An Interdisciplinary Capstone Design Experience on Critical Embedded Systems using Agile Methods

Pascal Acco, Guillaume Auriol, Elodie Chanthery, M.-A Détourbe, Pierre Emmanuel Hladik, Didier Le Botlan, N Noullet, Audine Subias

▶ **To cite this version:**

Pascal Acco, Guillaume Auriol, Elodie Chanthery, M.-A Détourbe, Pierre Emmanuel Hladik, et al.. An Interdisciplinary Capstone Design Experience on Critical Embedded Systems using Agile Methods. Journal sur l'enseignement des sciences et technologies de l'information et des systèmes, 2019, 18 (0001), 23p. ⟨10.1051/j3ea/20190001⟩. ⟨hal-02189482⟩

HAL Id: hal-02189482

https://laas.hal.science/hal-02189482v1

Submitted on 23 Jul 2019

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

HAL Authorization

An Interdisciplinary Capstone Design Experience on Critical Embedded Systems using Agile Methods

P. Acco[1] G. Auriol[1] E. Chanthery[1] M.-A. Détourbe[1]
P.-E. Hladik[1] D. Le Botlan[1] N. Noullet[2]
A. Subias[1]
[1] LAAS-CNRS, Université de Toulouse, CNRS, INSA, Toulouse*
[2] Kagilum, Toulouse

Résumé

This paper relates a capstone design project conceived through an engineering problem-based learning approach. The project is applied to critical embedded systems and the Scrum Agile method is used. The focus of the paper is on the pedagogical experience, so the Intended Learning Outcomes and pedagogical sequence over one semester are presented. Two platforms (the Parrot AR.Drone 2 and a toy car) were used for the project. Pros and cons of such platforms are debated. Tools like Git for the version control and iceScrum for the Agile management are presented. The different roles of teachers as clients, counselors, and evaluators are detailed. This paper shows that an interdisciplinary teaching team is very important in this kind of project, so is the teachers' engagement in the Agile philosophy. Feedback, analyses and some suggested improvements are given, based on our 5 years' experience in such projects and surveys.

Résumé

Cet article relate notre expérience de mise en œuvre d'un apprentissage basé sur la résolution de problèmes d'ingénierie au travers d'un projet interdisciplinaire. Ce projet a pour domaine d'application les systèmes embarqués critiques. La méthode agile Scrum est utilisée comme méthode de gestion de projet. L'article porte principalement

*Authors listed in alphabetical order.
Corresponding authors : elodie.chanthery@laas.fr ; pehladik@laas.fr

sur notre expérience pédagogique : il présente les acquis d'apprentissage visés et la séquence pédagogique tout au long du semestre. Deux plates-formes (le Parrot AR.Drone 2 et une voiturette pour enfants) ont été successivement utilisées pour ce projet. Les avantages et les inconvénients des deux plates-formes sont débattus. Des outils tels que Git pour le contrôle de version et iceScrum pour la gestion agile sont présentés. Les différents rôles des enseignants en tant que clients, conseillers et évaluateurs sont détaillés. Cet article montre qu'une équipe pédagogique interdisciplinaire est très importante dans ce type de projet, de même que l'engagement des enseignants dans la philosophie Agile. L'article donne un retour d'expériences, des analyses sur les résultats obtenus, des suggestions d'amélioration sur la base de nos cinq années d'expérience et sur des sondages effectués auprès des différentes cohortes d'étudiants.
Mots-clés : projet interdisciplinaire, systèmes embarqués, méthode agile, apprentissage par résolution de problèmes d'ingénierie

1 Introduction

Our Institute is an international, pluridisciplinary, state-funded engineering university which provides a range of 8 engineering specializations. Several interdisciplinary courses are offered in 4th and 5th year (Master's degree). A specific feature of these courses is that they can be accessed by students who majored in different fields, which enables them to build out original profiles at the crossroads between several disciplines. The Software Critical Embedded Systems course is for 5th year students who majored either in Automatic Control and Electronics, or Computer Science and Network Engineering.

The development of critical embedded systems is known to be a strongly interdisciplinary domain where the students need to mobilize technical skills in electronics, control, software development, communication buses, interfacing analogic and digital systems, safety, hardware and software test,... but also project management skills. In practice, we note that our students develop all theses skills through a multidisciplinary approach, whereby they "draw on knowledge from different disciplines but stay within their boundaries," as opposed to an "interdisciplinary [approach which] analyzes, synthesizes and harmonizes links between disciplines into a coordinated and coherent whole"[7].

Moreover, even if Agile methods are not commonly used to manage critical embedded systems development projects, recent studies such as [13, 9] show that these Agile practices can be efficient, and major companies like

Thales Avionic [6] encourage their use. We are convinced that Agile methods have to become part and parcel of any engineer's education in the field of critical embedded systems.

Objectives. This paper presents a project run for the first time in the academic year 2012-2013. It was designed to integrate the main concepts taught in the Critical Embedded Systems Software Engineering while focusing on the design of embedded systems including real-time critical aspects. The goal of this kind of project is to build stronger connections between what the students learn or have learned in other courses. The course was built through a Problem-Based Learning approach and the technical development of a product was combined with the use of Agile methods. Through this article we aim to provide feedback on this course, give some advice to overcome the difficulties that we have met and disseminate Agile methods for the development of critical embedded systems.

Structure. The paper is organized as follows : Section 2 sets out the main principles of Project-Based Learning (PBL) and Agile methodologies, and some existing pedagogical works mixing PBL and agility are presented. Section 3 presents the main concepts of Scrum. Then the intended learning outcomes of the project and its pedagogical sequence are detailed in Section 4. The platforms and tools used during the project are presented in Section 5 and the role of the teachers in Section 6. Finally, Section 7 sums up some feedback, analysis and improvements we want to share about the project. Finally, Section 8 concludes the paper.

