IJIRST –International Journal for Innovative Research in Science & Technology| Volume 1 | Issue 12 | May 2015
ISSN (online): 2349-6010

Software Engineering - Agile Methodology using SCRUM

Abhishek Borad
Student
Department of Computer Engineering
Hasmukh Goswami College of Engineering, Vahelal
Ahmedabad, India

Prof. Indr Jeet Rajput
Professor
Department of Computer Engineering
Hasmukh Goswami College of Engineering, Vahelal
Ahmedabad, India

Abstract

As agile software development becomes a trend, Lots of organizations are using the agile methodology for better life cycle of product development. To achieve the quality product methods needs to be the more structured definition of what agile methodologies is and what is the better level of its maturity. Traditional development methodologies rely on documents and process defined and used from one analyst to the next. SCRUM has some key challenges to use it. First customer's goals are set without the knowledge of their requirements and needs. The client's requirements are not be good enough from the need perspective. Then, given the fact that a client thinks in terms of the minimum organizational set of features in a just-in-time process.

Keywords: agile methodologies, SCRUM

I. Introduction

The selection of a software development methodology is critical process in any software development. In this approach, the entire course was designed to fit Scrum's principles and adding some extra feature which will overcome some previous problems and make it advance. Also, the course's main project was to be developed in sprints, as proposed in Scrum.

As negative points, we highlight the impossibility of delivering complete products in earlier sprints, and some interaction and collaboration difficulties. The main conclusion of this research is that, for the approach to work in software companies, a regularly modified version of the Scrum methodology (Advance Scrum) is necessary.

II. Traditional Software Development Methodologies

A software development method helps the developer to select best methods among all methodologies. The software market demands readiness for unexpected and quick changes, and the earlier software engineering methodologies did not seem to be appropriate for many practitioners.

Agile methodologies, such as Scrum, are a response to this demand. As a direct consequence of this evolution, there is an increasing need to place more emphasis on teaching Scrum in order to prepare students for their professional careers [1].

By adopting a Scrum-based approach to teach software development technologies and software engineering concepts. In the approach, some Scrum practices, such as iterative and incremental development based on sprints, were integrated into the academic environment, so that the learning itself could happen in an iterative and incremental way.

To fit this format, the entire course had to be reorganized so that the disciplines could deliver the necessary contents for the sprints to succeed [2].

A. Waterfall Model

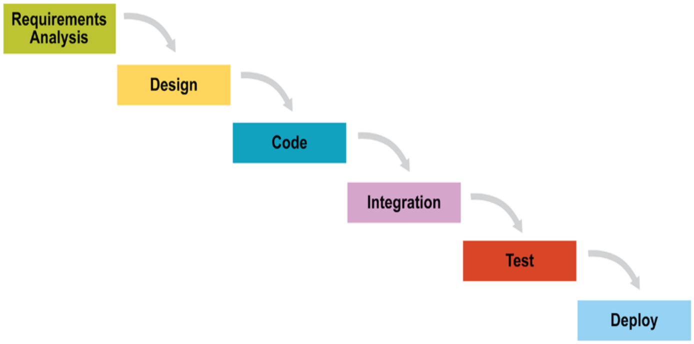


Fig. 1: Waterfall Model

Waterfall methodology is sequential model which follows the steps by step execution of the modules. There are various modules like Requirement Analysis, Design, Code, Integration, Test and deployment is the last. As the organizations learned more about developing software, certain methods for managing and predicting the cost of software development projects used in real world. The methodology managing software projects through the five stages shown in Figure.

A manifesto for waterfall software development described as below:

We can managed the software development projects by:

- Writing specifications and knowledge gathering that define how the software will look and what it will do.
- Do the design work and depth analysis to get development costs.
- Ensuring software developers obey the specifications.
- Testing the software and make sure it works as per the requirement after implementation.
- Delivering the final product to the client.

If these specifications are fulfilled then we can say the software will satisfy the customer, will be in budget, and will be delivered on time.

One of the most important differences between the agile and waterfall approaches is that waterfall gives the complete software at the end of its cycle while agile is working in sprint. Agile provides the repetitive outcomes after each cycle and complete the implementations as per the customer's requirements.

Waterfall works like it is possible to have perfect knowledge gathering of the requirements from the start. But in software development, clients often don't know what they want and can't pronounce their requirements. With waterfall, development rarely got what the customer wants even if it is what the customer asked for.

1) *Problems:*

- All works and developments are steps by steps which is not realistic.
- A good model do iteration between consecutive steps.
- However, testing comes at the end and may leads to the problems in the initial design
- The solution: do it twice

B. Iterative Model:

Iterative software development methodologies cyclically conduct and finish the defined phases. The main objective of methodologies with this particular proceeding strategy is to receive early customer feedback in order to identify shortcomings as soon as possible. Figure exemplarily shows that all components are already implemented after the first iteration. The later iterations only serve for refinements. At each iteration, design modifications are made and new functional capabilities are added [2].

The iteration will do the redesign and implementation of each module is to be simple, understandable and successive. Each module consist of the Planning, Analysis, Design, Implementation, Testing and Deployment. The analysis of an each iteration is based upon client's feedback. The project control list is changed as per the analysis results.

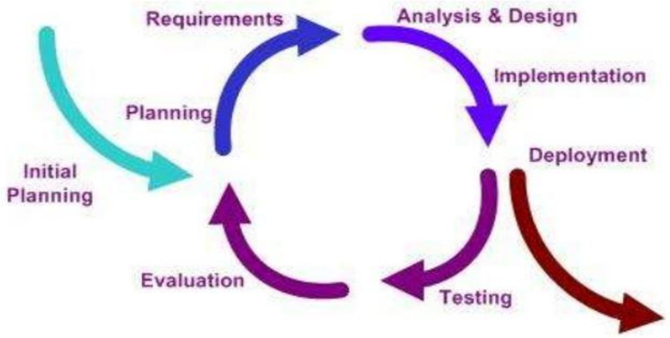


Fig. 2: Iterative Model

C. Spiral Model:

The spiral model is similar to the incremental model, with more emphases placed on risk analysis. The spiral model has four phases: Planning, Risk Analysis, Engineering and Evaluation. Analogously to the iterative proceeding strategy, the phases are cyclically conducted. However, after each cycle only a particular increment is delivered. Therefore, these increments respectively represent a subset of the whole system. In order to continuously minimize the risk of the project, the most critical components are usually implemented in the early iterations. Figure exemplarily shows how increments primary serve as extensions, not as refinements.

