Using Scrum for Software Engineering Class Projects

Ramrao Wagh
DCST, Goa University
Goa, India
e-mail:ramrao@unigoa.ac.in

Abstract – **Imparting industry relevant skills and knowledge for the graduating students in the field of Software Engineering is difficult but is necessary to make the students employable and productive right from the joining. With outdated curriculum and slow process of revision of syllabi it is difficult to achieve this objective. This paper discusses how one of the popular agile project management frameworks, Scrum can be taught and used to teach basic concepts of project management without necessitating revision of the syllabus. It also discusses the rise in motivation and interest level of students due to adoption of this approach. It also shows the flexibility of this agile approach to adapt to a situation different than a normal software development scenario in an organization.**

Keywords-Software Engineering Education, Agile, Scrum, Projects

1. INTRODUCTION

It is a common cry of industry that the graduates coming out of Computer science courses are not directly employable and as such lot of time and resources are wasted in (re) training them on the development concepts, tools, as well as methodologies. On the other hand, teaching institutes focus mainly on imparting theoretical skills without much emphasis on practice and in majority of the cases, whatever practical components are included are also outdated and irrelevant. One common excuse often cited by the institutes and faculty members is that it is time-consuming to change or adapt the curriculum according to the needs of the industry. In the Indian context, the Bachelor in Engineering in Computer Science Engineering or Information Technology(BE(CSE or IT) and Master in Computer Applications (MCA) programmes that are accredited by All India Council for Technical Education(AICTE) are the mainstay for providing IT manpower needed for the burgeoning IT Industry of India. The standard curriculum of both these programmes includes courses on Software Engineering in the later semesters.

A typical Software Engineering (SE) curriculum that is part of BE/MCA courses shows that it is highly theoretical. It aims to teach lifecycle and process concepts, requirements engineering, design, testing, maintenance, quality assurance and control and project management based on some standard text-books such as [1], [2], [3]. These books try to provide glimpses of each of this activity but fail to give any clear-cut guidance on practicing the principles that they explain nicely. Since the Software Industry is project-driven it is very much essential to teach project management in greater details but in reality, it is paid a lip service. Moreover, students are also not provided with opportunity to apply whatever little is taught to them by not offering any practical/case study component for SE courses. It is true that given the limited time available within a semester, it is difficult to apply the traditional project management approach to implement a course project. What is needed is a lightweight, adaptive approach that can be tailored to the limited time and resources available as well as that provides enough flexibility to define various project attributes and that clearly shows the project progress, facilitates team work and provides an exciting learning opportunity to the learners. This paper describes an approach where the SE and project management principles are taught using Scrum [4] as a project management framework by applying it on type of projects that are usually handled by software industry.

2. BACKGROUND

MCA programme of Goa University, Goa , India was introduced in 1987 with an intake capacity of 30 students by department of Computer Science and Technology. Since then it has produced around 600 software professionals with focus on developing business applications. Its alumni work in almost all major global and national software companies. MCA is a three year, six-semester programme which was promoted all over India by Department of Electronics (DoE) of Government of India in order to fill the gap of software professionals by allowing non-engineering graduates to pursue software development. Our MCA programme offers basic problem solving and programming courses as well as the other fundamental computer science courses such as DBMS, Networks in the first four semesters. Fifth semester has a

978-0-7695-4657-5/12 $26.00 © 2012 IEEE
DOI 10.1109/AgileIndia.2012.17

course on SE along with other courses. It has a separate theory (CS51) and laboratory course (PL57). Both the courses are taught by this author. The syllabus for these courses were also designed by the author three years back and has not been revised formally since then by University. The Sixth semester of MCA programme involves full-time project work as internship, preferably in an industry.

CS51 theory course syllabus is a typical SE curriculum with requirements, design, architecture, HCI, Testing, project management and configuration management as major topics. One of the main prerequisite for this course is theory and lab courses on OOAD (CS41 OOAD & PL46) in the previous semester in which UML/Java is introduced and use case based development approach [5] is taught.

