Teaching Practical Software Engineering and Global Software Engineering: Case Study and Recommendations

Dragutin Petkovic[1], Rainer Todtenhoefer[2], Gary Thompson[3]

Abstract - **In this paper we present the innovative methods and experiences from several years of teaching practical software (SW) engineering at the Computer Science Departments of San Francisco State University (SFSU), USA in conjunction with the University of Applied Sciences, Fulda University, Germany. The key objectives and desired outcomes of our course were to train future SW developers, technical leads and managers in practical SW engineering practices, including global SW Engineering, where the team members work in different locations. Our key approach was to combine and synchronize class teaching about SW engineering methods and processes with actual SW development work in a setting designed to simulate a small SW company. Students were divided in "local" groups of 4-6, each forming a small SW "company" in charge of developing a complete working WWW application as a final class project. Several groups of students at SFSU were "virtually" paired with groups of students at Fulda University, whom they never met face to face, to form "global" groups, thus simulating global SW engineering in a realistic setting. Students developed final project incorporating five well-defined milestones typical for SW development lifecycle. Instructors spent considerable time supervising and interacting with student "companies" in the role of company customers, CTO and VPs of engineering, marketing and sales. In addition, student filed weekly "time sheets" which were questionnaires designed to provide feedback, which was used to analyze and understand the teaching effectiveness and improve the class. All student groups (including global ones) were able to produce impressive final project applications and gave very positive feedback for this class.**

Index Terms – Software Engineering Education, Distributed Software Development, Global Software Engineering, Outsourcing Software Development

INTRODUCTION

SW Engineering (SE) education (curriculum, outcomes and delivery) has received considerable attention from IEEE and ACM societies [1]. There SE is defined as "The application of a systematic, disciplined, quantifiable approach to the development, operation, and maintenance of software". Guiding Principles from [1] clearly state that SE education needs to combine computer science foundations with engineering, organizational, teamwork, communication and project management issues. Guidelines for curriculum delivery from the same reference clearly point to the need for practical project and team-oriented exercises with significant capstone project.

In the last several years some new trends significantly influenced the methods and practices of SE. These trends can be summarized as follows:

- Maturing of SW technology and the availability of powerful tools to aide in SW development and management coupled with increased size and complexity of Software
- Emergence of WWW interconnected world where tools, practices and code can be easily shared and leveraged
- Globalization of SW development (we call it "global SE"), where geographically distributed teams in different time zones, from different countries and cultures collaborate in the development of SW
- Open source SW development
- Increased importance on creating software with a high level of usability
- Increased emphasis and requirements for teamwork and project skills of SW developers.

Furthermore, in many studies [e.g. 2], and consistent with our industrial experience as well as feedback from our industrial partners, most of the failures of delivering SW to specs, on time and budget, and to a user "delight" were in misunderstanding user needs, poor design, planning and organization. Successful industrial hires today are required to have the ability to work within a team type framework, and understand user needs, participate or manage in SW development, and understand SE processes. Further complicating the issue is the emergence of global SE, where geographically and culturally disparate individuals are grouped into teams that are involved in building SW products. This not only complicates the already challenging SE

[1] Dragutin Petkovic, CS Department Chair SFSU dpetkovic@cs.sfsu.edu
[2] Rainer Todtenhoefer Professor, University of Applied Science, Fulda Germany Rainer.Todtenhoefer@Informatik.fh-fulda.de
[3] Gary Thompson, SFSU and Program Manager, java.net gary.thompson@sun.com

management issues [6, 7] but also raises various cross-cultural issues [5, 7].

More and more universities are recognizing the need for changing SE education to address the above needs but we believe that much more attention has to be paid to this. In many universities SE courses actually cover advanced programming techniques, and not SE processes and teamwork, let alone global SE issues. One example of effective teaching of SE as defined in this paper is at the University of Victoria's class on Global SW Development, which addresses high-level requirements, and specification development [8]. We use similar methods, but we also include full development and implementation in local and global context, thus covering the full SW lifecycle with all its SE implications.