2 Related work

2.1 Project-Based Learning

Problem-Based Learning (PBL) belongs to the wider category of student-centered learning, an approach which has gained wide support in various higher education settings over the last decades. This approach has long been introduced in engineering education, through interdisciplinary capstone design projects for instance [12, 8]. The core principles underlying student-centered learning can be summed up as follows [3] : the reliance on active rather than passive learning ; an emphasis on deep learning and understanding ; increased responsibility and accountability on the part of the student ; an increased sense of autonomy for the learner ; an interdependence between teacher and learner ; mutual respect within the learner-teacher relationship ; a reflexive approach to the teaching and learning processes on the part of both the teacher and the learner.

In PBL approaches, a group of students is given a scenario (a problem to solve, an issue to understand, etc.) designed by the teacher(s). They must get organized as a team in order to analyze the problem, identify the relevant areas of knowledge they need to command, and devise possible solutions while drawing on each student's prior knowledge, sharing such knowledge, gaining further knowledge, if necessary, meeting specific deadlines and communicating their findings and solutions in an appropriate fashion. The teacher's role is to guide the students throughout the learning process, and to design adapted modes of assessment. In engineering education, the scenario is framed by the expected delivery of a specific output, side deliverables such as written documents (reports, specifications, etc.) or progress reviews, and a strict time-line which requires specific modes of organization. The demands placed upon the students may be even greater when external clients from the industry or the corporate world are involved.

Thus, the benefits of PBL go beyond the mere acquisition of knowledge and encompass generic soft skills such as "communication, teamwork, problem-solving, independent responsibility for learning, sharing information, and respect for others" [20]. It is particularly adapted to the context of engineering education as future engineers must now get ready to solve increasingly complex problems which require specific skills related to their particular technical field but also the ability to identify non-technical aspects of problems, the interaction between these aspects as well as possible solutions [14]. PBL has also been found to favor stronger student engagement while increasing their motivation ; it gives students the opportunity to activate prior knowledge and to develop specific learning styles while participating in group dynamics. On top of other benefits, it is a powerful tool for handling student diversity.

As a result, it comes as no surprise that PBL approaches should meet with great success in engineering education, as visible through the literature : while some teachers and institutions resort to PBL in order to increase student satisfaction, or students' intrinsic motivation, many others rely on PBL to strengthen students' learning outcomes in various fields, e.g., [21, 19].

2.2 Agile methodologies

For decades, the main formal software project management process was the Waterfall model, where the project life cycle is split into activities - requirement analysis, design, implementation, verification and maintenance - executed in a single sequence over several months or years. While it can be successful in other domains, the effectiveness of this process for software de-

velopment has been challenged. In 2001, new perspectives were opened with the Manifesto for Agile Software Development [5], co-signed by 17 software developers from the software industry.

The term "agile" itself has no formal definition. Rather, it encompasses values, principles, practices and methodologies which are described in [5] or developed by its co-signatories : product quality and customers' needs are prioritized over other aspects of project management. Means of achieving this goal include regular inspection, continuous improvement, adaptation, and collaboration through successive iterations that produce small product increments at a steady pace. Thus, as opposed to the Waterfall model, the main activities are not split across the whole project, but rather executed individually for every requirement from its definition to its validation.

The iterative and incremental nature of Agile project development makes it possible to take into account changes in the requirements at any moment of the project's life cycle, which may be required in contexts characterized by significant uncertainty. In software project management, this uncertainty can be met both at the business level with the difficulty of defining exhaustive and correct requirements in advance, and at the technical level due to the use of technologies that may have unexpected limitations.

Two of the methodologies that stemmed from the Agile philosophy became quite popular : Extreme Programming [4] and Scrum [18]. It appears that Scrum has now become the most prevalent Agile methodology in the software industry.

2.3 Agile methodology in education

Numerous recent articles focus on Agile practices in education, e.g., [15, 10, 17, 2], which indicates the will and the need to teach students this methodology, especially in Computer Science and Information Technology. These articles show that the use of Agile methods in educational activities has very positive effects. Many of them also highlight the fact that both the pedagogical sequence and the teachers' roles have to be well defined and understood by the teaching team as a whole.

However, the Agile approach is not commonly used by the industry for embedded systems, and even less so for critical systems. Nevertheless, papers such as [13, 9] describe the challenges and opportunities raised by the use of Agile methods in the field of safety critical systems development, and we believe that Agile methods should be included in the educational background of the engineers who develop critical embedded systems. Note that the ACM and IEEE Curriculum Guidelines for Undergraduate Degree Programs in

Computer Engineering [1] introduced the Agile methods for hardware and software design as a Knowledge Unit in 2016.

Experiments to introduce Agile management to develop embedded systems have been already conducted. The authors of [16] present an academic project for a spatial critical embedded system using the Scrum method. The project was divided into 5 components, each one being managed by one Scrum Team. The paper is focused on student work and not on pedagogical design. In [11], overview, experiences and lessons learned from an Agile embedded system project are presented. The projects under study are centered on the link with industrial issues and real platforms are not necessarily used, which differs from our own approach. Moreover, there is no use of a specific tool for Scrum. The article mentions that the ideal approach is for interdisciplinary teams to collaborate toward a complete project, which is precisely what we did.

3 Scrum

Software projects often do not meet the planned schedule, which usually results in belated delivery of the product. Scrum [18] tackles this issue by defining set periods of time named **Timeboxes**. When unexpected events occur, instead of adjusting the schedule, the **Team** adjusts the functional scope according to the work achieved over the predefined period.

A major working version of the product is expected at the end of a high-level Timebox named **Release**. A Release, which can last several months, is split into iterations named **Sprints**. A Sprint usually lasts a few weeks.

Business software requirements are described by means of **User Stories**. The list of User Stories to be produced is called the **Product Backlog**. Unlike predefined requirement specifications, User Stories are created and updated in the Product Backlog in order to take changes into account throughout the project. The person in charge of maintaining the Product Backlog and its User Stories is the **Product Owner**, who embodies the business vision of the product.

