Web Enabled Learning Tool for Software Requirements Analysis

N. Pratheesh
Department of Computer Applications
School of Computer Science and Engineering
Bharathiar University,
Coimbatore – 641 046, India
pratheesh_n@hotmail.com

T. Devi
Department of Computer Applications
School of Computer Science and Engineering
Bharathiar University,
Coimbatore – 641 046, India
tdevi5@gmail.com

P. Prithvi Vidhya
Department of Computer Applications
School of Computer Science and Engineering
Bharathiar University,
Coimbatore – 641 046, India
prithvi.indira@gmail.com

Abstract – Software Engineering courses are core elements of the Computer Science curricula. While the main aim of such courses is to give students practical industry-relevant software engineering knowledge, often such courses fall short of this important target due to lack of industrial experience and support infrastructure, degenerating the course into "one big coding assignment". Therefore, it is necessary to design and develop better infrastructure support for teaching such courses. This would benefit instructors and students world over. The authors are trying to create knowledge and tool infrastructure for the benefit of instructors and students of Software engineering courses. This paper details the problems in teaching software engineering and describes the prototype development for teaching requirements analysis phase of software engineering.

Keywords: Tool Support Environment, Software Engineering Education, Requirements Analysis, Blended Learning, Collaborative Learning, Simulation Software Tools

1. INTRODUCTION

Computer software is the single most important technology worldwide. Software has enabled the creation of new technologies such as genetic engineering, the extension of technologies such as telecommunication, and the demise of older technologies such as printing industries. When computer software succeeds – when it meets the needs of the people who use it, when it performs flawlessly over a long period of time, when it is easy to modify and even easier to use – it can and does change things for the better. But when software fails – when its users are dissatisfied, when it is error prone, when it is difficult to change and even harder to use – bad things happen. All want to build software that makes things better, avoiding the bad things that lurk in the shadow of failed efforts. To succeed, need discipline when software is designed and built [15].

2. TEACHING SOFTWARE ENGINEERING (TSE)

Software engineering education gets more importance in the current digital world. Teaching software engineering is extremely a very difficult task, because it contains huge areas of knowledge such as requirements analysis, software solution architecture, testing, project estimation and project management. To deliver the knowledge of whole areas of software engineering to the students, academic institutions face the problems as allocated hours are not sufficient [7, 16]. Presently teaching software engineering follows the traditional teaching approach and provides the theoretical knowledge through lectures and performance based assessments i.e., course is divided into number of modules and end of each module, there are assessments and there is a final assessment at the end of the course [7, 18]. Academic institutions give more importance to teach this subject and try to produce the qualified software engineers to satisfy the IT industry's contemporary demands but the gap is not filled [5].

Normally software engineering syllabus does not cover all aspects, only provides the theoretical knowledge to the selected area of the software engineering, because of the allocated time and most of the students failed to have the practical implementations, very few students get practical knowledge when they implement the mini project in this frame [4, 7]. In this academic frame, students can get their theoretical knowledge properly but they fail to fulfill their practical knowledge, due to this, they fail to handle the real time projects [1, 10, 16]. Software engineering techniques are very important when working on large projects in team, specially project maintenance and extension are the real goals for the success of the project. This mini project experience cannot provide the working experience to handle the real world problem in software engineering and the important concepts such as quality of testing, cost estimation, etc., because students fail to apply the concepts and use such tools for their developed projects, they concentrate

Print-ISSN: 2231-2048 e-ISSN: 2231-0320

to complete the end product as early and obtain the higher grades in their semester examination [2, 7, 16].

Recent trends and studies exhibit that the lack of adequately trained professionals will be a major roadblock in sustenance and further growth of this industry in India. This lacuna is a direct resultant of poor Software Engineering (SE) education and training infrastructure in the country [7]. The graduate program syllabi in Computer Science and SE are not structured or standardized across the country, there has to be an essential training component that will ensure that a minimum level of expertise for a minimum level of skills achieved by all entry level recruits [5, 7, 16]. Currently, only 25% of the technical graduate students are considered to be directly employable by the industry [5, 7]. Since the demands are high and the supply is quite low, the hit rate (ratio of number of actual recruitments to the number of job applications) is around 5-10% for major software services companies [5, 7, 16].