GOALS AND OBJECTIVES OF THE SFSU/FULDA SOFTWARE ENGINEERING CLASS

At the SFSU Computer Science Department (www.cs.sfsu.edu) we educate students for BS and MS degrees in Computer Science. Most students aspire to work in high tech industry within "Silicon Valley". A significant proportion of BS students continue toward a MS degree and some MS students (5-10%) chose to pursue a Ph. D. at other Universities. Many students have significant work experience. SFSU also maintains a large population of foreign students. Entry surveys show, however, that most students enrolling in the SE class do not have significant teamwork or project experience, nor do they have experience in the SE processes. We started a new SE class in Fall 2003 to address the needs for novel SE education. In addition, since Fall 2004 the SE class from SFSU has worked in conjunction with the Software Engineering class from the University of Applied Sciences, Fulda, Germany (http://www2.fh-fulda.de/isu/2005/fh_en.htm) to add an element of globalization to the educational experience. Fulda students, similarly to ours, generally pursue careers in industry.

The high level goals in our new SE class are:

a) To provide practical and competitive SE education for SW developers, tech leads and managers with organizational, communication, process and teamwork focus
b) To analyze, understand and improve SE education (including how to teach global SE).

Note that other components of The SE education technical requirements, as recommended by [1], are provided in other classes at SFSU, and that students are expected to have considerable programming knowledge before joining this SE class.

The learning outcomes we expect from students successfully completing our SE class are consistent with [1] and include:

- Knowledge of basic SE engineering methods and practices, and their appropriate application
- Knowledge of basic components and tools of full SW development lifecycle, from initial specs to final documentation
- Knowledge of design and build practices for SW that is easy to use and maintain using modern multi-tier architectures
- Knowledge of basic SW dependability metrics, quality metrics, and basic architectural models
- Ability to constantly iterate and re-prioritize goals based on user needs, budget, schedule and resources
- Development of significant teamwork and project based experience, as close as possible to real life
- Exposure to global and open-source SE methods and practices.

TEACHING METHODS

Our SE class combines and synchronizes traditional classroom teaching with a significant class project developed by groups of students, as well as intensive and constant interactions between instructors and local and global student groups.

At high level, to address the goal a), our course covers main SE topics in a typical classroom teaching mode, such as: Overview of several basic SE methodologies with emphasis on Iterative and Incremental Development and User Centered Design; Usability and UI Design Principles and Practices; Basic Components of SW Engineering Process: Requirements and Specifications; Iterative Design, Rapid Prototyping, Mockups; Software Design; Coding and Documentation Techniques (high level only); Open Source SW Development and Management; Software Configuration Management, Delivery, Installation, and Documentation; Software Metrics, Software QA and Testing; Software Architecture and Middleware (high level only); Project Management issues; Teamwork and Communication issues; Issues related to global SW engineering; Real life examples and cases from instructor and students. The books [3,4] were primary classroom material. For issues related to global SE we used [7] and various papers.

In parallel and synchronized with the teaching of the above topics students were formed into groups of 4-6 where they were required to build a complete WWW application for a final class project by implementing the SE theory they learned in the class, simulating a small SW company. We formed "local" groups with students from SFSU and Fulda respectively (usually around 10 groups), as well as about 3-5 "global groups" consisting of students from both universities (3 from each university). Global groups had the same tasks and milestones as local, groups, and were offered a bit more coaching and help from instructors, as well as additional tools for distributed collaboration. Instructors played the role of senior management, and gave user feedback. This group activity and intensive interaction with instructors simulating

small SW companies, with development milestones synchronized with classroom teaching, is, in our opinion, critical for providing modern and practical SE education.

To address goal b), we had our students fill out timely (weekly) questionnaires and provide written feedback, which we plan to analyze and use to improve the class itself in an ongoing manner. Students also filled out standard class evaluation forms at the end of the class.

In order to encourage interaction and learning of SE methods, grading of this final project was equally based on the following criteria: a) adherence to SE methods b) student interaction, and c) quality of final delivery (documentation, ease of use, functionality, architecture etc.). In order to make teaching as effective as possible, and the simulation of real-life SW lifecycle as realistic as possible, the following methodology was used: a) instructors played the roles of "management (coaching) staff" (CEO, VIP Engineering, VIP Marketing, CTO, customer representative etc.) and customer advocates, and students were the developers. The Professor from Fulda was a "CTO" for global SE groups, with Prof. From SFSU a "CEO" and customer representative; b) student team members were selected by the "management staff" i.e. instructors; c) management staff provided only the initial high level project concept, and students themselves had to develop real requirements and specs following the appropriate methodologies; d) management staff imposed a strict delivery timetable in the form of five well defined milestones, consisting of:

- High-level requirements and specs;
- Detailed requirements and specs (including functional, non-functional specs, use cases, priorities, mockups of UI, high level architecture, data and content dictionaries etc.);
- First prototype (with formal one hour walk through meeting outside of class);
- Beta launch with formal usability evaluation performed by instructors;
- Final delivery and demo in "trade show" presentation

Students completed weekly "timesheets" for class and SE process evaluation. A set of collaboration and SW lifecycle management tools and basic WWW products for hosting student projects was also provided. Classroom teaching was synchronized in order to prepare students for each milestone, and at least 45 min. at the end of each class meeting was used for group exercises where student groups would work together while instructors performed "management by walking around".

The instructors collaborated before the class to determine a suitable WWW-based product concept for final class project. The projects completed over the three-year period include stock photo site, digital library of student documents, and video rental store. A one page high-level description of the project (intentionally vague, and not necessarily complete) was the only document provided to the groups as the development assignment. Using the above five predefined milestones, and interacting with "management staff and customers" (i.e. instructors) the students were asked to design, develop and demo the full product. Students developed and documented each milestone after doing their own designs and evaluations, while interacting with instructors. Each milestone was formally reviewed by instructors (and not graded at that time), feedback was given, and then it was revised by students before moving forward, thus promoting iterative development and teaching students how to deal with constructive feedback. Documentation from all milestones was combined to form a documentation package for final class project delivery and grading. During the final milestone, each group demonstrated their project to the class, which was graded for overall quality, functionality, ease of use, implementation and architecture.

Instructors' interaction with student groups was the cornerstone of the class. It was made clear to students that iteration and interactions were encouraged and not graded, so that they felt comfortable asking questions, getting clarifications and challenging mandates. This interaction included both "user feedback" (on functionality, UI, ease of use etc.) as well as technical and organizational help.

It was important to create student groups with balanced set of skills. The SFSU class involved graduate and undergraduate students. In order to accomplish this balance, a survey on background and experience of each student was done at the beginning of the class. Determining criteria included a self-assessment of experience in:

- General programming
- Web based development including browser GUI
- Working in groups including distributed groups;
- Project management.
- The desire to work in a global group was also polled.

Management staff then selected group members, including global group members, to best balance the skill level of all groups. Starting Fall 2005, based on student feedback and previous experience, a student group lead was selected by management staff in order to facilitate communication between each group and the instructors.

Tools provided to effectively complete the project were of two types: *collaboration* tools and *development* tools. For collaboration tools we chose java.net. Java.net is a free collaborative software development hosting site (http://www.java.net/). It provides various professional collaborative development tools including:

- Archived email lists
- Archived discussion forums
- CVS repository
- File sharing
- Customizable user interface
- Ability to select private or public visibility of work within the project.

A public parent project (i.e. http://videostore.dev.java.net) was created to serve as the corporate intranet. This collaborative aspect was useful for the local development teams as well as the global teams. Additional collaboration tools provided to the global groups included regular phone, a VoIP telephony service (Skype http://www.skype.com/) and Virtual Network Connection (VNC) server and client to allow shared computer desktop. For presentations of global groups, a Web Browser was running on a server in Fulda, and students from SFSU connected to this browser, and also used phone or VoIP. The development tool set included choice of two programming languages, Java and PHP, as well as a relational database (mySQL), web or application server (Tomcat or Sun Application server), an IDE (NetBeans or Eclipse) and a standalone CVS client (tortoiseCVS). An application hosting service loaded with required software was also provided as part of SFSU CS infrastructure. Local student groups had a choice of selecting PHP or Java (making this choice was actually part of their class assignment), while global groups were required to use Java.

Course grading was as follows: 40% was based on a final closed book exam testing student knowledge of basic SE topics, 20% was based on individual programming assignment (usually WWW based front end form filling application graded for functionality and usability), and 40% was based on final class project including milestone documentation, and quality of final project delivery, judged subjectively by all instructors. All student group members received the same grade for final project, unless it was determined (usually via student feedback) that some students did not fully contribute to the best of their abilities.