The Team commits to the order in which it will produce the User Stories. Story prioritization is chosen by the Product Owner and consists in ordering the Stories in the Product Backlog.

Another important role in Scrum is that of the **ScrumMaster**, who is here to help the Team reach its goals. ScrumMaster responsibilities include anticipating issues, fostering good practices and helping organize Team activities. The ScrumMaster has no authority over the Team.

Before a Sprint begins, the Team and the Product Owner gather at the **Sprint planning meeting** to plan a subset of the highest priority User Stories in the Sprint Backlog. Then, Team members split stories into **Tasks**, which are the activities required to implement and deliver the stories. Every working day, the Team starts with the **Stand-up meeting**, a short meeting where Team members alternately review Task completion and potential problems. At the end of the Sprint, the Team presents the achieved Stories to the Product Owner and stakeholders at the **Sprint review meeting**. Before starting a new Sprint, the Team meets for the **Retrospective meeting** during which its members discuss how the Sprint went and then plan a few corrective actions accordingly.

4 Intended Learning Outcomes and Pedagogical Sequence

4.1 Intended Learning Outcomes

The project was run for the first time in the academic year 2012-2013. It was designed to integrate the main concepts of the Critical Embedded Systems Software Engineering focused on the design of embedded systems including real-time and critical aspects. The goal of this kind of project is to build bridges between what students have studied in other courses and therefore knock down disciplinary barriers.

Regarding the Intended Learning Outcomes, at the end of the project, each student should be able to :

1. explain the principles and the specifics of an Agile approach in comparison with other project management methods,
2. apply Scrum in a group to manage a product development,
3. apply knowledge and skills from earlier courses in their specialization to design and implement a critical embedded system,
4. acquire and evaluate solutions in an autonomous way to solve a technical requirement when designing an embedded system,
5. communicate in an interdisciplinary context and with various actors who have heterogeneous skills,
6. adapt written and oral communication to meet various expectations (client, decision-maker, evaluator, general public) through various media (web site, report, abstract, poster).

4.2 Pedagogical Sequence

The project is usually carried out by about 24 students divided into 4 or 5 teams. If the platform is the same for all the teams, each Team has a specific project on the basis of the user requirements provided by one or two domain experts playing the role of Product Owners. Consequently, some tasks are similar from one Team to the next, but each project has its own specificities. Each Team chooses a project from a list provided by the teaching team.

The project lasts 14 weeks (a semester) with 2 project sessions of 2.75 hours per week plus additional personal work. It is worth 8 European Credit Transfert and Accumulation System (ECTS) points. The expected workload for each student (including contact hours) is about 100 hours. The project is coupled with the English courses.

The project starts with formal project management lectures (15 hours) to teach the students the Agile philosophy and its application to critical embedded systems. It is then divided into six Sprints, each lasting two weeks as shown in Figure 1.

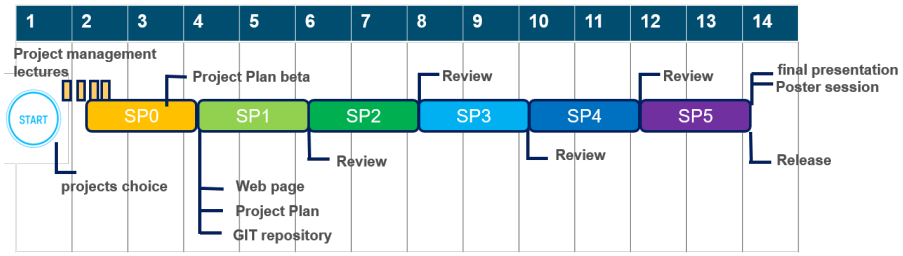

FIGURE 1 – Pedagogical Sequence

The first two weeks (called Sprint 0 since it is a preparatory Sprint before the project actually starts) are also used to define the objectives of the project. At the end of Sprint 0, students deliver a project plan and a web page. They provide an executive summary of the project (project description, stakeholders, release vision, features of the product), analyze their Team skills, the milestones and deliverables of the project and the project planning (first Sprint objective, Product Backlog with prioritized Stories, schedule control) as well as the risk management plan. The five other Sprints start with a retrospective during which the Sprint planning is discussed. Each Team has to maintain its Sprint backlogs, adding new Tasks if required and updating data on the what has been done and what is left to do. During the Sprints, the supervisors (one or two teachers) meet the Team once a

week. These meetings are scientific meetings. One student is identified as the ScrumMaster of the Team for one Sprint. In order to familiarize each student with this role, we demand that the ScrumMaster change at each Sprint so that every student experiences this role at least once during the project. The role of the ScrumMaster is not the same as that of project manager : (s)he advises the team to help it improve (for example, the ScrumMaster does not plan the work, it is up to the whole team to do it). The ScrumMaster is also in charge of giving feedback to its tutors after each work session.

At the end of each Sprint, the Sprint review meeting takes place : each Team presents what has been achieved during the Sprint. Results are discussed with other students and with the teachers who play the role of clients. The goal is to see the actual progress of the product through a demonstration of the current version, and to discuss the future directions. From a pedagogical point of view, teachers assess the students' ability to express themselves, to demonstrate their results and to manage their project. They also give suggestions for the future Sprint as clients.

After five Sprints, the first Release should be complete. The aim of the final presentation is to review each project one last time : each Team presents their initial objectives, the technical means used to achieve them, the concrete results obtained and the possible evolutions. The audience is a specialized one in this case.

There is no formal final exam, but each step of the project is assessed (project plan, web page, each review, final review) in order to determine the extent to which the intended learning outcomes have been reached by each student. At the end of the project, a poster session takes place. People from embedded and/or critical systems industries are invited. Students from other majors and other teachers are also invited. The goal is for the students to communicate about the project, show the final product and promote the work done throughout the semester in a professional context. The audience is both specialized and non-specialized this time.

