Beyond the Comfort Zone of Scrum

Vladimir Blagojević, Wim Codenie, Jeroen Deleu, Olivier Biot
Software Engineering Group, Sirris, Belgium
{vladimir.blagojevic,wim.codenie,jeroen.deleu,olivier.biot}@sirris.be

Abstract

Scrum is a successful agile management method for incremental software development whose success has raised an interest outside the software community. Can Scrum principles be applied to develop other types of products than software products? In this paper, several such alternative applications of Scrum are studied. The paper identifies four different models for applying Scrum based on two parameters: team's shared commitment and resolving ability.

1. Introduction

Scrum is an agile management mechanism for building software products in an incremental way. In short, Scrum divides a project in several iterations (called *Sprints*) of approximately 2-4 weeks. The *Sprint goal* is typically to develop a (potentially shippable) *product increment* [16], but it may also be something else that adds significant business value [9] or significantly reduces risk (e.g. a trade fair demonstrator, a technology feasibility study…). A Sprint starts with a *Sprint planning meeting* where a *Sprint backlog* is created. The backlog contains *Sprint backlog items* describing all the work required to achieve the Sprint goal [16]. During the Sprint the team meets every day on a *daily Scrum meeting* to answer three questions: "What did I do during the last 24 hours?", "What are my obstacles?" and "What will I do the next 24 hours?". Scrum identifies three roles. The *Scrum Master* is responsible for removing obstacles and for creating the Scrum environment. The *Product Owner* is responsible for the Product Backlog which contains all relevant requirements for the product. The *Scrum Team* is a self-managing, self-organizing and cross-functional team responsible for developing a product increment [16].

Scrum has grown into a proven method to deliver value in software development [1] [14]. This success has raised an interest in Scrum outside the software community. Organizations want to copy this success and are exploring the boundaries of how far Scrum can be stretched. Sirris[1] conducted a study[2] to better understand this phenomenon. The goal of this study was to evaluate how Scrum can be applied to development of other kinds of products than "software products". The preliminary results of this study are presented in this paper.

2. Leaving the Comfort Zone of Scrum

In this paper, leaving the comfort zone of Scrum will be interpreted as application of Scrum to the development of other kinds of products than software products. This paper will consider application of Scrum for the following alternative product developments[3]:

Development of Software-Intensive Products. Software-intensive products are a mix of software and other technologies. There is a high interest in applying Scrum to the development of this type of products [1] [15] because in many sectors software is becoming a major driver for innovation [8]. To study the application of Scrum to development of software-intensive products, three cases were identified in the domain of hardware/software co-development.

Development of Services. Offering services to customers is a strategy adhered to by an increasing number of companies [10]. To study the application of Scrum to development of services, two cases were identified in which a team was given the assignment to define a set of advisory services.

[1] Sirris is the collective centre of the Belgian technology industry. Sirris advises and supports companies on the implementation of technological innovations, enabling them to strengthen their competitive position over the long-term.

[2] The study was made with the support of ISRIB (Institute for the encouragement of Scientific Research and Innovation of Brussels), IWT (Institute for the Promotion of Innovation by Science and Technology in Flanders) and the Brussels-Capital Region.

[3] Development of material, non-software products is intentionally not considered due to the abundant state of the art available in lean product development and manufacturing [11].

Development of Knowledge. Importance of innovation for companies [19] and the current knowledge economy make development of knowledge increasingly important. To study the application of Scrum in this context, two cases were selected where Scrum was applied in management of scientific research projects, as the primary goal of these projects is developing new knowledge.

Development of a Portfolio of Products. Product segmentation is a basic strategy of a majority of companies [12]. Managing and developing a product portfolio (e.g. a set of different products and/or services [5]) is therefore an important activity for them. To study the application of Scrum to product portfolio development, two cases were identified where Scrum was applied to program management: the management of multiple interdependent projects and potentially other programs[4].

The focus was to study single teams while developing these alternative kinds of products. Inter-team collaborations, such as Scrum of Scrums [17] are beyond the scope of this study. In order to draw conclusions solely on the basis of leaving the comfort zone of Scrum, an important criterion for selection of cases was to keep the environment as ideal as possible for the introduction of Scrum (i.e. small collocated teams) [14] [13] [6].

3. Shared Commitment & Resolving Ability

In some cases Scrum worked well, whereas in others the Scrum experiment did not succeed. It was not possible to relate success to one of the product types, as within most types both successes and failures were reported. Additional analysis was needed to reveal the underlying pattern of success and failure.

One characteristic of situations that faced more difficulties in Scrum introduction was the presence of different professions in the team (e.g. hardware mixed with software, software mixed with marketing). Indeed, there are several studies available that describe the difficulties of multi-disciplinary product development [3]. However at a first glance this seemed strange – after all, isn't a Scrum team supposed to be cross functional? Indeed, many experience reports describe success stories of applying Scrum in a team where people of different professions work together. Typically, these are business analysts [20] (a.k.a. on-site customers [2]), developers and testers.