EVALUATION AND FEEDBACK

Class evaluation was done as follows: 1) quality of student output (i.e. final project and the documentation); 2) formal class evaluation at the end, as mandated by SFSU; 3) solicited and unsolicited student feedback; and 4) class enrollment. In addition, over 500 weekly student questionnaires are being analyzed at this point.

All groups (local and global) were able to produce working WWW applications of excellent quality, given their experience and knowledge at the beginning of the class. They achieved very high grades for final project (on average 95%). Some examples can be found at https://csc640-09.dev.java.net/ for local and https://csc640-11.dev.java.net/ for global groups.

Final class evaluation was measured by answering a standard SFSU questionnaire with graded responses. We used the question "The course was organized in a way that helped my learning" to assess the usefulness of the class. The average score for the there years of teaching this class was 1.76 (1 being the best, 5 the worst) which is considered very good.

Solicited and unsolicited student feedback was also collected, including those students who finished the class and were using that experience in their work or jobs search. The feedback was generally very positive including student comments that interviewers gave very positive feedback on their class project during the interview (critical for students with no work experience), and that they were able to implement knowledge and experience from the class in their current work.

Finally, one indicator of the class success was enrolment. Our class would fill out in the first day of enrolment, in spite of challenges in CS enrolment these days.

Based on all this we claim that class is successful and that it is very relevant to student education and training. More specific feedback follows.

Quality of final "SW product": All groups produced projects of considerable quality. By comparing the grades given on final project, we conclude that both local and global groups were able to produce products of comparable quality. However, global groups suffered more "stress" and frustration, evidenced by their feedback on the weekly questionnaire.

Course topics and coverage: The focus of the class at SFSU was on methods, processes and design for usability, and not on SW architectures or programming techniques. This was important to explain to students at the beginning due to the many ways SE courses are currently defined at academic institutions. The Fulda course covered the some topics but had additionally a stronger focus on SW Architecture and Programming Techniques. However, there were some requests for more formal instruction on specific technologies, tools and the architecture upon which the final class project was built. Some students had to spend significant time learning those aspects independently. Students in Fulda had a long experience in programming and J2EE technology, hence they also coached their San Francisco team members on those issues. Students liked very much large number of real-life examples and case studies covered in the class.

Group selection, teamwork and organization: Having well balanced groups in terms of skills was very important, and the early class survey helped achieve this. At SFSU, we combined graduate and undergraduate students in each group. There was concern that, because management staff (instructors) selected the membership of each group, there would be significant dissention and an attempt by students to re-organize groups. In fact this issue hardly arose. Usually, only 2-3 students per class requested some changes. After the second year of teaching it became apparent that a number of groups suffered from a lack of internal leadership (someone responsible for calling meetings, communicating with instructors, checking schedules etc.). To alleviate this problem, in Fall 2005 the instructors selected, based on the best of their knowledge (self-survey, meetings), one member of each group to serve as a group leader. This individual

had all the same responsibilities of all the other member in the group, but was also responsible for organizing group meetings and serving as a single conduit of communication between the development group and the management staff external to classroom meetings. Having a group leader helped overall. Finally, the same grade for final group project was given to all group members, unless the group formally alerted the instructor in case some student did not contribute adequately. This happened in only a very few cases, but was important since it influenced group dynamics and morale. Due to the fact that many of SFSU students are working professionals scheduling group meeting times outside the established class meeting times was found to be very difficult. As a result of this, in Fall 2005 at SFSU approximately 45 minutes at the end of each formal class meeting was devoted to group work. During that time student groups worked together and 2 instructors would spend 10 min. with each group checking progress and issues. This was found to be very effective and resulted in considerable improvement of student learning and exposure to real teamwork.