One of the best practices of our MCA programme is that every CS course having a laboratory component has a mini-project associated with it apart from individual lab assignments. The mini-project is usually a group project wherein students have to implement and demonstrate the project working.

Since CS41 and CS51 are both taught by the author in the successive semesters, we usually carry the same project for both the semesters so that, design and sometimes a bare-bone implementation is submitted by students at the end of fourth semester and full-fledged implementation with testing is carried out in fifth semester. Since earlier the use case based approach was followed, it was observed that the teams usually have a comprehensive analysis, design and occasionally code and tests for each of the use case but the overall selection of which use cases to include in the implementation and how a team should carry out the various tasks required to implement the functionality were a grey area. As a result, it was difficult to monitor the project progress as well as to ensure that the team as a whole is involved.

3. DESIGNING THE COURSE WORK

This year the same practice was continued but the approach was changed. In the fourth semester, we had five groups composed of 5-6 students per group who were assigned separate projects. They had worked on use case modeling followed by analysis and design for a given problem. Every group was given a different problem related to business applications development. They were told that project will be implemented in the forthcoming semester. In the fifth semester course(CS51 and PL57), initial focus was teaching basic SE topics in theory course and advanced programming techniques and introducing development and project management tools in the lab course.

Accordingly, in theory classes, we started with introduction to SE with a customary look at various lifecycle paradigm from waterfall to agile, followed by sessions on requirements engineering using [6]. With this by using activities and exercises, various groups came up with a vision statement outlining the problem and requisite features to solve the problems. Simultaneously, in the lab, pair programming was introduced and design problems were given for which they have to submit the solutions online through CMS tool Moodle [7]. The focus of these lab sessions was on learning how to refactor the code [8], use of design patterns [9] and writing test cases using Junit [10]. Some of the most useful design patterns such as strategy, observer, adapter, composite, decorator, proxy, factory, singleton [9] were learnt by the students in context. Faculty approach was to give a problem. For which students will provide a solution which would be then gradually enhanced using an appropriate design pattern. They were also taught how to use Subversion [11] to manage the team development and build projects using Ant build [12] tool.

4. USING SCRUM FOR CLASS PROJECT

With these basic development abilities required for team development and building on their OO knowledge by using design patterns and refactoring they were introduced to agile approach with the use of resources such as slides and presentations on Scrum [4][13][14]. An excellent case paper on introducing Scrum at Yahoo! [15] was also provided to understand how scrum was introduced in the practical setting. The other supporting material such as [16], and scrum checklist [17] was also uploaded and circulated through Moodle.

After around one month of teaching days spent on above activities, the student groups were asked to come up with user stories [18] and prioritize them using a requirement workshop conducted in a two hour classroom session. Different team members assumed the roles of Scrum Coach, Customer and Product-owner and Scrum team members. Customer role was taken up by doubling one of the team members as Customer. The author served as a manager for the teams. Each group used its previous submission of earlier course and listed around ten user stories each with priority and estimate in story points to create a Product backlog.

The next task was to create a project wall to monitor the project progress. The classroom walls were used to stick project wall charts. Each group was given two chart-papers each and other material and they were asked to prepare the project chart boards. The first chart paper was used to define the sprint in which they were asked to draw columns for user stories planned, in progress and done [19] for a sprint. The second chart was used to draw a sprint burndown chart and to use remaining space to use for stories planned but could not be taken up during the sprint. Both these chart papers were pasted on classroom walls close to each other. They were asked to use Post-it notes to write user stories and tasks. Bigger sized post-it notes were used to write the story in the standard format and smaller post-it notes were used to define the tasks required to complete the story. They were asked to estimate the stories using story points. Planning poker was used to estimate the story points. Once this task was performed, they were asked to define the first sprint by including stories in the first column.

