Agile Development Methodology

Benefits, Barriers, Beginning

Martin Kearns & Michael Stange

with a foreword by Mike Cohn

Part 1: The Need for Agility

November 2008

Foreword

In the nearly eight years since the creation of the Agile Manifesto, Agile software development has changed from an outlying approach, useful on trivial applications to a valid option worthy of consideration for nearly every software development project. Agile continues, of course, to be used by small teams to deliver websites, in-house software, and all manner of small projects. It is also being used, however, by teams of teams, scaling to over a thousand developers on some projects. Agile processes are being used to deliver just about every type of project you can imagine: from embedded software to Software-as-a-Service, from websites to mainframe, from video games to banking.

In this whitepaper, experienced Agile coaches Martin Kearns and Michael Stange will introduce you to Agile software development. They describe how the need for Agile arose, what it means to be Agile, and conclude with advice on getting started. Along the way, they share some of their experiences with teams and projects that benefitted from some of the benefits of agility, including faster time to market, products that better meet user expectations, improved visibility into the development process, improved quality, and more. The benefits are tangible and available to any organization willing to undertake the hard work of adopting an Agile mindset and approach to its work. Martin and Michael outline the first steps to take toward achieving them.

Mike Cohn
Agile author and founder of Mountain Goat Software

Executive Summary

This White Paper takes a practical look at the benefits, components and implementation of the Agile Development Methodology and examines why organisations are adopting Agile as their software development process. These topics are covered in three parts:

Part 1: The Need for Agility looks at the challenges facing organisations involved in developing business applications today, and reviews the benefits of adopting an Agile approach.

Part 2: What is Agile? provides an explanation of the Agile Methodology, from the reasons it was conceived through to an outline of the processes it employs.

Part 3: Implementing Agile looks at what is involved in adopting Agile for your first software development project. It covers advanced techniques, some dos and don'ts, and other recommendations based on Renewtek's experiences with Agile to help you make your first implementation a success.

Part 1: The Need for Agility

In this section, we glimpse at the challenges facing organisations developing new software applications in today's fast-paced and increasingly complex business and technical environment. We then examine why the Agile Development Methodology is now considered by many to provide the best chance for success.

The Development Challenge

When software programs were first introduced to the corporate world in the decade following the mid 1950s, applications were used to either automate aspects of manufacturing or to perform administrative tasks such as payroll and operations management. In general, the business processes that were automated could be characterised as sequential, well understood and clearly documented. Computer programs at the time ran on a single mainframe platform. This meant that design and construction were distinct and unique phases, requiring a simple yet rigid process theory. These approaches originated in government and defence industries, which had the funds for software development projects in the earliest days, and reinforced the linear and sequential nature of these processes.

In 1970, as computing and the application of these project approaches became more widespread, Winston Royce proposed a sequential software design process known as the 'Waterfall Model'. Characterised by distinctive phases of development that flow steadily downwards, the model advocates investment in the early stages of development – such as requirements documentation and up-front design. Based on the assumption that all requirements were static, the principal supporting argument for this theory was that early investment reduces the chances of late and expensive penalties caused by change, gaps and flawed design. Although Winston Royce's original proposal was for a double execution of the linear phases, most organisations implement the Waterfall Model as a single pass sequence of events that does not allow for much feedback.[1]

The reasons for the widespread adoption of the model are easily understood. Its disciplined and structured approach allow for clear delineation and division of milestones/responsibilities, making it relatively easy to adopt in organisations that are either operationally or commercially segregated. However, the approach is based on a number of flawed assumptions, namely the deterministic nature of business and that all the requirements are known or discoverable at the beginning of the project and that the requirements do not change during the project execution. We now recognise these assumptions to be false and that following this approach gives the organisation a false sense of control and predictability.

[1] Royce, *Managing the Development of Large Software Systems*

Today, applications are being deployed across multiple architectures and technology platforms; there are infinitely more 'pieces to place', higher expectations and increased demands for change from the business – rendering this simple process theory impracticable. The risks are that, after completing a large development, the end result does not provide the desired outcomes and the application fails to meet user acceptance and expectations.

