Evolution of an integrated course towards a sandwich course

Vincent Ribaud[1]*, Philippe Saliou*[1]

[1] *University of Brest, Computer Science Department, C.S. 93837, 29238 Brest, France*

Recently, local software companies in Brest asked for work placement students at Masters level. In 2007-2008, an innovative programme in software engineering was adapted to the work placement requirements. In this programme, students learn software engineering by doing, with a long-term project as the foundation of all apprenticeships. Apprenticeship periods are intertwined with the work placement periods. We present adaptations we made whilst keeping the programme objectives. The programme uses several hierarchical models that meet exactly at the two first levels: an activity model coming from the ISO/IEC 12207; an apprenticeship model; an ability model with 3 domains, 13 families, and 48 abilities together with 11 transversal competencies. At 4 key moments of the year, each student is asked to self-analyse the activities he/she did with regards to the immersion system's ability model and self-assess abilities on a scale from 1 to 5. Self-assessment averages of the 2006-2007 and the 2007-2008 cohorts are used to compare existing and adapted systems. Finally, we draft perspectives on the repercussions resulting from the work placement system.

Keywords

Learning by doing, self-assessment, software engineering, work placement.

1. Introduction

Sandwich courses were so named to describe the alternation of a study period in college and a training period in industry which characterizes them; in France a common arrangement is one week in college followed by three weeks in a training situation during the period of studying for the degree, but a variety of periods and sequences exist.

In spring 2007, local employers in Brest decided to implement a recent French law on professional training. This law requires that 3% of employees be under 'sandwich' (or work placement) conditions. Companies asking for work placement students in the software field choose to use a system called "Contrat de professionnalisation" (*professionalization contract*) over a period of 12 months. During these 12 months, the work placement student is a full-time employee, although also attending university for certain periods. Salary is about 80% of the salary corresponding to the job that the course leads to.

For such contracts involving Brest University, the employers' demand has been essentially for an innovative program called "Software Engineering by Immersion" (*'Ingénierie du Logiciel par Immersion').* The main feature of this last year of the Masters programme is to learn software engineering by doing, with a long-term project as the foundation of all apprenticeships. Due to demand from companies, this programme has been adapted and the academic year 2007-2008 is running as a work placement course.

This paper presents, in section 2, the main changes made to the programme structure; an assessment framework in section 3; some comparison elements in section 4; and ends with perspectives on the repercussions resulting from the work placement system, and a conclusion.

Post-print submitted to ACM-IFIP IEEIII 2008
Informatics Education Europe III Conference
Venice, Italy, December 4-5, 2008

2. Structure of the program

This section is intended to present how we adapted the program to the work placement system requirements in order to maintain its original objectives, structure and assessment.

2.1 Overview

Since 2002, Brest University has provided the software engineering by immersion paradigm as an alternative to other education systems. The immersion system is born from the necessity (due to the Bologna process) to design a fifth year for an existing and well tried 4-year technological education system. Since the 4-year system was sound and complete, this opportunity allows us to address the problem of educating software engineers from an unusual perspective for a university: to actually perform a significant software project - that is a sequence of stages organizing the activities intended to transform initial client requirements into a software product – always bearing in mind the goal of learning how to carry out these software engineering activities.

The year is divided into three periods, called iterations:

- a tutored apprenticeship period allows students to acquire software engineering knowledge and skills,
- an accompanied application period must transform knowledge and skills into competencies,
- and finally an internship period to put this into practice in a firm.

Learning is entirely based on a 7-month project, performed by a 6-student team within a virtual company, and tutored by an experienced software engineer. With the exception of English and communication courses, no lectures are given. The apprenticeship process was designed to be achieved in two iterations. During the first iteration (4.5 months), students are swapped around the different tasks required by engineering activities, and strongly guided by the tutor. During the second iteration (2 months), roles are fixed within each team and teams are relatively autonomous in completing the project, the tutor performing mainly a supervising and rescuing activity. During this second iteration, students rely on the apprenticeship process in order to autonomously perform a software development process supporting the production of the software product. An example of project is given in the figure 1 below.

