US-Scrum: A Methodology for Developing Software with Enhanced Correctness, Usability and Security

Usman Rafi, Tasleem Mustafa, Nayyar Iqbal, Waseeq-Ul-Islam Zafar

Abstract— In this modern era customized software especially web applications are given more importance because of global and simultaneous access. Software quality is an important concern of the software developers and customers. Certain number of software quality attributes such as availability, integrity, interoperability, correctness, maintainability, performance, reliability, reusability, scalability, security, testability and usability are defined for quality assessment. Agile software development methods though flexible in coping with changes and organize the software development around functional requirements. Due to advancements in web technology security threats and usability issues have also increased. Agile involves the limelight for ensuring correctness of software while ignoring important quality attributes like security and usability. The purpose of this research is to propose a hybrid model based on Feature Driven Development (FDD) and Scrum principles to accommodate quality focus in terms of correctness, security and usability. The proposed model is organized around multiple sprints, specialized managers and specialized teams. The proposed model provides the benefits of being more user focused and increased customer satisfaction by employing comprehensive verification and validation process.

Index Terms— US-Scrum, Security in Scrum, Usability in Scrum, Scrum Quality Enhancement, Hybrid Agile Model, Specilaist Roles in Scrum, Hybrid Scrum.

1 Introduction

Software applications are of chief importance for variety of users because software facilitate daily operations of business. A software helps users in performing everyday tasks quickly, efficiently, effectively and accurately. Two broad categories of software are general purpose and customized. General purpose software applications are developed for general public. Users belonging to different walks of life use them in order to perform their tasks in an efficient manner. Customized software are developed for specific users or organization. These software are built on requirements of end users and organization. Software are built by following a predefined sequence of steps that breaks down whole task into certain steps and work products. Collection of such steps is named as System Development Life Cycle (SDLC). System Development Life Cycle (SDLC) can also be termed as Information Systems Development Cycle (ISDC). All modules of the information system could be included under the rubric System Life Cycle (SLC). Two methods were used beneath System Life Cycle (SLC) rubric. The first method was traditional Waterfall model that developed a system following sequential phases and provided no iteration. The second method was Prototyping and it included the phases of waterfall model and it allowed looping back to an earlier phase upon changes [1].

- *Usman Rafi, Research Scholar, Department of Computer Science, University of Agriculture, Faisalabad, Pakistan. E-mail: usmanrafimscs@gmail.com*
- *Tasleem Mustafa, Chairman, Department of Computer Science, University of Agriculture, Faisalabad, Pakistan. E-mail: tasleemustafa@uaf.edu.pk*
- *Nayyar Iqbal, Lecturer, Department of Computer Science, Univeristy of Agriculture, Faisalabad, Pakistan. E-mail: nayyar_iqbal@hotmail.com*
- *Waseeq-Ul-Islam, Lecturer, Department of CS and IT, University of Sargodha Lyallpur Campus, Faisalabad, Pakistan. E-mail: waseeq_islam@yahoo.com*

The most critical and important phase of System Development Life Cycle (SDLC) is requirements. This phase allows developers to gather broad level and detailed level requirements of the software. Moreover, software is accepted by customer only if it fulfils all the stated and agreed requirements. All other software development phases are based on this phase. Functional Requirements (FRs) and Non-Functional Requirements (NFRs) are major types of requirements.

As a result of advancement in telecommunication technologies, internet evolved. Internet is collection of various sub networks spread across the world and is based on client server architecture. E-Commerce has emerged a great deal because of development of Hyper Text Transfer Protocol Secure (HTTPS) before which trading partners were reluctant to use internet as a medium of trade due to security threats.

Desktop applications and Web applications are common software application types. Desktop applications are single user applications. These applications installed and used on a single computer. This is its major drawback. Web applications are preferred these days because of their scalability to get accessed by multiple users globally and simultaneously.

Because of world wide access, web applications are becoming popular. In order to implement web applications, web servers and database servers are used. Web servers are identified by using domain names and associated Internet Protocol (IP) addresses. Users termed as clients, access web servers using web clients or web browsers. Web clients don't have names instead these are identified using Internet Protocol (IP) addresses. Web servers contains web sites and web pages. Web pages are developed using Hyper Text Markup Language (HTML). Web clients send requests to web servers and receive responses in the form of web pages. This is done with the help of Hyper Text Transfer Protocol (HTTP) which is used as request-response protocol. Transmission Control Protocol (TCP) provides a way of communication for servers and clients [2].

