SCRUM

Darrell Hareford, Sarah Paulsen, Jonathan Brown

A brief description of Scrum as an Agile development method.
A comparison of Scrum with traditional software development.
Advantages/disadvantages of a Scrum approach to software development.

CIS 3300 – Systems Analysis

Georgia State University

5/2/2014

What is Scrum? This is a question that many students ask when studying software development. In this paper, the agile software development method Scrum will be introduced and will be compared to traditional software development methods. The key roles in the Scrum process will be identified and the Scrum process itself will be described. The paper will end with a description of the advantages and disadvantages of Scrum. In order to understand Scrum though, it is necessary to know a little about software development.

There are two competing schools of thought on software development. The traditional form focuses on process oriented development. One such approach is characterized by the classic waterfall scheme of development whereby software development proceeds in sequence from one phase to the next.[1] The key deliverables for each phase are typically very long (often hundreds of pages in length) and are presented to the project sponsor for approval as the project moves from phase to phase.[2] Once the sponsor approves the work conducted for a phase, the phase ends and the next one begins.[3] There are process oriented development methods outside of the waterfall structure, but they maintain many characteristics of waterfall. Agile development is an often cited alternative to the process oriented development methods, traditionally presented as the classic waterfall. "The strong points of such methods [agile development] are that by employing them, the development process becomes more responsive to a changing environment, working software is chosen over extensive documentation, individuals and interactions are considered more important than tools and processes, and customer collaboration is valued more than contract negotiation."[4] "The emphasis of the agile development method is to focus the developers on the working conditions of the developers, the working software, the customers, and addressing changing requirements instead of focusing on detailed systems development processes, tools, all inclusive documentation, legal contracts, and detailed plans. These programming – centric methodologies have few rules and practices all of which are fairly easy-to-follow."[5]

Scrum is one development method within the family of agile development methodologies.[6] Agile software development methodologies focus upon incremental design unlike the sequential design process associated with process oriented development. So why Scrum? Why do some software development firms prefer agile development to classic process oriented software development methods and prefer Scrum over the many agile methods? Scrum uses incremental development procedures with a minimum emphasis on upfront design.[7] The objective is to get working software into the hands of the stakeholders as quickly as possible. This puts business value functions into stakeholder possession early on in the software development life cycle.[8] This differs from process oriented development in that stakeholders generally would not have access to any software produced until far later in the process. Scrum in

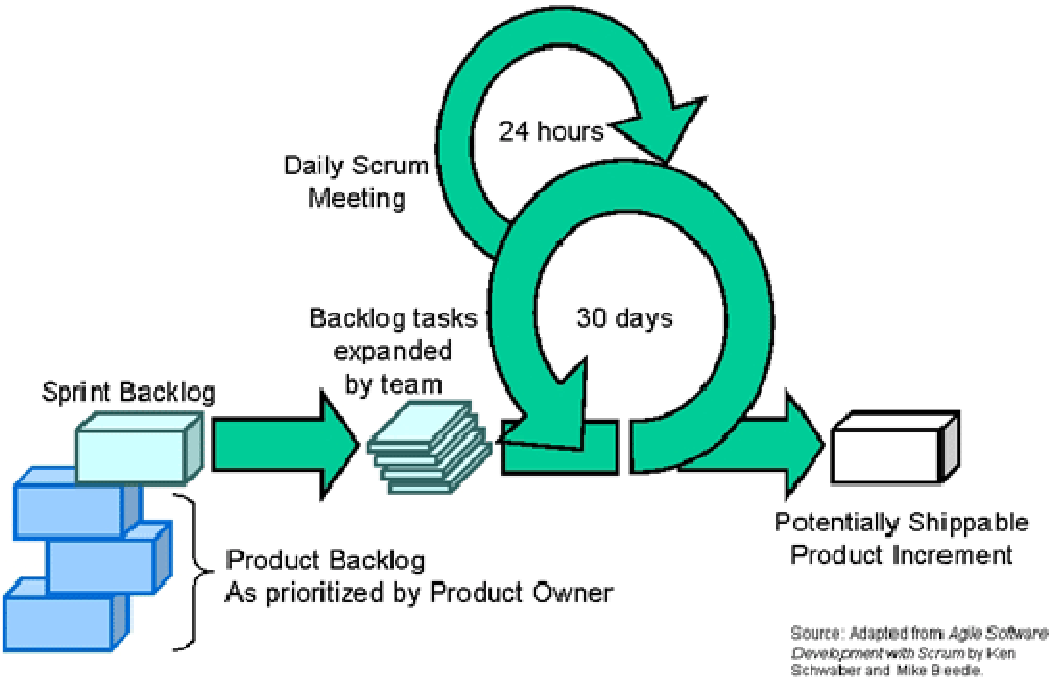
[1] (Barlow, et al. 2011)
[2] (Barlow, et al. 2011)
[3] (Dennis, Wixom and Tegarden 2012) 8
[4] (Vlaanderen, et al. 2011)
[5] (Dennis, Wixom and Tegarden 2012) 14
[6] (Ionel 2008) 6
[7] (Dathan and Menon 2010) 1
[8] (Dathan and Menon 2010) 19

