International Journal of Natural and Engineering Sciences 8 (1): 42-48, 2014
ISSN: 1307-1149, E-ISSN: 2146-0086, www.nobel.gen.tr

Software Quality Assurance of Medium Scale Projects by using DXPRUM Methodology

Muhammad FAHAD[1]* Salman QADRI[2] Syed Shah MUHAMMAD[1] Mujtaba HUSNAIN[2]
[1]Department of Computer Science, Virtual University of Pakistan, Lahore, Pakistan
[2]Department of Computer Science and Information Technology, The Islamia University of Bahawalpur, Pakistan

*Corresponding author:
Email: ms100400039@vu.edu.pk

Received: March 20, 2014
Accepted: April 28, 2014

Abstract

Agile methods are being used as a new way of developing software. They are basically used to fasten software development cycle by keeping an eye on its quality. Using Agile methods in any project along with quality assurance perspective is a crucial task. Many software developing organizations are using Agile methods now a days in their projects to reduce cost involved in a project. Agile methods are best suited for small projects as they require less resources and able to generate output in less time. Testing of these small projects is also an easy task. However, many organizations are also using Agile methods for medium and large scale projects. These projects also have large as well as geographically distributed teams. Further testing of these projects cannot be done on regular basis due to the size of the projects. Assuring quality of these projects using Agile method is a challenging task. Many Agile methods are being used by software developing organizations to build high quality software. Each Agile method has its own strengths and weaknesses. This research work is based on purposing a novel hybrid Agile methodology DXPRUM. The DXPRUM is a combination of three of the Agile models named as Dynamic Systems Development Method, Extreme Programming and Scrum (D comes from DSDM, XP from Extreme Programing, RUM from Scrum). The objective is to combines the strengths of all the three Agile methods by removing their weaknesses. The main strength of DXPRUM will be the in time delivery of the project to customer with reduced cost and high quality.

Keywords: Dynamic Systems Development Method, Extreme Programing, Agile Software Development, Scrum, SDLC

INTRODUCTION

Software engineering can be depicted as a phenomenon which embodies number of steps for the purpose of producing high quality software according to the requirements of the clients [1]. Creation of high quality software is an intricate task. It is due to software engineering that we are able to design software and even improve its quality throughout software development life cycle (SDLC). A framework which provides different ways to structure, plan, and control the process of developing software is called a software development methodology [2]. A methodology consists of policies, principals, methods and processes used to implement software. Many software development methodologies are available in software industry. Every methodology has its own software development life cycle which differ it from other ones. Some of the famous names include Waterfall, prototyping, incremental and spiral development. These are traditional ways of development of software and are recognized as heavy weight methodologies [3]. These methodologies contain gigantic documentations, less customer involvement and low flexibility for creeping requirements. A survey by Cockburn shows that in case of a business change, 65% of the delivered functionalities are rarely used or are never used [4]. During last two decades Agile software development methodologies become extremely famous in software industry. These are basically light weight development methodologies. Some of the famous Agile methodologies includes scrum, extreme programing (XP), dynamic systems development method (DSDM), feature driven development (FDD), Kanban, Lean Software Development etc. Agile methodologies are best known for their fast development life cycles, less documentations and low costs for projects [5]. In Agile the developers and testers work in a strong collaboration in order to reduce chances of errors [6]. Agile Methodologies are not rigid and contains more customer involvement as a quality assurance practice [7]. Each software development methodology has its own pros and cons. According to a survey, the success rate of Agile software development is about 71.5% [8]. Agile methodologies are being adopted by more and more companies in recent past due to rigid behavior of traditional frameworks and because of the rising complexity of the software projects [9].