Software development process must ensure quality in the developed product and it is ensured by certain software quality attributes.

Section 2 discusses different software quality attributes for the study. Section 3 describes some of popular agile methods. Section 4 is organized around problems in agile software development. A model is proposed and its potential benefits are discussed in section 6 and section 7 respectively.

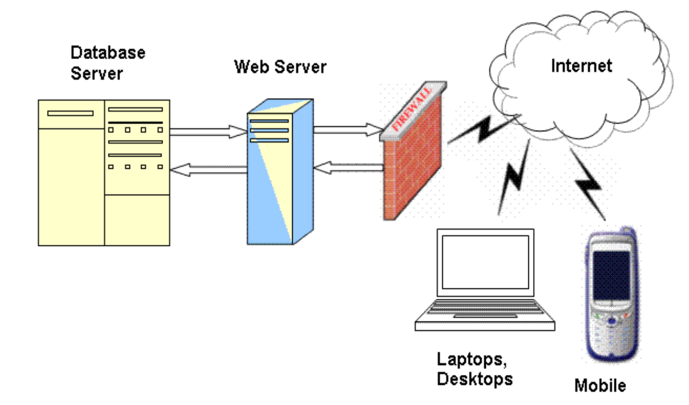

Fig. 1. Web Application

2 Software Quality Attributes

According to ISO/IEC 9001 quality is defined as the capability of a product in conforming to requirements and quality standards. Software Quality is just actually conformance of developed software with functional requirements, non-functional requirements, domain requirements, performance requirements, certain documented standards and other related implicit characteristics. There is considerable difference between quality focus of customers and developers. For facilitating this focus and measurement, a hierarchal model comprising of various quality attributes is defined. Quality attributes have impact on software product and hence are of major concern of software engineers. It is impossible to fulfil all the attributes. Always trade-off is made in deciding attributes.

Quality attributes define the extent to which different characteristics possessed and presented by the software after its implementation. Entire software development process is organized to ensure that quality attributes are exhibited by developed software. Development quality requirements and Operational quality requirements are major types of quality attributes. Development quality requirements comprise of flexibility, understandability and maintainability. Performance, security and usability are Operational quality requirements [3].

Correctness, Security and Usability are the quality attributes for the concerned study.

2.1 Correctness

Correctness is ability of a software system to meet exact needs in performing different tasks in accurate manner. Correctness ensures that the software meets functional specifications of software.

Correctness of a software was guaranteed by comprehensive testing. After conducting engineering practices in developing software, testing is carried out. Testing helps in detecting and removing software defects. In this way it improves software performance and reliability [4].

2.2 Security

Security measures the capability of a system in lowering the probability of malicious attacks, prevention of unauthorized access/modifications/disclosure of information and ability to withstand the attack if it occurs. Authentication, authorization, encryption, auditing, and logging are some of features that are used to implement security for security concerns like Confidentiality, Integrity and Availability (CIA).

Because of rapid growth of internet, information dysfunction resulted in information leakage and financial loss. To secure vital information, information security management is of major concern. In order to meet this objective, a certification system named as Information Security Management System (ISMS) has been in conduction in Korea since 2000. Procedures and processes to fulfil the purpose of security, confidentiality and integrity are established and documented in this certification [5].

2.3 Usability

Usability measures the ease with which the software system could be learned, understood, remembered and used by its users. It deals with user experience of software interfaces. Software interfaces should be designed in such a way to have minimum data entry screens, minimization of effort and maximization of easiness.

Usability is an important quality characteristic of software because it measure quality of use. Quality of use is concerned with ease of use, ease of learning and attractiveness. Usability also refers to the extent of efficiency, effectiveness and satisfaction of user in using software interfaces. Productivity improvement, uplifting of team morale and reduction in user training costs are some major benefits of usability [6].

Software Quality Assurance (SQA) is a continuous executing process throughout the System Development Lifecycle (SDLC) for ensuring the quality of software products. Software Quality Assurance (SQA) process checks that the product conforms to its requirements and various defined standards.

