REQUIREMENTS PRIORITIZATION IN SPRINT VIA TIME QUADRANTS

Khurram Shahzad
MS. Computer Engineering
SZABIST, Karachi, Pakistan
khurram.shahzad.bs@hotmail.com

Abstract- Agile development methods are now being recognized as popular and efficient overtures to the development of software related systems that have features such as a short delivery period and unclear requirements. Scrum is one of the most popular methods that are currently being used. Some backlogs, especially high priority backlogs that are functional requirements of customers, are developed repeatedly at each Sprint period. There are many ways of setting priorities like brainstorming, multi-voting, 100 voting and nominal grouping. This means a list of criteria should be proposed to give a solution for setting priorities. In scrum and sprint the criteria cannot work due to its nature of rapid development.

My research will introduce new criteria for setting sprint priorities by mapping them into time quadrants. To identify the difference between urgent and important is something many people do not understand. Usually people incline to focus on urgent activities. By setting the criteria, the quadrants will represent how to identify the right one sprint backlog, measure them and map them into specific quadrants. All tasks will be evaluated using the criteria important/unimportant and urgent/not urgent basis and put in according to the quadrants. By using time quadrants we can work towards more effective business results. Each of these quadrants represents the development team and product owner with a number of decisions regarding the next steps in line.

I. INTRODUCTION & OVERVIEW

Agile methods were created in the early 1990's, however, in last five years we have seen agile methodology adoption and popularity increase due to its nature of quick and rapid development. Scrum is one of the most popular methods that are currently being used. Some backlogs, especially high priority backlogs, that are functional requirements of customers, are developed repeatedly at each SPRINT period. The Sprint is of a fixed duration time box which means some kind of results need to be observable at the end of a specific date whether the sprint iteration has been accomplished or not, and is never extended, at the beginning of each Sprint, a cross-functional team selects items i.e. user stories from a prioritized list of requirements, and attempt to complete them by the end of the Sprint [1].

A. Scrum Overview and Process

Takeuchi introduced Scrum for the first time with DeGrace, Schwaber, and others in the early 1990s [11]. Based on iterative and incremental development, Scrum is an unsmooth outline of a process. Scrum is preferable particularly in large sized circulated teams, it is imperative that all team members need to be acquainted with what is going on and where the team stands (Figure 1). The possibility for detailed planning is negligible and merely covers a distinct and uniquely differenciating iteration.

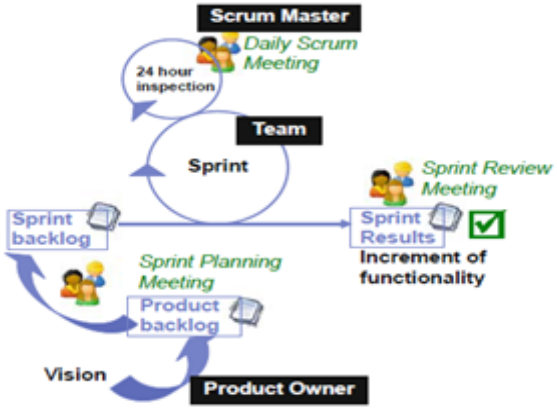

Fig 1: The Generic Process of Scrum

B. User Stories and Product Backlog

In scrum development, user stories are a very important artifact of the product backlog. **I**t is a reminder to have a conversation with your customer. User stories are very light and high-level requirement artifacts. User stories encourage deferring details until you have the best understanding that you are going to have about what you really need. User stories are often written on sticky notes stored in a show box or written on index cards. The format of a formal user story is;

As a **<role>** I want **<something>** so that **<benefit>**

We can take the example of an employee user story which could be rewritten as "As a Scholar I want to buy a parking pass so that I can drive to the university" [7]. This assists the developer to think about what a certain attribute is build for and the reason behind it and as a result, this is an approach that we classically favour to take. Building and organizing the product backlog is always a work in progress.

The product backlog items (called user stories) are detailed appropriately, as shown in Figure 2 where the higher priority items are explained in more detail than the lower priority items. It is to be noted here that these requirements are decomposed, discovered and processed throughout the entire project.