When working in a complex environment with multiple architectures, technologies and touchpoints, it is logical to regularly stop and check that the project is on track. These 'reflection points' enable small changes to be made, if necessary, and also act as a series of small, realistic goals to be achieved over the life of the project.

This realisation led a group of software development veterans to create the Agile Manifesto – a new approach to the design, management and delivery of significant software development projects which addresses these challenges head-on.

Since the publication of the Agile Manifesto in 2001, adoption has been rapid. According to Forrester Research, by mid-2007[2]:

- A quarter of enterprises were using Agile processes, and nearly half of the remaining firms were aware of Agile.
- The majority of enterprise that were aware of Agile were interested in adopting it or were planning to adopt it.
- The larger the company, the more likely it was to use Agile, with 34% of companies with 20,000+ employees already using Agile, compared to 21% of companies with 1,000-4,999 employees.

How can an Agile Approach Help?

Breaking down the challenges facing today's development teams, we can see that any 'better' way of conducting application development must address the following criteria:

- application fit to business requirements
- speed to market
- reducing project risk
- maintaining project control
- delivering quality
- delivering business value

Best fit to requirements

The business requirements of a software system continually change. With a Waterfall approach this leads to depreciation of the system – as illustrated in Figure 1 below. This rate of depreciation is faster than the rate at which software can be developed and, eventually, the software must retire and be replaced.

Under the Agile Development Methodology, at the beginning of each iteration the developers collaborate with the end-users through a product owner, in order to understand the next highest value business features to be included in the iteration. This increased control allows stakeholders to continually evolve and revise priority of requirements throughout the project. As Figure 1 illustrates, regular feedback, prioritisation and adaption to change allows the developing product to grow in a climate of change, thereby appreciating its value to the business. With iterative, collaborative requirements analysis, the design evolves in accordance with the business need, so each iteration steadily increases the business value delivered.

[2] Forrester Research, *Enterprise and SMB Software Survey, North America and Europe Q3 2007*