In order to avoid bad reputation in software industry, a software development company must ensure that the software products produced are of high quality. Detection and removal of defects as early as possible is key to achieve high quality. Software Quality Assurance (SQA) process ensures that quality of software is maintained by following quality assurance processes and procedures. Standards like Capability Maturity Model Integration (CMMI) facilitates organizations in maturing their processes and thus improving product quality. Unrealistic deadlines, congested budget, less focus on quality standards, lack of SQA specialists, compromise on quality due to profit considerations, developer attitude, internal politics and various team formations are major factors affecting software quality [7].

3 Agile Software Development

Agile word is derived from word agility. Agility enables software engineers in dealing with rapidly occurring changes in software requirements, teams, technology and other changes having impact on software products. Major driving forces behind agile software development are better customer satisfaction, quick release of software increments, minimal software work products, simple development procedures, dedicated collaborating teams and high customer collaborations. Agile teams include people from software developer organization and customer side. This achieves the objective of providing same view of the software to both developer side and customer side through effective collaboration mechanisms.

The foundations of agile development are iterative and incremental development of software. Agile methods enable developers to develop software in cost effective and time effective manner. Agile methods welcome changes in requirements as these methods do not fix the requirements. Flexibility of design, comfortable changes, quick increments and customer satisfaction are some of features of agile development methodologies [8].

Feature Driven Development (FDD), Test Driven Development (TDD), Crystal, Kanban, Scrum, Extreme Programming (XP), Dynamic System Development Method (DSDM) and Agile Unified Process (AUP) are some commonly used agile methods. But for the sake of simplicity and to be precise only Feature Driven Development (FDD) and Scrum methods will be discussed because of scope of this paper.

3.1 Feature Driven Development (FDD)

Feature Driven Development (FDD) is one of agile methods developed by Peter Codd for supporting Object Oriented Software Engineering (OOSE). It introduces a term “Feature” which is simply a function described by customer.

Develop an Overall Model (phase I) of this model involves study of system scope, construction of detailed domain models, review of models and merging various models belonging to various domain areas for constructing an overall system model. Build a Feature List (phase II) is concerned with creation of feature list on the basis of domain areas. Features in the list are sub-divided into sub features, categorized and prioritized. Identification of feature classes and ownerships are done in Plan by Feature (phase III).

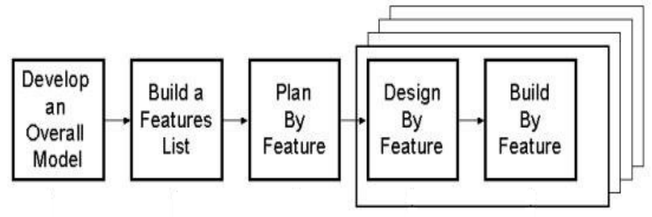

Fig. 2. Feature Driven Development (FDD) Model

Designing of the features are done by chief programmers and class owners in terms of sequence diagrams in Design by Feature (phase IV). Build by Feature (phase V) is concerned with code writing, code inspections and unit testing. After execution of the activities the software increment is added to main build.

3.2 Scrum

Scrum was developed by Jeff Sutherland and his colleagues. However, the word scrum was originated from rugby. Scrum is a lightweight process model for developing small to medium sized software applications. Product Backlog, Sprint Planning, Sprint Backlog, Sprint, Daily Scrum, Product Owner, Scrum Team, Scrum Master, Sprint Review and Sprint Retrospective are some concepts around which entire scrum process is organized.

Product backlog contains prioritized list of requirements in the form of features, bug fixes, non-functional requirements and change requests. Sprint planning is a meeting held between scrum team members for choosing the work for implementation. This meeting is held before beginning a sprint. Sprint backlog is composed of requirements to be implemented in the sprint as decided in sprint meeting. Sprint is actually a unit for measuring completed work. It is time boxed activity and its duration is normally 1-4 weeks. The major focus of sprint is on the implementation of sprint backlog items. Daily scrum is a daily fifteen minutes meeting held between scrum team members for discussion on completed work, pending work and associated problems. Product owner is responsible for developing and managing the product backlog. Scrum team is a team of at most nine developers for performing technical activities such as analysis, design, coding and testing. Scrum master ensures the smooth running of scrum process. Sprint review involves review of the sprint for completed work and pending work. Sprint retrospective is concerned with process improvement through lessons learned.