3. IMPORTANCE OF SOFTWARE REQUIREMENTS ANALYSIS

The primary measure of victory of a software system is the degree to which it meets the purpose for which it was proposed. Software analysis is the process of discovering that purpose, by identifying stakeholders and their requirements, and documenting these in a form that is amenable to analysis, communication, and subsequent implementation. Requirements analysis is the stem of software engineering concerned with the real-world goals for, functions of, and constraints on software systems [14, 15, 17]. It is also concerned with the association of these factors to precise specifications of software behavior, and to their evolution over time and across software families. Requirements analysis reflects the importance of real-world goals that encourage the development of a software system and represent the "why" as well as the "what" of a system [15, 17].

Requirements analysis is a software engineering task that bridges the gap between system level requirements engineering and software design; therefore it is very important concept in the software development process, because of this reason, in software engineering syllabi, requirements analysis takes important role in undergraduate and postgraduate studies [9]. Current study pattern of requirements analysis includes traditional teaching methods and provides the wide knowledge of theoretical concepts to the students, but it fails to give real world experiences them. Students handle the practical problems to satisfy their academic level requirement but not that matches the industry standard. Therefore students struggle when they try their hands into the industry environment [7, 12].

Textbook definitions of requirements analysis refer to the creation of cost-effective solutions to practical problems by applying scientific knowledge but it is an essential part of an engineering process, being the part concerned with anchoring development activities to a real-world problem, so that the appropriateness and cost-effectiveness of the solution can then be analyzed [14, 15, 17]. Requirements engineering refers to the idea that specifications themselves need to be engineered, and requirements analysis represents a series of engineering decisions that lead from recognition of a problem to be solved to a detailed specification of that problem [7, 12, 14]. Software cannot function in isolation from the system in which it is implanted, and hence requirements analysis has to encompass a systems level view.

Requirements analysis is a branch of software engineering, whose vital goal is to deliver some systems behaviour to its stakeholders [3, 4, 18]. Viewed at the systems level or the software level, requirements analysis is a multi-disciplinary, human-centered process [15, 14, 17]. The tools and techniques used in requirements analysis draw upon a range of disciplines, and it may be expected to master skills from a number of different disciplines [8]. Theoretical computer science provides the framework to assess the feasibility of requirements, while practical computer science provides the tools by which software solutions are developed. Software engineering still lacks a mature science of software behaviour on which to draw, requirements analysis need such a science in order to understand how to specify the required behaviour of software, work on characterizing systems, identifying their precincts, and managing their development life cycle[2, 14, 18].

4. E-LEARNING AND ONLINE LEARNING

The emergence of the Internet and advances made in information and communication technology as well as the technological advances made in multimedia, personal computers, and networking had driven the development of distance learning in the information age. The need for "anytime, anywhere" learning has led to the development of e-learning, also known as web-based learning or online learning, which uses telecommunication technology to deliver information for education and training [11]. Online learning is a sub-set of flexible teaching and learning that seeks to provide greater access to learning for all students.

Online learning atmosphere is one that goes beyond the replication of learning events that have traditionally occurred in the classroom and are now available through the Internet. It provides for different ways of learning and the construction of a potentially richer learning environment that provides for fresh approaches to learning, caters for different learning styles as well as allowing for

greater diversification in learning and greater access to learning. An online learning environment can complement a traditional face-to-face learning environment or it may provide a complete learning package that requires little face-to-face contact.

E-Learning comprises all forms of electronically supported learning and teaching. The information and communication systems, whether networked or not, serve as specific media to implement the learning process. The term will still most likely be utilized to reference out-of-classroom and in-classroom educational experiences via technology, even as advances continue in regard to devices and curriculum [1]. E-learning is essentially the computer and network-enabled transfer of skills and knowledge to the user. E-learning applications and processes include web-based learning, computer-based learning, virtual classroom opportunities, and digital collaboration and content is delivered via the internet, intranet/extranet, audio or video tape, satellite TV and CD-ROM [1]. It can be self-paced or instructor-led and includes media in the form of text, image, animation, streaming video, and audio. E-learning is used to describe any type of learning environment that is computer enhanced and multiple technologies that can be employed in e-learning. Distance learning is something that has evolved from e-learning and it is used to describe a learning environment that takes place away from the actual traditional classroom.