5 Platforms and Tools

5.1 Platforms

Until recently (2012-2016), we used the Parrot [1] AR.Drone 2 platform. The choice of an appropriate platform is essential. Here are some criteria for choosing the "ideal" platform :

1. See https ://www.parrot.com/uk/drones/parrot-ardrone-20-elite-edition

1. an industry oriented platform : the platform has to be consistent with industrial practice ;
2. a robust but unexpensive platform : an expensive platform is quickly rejected when you need multiple instances, given the budget constraints in higher education institutions ;
3. a compact size : it is easier to store and to operate ;
4. an attractive platform to sustain students' enthusiasm throughout the project : aesthetics matter too ;
5. A worthy experience : the project is a very important part of the students' final year curriculum. It often represents a robust industrial and professional experience which can be highlighted in their CVs ;
6. a platform that fits academic specific safety rules : some industrial products are not suitable in an academic context due to specific safety rules.

In our case, other characteristics have been considered. Because the students have different academic backgrounds, the project should encompass many different skills, so we need :

7. an interdisciplinary platform : the students should be able to access the control models, communication protocols, and algorithms already embedded on the platform, in order to be able to develop new features quickly, so we chose (viii) an open platform ;
8. an active community : it is needed because most of the time, the students work independently with minimal supervision so a platform with well documented components and an active community helps increase students' autonomy.

Initially, the Parrot AR.Drone 2.0 platform seemed to meet all the requirements. It is low-cost, small, and includes many built-in features such as altimeter devices, movie camera, WiFi connections... Drones are innovative smart systems so they are attractive for students and can be put forward in a job interview. The platform is made of EPP (expanded polyproylene), so there is no body harm if an uncontrolled drone accidentally crashes on a student or a teacher. Parrot also includes a software development kit (SDK) [2] and numerous open projects such as Paparazzi [3] or ROS [4] use this platform.

2. See http ://developer.parrot.com
3. See https ://blog.paparazziuav.org
4. See http ://bebop-autonomy.readthedocs.io/en/latest/

The main skills used on this platform were related to software development, including real-time movie processing. There were only a few control-centered projects and even fewer electronic design-based projects. These two points can be easily explained by the fact that (1) the AR.Drone has very low airlift capability, which prevents the students from embedding additional electronic devices of their own. The maximum cargo capacity is below 100g, hence a low autonomy (a couple of minutes) ; and (2) the SDK provided by Parrot does not give full open access to the control library, so it is very complex for the students to implement their own control law without having a realistic behavioral model of the drone.

After using the drone for 4 years, we decided to switch to a new platform with a cargo lift offering more possibilities and full open access to the software components (control algorithm, communication protocol, movie camera software,...). It was not possible to pick a more powerful drone for safety reasons. Consequently, we chose a cheap car toy (Figure 2). We only kept the chassis frame, which provides a cargo lift of around 15kg, the wheel motors, the steering system, as well as some very well known Raspberry Pi [5] outfitted with movie camera, WiFi connection and display screen. An SMT32F103 micro-controller was used for the low level control and a CAN bus was used for the communication between STM32 and Raspberry. The students added a lot of electric devices : ultrasonic sensors, GPS, additional movie camera, lidar, other micro-controllers... The Raspberry system is based on a free Linux distribution, ensuring full open access.

5.2 Versioning and Agile tools

In professional contexts, teams usually rely on tools to structure their collaboration and increase the chances to meet their goals. Such formalization also enhances communication with stakeholders, ensures knowledge retention and it may even be required if the project is subject to certification. Thus, one of the expected outcomes of the pedagogical sequence is to provide the students with experience in using professional tools to organize their work.

The projects require that students spend a significant amount of time programming. All the team members have to update source codes in their shared project codebase. A common solution to manage the codebase consists in using a Version Control System (VCS). In our case, the students have to use Git [6], a popular VCS, in order to store their source code and manage revisions. Git is distributed so it does not need to be used in a client-server

5. See https ://www.raspberrypi.org/
6. See https ://git-scm.com/

FIGURE 2 – Car platform

model, and projects that do not require its distributed features can use an online hosting service as a Git server. Students are encouraged to use GitHub [7] as a Git server.

Agile methodologies introduce many new terms and concepts which may be hard to learn and use properly, even with a proper theoretical training. A tool with first-class support for Agile concepts and items helps students use them. The students must manage their projects in a dedicated iceScrum [8] server, an open source Agile project management tool. Students must enter all their User Stories in iceScrum, prioritize and estimate them in the Product Backlog, and plan them into Sprints according to their ability to produce stories in the previous sprints. To do so, they rely on indicators and charts computed by iceScrum from their project data. Students start their co-located work sessions by a stand-up meeting. They are encouraged to manage their Tasks in iceScrum but they are not required to enter them all in the tool.

6 Role of the Teachers

The project was run five times from 2012 to 2017. The roles of the teachers evolved over the years to better meet the pedagogical needs.

One difficulty was to define the role of the teachers in relation to specific

7. See https ://github.com/
8. See https ://www.icescrum.com/

students' activities. Indeed, the teachers have to take on different roles during the pedagogical sequence. They play the role of clients in Sprint 0, the reviews and the release. This role consists in defining the User Stories of the final product with the students and validating requirements. At weekly meetings, teachers take on a more traditional role as advisers : they check the students' work progress and help them choose efficient technical solutions to implement features. They also have to check that Agile concepts are well applied and that every student is involved in the work.

Finally, teachers also have to endorse the role of evaluators. Feedback is written collectively by all the teachers at the end of each review and emailed to each group of students before the retrospective of the next Sprint. The final evaluation aggregates all these intermediate feedbacks in relation to the Intended Learning Outcomes. In case of relational problems within a group, the teachers have to take on the role of facilitators and find solutions to ease the communication between students. This kind of situation has to be detected rapidly, otherwise it may slow down or thwart team work.