concept is straightforward in its approach to software development. The simplicity, and the ability to scale Scrum to some degree support its choice as a leading development method within the agile family of software development processes.

Scrum starts with the business case, or reason to justify the time and expense of developing a software project, just as one would do with process oriented development. From that point it diverges from the traditional development methods. Stakeholders provide user stories, or short descriptions of project requirements, that describe what they want in the software package.[9] A "product owner" selects the user stories for cases that will provide optimal business value and be used to create the software. Un-desired, un-needed user story requirements, or those outside of the scope and business vision (as determined by the product owner), are eliminated. The user stories remaining then become a product backlog which the product owner prioritizes in order of providing optimal business value.[10] Then the product owner selects a subset of the product backlog for development with business value priority. This is referred to as a sprint backlog.[11] A Scrum meeting is held with the entire team where the plans for the upcoming sprint, and time parameters for development are discussed. Each user story is assigned an expected time period for development.[12] Identification and communication of methods to complete the work needed for sprint completion are finalized. All sprint backlog user story development times are then summed to produce a burn down chart. The burn down chart represents the expected time to develop, test, and document this sprint backlog for a deliverable operating software package, albeit with limited functionality at this point.[13] The objective is to produce operating, tested, and documented software that is a functioning product that could be shipped as a finished product.[14] Sprints may last from one week to a month. Time for the sprint is a function of the number of user stories of the sprint, complexity of software development, skill sets of the developers, and management's (the ScrumMaster's) ability to manage problems, protect/shepherd the developers, and maintain project focus.[15] The number of user stories selected for the sprint backlog are a function of maintaining control of a short development cycle, and providing a useable subset of the desired software package. The burn down chart, which is updated daily, provides a visual representation of the success of the project in terms of meeting the budgeted time objectives. Individual team members pull user stories of their choice from the sprint backlog to complete until all sprint backlog items are completed. At the end of the sprint a product demonstration is held for the stakeholders and a brief meeting is held to discuss what went right and why, what went wrong and why, and how to improve on the next iteration.[16] Then the process is repeated; the user stories are gathered and reprioritized for the next implementation of a new sprint backlog. A visual representation of the sprint process follows.

---

[9] (Boschetti, et al. 2014)
[10] (Dathan and Menon 2010) 19
[11] (Vlaanderen, et al. 2011)
[12] (Boschetti, et al. 2014)
[13] (Vlaanderen, et al. 2011)
[14] (Vlaanderen, et al. 2011)
[15] (Ionel 2008) 2
[16] (Ionel 2008) 4



The primary players in Scrum are the product owners, the ScrumMaster, and the development team.[17] The product owners are just that; they own the developed software project. They accept the deliverables, the final product, or call off or cut the project down if time, budget, or priorities change. The ScrumMaster assumes the position generally referred to as a project manager, with the caveat that the ScrumMaster is a knowledgeable practitioner of the Scrum process.[18] The project team is a self-organizing, cross functional group of developers and testers.[19]

Tools initially used for running a Scrum operation involved a white board with sticky notes for user stories or a corkboard with pins to attach user stories to a section of board (often called the Kanban board).[20] Developers would remove a user story note or move it into a work in progress section of the board, and complete the task to satisfy the user story requirement. The note would then be shifted to some other section of the board as completed.[21] MS Excel or the equivalent would be used to produce a burn down graphical chart and maintain it. Updates were performed daily on the board(s).[22] These methods are still employed, but the popularity of agile development has led to the creation and deployment of software packages which assist in the management of Scrum development activities and projects.[23] They are generally designed to create and estimate user stories, build sprint backlogs, identify team commitment and velocity,

[17] (Dathan and Menon 2010) 18
[18] (Dathan and Menon 2010) 18
[19] (Noordeloos, Manteli and Van Vliet 2012)
[20] (Gannon 2013)
[21] (Gannon 2013)
[22] (Gannon 2013)
[23] (Frost 2007)

visualize team activity, and report on team progress. Packages for Scrum development include Version One, Greenhopper, Microsoft TFS, Rally, and IBM Rational Team Concert & Jazz. There are also a number of training programs and certificates offered in Scrum and other agile development methods.[24]