In order to cope with pros and cons of different methodologies, the software development organizations are also using hybrid models for software development [10]. These hybrid models are combinations of two or more methodologies and are used with the quest to merge their benefits at one place so that organizations can have best of them available for obtaining maximum output [11]. According to Livermore, Scrum is often combined with Extreme Programing (XP) practices [12]. Both Scrum and XP has different flavors. According to Pressman, In order to build a successful software increment, DSDM may be combined with XP. This will actually generate a combo methodology which defines a powerful process model (the DSDM life cycle) by using nuts and bolts practices (XP) [13]. Some other researchers combine Agile with traditional methodologies. Lina and Dan combine Scrum with CMMI in order to get a better framework for small and medium sized organizations [14]. Keeping all these points in view, the researcher decided to propose a new Agile software development model by combining three already existing models (DSDM, XP and Scrum) and naming it DXPRUM (D comes from Dynamic Systems

Development Method, XP comes from Extreme Programing and RUM comes from Scrum). The new model contains all the strengths of three already existing models (DSDM, XP and Scrum) by removing their weaknesses. The model also provides better results in terms of quality assurance when applied to medium scale projects. The proposed model will also work well with continuously changing requirements. A brief introduction of these three Agile methodologies is as follows and also shown in figure 1.

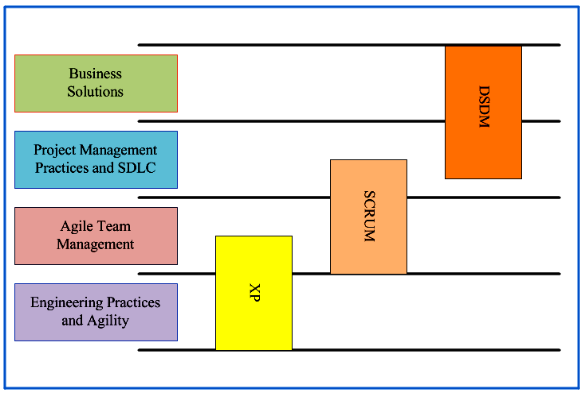

Business Solutions
Project Management Practices and SDLC
Agile Team Management
Engineering Practices and Agility
DSDM
SCRUM
XP

Figure 1. What Agile Methods Covers

Dynamic Systems Development Method (DSDM)

Dynamic Systems Development Method is a framework for building high quality business solutions. It is a well-organized framework and deals with the projects where there are tight time constraints. It uses incremental prototyping and follows an iterative approach. The best aspect of Dynamic Systems Development Method is that it provides an environment where all interested parties involved in a project can cooperate and collaborate for successful completion of the project [15]. The Dynamic Systems Development Method is a modified version of Pareto Principal. In Pareto Principal 80% functionality of a project is delivered in 20% of the time in order to deliver a complete 100% of project [16]. The remaining 20% of the functionality is deliberately left for later iterations. This strategy deals best with continuously changing requirements because we know that not 100% of the requirements are best known to the developers at the start of the projects and they kept on changing throughout the project life cycle. The process of Dynamic Systems Development Method consists of seven phases [17]. These include pre-project, feasibility study, business study, functional model iteration, design and build iteration, implementation and post project phases. The main benefits of DSDM includes quick and in time delivery of the project with also an eye on reducing the cost of the project. It also involves self-organizing and collaborated teams.

Extreme Programing (XP)

Extreme Programing is a scientific and disciplined Agile software development approach consisting of rules, ideas and practices used to build highest quality software with shorter development cycles. XP consists of small iterations with small releases and rapid feedbacks. The duration of each one of iteration is about three weeks long. And a project consists of 2-5 of such small iterations. It is a combination of 12 basic rules and techniques and is best implemented in small and medium sized companies. A team of 3-10 developers are used in XP. According to Beck, "the size of the team should be around 3-20 members” [18]. XP is a natural process and hence cannot be enforced. It is more of a developer focused technique with very little focus on management. Out of 12 rules, only 3 are related to management and other 9 are developers focused. According to Beck, “These practices support each other and the weakness of one is covered by the strength of others” [19].

Scrum

Scrum is a project management framework in Agile software development which deals with how to manage a particular software project. It is an iterative and incremental approach of developing software. The Scrum does not deal with how to engineer a product. Scrum deals best with continuously changing variables like requirements, time, cost, resources and technology. The process of Scrum is flexible enough to accommodate theses changing variables [20]. The process of Scrum uses rapid prototyping technique. . Each team in Scrum consists of 5-10 members. The duration of each sprint is between 2-4 weeks depending upon what team members thinks about complexity of that particular iteration. Each sprint has all phases of software development life cycle in it like analysis, system design, coding and testing. Multiple teams in Scrum worked at the same time to complete a project on time. The Scrum is one of the most popular Agile Methodology these days and is used by many organizations in software industry [20].

