Abstract

Agile development methods are key to the future of flexible software systems. Scrum is one of the vanguards of the new way to manage software development when business conditions are changing. However, the tricky part in agile software development is that there is no manual which tells you exactly how to do it. Scrum teams have to experiment and continuously adapt the process until it suits the specific situation and to overcome the challenges. The aim of this research paper is to address the quality challenges and issues in Scrum implementation and proposing solutions to overcome or minimize the issues. The common issues in Scrum implementation are Scope Creep, Requirement changes which is inherent, the biggest challenge of inadequate time to prepare test plans, minimal requirements documentation to prepare the test cases, highly compressed test execution cycles, minimal time for regression. In addition [10] communication and building trust is another issue if the team is working in a distributed environment. These factors may result in having an adverse impact on the quality of the product delivered. To overcome this to an extent the concept of the "principle of factor sparsity" or [13] pareto principle can be applied which states that, for many events, roughly 80% of the effects come from 20% of the causes. This was originally described by
Vilfredo Pareto and later formalized by Joseph Juran. The principle is just a rule of thumb, but an important one. Whether the percentages are really 80/20 or 70/30 or 90/10, the reality is that most things are caused by a few underlying factors. The same applies for Scrum and 80% of the quality issues would arise from the tasks that are performed related to 20% of the stories. So, concentration on the vital few stories or tasks to arrest the critical issues and having a technique to double check the code quality delivered by agile scrum teams would provide better results. UI programming is one such new technique proposed. For an increase in quality and decrease in the defects generation, the call for a UI programming can be taken by the product owner while creation of product backlog or can be taken by the team along with product owner during the sprint planning meeting when actually the tasks are defined. The tasks that are candidates for UI programming would base on the story/tasks criticality, complexity, likeliness of being source for defects bugs, the need for high quality. At times the story and all its tasks defined can be candidates for UI programming or specific tasks within the story can be candidates for UI programming. In nutshell, the UI programming can be used in any projects which are implementing scrum and are having considerable quality issues in the implementations.

References

- Coaching Agile Teams-Lyssa Adkins
- Agile Product Management with Scrum—Roman Pichler.
- Experience Report: Distributed agile: project management in a global environment Seiyoung Lee & Hwan-Seung Yong
- Agile Alliance, Principles behind the Agile Manifesto, "http://agilemanifesto.org/principles.html."
- High smith J, Cockburn A. Agile software development: The business of innovation
- Lycett M, Macredie RD, Patel C, Paul RJ. Migrating agile methods to standardized development practice
- A paper by Dr. Satya Prasad Ravi, B. Reddaiah, Lakshmi Sridhar Movva-named Distributed Agile Development: Practices for building trust in team through Effective communication
- Dr. Satya Prasad Ravi, B. Reddaiah, Lakshmi Sridhar Movva Framework to mitigate dynamic and static risks with respect to agile
- Dr. Satya Prasad Ravi, B. Reddaiah, Lakshmi Sridhar Movva, Rajasekhar Kilaparthi A Critical review and empirical study on success of risk management activity with respect to scrum
- Software Engineering-A practitioners guide –Roger S Pressman

Index Terms

Computer Science

Web Applications
Keywords

Agile methodologies  Scrum  Quality  Defects  UI programming  You and I  pair  programming  pace  task  switching  shuffling  Distributed environment