The baseline spiral, starting in the planning phase, requirements are gathered and risk is assessed. Each subsequent spiral builds on the baseline spiral. Requirements are gathered during the planning phase.

In the risk phase, a process is identifying the risk and solutions for it. A prototype phase is after the risk analysis phase. Software will be generated in the engineering phase. Last stage is testing. The evaluation phase provide the output of project to date before the project continues to the next spiral.

1) *Advantages:*
- Risk analysis on requirement.
- Better for critical and hard projects.
- Software will be generated early in initial cycle.

2) *Disadvantages:*
- It is costly model.
- Highly specific expertise needed in risk analysis.
- Risk analysis phase decide the software's success.
- Not for suitable for smaller projects.

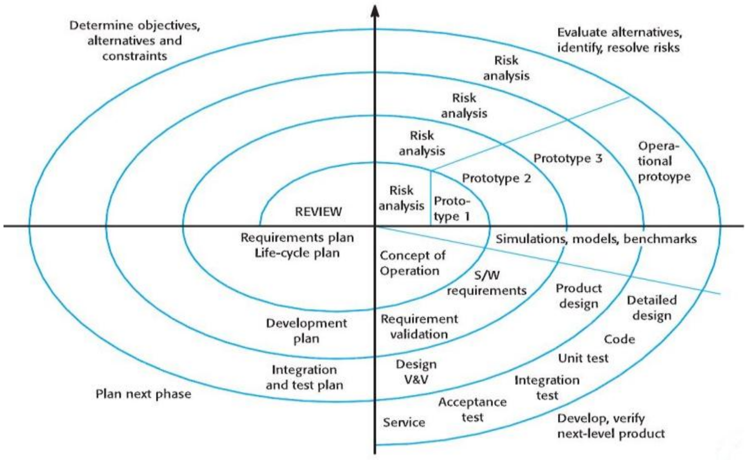


Fig. 3: Spiral Model

D. V - Model

V-Model life cycle is a sequential execution of all modules. Testing is important in this model. Testing will be done early in this model. All the test scripts as per the requirements are generated earlier than other models. As per the figure requirements gathering starts in life cycle model just like the waterfall model. Before software development is started, a system test plan is generated. The test plan obey to meet the requirements specified in documents [12].

1) *Advantages:*
- Simple and easy to use.
- Each phase has specific outcomes.
- Succession ratio is higher than waterfall model due to the early development of test plans.
- Works good for small projects where requirements are understandable.

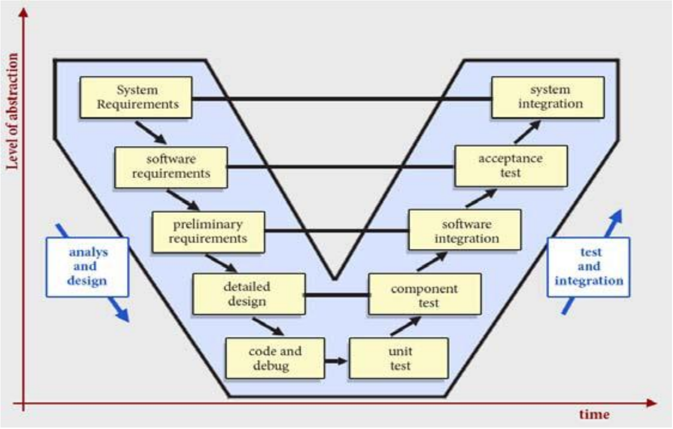


Fig. 4: V – Model

2) *Disadvantages:*
- Very strict as waterfall model.

- Software will be developed during the development phase, so no early prototypes of the software will be generated.
- Not providing the clear problems during the testing phases.

III. Introduction to Agile Methodology

Agile methods regarded as set of practices and techniques which have specific principles and values. They share the property of iterative incremental development that tackles requirement changes quickly, satisfy customer and produce quality products.

Over the last three decades, software development has been governed by process standards like CMM, CMMi, ISO and development methodologies such as structured, object-oriented and agile. From sequential, heavy weight methodologies which are more predictive in nature and lengthy in process, today there is a shifting focus towards simple, light weight methodologies involving prototyping which provides a part of the final output at every stage of development.

Most of studies have reported software project failures is due to lack of requirements gathering [24]. Global software development and agile methodology signify two important developments in software development today with respect to geographical scope and development process, intended to deliver cost efficiency and improved customer satisfaction. Global software development reduces cost by offshoring development to remote locations where resource pool is available at low cost but comes with inherent challenges of communication and coordination. Traditional agile methods recommend entire team working in single location closer to customers improving communication and coordination [25].

Agile methods regarded as set of practices and techniques which have specific principles and values. They share the property of iterative incremental development that tackles requirement changes quickly, satisfy customer and produce quality products. Their target is to improve business and innovate new ideas to meet the market demands by quickly redefining resources when the requirements or technology changed, fast response to the market changes or insensitive customer interaction [12].

The most popular agile methods within industry are XP, Scrum, Crystal Family and DSDM. While agile manifesto stresses less on documentation, any large organization is bound and governed by process related capability for which agile requires more changes. Segal suggested that with a balanced level of process and documentation agile methods can be adopted to improve the value for customer.

Bhalerao, Puntambekar and Ingle describe agile software development lifecycle as similar to traditional lifecycle methodology and argue that agile as a methodology is not a new paradigm, but a model which evolved from traditional method with some variations. All quality processes like testing, validation and verification occur much early in agile methods owing to its iterative nature and the development process then progresses through multiple sprints. Agile methods also improve the final output quality through periodic reviews [23].

Some studies have reported that agile methods provide improved customer satisfaction, flexible customization and usability, learning and adaptation and capability to adjust amicably for volatile business requirements [24]. Dyba and Dingsoyr [25] observed that agile methodology strongly advocates a shift in approach from leadership-collaboration to command and control, suggesting its suitability for organizations which embrace innovation and which are adaptive to changes rather than bureaucratic organizations.