MATERIALS AND METHODS

DXPRUM – A Proposed Hybrid Model

DXPRUM is the proposed hybrid model which is a combination of three of the widely used Agile models named Dynamic Systems Development Method (DSDM), Extreme Programing (XP) and Scrum. DXPRUM combines the strengths of all the three models by removing their weaknesses. It combines the project management practices of Scrum with business focused approach of DSDM and by covering whole SDLC under the umbrella of engineering practices of XP. This makes DXPRUM a more powerful model for medium scale projects. The best features of DXPRUM model includes management of software projects in a fixed time constraint and to work well with changing requirements. The defect rate of DXPRUM is also low when it compares to three Agile models. This results in high quality software product.

The DXPRUM have combined features of DSDM, XP and Scrum. It has pre-project and post-project phases of DSDM along with some SDLC features also. The system backlog, sprint backlog and DXPRUM increment features are those also present in Scrum. During whole SDLC the different phases are covered under the engineering practices of XP. These practices include coding standards, pair programming, refactoring, collective ownership of code and test driven development. Combination all these make DXPRUM an interesting model for software development organizations.

The DXPRUM model is shown in figure 2. It starts with pre-project phase. In this phase a feasibility study of the software to be built is conducted. This feasibility study ensures whether the software product may be completed within certain time constraints and budget. The pre-project phase is conducted before the project is officially started.

