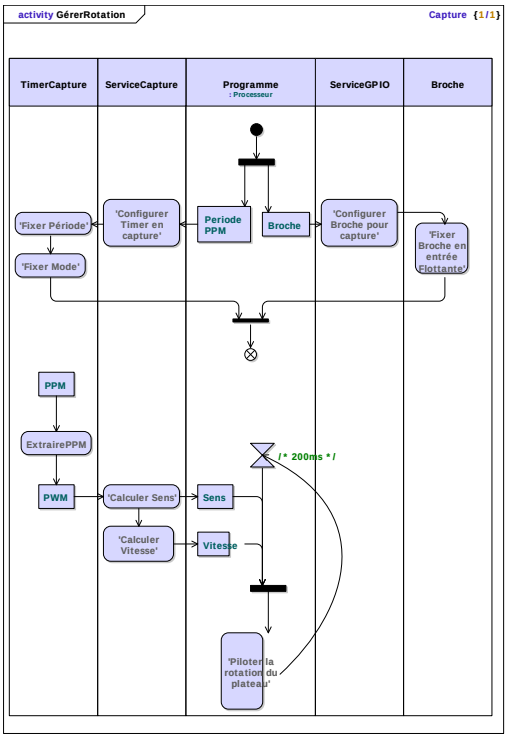

Figure 10 : Activité lors du traitement de la fonction « GérerRotation »

4 DÉROULEMENT DES SÉQUENCES PÉDAGOGIQUES

Afin que les deux disciplines puissent tirer partie de leur association, il est très important de construire une séquence pédagogique attentive à la progression des étudiants.

La partie COO comprend une séquence de :

- 5 cours magistraux de présentation du langage avec en parallèle la présentation d'une méthodologie de conception,
- 3 séances de travaux dirigés sur l'application des concepts vus en cours en étudiant un

exemple de projet de logiciel embarqué indépendant de la maquette de voilier,

- 4 séances de travaux pratiques en relation avec la maquette de voilier.

L'enseignement de la partie programmation de périphérique se fait uniquement sous forme de séances de travaux pratiques. Elle comporte 11 séances de travaux pratiques de 2h75 et se décompose en deux grandes phases. Dans un premier temps (entre 10h00 et 15H00 de séance) les étudiants travaillent en binôme sur des sujets de type TP. Durant cette phase très guidée, les étudiants travaillent sur des périphériques simples (ports, *timer*). Ils nous est en effet apparu nécessaire de passer par cette étape assez formatée pour permettre aux étudiants de rentrer plus facilement dans la problématique de programmation d'un microcontrôleur mais aussi pour leur monter peu à peu comment appréhender une unité particulière. Les textes des sujets de TP sont au départ très détaillés et très orientés. A titre d'exemple, dans le premier sujet sur les GPIO, nous leur demandons de dénombrer le nombre d'I/O sur le STM32 utilisé ; information pas si triviale que cela à trouver dans la documentation constructeur. Au fur et à mesure que les étudiants avancent dans le travail, les sujets s'allègent en consigne. Toujours à titre d'exemple pour l'unité PWM nous ne demandons plus aux étudiants de programmer les routines. Ils ont juste besoin de prévoir le prototypage des fonctions, prototypage qui ne peut se faire qu'en ayant compris les bases du fonctionnement de la PWM.

La deuxième phase est la partie projet à proprement parlé. Les étudiants sont regroupés en équipe (de 3 à 6 selon le feeling de l'enseignant) et doivent mettre en commun les programmes écrits dans la première phase pour construire l'application complète. Ils utilisent évidement toute l'approche qu'ils ont eue en Conception Orientés Objet. Ils se partagent alors les tâches et les tests. Ils sont aussi confrontés à la difficulté majeure de cette étape à savoir que la phase d'intégration, même si elle a été préparée par l'approche COO, n'est pas obligatoirement sans surprise. Une fonction peut être en effet complètement fonctionnelle lorsqu'elle est utilisée en unitaire mais ne pas être efficace quand elle est intégrée dans une application plus contraintes, notamment sur les aspects temporels.

Le tableau 1 illustre l'enchaînement sur 9 semaines des séquences des deux disciplines.

5 RETOUR D'EXPERIENCE ET CONCLUSION

Au final, l'ensemble des équipes parviennent à fournir lors de la séance de présentation, un délivrable du projet qui répond globalement au cahier des charges. Certains étudiants s'amusent même beaucoup sur ce genre de projet : un trinôme l'an passé a codé la PWM de telle sorte que le moteur chante en jouant sur la fréquence de la PWM. La *partition* était codée dans l'application avec donc la possibilité de faire jouer différentes mélodies… Le ressenti globale est donc particulièrement positif.

		COO			Périphérique	
		CM	TD	TP	TP	Projet
Semaines	1	2			1	
	2	2			1	
	3	1	1		2	
	4		2	2	1	
	5			1		2
	6			1		1
	7					1
	8					1
	9					1

Tableau 1 : Séquencement pédagogique

L'équipe enseignante a également largement tiré bénéfice de ce rapprochement et de cette interdisciplinarité. En effet comme en pratique nous avons demandé a paralléliser un maximum les séances de TP sur les différent groupes (jusqu'à 4 groupes en parallèle), des enseignants de formation et de sensibilité différentes (certains plus électroniciens d'autre plus habitué à la conception objet) ont pu mener l'ensemble du projet, en échangeant et en croisant les compétences.

En conclusion cette expérience est des plus enrichissantes et appelle à des prolongements ou à des évolutions. La première idée envisagée serait de diversifier les maquettes. Cela permettrait de pouvoir mettre en œuvre une autre piste. Celle-ci consisterait à adopter un fonctionnement plus mutualiste dans la conception et l'écriture des applications : deux équipes échangeraient leur conception. Ainsi les étudiants seraient amenés à coder et à tester une application qu'ils n'auraient pas conçue.

6 REMERCIEMENTS

Du point de vue technique, la maquette support de cet enseignement a nécessité le travail de plusieurs équipes dans divers domaines :

- Mécanique pure (Girouette) : José MOREAU
- Réalisation des plateaux, des bateaux : José PEREZ et Fabien NOUGAROLLES
- Electronique : José MARTIN

Nous les remercions pour leur travail conséquent et efficace, sans lequel ces maquettes (6 exemplaires) très attractives pour les étudiants n'auraient pu voir le jour.

7 BIBLIOGRAPHIE

[1] ST Micro RM0008 Reference manual, disponible sur http://www.st.com/internet/mcu/class/1734.jsp

[2] Mahout Vincent « Programmation en langage d'assemblage : ARM Cortex-M3 », *Hermes N°ISBN 978-2-7462-3211-2, 2011*

[3] Muller Pierre-Alain, Gaertner Nathalie, « Modélisation : objets avec UML », Eyrolles 2000