Project-oriented Course of Software Engineering Based on Essence

Denis O. Zmeev
Higher IT School
National Research Tomsk State University
Tomsk, Russian Federation
denis.zmeev@accounts.tsu.ru

Oleg A. Zmeev
Higher IT School
National Research Tomsk State University
Tomsk, Russian Federation
ozmeyev@gmail.com

***Abstract*—One of the most difficult parts in education of future software engineers is teaching how to combine software engineering hard skills (analysis and design of architecture, developing software, initial launch of software for customers) with soft skills (team work, time and resource management, communication with stakeholders of the project). Unfortunately, some universities only give theoretical basics of project management or do not even consider this part of software engineering as compulsory. In this paper we present our teaching experience of software engineering as a project-oriented course in a business-like environment. In this course we use Essence to combine the students' knowledge about software engineering projects with their experience in course projects.**

Keywords—Software development processes, Essence, SEMAT, Project based education

I. INTRODUCTION

From the moment of establishment of the SEMAT (Software Engineering Method and Theory) initiative [1] in 2009 its authors paid great attention to introducing new ideas on working with practices and methods of software engineering to the educational process of training specialists in the field of computer science (CS). In order to spread the SEMAT ideas among the software engineers it is necessary to make the basic parts of the Essence [2] standard an integral part of the educational process for computer science undergraduate programs. An educational track of SEMAT was promptly started to try to develop special courses intended to introduce the new standard [3] to future software engineers. Software tools are being developed that use the standard's notation [4, 5, 6], which can easily be used at various stages of the training process of CS students. As a result of this systematic work in 2019 a new fundamental work appears [7], which, in our opinion, will quickly become a classic textbook for all students studying software engineering as part of universities' curriculum. This paper provides an example of the use of Essence in the implementation of the discipline of software engineering for undergraduate students at the Computer Science Department at National Research Tomsk State University.

II. SOFT SKILLS AND SOFTWARE ENGINEERING

In the software engineering field, many soft skills, such as teamwork, time management, resource management, risk management are closely intertwined with such complex, highly professional issues as software development processes, analysis and requirements development, software architecture design, software prototype development, version control and management of development teams while following these processes. Moreover, problems related to the financial efficiency of the work on a project, legal issues, etc. also add to the work with the customer. Therefore, apart from the necessity of building each of the above-mentioned skills it is also vital for the future graduates to understand how all the software engineering processes interact with each other and to be able to understand their connection with the rest of the project and team. Nevertheless, it is very difficult to acquire these skills during the educational process based on lectures and repetitive practices.

In order to solve this problem, we used the project method [8] and solved the following main tasks: firstly, helped students to form an idea of modern methodological issues in the field of software development processes, secondly, showed how theoretical studies reflect on the real development processes, and thirdly, provided an opportunity to experience participating in a software development project, and, finally, formalized this experience using a project management environment. However, even after solving these problems, the problem of the conflict between different methods of running software development projects continued to exist in the development process industry. As the result, due to this problem in the previous years of teaching this discipline, teachers were forced to limit the students' teams offering them only the Unified Process, as the one covering most of the processes of business software development. However, the relatively recent emergence of SEMAT and Essence, has allowed us to improve this discipline, making it more adaptive without any loss in fundamentality.

III. GENERAL DESCRIPTION OF THE SUBJECT

In the structure of undergraduate training, the discipline in question completes a series of courses on object-oriented software development technologies. Thus, the discipline is studied by students who have an idea and practical skills related to such important issues as object-oriented programming, object-oriented analysis and design, and the development of software system architecture.

Preparing for the start of this discipline the teachers need to choose a project and a customer for the semester. A university employee is selected as the customer, who, firstly, is not an CS specialist (in order to avoid giving tips on how to develop this system in a better way), and secondly, feels the need for a similar product (to evaluate whether the system developed by the students could be applicable in real life). The project's complexity should equal to about two man-months.

To implement the project, students are divided into teams of 6-10 people (depending on the total amount of students in the course and the number of teams the customer is ready to work with). Each team has its own project manager. Typically, a project involves either one complex risk (technological or analytical) or several medium-level difficulties. Teams must report on the project's progress using the Redmine project management environment, which is adjusted for educational purposes.

978-1-7281-6807-4/20/$31.00 ©2020 IEEE

In the course of completing the project students find themselves in a situation that imitates the process of a real-world IT company: they perform certain tasks, acting a certain role in a certain development process, interact with other participants, and even earn some points (acting as virtual money) that are later converted into grades.

IV. Lectures

Considering the main goal of the course is the implementation of the project, the purpose of lectures changes. They can arbitrarily be divided into two types: factual, covering theoretical issues of modern software development, and practical classes, during which the mistakes made by the teams in the framework of the project are analyzed. The usage the methodological basis of Essence allows us to consider different practices and methods of software engineering based on a single descriptive language during the lectures of the first type. That allows students to form a holistic view of the various methods of software engineering and form the theoretical basis for comparing competing practices.

Practical classes are conducted by teachers and based on the example of the current project, which is the same for all teams. As a result, it is more explicit for all students, as all teams, carrying out the same project, experience similar difficulties. Problematic situations are most effectively explained using the visual components of the Essence language, which we partially implemented in the project management environment (Figure 1).