Another key condition for the smooth running of the project is for the teaching team to hold scientific skills for all the technical dimensions involved in the development of critical embedded systems, *e.g.*, hardware architecture, electronics, software programming, dependability, diagnosis, etc. Depending on their skills, teachers can be asked questions about specific technical problems which student face. Lastly, at the end of the project, all the teachers participate to a meeting to share their feedback on the project and to improve the next project session. Note that for feedback to be constructive, it is important that all the teachers take part in all the activities of the project (project organisation meeting, weekly meetings with students, Sprint reviews and final presentation).

In practice, the group of teachers consists of six to eight teachers. Experience shows that a twosome with different scientific skills is more effective. A teacher with expertise in Agile management also plays a central, cross-cutting role. This expert provides each group with guidance on good Agile practices, especially at the beginning of the project. This close and practical guidance complements very usefully lectures or MOOCs on Agile methods.

7 Results, feedbacks and Lessons learned

This multidisciplinary capstone design experience was assessed in different ways. First, a survey was done at the end of the project each year to collect remarks from the students and to evaluate the efficiency of the course.

Second, two surveys were conducted in the summer 2017 to assess the capstone design experience gains. One of the survey asked future students (4th year students) about their knowledge on agility and their expectations. This survey received 67 responses. Another one asked former students (since 2012) about their employment and further education, perceptions of the project, estimated gains in knowledge and skills, etc. There were about 70 responses to the survey. 80% of the interviewees now work as engineers. Finally, we also examined the marks given all along one year of project.

7.1 Gains

7.1.1 Estimation of immediate gains in knowledge and skills

At each review, the teaching team assessed each group according to the following criteria :

1. Is the sprint organization clearly presented ?
2. Are all the stories completed during the sprint presented ?
3. Are the acceptance tests well presented ?
4. Does the demonstration show every completed story ?
5. Is the schedule control explained ?
6. Are the stories of the next sprint presented ?
7. Are methods and technology on critical aspects highlighted ?
8. Are methods and technology on embedded aspects highlighted ?
9. Was the time allocated to the presentation respected ?

The assessment reports are sent to the students right after the review. Figure 3 shows the average mark evolution for all groups over the three reviews. It clearly shows that relevant and continuous feedback improves the results of the team. The evaluation criteria were the same from one review to the next. It is noticeable that over the three reviews, achievement on all the criteria improved. Moreover, we noticed that the Agile management approach was very useful in helping students improve their skills. One obvious reason is that it helps them control the project management better throughout the activity and understand the pedagogical objectives better. The feedback that the teaching team gives to the students at each sprint review allows them to better meet the expectations of the project while developing their skills.

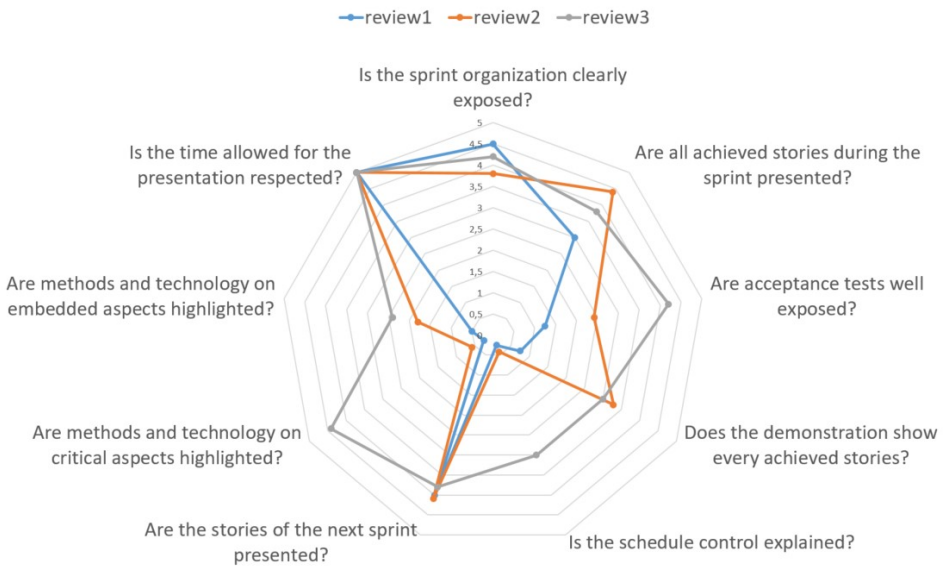

FIGURE 3 – Group assessment over 3 reviews

7.1.2 A valuable experience

From the students' point of view, the development of an end product with an Agile methodology is perceived as a real life experience valued by recruiters. The project is also a very positive point in their resume as well as in interviews : 65% of the students declare they have used in their first job interview the project to illustrate their skills.

Regarding our institution, the Open Day is an opportunity to show what students actually achieve in their final year of study. Visitors are invited to see the different projects and talk with students. The students display their project work and explain the science behind their projects : therefore, visitors have an insight into the study of critical embedded systems. Moreover, the platform itself has become a meaningful internal and external communication tool for the department.

7.1.3 Estimation of long term gains

It is difficult to assess precisely how much the students gained from the project. However, Figure 4 illustrates the answers to the question "Do you think that the project brought you useful skills and knowledge for your current work ?". More than 80% of the former students over the past 3 years

estimated that the project had brought them useful skills and knowledge. The clear improvement that can be observed after the first 2 years can be explained by the fact that we improved the integration between the technical work and the Agile methodology. In 2014, all the teachers involved in the project were trained to Agile method.