Functions - The main goal of the project is to provide a semantic annotation tool able to annotate (indexing through metadata) Web resources, search (on metadata) in different modes, browse (hierarchically or with facets), manage RDF vocabularies (semantic schemas), and deal with the scope of annotations (public or private). The project uses Jena - http://jena.sourceforge.net/ an open-source Semantic Web programmers' toolkit - as RDF API.

Technical environment - The system uses a three-tier architecture in which the user interface, functional process logic, computer data storage and data access are developed and maintained as independent modules, on separate platforms. Sub-systems are: Oracle database, Hibernate persistent layer, Spring framework running on Tomcat, JSF for the user-layer.

Documentation - The tutor wrote the statement of work with expected needs. Main deliverables provided by the students are: Meeting report, Project Plan, Requirement Specification, Software Analysis, Software Design, Code, Integration and Validation Plan, Software User Manual, and Software Operator Manual.

Besides these engineering documents, students produce other kinds of document related to their apprenticeship: case study, usage guide, evaluation report, book or article summary, best practices, etc.

Figure 1 An example of a long-term project: a semantic annotation tool.

Whilst keeping the programme objectives, we adapted the system in the following way:

- the tutored apprenticeship period remains unchanged, but is alternated with the second period (two weeks each per month from September until mid-May),

- the accompanied application period is performed in the company, and may use a different development process than the process learned by doing during the first period,
- the last period is no longer a period of work placement, but rather a salaried period because the student is now a full employee of the company.

2.2 Reference framework

The immersion system uses a breakdown of apprenticeships into software engineering processes subdivided into software engineering activities, together with a set of apprenticeship scenes which provide the learning environment and defining tasks. This hierarchical process/activity/scenes model is adapted from the ISO/IEC 12207 [1] and is used as a reference framework for the learning objectives. Table 1 presents the two first levels of this hierarchical breakdown. From the university point of view, this division is the reference framework in a diploma-awarding perspective. Processes are course categories within the programme, activities are courses and scenes are classes.

From the 25 processes of ISO/IEC 12207, we concentrate on those related to the software development cycle, that is: 5.3 Development, 6.1 Documentation, 6.2 Configuration Management, 6.3 Quality Assurance, 6.4 Verification, 6.5 Validation, 7.1 Management, and 7.2 Infrastructure. The ISO/IEC 12207 Amendment 1 proposes a grouping of processes into categories. We reorganized the selected processes above in a same manner. The main process considered is the Development process. The other considered processes are not as wide as the development process, so these are retrograded to activities and reorganized in two processes: Project Management and Development Support.

In the ISO/IEC 12207, the Development process consists of the following activities : System requirements analysis and system design; Software requirements analysis; Software design; Software construction; Software integration; Software qualification testing; System integration and qualification testing. Our breakdown differs slightly because we do not need system activities (only software) and we put emphasis on computing constraints, essentially Technical Architecture.

Table 1 Activities breakdown.

Processes	**Activities**
Software project management	Project management Quality insurance Software configuration management
Software development engineering	Requirements capture Software analysis Technical architecture Software design Software construction Integration and validation
Software development support	Technical support Methods and tools support Documentation Installation and deployment

2.3 Assessment

2.3.1 Evaluation of industrial production processes

The process approach (as advocated in the ISO 9001:2000 standard) allows a company to describe its organization in order to produce defined results. The more visible processes are

those undertaken to perform services or produce products that the company provides, which are often referred to as operational processes. In the software area, the supply activities chain provides deliverables (mainly documents) at each intermediary step of the development cycle. An important part of the quality management system deals with the evaluation of delivered documents, supported by two kinds of evaluations: those which inspect products during their elaboration and those which verify and validate (V&V) requirements.

2.3.2 Authentic assessment

The immersion system belongs to the constructivism approach and Tardif [9] states that the impacts of constructivism on assessment are numerous. In accordance with teaching practices, assessment necessarily relies on complete, complex and meaningful tasks. The assessment has to reflect effective, individual achievement of learning outcomes. The assessment must also take into account - directly or indirectly - cognitive strategies used by the learner. The constructivism paradigm implies that the teacher directly and frequently intervenes in the knowledge organization and hierarchy construction performed by learners. The role of the assessment is to report on the state of knowledge building. Assessments should take place in a context that is familiar to the student, using standards that are well known and presented in multiple forms [9].