However research in this area does not clearly articulate the demarcation of balance required between traditional and agile methodologies. The insistence is on agile manifesto principles but many large organizations are governed heavily by process orientation. While it is strongly discouraged for such organizations, an alternate approach for improved customer satisfaction is not clearly articulated. [12].

The principle is developing on a test-driven basis; that is to write test prior to writing code. A test suite is run on the application after any code change.

A. What is Agile?

Agile model works with the iterative approach. All the module are in iterative modes and repetitive at each sprint. The sprint is the collection of the all modules and all the phases. Each phase consists of the Analysis, Design, Implementation and Testing.

Here is a graphical illustration of the Agile Model:

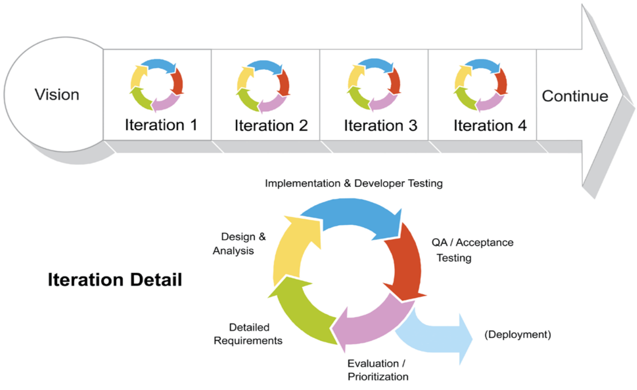


Fig. 5: Agile Model

B. Agile Principles:

The Agile Manifesto is based on following principles [5]:

- Repeated delivery to satisfy customer.
- Self-organizing teams.
- Capable of timely development.
- Software will be developed through the developers who are expert, motivated and trusted.
- Daily attention to coding quality and great design.
- Daily meeting to cooperation between business analyst and developers.
- Regular adaptation of the changing requirements.
- Welcome modified requirements, at last stage in development also.
- Proof of concepts like Demo, face-to-face conversation is the best form of communication.
- Delivery if software frequently like every week.

C. Agile Model Pros:

- Is a very realistic approach to software development
- Minimum Resource requirements.
- Provide teamwork and provide cross training.
- Gives flexibility to developers.
- Worthy for pre-defined or switching requirements.
- Easy to manage.
- Good model for environments that change steadily.
- Minimal rules and easy to document.
- Can be parallel development and delivery can be done as per planned.

D. Agile Model Cons:

- Not suitable for handling complex project.
- More risk of properties, new feature implementation.
- Fixed delivery date dictates the scope, functionality needs to be given on time, and meet the project plan.
- An agile leader and agile PM practice is a must without which it will not work.
- Minimum documentation leads to dependency on single person.
- Due to lack of documentation, transfer of technology to new development team may be challenging.
- Success of developed product is heavily relay on user interaction, so if client is not clear, they can be lead to the wrong direction.

E. Why Agile?

Most of software companies are working on latest technology and reason behind of these are following:

- To get better output.
- Better cost by less resources.
- Quality product.
- User and developer interaction makes the decision easy and quick.
- Easy to maintain product.
- If teams or developer changes then no impact should be on business.
- Easy to train people about developed product.
- Changeable product.
- Deadline meets as per delivery Plan.

Because of these points Agile are currently used by many organizations.

The well-known agile software development methods and/or process frameworks include:

- SCRUM
- Extreme Programming
- Crystal methodologies family
- Feature-Driven Development
- Adaptive Software Development

IV. AGILE METHODOLOGY USING SCRUM

A. *What is SCRUM?*

Scrum is an iterative and incremental process. The requirements that is being developed known as the product backlog is described and discussed among the developer members. The product backlog is divided into smaller requirements that are assigned to the team members. The set of tasks that the team works on during a sprint is called the sprint backlog. A physical board called the scrum board is used to keep track of all these tasks and assignments

In Scrum all requirements are divided in specific blocks, called Sprints, which are generally take two weeks or 30 days to develop. Scrum teams adhere the rules of method and developed the product requirements in each iteration.

There are three pillars of SCRUM:

- Transparency
- Inspection
- Adaptation

B. *Roles in Scrum*

- Product Owner
- Team
- Scrum Master.

The work of the traditional project manager role are divided among these three Scrum roles.

Fig. 6: Roles in Scrum

1) *Product Owner:*

- Decides when to ship.
- Contribute as a team member
- Responsible for product development.
- He can accepts or rejects each product iteration.
- Considers client's interests.
- He has full responsibility to decides whether to continue development.
- Continuously re-prioritizes the Product Backlog, decides any long-term prospects such as release plans.
- Final person of requirements questions.
- He is responsible for return on investment.

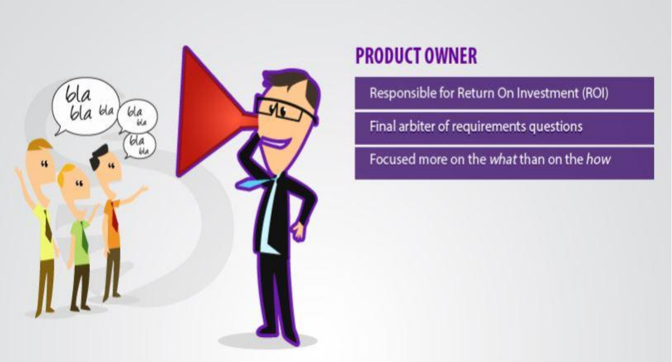

bla bla bla
bla bla
bla bla
PRODUCT OWNER
Responsible for Return On Investment (ROI)
Final arbiter of requirements questions
Focused more on the what than on the how

Fig. 7: Product Owner

2) *Scrum Development Team:*

- It is a Cross-functional Group. It include software developers: domain experts, business analysts and all members with testing skills etc.

- They decide how to reach commitments.
- Intensely cooperative.
- 7 ± 2 members
- Most successful with long-term, full-time membership. Scrum moves work to a flexible learning team and not moving people or splitting them between teams.
- Decide commitments with the Product Owner, one Sprint at a time.
- Self-organizing / self-managing, without externally assigned roles.

SCRUM DEVELOPMENT TEAM
Cross-functional group
Attempts to build a "potentially shippable product increment" every Sprint
Collaborates
Self-organizing

Fig. 8: Scrum Development Team

3) *Scrum Master:*

- Enforces timelines.
- Helps resolve problems.
- Facilitates the Scrum process.
- No management authorities.
- Creates an environment conducive to team self-organization.
- Shields the team from external interference and distractions to keep it in group flow.
- Keeps Scrum artifacts known.
- Promotes improved engineering practices.