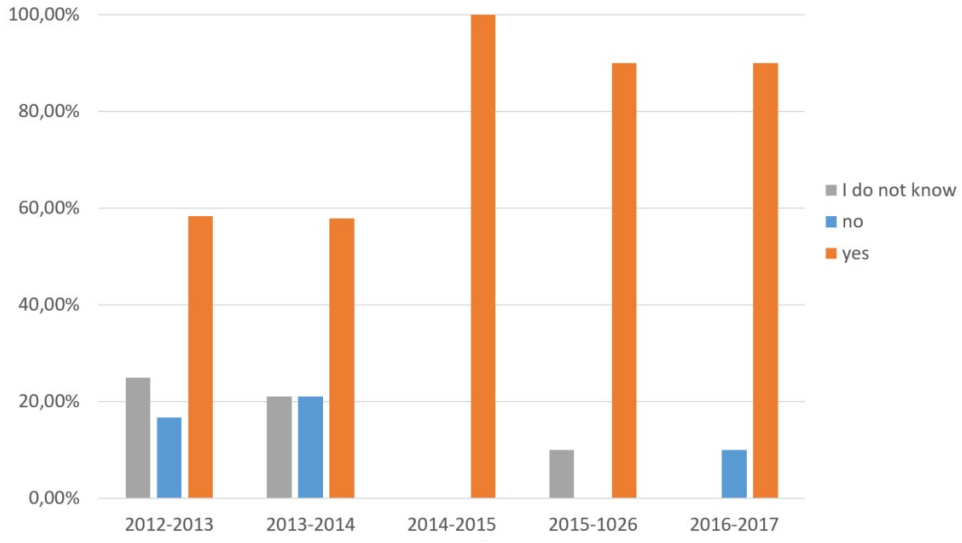

FIGURE 4 – Do you think that the project brought you useful skills and knowledge for your current work ?

Concerning the Agile skills, 58% of the young engineers said they used Agile methods at work. Figure 5 shows the answers to the question "Are the Agile methods used in the teams you work with ?". More than 60% of the former students over the 3 last years declare that they work with teams using agility.

7.2 Feedback

7.2.1 Industrial feedback on the Platforms

From an industrial perspective, both drones and cars are particularly relevant. Drones are now used by many industries like Total, Lafargue, Amazon... Self-driving and connected cars are now an emerging industrial object. Over the next five years, self-driving vehicles will be commonly used : Tesla Motors, Google, Renault, PSA, Uber, are currently investing huge amounts

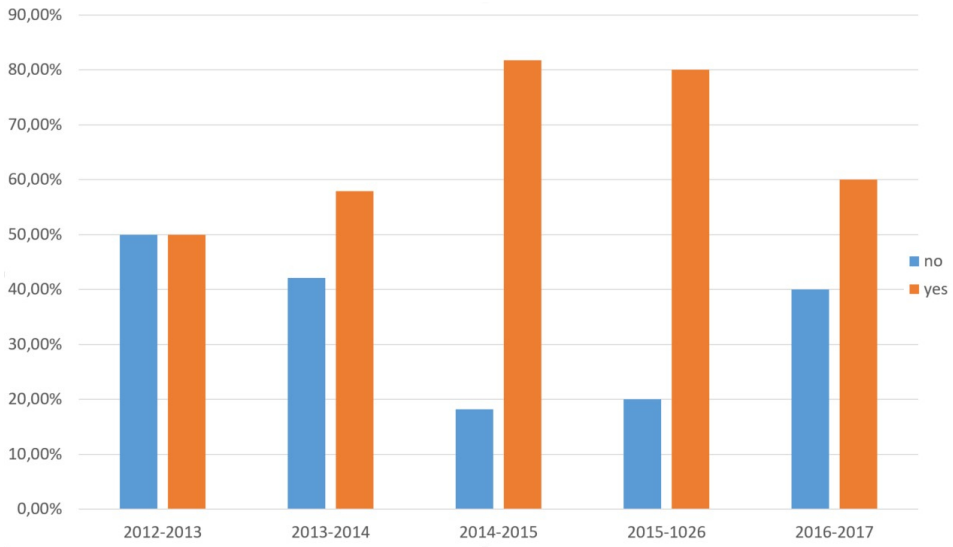

FIGURE 5 – Are the Agile methods used in the teams you work with ?

of money and effort to develop their own autonomous vehicles. This point is a real advantage, and our graduates receive excellent feedback on this in their job interviews.

Moreover, Figure 6 shows that more than 70% of the surveyed young engineers say that drones and cars are representative in the industrial context.

7.2.2 Project management outcomes

This project is the first time that our students have actually had to apply project management methods formally as part of their curriculum . Moreover, for more than 85% of them, it is the first time that they have had to use an Agile approach. The first difficulty for them is to adopt a formal project management approach.

That is why specific support is provided to each group early on in the project. The teacher helps students define the product ; describe and quantify a User Story ; formulate requirements and Definition of Done (DoD) ; organize a sprint ; use the management tool... From experience, this coaching is necessary if we want the students to follow the Agile process fully.

Another difficulty for students is to manage their time and to quantify the work they can achieve during a time lapse. Usually, teachers take care of this for students. *i.e.* when they design a pedagogical sequence, the quantity of

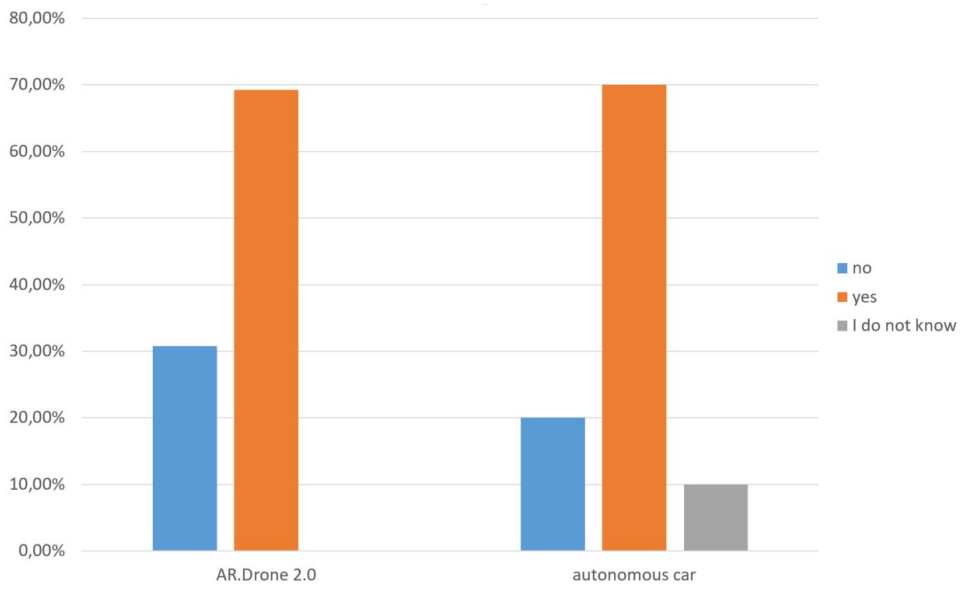

FIGURE 6 – Is the platform representative of a real industrial system ?