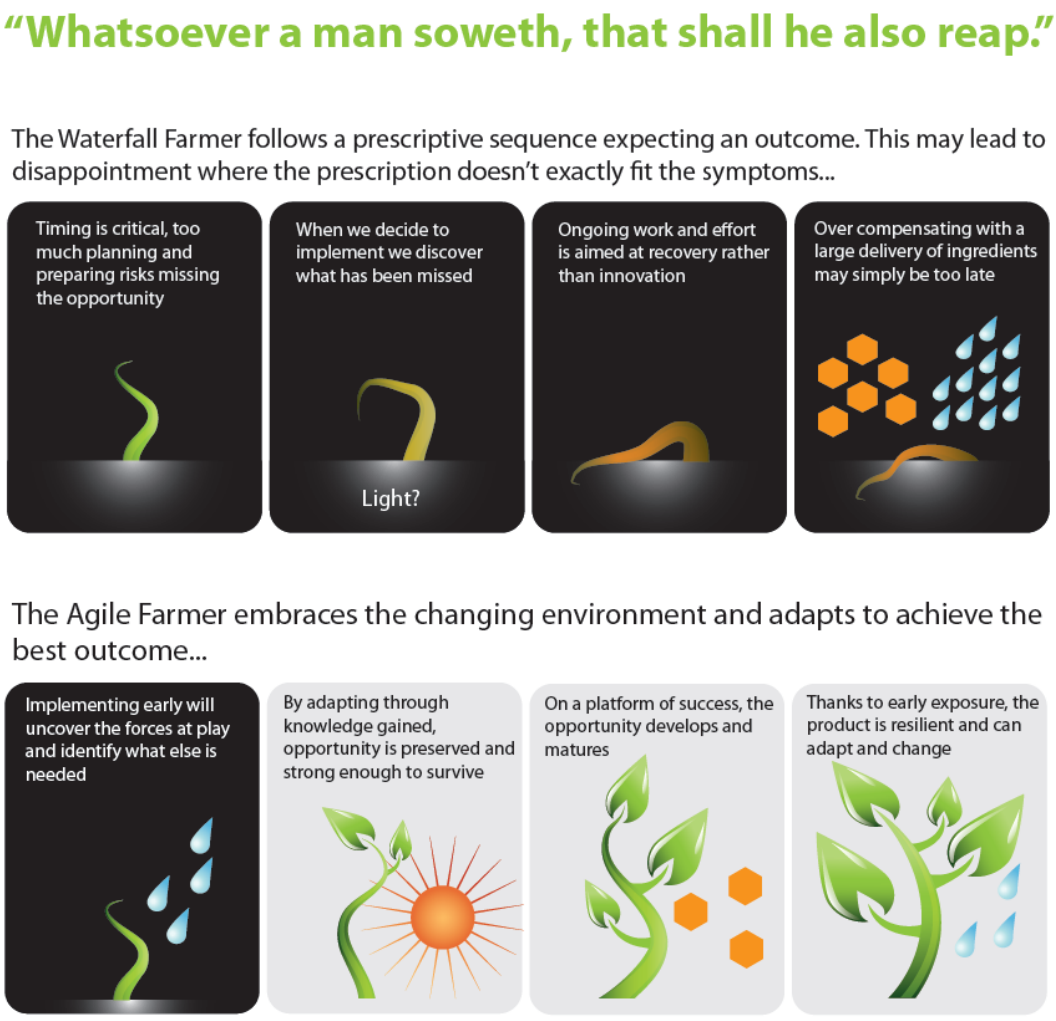

Figure 1: Comparison of the Waterfall Model to Agile methodology

This collaborative approach forges a close working relationship between developers and stakeholders, ensuring that the developed product accurately matches the business requirements and no precious domain information is lost.

When Waterfall methodology won't deliver in time…

Renewtek's client, the division of a major bank, was planning a three-year development project for global roll-out. The development team concluded that, using the Waterfall method, there would be no deliverables within the first year. It was decided that, in the case of this application, that was an unreasonable and unjustifiable investment, so the Agile approach was chosen and Renewtek engaged to assist in coaching the development team.

Reduced Time to Market

A collaborative approach to development can dramatically speed delivery of a software system. This involves empowering a co-located development team which consists of software specialists and stakeholders. Time delays are significantly reduced as decisions can be made in real time and minimum time wastage occurs; a huge advantage over traditional methods.

By focusing on high priority requirements and delivering them early in the project , the business is able to realise the business value by deploying this production quality software much earlier than ever thought possible. In this way, working software is the primary measure of project progress. Furthermore, stakeholders, be they business users or end customers, are able to provide vital

feedback through live production experience with the software, which will enhance the quality of requirements and relevance to the business problem.

Meanwhile, requirements that are only partially understood can either be deferred until they become more important, or explored and investigated until they become clearer.[3]