C. *Artifacts in Scrum:*

- Product Backlog
- Sprint Backlog
- Increment

The Product Backlog is a collection of the requirements provided by the Product Owner. These are the finalized requirement from client side. Product Backlog Items are also known as User Stories. In priority wise all the requirement are ordered as per figure. If any of requirements are reprioritized the list need to be modified and discussed in meeting for knowledge purpose. This will be discussed in the Backlog Refinement Meeting module [22].

Fig. 9: Scrum Master

The Sprint Backlog is the work which need to be done via dividing it to smaller piece. So user stories are divided into the smaller list like 100 stories are divided in to 10 block where each block contain the 10 stories.

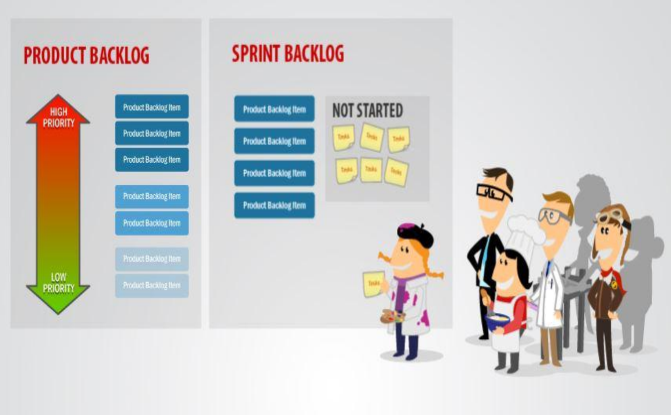


Fig. 10: Product Backlog

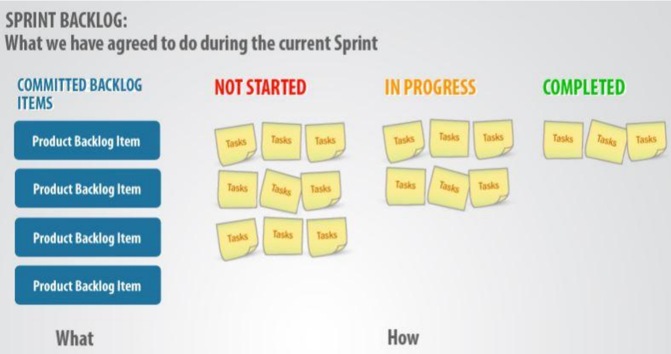


Fig. 11: Sprint Backlog

D. Scrum Meetings:

Scrum master is the responsible person for all the following meetings. But he has no authority to take any decision in these meetings. Just he can manage it.

Scrum has five meetings:

- Sprint Planning
- Daily Scrum
- The Sprint Review Meeting
- The Sprint Retrospective Meeting
- Backlog Refinement

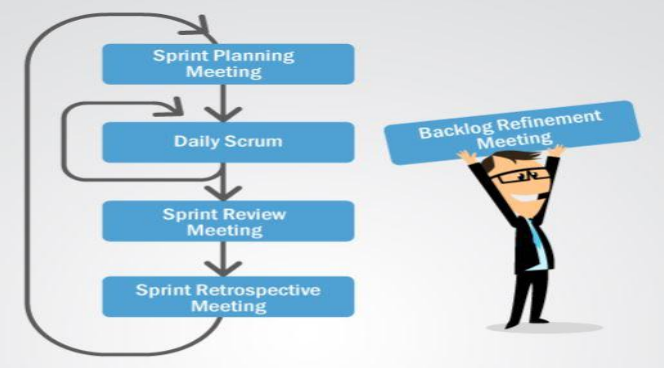


Fig. 12: Scrum Meetings

1) Sprint Planning:

In this meeting which tasks need to be set as a higher priority and which need to be set as a lowest priority order that is decide. Product owner is responding to the team’s argument over the requirement and guide them to the teams. At the beginning of each Sprint, they discussed which item they need to developed or not. Then they take the work from product backlog to sprint back log [22].

Team has to work together and complete the task which has priority high and complete the sprint backlog within the time period given. This rule need to be adhere by all developers so the quality and the timeline should be followed properly.

If development team has any problem and wants to change the tasks priority or need to remove it from the current sprint back log. They can discuss this with product owner and finalize the outcomes. There is a time limit to plan the sprint means for 30 days sprint they can spend 8 hours and proportionally.

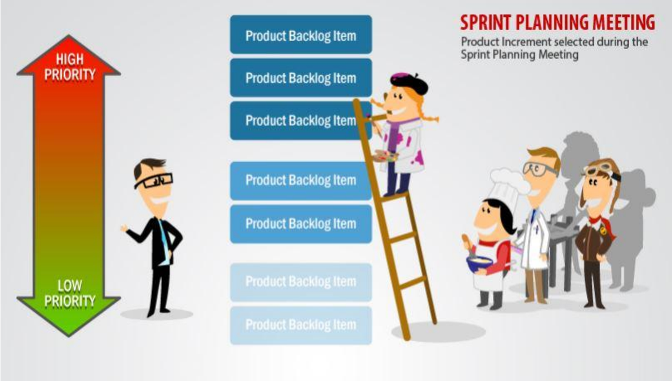


Fig. 13: Sprint Planning

2) Daily Scrum:

In daily scrum meeting, everyday scrum development team spend 15 minutes time and discussed with every developer what will they do today. What did they do yesterday? What is the current progress of user stories? Is there any difficult they faced during development etc. Some critical issues needs to be addressed in this meeting and finalize the outcomes of it.

Fig. 14: Daily Scrum Meeting

Team also taking care of the current and future progress plan. They also making the effort to obey that plan. Each developers and team member has a responsibility to complete their tasks within time.

3) The Sprint Review Meeting:

In Sprint review meeting, the team will demonstrate the work in front of the Product Owner. This is a live demo meeting, not a report presentation. After the demonstration, the Product Owner review the requirements and see the demo. Is the product fulfilled the requirements? If so then Product owner sign well and allows to move ahead with the other sprint development otherwise as per the comments and feedback received from the product owner.