work matches the number of allocated hours. In the framework of the project, students have to define the number of User Stories and organize their time to complete them. The Agile approach offers an efficient way of acquiring this skill and it is fully integrated in the definition of a Sprint : a workload is evaluated for each User Story with Story Points, and assessed globally at each sprint retrospective. In practice, we note a real improvement in the way students define a Sprint and organize their work. However, it is necessary to schedule the project sessions in the timetable for them, for instance two half-days per week.

Concerning the project management aspects, 88% of students recognize that formal lectures on project management are useful for the project and 100% declare that they used the methodology and that it was efficient in driving their projects. However, 57% drop the methodology half way through the project, not because they do not like it but because they run out of time.

7.3 Lessons learned

7.3.1 Groups of students

In the first sessions of the project, we tested different student group sizes. Eventually, the best compromise seems to be a number of five or six students

per group. With a lower number, students give up project management because their work can be easily planned. Indeed, with a smaller group of students the number of User Stories of the final product is necessarily a low number (the quantity of expected User Stories is generally proportional to the number of students in a group) and so it is easier to organize. On the contrary, a higher number of students per group induces disengagement of students and increases the number of potential relational problems. In order to identify relational or personal investment problems among team members, we submitted a survey at mid-project to get individual feedback on the other team members (sort of peer-review). The survey was not mandatory, but it allowed us to spot some problems and to react quickly by meeting some students.

Teachers impose the composition of each group at the beginning of the project. There are no affinity groups and we pay attention to various criteria like gender, natural leadership, level of prior learning, etc. The aim is for students to learn how to work with new partners and manage a diversity of practices and behaviors. Moreover, we make sure that the teams include students from different majors and with various learning levels ; our students usually have an academic background in Computer Engineering or Automatic Control and Electronics. Another advantage is to cover a larger number of scientific disciplines in each group and show the diversity of skills involved in developing a critical embedded system. Note that students must attend the reviews of all the groups, which enables the groups to share technical knowledge widely and possibly, work together on a specific task.

7.3.2 Use of agile methods in PBL

The main interest of using Agile methods in a PBL process is to set up a quick feedback loop for both the aims of the project and the skills developed by the students. Thus students are in a continuous improvement process, *i.e.* each team must propose an integrated work from start to end and therefore remove early the technical and functional risks, and adjust the product if necessary, and they choose their own approach, *i.e.* each team identifies and resolves quickly the organization and collaboration problems.

The number of sprints depends on the total duration of the project. Two-week sprints seem to be a good compromise to allow quick returns and avoid drifts. It also helps to keep students in a continuous learning rhythm. However, it is important to be aware of the overall workload involved and therefore ensure that students assess this workload based on how much time they can devote to it. This point is often overlooked in traditional teaching and lear-

ning. Indeed, since pedagogical sequences are usually built by teachers, the constraint linking working time to student workload is (normally) already taken into account by teachers. The use of agile methods allows students to learn how to evaluate and quantify a workload.

7.3.3 Multidisciplinarity

One goal of this training is for students to work on a multidisciplinary project, which entails making sure that the final product covers a large set of technical domains. To reach this goal, we defined two mandatory technical domains (in relation to the curricula of students) : distributed embedded systems and dependable systems. Then the tutors had to check that each team's product integrated User Stories which did cover these two domains (the mandatory technical domains can be customized to adapt the project to other engineering domains).

The survey shows that students found it hard to put together their various fields of knowledge. In this case, the role of the tutor is essential to help students find a solution to a technical problem in their prior fields of knowledge, or investigate new technical approaches.

Since the project takes place in parallel with other courses, students cannot mobilize the specific skills of their specialized scientific field from the beginning, which is not really a problem. In practice, students start with a phase of specification and discovery of the tools and platform. They then enter a second phase where they need to mobilize specialized knowledge (operational reliability, diagnosis, verification, etc.). They can therefore apply their new skills to the project. However, this time gap between new learning and project start requires that the supervisors carefully guide students during the specification phase towards solutions that they will be able to address later.

7.3.4 Teachers

We have noticed over the years that the Agile approach needs to be understood and accepted by all the teachers involved in the project. Indeed, it can be difficult for engineering students to implement management practices, so it is essential that all the teachers be able to drive students to adopt these practices and guide them. That is why all the teachers who take part in the project need to be trained to Agile methodology and agree with using a PBL approach.

As mentioned before, it is not necessary for all the teachers to know about

all the technical domains covered by a project ; this gap can be bridged by working in a pair or clearly identifying the skills of the different teachers. Surveys show that students think teachers play an essential (more than 80%) and useful (88%) role.

Here is an estimate of a teacher's workload for one project. The Agile methodology training lasts about 12 hours (only once). Each teacher supervises 2 teams and meets them weekly for about 30 minutes all along the project, so the "short meetings" last about 12 hours overall. Additionally, all the teachers attend the review meetings that last 2 hours each. The "reviews" hours are then about 12 hours overall. The teachers also read the documents written by teams (project plan, web page, posters) and answer their technical questions all along the project. We estimate this time to 3 hours per teacher, plus at least 3 organization meetings (about 4 hours). A 'standard' teacher without any responsibility for platforms/tools or organization then spends 24 face-to-face hours and 7 hours without students, plus the initial agile training of 12 hours. Each year, the teaching load for the project is reevaluated depending on the number of students, but for example in 2017-2018 it was 12 hours per teacher. Additionally, 2 teachers are in charge of the platforms/tools and organization of the project. They invest about 1 additional hour per week during the project to organize it (planning, evaluation, feedback,...), communicate with students, solve technical/platform problems, liaise with the participants (Agile coach, English teacher, technical team, industrials,...) and are in charge of the improvement of the platform and the pedagogical process each year. The extra teaching load for these responsibilities is 5 hours per teacher.