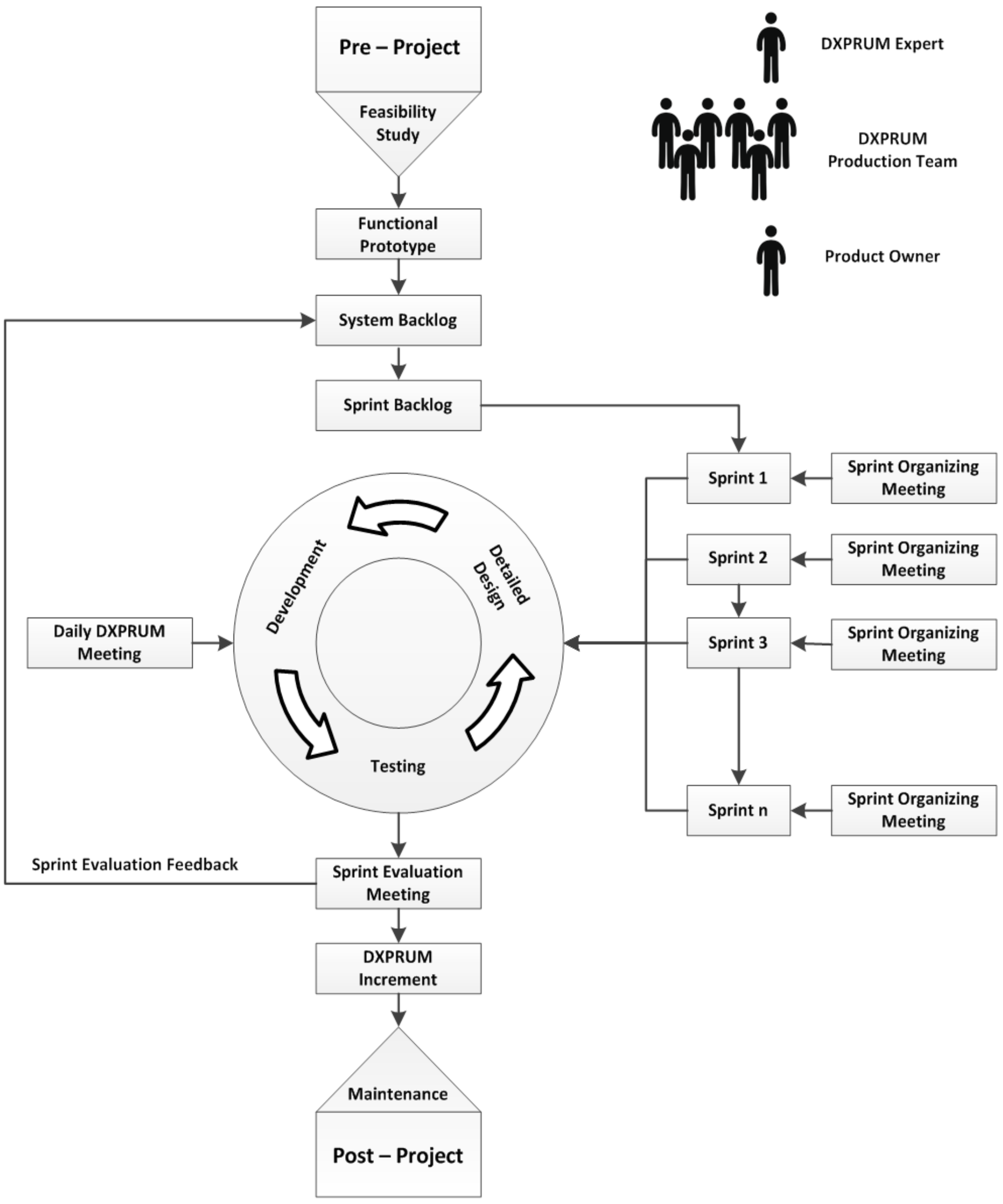


Figure 2. The DXPRUM Model

Then the first important phase of software development life cycle begins. This is the functional prototype phase. This is the phase where initial design of the product is constructed. The input to this phase is the SRS document. Then the system backlog is created. Here the requirements classification is done according to their priorities. The whole of the process is completed with the consultation of the stakeholders. These involve DXPRUM expert, product owner and DXPRUM production team. Then the product is divided in to sprints. The high priority requirements should be completed in first. Then the main phases of SDLC are started. These include detailed design of the product, development and testing phases. These phases are conducted under the umbrella of XP practices. These include coding standards, pair programing, refactoring, collective ownership of code and test driven development. After a sprint is completed, a sprint evaluation meeting is conducted. The meeting ensures whether all the requirements of that sprint are implemented or not. If there are any remaining or new requirements explored in that meeting these become part of the sprint evaluation feedback (SEF) document which is the input to system backlog for implementation in next sprint. If whole requirements are implemented and the product owner is satisfied from the sprint output then the sprint release is considered as DXPRUM increment. The final stage of the DXPRUM is the post-project phase. It is the one where maintenance of the project is done in future. If any bug is found by the customer in the product in future this will be resolved in maintenance phase.

The complete process of DXPRUM intends to cover all the aspects of SDLC that are necessary for the production of software with premium quality, low cost and low defects. Each sprint cycle in DXPRUM is of 7-10 days long. A medium size project contains 4-8 of such sprints. Thus the duration of the project varies from 1.5 to 3 months long. The sprint organizing meeting in DXPRUM is of 2-4 hours of duration. The daily DXPRUM meeting is of 30-45 minutes of duration. Once again the sprint evaluation meeting is of 2-4 hours of duration. The different roles of DXPRUM model are DXPRUM expert, product owner and DXPRUM production team. The DXPRUM roles, events, artifacts and phases are explained in detail in next sections.

Roles of DXPRUM

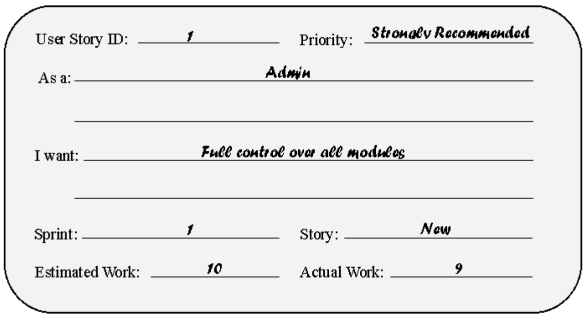
The Different roles in DXPRUM include DXPRUM Expert, Product Owner and DXPRUM Production Team. The DXPRUM expert is the person who is responsible for overall progress of the development process in DXPRUM model. The DXPRUM expert acts like a manager and is responsible for the successful completion of the error free software in time and within proposed budget. The product owner is also a stakeholder in DXPRUM model and has the responsibility of deciding what work to be done? The product owner is the person who writes and prioritizes software requirements in the form of user stories. These user stories then become part of system backlog. A group of cross-functional and technically skilled professional people who are responsible for the delivery of error free software according to a set of given functional requirements are called production team. In DXPRUM model the DXPRUM production team consists of 4-7 full-time members.