Fig. 15: Sprint Review Meeting

Scrum development team need to start rework and modified the code based on the feedback and changed. The scrum master helps to convert the feedback in to the requirements and make sure to adhere to that requirements in next sprint review meeting.

If there will be any new requirements from the product owner than new user stories can be determined by the scrum master and will add this to the sprint backlog.

4) The Sprint Retrospective Meeting:

In this meeting, after the demonstration what thing goes well and what was not well will be discussed. After getting the feedback all need to be expert to solve the current situation and found the best way in less time to implement it [22].

Scrum master finds the alternative way to implement those new or modified requirements so it can be done in less time. Discuss the stories which need to be modified and which need to be frizzed.

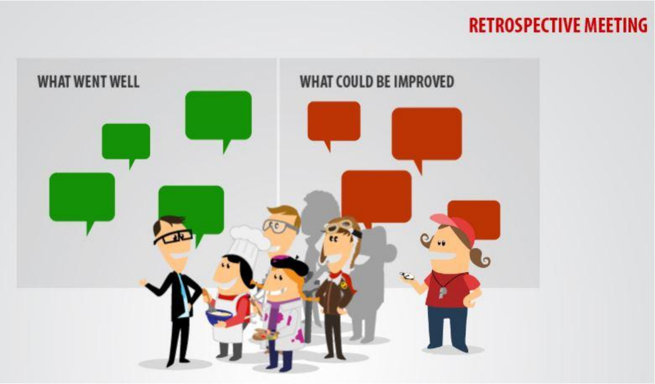


Fig. 16: Sprint Retrospective Meeting

5) Backlog Refinement Meeting:

In the Backlog Refinement Meeting, the team spend the time to complete the team considers the effort they would expend development and provides other technical helps to the product owner to prioritize the product backlog. Scrum master helps them to prioritize them.

A skilled Scrum Master can help the team to work accordingly stories and also help the product owner to make the stories that way the developer team can follow the project plan and complete the sprint on time.

Fig. 17: Backlog Refinement Meeting

Figure describe the all over process of the scrum which we have discussed till now.

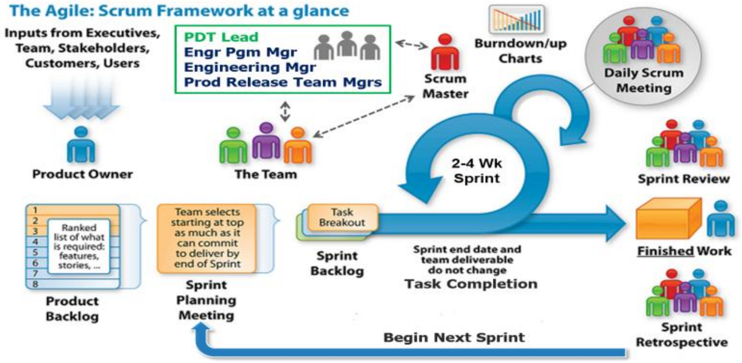


Fig. 18: The Agile - Scrum Framework at a glance

V. Implementation

A. Features Required:

Any software's quality is depend on the quality of code which is the overall quality of code developed by the developers. Quality of code is the important factor for the software development cycle. Base on this parameter the final product or software is rated and compared in market with other products. Software's development time is also the key feature.

Time to develop the code and remove the false code which is causing fault. All this factors are not included in the scrum method which should be a part of method to improve the code quality.

Scrum methodology limitation is that method is not focusing on the quality of code which is the key factor of any software development. Method is not managing the assignment of tasks to developer whose code quality is better than others.

B. Code Review Tool:

To overcome critical factor code quality, code review tool has been developed to fulfill the coding standards. Among them, coding standards are the fundamental way to improve code readability, help individual developers produce consistent code, and prevent error-prone coding styles.

To improve the quality of software, assignment of tasks/requirements to the developers are scanned/reviewed by code review tool. Tool provides the score of developed code and also provides the expert comments/violations to the developer to improve the quality of code.

This tool scans the .NET code. Based on the comments, developer can make necessary changes and improve the quality of code. Each developer's score can also be used at a time of task assignment to the developers. Daily work plan and weekly report of each developer can give the track of tasks assigned.

The code review tool scan the code of developers and provide the necessary comments which need to be followed by developer to achieve the better quality of code. Tool provide the recommendations to the developer how he can correct the existing code.

To improve the scrum process this tool can help to consider the quality factor of the software. Most of time is taken by the code review process when the code is developed. Review process is the vital and important aspects of software development.

This tool reduce the review process time and provide the quick solution for each violations for which developer can easily implement and improve the quality of code. Daily work plan report and weekly report of developer can give the exact scenario of the tasks and help to analyze the software timelines.

C. Tool Framework:

1) Input:

- .NET Project folder path or .NET single code file path can be given as an input. (Ex. "D:\Demo\" or "D:\Demo\Demo.cs")
- Using the input path, tool will get the file(s) which need to be reviewed.
- If .NET project folder path is provided for scanning then tool will automatically get the collection of files and store it on INPUTFILES collection object otherwise single code file will be store in INPUTFILES object.

2) .NET Framework:

- .NET framework scans each and every code files of project sporadic from INPUTFILES object.
- When tool runs on any project folder (collection of files) or single code file, first specific XML rules are fetched from Rule Engine.
- Tool Engine select one file from INPUTFILES object and store content of that file in TEMP Variable.
- Scanning process implements all rules on TEMP variable for the results whether rules are meeting the requirements or not.
- Implementation of each rules in Tool Engine are defined in C#.Net code.