5. WEB ENABLED LEARNING TOOLS

E-learning is one of the promising desires of the information era. That provides a lot of potential to the distance learning. A system that blends virtual realities and e-learning provides a natural and interactive environment to students. From student's viewpoint, the technicalities of on-line learning are as straightforward as logging on to the Internet. To run most web enabled learning tools, the learner will need a Pentium-class PC with the latest version of operating system, adequate random access memory and a modem that operates at 56 kb/s or higher. Some of the learning tools are: Self-Regulated Learning (SRL), ARGUNAUT, Cognitive Tutor Algebra (CTA), Extensible 3D (X3D) and OASIS.

5.1 Self-Regulated Learning (SRL)

Self – Regulated Learning (SLR) contains the motivational elements to define relevant learning goals and has enough motivation to get occupied in proper collaborative learning and knowledge edifice deeds to reach these ambitions, reflect upon and share how to acquire the obligatory knowledge, so there would be less harass for others in need of the same competencies. SRL in an organizational context should also believe that in order to perform inherently stirred learning [11].

5.2 ARGUNAUT

This web enabled learning tool provides the feedback to a teacher so that he or she can help students stay on topic, brings out contributions from all members of the collaboration groups, proposes the use of supported avers and arguments, and generally steers the learners toward fruitful discussion and collaboration. Yoke of moderating multiple, simultaneous e-discussions i.e., supporting many groups of students at the same time, too knotty for individual teachers, ARGUNAUT attempts to summarize the students' discussions and alert teachers to critical aspects and events in the e-discussions. The ARGUNAUT system provides the teachers with online, automated feedback regarding important aspects and characteristics of each discussion, explicitly focusing attention on events or situations that may require the teacher's intervention or support [13].

5.3 Cognitive Tutor Algebra (CTA)

Cognitive Tutor Algebra (CTA) is a widely-used tutoring system for mathematics instruction on high-school level. It wraps dissimilar aspects of algebra learning such as linear equations and inequalities. To provide adaptive tutoring, the CTA appraises the learner's problem-solving actions by comparing them to a cognitive model of successful learner performance, represented using a set of production rules. If an error is detected, the CTA immediately marks it as incorrect and provides context-sensitive feedback [13].

5.4 Extensible 3D (X3D)

One of the most recent applications of virtual reality is the interface to E-learning applications. Using this technology, it is possible to get a sense of a three dimensional environment of such web sites equipped with 3D object viewing. When learners work with a 3D viewing capable browser then the powerful of 3D environment appears in the capability to view the whole milieu in 360-degree with the ability to zoom your scene, and quickly navigate through the assorted places in your world and viewing your world through different viewpoints. The effect of 3D objects modeling appears in increasing the learner attention and interactivity with the objects as in real world. For providing immersive environments (virtual classes and labs) where learners can experience the steering of virtual milieu is both 3D and interactive, the standard language for designing such immersive environments is X3D [6].

5.5 OASIS

This learning tool includes a large questions database and server-side program that delivers questions, marks student responses, provides prompt feedback and records students' activities,

and well suited to student-centered and large-class learning. Students can practice each question until satisfied that they have mastered the particular skill, situation, or concept. The answers for all numerical variations of each question have been previously premeditated and are stored in the question database. Marking is very fast as it generally involves comparison rather than calculation. However, when multipart questions are marked consequentially, then some calculation is involved during marking [6].