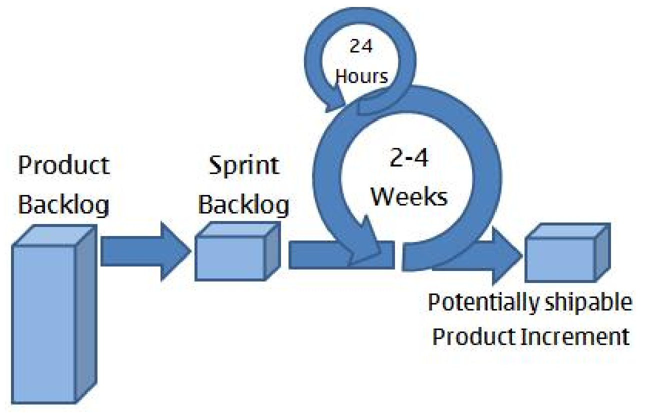

Fig. 3. Scrum Model

Increased productivity is major benefit of scrum. The whole development process is organized around sprints which are formed by sub-dividing software into collection of increments. Scrum starts with collection of requirements and progresses through prioritization, sprint planning, scrum meetings, sprint review and increment delivery. Scrum provides opportunity to cope with changing requirements of customer [9].

4 Problems With Agile

Agile methods although flexible in coping with changes and develop complete software in terms of customer functional needs lacks certain quality characteristics. It is more severe in case of web applications. Correctness is determined in agile by in-sprint testing. To build a complete software, correctness must be ensured by considerable amount of testing. As multiple things are involved in software, so equivalent types of testing strategies must be employed.

Due to global accessibility, web applications are exposed to variety of users trying to breach the security of system to get unauthorized access. Variety of security vulnerabilities and security threats pose a serious problem to security of software and its associated data. Agile methods are silent about security quality attribute. Because of whole focus towards implementation of functional features, security is overlooked.

Software development methodologies are organized around quality management of software. Scrum is limited in terms of safety critical software. Moreover, quality assurance activities, testing and inspections are also not concentrated in scrum [10].

Major cause of software vulnerabilities are lack of secure coding practices comprising of certain programming rules. Following these rules enable developers in managing security. Static analysis is best method for identification of security related vulnerabilities [11].

Usability is also ignored in agile methods especially scrum. Usability is not just about interface design and colouring. Instead it is closely related to Human Computer Interaction (HCI). Human Computer Interaction (HCI) is composed of certain human factors. If such factors are employed properly then software usability could be enhanced. Usability also helps in discovering certain hidden requirements from end users. Agile methods lacked usability by ignoring usability requirements and their implementations. Research studies have shown that complete software increments were even rejected by end users due to lack of usability. Usability improvements resulted in cost and time loss.

Software are impacted badly by lack of usability. In practices, requirement engineering team lacks usability engineers for usability requirement elicitation and thus create gap between requirement engineers and usability specialists. Moreover, developers are held responsible for User Interface (UI) design and thus lacking usability specialists. Usability issues were resulted in scrum because of absence of usability features from product backlog [12].

5 Related Work

Software development involved trade-offs between software quality attributes. There exist positive or negative relationships between quality attributes. Correctness, Testability, Usability and Reliability have positive relationships among them [13].

The traditional Feature Driven Development (FDD) model could be extended to include security features. Overall model development comprised security expert to develop security architecture by identifying security risks. Security is incorporated in phase of build security by feature which is followed by security tests after complete implementation of functional features [14].

Secure Scrum (S-Scrum) model was proposed for managing security in scrum. S-Scrum involved addition of phases for security spikes named as security analysis and security design. Sprint planning is initiated after security considerations. Top priority backlog item with security modelling is decided and implemented during sprint followed by penetration testing approach [15].

Assurance Case Model for Iterative Development of Security Features was proposed for secure software development. It involved three main phases named pre-game phase, game phase and post-game phase. The pre-game phase considered security requirements before functional requirements and involves selection of security features, security requirement gathering, designing of secure feature and preparation of security assurance case for validation. Identification of user stories, functional requirement gathering, design of functional features, implementation of functional features, preparation of test scenarios and testing of stories is involved in game phase. The post-game phase involves validation of complete feature with functional and security features which is followed by demonstration if validated [16].