DXRUM Events

The different events of DXPRUM include Sprint Organizing Meeting, daily DXPRUM Meeting and Sprint Evaluation Meeting. The sprint organizing meeting is conducted at the start of each sprint in DXPRUM to plan about the sprint which is going to be started. The different participants of sprint organizing meeting is DXPRUM expert, product owner and DXPRUM production team. The daily DXPRUM meetings are conducted on daily basis to measure the progress of the production team. The participants of these meetings include DXPRUM expert and DXPRUM production team. The sprint evaluation meeting is conducted at the end of each sprint to measure the progress of all activities of previous sprint. The participants of sprint evaluation meeting include DXPRUM expert, DXPRUM production team, product owner, customer and any other business partner for that project.

DXPRUM Artifacts

The different DXPRUM artifacts include System Backlog, Sprint Backlog, DXPRUM increment, and Sprint Evaluation Feedback (SEF). The system backlog in DXPRUM model contains all the prioritized functional and non-functional requirements of the system to be implemented. The DXPRUM expert gathers these requirements from product owner in the form of user stories on user story cards. The system backlog and user story card is shown in table 1 and figure 3 respectively. The system backlog is further divided into sprint backlogs for implementations. Each sprint in DXPRUM has its sprint backlog. The sprint backlog is shown in table 2. The output of each sprint is a working set of the product and is known as DXPRUM increment. The sprint evaluation feedback (SEF) is a document created during sprint evaluation meeting and contains remaining or incomplete requirements. These requirements in the SEF document acts as input to system backlog to be implemented in next sprint cycle.



User Story ID: 1 Priority: Strongly Recommended

As a: Admin

I want: Full control over all modules

Sprint: 1 Story: New

Estimated Work: 10 Actual Work: 9

Figure 3. User Story Card

Table 1. System Backlog

User Story ID	User Story	Priority	Sprint	Status	Story	Estimated Work	Actual Work
1	As admin I want full control over all modules	Strongly Recommended	1	Closed	New	10	9
2	I want every customer to register himself first	Strongly Recommended	1	Closed	New	13	11
3	I want product search module in my site	Recommended	2	Open	Feed-back	8	0
4	I want my customer to recommend product to their friends	Not Recommended	4	Open	New	12	0
5	I want customers to access my Facebook page	Less Recommended	3	Open	New	10	0
6	I want advanced search feature	Medium Recommended	3	Open	New	6	0
7	I want to customize my view	Less Recommended	4	Open	New	4	0
8	I want control to sign-out any customer at any time	Strongly Recommended	1	Active	New	10	0
9	Any registered customer should be authenticated first by admin	Recommended	2	Open	New	7	0
10	As admin I can view all sales and purchase reports	Strongly Recommended	1	Open	New	14	0

Table 2. Sprint Backlog

User Story ID	User Story	Tasks	Day 1	Day 2	Day 3	Day 4	Day 5	…..
1	As admin I want full control over all modules	Design the ……	3	2	1	1		
		Code the …..	2	2	1	2		
		Meet with Development Team	1	1	2	1		
		Design the user interface …..	0	3	0	2		
		Test the …..	1	1	1	1		
2	I want every customer to register himself first	Design the ……	4	3	2	2		
		Code the …..	3	2	2	3		
		Meet with Development Team	1	2	0	0		
		Design the user interface …..	0	2	1	1		
		Test the …..	1	1	1	1		