8 Conclusion

In this paper, we presented a capstone design course based on the development of critical embedded systems with an Agile methodology ; the project was conducted for five years in a row at our Institute. We explained how the pedagogical sequence was designed to help students practice Agile management and develop a real product. A key point to succeed is to organize regular and well-scheduled sessions dedicated to Agile methodology with a coach. Moreover, supervisors need to be trained to Agile methods and endorse different roles such as client, technical expert, facilitator, etc.

The choice of the platform and tools also matters for the success of the project and student motivation. Our experience shows that it is better to develop one's own platform with open components and an active community

than using an integrated platform. The use of iceScrum, a tool dedicated to the agile management, also helps the students acquire agile methods and provides them with interesting feedback on their practice.

Based on this experience, capstone design projects appear as an essential component of engineering curricula as they increase the multidisciplinary skills of the students, and the Agile approach is particularly well-suited to managing and organizing a PBL activity.

Références

[1] ACM/IEEECS. Computer engineering curricula 2016 – curriculum guidelines for undergraduate degree programs in computer engineering. Technical report, ACM – IEEE Computer Society, 2016.

[2] C. Anslow and F. Maurer. An experience report at teaching a group based agile software development project course. In *Proc. of the ACM Technical Symposium on Computer Science Education (SIGCSE)*, 2015.

[3] A. Attard, E. D. Iorio, K. Geven, and R. Santa. Student-centred learning. toolkit for students, staff and higher education institutions. Technical report, Education International & ESU, 2010.

[4] K. Beck. *Extreme programming explained : embrace change.* addison-wesley professional, 2000.

[5] K. Beck, M. Beedle, A. van Bennekum, A. Cockburn, W. Cunningham, M. Fowler, J. Grenning, J. Highsmith, A. Hunt, R. Jeffries, J. Kern, B. Marick, R. C. Martin, S. Mellor, K. Schwaber, J. Sutherland, and D. Thomas. Manifesto for Agile Software Development. `http://agilemanifesto.org/`.

[6] E. Chenu. Agile & lean software development for avionic software. In *Proc. of Embedded Real Time Software and Systems (ERTS)*, 2012.

[7] B. C. Choi and A. W. Pak. Multidisciplinarity, interdisciplinarity and transdisciplinarity in health research, services, education and policy : Definitions, objectives, and evidence of effectiveness. *Clin. Invest. Med.*, 29(6), 2006.

[8] G. Clough. *Educating the Engineer of 2020 : Adapting Engineering Education to the New Century.* National Academy Press, 2005.

[9] O. Doss and T. Kelly. Challenges and opportunities in agile development in safety critical systems : A survey. *ACM SIGSOFT Software Engineering Notes*, 41(2), 2016.

[10] N. H. El-Khalili. Teaching agile software engineering using problem-based learning. *Int. J. Inf. Commun. Technol. Educ.*, 9(3), 2013.

[11] G. S. Goncalves, G. L. B. Lima, R. E. Maria, R. T. Wisnieski, M. V. M. dos Santos, M. A. Ferreira, A. C. da Silva, A. Olimpio, A. G. L. Otero, L. E. G. de Vasconcelos, L. Y. C. Sato, H. N. A. Silva, J. C. Marques, A. L. P. Mattei, A. M. da Cunha, L. A. V. Dias, and O. Saotome. An interdisciplinary academic project for spatial critical embedded system agile development. In *2015 IEEE/AIAA 34th Digital Avionics Systems Conference (DASC)*, 2015.

[12] B. Hyman, S. Khanna, Y. Lin, and J. Borgford-Parnell. A case study of using capstone design as basis for curriculum-wide project-based learning. In *ASME 2011 International Mechanical Engineering Congress and Exposition*, 2011.

[13] K. Könnölä, S. Suomi, T. Mäkilä, T. Jokela, V. Rantala, and T. Lehtonen. Agile methods in embedded system development. *J. Syst. Softw.*, 118(C) :134–150, 2016.

[14] M. Lehmann, P. Christensena, X. Du, and M. Thrane. Problem-oriented and project-based learning (popbl) as an innovative learning strategy for sustainable development in engineering education. *European Journal of Engineering Education*, 33(3), 2008.

[15] V. Mahnic. A capstone course on agile software development using scrum. *IEEE Transactions on Education*, 55(1), 2012.

[16] A. Mondragon-Torres. An agile embedded systems capstone course. In *Proc. of IEEE Frontiers in Education (FIE)*, 2013.

[17] D. Rover, C. Ullerich, R. Scheel, J. Wegter, and C. Whipple. Advantages of agile methodologies for software and product development in a capstone design project. In *Proc. of IEEE Frontiers in Education Conference (FIE)*, 2014.

[18] K. Schwaber and M. Beedle. *Agile software development with Scrum*, volume 1. Prentice Hall Upper Saddle River, 2002.

[19] T. C. Vidal, S. C. D. Santos, and R. S. Carvalho. PBL-tutor canvas : A tool based on backward design to plan pbl in computing education. In *IEEE Frontiers in Education Conference (FIE)*, 2016.

[20] D. Wood. Problem based learning. abc of learning and teaching in medicine. *British Medical Journal*, 326, 2003.

[21] C. Zhou, A. Kolmos, and J. F. D. Nielsen. A problem and project-based learning (pbl) approach to motivate group creativity in engineering education. *International Journal of Engineering Education*, 2012.