2.3.3 Two kinds of activities in the immersion system

Samurçay and Rabardel [7] distinguish two faces in human activities. The productive activity is an activity made, oriented and controlled by the human subject to perform tasks he/she has to achieve with regards to the situation features. The constructive activity is oriented and controlled by the subject that performs it in order to build and develop competencies with regards to the situation and professional areas of action.

A formally organized education system has to make a distinction between a support activity which is a productive activity with learning objectives, and a constructive activity of competency development which happens on the occasion of the productive activity (with learning objectives) [5]. Our learning process organizes situation-problems present in the work activity in order to schedule productive activities (with learning objectives) and to control constructive activities that happen on these occasions.

2.3.4 Two kinds of assessment in the immersion system

Distinguishing two kinds of activities lead us to distinguish two kinds of assessment.

The immersion system attempts to mimic an operational environment, and productive activities (with learning objectives) have to be assessed with assessment procedures mimicking industrial usages. Hence, a first type, called Verification and Validation assessment, consists of evaluations based on the review of apprenticeship stages and on the qualification of software products. These evaluations are Verification and Validation (V&V) activities, performed in line with a predefined schedule.

However, most activities (especially during the first iteration) are constructive and need an (authentic) assessment that is fully integrated with the learning process. Thus, the second type is constituted by the tutor/author feedback cycle, the weekly progress meeting and peer reviews. These activities directly sustain knowledge building because they provide continuous feedback to learners. We call it regulation assessment referring to De Ketele “[…] an open process whose priority function is to improve the working order […] of a part or of the whole system” [2]. These evaluations are performed continuously and there is no firm separation between assessment and apprenticeship processes. Assessment sustains learning, giving useful information to students about the evolution of their learning process, and making them aware of knowledge they have, or are lacking.

2.4 Without work placements

2.4.1 Structure

Until 2007, the year was divided into three iterations: a 4.5-month tutored apprenticeship period, a 2-month accompanied application period and a training period of 4-6 months in a firm (see table 2). The acronym UE (*Unité d'Enseignement*) means Course Unit.

Table 2 Structure of the previous course.

Iteration	Period	Content
Tutored apprenticeship	From September to January	UE1 : Software project management UE2 : Software development engineering UE3 : Software development support UE4 : Communication and English language
Accompanied application	From February to March	UE5 : Putting into practice the knowledge, skills and competencies acquired during the first period
Internship	From April to August	UE6 : Performing an operational mission

The guideline for the two first iterations is the software development project, in which students will be immersed. Each company is associated with a company tutor, who plays different roles in the first and second iteration. The objective of the project entrusted to each company is to manufacture an information system, which responds to a real need of the computer science department (see figure 1 for an overview of a past project).

2.4.2 Objectives

The training course starts after the response to solicitation phase. The role-play partly consists of simulating the client-supplier relationship. The company tutor has to write requirements, then a response to solicitation including a technical and commercial offer responding to the anticipated requirements. These documents are very similar to real documents. They are only different in order to incorporate needs induced by our apprenticeship system:

- Two work packages are expected in order to map the two first iterations.
- Lead-time and cost are adapted to the size of teams and the time available for the training course (4.5 then 2 months).
- Technical constraints imposed by the client correspond to the objectives and means of the course.

During the first tutored apprenticeship iteration (4.5 months), an incomplete version of the software is built within a framework entirely driven and tutored by the company tutor. All software engineering activities are put into practice within a complete software development cycle. The expected software product at the end of this iteration corresponds to the first work package as defined in the requirements document. Our assessment process fits exactly into the apprenticeship situations. Students are continuously building knowledge and developing skills. Tutors regularly observe what is happening in order to provide a sound feedback and to perform authentic assessment.

During the second accompanied application iteration (2 months), each company is relatively autonomous in putting into practice the knowledge, skills and competencies acquired during the first iteration. A set organisation of the team is set up for the second iteration, structured around roles: project manager, analyst, architect, configuration and version manager,

developer, integrator and qualifier. The expected software product at this end of this iteration corresponds to the whole software product as defined in the requirements document. During this iteration, the company is still accompanied by the company tutor, whose role is mainly to supervise and rescue.

