Scrum with XP

By Kane Mar, Ken Schwaber.

Introduction

Scrum and extreme programming (XP) are both Agile methodologies. We've heard controversy regarding the value of each, with people familiar with each tending to disparage the other. Yet we found these methodologies to be complementary in a recent project at a large energy company where we got to implement them jointly.

Scrum

Scrum is a product development methodology consisting of practices and rules to be used by management, customers, and project management to maximize the productivity and value of a development effort. Scrum takes the responsibility for development projects out of engineering and IT and puts them squarely back in the business. With Scrum, businesses own and manage projects rather than "tossing them over the wall" to IT and hoping for the best. Scrum reintroduces accountability for IT projects to the business, requiring the business to maximize the ROI, without excuses. A business uses Scrum to run development projects in a businesslike manner, paying particular attention to realizing the value of the investments as soon as possible.

Scrum doesn't have any engineering practices, wrapping and using those at the organization where it is implemented. When these engineering practices are weak, overall productivity is lessened.

XP

Extreme programming (XP) is an engineering methodology consisting of practices that ensure top-quality, focused code. XP begins with four values:

- Communication
- Feedback
- Simplicity
- Courage

It then builds up to a dozen practices, weaving them into a synergistic whole in which each one is reinforced by the others and is required for the whole to work. These values and their underlying practices and techniques are not divisible and individually selectable; they form a coherent, whole process. Teams that use XP practices are adhering to strong engineering disciplines. Like guilds, the teams that follow these practices generate good products.

XP doesn't have any management practices. XP tells management where it needs them, but offers few insights into maximizing value.

Scrum and XP

Scrum and extreme programming provide complementary practices and rules. They overlap at the planning game (XP) and Sprit planning (Scrum). Both encourage similar values, minimizing otherwise troublesome disconnects between management and developers. Combined, they provide a structure within which a customer can evolve a software product that best meets his or her needs, and can implement quality functionality incrementally to take advantage of business opportunities. Following are several shared practices that facilitate this functionality:

- **Iterations.** All work is done iteratively, with the customer being able to steer and direct the project every iteration.
- **Increments.** Every iteration produces an increment of the customer's highest-priority functionality. If desired, the customer can direct the developers to turn these increments into live, operational functionality at any time.
- **Emergence.** Only that functionality that the customer has selected for the next iteration is considered and built. The customer doesn't pay for functionality that he or she might not select, and the developers don't have to code, debug, and maintain irrelevant code.
- **Self-organization.** The customer says what he or she wants; development determines how much they can develop during an iteration and figures out the tasks to do so.
- **Collaboration.** Business and engineering collaborate about how best to build the product and what the product should do between iterations.

Introduction to Project X

Project X had a significantly troubled background. The requirements and data were so complicated that prior efforts never got beyond analysis. After two highly visible failures to deliver software, senior management felt a need for a radical change in approach.

The client had previously experimented with Agile methodologies as part of its ongoing research into software development and had had some success. In particular, the client had successfully used XP with engineering and infrastructure-related projects and had experimented with various Scrum practices. Following the most recent failures of Project X, it was felt that the time was ripe to introduce Agile methodologies into a formal business unit.

Ken led a three-day workshop consisting of business users, management, and engineers. Kane took on the role of ScrumMaster. The engineers were already skilled in XP. At the end of the workshop, the team was working on the first Sprint (iteration), using XP and Scrum.

Scrum Process with XP Engineering Practices

The phrase *Scrum process with XP engineering practices* refers to the use of Scrum to manage the steps taken to develop software, in conjunction with the use of XP to ensure the quality of the software.

These are the Scrum processes that were used for Project X:

- **Sprints and Sprint planning meetings.** Sprints are iterations of approximately 30 days. A Sprint is initiated by holding a Sprint planning meeting, at which the goal of the Sprint is defined and the Sprint backlog is started.
- **Daily Scrums.** These are daily meetings in which the project team discusses what work is currently in progress, in addition to any impediments.
- **Product backlog and product owner.** The *product backlog* is the prioritized list of products that is owned by the *product owner*.
- **Sprint backlog.** The *Sprint backlog* is the list of work that's currently assigned to the Sprint, and the amount of work remaining to complete an item of work. It's owned by the project team and is updated on a daily basis.
- **ScrumMaster.** The *ScrumMaster* is the unfortunate individual who has been charged with removing any impediments that the team may have.

