AGILE Software Development Using Scrum Methodology

Debaswapna Mishra, Narendra.Kumar Kamila

Abstract- This paper examines software developers with an understanding of the SCRUM development practices and methods in the increasingly complex and customer demanding faster time to market scenarios. Scrum has risen from being a method used by a number of enthusiasts at the Easel Corporation in 1993, to one of the world's most popular and well-known frameworks for development of software. The continued expansion of the global rollout of Scrum is testimony to the fact that Scrum delivers on its promise. While it is often said that Scrum is not a silver bullet, Scrum can be like a heat-seeking missile when pointed in the right direction. It's inspect and adapt approach to continuous quality improvement can do serious damage to outmoded business practices. By focusing on building communities of stakeholders, encouraging a better life for developers, and delivering extreme business value to customers Scrum can release creativity and team spirit in practitioners and make the world a better place to live and work. Scrum has emerged from a rough structure for iterative, incremental development to a refined, well-structured, straightforward framework for complex product development. We see from a deep research that it leads folks to adjust, test, and adjust it again until it is solid. This framework is fully defined in last decade and is of immense value to organizational success. This study is aimed at bringing a through and detailed understanding of SCRUM, its advantages over other development methodologies and the execution from a practitioner's standpoint.

The study is presented in two main stages. The first stage describes the evolution of software development techniques and how they have been applied throughout the software development lifecycle, progressing up to Agile Software Development practices and SCRUM based Development. The second stage of the study presents a detailed set of tools, artifacts and practices used in real world scenarios.

Key words: Agile, Scrum, Product development

1. INTRODUCTION

The main concern of this study is on showing how software development methodologies have undergone a sea change with the introduction of various agile methodologies and we will look at Scrum methodology in particular as various companies have adopted it as a means to improve their go-to-market.It is extremely important to understand some of the basic nature of a project. Projects may have a relatively clear mission, but the specific requirements can be volatile and evolving as customers and development teams alike explore the unknown. (Jim, 2003). While interest in agile methodologies has blossomed in the past two years, its roots go back more than a decade. Teams using early versions of Scrum, Dynamic Systems Development Methodology (DSDM), and adaptive software development (ASD) were delivering successful projects in the early- to mid-1990s.

So let's understand - "What constitutes agile software development?" All problems are different and require different strategies.

Manuscript received December, 2013.

Debaswapna Mishra, O.U.A.T Bhubaneswar, Odisha, India.
Narendra.Kumar Kamila, CVRCE, Bhubaneswar, Odisha, India.

While battlefield commanders plan extensively, they realize that plans are just a beginning; probing enemy defenses (creating change) and responding to enemy actions (responding to change) are more important. Battlefield commanders succeed by defeating the enemy (the mission), not conforming to a plan. We cannot imagine a battlefield commander saying, "We lost the battle, but by golly, we were successful because we followed our plan to the letter." Battlefields are messy, turbulent, uncertain, and full of change. No battlefield commander would say, "If we just plan this battle long and hard enough, and put repeatable processes in place, we can eliminate change early in the battle and not have to deal with it later on."

A growing number of software projects operate in the equivalent of a battle zone – they are extreme projects. This is where agile approaches shine. Project teams operating in this zone attempt to utilize leading or bleeding-edge technologies, respond to erratic requirements changes, and deliver products quickly. These projects, which we call high-exploration factor projects, do not succumb to rigorous, plan-driven methods.The critical issues with high-exploration factor projects are as follows: first, identifying them; second, managing them in a different way; and third, measuring their success differently. Just as winning is the primary measure of success for a battlefield commander, delivering customer value (however the customer defines it)

In an attempt to improve development productivity, many IT managers and CXOs of companies are jumping to agile development as they clearly see a value in agile (Victor, 2004). So it is a paradigm shift to move away from traditional waterfall model and adopt iterative model which has agile as a subset and SCRUM provides a well thought framework to take projects to completion.

1.1 Strategic shifts in Software development

The last decade has witnessed evolution of alternate software development methodologies as per the business demands and there is a strategic shift to deliver quality product quicker Victor, 2004).