Finally, the third iteration, i.e. the internship period, should ensure the continuity of the training course. Inside a firm, a software project (from A to Z) could be entrusted to the student, in line with the development process learned during his/her education. Practicing of software engineering activities is an alternative to the training period - for example, managing software configuration, capturing users' needs, analysis or design modelling, etc.

2.4.3 Assessment

Each iteration has its own pedagogical goals, and hence its own assessment characteristics. Throughout the first iteration, evaluations are regulation assessments (belonging to the second type defined in § 2.3.4) intended to support knowledge and skill-building whilst providing continuous feedback to learners. Each apprenticeship scene gives rise to one or several deliverables. Each deliverable is carefully examined and annotated by tutors, then the tutor feeds back comments to the authors together with improvements to be brought about. This assessment and feedback process is iterated (at least twice) until that the deliverable is considered good enough for future exploitation (problems could arise if the final delivery is not judged to be good enough). Each apprenticeship scene is assessed and awarded a mark. Because each scene is related to a software activity belonging to a software process, marks are consolidated in order to provide an average assessment for each activity/process.

The first iteration ends with the delivery of the information system corresponding to the first work package, qualified according to the validation protocol which was elaborated by the project team. This qualification is a V&V activity (belonging to the first type defined in § 2.3.4). Success of this activity (and hence of the evaluation) is related to the quality of the system produced, and is no longer tied to the completion of the learning process (it is still a formative experience because the tutor is giving feedback during and after the V&V activity).

The objective of the second iteration is to put into practice the knowledge and skills acquired during the first iteration. This second period always takes place over a period of 8 weeks. Thus, students spontaneously shift in terms of production rhythm. Assessment is performed on achieved deliverables according to production criteria: compliance and schedule. We did not have enough time to measure individual competencies and performances. We gave three marks for this iteration: one for the project itself (and the work carried out), one for the group viva voce examination, and one for individual reports on personal and group work. The mark given for the project relies on assessments of the essential deliverables of a software project, and is the result of a V&V activity.

Internship periods, as the third iteration, are assessed in our department, and awarded three marks: one for performance at work, established by the industrial tutor using a questionnaire provided by the faculty, one for an individual viva voce examination (with the participation of the industrial tutors), and one for an individual report on the work carried out during the internship.

2.5 With alternation

2.5.1 Structure

The principle of a program with work placements is that the educational objectives and assessment of course units are nearly the same as those of a program without work placements, but that some of the course units can be performed in a different way or during the periods in industry. In our case, two course units, UE5 Accompanied application and UE6 Internship were good candidates to be performed during the industrial periods.

In the previous system, the UE5 Accompanied application was intended to autonomously perform software engineering activities under the supervision of a faculty tutor. In the work placement system, the objectives are the same - but now it happens in a firm under the guidance of an industrial tutor. Assessment is still performed by the faculty under the rules of an internship period.

The UE6 Internship was designed to be an operational mission, and played the role of a pre-hiring period. In the new system, students are salaried employees and de facto in an operational position. Hence, the objectives are stuck to, and assessment performed as before.

The remaining course units have to be intertwined with the industrial periods. We choose a two weeks / two weeks rhythm from September to mid-May.

The year is now divided into two periods, the former with movement between university and company, the latter with a full-time period at the company (see table 3).

Table 3 Structure of the new course.

Iteration	Period	Content
Tutored apprenticeship	Half-time from September to mid-May (9 * 2 weeks)	UE1 : Software project management UE2 : Software development engineering UE3 : Software development support UE4 : Communication and English language
Work placement period in company	Half-time from September to mid-May (8 * 2 weeks)	UE5 : To perform software engineering activities in a real situation
Full-time period in company	From mid-May to September	UE6 : To perform an operational and salaried mission

The structure remains identical to the previous one, except in that the Course Unit 5 is renamed “Work placement period in company” and Course Unit 6 is renamed “Full-time period in company”. For the two modified course units, ECTS are identical, objectives remain the same and assessment differs only slightly, as explained above and below.

Unfortunately, first and second iterations are performed on different projects. The former is an apprenticeship project driven by the university and the latter is an industrial project driven by the companies with whom students are placed. However, the same reference framework (see table 1) and a unique abilities assessment framework (see section 3) are used throughout the year, providing students with a link between apprenticeship and work experiences.