Phases of DXPRUM

The phases of DXPRUM are the backbone of this hybrid model. These must be conducted smoothly in order to get a high quality defect free product. These phases include Feasibility Study, Functional Prototype, Detailed Design, Development, Testing and Maintenance. During feasibility study a feasibility report of the project is generated which includes answer to different important questions about the project. The resources, completion time, risks and cost involved and usefulness of the project is discussed in detail in feasibility report. The functional prototype in DXPRUM model is the initial design of the project. This initial design is made on paper and is then reviewed. It also provides best ways of solving the problems that occur during development. The detailed design phase in DXPRUM model occurs once the sprints are decided in sprint organizing meeting. The detailed design phase gives a detailed design of the modules that are implemented in that particular sprint. These modules are designed and developed in iterations. During development the designed modules are coded for implementation. The DXPRUM production team takes part in development process. The DXPRUM expert is also there to help production team in case they need any help. Whole of the development process is conducted under the umbrella of XP practices. These practices include pair programming, coding standards, refactoring, collective ownership of code and 40 hours per week work. Testing is an important activity in DXPRUM model. During this stage the software increment is tested against all functional and non-functional requirements. Different types of testing include unit testing, integration testing and system testing. The maintenance is the last phase in DXPRUM model. It is a post-project phase. The maintenance phase is started once the software is handed over to the customers. During use if customers found any bug in the software, he/she report it to production team for maintenance. The production team removes that bug by issuing some update or releasing some patch. The maintenance is done using same sprint cycle as done earlier to build the software.

RESULTS AND DISCUSSIONS

A controlled case study was conducted to validate proposed model of DXPRUM. The results of the case study were recorded from very first day. We have also maintained a table showing total meetings that were held during implementation of DXPRUM model on PC Arena Online Shopping cart project. Four DXPRUM increments were produced during this case study. First two increments were of 2 weeks of duration while last 2 increments were of 1 week of duration each. We have recorded data from all four sprints. The numerical data collected from the project is shown in table 3. We can see from the table that data from all four sprints are shown individually in the form of columns. The last column shows the total of all four sprints. The gathered data is explained here in detail.

The first row shows that first two sprints are completed in 2 weeks each while last two sprints took 1 week each for completion. This makes a total of 6 sprints to complete this project. The next row shows the total number of modules implemented during each sprint cycle. 15 modules are implemented in first sprint and 6 are implemented in last sprint. A total of 39 modules are implemented during this project. We can see from 3rd row that a total of 46 user stories are implemented in this project in all four sprints. These user stories are actual user requirements. The 4th row shows a total of the estimated work hours required to complete this project. We can see that a total of 1152 hours are estimated for completion of this project. The estimated work hours are calculated by using this formula.

EWE = Duration of a sprint in weeks * Total working days in a week * Total working hours in a day * Size of production team

For example for sprint 1 the EWE can be calculated

EWE = 2 * 6 * 8 * 4 = 384

Table 3. Results of DXPRUM Case Study

Sr No.	Parameter	Sprint 1	Sprint 2	Sprint 3	Sprint 4	Total
1	Sprint Duration (in Weeks)	2	2	1	1	6
2	No of Sprint Modules	15	11	7	6	39
3	No of User Stories	18	14	8	6	46
4	Estimated Work Effort (EWE) (in Hours)	384	384	192	192	1152
5	Actual Work Effort (AWE) (in Hours)	322	304	156	148	930
6	No of Implemented Classes	65	44	34	22	165
7	Total Lines of Code (LOC)	21034	14026	10236	9422	54718
8	Kilo Lines of Code (KLOC)	21.03	14.03	10.24	9.42	54.72
9	Tested Lines of Code	9125	5035	4023	9853	28036
10	Pre-Release Defects	10	8	6	4	28
11	Post-Release Defects	6	3	2	2	13
12	Performance / Quality (Post-Release Defects / KLOC)	0.29	0.21	0.20	0.21	0.23 (Avg)
13	No of Sprint Evaluation Feedback Suggestions	6	5	3	3	17
14	No of Unit Tests	256	212	195	162	825
15	Customer Satisfaction (in Average) (Shodan Survey)	95%	85%	95%	93%	92% (Avg)
16	Team Productivity (LOC per hour)	65.32	46.14	65.62	63.66	60.19 (Avg)
17	Time to implement a user story (AWE / No of user stories) (in hours)	17.89	21.71	19.50	24.67	20.94 (Avg)

The row 5 shows the actual work effort in hours that is recorded to complete a sprint. We can see that first sprint took 322 hours for completion. While total of four sprints are 930 hours. The row 6 shows the total no of classes that are implemented in each sprint. A total of 165 classes are implemented in this project. The row 7 shows the total lines of code for the project. The production team coded a total of 54718 lines in this project. The row 8 shows total kilo lines of code that are 54.72. This can be calculated using the following formula.