1.2 Shortcomings of Traditional Waterfall Approach

The essence of waterfall software development is that complex software systems can be built in a sequential, phase-wise manner where all of the requirements are gathered at the beginning, all of the design is completed next, and finally the master design is implemented into production quality software. This approach holds that complex systems can be built in a single pass, without going back and revisiting requirements or design ideas in light of changing business or technology conditions. It was first introduced in an article written by Winston Royce, 1970, primarily intended for use in government projects (Winston, 1970).

Waterfall equates software development to a production line conveyor belt. "Requirements analysts" compile the system specifications until they pass the finished requirements specification document to "software designers" who plan the software system and create diagrams documenting how the

code should be written. The design diagrams are then passed to the "developers" who implement the code from the design (See Figure 1). Under the waterfall approach, traditional IT managers have made valiant efforts to craft and adhere to large-scale development plans. These plans are typically laid out in advance of development projects using Gantt or PERT charts to map detailed tasks and dependencies for each member of the development group months or years down the line. However, studies of past software projects show that only 9% to 16% are considered on-time and on-budget,(Standish, 1994). As we attempt to summarize current thinking among computer scientists on why waterfall fails in so many cases, we also explore a leading alternative to waterfall: "Agile" methods that focus on incremental and iterative development where requirements, design, implementation, and testing continue throughout the project lifecycle.

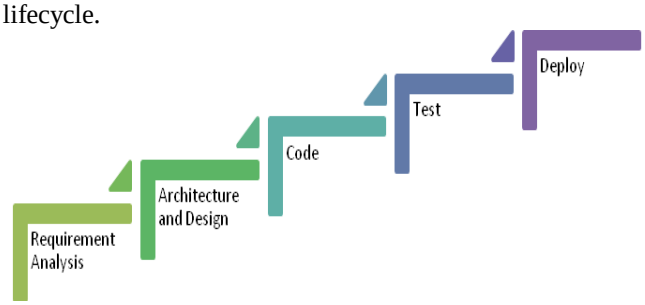

Fig 1.0. Traditional Methods: sequential phased approach

2. METHODOLOGY

SCRUM is not an acronym, but mechanism in the game of rugby for getting an out-of-play ball back into play. It relates to Project out of play for a year – Cost overrun; Scope creep; whenever a project is going out of play – the rules are designed to SCRUM (i.e. put it back into play). It is an iterative, incremental process for developing any product or managing any work. At the end it produces a potentially shippable set of functionality at the end of every iteration.

Characteristics of SCRUM

- Empirical management and control process for any development project
- Used at product companies and IT organizations since 1990
- Wraps existing engineering practices, including Extreme Programming
- Delivers business functionality in 30 days
- Addresses shared development and enhancement of common facilities
- Scalable to distributed, large, and long projects
- CMM Level/2 and Level/3 compliant
- Extremely simple but very hard

Agile Estimating and Planning

Planning is a core activity and it happens at different levels as executed below (Cohn, 2010). The planning Onion is a key pointer as below:

Fig 2:Agile Planning

We have earlier discussed about the Product backlog and Sprint backlog. We can now relate the different layer of planning with sample examples as below:

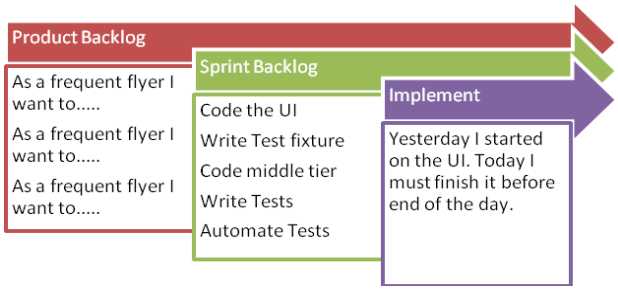

Fig 3. Relationship between Product backlog and Sprint Backlog

Product, Release and Iteration Planning

As the entire development happens in the agile manner, the estimates happen very quickly and story points are the unit used to measure the quantity of work done in a given sprint. Probably the most commonly used estimating unit among agile teams today. The Name is derived from agile teams commonly expressing requirements as "user stories". Based on everything that influences the effort to develop a feature, it is Unitless but numerically relevant estimates. A 10-point user story is expected to take twice as long as a 5-point user story.