Where does it make sense to use Scrum? Where do agile development methods and Scrum in particular shine? Within the Scrum framework, customer collaboration and development team co-location is highly encouraged. "Software development is unique due to its potential for rapid feedback loops. … Unlike other human efforts, software has been one the fastest models for understanding intense human collaborative efforts. Most of the efforts in software development sit on the boundary between… complex… [and] chaotic. Software development has enabled a rapid viral model for understanding how people work together to build complex things. The Scrum framework has demonstrated broad applicability to any endeavor of work that lives within the complex to chaotic boundary for a team's work. … Scrum is a project or product development method that contains short rapid cycles for evolving products."[25] Therein lies the strength that can be ascribed to Scrum vs. other traditional development methods. Much of the literature espouses Scrum's advantages for software development in volatile development environments where speed, rapidly changing technology, changing requirements, and competitive demands require rapid product evolution.[26] Scrum's rapid development cycle with the key ingredients of a working software package, tested and documented at the end of each sprint, lends itself to this type of environment. Additional Scrum iterations add features, functionalities, requirements changes, and flesh out a more polished product.[27] The up-front investment in detailed requirements gathering, analysis, and design common to traditional process oriented development are essentially non-existent in Scrum, which allows for the rapid pace of product development.[28] There are downsides to this approach, however.

Scrum allows for some scalability, by increasing the number of development teams, with separate ScrumMasters who work on separate subsections of a product (which must be integrated together at some point), but overall does not scale well to large projects.[29] The key to Scrum development when scaled is to be able to break the project down into logical, reasonably sized deliverables. Many large projects may not easily break down into the deliverable sizes that work within the Scrum framework. The final product developed through Scrum methods may not match well with the original concept since the requirements for that concept were without the specific definition that would have occurred in a procedural development environment.[30] That may be a good, or bad thing. Multiple iterations may have carried the project/product beyond the

[24] (Agile Delivering Value 2013)
[25] (Why Choose Scrum? n.d.)
[26] (Vlaanderen, et al. 2011)
[27] (Barlow, et al. 2011)
[28] (Barlow, et al. 2011)
[29] (Ionel 2008)
[30] (Barlow, et al. 2011)

scope of the original concept, allowing for scope creep.[31] The product owner must maintain control of defining what should, or should not be included.

Scrum and agile development methods in general are best suited for non-mission critical tasks. Procedural programming establishes deliverables in requirements, analysis, risk management, and design that make considerations for the demands of mission critical work. Scrum does not have these requirement deliverables.[32] If your social network link crashes interminably it probably won't result in anything more than localized (your) mayhem. If a train signal controlled by software changes from red to green, when it should be red, or your software controlled rocket thruster doesn't fire for docking when it is necessary, or the software controlled neutron absorption rods for the nuclear power plant are not inserted when (and to the depth) required, serious issues may arise (mayhem). Mission critical software requires a depth of investigation that agile just is not equipped or designed to provide. Procedural development requires many eyes, minds, and disciplines to investigate and sign-off on a project/product before development begins.[33] While it may not, in every instance, catch an errant divergence from the intended goal, it has procedures in place to reduce its likelihood.

The size and scope of some projects do not lend themselves to Scrum or any other agile development method.[34] The recent introduction of a more nationalized health care system within the US would serve as an example. The actual requirements go far beyond that of a web site which has been the forefront of public debate. The full implementation of such a system requires extensive work in developing the requirements, performing the analysis, analyzing the risk, and creating a design in order to achieve a software system which integrates across a myriad of healthcare providers, insurance providers, government agencies, and the general public. That has to be performed before a line of code is written. Scrum would not be up to the task. Now that does not preclude some aspects of well-defined subsystems from being developed using Scrum development methods. Scrum and agile simply do not have the framework to take massive development projects to task.