Here there was a major deviation from how scrum is used in real world to that in classroom setting. In scrum a sprint is about 2-4 weeks that includes all the working days within that period. But since, apart from SE courses, our students have to attend and do work related to many other courses, a sprint was defined as follows. Considering the remaining time of around two and half months for the semester we divided that time to accommodate approximately two sprints of one month each leaving out the scheduled examination days. The number of lab sessions available during this period was around 12 sessions of three hours each. So we defined a sprint as five sessions amounting to 15 hours of laboratory time. It was informed that they are free to work on the project at any other extra time if possible. Each team was asked to prioritize the user stories and then select user stories so that not more than 15 story points are taken for first sprint. This was done by equating 1 story point to 1 hour of lab work as a guess because it is not possible to know the team velocity in the first few sprints. The teams selected around 2-3 stories based on this criteria for the first sprint. They also created the sprint burndown chart on the second chart paper to draw with the ideal burndown rate drawn for 15 story points as sprint backlog total and five lab sessions as sprint duration. The entire exercise lasting for around two hours with intense group activity was thoroughly enjoyed by the students and they learnt major aspects of Scrum by doing this activity. All the concepts such as sprint, backlog, story, story point, burndown rate, task of the stories were understood by team members when they did this activity. They also learnt one of the important requirements of scrum i.e. self-organizing teams as everyone played their role to complete this task of creating project wall chart by contributing in whatever way they can without anyone allocating a specific work to each one of them.

There was a marked change in the approach to project from this moment onwards in all the teams. As the project wall charts were mounted in classroom where they attend their regular classes, it was a constant reminder to them to think and work on the project.

The first lab session after this activity was devoted to the first daily stand-up meeting of each team. Each team conducted this meeting facing the project wall chart. They moved the task of the selected stories as they took up that task. They learnt how to volunteer and own responsibility for the work. With this, teams set into pairs or worked individually by sitting close to each other in laboratory to start implementing the task of the user stories selected by them in the first sprint. This practice of daily stand-up was adhered for every lab session later.

The first sprint is usually characterized by creating and configuring the necessary infrastructure and tools required for the project. All the teams were asked to use Eclipse [20] and implement a web-based implementation using Java EE. The teams created the projects and checked in Subversion repository and also worked upon other aspects such as use of DBMS etc.

As the teams went on implementing the user stories in the first sprint backlog, various issues- technical as well as non-technical started interfering with the project implementation. On the technical front, the teams were asked to implement project using Eclipse as the basic IDE and were encouraged to try frameworks such as Hibernate and Spring. The project management aspects were taken care of through Scrum although as mentioned earlier, a vision document was prepared to serve as a guiding document. Initially, the teams struggled to learn Spring [21] and Hibernate [22] framework on Eclipse and spent lot of time. But, due to lot of problems related to configuration faced by the teams as there was no expertise available to sort them out, all the teams finally abandoned the idea of using Spring and Hibernate and adopted a plain JSP based approach.

On the project management front, instead of earlier planned two sprints, we had to focus on completion of only one sprint as lot of sessions were cancelled due to some other events and exams. Only one team reported that they worked on two sprints by working extra hours outside scheduled lab hours.

5. EVALUATING THE OUTCOME

An online feedback of the students was taken by creating a feedback form to be filled anonymously by the students. The objective of the feedback was to understand the effectiveness of the approach and to know how much they have learnt about Scrum. There were eight questions to which the students were supposed to chose appropriate responses. All the 29 students responded to the online feedback made available on course page on university moodle site. It is seen that around 85% students felt that there were improvements in the way a project was managed due to use of Scrum. 85% responded that they were provided with adequate knowledge to use Scrum. However, with regard to the allocation of scrum roles within a team, the opinion were equally divided with 45% saying roles were properly defined while 45% were not sure of role allocation. Almost 80% felt that teamwork was improved due to use of scrum. To the question about the knowledge level of faculty about Scrum, all the students felt that the faculty member has adequate knowledge of scrum framework. To the question whether team members will use Scrum in future projects, 46% responded that they will always use it while 46% said that they will use Scrum provided management asks them to use it. Out of the various Scrum activities/artifacts, daily standup with 38% was the most popular technique followed by project wall chart (28%), sprint planning (19%) and user stories and estimation (17%).