KLOC = LOC / 1000

The row 9 shows the tested lines of code. These are lines of code included in test cases. A total of 28036 lines are included in test cases out of 54718 lines. The row 10 shows the pre-release defects of each sprint. The pre-release defects are pointed out by production team during development stage and they can be fixed using iterative approach. Last column shows the total of pre-release defects which are 28. The row 11 shows a total of post-release defects which are 13. These are the defects which are pointed out by product owner or customer during sprint evaluation meeting. These defects are then recoded in sprint evaluation feedback (SEF) document which becomes part of product backlog and are implemented during next sprint. The next row 12 shows the performance or quality of the project. The less the number is the performance or quality of the project is high. We can see from the table that DXPRUM has a value of 0.23.This shows that the product is of high quality.

Performance / Quality = Post-release Defects / KLOC

We can see from row 13 that sprint 1 contain 6, sprint 2 contain 5, sprint 3 contain 3 and sprint 4 also contain 3sprint evaluation feedback suggestions. This makes a total of 17 sprint evaluation feedback (SEF) suggestions occurred during this project. These include post-release defects as well as any suggestions or new requirements that are indicated by customer. The row 14 shws the total number of unit test that are conducted during testing phase. A total of 825 unit tests are conducted during this project. The row 15 shows the percentage of customer satisfaction level during each sprint. Also last column shows the average of all sprints. During this project an average of 92% customer satisfaction is achieved. This is obtained by conducting surveys during project implementation. The row 16 shows the team productivity. This is explained as the number of lines coded by development team in an hour. The average of the project is shown in last column which is 60.19 lines per hour.

TP = LOC / Actual work effort (in hours)

The last row shows time to implement a user story in each of the four sprints. The last column shows the average of all four sprints. The time to implement a user story in any sprint can be calculated by using the formula below.

Time to implement a user story = Actual work effort / No of user stories.

CONCLUSION

The motivation for the proposed DXPRUM model is to combine the best features of DSDM, XP and Scrum and removing their weakness. DSDM provides a complete software development life cycle but it only works best when used for business projects. It is not suited for all types of projects. This weakness is overcome by Scrum and XP. XP focuses on team work. It is a combination of engineering practices but lacks proper SDLC as well as proper documentation. There is also lack of planning in XP. It also has poor performance for medium and large scale projects. Both Scrum and DSDM overcome these weaknesses of XP. Scrum is a project management framework. It does not provide proper planning for a project. It also lacks proper SDLC as well as engineering practices. DSDM and XP both merged in order to overcome the weaknesses of Scrum.

In this paper we have developed a new hybrid Agile model named as DXPRUM which provides a complete software development life cycle for software developing organizations. We have also validated it with the help of a case study. We have discussed many factors in this case study that affect the quality of a medium size projects.

The case study results clearly showed that DXPRUM is more compact, elegant and powerful model than other Agile methodologies. We have showed by results that the main strengths of DXPRUM model are the in time delivery of the project to customers with reduced cost. The model also worked well with continuously changing requirements. This model also overcomes the weaknesses such as to reduce resource utilization without affecting the output and to remove the overlapped resources.

Future Work

In this paper, we have presented DXPRUM model for use in medium scale projects. In future this model can further be modified for use in large scale projects by implementing some other features of DSDM, XP and Scrum. For example Scrum of Scrums feature of Scrum model may be used here for large scale projects. Similarly some XP practices along with some phases of DSDM may also be used. The requirement classification phase of DXPRUM model can be automated in future by using artificial intelligence techniques. An intelligent neural network based system can be build that can take important decisions like allotment of priority tags during requirements classification. The author considers it as a future work.

REFERENCES

[1] Pahl G, Beitz W, Feldhusen J, Grote K. 2007. Engineering Design: A Systematic Approach. 3rd Ed. The Springer London. (London, UK). ISBN: 978-1-84628-318-5.