2.5.2 Objectives and assessment

As before, the guideline for the first iteration is the software development project, in which students will be immersed. But most of the objectives from the two previous iterations are now assigned to the new, single one. That means, for one thing, that the shift from apprenticeship to production must be made during this iteration, whereas previously the first iteration was focused on apprenticeship and the second on production. Hopefully, students are maturing in parallel, thanks to the work placement periods, and are naturally shifting towards a professional attitude.

As before, the training course starts after the response to solicitation phase. Reference documents are project requirements and a response to solicitation. A unique work package is

expected, and lead-time and cost are adapted to temporal organization of the training course (4.5 months over a period of 8.5 months).

During this unique apprenticeship/production iteration, the whole software product as defined in the requirements document is built within a framework entirely driven by the company tutor. But in contrast with the previous system, the tutor has to gradually reduce the help on offer to the student. During the first phases of the development cycle, products are assessed at the moment they are delivered, then feedback and corrective measures are provided by the tutor. In the latter phases, less feedback and help (or none at all) are given to students, and they have to do it by themselves. The tutor's main tasks gradually become broad supervision, assessment and rescuing if needed. Assessment shifts from being regulation evaluation to being V&V evaluation.

During the work placement period (4 months over a period of 8.5 months), each student is autonomous so as to put into practice (in his/her industrial position) the knowledge, skills and competencies that he/she is currently acquiring during the first iteration. It could happen that skills required by the industrial missions are learned later in the academic cycle and students have to resolve this deficiency by themselves. Assessment is still performed by the faculty under the rules of an internship period: one mark is awarded for performance at work and established by the industrial tutor, one mark for an individual viva voce examination, and one mark given for an individual report on the work done during the intertwined work placement periods. The mark given by the industrial tutor assesses the student's activity and is mostly measured from the produced deliverables - and is therefore the result of a V&V activity - as before.

For the viva voce examination and the individual report, the student is asked to establish clear links between the work carried out and the apprenticeship reference framework. A kind of reflection-on-action [8] is still required in order to show examiners how they have matured in the course of their work placement. In order to facilitate this difficult exercise, a half-day is devoted each month to a group meeting at which each student presents an account of what happened at work, how and why he/she acted as he/she did, to his/her peers (and tutors). Thus, an attempt at assessing meta-cognitive abilities is introduced - but it needs improvement.

The third iteration (4 months) is no longer an internship period. Students are no longer new employees; they are fully integrated within their companies and are paid the going rate. Companies both see and use them as full employees. Assessment is performed jointly by the faculty and industrial managers (work, report, oral) but using industry expectations.

3. Abilities assessment framework

3.1 Linking processes and activities to apprenticeship situations

Each activity represented in table 1 can be analysed from various points of view: expected knowledge and skills; stakeholder roles; input and output deliverables; required tools and resources, etc. At work, what makes sense for these multiple viewpoints is their articulation within the activity situation (the cohesion of the work situation). During the apprenticeship, it is also the (apprenticeship) situation which has enabled the multi-dimensional nature of activity to be understood. The (apprenticeship) situation has to transform students into learning humans, allowing them to put their own skills to work; the task to be performed allows the learning to take place. Situations are not natural: they have to be provided by the education system [4]. Putting students in a learning position - the situation - is the vital lead of educational design and practice. We distinguish the conceptual situation from the lived situation. A conceptual situation (called apprenticeship situation) is a situation-problem that "has to place fundamentals acquisition in actions that provide a goal to the students" [4]. An

apprenticeship scene is the materialization - in action - of the conceptual apprenticeship situation. The system by immersion is a theatre play where, during a scene, different actors play different professional roles in order to learn professional [software engineering] activities.

There are two reference decompositions that meet exactly for the two first levels; the former is an activity model coming from the ISO/IEC 12207: process, activity, task; the latter is an apprenticeship model: process, activity, apprenticeship situation (scenes).

