Enhancing Quality in Scrum Software Projects

Amrita Raj Mukker[1], Anil Kumar Mishra[2], Latika Singh[3]

[1]School of Computer Science and Engineering, ITM University, Gurgaon, India

[2]School of Computer Science and Engineering, Assistant Professor, ITM University, Gurgaon, India

[3]School of Computer Science and Engineering, Associate Professor, ITM University, Gurgaon, India

Abstract: ***Agile methodology has gained popularity in the last few years. Many large scale projects are developed using agile methodology. Scrum techniques are applied to many large scale projects, this paper discusses one such project. Here a case study has been discussed of a large scale project which uses Scrum Technique for development of its software project. The project is of a telecommunication company which uses Scrum for the development of its product Xsset. This case study discusses various tools to optimize the performance of the product. Here in this paper the different tracking tools and representation used to optimize the performance like run chart, JIRA, SVN and SCTM has also been discussed and using these tools few quality metrics has been calculated. This paper has also proposed a way in which we can optimize the cost and effort in Scrum Projects using function point with COCOMO model.***

Keywords: Agile, COCOMO Model, Effort, Quality, Scrum, Optimization Tools.

1. Introduction

This paper discuss about the agile methodology. Basically two methodologies are frequently used in agile projects: eXtreme Programming and Scrum. In XP methodology it requires onsite customer, contains frequent feedback opportunities. It is most widely known and adopted approach and has Strong technical practices. On the other hand Scrum Complements existing practices, it contains self organizing teams and feedback, Customer participation and steering. Here priorities are based on business value. Scrum has approaches that have a certification process. In this paper we will be talking about Scrum and how it is used in companies for its betterment. We have taken a project of a telecommunication company known as Xsset and worked on various tools used in the development of the product. Here in this paper a case study has been show which shows how the use of Scrum development process enhances the quality. We have listed few quality metrics here and categorized them under specific tool which is enhancing the quality of the product. This paper has also proposed a way in which we can improve the effort and cost estimation method. Here we have proposed that if we merge function point with COCOMO model then we can reduce the effort. So an algorithm has been proposed which merges function point with COCOMO model and we used this algorithm on the product Xsset and using its data we have reduced the effort. Thus we validate the algorithm and show how effort can we improved.

2. Scrum

Scrum is a framework within which people can address complex adaptive problems, while productively and creatively delivering products of the highest possible value. Scrum is:

- Lightweight
- Simple to understand
- Extremely difficult to master

2.1. Scrum Theory

Scrum is founded on empirical process control theory, or empiricism. Empiricism asserts that knowledge comes from experience and making decisions based on what is known. Scrum employs an iterative, incremental approach to optimize predictability and control risk. Three pillars uphold every implementation of empirical process control:

i. Transparency
ii. Inspection
iii. Adaptation

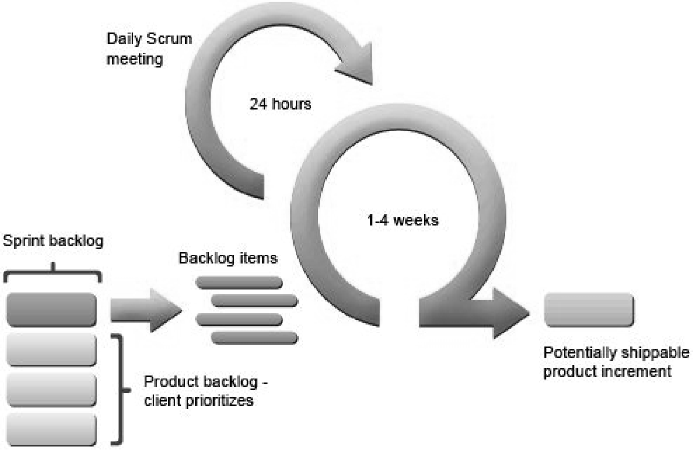
Daily Scrum meeting
24 hours
1-4 weeks
Sprint backlog
Backlog items
Product backlog - client prioritizes
Potentially shippable product increment

Figure 1: Scrum Lifecycle

The Scrum framework consists of Scrum Teams and their associated roles, events, artefacts, and rules. Each component within the framework serves a specific purpose and is essential to Scrum's success and usage.

