TEACHING IN THE TECH-LAB USING THE SOFTWARE FACTORY METHOD*

Alejandro Bia[1], Ramón P. Ñeco[2]

[1]Centro de Investigación Operativa, Universidad Miguel Hernández
[2]Depto. de Ingeniería de Sistemas y Automática, Universidad Miguel Hernández
abia@umh.es, ramon.neco@umh.es

Abstract

The Software Factory method (SFM) meant to take some actions and to apply software tools, with two purposes: (1) to make practical sessions of technology subjects to resemble as much as possible the actual work in business or industry and (2) to promote the use of Web and portable tools that allow new flexible and dynamic forms of work, such as teleworking, mobile working and remote collaborative work. This article describes the experiences of the authors in this regard.

Keywords: Innovation, technology, research projects, etc. [Arial 10-point, justified alignment].

1 INTRODUCTION

The methodology proposed in this paper aims to simulate in the classroom, as realistically as possible, the working conditions of a software development company carrying out projects in a modern and well defined environment. The goal is to prepare students for job placement in the real tech/business world. As additional objectives, we propose the integration of skills and knowledge of various subjects and promoting teamwork. We call this method Software Factory, in honor of those initiatives that emerged in the '70s and '80s to systematize the production of software, as if it were an industrial assembly line [3].

Although it has been created for the practical sessions of Software Engineering and Project Management courses, this methodology is applicable to the practical teaching of other subjects, primarily of technological or business type. For instance, in the academic year 2011-2012 it has been successfully applied to Industrial Automation in Mechanical Engineering.

2 RELATED WORK

There are some previous works that describe experiences with methods related to described in this article. For example, in [1] students pretend to be working in an IT company, working in teams each of which must perform different real projects. There are also numerous papers related to teaching in groups. In [6], the benefits of student group work from the point of view of the relationships and interdependencies that are established are analyzed, based on practical experiences of the authors. In [2], the skills students learn through real projects in groups are described. More recently, Jones and Issroff [5] conducted a study of social skills and attitude obtained with teamwork, in a similar way as used in the method proposed in this article.

3 DETAILED OBJECTIVES

- The overall objective is to simulate in practical classes, as realistically as possible, the working conditions of real company projects.

- To prepare students for integration into working life.

- To capture requirements simulating real business situations: the use of artificial classroom problem-statements will be avoided (employers and customers in real life do not explain their requirements by means of written statements). Students will "capture" requirements for the definition of their classroom

* This work was partially funded by the University Miguel Hernandez: grant PN1238-ILZU02 of the 2011 *Call for Research Projects in Humanities and Social Sciences*.

assignments. The lecturer will play the role of the customer for such purpose. Real companies can be approached when possible.

- Assignment of roles: role-playing where the teacher/trainer acts as a customer, boss or employer, and students, organized in development teams, assume different project roles and responsibilities which should be clearly defined using work breakdown structures (WBS).

- Use of technical inspections: the development of the classroom projects is controlled by regularly scheduled inspections performed by the teacher/trainer, targeted to evaluate expected results and milestone accomplishments of the assigned projects.

- Actual software implementation or functional prototypes are expected (not just theoretical developments or non-functional prototypes).

- Integration of skills and knowledge from several subjects: the proposed project topics should involve knowledge of several subjects (as in real life), avoiding being too focused on the subject-matter of the actual course the assignment belongs to.

- To foster teamwork, and the development of a "business culture": The notion of mutual responsibility as a need for the success of a project, and the development of intercommunication and coordination skills should be specifically addressed by the teacher. These skills are within the most important teaching goals of the method.

- Dedicated development platform for each project team: for the duration of the practice (usually several months, and maybe the whole semester), each project team, will be responsible for maintaining the hardware and software used in their project assignments. They must assemble, install and maintain their working platform (intentionally without the aid of the lab technical assistants). If the hardware available is not enough to be allocated to project teams in dedicated mode, virtual machines can do the trick.

- To promote teleworking and colaborative work: the use of communication tools for remote working, collaborative environment for teamwork, and social networking applied to lab projects will be encouraged.

4 DESCRIPTION OF THE PROPOSED METHOD

The traditional method of teaching computer problem solving in class, consist of a statement which describes the state of a fictitious company. By means of this statement, the requirements of the problem are presented.