3) Rule Engine:

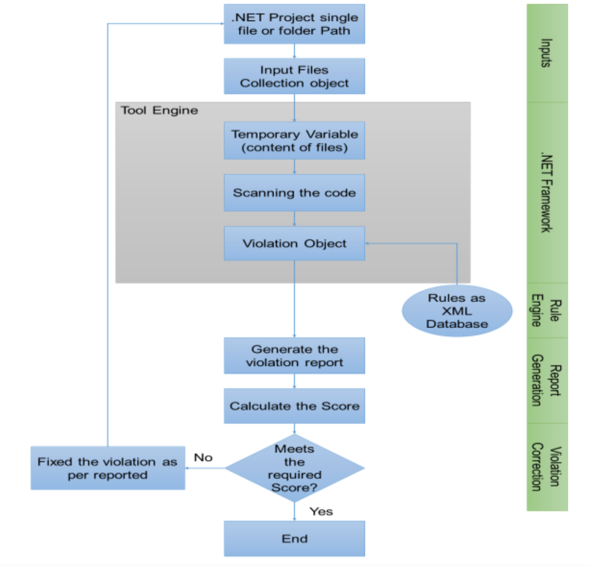

.NET Project single file or folder Path
Input Files Collection object
Tool Engine
Temporary Variable (content of files)
Scanning the code
Violation Object
Rules as XML Database
Generate the violation report
Calculate the Score
Meets the required Score?
No
Yes
Fixed the violation as per reported
End
Inputs
.NET Framework
Rule Engine
Report Generation
Violation Correction

Fig. 19: Process Flow

- Rules are predefine in XML document and stored in XML file. These rules are standard for the development of code.
- Each and every rule’s description, severity, resolution are defined in this document.
- Severity 1 means Information, severity 2 means Warning and severity 3 means Error.

Tool provided the resolution details for each rules defined in XML database so developer can solve the violations.

4) *Report Generation:*

- If any rules are not meeting the requirements and violating the defined rules then violations are generated for that rules. All violation are store in Violation collection object.
- At the end of all code scan these all violation are reported and report can be exported for further analysis of violation.
- Report displays the total number of violation, violation rule’s description, severity and resolution steps to fix or identify the violation.

5) *Violation Correction:*

- Violation reported in report needs to be solved by developer to achieve the good quality code. Developer need to solve the violation so score which is predefined for quality code can be achieved.
- After solving the violations tool needs to be run again to make sure that current code has better quality than earlier.

D. Advantages:

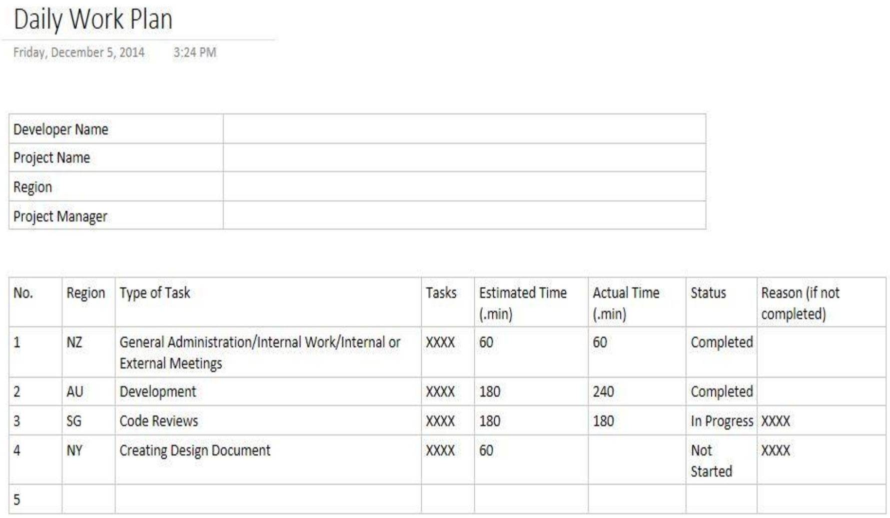
- Reduce the code review effort and less chance of human error while reviewing.
- Code review can be quick and save lots of time.
- Provide solution to solve the violation.
- Easy to use the tool. In 2-3 click, report can be generated.
- Ideally, the person is a developer that you admire, but almost any developer will do. If someone can't understand parts of your code, that's a red flag. But this tool will work as same reviewer for all developer.

E. Daily Work Plan:

The first, and most important step to take is to plan your work. This can make your work efficiency more than previous one. It is important that your plan be written down somewhere - on a calendar, note pad, or stored on your computer.

If you have written plan somewhere then it will helps you to remind and help to prevent from the disaster because of you forget something important. Here is the sample format of daily work plan which you can send it to your manager for daily. So he can also track that on which items you are currently working now.

Manger also get the idea that which person is equipped with some amount of work and which are not. So he can assign some other or critical tasks to them. Plan can give you a better picture of the work and you can work accordingly and prevent for being confused.



Daily Work Plan

Friday, December 5, 2014 3:24 PM

Developer Name	
Project Name	
Region	
Project Manager	

No.	Region	Type of Task	Tasks	Estimated Time (.min)	Actual Time (.min)	Status	Reason (if not completed)
1	NZ	General Administration/Internal Work/Internal or External Meetings	XXXX	60	60	Completed	
2	AU	Development	XXXX	180	240	Completed	
3	SG	Code Reviews	XXXX	180	180	In Progress	XXXX
4	NY	Creating Design Document	XXXX	60		Not Started	XXXX
5							