2.2. Scrum Events

2.2.1. Daily Scrum

The Daily Scrum is a 15-minute time-boxed event for the Development Team to synchronize activities and create a plan for the next 24 hours.

2.2.2. Sprint Review

A Sprint Review is held at the end of the Sprint to inspect the Increment and adapt the Product Backlog if needed.

2.2.3. Sprint Backlog

The Sprint Backlog is the set of Product Backlog items selected for the Sprint plus a plan for delivering the product Increment and realizing the Sprint Goal.

3. Telecommunication Company

It is the largest vendor of network planning and optimization software. Its range of integrated products is designed to assist operators in planning, deploying, optimizing and managing mobile networks. Its open, standards-based application framework, I-VIEW, provides a common basis for a standards based, integrated and scalable solution. Its products enable operators to regain visibility and control of their network, allowing them to become more efficient, more agile and more profitable

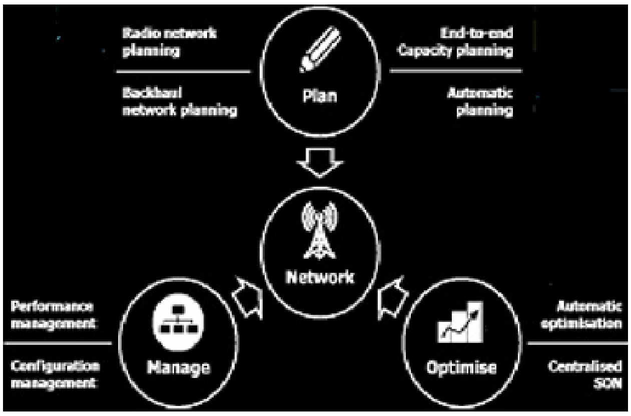

Figure 2: Process followed

3.1. Their Motive

The market leader in the provision and deployment of network engineering tools, its products are in use across 155+ countries by over half the world's mobile operators. Every day, the 20 top global operators depend upon the companies' tools and consultants to improve network coverage and quality for more than 2 billion subscribers worldwide. Providing expertise since 1995, they have built their reputation on creating and releasing additional value from within mobile networks

3.2. Planning Product

The company's portfolio of planning products cover Radio Network Planning, Backhaul Network Planning and End-to-end Capacity Planning across a wide range of technologies including GSM, UMTS, HSPA, LTE, CDMA, EV-DO and WiMax. Key to the success of our products is their ability to integrate with each other as well as the ability to integrate seamlessly into your wider OSS ecosystem allowing you to easily transfer data to and from 3rd party systems to improve overall planning efficiency and accuracy.

3.3. XSSET

Its **product** is the market-leading radio network planning product that includes comprehensive planning/optimization and automation capabilities. XSSET has been engineered to facilitate the most demanding needs of today and tomorrow's multi-vendor, multi-operator, multi-technology mobile networks.

4. Tools and Representation Used to Optimize the Performance

4.1. Target Progress Tool (TP)

Target Process 2.7 has been released, the tool has been mentioned here previously for its 2.0and 2.3 releases. Target Process is an Agile Process Management tool that automates many of the tasks associated with an agile project. It helps simplify planning, tracking and QA. It provides real time reports, historical data and allows upper management to see the status of several projects at a glance. A number of new features have been added, including Visual Iteration Planning and Program Level Release Planning.

4.1.1. Visual Iteration Planning

This clever feature displays a large box that shows you much "room" is left in your iteration (based on your previous iterations velocity). Inside the box are the stories that the team has committed to for the iteration, each with its estimated size. In addition, bugs are marked with a red bar and a small icon. This seems like a very elegant way of planning

4.1.2. Program Level Release Planning

Have a large product with several teams? This new feature makes it easier to track a number of projects within a single program. It makes it possible to see how releases will line up. Agile methods are predicated on the belief that the value created by the team is greater than the sum of the individuals. A focus on the performance and reward of individuals can motivate team members to look out for themselves and not the team. This might manifest itself with some team members refusing to coach or avoiding parts of the project that might not make them look good.

