Developing a Course on Designing Software in Globally Distributed Teams

Patricia Lago  
VU University Amsterdam, Netherlands, patricia@cs.vu.nl

Henry Muccini  
Univ. of L’Aquila, Italy  
muccini@di.univaq.it

Muhammad Ali Babar  
Lero, Univ. of Limerick, Ireland  
Muhammad.AliBabar@lero.ie

Abstract

Present-day software engineering combines technical and social skills, as well as collaboration among people with different backgrounds (e.g. due to global development and outsourcing). In this paper we address the problem of teaching “globally distributed development”, and specifically software design. Our goal is to develop a joint Master course teaching software design in a global setting. To this end, we contribute with a list of characteristics to be developed, in the form of an ‘orientation map’ for educators. We use this map to build a joint course between two European universities.

1. Introduction

Software engineering (SE) combines technical and social skills. One of the most important social skills required in SE is collaboration among people with different backgrounds. Inter- and intra-team collaboration is becoming increasingly important in modern SE, which is characterized by new paradigms like outsourcing and offshoring in the context of global software development. This situation adds a further difficulty to the already difficult task of teaching realistic SE in an academic context, as well as setting up associated projects. In this paper we describe our approach to developing a joint course on designing software in globally distributed teams for students from two different universities as well as different cultural backgrounds. Our main goal is to develop a joint Master course on designing software in a global software development setting for students from the VU University Amsterdam (VU) and the University of L’Aquila (UDA). A joint course on designing software is expected to provide students from two countries with complementary knowledge about software design methods, notations, best practices and project management. We want our students to apply their skills and knowledge to deal with cultural differences and different aspects of collaboration in geographically distributed teams. For developing and delivering a joint course for Dutch and Italian students, we have been combining the incremental and iterative approaches involving experimentation. We have performed three iterations during our progress towards designing and delivering a joint course. Our approach takes into account both technical and non-technical backgrounds and characteristics of the students who are generally enrolled in two separate courses at two different universities. During each iteration, we have been experimenting with course curricula and logistics and getting feedback for the next iteration. The incremental and iterative approach has provided us with several opportunities to learn about different aspects of teaching software design to geographically distributed teams of students and their learning patterns and requirements. We have also collected data about the SE background, organizational skills, and cultural differences of the students who were enrolled in two universities. Analysis of the data collected during three iterations has enabled us to draw some general conclusions that we believe are worth sharing with the community. Having worked on this approach for the last three years, we believe that the steps involved in our approach to developing the course can be used in general by educators aiming at introducing global and distributed issues in their student populations, in courses addressing SE topics in general.

Few works address culture in SE education. Navarro & v.d. Hoek (6) experimented with projects involving students coming from different specializations, hence bringing in interdisciplinary backgrounds. Brereton & Lees (2) investigated the factors influencing project outcomes. While they do not observe relevant advantages in mixing technical and non-technical students, we count on complementarities: Verkamo et al. (3) focus on distributed cross-cultural SE. In a case study in Finland and Russia they observe that initial cultural differences are solved by experience during the project. This is also one of our objectives. Recent publications provide further inspiring input, e.g. Avritzer et al. (1) introduce local experts to bridge cultural differences.

The paper describes the steps of our approach to developing and delivering a joint course on designing software in global software development context. The paper elaborates on how each step has been carried out to identify a number of characteristics (highlighted in boldface) for design-

Authorized licensed use limited to: University of Limerick. Downloaded on May 14, 2009 at 11:13 from IEEE Xplore. Restrictions apply.
ing our joint course. We also provide some findings from one of the preliminary study that we have been performing in the process of designing the course. The joint course is part of a broader project creating a European Master programme in Global SE (4), where successful SE in global settings is a central learning objective.

2. Course Planning

We have taken experimental approach to seamlessly integrating two separate courses being offered at different universities. We have been studying different aspects of the contents and logistics of two separate courses and typical student population over the last three years in order to make a smooth transition to a completely joint course without disrupting the degree programs to which the two courses belong. Our course design approach is based on our observations and analysis of the data collected each year from students in the local teams (i.e. students in the same university) and international teams (i.e. students from two universities). Here we explain our course design approach by discussing our observations over the past three years as well as analysis of the data collected from our first experimental study.

It is obvious that the students from different universities will have different technical backgrounds, experience, and cultures. Hence, we believe that it is important to analyze their prior education. The main objectives of this step are to answer the following questions: 1) Will the students’ own experience bring some benefits to the joint course? 2) When working on a distributed joint project, will they learn from each other more than by simply working with colleagues with a similar prior education?