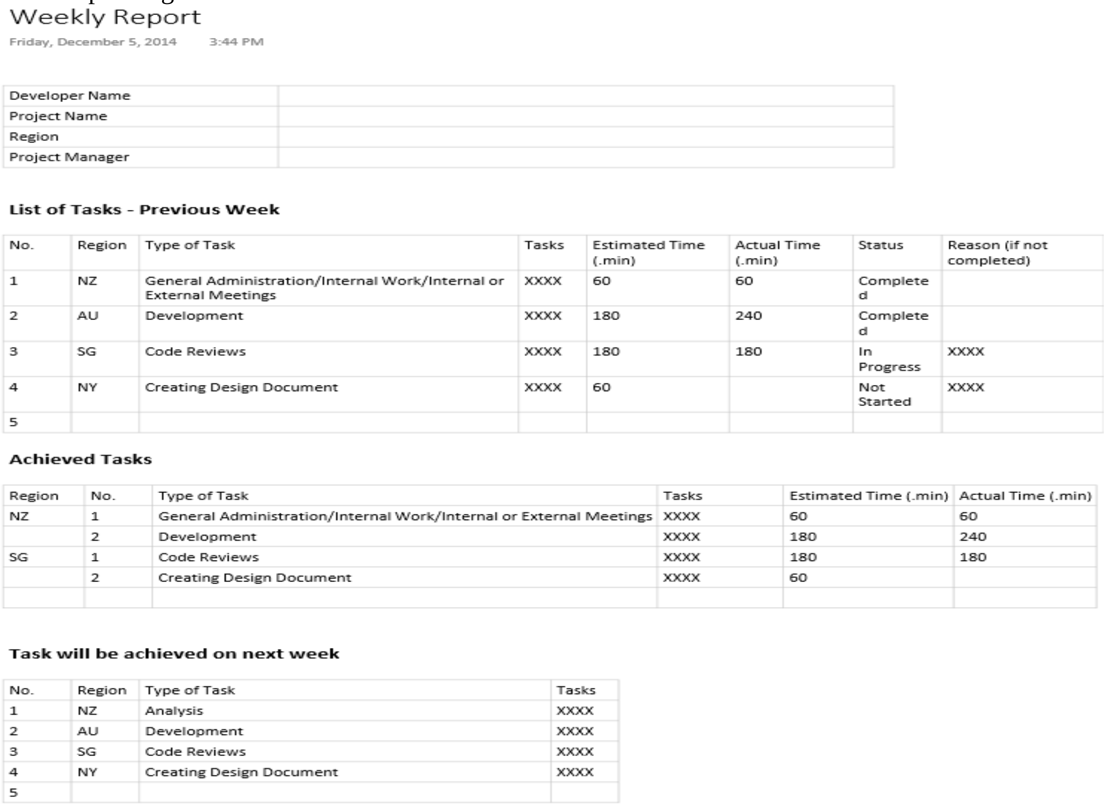
Fig. 20: Daily Work Plan

- Getting Rid Of Daily Distractions with a Plan
- A Plan Can Help You Avoid Taking on More Than You Can Handle
- Creating Your Attack Plan

F. Weekly Progress Report:

The Weekly status report is used at the manager level to get the view of the current work progress and based on that they can decide the future work progress. It will decide the progress of each quarter. Each manager can build the report and provide it to upper level so top management can get the idea of the work simulation among the company. So they can take some necessary steps to work on next quarter.

Whether this report can be positive or negative based on that op management can take the further steps. Figure describe the completed tasks and pending tasks with their reason.



Weekly Report

Friday, December 5, 2014 3:44 PM

Developer Name	
Project Name	
Region	
Project Manager	

List of Tasks - Previous Week

No.	Region	Type of Task	Tasks	Estimated Time (.min)	Actual Time (.min)	Status	Reason (if not completed)
1	NZ	General Administration/Internal Work/Internal or External Meetings	XXXX	60	60	Complete d	
2	AU	Development	XXXX	180	240	Complete d	
3	SG	Code Reviews	XXXX	180	180	In Progress	XXXX
4	NY	Creating Design Document	XXXX	60		Not Started	XXXX
5							

Achieved Tasks

Region	No.	Type of Task	Tasks	Estimated Time (.min)	Actual Time (.min)
NZ	1	General Administration/Internal Work/Internal or External Meetings	XXXX	60	60
	2	Development	XXXX	180	240
SG	1	Code Reviews	XXXX	180	180
	2	Creating Design Document	XXXX	60	

Task will be achieved on next week

No.	Region	Type of Task	Tasks
1	NZ	Analysis	XXXX
2	AU	Development	XXXX
3	SG	Code Reviews	XXXX
4	NY	Creating Design Document	XXXX
5			

Fig. 11: Weekly Progress Report

G. Quarterly Report:

The Quarterly report are created by using the weekly report which can give the leaders details about the performance measures through out of four months. This report can also provide the details about the future project work and approx. timeline of the projects.

Data used in this reports are the collection of the each and every employee in company. So company can measure the overall performance. Also the feedback of each team will be provided by the leaders to improve the throughput of each team which leads to overall outcomes.

The report format are depended on the leader which type of artifacts they want to consider to measure the performance. Artifacts may be different for leader to leader. But the performance outcome should be satisfactory.

H. Minutes of Meeting:

Meeting minutes are the written or recorded documentation that is the description of what was done in the meeting. This is carried out during the meetings so nothing can be missed and all thing get recorded.

Minutes usually include:

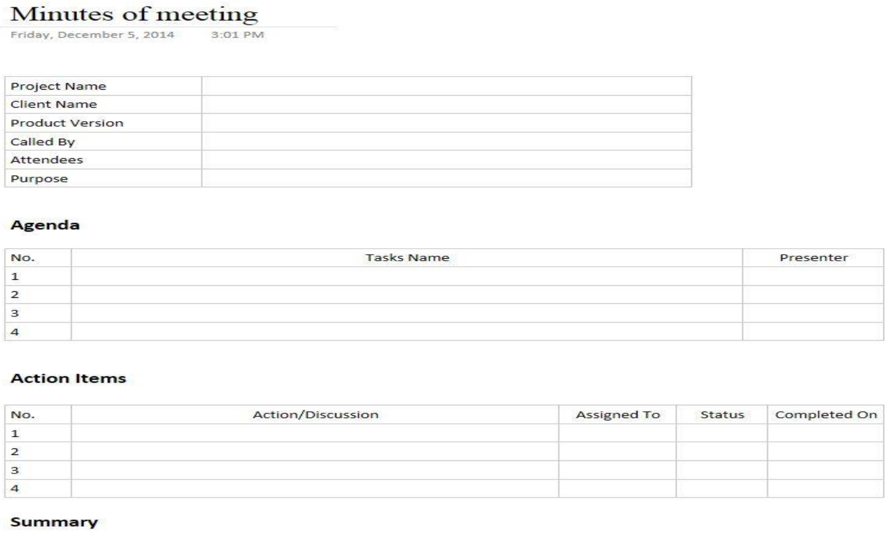
- The names of the participants.
- The agenda.
- Action items.
- The follow-up by participants.
- Due dates for the completion of action items and history of action items.

1) Why write meeting minutes?

May be we think that meeting of minutes is the unimportant task but it is very important. To track each action work for each individual can be monitor by this minutes. There may be happened that your colleague has different idea about your work if this was not carried out.

If you have a proper document then you can claim anytime that following task was assigned to me and was done at that time. Figure give us an example of minutes which can be followed by organization.

If tasks was not carried out at the time of meeting then going to the previous document of meeting of minute can give idea which tasks/action was assigned to whom. No blame game can be happens if proper documentation is available. It is a good practice to write a meeting of minutes.