Global SE issues: As said before, several student groups formed global groups each with 3 students form SFSU, and 3 from Fulda. Global groups suffered from similar problems as in any SW company in regard to managing global teams: slow start, communication problems, time difference (9 hours in our case), difference in cultures and languages etc. Generally, these groups started slower and instructors spent more time with them. The difference of nine hours presented a significant difficulty in arranging meetings. Cultural differences also played some role (attending meetings in a timely manner was one of them). There were some issues surrounding communication and language skills within the global groups. Interestingly, these arose from the fact that there was a large number of international students at SFSU for whom English is a second language. Finally, students suggested that in the future more effort should be devoted up front to getting global group members to know each other (e.g. create WWW site with pictures and videos etc.) – this is a technique now advocated by experienced managers of global teams. One of the challenges was to reconcile different class material and different class start times between the two universities such that the groups could work smoothly. For example SFSU students were being educated more on SE process, while Fulda students were being educated more on practical SW architectures. This skill difference also influenced the task assignments for global group members. In addition, SFSU class started 3 weeks earlier than Fulda, so SFSU students did the first pass on high-level requirements and then shared with Fulda students as soon as that class started. As for communication tools, the groups used e-mail, java.net, Skype, VNC, VoIP, and shared WWW browser successfully. In summary, students in global groups had more difficult time, but they were also committed to this and at the end were able to develop final project of comparable quality to other local groups.

Grading: Students liked the fact that interaction and feedback was not graded and felt that this encouraged them to interact, iterate and ask questions. They generally felt that the final class project should carry more grade points since it required significant work (e.g. 50%).

SUMMARY AND RECOMMENDATIONS

New trends in SW development such as globalization, open source SW, need for developers and managers to have significant teamwork, communication skills and be able to deliver easy to use SW on time, specs and budget necessitate changes in the ways we do SE education and training. SFSU and Fulda University have recognized this and embarked on an effort to create a new SE class to focus on SE methods, process, communication and organizational issues, both in local and global setting. Our SE class has become a success and it offers valuable training our students, who are going to industry as developers, tech leads or managers. While detailed analysis of student written feedback is ongoing, we already learned valuable lessons. Our recommendation for teaching such a practical SE class can be summarized as follows.

- The combination of teaching of SE processes and methods with significant student group project exercises, intensive instructor interaction, and simulation of full SW lifecycle is critical. The synchronization of class material with final project milestones is very beneficial to students
- Students should form small groups of 4-6 to simulate real life SW companies and go through full SW lifecycle via well-defined milestones. Student groups should be selected by instructors, and have balanced skills, with one student selected as a group lead. Early class surveys can help in such selection. Course instructors should play the role of senior management and end users for those “SW companies”.
- Students should have structured time where they can work in a group setting, preferably immediately after classroom teaching portion of the class.
- Grading polices should reflect and encourage adherence to SE methods, group interaction and iterations, and not be based only on the final delivery.
- Where possible, implementation or simulation of global SE should be done by having student groups formed of members they do not meet face to face. Care should be taken up front to ensure that local and global groups get to know each other at the beginning and effectively establish teamwork.
- Some class time should be devoted in educating students on the particular SW tools to be used by their “SW companies”.
- In the case of global SE instructor preparation should include close cooperation between instructors of all participating institutions in establishing similar or complementary class goals, material coverage, grading policy etc.

- It is highly desirable that instructors have real-life experience in delivering SW products and services.
- Student final projects should be kept live for some time after the class in order to serve as their e-portfolios

REFERENCES

[1] "Software Engineering 2004": Curriculum Guidelines for Undergraduate Degree Programs in Software Engineering, The joint Task Force on Computing Curricula, IEEE Computer Society, Association for Computing Machinery, August 2004 http://sites.computer.org/ccse/SE2004Volume.pdf

[2] Charette, R. N., “Why Software Fails”, IEEE Spectrum, September 2005, pp. 42

[3] Somerville, Ian, "Software Engineering", 6th edition, Addison Wesley, 2000 (7th edition also applies)

[4] Torres R. J., “Practitioner’s Handbook for User Interface Design and Development”, Prentice Hall PTR, 2002 [Carmel 1999] E. Carmel : “Global Software teams” , Prentice Hall, 1999

[5]Krishna, S, Sahay,, S, Walsham G, “Managing Cross-Cultural Issues in Global Software Outsourcing”, Comm. Of ACM, April 2004, Vol. 47, No. 4.

[6] Gopal, A, Mukhopadhay, T, Krishnan, M, “The Role of Software Processes and Communication in Offshore Software development”, Comm. Of ACM, April 2002, Vol. 45, No. 4

[7] Carmel, E., ”Global Software teams”, Prentice hall, New Jersey, 1999

[8] http://segal.cs.uvic.ca/csc576b/