4.2. Burn Down Charts

A **burn down chart** is a graphical representation of work left to do versus time. The outstanding work (or backlog) is often on the vertical axis, with time along the horizontal. That is, it is a run chart of outstanding work. It is useful for predicting when all of the work will be completed. It is often used in agile software development methodologies such as Scrum.

In figure 1, the blue line shows the ideal scenario if your team performs exactly as predicted by your task estimates and the red line shows the actual performance. At day 0 (the first day of the iteration), the remaining effort is at its highest because nothing has been completed. At the end of the iteration (day 20), the sum should be 0 because there are no tasks left to be completed.

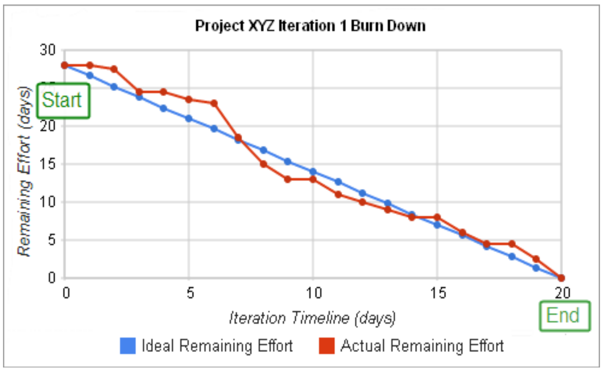

Figure 3: Burn down chart

4.3. JIRA

Jira is a very powerful tool and can be used as defect tracking system as well as planning tool for Agile projects. Like many tools, Jira provide you capabilities and how you use it to increase your productivity is up to you. JIRA lets us prioritize, assign, track, report and audit our 'issues,' whatever they may be —from software bugs and hardware defects, to improvement and change requests. It can Customizable reporting allows us to monitor the progress of our issues with detailed graphs and charts JIRA's powerful workflow capabilities allow us to map an issue's lifecycle to match our processes.

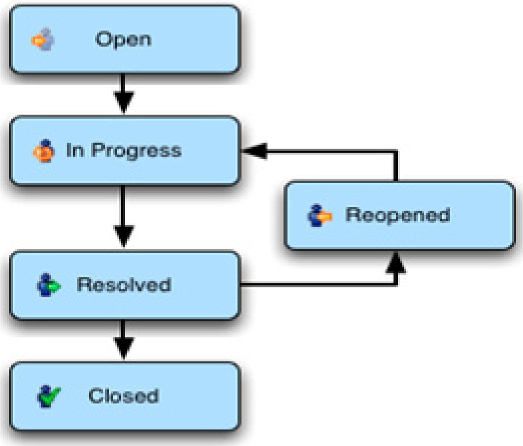

Figure 4: JIRA- Flow Diagram

4.3.1 Steps Followed

- Login
- Manage Dashboard
- Enter a new Project
- Enter a new Component
- Enter a Defect
- Manage Defect
- Resolve and Close
- Search, Reports, Email, etc

4.3.2 Ongoing Usage:

a) QA Team

- Ensure issues are being resolved
- Provide support to teams in system development, features
- Provide monthly status reports on Defect Stats
- Provide ongoing training to all users
- Work with Software team to see if this is a good fit vs. Bugzilla

b) System and Subsystem Leads:

- Ensure review of open issues is taking place
- Use workflow and tracking tools to monitor issue progress
- Use JIRA to track issues, reference JIRA entries in Elog

4.4. Silk Central Test Management (SCTM)

Silk Central is test management software developed by Borland with integrated framework that is built to improve productivity, traceability, and visibility for all types of software testing. Silk Central Test Manager is an open software test management solution that supports both responsive and traditional development projects.