It is clear that Scrum has its place within the various categories of software development environments. Non mission-critical and small-to-intermediate software projects which can be broken down into discrete, easily-managed requirement subsets that can be completed within short time spans, lend themselves to software development by small teams of developers using Scrum. Scrum offers a simple straightforward development method which has demonstrated success in a wide variety of software development projects.[35] Many companies have adopted Scrum as a development platform to provide rapid development of software to meet the needs of business, industry, and consumer goods.[36] Large companies such as Microsoft and IBM perform some development tasks using Scrum, and both of these companies have produced commercial

[31] (Noordeloos, Manteli and Van Vliet 2012)
[32] (Barlow, et al. 2011)
[33] (Stoica, Mircea and Ghilic-Micu 2013)
[34] (Barlow, et al. 2011)
[35] (Dathan and Menon 2010) 19
[36] (Ionel 2008) 1

tools to manage Scrum development.[37] Scrum is designed to focus on the needs of the stakeholders through the constant involvement of the product owner throughout the development process, maintaining project focus through short daily Scrum meetings, and delivery of a viable, tested, documented product through the collaborative efforts of the developers under the management of the ScrumMaster.[38]

Scrum, like its namesake heritage from the game of Rugby is designed to get the ball over the goalpost through a collaborative effort of all team members working as an integrated unit.[39] Developmental costs tend to be more difficult to nail down since the requirements often are not well defined. Procedural programming methods tend to hand off a task to another developmental increment only after that task has been completed in full, much like a relay race passing a baton. Far more requirements documentation, design, and analysis is performed in procedural programming.[40] Costs tend to be far higher on the front end of procedural programming methodologies. A great deal of work is performed before code is ever written. It helps to insure that mission critical projects will in fact fulfill that mission. Procedural programming supports projects which are too large for the Scrum framework to engage. The downside to procedural programming is that it is quite rigid.[41] Changes in requirements late in the development process can have a dramatic impact on project completion schedules and budgets.[42] Testing is often not performed until much later in the project. Issues which arise during testing may be difficult to repair, or fulfill within budget. Here again we see an advantage to the Scrum software development method. Because Scrum is an iterative process with close collaboration with the product owner, changes in requirements are easier and less expensive to implement.[43]

There are clear advantages and disadvantages to Scrum and other agile software development methods. While the traditional approaches to software development are still going strong, and will continue to be used in mission-critical and very large software projects, it is no longer the only way to manage IT projects. As the industry moves forward, and more companies adopt agile development methods, it will be increasingly important for students and developers to learn and understand flexible, iterative processes like Scrum.

[37] (Wilson, Brown and Burke 2013)
[38] (Dathan and Menon 2010) 2
[39] (Dathan and Menon 2010) 2
[40] (Stoica, Mircea and Ghilic-Micu 2013)
[41] (Stoica, Mircea and Ghilic-Micu 2013)
[42] (Barlow, et al. 2011)
[43] (Dathan and Menon 2010)

Bibliography

2013. "Agile Delivering Value." *ITNow* 18-19.

Barlow, Jordan B., Justin Scott Giboney, Mark Jeffrey Keith, and David W. Wilson. 2011. "Overview and Guidance on Agile Development in Large Organizations." *Communications of the Association for Information Systems* 25-44.

Boschetti, Marco A., Matteo Golfarelli, Stefano Rizzi, and Elisa Turricchia. 2014. "A Lagrangian Heuristic for Sprint Planning in Agile Software Development." *Computers & Operations Research* 116-128.

Dathan, Rajshankar, and Roopa Menon. 2010. "Managing Testing Projects - the SCRUM way." *Journal of the Quality Assurance Institute* 1-3.

Dennis, Wixom, and Tegarden. 2012. *Systems Analysis & Design 4th Ed.* Hoboken, NJ: John Wiley & Sons, Inc.

Frost, R. 2007. "Jazz and the Eclipse Way of Collaboration." *Software.*

Gannon, Michele. 2013. "An Agile Implementation of Scrum." *Aerospace Conference .*

Ionel, Naftanaila. 2008. "Critical Analysis of the Scrum Project Management Methodology." *Annals of the University of Oradea, Economic Science Series* 1-8.

Noordeloos, R., C. Manteli, and H. Van Vliet. 2012. "From RUP to Scrum in Global Software Development: A Case Study." *Global Software Engineering* 31-40.

Stoica, Marian, Marinela Mircea, and Bogdan Ghilic-Micu. 2013. "Software Development: Agile vs. Traditional." *Informatica Economica.*

Vlaanderen, Kevin, Slinger Jansen, Sjaak Brinkkemper, and Erik Jaspers. 2011. "The Agile Requirements Refinery: Applying SCRUM Principles to Software Product Management." *Information and Software Technology* 58-70.

n.d. *Why Choose Scrum?* Accessed April 28, 2014. http://3back.com/scrum/.

Wilson, Daniel G., Joshua Brown, and Adam A. Burke. 2013. "LET'S Scrum! LEARNING DIGITAL MEDIA COLLABORATIVELY." *Technology & Engineering Teacher.*