To answer these questions, this section analyzes the prior education of students from UDA and VU attending the MWA (Modeling Web Application) and ATSD (Advanced Topics in Software Design) courses in 2005. This assessment intends to characterize the two student populations. The characterization is used to design the global course implementing the auspicated benefits. The assessment of prior education has been based on the analysis of the self-reported data collected through a questionnaire, which has the following questions:

- SE courses attended in the Bachelor, teaching: SE introduction to theory, SE project or Lab, Advanced SE, and UML and/or Web related courses.
- Job experience: questions were, Did you ever work in any company (Y/N, if Y then explain)? Are you still working in any company (Y/N, if Y then explain)?
- Implementation skills about: Web technologies previously studied/used (Y/N, if Y then explain level (1=low, 5=high); and other implementation technologies, (with level).

The questionnaire was distributed to the 28 students enrolled in the UDA Master on Web Technology. The analysis of the collected data revealed that about 46% of the students has previously attended an introductory SE course. This is mostly a theoretical course on basic SE principles. About 43% of the students attended the UML Lab, a project-based course, where UML and RUP are taught. In summary, about 28% of the students attended both courses, while 39% did not attend any SE classes. 85% of the students previously attended some course on Web Engineering. About implementation skills, about 78% of the UDA students had academic experience in projects with PHP, HTML, XML, Javascript, Java, C++. Only 7% of the students had experience in industrial projects.

There were 41 responses from the VU students enrolled in the course ATSD. The data analysis found that about 93% of them has previously attended the introductory course in SE theory. We excluded 3 students from the analysis as they were international students who have attended other equivalent courses abroad. This means that all students have a SE theoretical background. 85% of the VU students attended the Project SE, in which they carry out a large project in teams. Education objectives of this course are: putting SE theory into practice, experience teamwork, project management. On average, 19% of the students did attend additional optional courses around Web-based SE. More than 50% already attended the Software Architecture Master course, which provides them with professional background and practice in carrying out a project, teamwork, documenting technical results. About implementation skills, about 90% held extensive experience with Web technologies, such as PHP, HTML, XML, Flash, JSP, ASP, Javascript, J2EE. All students have programming experience with at least Java and C++, in either/both academic or/and industrial projects. 32/41 students (78%) work in IT companies in parallel with their studies. In many cases, they already own their own small company in IT consultancy or software and Web development.

In summary, the analysis of the data on prior education, skills and work experience collected from students at the two universities provided us with some interesting differences in the two student populations. Less than 50% of the UDA students attended SE related courses, but these courses go deeper into few specific SE techniques and the use of UML. This is in contrast with the VU population having a quite standard SE background and further shorter courses related to Web technologies. This reveals that while UDA students have a more technical in-depth education background, VU students have a more general, standard education background. Concerning job experience and implementation skills, UDA students are more aca-
3. Assessment of the Present Education

In this section we analyze the MWA and ATSD courses, to identify the characteristics we want to combine in the new joint course. The parameters we used to describe the two courses are: SE contents and workload (lessons given and how many hours are spent on related contents), propaedeutic background (background education, knowledge of web technologies, and work experience), grading method (written exam, project, or oral exam).

The UDA Master in Web Technology is a one-year Master composed of eight courses, for a total of 480 hours of face-to-face teaching, plus a 120-hours industrial stage (for a total of 600 hours and 60 ECTS). The MWA course is a 6 ECTS course. The exam has been realized through the assignment of three projects: a purely UML small project, UML modeling of a small web application in the UWE (UML for Web Engineering) approach, and a bigger project that required the modeling of a mid-complexity web application in one of the two modeling techniques. Much attention is given to the quality and thoroughness of the delivered design, to the ample usage of the learned modeling concepts, and traceability among different models.

The VU CS Master is a two-years master of 120 ECTS. Most of the Master courses include a project putting theory into practice. In some cases, the project is carried out at companies. In ATSD (6 ECTS) the students are graded depending on the results obtained in the course project: Much attention is given to the quality aspects of the delivered design document, documentation of the rationale, correct and motivated use of the selected topic. In summary, both courses include a students project. Differently, the MWA course draws much attention on correct use of one Web design method, whereas the ATSD course is focused on the motivated selection of one among multiple methods (specialization topics) together with documentation of design rationale. A more detailed analysis of commonalities and complementarities is given in Section 5.