6. PROPOSED SYSTEM

The traditional method of teaching Software Requirements Analysis includes the activities such as giving Lectures, Presentations, Group Discussions, Seminars and Illustrations where it places the yoke of encourage learning fully on the teacher, unless it is integrated with other techniques. This may not be suitable for the different learning rates of the partaker in every field. E-Learning is beneficial to academic institutions and to all types of learners and affordable, conserves time, and fabricates reckonable results. By this proposed prototype, students can ascertain the requirements elicitation techniques, methods for analyzing them, and making prototypes for the design phase devoid of the help of a teacher or instructor. This proposed tool is used to teach software requirements elicitation phase and leads the trouble-free way to educate the software engineering through online.

Proposed system facilitates the students to learn the techniques in elicitation and analysis of the requirements and ways in prototyping a model which will be the base for the next phase of software development. Video clips of requirements elicitation technique, requirement management and how to get requirements from the stakeholders enable better understanding of the concepts and quiz allows to compute the self understanding level. Students can engage in online discussions with fellow classmates enabling them to bounce ideas off each other in a way that might not happen if forced to rely on face-to-face study groups. Online discussion boards and chat rooms also offer greater accessibility for all students. Students can post comments and questions and participate in class discussions at their own convenience. Improvement in quality and convenience of communication improves student contentment which turns to lead to better outcome and sensation. This tool makes the students to learn the requirement analysis techniques in an efficient way and also attrition in distance education, an eminent problem would diminish by improving the flexibility and quality of communication through this tool.

7. RESULTS AND DISCUSSIONS

Web enabled learning tool for teaching software requirements elicitation techniques and analysis developed to promote the students to become good software developers. This tool has been designed in three frames format: title located in the top frame, heading and its sub headings of the software requirements analysis positioned in the left frame, and information for the specific link in right. By this tool, students can learn the requirements elicitation techniques, all the techniques used for requirements elicitation, methods of application, methods for analyzing them, and making prototypes for the design phase without the help of an instructor. This is a very good tool for the self-learners from software engineering or non-software engineering background to get the knowledge about software requirements analysis.

Figure 1: Requirements Analysis

Figure 2: Requirements Elicitation for Software

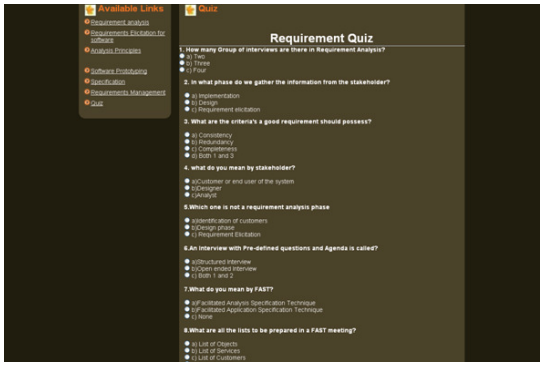

Figure 3: Quiz

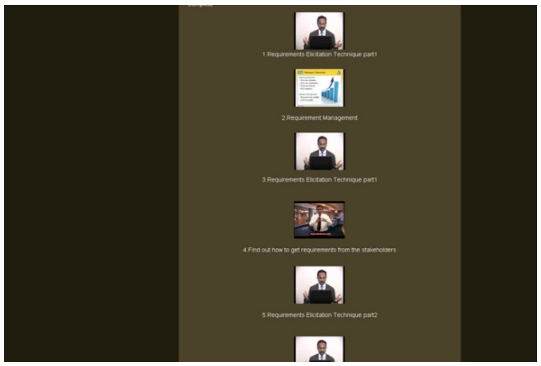

Figure 4: Video Clips

Fig.1 shows the screen shot of the requirements analysis. It articulates the concept of the requirements analysis and its process. Fig.2 provides the information about the requirements elicitations for software and illustrates all the sub headings. User can garner the knowledge from the relevant links. Fig.3 shows the quiz related to software requirement analysis. It contains twenty five multiple choice questions. This helps to appraises the knowledge of the individual and make them more recognizable in software engineering.

Fig.4 embraces some real time practical video clips relevant to the requirements analysis such as requirements elicitation techniques and requirements management which facilitate the beginner to fulfill in software engineering especially how to acquire the requirements from stakeholders for the successful product development.