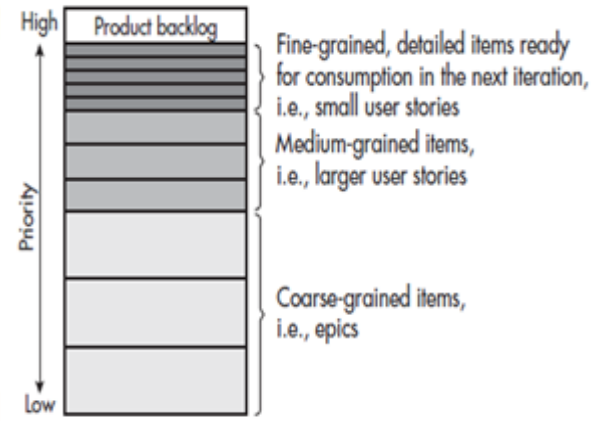

Fig 2: The level of Details shown in Product Backlog Prioritization

C. *Sprint Planning*

The endeavour of the sprint planning meeting is to provide the scrum team with adequate information to be able to work in uninterrupted peace for a few weeks (usually 1-3 weeks), also to give the product owner enough assurance to let them do so like;

[1] A sprint goal clearly defined
[2] A list of all team members (and levels of their commitment)
[3] A sprint backlog, a list of highly prioritized user stories incorporated in the sprint
[4] A clearly defined date of sprint demo
[5] A defined daily scrum meeting's place and time

D. *Time Quadrants: Do the Right Things at the Right Time*

I believe that the best explanation of time management can be summarized in a single phrase as the act of "organizing and executing around priorities". This section provides the conceptual basis for the entire scrum and sprint process discipline via time quadrants. It classifies projects based on the extent of each task's urgency and importance. It describes a four quadrant model as the basis of a discussion of urgent work, risk or uncertainty, team consistency, communications, customer and stockholder involvement, change specification and business values. The time management matrix shows you in which quadrant you really spend your time. Apparently, time usage falls into any one of four quadrants and time Management is about controlling the use of the most valuable and undervalued resources. The figure below shows an example grid.

Fig 3: Basic Time Management Grid

This table highlights the scope of the Time Management Quadrants; the vertical column is the importance column. The top small Quadrants contain more important tasks while the activities in the bottom are less important.

II. TWO ABSTRACT PARTS OF OUR STUDY AND THE PROPOSED SOLUTION

In my research, I have two abstract parts. First, I will set the criteria for requirement prioritization and second, I will describe how these prioritized requirements can be mapped in sprint via Time Quadrants.

A. *Requirement Prioritization Setting Criteria*

Product Backlogs usually contain user stories and in our case, customer requirements are in the form of user stories so we have to introduce the criteria where these user stories can be sorted out and prioritized according to the cost, business value(s) and risk(s).

1. Prioritization Scale: The aim of prioritization is to give specific values to each user story that allow the formation of use stories in a relative order. In my work, the requirements which need to be prioritized are user stories. The prioritization can be accomplished with prioritization scales and variety to identify how you will rate your user stories against the criteria. You can choose 1 to 3 scale of low, medium or high respectively.

2. Giving Weights to Prioritization based on Value, Cost, and Risk: Weighting is a criterion that involves determining whether some requirements are more significant than others. A criterion where something is very important gets a value of 3 and low in importance acquire a weight of 1. This can help you to differentiate the core functionalities that must be present for that feature, including even the low-priority requirements. Their priority may change over time so it will help if you plan in advance for future developments. The following three important factors which are involved in prioritizing the product backlog are Business Value = 4, Cost of Implementation =3, and Risk or Uncertainty = 5.

	High = 3, Medium = 2, Low = 1			
	Business Value	**Cost to Implement**	**Risk or Uncertainty**	**Total Value**
Weight	**4**	**3**	**5**	
User Story - A				
User Story - B				
User Story - C				
User Story - D				
User Story - E				
User Story - F				
User Story - G				
User Story - H				
User Story - I				
User Story - J				
User Story - K				
User Story - L				
User Story - M				

Fig 4: Prioritizing User Stories

Now proceeding with the scoring process, first I assigned ratings for each user story against each criterion. We tried to put the rating in the far left part of each cell (tuple of table) as shown in below example.

	High = 3, Medium = 2, Low = 1			
	Business Value	Cost to Implement	Risk or Uncertainty	Total Value
Weight	4	3	5	
User Story - A	2	2	3	
User Story - B	3	1	3	
User Story - C	3	3	2	
User Story - D	1	2	1	
User Story - E	2	1	2	
User Story - F	3	3	3	
User Story - G	2	2	1	
User Story - H	2	1	2	
User Story - I	3	2	1	
User Story - J	3	2	3	
User Story - K	1	1	1	
User Story - L	2	3	3	
User Story - M	3	2	2	

Fig 5: Evaluating each User Story

Finally, I proceeded to refine the table by multiplying the ratings through the weighting value and adding up the scores.