4. Assessment of Cultural Aspects

Hofstede and Hofstede (5) explain culture manifestations in terms of five levels of depth, namely symbols (e.g. the colors of the national soccer team), heroes (its best soccer player), rituals (singing the national inn to open an international soccer game), values (national pride) and practices (listening to the prime minister’s yearly speech to the nation). While the first three levels above are quite easy to identify, values and practices hold a very hidden and profound meaning, and are therefore much more difficult to grasp. In this work we could identify university cultural facets at the level of rituals and values. Specifically, there are some cultural facets that influence the way SE projects are addressed and carried out by the students. A positive aspect influencing teamwork is that the VU students like being independent. They highly value a certain degree of freedom in the way they manage work, deadlines, assignment of roles and tasks. The same holds for the way project assignments are formulated: the possibility to autonomously decide on the specific characteristics of the software application under development, or the technologies used, is highly appreciated. Freedom in the project increases motivation. UDA students are more academic, more concerned with specific technical issues like solving a programming exercise. At the same time, VU students are good project managers. They are rarely late for lectures or project meetings, they take their responsibilities. This quite often results in good teamwork. Management skills are not part of the UDA education. A cultural aspect that sometimes influences the quality of the results in less positive ways is that the VU students perceive requesting help from the teacher negatively, and when they have cooperation problems they insist hiding them until it is too late. The UDA students, instead, consider it normal (as part of the education) to ask support from the teacher. Lastly, the Dutch society does not generally reward high grades. This results in a low level of competition: VU students often expect a large amount of work ensures a sufficient grade, even if the resulting deliverables are of very poor quality. The Italian society, instead, is based on results: during interviews companies ask about the final Master project grade.

5. Analysis

Sections 2-4 have identified a number of characteristics of the courses and the student populations at the two universities. Here we analyze them to decide how to use them in a joint course. To this end, we define the schema shown in Figure 1: on the x-axis we find the characteristics themselves (classified as present and prior education, professional skills and cultural aspects); on the y-axis, if these characteristics are common to both universities (marked with “≡”) or if they instead are complementary (marked with “≠”). We are especially interested in commonalities and complementarities because we start from the following hypothesis: by combining in a joint course the students’ education background, their skills and cultural aspects, and the contents of the two courses, the students can learn from each other faster than studying the same material on books.

The figure provides what we call an orientation map for educators: we use it to decide on which characteristics can
be exploited in the joint course, to achieve which learning objectives. In Section 7 we further describe how to realize these learning objectives, i.e. the means shown on the left-hand side of Figure 1 (i.e. learn-by-osmosis and learn-by-doing). In more details:

(≡) About Education, both courses include student projects. Further, they both cover Web design, but (see next point) . . .

(≠) . . . the VU course has just one lecture on modeling Web applications (i.e. one among five specialization design topics); the same contents are taught at the UDA, but spread over four lectures and in much more detail. The other way around, the UDA course does not address quality of the documentation, and design rationale (fundamental topics of the VU course).

(≡) About Skills: VU students have a broader but more superficial knowledge of existing/industrial Web technologies, and programming experience. UDA students have a more academic background, and in-depth knowledge on specific SE topics.

(≠) VU students both study and work. As a consequence they are more pragmatic than UDA students holding less industrial experience.

(≠) About Cultural aspects: the VU is much more international than the UDA. Therefore, the VU students are used to cooperate in multi-lingual and multi-cultural teams. By working in a project team, they also contribute the prior background developed at their home university. Nonetheless, experience learns that international teams benefit from this prior background only if this is integrated in the project (i.e. made available to other students).

(≠) The education curricula at the VU include many student projects, which stress decision making and freedom of selecting an appropriate SE technique, and are often carried out in companies. In these projects the students develop skills on project management and the abilities to collaborate in teams. Differently, UDA education includes fewer projects, focused on a specific UML method, and not involving industries. The students do not do project management.

(≠) UDA students are more competitive than VU ones.

In summary, in a joint course VU students bring in horizontal/broad experience (e.g. project management, job experience, teamwork) and high-level education about software design and documentation. UDA students bring in vertical/in-depth experience in academic projects, and very detailed education in UML and web-based SE methods.

6. Preliminary Study

The discussion so far is mainly based on our observations and the data gathered with the questionnaire. The next step was to verify our conclusions in practice. To this end, we have carried out a preliminary study, which was focused to study one of the characteristics shown in Figure 1, the complementary education about ‘Documentation & rationale’ and ‘In-depth technical design’. We realized the means to transfer the students’ knowledge as follows. The VU students were given a two-hours lecture on modeling web applications with the UWE approach. The UDA students attended the usual eight hours on Web modeling in UWE (preceded by 24 hours on UML theory and practice). Eight VU teams, composed of three members each, participated in this study. Their project focused on combining Design Rationale and UML modeling with UWE. Nine teams at UDA, instead, applied the UWE method as-is, for modeling the same system.

