Case Study

Software Development with Agile Methods

Introduction:

Web application development is a much studied, heavily practiced activity. That is, capturing and validating user requirements, estimating feasibility, analysis, design, testing, implementation and maintenance. The current 'post-dot-bomb era' has seen a new method paradigm emerge – championed by individuals such as Kent Beck, Martin Fowler, Robert Martin, and many others. This new method paradigm is referred as 'Agile Methods'.

The current dominant software development paradigm has evolved from the structured analysis and the object-oriented analysis. This paradigm is referred to as the traditional approach or waterfall model and is characterized as plan-driven and process oriented. During the post 2000 era, the environment in which software is conceived, specified, created, and maintained continues to change rapidly and significantly.

What is Agile?

Agile Methodologies are a group of software development methods based on iterative and incremental development. Requirements and solutions evolve through collaboration between members of different cross functional teams. They leverage adaptive planning, iterative development and delivery, a fixed iterative approach, and encourage rapid and flexible response to change.

We describe Agile methods as encouraging "programmers to shed their heavyweight process chains, embrace change, and escape into agility. Advocated methods have short cycle times, close customer involvement, and an adaptive rather than predictive mind set."

Agile method is a "just enough" method strategy because agile methods aim to avoid prescribing cumbersome and time-consuming processes that add little value to the software product and actually elongate the development process.

Why Agile?

Waterfall is like traditional waterfall, literally. It starts at the top and drops in a linear fashion until it is "done". Agile is not as predictive and allows for iterative changes and process. Waterfall is better for projects that are predefined and understood. Agile is better in situations where there is still a bit of uncertainty regarding end solution. Waterfall does not allow for interactive development. Agile enables the team to close the feedback loop quickly and enables communication to start early in the process.

Waterfall method, does not allow going back, once the software is designed and implemented it is hard to change according to time and user needs. The problem can only be fixed by going back and designing an entirely new system, a very costly and inefficient method.

Whereas, Agile methods adapt to change, as at the end of each stage, the logical program, designed to cope and adapt to new ideas from the outset, allows changes to be made easily. With Agile, changes can be made if necessary without getting the entire program rewritten. This approach not only reduces overheads, it also helps in the upgrading of programs.

Agile method has a working product at the end of each tested stage. This ensures bugs are caught and eliminated in the development cycle, and the product is double tested again after the first bug elimination. This is not possible for the Waterfall method, since the product is tested only at the very end.

The following graph presents the success percentage of Agile and Waterfall methodologies:

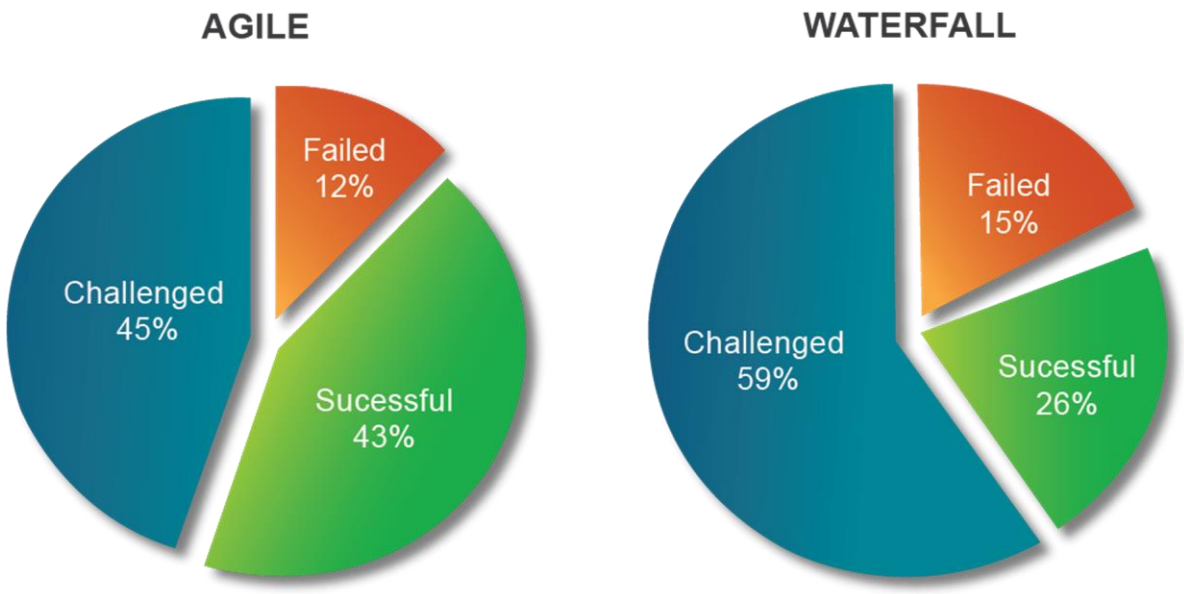

Figure 1: Agile Versus Waterfall

Agile Principles

- Agile's 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 time scale.
- 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 it's behavior accordingly.

Popular Agile Methods:

Adaptive software development: ASD promotes a change-oriented strategy to the software development of large, complex systems. The method encourages incremental and iterative development with constant prototyping, states that "*ASD claims to provide a framework with enough guidance to prevent projects from falling into chaos, but not too much, which could suppress emergence and creativity.*"

Agile modeling: describes the key points of AM as the agile practices and cultural principles. The AM modeling practices encourage developers to produce sufficiently advanced models to meet design needs and all documentation purposes. The cultural principles promote communication, team structure organization and team work practices.

Crystal family: describe a framework of related methods that address the variability of the environment and the specific characteristics of projects. The term "*Crystal*" is used as a metaphor to describe the "*color*" and "*hardness*" or "*heaviness*" of each method. The appropriate Crystal method is selected according to development team size and project criticality.

Dynamic systems development method: describe more of a framework for developing software rather than a particular method. The five phase DSDM life cycle provides for project management activities and risk management.

"*The fundamental idea behind DSDM is that instead of fixing the amount of functionality in a product, and then adjusting time and resources to reach that functionality, it is preferred to fix time and resources, and then adjust the amount of functionality accordingly.*" DSDM is consistently described as the first truly agile software development method.

Feature-driven development: focuses on simple process, efficient modeling, and short, iterative cycles. "FDD depends heavily on good people for domain knowledge, design, and development. A central goal is to have the process in the background to support rather than drive the team." FDD does not assign collective ownership of project tasks.

Scrum: Scrum is unique of all the Agile methodologies because it introduced the idea of "empirical process control." That is, Scrum uses the real-world progress of a project — not a best guess or uninformed forecast — to plan and schedule releases. In Scrum, projects are divided into succinct work cadences, known as sprints, which are typically one week, two weeks, or three weeks in duration. At the end of each sprint, stakeholders and team members meet to assess the progress of a project and plan its next steps. This allows a project's direction to be adjusted or reoriented based on completed work, not speculation or predictions.

Some of the key Scrum practices include:

- Self directed and self organizing teams
- No external addition of work to an iteration, once chosen
- Daily stand up meetings, with special questions
- 30 calendar day iterations
- Demo to external stakeholders at the end of each iteration
- For each iteration, client-driven, adaptive planning

Scrum hangs all of its practices on an iterative, incremental process skeleton. Scrum's skeleton is shown in the diagram above. The lower circle represents an iteration of development activities that occur; one after another. The output of each iteration is an increment of the product. The upper circle represents the daily inspection that occurs during the iteration, in which the individual team members meet to inspect each other's activities and make appropriate adaptations. Driving the iteration is a list of requirements. This cycle repeats until the project is no longer funded.

The SCRUM process overview:

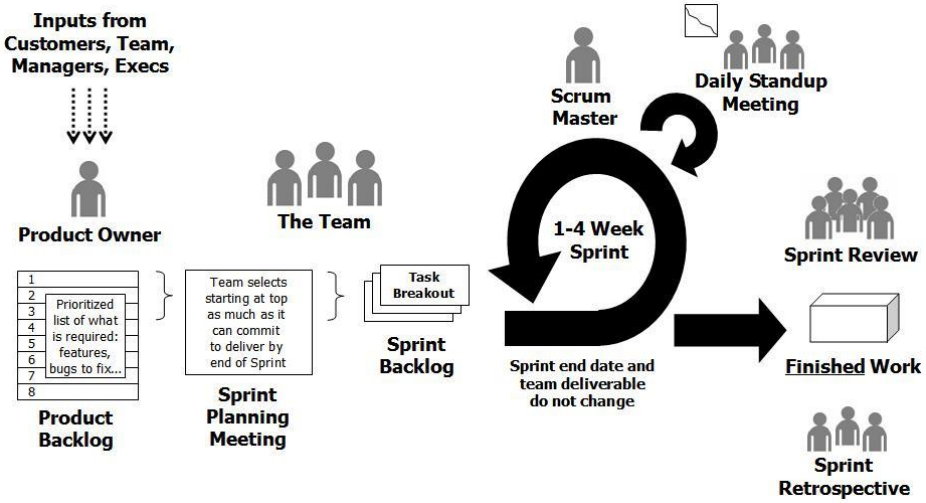

Figure 2: SCRUM Process

The key SCRUM glossary

Product Backlog	All features of the Product
Release backlog	A subset of the product backlog, targeted at the next production quality release
Sprint backlog	Tasks for the iteration. Granularity 4-16 hours
Sprint	Iteration of 30 days duration
Daily scrum meeting	Daily stand up meetings
Team introspections	Reflect and improve upon the learnings
Teams	The team is responsible for developing functionality
Scrum Master	The Scrum Master is responsible for the scrum process, for teaching scrm to everyone involved in the project, for implementing scrum so that it fits within an organization's culture and still deliver the expected benefits, and for ensuring that every one follows scrum rules and practices

Figure 3: SCRUM Glossary

Agile Model

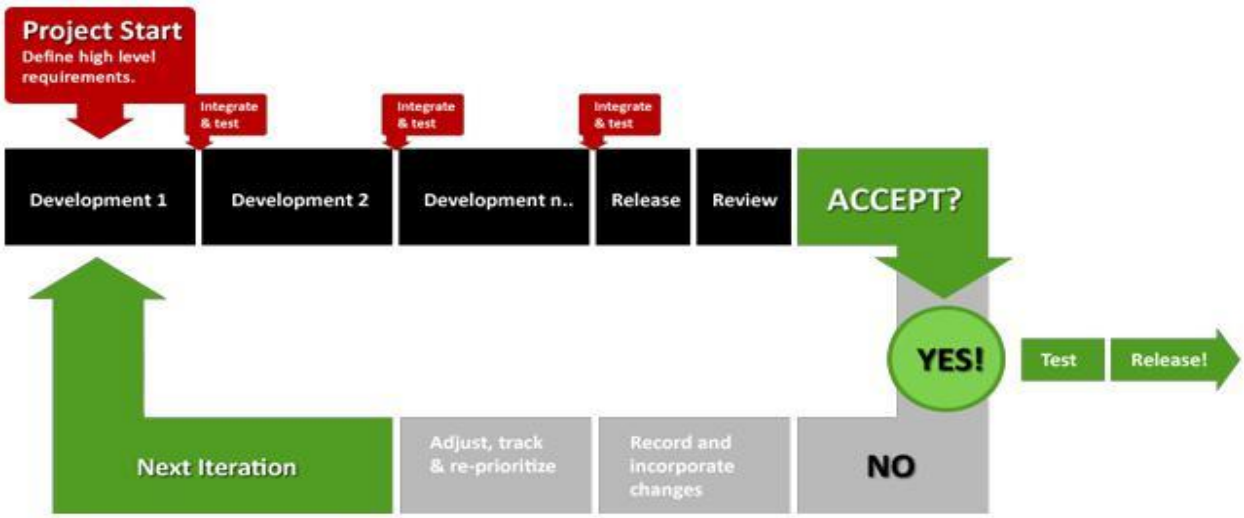

Figure 4: Agile Model

Conclusion: In lieu of the current technological paradigm, the benefits of agile development are both numerous and critically important. To foster long-term value and reduce the threats of project success, it pays to be flexible.

- Customers benefit from Agile as it ensures high customer satisfaction courtesy frequent releases.
- Managers benefit from Agile as it makes life easier for them. No overheads, minimal documentation, no useless meetings and clear & crisp communication with all stakeholders.
- Development team benefits from Agile as it shortens the learning curve and development time..
- Testing team benefits from Agile as they get 'Early Access' to the software to test.
- All stakeholders have a better visibility of the project and 'right expectations' from each other.