Usability could be improved by using Usability Engineering Procedural Model and Usability Engineering Methods. Usage of sketches from requirement analysis could be used for immediate usability testing. In order to have focus on functionality and usability, extreme usability can be employed. Extreme Usability (EU) could be accomplished by customers, developers and developer having usability knowledge [17].

A new model was proposed that incorporated usability in scrum. U-Scrum involved two owners for project plan, user experience vision and backlog management. Usability owner develops usability artefacts with the help of vision and personas. Developed artefacts are shared with customers and developers for implementation and validation of prototypes [18].

There are various challenges in agile implementation. Requirement consistency, ambiguity and traceability was first challenge. Agile demanded developers to specialize various skills and hence cause fears in developers. Lack of knowledge regarding agile implementation and business knowledge was another major challenge [19].

Penetration testing conducted by white hat hackers identified vulnerabilities effectively. Scrum used both automated and manual penetration tests to consider security test earlier in the development [20].

Software defects, if unattended, resulted in software failure. Some of important factors for defects were lack of proper logic, lack of testing activities and developer attitude. Lack of code planning by developers was also a cause of inducing failures. Quick and regular changes also inserted certain number of vulnerabilities and defects [21].

Object Oriented Analysis and Design (OOAD) activities could be conducted for management of security. Security requirements could be modelled using Unified Modelling Language Secure (UMLSec) for conceptual model generation. The generated models could be translated into coding implementation using Object Oriented Programming (OOP) [22].

Object Oriented Software Engineering (OOSE) involves Object Oriented Analysis (OOA), Object Oriented Design (OOD) and Object Oriented Programming (OOP) techniques. Such techniques are more focused toward software with high maintainability, correctness, security, flexibility and reusability [23].

For verifying and validating a software testing process is executed. Testing results in identification and removal of software bugs. Various quality attributes such as correctness, reliability, usability, integrity, compatability and security are tested with the help of testing [24].

Agile is focused on requested functionality features as described by customer. Security is one of the quality attribute challenged by agile. Security requirements could not be elicited by discussion with customer [25].

6 Proposed Model (US-Scrum)

In this paper, a new model named US-Scrum is proposed for the enhancement of correctness, security and usability of software products. The US-Scrum model is an extension of existing scrum model developed by combining phases of Feature Driven Development (FDD) and Scrum model. The basic ideology behind this model is to provide an agile model that could address the issue of correctness, security and usability.

Build a Feature List, Sprint Backlog, Plan by Feature, Sprints, User Acceptance and Scrum Meetings are phases of the proposed model.

Build a Feature List phases is concerned with building and managing product backlog. Product backlog in the proposed model is divided into three sections named as Functional Feature List, Security Feature List and Usability Feature List. All these features are traceable using traceability matrices. These classified needs are managed by classified and dedicated managers called masters.

Sprint Backlog involves the selection of related functional, security and usability features selected that can be easily implemented in the coming sprint.

Plan by Feature is alternative phase to sprint planning. It involves dedicated managers and teams to first select the functional features to be implemented during the sprint. Planning of classified needs is done by dedicated masters and teams. Some basic overall models and architectures are developed in this phase as a part of planning.

Parallel, dedicated sprints called "Functional Sprint" with dedicated teams are employed in Sprints. A functional sprint and associated team is responsible for requirement elaboration, analysis, detailed design, coding and testing of the functional features. The duration of functional sprint is same as that of sprint in traditional scrum. Security considerations and activities are conducted by specialized team and sprint executing called "Security Sprint" in parallel to the functional sprint. However, security sprint will be executed two times in a sprint firstly halfway through sprint and secondly at the end of the sprint. Usability is be managed by dedicated sprint known as "Usability Sprint" and team as a parallel activity. Usability sprint is carried out four times during the entire sprint duration. End users are aslo involved and collaborated during usability sprint for usability enhancements and validations.

The proposed model involves two scrum meetings per day (conducted after 24 hours) instead of one per day. One meeting is conducted between top management including product owner and dedicated managers. The second level of meeting is led by corresponding manager and its participants will be corresponding team members. The second level meeting ensures that only task related people are involved in the meeting session.