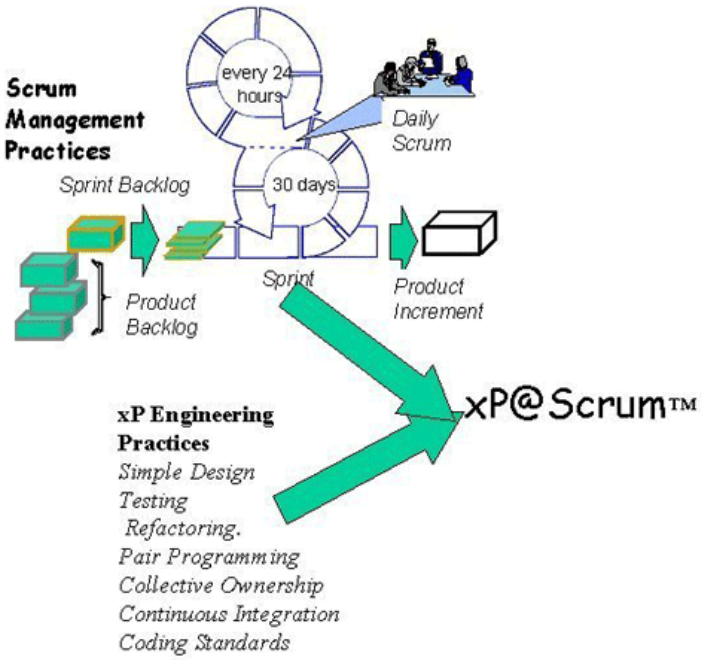

Figure 1 Scrum and extreme programming.

These are the XP engineering practices that were used for Project X:

- **Simple system design.** An emphasis was placed on designing only what was needed to support the functionality being implemented.
- **Test-first coding.** Unit tests were written prior to the construction of code. This practice forces developers to understand the interface and expected functionality of a class. The tests accumulate over the duration of a project, providing a library of regressions tests.
- **Continuous integration.** The Open Source product Cruise Control was installed and configured to run every five minutes. This process would automatically check out all code from the repository, build the code, and run the library of tests. By continuously building and testing the code, the project team could ensure that the base software was stable and of a high quality.
- **Refactoring.** Refactoring allows for the incremental improvement of the design and class structure to support new functionality.
- **Pair programming.** The project team took pair programming to the extreme, doing all work in small groups of either two or three. This included activities such as analyzing data and creating the data model, in addition to programming.

Scrum Requires a Mindset Change by the Client

The initial use of Scrum is intimidating for any member of senior management (project managers and above). It's a significant departure from methodologies that have become the norm throughout the industry, and thus requires a change in mindset for management. Most management is used to directing the project, telling the team what to do and then ensuring that they do it. Scrum and XP rely on self-organization, with the team deciding what to do while management runs interference and removes roadblocks.

We've encountered this mindset change before, but it was particularly noticeable in this project. The project manager was consumed with trying to figure out who would be doing what. She also was occupied by trying to get the development environment into place. The team sat passively, waiting for their assignments. During the workshop we had the customer and team determine what to tackle for the first Sprint. Then we asked the team to figure out what it had to do to build this functionality. We then stood back. The project manager initially tried to hand out assignments and defer some work that she hadn't done yet, but we asked

her to stand aside—this was the team's job. The team suddenly started talking—brainstorming, plotting, scheming, planning how to get this code built. They explored various alternatives, got to know each other, and devised their work plan. The team experienced an epiphany, as they realized they were free to proceed however they chose. The manager experienced an epiphany, as she realized that she didn't have to tell the team what to do (and ensure that it was done). Her new job was helping the team, expediting their work and removing any impediments. They were all cooperating and optimizing each other.