3.2 Linking processes and activities to abilities (competencies)

Meirieu [3] defines a competency as "the ability of a person to act in a pertinent way in a given situation in order to achieve specific purposes". Competencies are means of action which a person has to perform his/her activity with regards to the situation he/she has to deal with. The immersion system aims to acquire professional competencies that we prefer to call abilities. We carefully analysed the entire apprenticeship scenes for each activity in order to establish the abilities that theses scenes are intended to develop. We tried to answer to the question "what is the student able to do, once the scene has been performed?". This analysis gave us a set of abilities for each activity.

So we kept the 2-level breakdown of our reference framework, the first level being called competency areas (corresponding to processes) and the second level competency families (corresponding to activities), and we placed knowledge and abilities within each family (see an example in table 5). This breakdown contains 3 domains, 13 families, and 48 abilities together with 11 transversal competencies. We call the whole breakdown an ability model of our education system [6].

Table 5 An example of a competency family: "Software project management".

Knowledge area	Associated abilities or skills
• Software life-cycle model • Estimation and follow-up of development of software components: organization, workflow … • Traceability and requirements conformity • Project Plan	To use an ISO 9001 development baseline
	To apply a Project Plan and to update it if necessary
	Planning and project progress

3.3 Auto-assessment of abilities

The structure and definitions of this ability model are recorded in a document called the competencies compendium. Applied to the software engineering apprenticeship field, our ability model establishes a structure that directly supports the personal and team construction process of the knowledge and skills required to practice engineering of a software project. For each ability or transverse competency, the student assesses himself/herself at a maturity level. The assessment scale grows from 1 to 5; - 1 - Smog: vague idea (or even no idea at all); - 2 - Notion: has a notion, a general idea but insufficient to an operational undertaken; - 3 - User: is able to perform the ability with the help of an experienced colleague and has a first experience of its achievement; - 4 - Autonomous: is able to work autonomously; - 5 - Expert: is able to act as an expert to modify, enrich or develop the ability.

It is not easy for a student plunged into the 'doing' to keep in mind the abilities aimed at by the apprenticeship scene (and by the education system) and to establish links between all

kinds of learning. That is the reason why, at four key moments of the year, each student is asked to self-analyse the activities he/she did with regards to the immersion system's ability model. So, four times during the year, students have to establish this inventory and communicate it to their tutor. We call this periodic inventory the Personal Follow-up of Competencies (PFoC) and it is, among others, intended to initialize a personal follow-up of competencies that could be pursued in a professional career.

4. First elements of comparison

4.1 Quantitative approach

It is not easy to measure the efficiency of an education system, and the impacts of evolutions within an existing system. In order to compare the system with work placements from the previous one, an indication can be drawn from the personal follow-up of competencies. For the 13 competency families, Table 6 presents the self-assessment average of the 2006-2007 cohort and the 2007-2008 cohort. Each cohort has 12 students.

Table 6 Personal follow-up of technical competencies for the 2006-2007 and 2007-2008 cohorts

		2006-2007 cohort			**2007-2008 cohort**		
Competency area	**Competency family**	**Sept. 2006**	**Feb. 2007**	**May 2007**	**Sept. 2007**	**Feb. 2008**	**May 2008**
Software project management	Project management	1.5	2.8	3.4	1.3	2.7	2.9
	Quality insurance	1.1	2.4	2.8	1.4	2.3	2.4
	Configuration management	1.2	1.8	2.9	1.6	2.9	3.0
Software development engineering	Requirements capture	2.1	3.2	3.6	1.8	2.8	3.0
	Software analysis	3.6	3.7	3.9	2.4	3.0	3.3
	Technical architecture	1.4	2.4	3.0	2.0	2.8	2.9
	Software design	2.8	3.2	3.5	2.3	3.1	3.6
	Software construction	2.7	2.7	3.1	2.5	2.9	3.4
	Integration and validation	1.2	1.3	2.7	1.3	2.0	3.2
Software development support	Technical support	2.3	3.0	3.4	2.4	3.1	3.5
	Methods and tools support	1.7	2.6	3.2	2.0	2.5	2.9
	Documentation	2.8	3.3	3.5	3.1	3.3	3.7
	Installation and deployment	2.4	3.3	3.5	2.9	3.3	3.7

All families follow a regular and roughly equivalent progress, with or without work placements. Due to the low number of students in cohorts, and the paucity of our statistical knowledge, no statistical comparison was performed. Yet some small differences could be pointed out . We remark that the final levels are slightly lower in the new structure than in the old. That is probably due to the loss of the second iteration in the new structure, because the breaking down into two iterations provided a learning process that was easier to follow. In the new structure, the intertwined iterations are, for some students, experienced as two intertwined learning processes which may be in conflict, slowing down the overall learning process. But the new structure achieves other goals, and is an answer to external pressure from local employers that could not be ignored.