4.4.1. Usability In Business

- Reduce costs and increase quality of applications
- Ensure delivery of projects
- Define optimal testing practices across teams

4.4.2. Benefits

- Collect prioritize and control all aspects of test management.
- Deliver advanced reporting, including project status, requirements and test coverage.
- Resolve capacity problems before they have an operational impact.

4.4.3. Key Features

- Personalized dashboard for different stakeholders.
- Unified framework for test management across Distributed teams.
- Open integration with software testing tools
- JIRA is a Java EE web-based bug tracking and issue tracking application developed by Atlassian Software Systems. This bug tacking tool can theoretically be integrated to work in conjunction with Silk Central Test Manager by using the Silk Central API.

4.5. Tortoise SVN

Tortoise SVN is a free open-source Windows client for the Apache™ Subversion® version control system. That is, Tortoise SVN manages files and directories over time. Files are stored in a central repository. The repository is much like an ordinary file server, except that it remembers every change ever made to your files and directories. This allows you to recover older versions of your files and examine the history of how and when your data changed, and who changed it. This is why many people think of Subversion and version control systems in general as a sort of "time machine". Some version control systems are also software configuration management (SCM) systems. These systems are specifically tailored to manage trees of source code, and have many features that are specific to software development - such as natively understanding programming languages, or supplying tools for building software. Subversion, however, is not one of these systems; it is a general system that can be used to manage any collection of files, including source code.

4.5.1. Features

1. Icon overlays
2. Graphical User Interface
3. Easy access to Subversion commands
4. Directory versioning
5. Atomic commits
6. Versioned metadata
7. Choice of network layers
8. Consistent data handling

TortoiseSVN is stable:

1. Before every release, we create one or more "release candidates" for adventurous people to test first.
2. During development cycles, many people test intermediate builds. These are built every night automatically and made available to all our users. This helps finding bugs very early so they won't even get into an official release.
3. A big user community helps out with testing each build before we release it.

A custom crash report tool is included in every TortoiseSVN release which helps us fix the bugs much faster, even if you can't remember exactly what you did to trigger it.

5. Proposed Work:

In the case study i gathered data for 2 sprints where each sprint was for 15 days and the target process (TP) was used for tracking the data of the product. JIRA was the tool used for tracking the defects and SCTM was used to test and enhance the quality of the product. Total 10 members were working on this project, out of which 3 were developing the code, 3 members were in the quality assurance team, 2 members were analyzing the requirements and 2 members were from the validating team who were validating the code. After doing the complete case study of the telecommunication company and its product i have proposed 2 things:

i. Quality Metrics
ii. Effort Improvement.

5.1. Quality Metrics

After working on the project XSSET and attending daily meetings I collected data on how the development process was carried out and using those data I have calculated values of various existing quality metrics for each tool which can help to check the enhanced quality of the product. Below i have categorized the metrics on the basis of the various tools. Here I have listed few metrics and how these tools improve the quality of the telecommunication company's product XSSET. Below i have listed the data that i gathered. Sprint March.

Table 1: Sprint's details for March

Sr. No	*Data*	*March A*	*March B*
1	Defects	6	7
2	Story	4	20
3	Planned effort	55.5	55.5
4	Initial effort	76	76
5	Function Point	99	144
6	LOC	2670	4810
7	Defects that are reported during the user acceptance testing.	0	0

5.1.1. Target Performance tool (TP)

Table 2: Quality Metrics improved using TP

Sr. No	*Metrics*	*Description*	*Outcome*
1	Availability	MTBF/MTBF+MTTR	High
2	Story Completion	Actual story completed committed story	20+4/20+4= 24/24= 1
3	Effort	Planned effort/Initia Estimate	55.5/76=0.73 Low
4	Retrospective Process Improvement	The purpose of the [Sprint Retrospective is to inspec how the last Sprint went ir regards to people relationships, process anc tools	High JIRA optimize and improve the process efficiency
5	Efficiency	It is the ease with which the tool performance.	High and much improved
6	Backlogs	The total value of contrac commitments yet to be executed. (Total Backlog = Previous Fiscal Years Commitments + Lates Fiscal Years Sales - Lates Fiscal Years Revenue.)	Low