In real companies, computer technicians are not given a written statement, but instead they must collect the details of the problem to solve on their own, with much ingenuity and effort. This task of software engineering is called requirements capturing, which is a crucial phase in the software design process. In fact, the term requirements engineering is very common, to the point that there are many books under that title. In the SF method, we propose not to provide written assignment statements, but instead, the teacher will act as the employer or client and be asked for the requirements verbally, as it would be done in real practice. Then it is up to the development team (group of students) to apply techniques to verify requirements capture, refine and complete the list of requirements of the problem to solve, and develop a correct specification of the solution.

The following sections describe the features of the proposed method.

4.1 TEAMWORK AND ROLE PLAYING

Unlike the development of small programs, in medium and large software development projects there is a need for teamwork. When working in teams, communication, understanding, coordination, collaboration, and interpersonal relationship problems arise. Therefore, in the software factory method teamwork is encouraged, so that all these communication and personal interaction problems are experienced. Moreover, in real companies computer scientists have to work in different projects with different teammates. For this reason, the proposed method also promotes the realization of different practical assignments with different teammates.

The study of the reasons why some groups work very well and others do not, is a very interesting problem from the point of view of software engineering. In this regard, as part of the proposed method, after the final delivery of the practical project, we ask members of each group to make a list of their

teammates sorted by the perceived merits of each one. In this way, the instructor gets a list of perceived effort and contributions that each group member made to the success of the project. After this assessment of the group performance, a friendly discussion can take place about what went wrong, what was right, and how can it be done any better.

In relation to teamwork, in the software factory method roles are assigned to group members, similar to those found in the computer industry. It is common for each group member to have more than one role, and assume different responsibilities. Some of the roles used in software engineering courses were: project coordinator or project leader, system analyst, designer, programmer, database manager (DBManager), server manager (WebMaster), or system administrator.

4.2 HARDWARE AND SOFTWARE USED

In universities, it is common for computer labs to have computers that are configured and maintained by a laboratory technician. Also, normally any data or programs installed by the students will be deleted when the computers are restarted.

However, in real companies there are no classrooms nor computer laboratories. The people in charge of a project should install and connect the hardware, and install the necessary programs. In the software factory method something similar is done. A computer is assigned to each group of practices (project team), and the group must take responsibility to configure and install everything they need to carry out their project. The laboratory technician of the University should only provide the necessary software when the software is under license. When open-source software is used, which is encouraged in the SF approach, team members should download it and install it themselves. Software maintenance of this computer will be the responsibility of the ones who fulfill the role of System Administrator and/or WebMaster.

Although the courses in which the methodology proposed has been applied deal mainly with system analysis and visual high-level design (using diagrams), we ask the students to build the actual programs, with the main purpose of obtaining an integrated view of what a complete software development project is, and secondary, to improve their programming skills or maybe learn a new programming language.

4.3 TELEWORK

The software factory method encourages telecommuting or teleworking. This is a very interesting way of working that provides many benefits to workers, businesses, and the environment by allowing workers to perform their tasks wherever they are, without having to travel to a workplace. In this sense, we encourage the use of portable hardware and software, web-based “cloud” services and collaborative tools. In the software factory method we use teleworking so that team members can communicate and share files when they are not in class.

4.4 PERIODIC INSPECTIONS , FINAL DELIVERY AND DEFENCE

During the development phase of the practical assignment, periodic programmed inspections take place. In this way, the teacher can monitor the evolution of each development team. These inspections are intended to lead students to develop projects in a gradual and continuous way, being also a recommendable practice in real industry projects. In the learning environment, periodic inspections are also a good technique of continuous assessment.

The final project work is delivered with two deadlines. The first meant for delivery of the resulting products (software, documentation, etc.) as well as oral defence of the work. The second is meant mainly for discussion, but also for fixing or completing deliverables. If the first delivery has minor flaws, or missing parts, we give the group the opportunity to fix the errors and deliver the project again a few days/weeks later. Although we stress the importance of meeting the first deadline (students lose points when they fail to do so), we also consider important that a project should be thoroughly completed. In the second delivery session, we also ask students to discuss what has been done, promoting comparative analysis of results amongst different project groups.