So what is the difference between a traditional cross-functional team and the studied multi-disciplinary teams? To understand this we had to analyze their dynamics at the level of Sprints. A traditional cross-functional Scrum team works together to develop a software increment. Each team member uses specific skills to contribute to the increment. A cross-functional Scrum team has a *high level of shared commitment to the Sprint goals*.

In this context, building increments is feasible because the deliverable is software. For non-software deliverables this is not always possible. In the case of software/hardware co-development for example, software and hardware developers work more independently from each other. Integration typically happens only at a late stage in the project. Building hardware feature by feature makes little sense. Hardware designers spend much more time thinking upfront – because the price paid for design mistakes is higher than in the case of software development. This all means that during many Sprints, the focus is mainly on discipline-related goals: software engineers write software while hardware engineers design hardware. The engineers in fact commit to different goals during this period, or in other words, the *level of shared commitment to Sprint goals* is relatively low.

However, some of the studied teams had a low level of shared commitment, and still reported many benefits of using Scrum. They observed that Scrum offered them a "formal" instrument to help each other and to overcome obstacles. Teams reporting these benefits typically have a lot of common ground and the ability to help each other, or in other words – a *high level of resolving ability*. This is indeed a key difference compared to teams consisting of software and hardware engineers: since engineers and designers are educated and trained discipline-wise, they typically have insufficient knowledge about other disciplines. In other words, their *resolving ability is low* (with respect to the other discipline).

In conclusion, success of Scrum does not seem to depend on the type of product developed, but rather on the following two key parameters.

The **level of a team's shared commitment** is a parameter that captures the level of shared commitment to Sprint goals in a team. It is defined as the number of team members committed to the Sprint's goals, averaged over all Sprint goals and divided by the size of the team:

[4] Using Scrum to manage multiple interdependent projects and programs will naturally result in defining multiple Sprint goals. The term "Sprint goals" will therefore be used in the remainder of the paper to refer to what is generally known as the Sprint goal.

$$\frac{1}{m \times n} \sum_{i=1}^{n} commitment(g_i)$$

In this formula $g_1, g_2 \ldots, g_n$ are the n Sprint goals, *commitment*(g_i) is a function that returns the number of team members committed to a given goal g_i, and m represents the team size (number of people in the team)[5]. This formula yields a value between 0 (no members committed to any of the Sprint goals) and 1 (all members committed to all Sprint goals). This definition allows parameter values to be compared between different teams[6].

The **level of a team's resolving ability** is a parameter that captures the team members' ability to help each other. The level of a team's resolving ability is defined as the number of team members able to resolve obstacles of the Sprint backlog items, averaged over all Sprint backlog items and divided by the size of the team:

$$\frac{1}{m \times p} \sum_{j=1}^{p} resolvers(b_j)$$

In this formula $b_1, b_2 \ldots, b_p$ are the p Sprint backlog items, *resolvers*(b_j) is a function that returns the number of team members able to resolve obstacles of a given backlog item b_j, and m represents the team size (number of people in the team). This formula yields a value between 0 (no members able to resolve obstacles for any of the Sprint backlog items) and 1 (all members able to resolve obstacles for any of the Sprint backlog items). This definition allows parameter values to be compared between different teams.

4. Four Scrum Application Models

During the study, different variants of Scrum emerged in four distinct application models. The two key parameters seem to play an important role in understanding the success of Scrum in each model.

Homogenous Model: High level of shared commitment and high resolving ability. Team members have a lot of common ground and shared commitment to Sprint goals. The team's truck number[7] is often high, as is the level of shared responsibility and success. A typical example is a software development team where all team members are software engineers with a lot of common ground [9].

Coordination Model: High level of shared commitment but low resolving ability. Team members may play a different role in the team and even have different professions, but they share commitment to Sprint goals. A typical example is a cross-functional team with a business analyst, a developer and a tester.

Co-opetion[8] Model: Low level of shared commitment but high resolving ability. Team members share a lot of common ground, and are able to help each other. This provides a basis for cooperation. At the same time many "individual" Sprint goals (or goals relevant for only a small subset of the team) are present; sometimes creating competition between goals. A typical example is a team using Scrum for program management (management of multiple interdependent projects and programs). An important characteristic of this application is that programs can be long running and without a real end.

Dependency Model: Low level of shared commitment and low resolving ability. There is little sharing of commitment and the resolving ability is low, but critical interdependency between goals exists. An example is a hardware/software co-development team during the detailed design phase[9]. Software and hardware engineers work on their own "islands", each developing their own subsystems, and they can rarely help each other. At the same time, a lot of interdependencies are present, because in the end product, software and hardware need to work seamlessly together.

The study also revealed that in the Homogenous and in the Coordination Model, Scrum can be applied without major modifications to the process. In the Co-opetion Model Scrum will most likely work well, but a number of modifications to the Scrum process can be expected. In the Dependency Model many difficulties may be expected. With respect to success of application of Scrum, these models can be categorized in three areas as depicted on figure 1.

[5] For a team size m, a Sprint goal can have between 0 and m people committed to it. Having a value 0 for a given goal indicates a problem in the Sprint (there is no commitment to this goal).