Minutes of meeting

Friday, December 5, 2014   3:01 PM

Project Name	
Client Name	
Product Version	
Called By	
Attendees	
Purpose	

Agenda

No.	Tasks Name	Presenter
1		
2		
3		
4		

Action Items

No.	Action/Discussion	Assigned To	Status	Completed On
1				
2				
3				
4				

Summary

Fig. 22: Minutes of meeting

I. *Splitting the Product Backlog:*

The splitting the product backlog into the sprint backlog is one of the most important task. To avoid the conflicts between the developers during the development. It is important to give importance to the splitting tasks.

When two developers are working on the same functionality there are major possibility of conflicts. To avoid the possible attack splitting of Product Backlog should be done wisely.

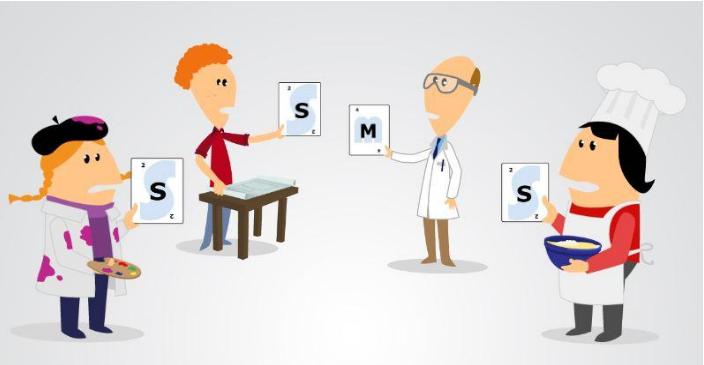


Fig. 23: Splitting the Product Backlog

Before doing the tasks splitting, each developers should have a quick look of product backlog and sprint backlog. Decide that whether any requirements are conflicting. If no then they are good to go ahead, otherwise they have to inform this to product owner or scrum mater to resolve.

If there are any conflicts then priority of those tasks should be neglected. So it can be put on another sprint or given to the same developer. To avoid these scenario, product owner or scrum mater should differentiate tasks which has similar functionality or technically similar [23].

VI. Conclusion

The code review tool scan the code of developers and provide the necessary comments which need to be followed by developer to achieve the better quality of code. Tool provide the recommendations to the developer how he can correct the existing code. To improve the scrum process this tool can help to consider the quality factor of the software.

Most of time is taken by the code review process when the code is developed. Review process is the vital and important aspects of software development. This tool reduce the review process time and provide the quick solution for each violations for which developer can easily implement and improve the quality of code.

Daily work plan report and weekly report of developer can give the exact scenario of the tasks and help to analyze the software timelines.

REFERENCES

[1] Astemark, “Agile@ITTE“, Internal Ericsson Documentation, Sweden 2011

[2] D. Duka, “Agile approach in Software Development“, Proceedings of the 35th Mipro CTI conference, 2012

[3] Sääskilahti, J. Röning “Challenges with Software Security on Agile Software development, Internal Ericsson Documentation, Sweden 2011

[4] R. Winston, “Managing the Development of Large Software Systems“, Proceedings, IEEE WESCON, August 1970, pp.1-9. http://www.cs.umd.edu/class/spring2003/cmsc838p/Process/waterfall.pdf

[5] Beck, “Manifesto for Agile Software Development”, 2001 http:// Agilemanifesto.org

[6] and T. Poppendieck, “Lean Software Development Principles”, http://www.poppendieck.com/

[7] T. Stober, U. Hansmann. "Overview of Agile Software Development", Agile Software Development: Best Practices for Large Software Development Projects, Springer. 2010.

[8] S. W. Ambler, “Agile Enterprise Architecture”, http://www.Agiledata.org

[9] S. W. Ambler, “Agile Architecture: Strategies for Scaling Agile development”, http://www. Agilemodeling.com.org

[10] K. Mikkonen, “How We Learned to Stop Worrying and Live with the Uncertainties“, Internal Ericsson Documentation, Sweden 2011

[11] Rising, L. and Janoff, N.S. Jul-Aug 2000. “The Scrum software development process for small teams”, Software, IEEE, vol.17, no.4, pp.26-32.

[12] Mahnic, V. “A Capstone Course on Agile Software Development Using Scrum” Feb. 2012. IEEE Transactions on Education, vol.55, no.1, pp.99-106. doi: 10.1109/TE.2011.2142311

[13] Dahiya, D. 2010. “Teaching software engineering: a practical approach”. SIGSOFT Software Engineering Notes, Vol.35, pp.1–5.

[14] Schild, J., Walter, R., and Masuch, M. “ABC-Sprints: adapting Scrum toacademic game development courses”. 2010. Proceedings of the Fifth International Conference on the Foundations of Digital Games (FDG '10) ACM, New York, NY, USA, pp.187-194.

[15] Schneider, J.G.; Johnston, L. and Joyce, P. “Curriculum development in educating undergraduate software engineers - are students being prepared for the profession?”. 29 March-1 April 2005. Software Engineering Conference. Proceedings. 2005 Australian pp.314-323.

[16] “CentOs” Internet: http://www.centosbr.org [Apr 02, 2013]

[17] “Welcome to NetBeans.” Internet: http://www.netbeans.org [Apr 02, 2013]

[18] “PostgreSQL: The world's most advanced open source database” Internet: www.postgresql.org [Apr 02, 2013]

[19] Singh, M. "U-SCRUM: An Agile Methodology for Promoting Usability," 4-8 Aug. 2008. AGILE '08. Conference, Proceedings, pp. 555-560.

[20] Schwaber, K. and Beedle, C. 2002. “Agile Software Devlopment with Scrum”, Prentice Hall.

[21] Sink, E. "Stories from My Experiences Learning Scrum," 7-13 Aug. 2011. Agile Conference (AGILE), 2011, pp.216-222.

[22] Michael James "The Scrum Reference Card, and Other Articles". http://scrumreferencecard.com/.

[23] Nabil Mohammed Ali Munassar1 and A. Govardhan2 “A Comparison between Five Models of Software Engineering” 5 Sep 2010.

[24] S. Ghaisas, “Method for identifying unobvious requirements in globally distributed software projects,” In Proceedings of SENSE2009, Kaiserslautern, Germany, pp. 297-308.

[25] Paasivaara and C. Lassenius, “Could Global Software Development Benefit from Agile Methods?” in IEEE International Conference on Global Software Engineering, pp. 109-113, 2006.