	High = 3, Medium = 2, Low = 1			
	Business Value	Cost to Implement	Risk or Uncertainty	Total Value
Weight	4	3	5	
User Story - A	2 x 4 = 8	2 x 3 = 6	3 x 5 = 15	29
User Story - B	3 x 4 = 12	1 x 3 = 3	3 x 5 = 15	30
User Story - C	3 x 4 = 12	3 x 3 = 9	2 x 5 = 10	31
User Story - D	1 x 4 = 4	2 x 3 = 6	1 x 5 = 5	15
User Story - E	2 x 4 = 8	1 x 3 = 3	2 x 5 = 10	21
User Story - F	3 x 4 = 12	3 x 3 = 9	3 x 5 = 15	36
User Story - G	2 x 4 = 8	2 x 3 = 6	1 x 5 = 5	19
User Story - H	2 x 4 = 8	1 x 3 = 3	2 x 5 = 10	21
User Story - I	3 x 4 = 12	2 x 3 = 6	1 x 5 = 5	23
User Story - J	3 x 4 = 12	2 x 3 = 6	3 x 5 = 15	33
User Story - K	1 x 4 = 4	1 x 3 = 3	1 x 5 = 5	12
User Story - L	2 x 4 = 8	3 x 3 = 9	3 x 5 = 15	32
User Story - M	3 x 4 = 12	2 x 3 = 6	2 x 5 = 10	28

Fig 6: Complete Evaluation of Tasks in User Stories

In this scenario, with the weighting you can see that User Story-F is on the highest priority with a total value of 36 and then User Story-J is on 2nd highest priority and so on. So, by using priority scales and weighting criteria we can prioritize our user stories in the product backlog.

B. *Planning for Sprint via Time Quadrants*

Sprint Backlog is the initial point for each Sprint. Above I have set the criteria for requirements (in the form of user stories) and prioritized high value stories which have been decided in the sprint planning meeting. In the meeting, it was decided as to which stories will be implemented in the next Sprint. The activities or the whole process in Sprint can be handled via Time Management Quadrants. As was described in the previous section, the time management matrix shows you on which quadrant you should really spend your time. Apparently, your time usage falls into one of four quadrants and time Management is all about controlling the use of the most valuable and undervalued resources. By using Time Management Quadrants, we classified and categorized all the activities of the Sprint on the basis of urgency and importance.

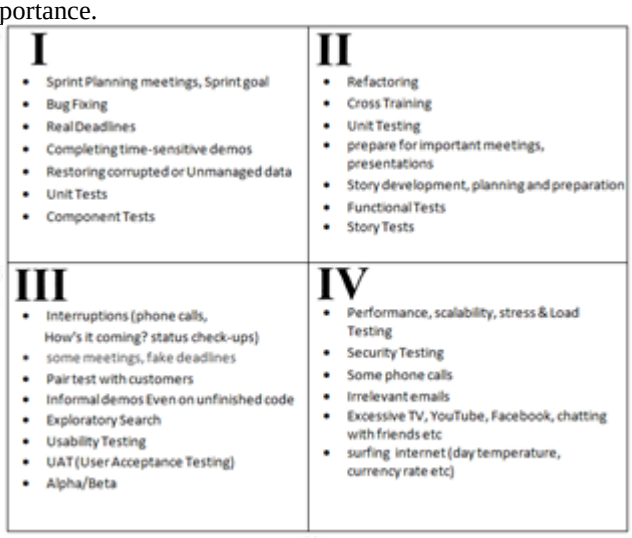

Fig 7: The Division of Tasks into the 4 Quadrants

1. *First Quadrant Activities and Goals:* The first Quadrant works on important and urgent based activities. Sprint Planning meetings are very important because without sprint planning and defining goal we can't start the sprint development process, those bugs which were raised in previous iterations must be resolved first to make product stable. In sprint planning meetings, real deadlines of sprint completion must be decided in the first quadrant and time-sensitive demos must be set here and during sprint development iteration, you should take an overview of the whole code once and check if there is any corrupted data or unmanaged data existing in the sprint and then manage and restore corrupted data in this quadrant if it appears, at the end unit testing needs to be done on what you have developed and also the component testing if required.

2. *Second Quadrant Activities and Goals:* As second Quadrant works on important and not urgent based activities, the Sprint development team should spend more time in this quadrant. Almost 70% of the sprint duration restricts the activities of this quadrant till their completion otherwise the activities which were left will move to the first quadrant and become urgent. Preparation for important meetings and presentation need to be done by living in this quadrant also development and planning of user stories can be done in this quadrant. Refactoring needs to be done to make the code faster and remove unnecessary code. Also, cross training should be given to the sprint team if required, for testing purposes, functional testing should be performed to test the whole user story.

