The intent and use of agile software engineering methodology's

F. Rietveld
The University of Amsterdam, Netherlands
March 17, 2013

1 Overview and intent

1.1 Intent

The intent behind agile methods like scrum and XP is to let the team within the sprint be pretty much completely free to implement whatever process they like to deliver the sprint goal.

The practice of scrum is focused on delivering the highest value to customers in iterations that allow teams to learn from each iteration and improve for the next iteration to deliver more value to the customers. (Based on the Agile manifesto [3])

1.2 Overview scrum

A typical scrum team has between five and nine people, but scrum projects can easily be scaled up, smaller 1 person–team scrum process is also a possibility. The team does not include any of the traditional software engineering roles. Everyone on the project works together to complete the set of work they have collectively committed to complete within a sprint. The team needs however the necessary skills to complete the sprint.

The product backlog is a prioritized list containing every desired feature of the product.

At the start of each sprint, a sprint planning meeting is held during which the product owner picks the top priority items from the product backlog, and the scrum team selects the work they can complete during the coming sprint.

That work is then moved from the product backlog to a sprint backlog.

Each day during the sprint, there is a brief meeting called the daily scrum it is conducted as a standoff meeting. This meeting helps set the context for each days work and helps the team stay on track. All team members are required to attend the daily scrum.

At the end of each sprint, the team demonstrates the completed functionality at a sprint review meeting, during which, the team shows what they accomplished during the sprint. Typically, this takes the form of a demonstration of the new features. The meeting must not become a task in itself nor a distraction from the process.

Also at the end of each sprint, the team conducts a sprint retrospective, which is a meeting during which the team (including its scrum Master and product owner) reflect on how well scrum is working for them and what changes they may wish to make for it to work even better.
1.3 Scrum and Extreme programming (XP)

Scrum and XP are both agile software development methodologies. Both scrum and XP implement a way of iterative and incremental development. They have a lot of overlapping techniques but they also have their differences and here is an attempt to briefly highlight the differences:

- Scrum is more high level, focusing on the management of the project (e.g., the requirements or features are managed) rather than specifying or defining engineering practice such as pair programming or test driven development.
- The length of an iteration in XP is usually 1-3 weeks whereas, in scrum sprints are 1-4 weeks.
- Once sprint (or iteration in XP) starts, in scrum the customer cannot change the requirements, in other words the customer will have to wait until the sprint finishes. In XP however, requirements can change anytime changes are embraced. (see [1]).
- In XP features are developed in a strict order whereas in scrum, the team is free to choose the features to be developed. Sequence does not matter the scrum team picks the features to implement from the sprint backlog.
- Both XP and scrum define the role of a coach, In scrum it is called scrum master most scrum processes prefer a certified scrum master, whereas, XP defines the role of coach more loosely and can be a changing member of the team.

Scrum and XP are often used together the Scrum part defines the framework and XP defines the engineering practices [1], like pair programming or TDD, and they fit together nicely. In [2] is a overview of the overlap of techniques used in several agile methods.

2 Questions

Question 1 Under what conditions will agile methods like scrum or xp work and under what conditions will they not suffice?

- Agile software development depends on an effective, day-to-day working partnership between the development team, and empowered users [6]. If this partnership does not exist, or is weak, an agile project will face much higher risks than it should.
- In scrum it is the development team who is responsible realization of the product, the scrum team needs the necessary skillset to make the implementation.
- The management team has to create an environment in which the scrum team feels itself empowered, and is, to arrange their own process.

These are the tree most important conditions for success of a scrum process, in [5] is a extensif list of possible failure and success factors.

Question 2 What are the shortcomings / limitations / tradeoffs of the agile methods?

- In case of large software deliverables, it is difficult to assess the effort required at the beginning of the software development life cycle.
- There is lack of emphasis on necessary designing and documentation. Information about the used design is mainly tacit.
- The project can easily get taken off track if the customer representative is not clear what final outcome that they want.
- Only senior programmers are capable of taking the kind of decisions required during the development process. Hence it has no place for newbie developers [7], unless combined with experienced resources. (see Appendix A)
- If the scope of the project in not defined at the beginning of the project (possible in user stories) there will be an incomplete product backlog increasing the chance of project failure.

Question 3 To what extent would it have solved the problems in the failed project you studied?

After analyzing the Therac-25 case it had become clear that during the start and development of the project there where no:
• Set of software requirements
  1. Safety based
  2. Usability based.
• Plans for activities related to risk reduction
  1. Hazard analysis (HA)
  2. Complete fault tree analysis (FTA)
  3. Failure mode effect analysis (FMEA)
• Plan for system documentation
• Plans for safety tests.

It came down to one programmer, with unclear education responsible for building complex controlling software for a machine that nowadays would have a SIL-3 classification.

Building the Therac-25 according to scrum could have eliminated a lot of the issues related to the functioning of the machine.

Team construction

As stated in [4] the skillset of the scrum team should have been chosen with care and should contain at least: Software engineer, Robotics specialist, Usability expert. Medical Radiation specialist, end user.

On the side of the customer there should be, throughout the development process, access to a specialist who can clarify safety requirements and regulations. The need for the expert is additional to the need of access to the end users both operator and medical.

During sprints

The team should consider six key areas during the implementation of users stories, HA, Safety requirements, designing for safety, testing and certification and standards.

At the end of the sprint, an evaluation of the build software architecture should be done with method like: Architecture Tradeoff Analysis Method (ATAM).

Conclusion

I think that safety–critical system software can be build with agile methods like XP or scrum the iterative delivery of functionality allows a thorough evaluation of safety aspects. Special care should be taken in the construction of the building teams and the contact with the customer should be extended with a safety specialist. During the sprints or iteration the team should be constantly aware to design and build for safety [4].

References


A Principles behind the Agile Manifesto

• We follow these principles: Our highest priority is to satisfy the customer through early and continuous delivery of valuable software.
• Welcome changing requirements, even late in development. Agile processes harness change for the customer’s competitive advantage.

• Deliver working software frequently, from a couple of weeks to a couple of months, with a preference to the shorter timescale.

• Business people and developers must work together daily throughout the project.

• Build projects around motivated individuals. Give them the environment and support they need, and trust them to get the job done.

• The most efficient and effective method of conveying information to and within a development team is face-to-face conversation.

• Working software is the primary measure of progress.

• Agile processes promote sustainable development. The sponsors, developers, and users should be able to maintain a constant pace indefinitely.

• Continuous attention to technical excellence and good design enhances agility.

• Simplicity—the art of maximizing the amount of work not done—is essential.

• The best architectures, requirements, and designs emerge from self-organizing teams.

• At regular intervals, the team reflects on how to become more effective, then tunes and adjusts its behavior accordingly.