We were lucky in that we had full support from senior management, allowing the project to move ahead very quickly with little impedance from other parts of the organization. A great deal has already been written on this topic, and the reader is referred to our book, Agile Software Development with Scrum (Prentice Hall, 2001, ISBN 0-13-067634-9), for a more in-depth discussion.

Increased Visibility When Using Scrum

The use of Scrum on Project X has greatly increased the visibility of the project with both the client and senior management. Initially, there was some skepticism as to whether the team would be able to deliver *anything* within a single Sprint. This was due in part to the history of the project and the implementation of new development methodologies.

Any concerns about the use of XP and Scrum were quickly dissipated when the project team was able to demonstrate working software that addressed both interface and data quality problems during the first Sprint review. The functionality wasn't complete, but it showed visible progress and tangible benefit. The demonstration of working software allowed interested parties to have a concrete understanding of the functionality and quality of the software that had been developed. In addition, it communicated the team's progress much more effectively than written status reports.

At the Sprint review, IT and business management were able to see the team and business owner collaborating about what had been done and what to do next. As the senior IT person said, "Agile (Scrum and XP) solves my problem of customer involvement." Management queried the team, "How have you been able to make so much progress?" The team attributed much of the progress to close communication and collocation of business users and the team, optimizing communication and productivity.

Interested parties were also invited to communicate directly with the team. This allowed both parties to communicate freely and to explore some of the issues that were resolved by the project team during the Sprint.

XP and Software Quality

The use of XP engineering practices within the project was found to cause a marked improvement in the quality of the code, in comparison to previous projects. Although each of the practices addresses different aspects of code quality, they were found to have a profound overall effect over a period of time. Part of the reason is the incremental fashion in which they were applied. By gradually improving the existing code base while working toward the final solution, the code base gradually converges on an elegant and workable solution.

In summary, the application of XP engineering practices outlined above directly resulted in an increase in the quality of the completed code and test suites.

Scrum and Productivity

The application of Scrum was found to increase the productivity of the project team. There may be several explanations, but we feel that the single most important reason is that Scrum allows developers to focus on delivering usable functionality to the client, rather than delivering artifacts of questionable value. Analysis and design deliverables were produced when necessary during the project—when it was felt that they added value. However, when there was no need to produce written deliverables, the project team comfortably ignored them. Because both Scrum and XP rely on emergence, the combination allowed the business to understand how to leverage engineering resources.

It's important to note that this mode of operation is only sustainable when the project team is small. In the case of Project X, the project team consisted of only 14 individuals, with varying degrees of experience and expertise. In larger projects, some initial system and product architecture artifacts would probably be required to help coordinate the teams. However, we successfully delivered more functionality within the first few Sprints than the previous team did in nine months.

Scrum also insists that the client prioritize the required functionality (that is, products) in the product backlog. This practice allows the project team to deliver the functionality that the client wants the most. A beneficial side effect is that the client is genuinely excited by any functionality demonstrated at the Sprint reviews. This excitement is often transferred to the project team, boosting team morale.

Conclusions

The use of Scrum with XP engineering practices was found to have a significant impact on the productivity of the project team. We believe that the adoption of either of these methodologies will have a beneficial impact on a project team. We believe that the joint application of these methodologies instills practices and values in management, customers, and development that allow them to create a unified, disciplined team. In our experience, these two practices are complementary; when used together, they can have a significant impact on both the productivity of a team and the quality of its outputs.

References

- Ken Schwaber and Mike Beedle, Agile Software Development with Scrum (Prentice Hall, 2001, ISBN 0-13-067634-9).

- Kane Mar has been developing software and managing projects for the past 13 years. He is experienced in the use of traditional software development methodologies and has spent the last two years investigating and using Agile methodologies. Kane is currently a consultant with ThoughtWorks, Inc. He can be reached via email at ScrumMaster@ThoughtWorks.com.