The project demonstration was conducted by every team as part of the end semester exam of the lab course. Each team reflected upon how they used Scrum to manage the project and demonstrated the working of the project as completed as part of the sprint(s). The presentations were jointly evaluated by the faculty and external examiner appointed by university. The project demonstration clearly showed that using Scrum, the features that were developed by the team was complete and cohesive compared to splintered development work that used to be accomplished by following traditional approach. Although the number of user stories that could be completed were very small (2~6) but still students realized the importance of focusing on achievable work within a sprint.

6. LESSONS LEARNT AND FUTURE WORK

Focus of this paper was on how to use Scrum by adapting it to the classroom situation wherein such difficulties and impediments come up regularly. Our earlier approach of giving students a free hand in deciding on how to complete the assigned task was not yielding good results and most of the time, students used to do some last minute work before the deadline and present the same. With Scrum-based approach, they are now clear about what they will achieve in a given time-frame and are able to achieve better and visible progress. But the major gain was rather than study project management in a theoretical ways, they understood lot of important concepts related to project management such as defining the scope of the project, estimating project duration and outcome, managing project though stand –up meetings.

One of the major purpose of this exercise was to show how without officially revising the curriculum, how modern approaches can be accommodated in the teaching-learning process using the activity based approach.

We tried to use Scrum by following the bare essential things and modifying and adopting it to suit the classroom limitations but highlighting the best aspects of scrum. In the future more quantitative data about how effective this approach is will be collected and analyzed. We wish to repeat and enhance the experiment by spreading this exercise over two semesters instead of just one semester so that we can accommodate more sprints. That will enable us to measure the velocity properly, introduce the additional Scrum techniques such as retrospectives and to manage the Scrum roles much more effectively.

The students who participated in this exercise are in a better position to appreciate and embrace agile methodologies which are increasingly adopted by the industry.

REFERENCES

[1] Pressman Roger, Software Engineering: A Practitioners Approach, Sixth Edition. McGraw Hill, 2005

[2] Sommerville Ian, Software Engineering, 9th Edition. Addison Wesley, 2007.

[3] Jalote Pankaj, An Integrated Approach to Software Engineering, Third Edition. Springer 2008.

[4] Schwaber, Ken. Agile Project Management with Scrum. Redmond, WA: Microsoft Press, 2004.

[5] Larman Craig, Applying UML and Patterns, Third Edition, Prentice Hall, 2003.

[6] Leffinwell Dean, Widrig Don, Managing Software Requirements: A use-case Approach. Second Edition.Addison Wesley.

[7] Moodle homepage www.moodle.org

[8] Fowler Martin, Beck Kent, Brant John, Opdyke William, Roberts Don, Refactoring: Improving the Design of Existing Code. Addison Wesley, 2003.

[9] Erich Gamma, Richard Helm, Ralph Johnson and John Vlissides, Design Patterns: Elements of Reusable Object-Oriented Software. Addison Wesley, 1994.

[10] Junit homepage: www.junit.org

[11] Subversion www.subversion.tigris.org

[12] Ant Build www.ant.apache.org

[13] Presentation on Scrum by Mike Cohn http://www.mountaingoatsoftware.com/presentations

[14] Scrum guide www.**scrum**.org/**scrumguides**

[15] Benefield Gabrielle, Rolling Out Agile In a Large Enterprise, Proceedings of the 41st Annual Hawaii International Conference on System Sciences (HICSS 2008)

[16] Kniberg Henrik, Scrum and XP from the Trenches (Enterprise Software Development), Addison Wesley.

[17] Scrum checklist available on http://www.infoq.com/minibooks/scrum-checklists

[18] Mike Cohn, User stories User Stories Applied: For Agile Software Development, Addison Wesley.

[19] Mike kohn, Agile Estimating and Planning, Addison Wesley.

[20] Eclipse www.eclipse.org

[21] Spring www.springsource.org

[22] Hibernate www.hibernate.org