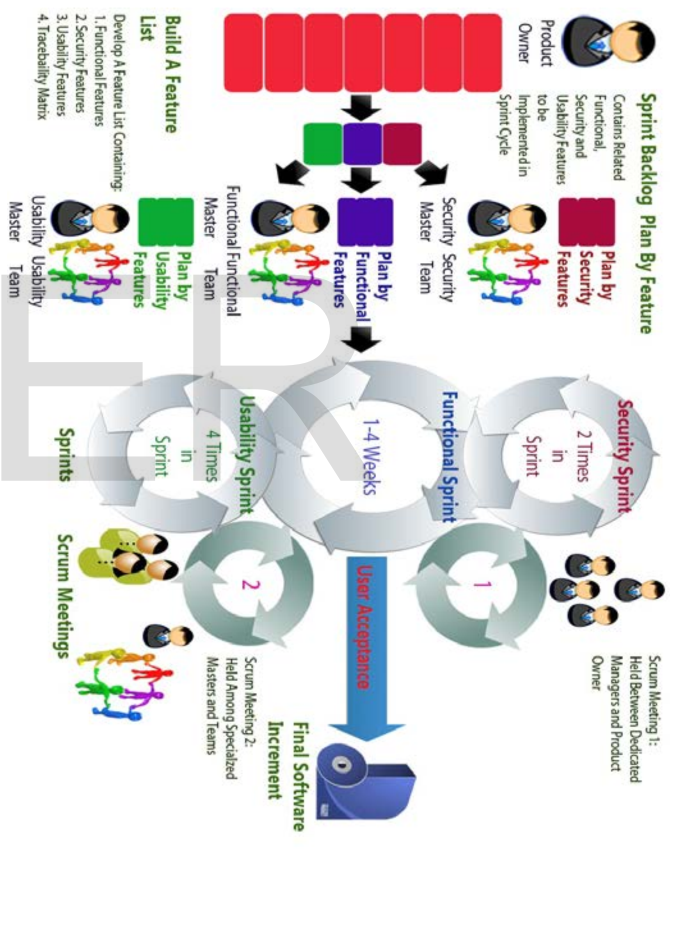

Fig. 4. Proposed Model (US-Scrum)

After the completion of sprint duration, final acceptance tests will be conducted by the customers for validation. After these tests, the developed product with enhanced correctness, security and usability will be released to the main build as a software increment.

Product Owner, Functional Master, Security Master, Usability Master, Functional Team, Security Team and Usability Team are the roles proposed in the model. Each role in the team has its own specialized skills, responsibilities and tasks to be performed. The key to the success of this model is strong collaboration among teams which is facilitated by two level meetings.

Product owner here will act as project manager and will represent voice of customer. Product owner will be in close communication with functional, security and usability masters. Product owner will also be responsible for managing "Feature List".
Functional Master will be responsible for managing the functional features and their traceability. Managing the correct implementation of functional features will also be the responsibility of functional master.

Security Master will be responsible for managing the security features and their traceability. Managing the correct implementation of security features will also be the responsibility of security master.

Security aspects can be included in agile by developing user stories related to security. However, prioritization and trade-offs become difficult if security experts are not included in stakeholder meetings [26].

Security can be ensured with the help of vulnerability testing. Statis analysis approach is easy one but tests have difficult in execution, difficult automation and unavailability of security expert. Also deploymeny environment is beyond the control of developers [27].

Usability Master will be responsible for managing the usability features and their traceability. Managing the correct implementation of usability features will also be the responsibility of usability master.

Functionality team will be headed by the functional master andwill be responsible for performing activities related to functional features such as functional requirement elicitation, analysis, design, implementation (little bit of front-end, back-end and business logic) and functionality testing. It will be a team of professionals having expertise in business process requirements and implementations.

Security team will be composed of people expert in security related matters. This team will be headed by security master and will be responsible for conducting security related activities such as security requirement elicitation, security modeling, secure coding practices and security testing.

Usability team, headed by usability master, will be responsible for executing usability related activities such as usability requirements gathering, lo-fi prototyping, hi-fi prototyping, end user collaborations, usability testing and enhancements.

7 Potential Benefits Of The Proposed Model

The proposed may have many benefits for customers and developers. However, following are some important potential benefits of adopting the proposed model:

7.1 Increased User Focus

The proposed model encourages detailed requirement gathering, user involvement and user acceptance tests for implementation of various features. In this way it is more focused towards users of the software. Usability is given chief importance in this model.