5 SOFTWARE TOOLS

Depending on the specific subject matter, different software tools have been used, such as:

5.1 COLLABORATIVE

- Digital Humanities Workbench: on-line tools for XML processing: <equis.umh.es/dhw/>
- Google Apps (GoUMH[1]): email, chat, calendar, groups, office documents, websites: <http://www.goumh.es/>
- Dropbox: sharing files through synchronization of folders: <http://www.dropbox.com/>
- Collabtive: collaborative work environment: <http://collabtive.o-dyn.de/>
- Subversion: version control software: <http://subversion.tigris.org/>
- GIT: version control software designed by Linus Torvalds <http://git-scm.com/>
- Moodle: educational collaborative environment: <http://moodle.org/>

5.2 PROJECT MANAGEMENT

- Gant-Project: project Management: <http://www.ganttproject.biz/>
- DotProject: project Management: <http://www.dotproject.net/>
- Web2Project: project management: <http://web2project.net/>
- Grindstone: manages time and tasks: <http://www.epiforge.com/Grindstone/>
- Project-Open: project management and business management: <http://www.project-open.org/>

5.3 OTHER

- Riskology: simulation for risk analysis: <http://www.systemsguild.com/riskology/>
- Skype: VoIP: <http://www.skype.com/intl/es-es/home>
- Social networking: eg Facebook

6 RESULTS AND CONCLUSIONS

To evaluate the results and effectiveness of the proposed teaching method we use the following techniques:

(1) Conducting opinion polls to be filled by students

(2) Collect metrics from project teams ant their work

(3) Take notes during routine inspections and produce reports

(4) Student-teacher discussions about the achievements, failures and results of projects after completion (this is done in the second delivery deadline)

(5) Discussion and criticism amongst the teachers involved

The main conclusions obtained after the application of the method described can be summarized as follows:

- Concerning the avoidance of using explicit and detailed written assignment statements, we sometimes found some resistance and complaints from students. However, if we explain the reasons why this is done, students understand them, and soon we see them involved in requirements capture.

[1] GoUMH is Google-Apps adapted to be used by the Miguel Hernandez University: https://sites.google.com/a/goumh.umh.es/goumh-going-google/

- The use of new technologies and experimenting with open-source and collaborative software motivates students. Furthermore, the availability of portable software allows them to work anywhere in similar conditions.
- The post-delivery (second deadline) discussion proved to be a very useful tool, being a chance to reinforce the practical knowledge and skills acquired, as well as an opportunity to comparatively review the errors, the scheduling, the risks assessment, the various design decisions taken, and the use of different tools and platforms.
- Sometimes human-interaction or cooperation problems have been detected within practice groups. The “secret ballot” technique described above has proved very useful to show that there is a “group consciousness”, and that lazy or non-cooperative attitudes are perceived by the rest of the group.
- Work in large groups (of 4 or more students), and the use of a dedicated computer/server for each group, have facilitated the sharing of tasks, roles and responsibilities, as well as the emergence of human-interaction problems, that need to be properly tackled and managed by the group, a learning target in itself.

Some lines of future work include the extension and consolidation of the SF method to other engineering subjects and the definition of a measure of adaptation effort from traditional academic project work to industry-like project work from which teachers could show the baseline before applying the SF methodology, and compare to the results obtained after its application.

REFERENCES

[1] Bermejo, M., Fernández, R. Alumno Rupérez, está usted despedido, XI Jornadas de Enseñanza Universitaria de la Informática, JENUI, Madrid, 2005.

[2] Colbeck, C. L., Campbell, S. E. and Bjorklund, S. A., Grouping in the dark: what college students learn from group projects, The Journal of Higher Education, 71(1), p: 60-83 (2000)

[3] Cusumano, M.A., The software factory: a historical interpretation, IEEE Software, 6(2), pgs. 23-30, ISSN: 0740-7459, DOI:10.1109/MS.1989.1430446, IEEE (1989)

[4] Dougiamas, M., Moodle: Module Object-Oriented Dynamic Learning Environment. URL: http://www.moodle.org

[5] Jones, A. and Issroff, K., Learning technologies: affective and social issues in computer-supported collaborative learning. Computers and Education, 44(4), p: 395-408 (2005)

[6] Nam, W.C., and Zellner, R.D. The relative effects of positive interdependence and group processing on student achievement and attitude in online cooperative learning, Computers and Education, 56(3), April 2011, p: 680-688 (2011)