Fig. 1. Initial state of the project in Redmine

The most important part of the Essence language that was implemented is Alphas (abstract level project health in Essence) desk for a specific project. By using this desk teachers can show different stages of the project and what students' team should do if they want to achieve the main milestones for the subject. At this desk teachers can place all 7 usual Alphas and their states at different columns and work with their checkboxes.

V. Practical classes

During the practical classes the teachers analyze the current state of their project with each team individually. Teachers do not interfere directly in the development of the system for the customer (i.e. they do not review the code, design architecture or write any code, etc.), this part the students mostly do on their own (if the team has a crisis, then the teachers can help, but this happens on average once per semester and only to one team). The teachers mostly focus on how the students organize their work and apply the knowledge gained on the development processes and project management. In fact, the format of practical classes begins to be divided into two different consecutive phases: initial and work (an example is shown in Figure 2).

Fig. 2. The first states of Way of Working

The teachers' task during the initial phase is to show possible results of choosing certain practices for the team. For example, if a team chooses one of the Scrum variations for work, then it becomes a duty for them to record customer requirements using the Product backlog and adjust plans every week (practice shows that it is most convenient to manage weekly iterations). That means that the teachers' task in this format is to explain the difference between external characteristics of a practice and its contents, showing students the difference between the stereotypes that describe the processes and what form the practice will take in the course of the project. Each team goes through the first phase differently, in some cases it takes 2 weeks to determine almost all the main issues by themselves. One of the teams needed a month to do this (to emphasize that during this month the project had already started and the deadline for the project was approaching). So, using Essence language, at initial part of student's projects – main goal of teachers is help student to choose and establish Way of working

As soon as the team fixes its Way of Working, the work phase begins, during which the practical classes transform into consultations and quick checks of the status of the project. In some cases, certain issues are resolved on how better to fix certain issues of the project, depending on the practice selected by the students.

VI. Conclusions

As we mentioned before the main effect of using Essence to modify this subject was to allow students to choose different practices and methods of software engineering and at the same time allow the teacher to show the whole complex picture of other areas of developing. But as Essence is continuing to develop, we want to mention other results of using Essence for teaching students. In general, using Alphas has already allowed to improve students' skills and understanding (as we found out using personal interviews with students) but there seems to be some kind of difficulty related to the Essence work products related to corresponding Alphas.

1) The Requirements and Software System Work Products are very well understood by the students. Most methods and practices of software engineering focus on the solution area of concern. And the students can easily understand how to work and measure progress at each stage

of the project using well known and understandable Work Products templates and examples.

2) The Opportunity and Stakeholders Alphas are harder for the student's understanding. They require an additional lecture on business goals and general Stakeholders (not just users) to understand this level. But also, there is some difficulty with the Work Products. Students can find useful Work Product templates for Stakeholders (e.g. contacts, user profile, UX user profile), but it is hard for them to find good Work Product templates for the Opportunity Alpha that could help them formalize project information and allow to check the progress of the Alpha state.

3) The Endeavor area of concern appears to be the most difficult to understand and use by the students. These Alphas are perceived to measure the team's or company's performance rather than the project's progress. As opposed to other Alphas, on the Endeavor area of concern the student teams find difficulty with the Work Products examples. In other cases, when you execute an activity you create or update a Work Product, and this may result in progressing the Alpha states. However, in the Endeavor area of concern teams find it difficult to develop Work Products. Because there are no well-known and generally accepted Work Product templates for those Alphas, especially the Way of Working. For subject purposes teacher obligates students to develop a work product called "Team rules" at Redmine wiki, but it is very hard for students to find a good example of a Work Product with the same goal for real companies. But in general, it resembles a recursive rule: to advance project's state in the Way of Work area you need to write a Work Product which will describe your team's Way of Work, which is sometimes quite confusing and leads to problems in students understanding this area.

As additional result it is possible to download plugins (https://www.redmine.org/plugins/semat_alpha_state_cards), which we developed to implement Essence features to Redmine and use it for any purpose education or business

REFERENCES

[1] Jacobson I., Meyer B., Soley R. "Software engineering method and theory — a vision statement", Feb 2010

[2] 2. SEMAT. 2013. Essence – Kernel and Language for Software Engineering Methods. Available at: https://www.omg.org/spec/Essence/1.2/PDF [accessed February 5 2019]

[3] Zapata C., Jacobson I. "A first course in software egineering methods and theory". Dyna rev.fac.nac.minas [online]. 2014, vol.81, n.183, pp.231-241. ISSN 0012-7353. http://dx.doi.org/10.15446/dyna.v81n183.42293.

[4] Essence Enterprise 365. [Online]. Available: https://www.ivarjacobson.com/essence-enterprise.

[5] SEMAT Essence Kernel Tool https://semat.herokuapp.com/

[6] Graziotin, D and Abrahamsson, P 2013. A Web-based modeling tool for the SEMAT Essence theory of software engineering. Journal of Open Research Software 1(1):e4, DOI: http://dx.doi.org/10.5334/jors.ad

[7] Jacobson I., Lawson H., Ng P.,McHahon P., Goedicke M., "The Essentials of Modern Software Egineering", 2019

[8] Project method Encyclopedia of Educational Theory and Philosophy, ed. D.C. Phillips. Thousend Oaks, CA: Sage 2014. Vol. 2. Pp. 665-669.