8. CONCLUSIONS

This web enabled tool helps to motivate the students and increase their learning capacity in software engineering especially in requirements analysis. This proposed tool is cost effective, and can be easily accessed from anywhere at any time. The prototype is based on the teaching technique and increases the student's understanding power. Student's interest gets increased and they are attracted towards this tool. This proposed tool is effective more constructive not only to the software engineering students but also to non-software engineering students those who would like to scrutinize the software engineering especially in requirements analysis.

REFERENCES

[1] Brendan Quinn, Leonor Barroca, Bashar Nuseibeh, Juan Fernández-Ramil, Lucia Rapanotti, Pete Thomas, and Michel Wermelinger, "*Learning Software Engineering at a Distance*", IEEE Computer Society, IEEE, 2006.

[2] Carlo Ghezzi, Dino Mandrioli, "*The Challenges of Software Engineering Education*", ICSE, St. Louis, Missouri, USA, ACM, 2005.

[3] Christine Mingins, Jan Miller, Martin Dick and Margot Postema, "*How We Teach Software Engineering*", Monash University, Melbourne, Australia.

[4] Francisco S. Marcondes, Hamilton J. Brumatto, Eloiza H. Sonoda, Luiz C. Barboza, Jefferson Zannuto "*Problem on Software Engineering Learning: Domain Engineering*", Sixth International Conference on Information Technology: New Generations, IEEE, 2009.

[5] GVB Subrahmanyam, "*A Dynamic Framework for Software Engineering Education Curriculum to Reduce the Gap between the Software Organizations and Software Educational Institutions*", 22nd Conference on Software Engineering Education and Training, IEEE, 2009.

[6] H. M. Abdul-Kader, "*E-Learning Systems in Virtual Environment*", IEEE, 2008.

[7] Kirti Garg, Vasudeva Varma, "*Software Engineering Education in India: Issues and Challenges*", 21st Conference on Software Engineering Education and Training, IEEE, 2008.

[8] LIU Jianguo, "*Combination of Research and Teaching in Software Engineering Education*", WASE International Conference on Information Engineering, IEEE, 2009.

[9] Manar Abu Talib, Adel Khelifi, and Leon Jololian, "*Secure Software Engineering: A New Teaching Perspective Based on the SWEBOK*", Interdisciplinary Journal of Information, Knowledge, and Management, Volume 5, 2010.

[10] Melody M. Moore, "*Software Engineering Education*", IEEE SOFTWARE, 2002.

[11] Melody Siadaty, Jelena Jovanović, Kai Pata, Teresa Holocher-Ertl, Dragan Gašević, and Nikola Milikić, "*A Semantic Web-enabled Tool for Self-Regulated Learning in the Workplace*", 11th IEEE International Conference on Advanced Learning Technologies, 2011.

[12] Michael Gnatz, Leonid Kof, Franz Prilmeier, Tilman Seifert, "*A Practical Approach of Teaching Software Engineering*", 16th Conference on Software Engineering Education and Training, IEEE, 2003.

[13] Pierre Tchounikine, Nikol Rummel, and Bruce M. McLaren, "*Computer Supported Collaborative Learning and Intelligent Tutoring Systems*", Advances in Intelligent Tutoring Systems, SCI 308, pp. 447–463, Springer-Verlag Berlin Heidelberg, 2010.

[14] Ralph R. Young, "*The Requirements Engineering Handbook*", Artech House, 2003.

[15] Roger S. Pressman, "*Software Engineering-A Practitioner's Approach*", 5th edition, McGraw Hill Higher Education, 2001.

[16] Rupa Mahanti, P. K. Mahanti, "*Software Engineering Education from Indian Perspective*", 18th Conference on Software Engineering Education & Training (CSEET'05), IEEE, 2005.

[17] Sommerville, "*Software Engineering*", Pearson Education India, 2009

[18] Thomas B. Hilburn, Massood Towhidnejad, Sumeera Nangia, and Li Shen, "*A Case Study Project for Software Engineering Education*", 36th ASEE/IEEE Frontiers in Education Conference, IEEE, 2006.