Advantages of Agile:

These are the advantages (Lederer& Prasad, 1998)

1. Forces the use of relative estimating

• Studies have shown we're better at this

2. Focuses us on estimating the size, not the duration

• We derive duration empirically by seeing how much we complete per iteration

3. Puts estimates in units that we can add together

• Time based estimates are not additive

We should make all attempts to avoid impact from irrelevant and misleading information on our cost estimates (Magne, 2006).

CONCLUSION

Does Agile/SCRUM work? Of course the proof is always in the pudding, and the mostrecent 2004 Standish Group CHAOS report on the success of software projects shows a dramatic improvement in the failure rate of software projects. In 1994, Standish reported a 31% failure rate that has improved to 15% in 2004.23 Standish Chairman Jim Johnson attributes the improvement to smaller projects using iterative processes as opposed to the waterfall method.24

The notion that Agile is a radical deviation from the long established, tried and true history of waterfall software development is incorrect. Although waterfall is often referred to as "traditional", software engineering has had a very short history relative to other engineering disciplines. Unlike bridge building, software development is not built on thousands of years of trial and error, and is therefore in a rapidly evolving infancy as an engineering discipline. Agile is simply the latest theory that is widely replacing the waterfall approach that itself will change and evolve well into the future.25. Powered by research and widespread acceptance, SCRUM is followed in all major business houses.

REFERENCES

1. Barry Boehm, "Software Engineering Economics", Prentice Hall PTR, 1981.
2. Beck, K., 1999, Extreme Programming Explained (Addison-Wesley) , p 30, 46
3. Boehm, B., Turner, R., 2003, Balancing Agility and Discipline – A Guide for the Perplexed (Addison-Wesley)
4. Cockburn, A., 2002, Agile Software Development (Addison-Wesley) , p 44
5. Cohn, Mike, 2010, Adapting to Agile , mountaingoatsoftware.com
6. Cohn, Mike, 2010, Introduction to Agile Estimating and Planning, mountaingoatsoftware.com
7. English, Ross, Speech on Agile-Scrum vs. Waterfall-Cycle Methodology
8. Craig Larman, Victor R. Basili, "Iterative and Incremental Development: A Brief History", Computer, IEEE CS
9. Press, June 2004, p. 48.
10. Highsmith, J., Cockburn A., 2001, Agile Software Development: The Business Of Innovation, IEEE Computer, Sept.
11. Highsmith, J., Cockburn A., 2002, What is Agile Software Development: The Business Of Innovation, IEEE Computer, Oct.

I. Nonaka, H. Takeuchi, "The New New Product Development Game", Harvard Business Review, January 1986, pp. 137-146.

12. Jim Johnson, "ROI, It's Your Job!", Published Keynote Third International Conference on Extreme Programming, 2002.
13. Ken, Orr., 2002, CMM Versus Agile Development Religious Wars and Software Development. Agile Project Management, 3(7)
14. Kent Beck, "Extreme Programming Explained: Embrace Change", Addison-Wesley, 2000, pp. 18-19.
15. Ken Schwaber, Mike Beedle, "Agile Software Development with Scrum", Prentice Hall, 2001, pp. 89-94
16. Martin Fowler, "Is Design Dead", http://martinfowler.com/articles/designDead.html , 2004.
17. Mary Poppendieck, Tom Poppendieck, "Lean Software Development An Agile Toolkit", Addison-Wesley, 2003,p 28-32
18. Paulk, M.C., Weber, C.V., Curtis, B., 1995, The Capability Maturity Model, Guidelines for Improving the Software Process(Addison-Wesley).
19. Sommerville, I., 1996, Software Engineering (Addison-Wesley) , p 446, 466, 471
20. Standish Group International, Inc., "Chaos Chronicles", 1994.