7.2 Better Customer Satisfaction

Because of user acceptance tests and validations, user requirements are understood and implemented precisely. Moreover, hidden requirements are also be elaborated. This increases customer satisfaction. Moreover, user involvement in usability sprints provides better feedback to UI designers and thus increase customer`s level of satisfaction.

7.3 Validation of Requirements

The model provides two step validation for end users i.e. in-sprint validation and after-sprint validation. Users will be able to validate requirement more precisely.

7.4 Strong Verification Process

The model involves considerable amount of test coverage by considering only sprint related testing. In this way, each sprint involves comprehensive test coverage for each feature type. Also it is easier to perform complete feature related tests on the product to be developed. This makes the proposed model to support strong verification process.

7.5 Proper Requirement Implementation Ownership

The use of dedicated requirements, dedicated sprints, dedicated managers and dedicated teams ensures the implementation ownerships in hierarchy. Each team can be designated as owner of its work and can be held responsible for any flaws or errors in it.

7.6 Better Project Management

Large scale projects normally involve lots of functionality features. With large functional features, large number of usability requirements are bound to come. Dedicated teams and managers enable the management of medium to large scale projects much easier and flexible.

Moreover, a single project is divided into multiple small projects and teams with specialist leaders. In this way it becomes easy to manage projects easily, effectively and efficiently.

7.7 Less Pressure on Developers

Dedicated teams and tasks require developers to be master their own filed of interest or experience. Developers are not be required to perform multiple tasks in a project. In this way, considerable amount of pressure is reduced from developers. Also there is no need for developers to learn multiple skills. Each developer can be designated to perform specific work.

7.8 Comprehensive Artefact Production

The use of proposed model allows the scrum teams to develop more artefacts related to software in the form of user stories, use scenarios, abuser stories, abuser scenarios, overall architecture, requirements use case models and other related documents.

8 Conclusion And Future Work

It is concluded that the proposed model takes into account detailed and granular requirements. It also provides collective feature development ownership. The steps in it are organized in such a way to develop quality driven product that satisfies quality criteria of correctness, usability and security. The application of dedicated teams and specialists wipe off the fears of developers for being exposed to skill deficiencies. Increase in customer satisfaction is one of major features of the proposed model. Besides, it also employs strong verification process, user validations and produces set of artefacts.

As a future work, the model can be applied to different case studies and compared with other agile processes addressing the issues of security and usability. Developer related quality attributes assurance model can also be constructed through agile enhancement.

ACKNOWLEDGMENT

It is a matter of great honour and pleasure for me to express my ineffable gratitude and profound indebtedness to my supervisor, committee members and teachers for their inspiring guidance, generous assistance, valuable comments and constructive criticism.

I also thank my affectionate parents for their prayers, guidance, encouragement and love for me. I am thankful to my friends for providing me guideline in education, research and routine life matters.

REFERENCES

[1] S. D. Seilheimer, "Information Management During Systems Development: A Model for Improvement in Productivity," *International Journal of Information Management*, vol. 20, no. 4, pp. 287-295, 2000.

[2] D. Gollmann, "Securing Web Applications," *Journal of Information Security*, vol. 13, no. 1, pp. 1-9, 2008.

[3] A. Chandraskar, S. Rajesh and P. Rajesh, "A Research Study on Software Quality Attributes," *International Journal of Scientific and Research Publications*, vol. 4, no. 1, pp. 1-4, 2014.

[4] K. Y. Cai, Z. Dong and K. Liu, "Software Testing Process as a Linear Dynamic System," *Journal of Information Sciences*, vol. 178, no. 6, pp. 1558-1597, 2008.

[5] S. Kwon, S. Jang, J. Lee and S. Kim, "Common Defects in the Information Security Management System of Korean Companies," *Journal of Systems and Software*, vol. 80, no. 10, pp. 1631-1638, 2007.

[6] N. Juristo, A. M. Moreno and M. I. S. Segura, 2007, "Analyzing the Impact of Usability on Software Design," *Journal of Systems and Software*, vol. 80, no. 9, pp. 1506-1516, 2007.

[7] A. Javed, M. Maqsood, K. A. Qazi and K. A. Shah, "How to Improve Software Quality Assurance in Developing Countries," *Advanced Computing: An International Journal*, vol. 3, no. 2, pp. 17-28, 2012.