[2] Boehm BW. 1988. A Spiral Model of Software Development and Enhancement. Int. Journal of IEEE, Computer. (CA, USA). 21(5): 61-72. ISSN: 0018-9162.

[3] Greenfield J, Short K. 2003. Software Factories: Assembling Applications with Patterns, Models, Frameworks and Tools. In Proc. of Int. Conference of Companion of the 18th Annual ACL SIGPLAN on Object-oriented Programming, Systems, Languages, and Applications. (NY, USA). pp: 16-27. ISBN: 1-58113-751-6.

[4] Cockburn A. 2004. Crystal Clear: A Human-Powered Methodology for Small Teams. 1st Ed. Addison-Wesley Professional. (Boston, USA). ISBN: 0201699478.

[5] Abrahamsson P, Warsta J, Siponen MT, Ronkainen J. 2003. New Directions on Agile Methods: A Comparative Analysis. In Proc. of 25th Int. Conference on Software Engineering. (Portland, Oregon, USA). pp: 244-254. ISSN: 0270-5257.

[6] Cohn M, Ford D. 2003. Introducing an Agile Process to an Organization. Int. Journal of IEEE Computer. (CA, USA). 36(6): 74-78. ISSN: 0018-9162.

[7] Awad MA. 2005. A Comparison between Agile and Traditional Software Development Methodologies. M. S. thesis, School of Computer Science and Software Engineering, The University of Western Australia. (Australia).

[8] Ambler SW. 2002. Agile Modeling: Effective Practices for Extreme Programing and the Unified Process. 1st Ed. John Wiley & Sons Inc. (New York, USA). ISBN: 0471202827.

[9] Naik K, Tripathy P. 2008. Software Testing and Quality Assurance, Theory and Practice. John Wiley & Sons, Inc. (Hoboken, New Jersey, USA). pp. 523-527. ISBN: 978-0-471-78911-6.

[10] Boehm BW. 2002. Get Ready for Agile Methods, with Care. Int. Journal of IEEE, Computer. (CA, USA). 35(1): 64-69. ISSN: 0018-9162.

[11] Mushtaq Z, Qureshi MRJ. 2012. Novel Hybrid Model: Integrating Scrum and XP. Int. Journal of Information Technology and Computer Science (IJITCS'12). 4(6): 39-44. ISSN: 2074-9015.

[12] Livermore J, 2008. Factors that Significantly Impact the Implementation of an Agile Software Development Methodology. Int. Journal of Software. (Oulu, Finland). 3(4): 31-36.

[13] Pressman RS. 2010. Software Engineering: A Practitioner's Approach. 7th Ed. The McGraw-Hill Companies, Inc. (New York, USA). pp. 65-94. ISBN: 978-0-07-337597-7.

[14] Lina Z, Dan S. 2012. Research on Combining Scrum with CMMI in Small and Medium Organizations. In Proc. of IEEE Int. Conf. on Computer Science and Electronics Engineering (ICCSEE'12). (Hangzhou, China). 1: 554-557.

[15] Stapleton J. 1999. DSDM: Dynamic Systems Development Method. In Proc. of Int. Conf. of Technology of Object-Oriented Languages and Systems. (Nancy, France). pp: 406. ISBN: 978-0-7695-0275-5.

[16] Stapleton J. 2003. DSDM: Business Focused Development. 2nd Ed. Addison Wesley Professional. The DSDM Consortium. (London, UK). pp. 145-163. ISBN: 0-321-11224-5.

[17] Voigt BJJ. 2004. Dynamic Systems development Method. M. S. Thesis, Department of Information Technology, University of Zurich. (Zurich, Switzerland).

[18] Beck K. 1999. Embracing Change with Extreme Programing. Int. Journal of IEEE Computer. (California, USA). 32 (10): 70-77. ISSN: 0018-9162.

[19] Beck K. 2000. Extreme Programing Explained: Embrace Change. 1st Ed. Addison- Wesley Professional. (Boston, USA). ISBN: 9780201616415.

[20] Schwaber K, Beedle M. 2002. Agile Software Development with Scrum: Advanced Development Methods. 1st Ed. Prentice Hall, Upper Saddle River. (New Jersey, USA). pp: 145-171. ISBN: 0130676349.