Teaching Software Engineering Subjects Using a Practical Oriented Approach at the University of Mumbai

http://dx.doi.org/10.3991/ijep.v3iS4.3218

Mayur S. Gondhalekar, Sachin M. Bojewar
Vidyalankar Institute of Technology, Mumbai, India

***Abstract*—The University of Mumbai is one of the oldest and renowned universities in India, catering to nearly 20,000 students in the Computer Engineering and Information Technology Engineering disciplines. This translates to approximately 12,000 qraduates entering the software industry. The structure of the coursework related to Software Engineering subjects - Software Engineering, Software Testing and Quality Assurance, Sofware Project Management and Software Architectures - relies heavily on theoretical classroom teaching. In this paper, we suggest changes to the structure of the coursework by shifting the focus to a project-oriented system, that will give the students a real-world-like experience of the Software Engineering subjects. Graduates with a higher level of practical experience of Software Engineering subjects is a win-win situation for everyone concerned, the software industry, the research institutes, and the students themselves.**

***Index Terms*—Software Engineering, pedagogy, project-oriented teaching methodology, curriculum design**

I. INTRODUCTION

The University of Mumbai, one of the oldest and renowned universities in India, caters to approximately 30,000 fresh engineering students every year across different majors[14, 21]. Syllabus wise, Computer Engineering and Information Technology Engineering are two similar yet different disciplines[13, 20]. Software Engineering is a common aspect of both majors. We would like to propose a modification of the current system, whereby students of both majors get to learn the subjects under the 'Software Engineering Umbrella': Software Engineering, Software Testing and Quality Assurance, and Software Project Management.

We would like to discuss a modification of the existing course structure, such that these subjects are taught in a practical-oriented, hands-on manner. This, we believe would give the student a strong understanding, making him better prepared for the IT industry and/or a Masters degree.

This paper is structured as follows: We describe the current system in place in II, the effectiveness of the current system in III, propose the new methodology in IV and finally conclude the paper.

II. CURRENT SYSTEM

The engineering degree courses at the University of Mumbai are four year courses, spanning eight semesters. Semesters 5 and 6 make up the Third Year of Engineering, while Semesters 7 and 8 make up the Final Year of Engineering. Each semester is typically 14 to 15 weeks long. Students follow a pre-set course structure, and are unable to select the courses of their choice except an elective each in the final year.

The University requires the students of the Final Year of Engineering (B.E.) to undertake a full year project in their respective disciplines, which we will refer to as 'Final-Year Project'. Software Engineering is introduced in the sixth semester of both, the Information Technology (I.T.) course as well as the Computer Engineering (C.E.) course. The intent is that the student learns the concepts and is able to derive practical knowledge by applying them to his Final-Year Project. As such, the student is expected to implement various techniques learnt in Software Engineering in his Final Year project. We find this to be of sound reasoning.

The Information Technology discipline aims to equip the student from a practical, job-oriented point of view, developing skills like Software Engineering, Computer Programming, Database and Computer Networking. The Computer Engineering discipline aims to equip the student also with the theoretical knowledge of computer science.

The Information Technology discipline requires the students to take subjects like Software Testing and Quality Assurance in semester seven, followed by Software Project Management in semester eight[13]. We agree with the choice of the follow-up subjects.

The Computer Engineering discipline, on the other hand, requires the students to take up Software Architectures directly in semester eight.

The objectives of the subject 'Software Testing and Quality Assurance' are to equip the student with a solid understanding of practices that support the production of

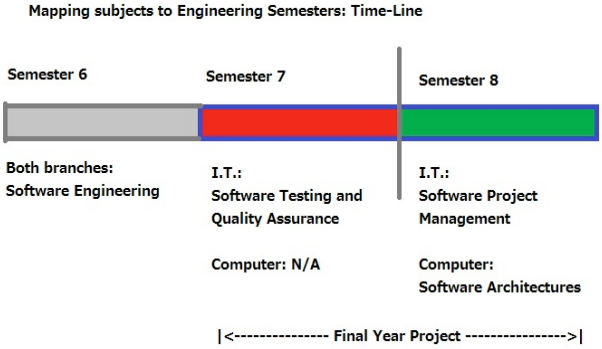

Figure 1. Mapping the Subjects to the Engineering Project Timeline

quality software, software testing techniques, life-cycle models for requirements, defects, test-cases and test results, different kinds of testing, and, quality models[13]. This subject goes into great technical depth regarding software quality.