[8] H. K. Flora and S. V. Chande, "A Systematic Study on Agile Software Development Methodologies and Practices," *International Journal of Computer Science and Information Technology*, vol. 5, pp. 3626-3637, 2014.

[9] S. Sharma, D. Sarkar and D. Gupta, "Agile Processes and Methodologies: A Conceptual Study," *International Journal on Computer Science and Engineering*, vol. 4, no. 5, pp. 892-898, 2012.

[10] R. Akif and H. Majeed, "Issues and Challenges in Scrum Implementation," *International Journal of Scientific and Engineering Research*, vol. 3, no. 8, pp. 1-4, 2012.

[11] J. Zhu, J. Xie, H. R. Lipford and B. Chu, "Supporting Secure Programming in Web Applications through Interactive Static Analysis," *Journal of Advanced Research*, vol. 5, no. 4, pp. 449-462, 2014.

[12] J. Heiskari, M. Kauppinen, M. Runonen and T. Mannisto, "Bridging the Gap Between Usability and Requirement Engineering," *17th IEEE Int. Requirement Engineering Conf.*, pp. 303-308, 2009.

[13] K. Henningsson and C. Wohlin, "Understanding the Relations between Software Quality Attributes – A Survey Approach," *Proc. 12th Int. Conf. for Software Quality*, pp. 1-12, 2002.

[14] A. Firdaus, I. Ghani and S. R. Jeong, "Secure Feature Driven Development (SFDD) Model for Secure Software Development," *2nd Int. Conf. on Innovation, Management and Technology Research*, pp. 546-553, 2013.

[15] D. Mougouei, N. F. M. Sani and M. M. Almasi, "S-Scrum: A Secure Methodology for Agile Development of Web Services," *World of Computer Science and Information Technology Journal*, vol. 3, no. 1, pp. 15-19, 2013.

[16] L. B. Othmane, P. Angin and B. Bhargave, "Using Assurance Case to Develop Iteratively Security Features Using Scrum," *9th Int. Conf. on Avaialability, Reliability and Security*, pp. 490-497, 2014.

[17] A. Holzinger, M. Errath, G. Searle, B. Thurnher and W. Slany, "From Extreme Programming and Usability Engineering to Extreme Usability in Software Engineering Education," *Proc. 29th Annual Int. Computer Software and Applications Conf.*, pp. 169-172, 2005.

[18] M. Singh, "An Agile Methodology for Promoting Usability," Agile' 08 Conf., pp. 555-560, 2008.

[19] H. Hajjdiab and A. S. Taleb, "Adopting Agile Software Development: Issues and Challenges," *International Journal of Managing Value and Supply Chains*, vol. 2, no. 3, pp. 1-10, 2011.

[20] M. Tomanek and T. Kalima, "Penetration Testing in Agile Software Development Projects," *International Journal on Cryptography and Information Security*, vol. 5, no. 1, pp. 1-7, 2015.

[21] E. E. Ogheneovo, "Software Dysfunction: Why Do Software Fail?," *Journal of Computer and Communications*, vol. 2, pp. 25-35, 2014.

[22] K. S. Joo and J. W. Woo, "Development of Object Oriented Analysis and Design Methodology for Secure Web Applications," *International Journal of Security and Its Applications*, vol. 8, no. 1, pp. 71-80, 2014.

[23] D. K. Saini, "Security Concerns of Object Oriented Software Architectures," *International Journal of Computer Applications*, vol. 40, no. 11, pp. 41-49, 2012.

[24] S. M. K. Quadri and S. U. Farooq, "Software Testing – Goals, Principles, and Limitations," *International Journal of Computer Applications*, vol. 6, no. 9, pp. 7-10, 2010.

[25] C. Woody, "Agile Security – Review of Current Research and Pilot Usage," *Software Engineering Institute, Carnige Mellon University*, pp. 1-8, 2013.

[26] K. Beznosov and P. Kruchtehn, "Towards Agile Security Assurance," *Proc. 2004 Workhop on New Security Paradigm*, pp. 47-54, 2005.

[27] V. Kongsli, "Towards Agile Security in Web Applications," *Companion to the 21st ACM SIGPLAN Conf. Object-Oriented Programming Systems, Languages and Applications*, pp. 805-808, 2006.