3) Third Quadrant Activities and Goals: The third Quadrant works on the not so important yet urgent based activities. The activities which are not important but urgent like Interruptions of phone calls, status check-ups, some urgent meetings with clients or scrum master or product owner, also the fake deadlines for achieving the goals before the actual deadline can be settled is in this quadrant. Pair testing with customers, informal demos even on unfinished code can be accomplished in this quadrant. Sometimes, programmers need some guidance through the internet to solve issues or to develop any functionality, for this purpose an exploratory search can be performed here because the way of finding a solution is an urgent work otherwise you will be stuck and won't go ahead in the sprint development process. Regarding testing, usability testing, user acceptance testing and alpha or beta testing can be performed in this quadrant.

4) Fourth Quadrant Activities and Goals: The fourth quadrant works on the not so important and the not urgent based activities. While in sprint we have a prioritized list of activities but still the need of the fourth quadrant is important because of the activities that we usually perform while we are in the first, second or third quadrant like some phone calls and talking to friends, wasting time for checking irrelevant emails, Excessive TB, watching clips on YouTube, giving time to Facebook, chatting through messengers, surfing on the internet like checking the temperature of the day or currency value etc.. These all are the types of activities which take you away from your schedule and real deadlines so be careful when you are doing any such type of activity and check whether you are not away from the exact schedule. If the team is working so consciously and progress is so well and if the team thinks that they can achieve their target before time then the team should do performance, scalability, stress and load testing which comes in the fourth quadrant, security testing can also performed here in this quadrant if security is not a major issue then security testing can also be performed here and if security is a major issue then it can be in the second quadrant and will exist in any user story.

III. CONCLUSION AND FUTURE RECOMMENDATION

In this paper, I first introduced the background knowledge for my research. In this research, I divided my study into two parts, the first part was regarding the requirements prioritization in the form of user stories, and for this we proposed a weighting table. Weighting is a criterion that involves determining whether some requirements are more important than others. Weighting tables are organized through prioritization scale and by giving weight based on business value, cost and risk/uncertainty. After some mathematical calculation on this weighting table you will have all highly prioritized stories from top to down respectively.

The second part of my study was to map these prioritized requirements in sprint via time management quadrants and for this I proposed a four quadrants solution of all activities that can be done during the sprint process development and classified activities according to their absolute quadrant. All activities in the sprint development process are evaluated using the criteria important/unimportant and urgent/not urgent basis and put in accordingly into the quadrants. Each of these quadrants represents the development team and product owner with a number of decisions regarding how to move forward. By using time management quadrants, it can be said that we can easily manage our time and can improve the effectiveness of business results and achieve our goals on time.

REFERENCES

VI. Zornitza Bakalova, Maya Daneva, Andrea Herrmann, Roel Wieringa "Agile requirements prioritization: what happens in practice and what is described in literature" REFSQ'11 Proceedings of the 17th international working conference on Requirements engineering, Springer-Verlag Berlin, Heidelberg, 2011

VII. Zornitza Bakalova, Maya Daneva, Klaas Sikkel, Andrea Herrmann, Roel Wieringa "Do We Know Enough about Requirements Prioritization in Agile Projects: Insights from a Case Study" RE '10 Proceedings of the 2010 18th IEEE International Requirements Engineering Conference IEEE Computer Society Washington, DC, USA , 2010

VIII. Carol K. Gonzales, Gondy Leroy, "Eliciting user requirements using Appreciative inquiry" Empirical Software Engineering archive Volume 16 Issue 6, Kluwer Academic Publishers Hingham, MA, USA, December 2011

IX. Andrea Herrmann, Barbara Paech, "Practical challenges of requirements prioritization based on risk estimation", Empirical Software Engineering archive Volume 14 Issue 6, Kluwer Academic Publishers Hingham, MA, USA, December 2009

X. EmamHossain, Muhammad Ali BabarandandHye-young Paik."Using Scrum in Global Software Development: A Systematic Literature Review", International Conference on Global Software Engineering IEEE2009 Fourth

XI. T. Dyba and T. Dingsoyr, "Empirical Studies of Agile Software Development: A Systematic Review", Information and Software Technology, Vol. 50, Issue 9-10, 2008, pp-833-859

XII. Cuong D. Nguyen, Erin Gallagher, Aaron Read, Gert-Jan De Vreede, "Generating user stories in groups", CRIWG'09 Proceedings of the 15th international conference on Groupware: design, implementation, and use Springer-Verlag Berlin, Heidelberg, 2009

XIII. Eduardo Cristiano Negrão, Eduardo Martins Guerra, "A case study for prioritizing features in environments with multiple stakeholders", SPLASH '11 Proceedings of the ACM international conference companion on Object oriented programming systems languages and applications companion ACM New York, NY, USA , 2011

XIV. Lisa Crispin,"Agile Test Planning with the Agile Testing Quadrants" Internet: http://lisacrispin.com/downloads/AdpTestPlanning.pdf , ADP Testing Workshop 2009 [January, 10, 2012].