5.1.2. JIRA

Table 3: Quality Metrics improved using JIRA

Sr.No	*Metrics*	*Description*	*Outcome*
1	Defect Density	No of Defects / Size (FP or KLOC)	15/243=0. 062 Low
2	Code Quality	JIRA improves the code Quality	Improved and high
3	Reliability	# Mean time between failures (MTBF) - Total operating time divided by the number of failures. MTBF is the inverse of failure rate.	#low
		# Mean time to repair (MTTR) - Total elapsed time from initial failure to the reinitiating of system status. Mean Time To Restore includes Mean Time To Repair (MTBF + MTTR = 1.)	#Low
		# Reliability ratio = (MTBF / MTTR)	#High
		#MMTR	#Low

4	Defect Removal Efficiency	1. Number of post-release defects (found by clients in field operation), categorized by level of severity 2. Ratio of defects found internally prior to release (via inspections and testing), as a percentage of all defects 3. All defects include defects found internally plus externally (by customers) in the first year after product delivery	Sprint : March A= 8/8=1 100% March B =7/7=1=100% So we can say its 100% efficient
6	Code Coverage	It the degree to which the source code of a program is tested by a particular test suite	Improved

5.1.3. SCTM(Silk Centre Test Manager)

Table 4: Quality Metrics improved using SCTM

Sr.No	*Metrics*	*Description*	*Outcome*
1	Test Effectiveness	t/(t+UAT) **Explanation:** Here "t" is the total number of defects reported during the testing, whereas UAT, means the total number of defects that are reported during the user acceptance testing.	High For Sprint March, t= 7+8=15 UAT=0 Test effectiveness = 15/15+0=1
2	Quality of Testing	No of defects found during Testing/(No of defects found during testing + No of acceptance defects found after delivery) *100	8+7/(0+15)*100= 1*100= 100%
3	Test Coverage	Number of units (KLOC/FP) tested / total size of the system	High
4	Cost Of Quality Activity	•Costs of reviews, inspections and preventive measures •Costs of test planning and preparation •Costs of test execution, defect tracking, version and change control •Costs of diagnostics, debugging and fixing •Costs of tools and tool support •Costs of test case library maintenance •Costs of testing & QA education associated with the product •Costs of monitoring and oversight by the QA organization (if separate from the development and test organizations)	Low
5	Test Planning Productivity	No of Test cases designed / Actual Effort for Design and Documentation	High
6	Test Execution Productivity	No of Test cycles executed / Actual Effort for testing	High

5.2. Effort Improvement

After working on the product XSSET I noted it uses Function point to calculate the cost and effort. So keeping this in mind I have proposed that if we merge function point with COCOMO Model then we can easily calculate the effort and cost of the project and further can improve the effort required. Below I have first explained function point and then the COCOMO model and finally merged the two and proposed an algorithm. Further I have validated the algorithm using data I gathered of the product XSSET and thus I validated my algorithm on this project.

5.2.1. Function Point

Function Point Analysis is an objective and structured technique to measure software size by quantifying its functionality provided to the user, based on the requirements and logical design. This technique breaks the system into smaller components so they can be better understood and analyzed. Function Point count can be applied to Development projects, Enhancement projects, and existing applications as well.

- Final FP Count:

After determining the Unadjusted Function Point count (UFP) out of transactional and data function types, and calculating the Value Adjustment Factor (VAF) by rating the general system characteristics, the final Function Point count can be calculated using the formula:
FP = Unadjusted Function Point count (UFP) * Value Adjustment Factor (VAF)

5.2.2. COCOMO Model

The **Constructive Cost Model** (**COCOMO**) is an algorithmic software cost estimation model developed by Barry W. Boehm. The model uses a basic regression formula with parameters that are derived from historical project data and current as well as future project characteristics.