UDA and VU had two different evaluation schemas (being the two courses different in concepts): the teacher at UDA evaluated the local projects according to significance of utilized diagrams, traceability among different models, conformance of modeling and implementation artifacts, good design of important design decisions, no more than 100 pages for projects’ documentation. The VU teacher evaluated local projects according to quality of requirements (e.g. identified relevant stakeholder?), quality of the design space (identified issues/options/decisions? for each issue: why important? options pros and cons? motivation for the decision?), design solution (identified why selection of views?), originality, documentation quality. The results of this experimental study have been gathered according to the following rules: the VU teacher graded her students according to the VU evaluation schema, the UDA teacher evaluated the UWE solution of both UDA and VU students according to the UDA evaluation schema. Thereafter, the two lecturers compared the design quality of those projects done at UDA with projects done at VU. The hypothesis was that VU students would produce less precise and thorough
The results have shown that four VU teams produced a still precise enough design (i.e., the diagrams, evaluated according to the UDA evaluation schema, were not fully precise or expressive, but generally correct), with good design decisions (according to the VU evaluation schema). Differently from UDA teams, those four teams produced documentation with "readers guide" instructions, using a "politically correct" terminology (i.e., using he and she), and documenting the intermediate and final decisions (those are attributes typically missing in UDA designs). One additional VU team has shown a design similar to the UDA ones, even if the design according to VU parameters was just sufficient. The three remaining VU teams provided a very low quality design solution, not conforming to UWE and with bad traceability. It was interesting to notice that in this case, a design to be considered of low quality according to the UDA evaluation schema, was instead a good one according to the VU schema: the reason is that even if the VU students were not using the UWE approach in a standard way, they were motivating their decisions, thus still producing a valid result according to the VU evaluation schema. Therefore, even if the experimental study did not fully support our hypothesis, it confirmed the characteristics of the student population as defined in Figure 1.

7. Design of the Joint Course

Section 5 discussed the commonalities and complementarities of the courses to join, and the related learning objectives. Here we decide which means need to be designed to achieve them. We recognize three means:

Learning by Osmosis: by collaborating in international projects, the complementary SE contents are transferred more effectively between the student populations, and the learning period is shortened.

Learning by Doing: by working in joint teams, the students can both put theory into practice, gain experience in how global design is carried out in a distributed setting (use of tools for distributed collaboration, effective documentation and communication), and gain insight in the core issues specific to global SE (e.g., working with different cultures);

Traditional classes: lectures are given to both teach topics specific to a certain course, and teach the topics shared among the joint courses.

As shown in Figure 1, complementarities can be better learned by osmosis, to:\n\(\text{(a)}\) Transfer across the joint courses the necessary knowledge about shared topics. For instance, the material about Web design taught in 8 hours at the UDA is also summarized as one of the specialization topics at the VU (2 hours). The other way around, the material about Design documentation & rationale, taught in 6 hours at the VU, is also summarized to the UDA students in one lesson (2 hours). In this way, the two student populations have their specific in-depth background and a general understanding of the background of their colleagues. Hence, they have the means to learn by osmosis from the international colleagues.\n\(\text{(b)}\) Transfer job experience, skills, cultures between members of the international team.

Learning by doing, instead, seems to be the best means to teach commonalities. I.e.:\n\(\text{(a)}\) Build (international) project teams.\n\(\text{(b)}\) Merge the two student populations in the teams, and allocate the tasks to delegate the main responsibility according to the specific background of the students, but at the same time ensuring knowledge transfer. For example, the UDA students would have main responsibility in using the UWE approach for the detailed system design, whereas the VU students would have to analyze requirements, make design decisions, review the UDA design for feedback, and ensure documentation quality.

In addition to Figure 1, traditional classes teach local SE contents defining the differentiating factors of the populations. A related issue is the need to define common evaluation criteria for both courses to assess the results of the joint projects. For example, we merged the criteria used in the preliminary experimental study, as: correctness, originality, consistence & traceability, documentation quality.

8. Conclusions

Teaching global software design to distributed teams poses new challenges dependent on students background and cultural differences. This study has analyzed how teams in similar projects but different universities/countries can perform when different background knowledge exists. We defined an orientation map providing directions on how to develop a joint course profiting from the cultural differences of distributed teams. This study has represented the first step in developing a joint European Master programme.

References