As a point of detail, both cohorts established their February compendium before having performed the validation phase, and that should explain the low level of progress in the

"Integration and validation" family. But as you can see, the 2007-2008 cohort is much more aware of this activity, probably due to exposure during the placement periods.

For the 11 transversal competencies, Table 7 presents the self-assessment average of the 2006-2007 cohort and the 2007-2008 cohort.

Table 7 Personal follow-up of transversal competencies for the 2006-2007 and 2007-2008 cohorts

	2006-2007 cohort			2007-2008 cohort		
Transverse competency	**Sept. 2006**	**Feb. 2007**	**May 2007**	**Sept. 2007**	**Feb. 2008**	**May 2008**
Organization	2.8	3.2	3.6	2.3	2.9	3.3
Cooperation	3.5	3.8	4.0	2.5	2.9	3.7
Communication	3.0	3.5	3.6	2.5	3.2	3.7
Transfer / Sharing	3.0	3.5	3.7	2.6	3.3	3.5
Analysis	3.1	3.6	3.7	2.5	2.9	3.4
Abstraction	3.1	3.3	3.5	2.4	3.1	3.3
Cleverness	2.9	3.1	3.4	2.8	3.0	3.1
Innovation	3.1	3.3	3.7	2.8	3.2	3.3
Listening / Flexibility	3.3	3.7	3.8	2.8	3.2	3.5
Adaptability	3.2	3.8	3.9	2.6	3.0	3.5
Reflection	2.8	3.2	3.6	2.3	2.4	3.5

Progress is nearly the same, depending on the initial assessment. As for technical competencies, on average these are lower in the new structure than they were in the older one. But industrial experience is higher, which improves students' employability.

This year, reflection activities were performed on the industrial periods rather on the university periods and we delayed it until January. This could explain the slow start of progress on this transverse competency for the 2007-2008 cohort.

4.2 Students' remarks and expectations

Missions entrusted to students during industrial periods could differ significantly from those that the programme prepares for. Only six students out of twelve have to perform a project which corresponds to the skills learned from A to Z. Four out of twelve students perform maintenance activities for which only a part of the acquired skills apply, as well as some essential skills which are lacking. And two students have to perform multiple tasks and missions for which the skills needed do not exactly fit with those of the programme.

So, it is not surprising that several students pointed out that there was little application of the education in the industrial environment, or that there was a gap or a shift between required skills in the industry and learned skills at university. Some students were afraid that the time spent in the industry reduces the amount of skills that can be acquired in the program.

Obviously, almost all students expected a significant gain in terms of experience and a subsequent payment from the placement system. Most of them were attracted by the idea of long-term work with regard to shorter periods of work placement performed before. One student pointed out that industrial periods raise the desire to evolve within a company – an idea that had never occurred to him before.

4.3 Tutors' remarks

We designed the immersion program in 2002 with a year divided into three periods, known as iterations: a 4.5-month tutored apprenticeship period, a 2-month accompanied application period and an internship period of 4-6 months in a firm. Objectives, achievement and assessment were different for each period, but were aligned with Donald Schön's idea of the reflective practitioner perspective [8].

It was at the request of local employers that the changes to the immersion system were instigated. The major difficulties encountered were in increasing the time spent on placement (and therefore decreasing time spent at university) and in intertwining the periods.

The second iteration (accompanied application) was a good candidate for taking place in a firm rather than at the university. Because the main objective is to acquire autonomy as a software engineer, and thanks to an assessment that is production-criteria oriented and based on achieved deliverables, there is no betrayal of the iteration in transferring it to a company. Unfortunately, intertwining apprenticeship and operational periods may cause additional difficulties for the apprentice - it all depends on what happens during periods in the firm, which is no longer under the university's control.

The third iteration (formerly an internship) is now an operational period but the goal - production – remains, and assessment remains roughly the same - although the performance level expected by employers may be higher.