Software Project Management focuses on familiarizing the student with the characteristics of projects, project management principles, risks, and management challenges. The students are expected to demonstrate competency in management of a project, scheduling and budgeting, and meeting project deadlines and goals.[13]

Software Architectures focuses on modeling techniques, design, implementation, deployment and system adaptation issues to enable the student to use the right tool for the job at hand.[20]

Currently, a lot of focus is given towards teaching the theoretical aspects in a classroom environment. Students are typically asked to prepare the documentation for an existing project as part of the two-hour per week laboratory sessions. There is a tendency to ask the students to 'study' and write about the various tools used in software testing. A lot has already been written about the deficiencies of such an approach in [2, 3, 5, 7, 8, 9, 10].

Such an approach tends to reduce the student's interest. Less than 10% of the students tend to apply these concepts to their Final-Year Project, without prodding from the guides. Typically, the only application that the students do, is to pick the correct software development process model for their project.

III. EFFECTIVENESS OF THE CURRENT METHODOLOGY

The students miss out on real-world like experiences of software engineering such as:

- interacting with clients
- misunderstandings that occur due to miscommunication
- the necessity of clear and complete undestanding
- laying out project timelines, milestones, and state charts
- timely completion of the project
- testing
- risk analysis and management
- and last but not the least, the importance of the above when one signs a binding contract.

Software Testing and Quality Assurance is correctly placed in the seventh semester for the Information Technology course. This carries forward from the testing phase introduced towards the end of the software engineering course. While the idea behind placing this course in the seventh semester is good, the current style of teaching does not quite achieve the ends. Students lack practical knowledge and that, unfortunately, leads them to do a basic level of testing towards the end of their Final-Year Project.

Software Project Management in misplaced in semester 8. Students are introduced to the principles of software project management after almost 75% of the project timeline is over, leading to a sense of "too late" and "what if". The authors conducted a survey of 120 students of the eighth semester Information Technology at their institute. 100% of the respondents felt that Software Project Management in semester 8 was too late. The consensus was that their project would have sailed in smoother waters had they been aware of project management concepts earlier in the time-line. 66% felt that the subject should be introduced in semester 7, while the rest opted for the subject to be in semester 6, along with Software Engineering.

Due to the hiatus of one semester after learning Software Engineering, Software Architectures seems like a jolt out of the blue for Computer Engineering students. The students are keen to get done with the Engineering degree, and, tend to focus only on getting their grades in Software Architectures. The subject, by itself, appears to be theoretical, and it seems to be introduced rather late to make any impact on the Final-Year Project.

IV. THE NEW METHODOLOGY

As the engineering course structure in the University of Mumbai is rigid, we feel the need to advocate a new methodology to teach 'Software Engineering Umbrella' subjects. We concur that Software Engineering is indeed the base or the pre-requisite, on which the student can build the knowledge of other processes and aspects of the "Software Engineering Umbrella". As a result, it would be prudent to leave Software Engineering in semester six for both disciplines. This would help lay the foundations and pave the way to undertake the year-long Final-Year Project, with confidence..

The following semester, we would recommend the introduction of both Software Testing and Quality Assurance, as well as Software Project Management, in Information Technology as well as Computer Engineering disciplines. This achieves the objective of creating a larger workforce that is well-versed in all the three aspects. The students aspiring to take up a Master's degree, would be well equipped to not only develop, but to test and manage more challenging projects.

Typically the students finalize the Final-Year Project approximate mid-way through the seventh semester, and spend the rest of the semester in documentation. Using our methodology, the learning of software testing and project management concepts would coincide with the half-way mark of the semester. The student can then apply the learnings to the Final-Year Project, to derive a much more rewarding experience.

In keeping with the theoretical aspects, we would like to retain Software Architectures for the eighth semester of Computer Engineering.