[6] An interesting observation can be made about the minimum level of shared commitment in Sprints where all goals have at least one committed team member. A 2-person team has a minimum of 0.5, and for a 5-person team this is 0.2. This reflects the fact that larger teams with low shared commitment will face more difficulties.

[7] Truck number is often defined as "the number of people on a team who have to be hit with a truck before the project is in serious trouble".

[8] Co-opetion is a combination of cooperation and competition [7].

[9] This may not be true during a different phase of co-development, where there is much more shared commitment (e.g. during the concept definition phase).

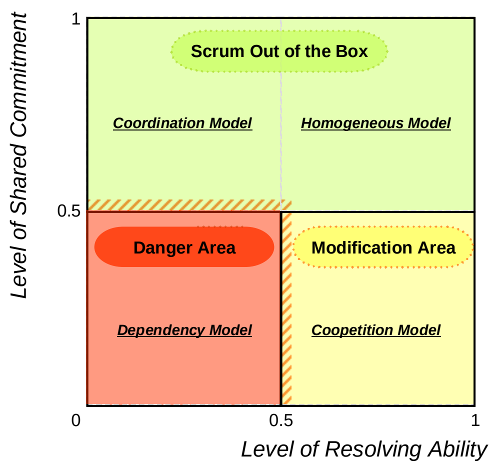

Figure 1. The four Scrum application models

5. Conclusions and Future Work

In this research, the applicability of Scrum was evaluated in the context of hardware/software co-development, development of services, program management and scientific research. Two parameters are important to understand these expectations: the team's resolving ability and their level of shared commitment to Sprint goals. Combinations of these parameters define four Scrum application models with fundamentally different Scrum dynamics.

A first goal of this research is to define the Agile Probe: a method for better understanding what to expect when leaving the comfort zone of Scrum, and where to adapt the Scrum model toward the specific application model.

Some interesting topics for further research are:

- How can Scrum contribute to the reduction of failures caused by interdisciplinary dependencies in hardware and software [21]?
- To what extent can Scrum be used as an instrument to enable open innovation [4]?

6. References

[1] Agile Software Development of Embedded Systems, ITEA project, http://www.agile-itea.org/

[2] Beck, K., Andres, C. (2004). Extreme Programming Explained: Embrace Change. Addison-Wesley Professional; 2nd edition (November 26, 2004). ISBN 978-0321278654.

[3] Bernstein, J. I. (2001). *"Multidisciplinary Design Problem Solving on Product Development Teams"*, 2001, Lean Advancement Initiative, MIT (http://lean.mit.edu/)

[4] Chesbrough, H. W. (2003). Open innovation: The New Imperative for Creating and Profiting from Technology. Harvard Business School Press, ISBN: 978-1578518371.

[5] Cusumano, M., The Business of Software: What Every Manager, Programmer, and Entrepreneur Must Know to Thrive and Survive in Good Times and Bad, Free Press, 2004. ISBN: 978-0743215800

[6] Flexi – *"Flexible Integration in Global Product Development"*, ITEA 2 Project, http://www.flexi-itea2.org/index.php

[7] IBM, "Strategic co-opetition: The value of relationships in the networked economy", http://www-935.ibm.com/services/uk/index.wss/multipage/igs/ibvstudy/a1008082?cntxt=a1006870

[8] ITEA Blue Book: the case for ITEA 2, Sept. 2005.

[9] Kniberg, H. (2007). Scrum and XP from the Trenches. C4Media, ISBN 978-1-4303-2264-1.

[10] OECD in Figures 2006-2007. Services as a share of Gross Domestic Product. http://www.oecd.org/

[11] Poppendieck, M., Poppendieck, T. (2003). Lean Software Development. Addison-Wesley, ISBN 0-321-15078-3.

[12] Porter, M. E., (1998). Competitive Advantage: Creating and Sustaining Superior Performance. Free Press; 1st edition (Jun 1 1998). ISBN 978-0684841465.

[13] Ramesh, B., Cao, L., Mohan, K., and Xu, P. 2006. Can distributed software development be agile? Commun. ACM 49, 10 (Oct. 2006), 41-46.

[14] Rising, L., Janoff, N.S. (2000). The Scrum Software Development Process for Small Teams. IEEE Software 17(4): 26–32.

[15] Ronkainen, J., Abrahamsson, P. (2003). Software Development under Stringent Hardware Constraints: Do Agile Methods Have a Chance? XP 2003: 73-79

[16] Schwaber, K. (2004). Agile Project Management with Scrum. Microsoft Press, ISBN 0-7356-1993-X (paperback).

[17] Schwaber, K. (2007). The Enterprise and Scrum. Microsoft Press, ISBN 978-0735623378.

[18] Sidky, A. (2007). Assessing Readiness for Agile Adoption using a Practical and Innovative Approach. Agile 2007 conference. Available from http://www.agile2007.org/

[19] Siemens' R&D: Tuned in to Today's Megatrends. In Siemens Pictures of the Future – Fall 2005.

[20] Suscheck, C. and York, M. (2007). Agile Development Teams Need Business Analysts. In Agile Journal: http://www.agilejournal.com/articles/articles/agile-development-teams-need-business-analysts.html

[21] TWINS, ITEA project: http://www.twins-itea.org/