The curriculum of our programme is based on a software engineering profession reference model, and it helped us transfer a part of the educational objectives (and related ECTS) to periods spent in a firm. However, the adaptation we made to our programme may be much more difficult to achieve with programmes based on a knowledge-oriented curriculum, and we are very conscious of the limits of our approach. The next session is an attempt to identify lessons that might be applicable in our situation, as well as to situations others may find themselves in.

5. Perspectives: repercussions of work placement

Adapting the immersion system to a work placement system produced and will continue to produce several repercussions, some on the system itself, and some on our university department. We tried to deal with this as well as possible, but we must engage in a long-term thinking process about it. We will try, in this section, to give some elements of these repercussions along the engineering perspective: strategic engineering, training engineering, and educational engineering.

5.1 Strategic engineering

There can be few links between a company hiring fresh graduates and the educational establishment from which these young employees graduated. But in a sandwich course, companies are paying students during the studying time and are paying the educational establishment for the education provided. So, the relationship between companies and educational establishments is quite different. There is a real need to build a veritable partnership with employers in order to, at least:

- Provide young graduates with a minimum level of professional experience and the skills required (technical and non-technical), making them rapidly adaptable and operational.
- Prepare students for the recruitment process.
- Tailor a single work placement programme to the context of each company

5.2 Training engineering

Several programs are candidates for adaptation to work placement systems. In our university, for example, there are work placement students in the science faculty (mechanics, electronics, computing, etc.), in the business faculty (management, bank-insurance, etc.), in the social science faculty (human resources, disability management, etc.) and so on. An educational establishment has to engage itself in a global strategy related to training engineering, putting emphasis on, at least:

- Promotion of a competency approach in teaching, learning and assessment
- Building of new degrees or adaptation of existing ones to the real needs of companies
- Encouragement of faculties, departments to develop work placement programmes
- Assessment of existing and new programmes

5.3 Educational engineering

The work placement system leads educational staff to re-think their teaching methods, re-organize programmes, and set up new modes of intervention. At the very least, we must pay attention to:

- Anchoring each student's individual learning path with his/her industrial experience and exploiting these experiences for educational purposes.
- Accompanying each student in the construction of his/her professional project.
- Favouring those periods outside the university by providing students with new skills gained through innovative teaching practices.

6. Conclusions

We presented adaptations we made to an existing 'learning software engineering by doing' programme in order to transform it into a work placement programme. The structure and the objectives remain identical to the previous course, and assessment differs only slightly.

Self-assessment of competencies of the 2006-2007 and the 2007-2008 cohorts were used to compare existing and adapted systems. No major differences were found. Students and employers are satisfied overall, but it is vital that we engage in a long-term thinking process about the repercussions of work placement, in terms of several engineering perspectives.

References

1 ISO/IEC 12207:1995, Information technology -- Software life cycle processes, International Organization for Standardization (ISO), Geneva, Switzerland.

2 De Ketele J.M., Roegiers X. Méthodologie du recueil d'informations. Bruxelles, De Boeck, 1993.

3 Meirieu P. Si la compétence n'existait pas, il faudrait l'inventer In IUFM de Paris Collège des CPE, 2005, http://cpe.paris.iufm.fr/spip.php?article1150 (last accessed May 20th, 2007).

4 Morandi F. Pratiques et logiques en pédagogie, Paris, Nathan, 2002.

5 Pastré P. Introduction In Recherche en didactique professionnelle, edited by Samurçay, R. and Rabardel, P., Toulouse (France), Octarès, 2004.

6 Ribaud V., Saliou P. Towards an ability model for software engineering apprenticeship. Italics, July 2007.

7 Samurçay R., Rabardel P. Work competencies: some reflections for a constructivist theoretical framework In Proceedings 2nd Work Process Knowledge Meeting: Theoretical approaches of competences at work, Courcelle sur Yvette (France), 1995.

8 Schön D. The reflective practitioner, New York, Basic Books, 1983.

9 Tardif J. L'évaluation dans le paradigme constructiviste In L'évaluation des apprentissages. Réflexions, nouvelles tendances et formation, Sherbrooke (Canada), Université de Sherbrooke. 1993.