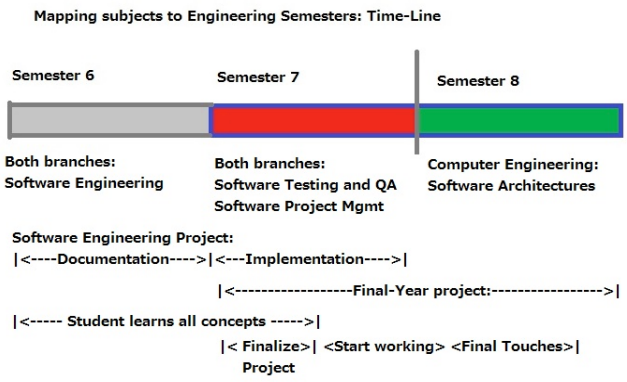

Figure 2. Software Engineering subjects mapped to the Final Year Project Timeline

We would also like to propose a radically different method to teach Software Engineering, Software Testing and Quality Assurance, and Software Project Management subjects.

Software Engineering must be integrated with a project. There could be classroom teaching using textbooks, personal experiences and case studies in the format suggested by Jianmin Zhang and Jian Li in [2]. Students would also take up a project as recommended by Yadav and Xiahou in [3]. Project group sizes of 5 are recommended as that figure is close to the actual size of a typical group in the software industry. One of the students can be elected as the group leader[3]. Students would have the option to follow any software development process model that they learn in the course.

These projects are to be handled by 'external agencies'. An 'external agency' is anyone who has knowledge about Software Engineering, but is not the 'current' teacher of the course. It could be a company external to the college, or teachers from any of the departments, so long as they are not teaching Software Engineering that semester. This gives the project a real-world feel, since an 'external client' is involved.

Students would benefit by experiencing first hand, the miscommunication, the risks involved, the thoroughness of planning and documentation required, the necessity of a buffer-zone (time and resources) and the importance of regular meetings. This phase could concentrate primarily on the initial documentation aspects. A similar method is followed at the University of Southern California, Software Engineering course (CS577A) by Dr. Barry Boehm.

We introduce Software Testing and Quality Assurance, and, Software Project Management in the seventh semester. The pattern of teaching could follow the recommendations of Gabriele Bavota, Andrea De Lucia, Fausto Fasano, Rocco Oliveto and Carlo Zottoli in [9] and Longjun Huang, Liping Dai, Bin Guo and Gang Lei in [10]. The theory could be taught in the classroom using case studies, while working on the implementation phase of the previous semester's project could be used to understand the shortcomings of working with partial knowlegde. This could act as the "too late" phase, and spur the student to handle the Final-Year Project with more interest and aplomb.

Additionally, implementing the previous semester's project will enable the students to get hands-on practice of Software Testing, whereas the Final Year project, being in the initial stages, gives students the chance to apply 'Quality Assurance' to the entire Software Development Life Cycle (SDLC).

Software Project Management is an experience dominated subject that cannot be learned by merely attending lectures[17,18,19]. It tries to ensure the successful implementation of software project by applying management experiences to the development lifecycle[10]. We advocate the Project Driven Teaching Model for Software Project Management as discussed at length by Longjun Huang, Liping Dai, Bin Guo, Gang Lei in [10].

Currently the Final-Year project is worked in groups of 3 or 4. The group size should be standardized to 5. Thus, the students would have an option to do the Final-Year project with the same group as that of Software Engineering, or to work on two different projects simultaneously. Here, the implementation phase of one project will overlap with the initial phase of the other project. Rather than look at this as an anathema, this is the perfect opportunity to experience a real-world scenario, because working on multiple projects simultaneously is often the norm in the software industry. Thus, through our proposed method, the student stands to gain not only great practical knowledge about the software engineering processes and principles but also experience life as it were in the industry, right in school.

The grading system for these subjects would follow the current pattern but split the marks equally between the final exam (theory) and the student's understanding of the concepts practically. The theory exam could be scored out of 50 marks, whereas the remaining 50 marks would be secured through the project presentation and the viva that follows. The candidate would be deemed to have passed if he secures the minimum required marks in each of the sections.

V. CONCLUSION

The current system of teaching software engineering at the undergraduate level at the University of Mumbai needs to undergo a change, with more focus on practical oriented learning, through significant software projects. The consequent emergence of graduate engineers, with significant practical experience and knowledge will benefit the software industry and the academia immensely.