COCOMO consists of a hierarchy of three increasingly detailed and accurate forms. The first level, Basic COCOMO is good for quick, early, rough order of magnitude estimates of software costs, but its accuracy is limited due to its lack of factors to account for difference in project attributes (Cost Drivers). Intermediate COCOMO takes these Cost Drivers into account and Detailed COCOMO additionally accounts for the influence of individual project phases.

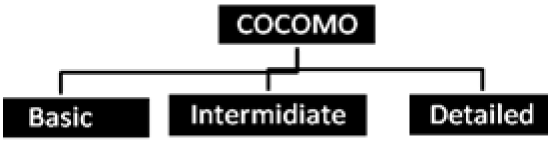

Figure 5: COCOMO types

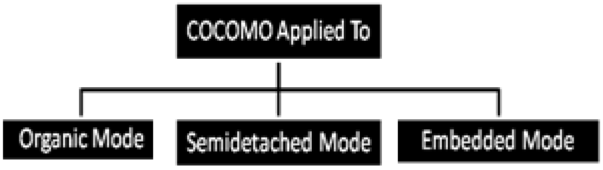

Figure 6: COCOMO applied to

Let us consider Basic COCOMO:
The basic COCOMO equations take the form

- **Effort Applied (E)** = $a_b(KLOC)^{b}{}_{b}$ **[person months]**
- **Development Time (D)** = $c_b(\text{Effort Applied})^{d}{}_{b}$ **[months]**
- **People required (P)** = Effort Applied / Development Time **[count]**

Where, **KLOC** is the estimated number of delivered lines (expressed in thousands) of code for project. The coefficients a_b, b_b, c_b and d_b are given in the following table:

Table 5: Basic COCOMO coefficients

Software project	a_b	b_b	c_b	d_b
Organic	2.4	1.05	2.5	0.38
Semi-detached	3.0	1.12	2.5	0.35
Embedded	3.6	1.20	2.5	0.32

Basic COCOMO is good for quick estimate of software costs. However it does not account for differences in hardware constraints, personnel quality and experience, use of modern tools and techniques, and so on.

5.2.3. Drawbacks of COCMO Model:

- It is KLOC constraints
- It does not ensure quality assurance cost.
- COCOMO model ignores requirements and all documentation.
- It ignores customer skills, cooperation, knowledge and other parameters.
- It oversimplifies the impact of safety/security aspects.
- It ignores hardware issues
- It ignores personnel turnover levels
- It is dependent on the amount of time spent in each phase.

5.2.4. Merging Function point with COCOMO Model:

The main drawback of COCOMO was it is KLOC Constraints. To overcome this problem I have proposed merging FP with COCOMO. Till yet COCOMO has been merged with the following:

1. Fuzzy Logic
2. Use Case
3. Neural Network
4. Expert Judgement

Till yet no one has merged with Function Point with COCOMO Model. By doing this we can overcome the drawback of COCOMO.

5.2.5. Algorithm Proposed

1. First analyse the function point and calculate the functional count for the source code of the product.
2. Then to calculate the KLOC for each FP we apply divide and conquer rule. Here we first divide all the FPs and then calculate the line of code for each and after calculating then merge them together and find the total KLOC.
3. After calculating the KLOC for the total we apply COCOMO model to it so as to calculate the cost and effort
4. We apply the formula and we can reduce the effort applied.

6. Validating the Algorithm

To validate the algorithm I randomly picked data of 2 sprints. The sprint was for 15 days each and both the sprints were for a single month (March). The 2 sprints were named as

6.1. Sprint 1- March B

It had 8 stories in total. For each story we calculated the FPs. Below a table has been listed where the FP for each story has been calculated and Witten, on each FP I have applied Divide and Conquer rule and I have calculated first LOC of each FP and then added all the LOC of each FP to get the total LOC.

Table 6: Size of the stories of sprint March B

Story	*Points*	*FP*	*Total LOC*
1	3	23	790
2	1	5	90
3	1	7	110
4	0.5	3	50
5	1	8	120
6	5	30	1450
7	5	48	1500
8	3	20	700

*1 point= 1 day
Now total LOC=4810
KLOC= 4.810
Now applying COCOMO Model
4.810 mean that it is in Organic Mode. After The calculation:
E= 12.5
D= 6.53
Avg. Staff Size= 2 approx.
Productivity= 0.4

6.2. Sprint 2- March A:

Sprint 2 had 4 stories in total. We did the same as we did above and we have calculated the fps and the total LOC as done in sprint 1 using the proposed algorithm listed in the table below.