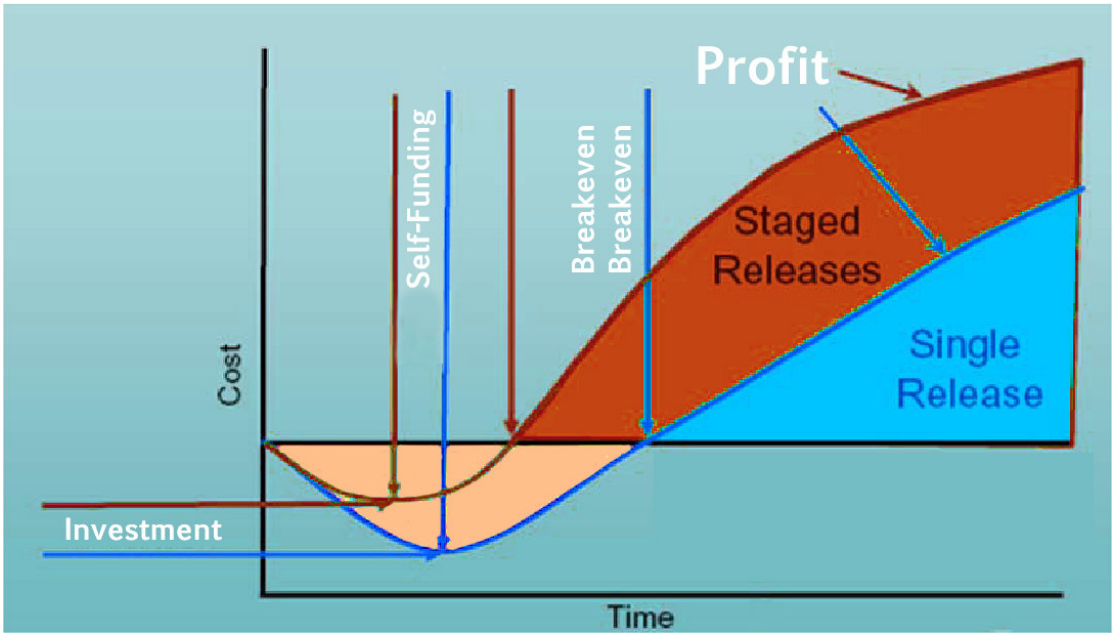

Figure 2: Comparison of single release approach to the iterative release approach

As Figure 2 above shows, the iterative approach requires less upfront investment and yields returns sooner. Going forward, costs of development are offset by the returns earned from earlier releases.

Sprinting towards success

In 2007, Renewtek was engaged to develop and deliver a middleware solution for a client in a seven-week timeframe. The goal was to develop 15 interfaces to support essential processes for stock and purchase orders to their warehouse and distribution partners.

This was our first engagement with this client and we anticipated a learning curve in understanding their needs. Further, the requirements were incomplete and would necessarily evolve throughout the project.

We approached this project by establishing an initial release backlog of all known work that was co-owned with the client. We then agreed on an Agile approach, with the client directing a series of one-week 'sprints'. A Solution Architecture evolved to enable the iterative development of messages and quickly gained the confidence of the business stakeholders, as their expectations were managed and met each week.

Increased Management Control

Change is inevitable as business needs and understanding evolves over time. An Agile approach embraces this and assists stakeholders in articulating their vision by using the iterative approach. It gives them the ability to continually change requirements and priorities, and to define the scope for the next iteration. Project funds can be released in a staged plan in which future investment is based on realised business value from previous iterations.

[3] Denne & Jane Cleland-Huang, *Software by Numbers*

By using the principles of Test-Driven Development and automated testing, time spent on regression testing is reduced while stakeholder confidence and speed of change are increased.

Reduced Risk

Agile methodology holds a principle of absolute transparency for project schedule and progress. It dramatically reduces the risk of projects by allowing stakeholders to evaluate the software after each iteration. Feedback is encouraged and can be prioritised for the next iteration. This has the effect of localising the impact of an issue to the size of the iteration. The higher the risk the shorter the iteration should be.[4]

Project progress reports are also transparent and readily available, providing a basis for project estimation that is based on the actual rate at which the team is progressing.

Issues and risks are managed by a clearly-defined escalation process and with an empowered, unified collaborative development team.

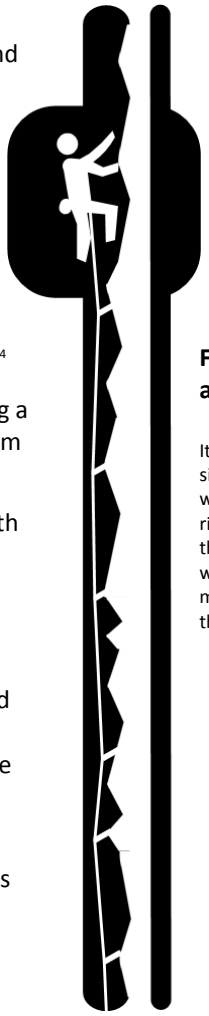

Figure 3: The climber analogy

Iterative development takes a similar approach to a climber, who would never consider the risk of falling from the top to the bottom, so secures the rope with bolts to ensure that any mistake will not cause a life threatening fall.