REFERENCES

[1] Thomas Reichlmayr, "The Agile approach in an Undergraduate Software Engineering Course Project", IEEE November 54,2003, Boulder, CO 33'd ASEEIIEEE Frontiers in Education Conference

[2] Jianmin Zhang, Jian Li, "Teaching Software Engineering Using Case Study" IEEE 2010

[3] Sohan Singh Yadav, Jianbing Xiahou, "Integrated Project Based Learning in Software Engineering Education", 2010 International Conference on Educational and Network Technology (ICENT 2010)

[4] Dennis J. Frailey, "Experience Teaching Barry Boehm's Techniques in Industrial and Academic Settings", Proceedings of the 19th Conference on Software Engineering Education & Training (CSEET'06)

[5] Salamah Salamah, Massood Towhidnejad, Thomas Hilburn, "Developing Case Modules for Teaching Software Engineering and Computer Science Concepts", 41st ASEE/IEEE Frontiers in Education Conference, Rapid City, SD

[6] Prof. Suthikshn Kumar, "Topic: Innovative Teaching of Software Engineering: Practical Approach with Labs", 22nd Conference on Software Engineering Education and Training

[7] J Barrie Thompson, Helen M Edwards, "How to Teach Practical Software Quality Assurance, An Experience Report", IEEE 2000

[8] Chris Ho-Stuart and Richard Thomas, "Laboratory Practice with Software Quality Assurance", IEEE 1996

[9] Gabriele Bavota, Andrea De Lucia, Fausto Fasano, Rocco Oliveto, Carlo Zottoli, "Teaching Software Engineering and Software Project Management: An Integrated and Practical Approach", Software Engineering Education, ICSE 2012, Zurich, Switzerland

[10] Longjun Huang, Liping Dai, Bin Guo, Gang Lei, "Project-Driven Teaching Model for Software Project Management Course", 2008 International Conference on Computer Science and Software Engineering

[11] Philippe Kruchten, "Experience Teaching Software Project Management in both Industrial and Academic Settings", CSEE&T IEEE 2011, Honolulu, Hawaii.

[12] S.Efremidis, S.Retalis, N.Papaspyrou, E.Skordalakis, "Teaching software engineering through the net", Proc. of the 1st

International Conference on Software Quality Engineering, pp. 37-46, May 1997

[13] University of Mumbai, Information Technology Course Syllabus, "http://www.mu.ac.in/Information Techn. Sem. VII & VIII Rev.pdf"

[14] University of Mumbai website, "http://www.mu.ac.in"

[15] Michael G. Murphy, Ph.D., "Teaching Software Project Management: A Response-Interaction Approach"

[16] Andreas Bollin, Elke Hochmüller, Ladislav Samueli, "Teaching Software Project Management using Simulations - The AMISE Environment: from Concepts to Class Room Experience" (Their references 5,6)

[17] M. Shaw: Software Engineering Education: A Roadmap; in A. Finkelstein (ed): Future of Software Engineering 2000; ACM, 2000, pp 373-380

[18] J.B.Thompson: Software Engineering Practice and Education: An International View; Proc. SEESE'08, ACM, 2008, pp 95-102

[19] Per Runeson and Peter Isacsson, "Software Quality Assurance - Concepts and Misconceptions", IEEE 1998

[20] University of Mumbai, Computer Engineering Course Syllabus, "http://www.mu.ac.in/Comp. Engg. vii-viii rev.cour.pdf"

[21] Directorate of Technical Education website: "http://dte.org.in/approvedinstitues/CMS/Content_Static.aspx?did=40"

AUTHORS

Mayur S. Gondhalekar is Assistant Professor at the Vidyalankar Institute of Technology, Mumbai, India (gondhalekar.mayur@gmail.com).

Sachin M. Bojewar is Associate Professor at the Vidyalankar Institute of Technology, Mumbai, India (sachin.bojewar@vit.edu.in).

This article is an extended and modified version of a paper presented at the 2013 IEEE International Conference on Teaching, Assessment and Learning for Engineering (TALE2013), held 26-29 August 2013, Bali Dynasty Resort, Kuta, Indonesia. Submitted 30 September 2013. Published as re-submitted by the authors 01 December 2013.