Table 7: Size of the stories of sprint March A

Story	*Points*	*FP*	*Total LOC*
1	1	15	90
2	3	25	330
3	3	28	360
4	6	31	1890

Now total LOC= 2670
KLOC= 2.670
Now applying COCOMO Model
2.670 mean that it is in Organic Mode. After The calculation:
E= 6.7
D= 4.98
Avg. Staff Size= 2 approx.
Productivity= 0.4

6.3. Outcome

The effort calculated is less than that calculated by the existing system. We conclude that after Merging FP with COCMO the effort was improved and estimation becomes very easy.

Table 8: comparison between the Efforts using proposed Algorithm

Sprint	*Actual Effort*	*Effort Calculated using the Algorithm*
Month A	10	6.7
Month B	19.5	12.5

7. Conclusion

When Scrum was not used these metrics didn't yield such great values but after the use of Scrum Methodology and these tools these metrics yield better and improved values which made the product very reliable, effective, efficient, available, defect free and much improved than before. After the case study we conclude agile methodology is very effective in software development. Scrum produces a product with high quality and customer's satisfaction is given the highest priority. Quality of the product is the main objective of Scrum Methodology. By merging the two models we can even improve the effort applied and thus we can save the cost.

By merging FP with COCOMO the effort can be improved. Thus if we implement this in the existing system the cost and effort estimation could be improved and it can enhance the effort.

References

[1] Monika Agarwal, Prof. Rana Majumdar, "Tracking Scrum Projects Tools, Metrics and myths about agile", www.ijetae.com (ISSN 2250-2459, Volume 2, Issue 3, March 2012.
[2] Software Engineering (3rd ed.), By K.K Aggarwal & Yogesh Singh, Copyright © New Age International Publishers, 2007, pg no 41 to 72
[3] SilkCentral Test Manager 2011 Release Notes
[4] Reiner R. Dumke, Andreas Schmietendorf, Martin Kunz, Konstantina Gorgieva, " Software Metrics for Agile Software Development", http://www.dasma.org, December 2007.
[5] Laurie Williams, Gabe Brown, Adam Meltzer, Nachiappan Nagappan, "Scrum + Engineering Practices: Experiences of Three Microsoft Teams",
[6] Schiffauerova, A. and Thomson, V., "A review of research on cost of quality models and best practices", International Journal of Quality and Reliability Management, Vol.23, No.4, 2006
[7] www.openair.com
[8] The Scrum Guide, "1991-2013 Ken Schwaber and Jeff Sutherland"
[9] "analyzer express silkcentral test manager", © 2012 Micro Focus Limited.
[10] http://en.wikipedia.org/wiki/Burn_down_chart
[11] http://www.ligo.caltech.edu/docs/G/G080616-00/G080616-00.pdf
[12] http://www.imcnetworks.com/Assets/DocSupport/WP-MTBF-0311.pdf
[13] file:///C:/Users/hp/Downloads/TortoiseSVN-1.8.4-en.pdf
[14] Thomas Fehlmann, "Defect Density Prediction with Six Sigma", http://www.e-p-o.com/downloads/defectdensityprediction.pdf
[15] http://www.ifpug.org/ISMA6/Thomas-Defect%20Density%20Measurement-Sept14.pdf
[16] http://people.scs.carleton.ca/~jeanpier/304W14/T1-%20intro%20and%20processes/extra%20material%20on%20Agile%20Methods/The%20Agile%20System%20Development%20Life%20Cycle%20(SDLC).pdf
[17] http://www.softwaretestinggenius.com/different-software-quality-metrics-used-by-expert-test-managers
[18] http://www.nohau.se/silkcentral-test-manager