Highest Quality Software

Agile strives to always produce the highest quality production software, through utilising automated daily builds, Continuous Integration, Automated Test Suites and Test-Driven Development. This practice of incorporating testing into the development cycle (rather than at the end) ensures software quality and user acceptance.

Agile also supports the creation of team synergy through self organisation. The development team is empowered to solve problems and make decisions themselves, creating a feeling of responsibility and ownership. The team is encouraged to find its own most efficient way of working together. Dependence on individuals is reduced, as information is shared throughout the team in the fast-paced collaborative environment.

Simplicity and practical solutions, where architecture concentrates on immediate functionality, is key to allowing good quality software to develop and evolve. For example, a simple whiteboard and cards can be used for real-time visual display of current tasks and progress. The best designs come from teams whose members continually define the tasks and components of work to be developed.

A Summary of Agile Benefits

Here are some of the ways in which using Agile for the development of large technology projects makes good business sense:

- Working software is given to the business in smaller, timelier releases, so business value is delivered up front.
- Feedback is received on a series of brief, iterative project stages – so development is less likely to 'go off track', which reduces wasted development time and thus development costs.
- An earlier break-even point is reached in the investment/project than with traditional development methodologies – which mitigates delivery risk.
- Agile provides transparency to process and delivery – giving the business tighter control over the speed of project delivery as well as deliverables.
- Agile supports – and, in fact, encourages – the business prioritisation of functionality.
- Agile delivers more functionality, more quickly – and at a higher level of quality – in short, sharp, focused releases to production.

[4] Schwaber & Beedle, *Agile Software Development with Scrum*

- The Agile process provides a strong interface for communications and engenders a highly collaborative business/technology team, empowered to make decisions.

In the next part of this White Paper, we are going to look more closely at what Agile Development Methodology involves.

In the next part of this White paper…

Part 2: What is Agile? looks at the why, what and how of the Agile Development Methodology. It covers the background behind the original articulation of the methodology; its particular attributes; its 'ecosystem' – or the roles various entities play within Agile; some of its component processes and tools; and three of its underlying 'lean' principles.

References

Mike Cohn, *Agile Estimating and Planning,* Prentice-Hall, 2005; *User Stories Applied: For Agile Software Development,* Addison-Wesley, 2004

Mark Denne & Jane Cleland-Huang, *Software by Numbers: Low-Risk, High-Return Development*, Prentice-Hall, 2003

Ron Jeffries, www.xprogramming.com, 2008

Winston W.Royce, *Managing the Development of Large Software Systems*, 1970 – see www.cs.umd.edu/class/spring2003/cmsc838p/Process/waterfall.pdf

Ken Schwaber & Mike Beedle, *Agile Software Development with Scrum*, Prentice Hall, 2001

Michele Sliger & Stacia Broderick, *The Software Project Manager's Bridge to Agility,* Addison-Wesley, 2008

Jeff Sutherland, *The Scrum Papers: Nuts, Bolts and Origins of an Agile Process*, updated 2007 – see http://jeffsutherland.com/scrum/ScrumPapers.pdf

Various, *The Agile Manifesto*, 2001 – see www.agilemanifesto.org

About the Authors

Martin Kearns, Principal Consultant (Agile Practice Lead), Renewtek

Martin is one of the first three Certified Scrum Coach coaches in the world. Working with Renewtek, he consistently uses the Scrum framework in conjunction with Agile methodologies where appropriate. Martin joined Renewtek in 2005 where he has responsibility for designing training courses around Scrum/Agile and leads the promotion of Agile principles within Renewtek and to its diverse client base. He offers consulting services to organisations on the creation and implementation of Agile delivery.

Martin conducts formal Scrum training to introduce concepts and core principles in a unique and creative way. He can also play a continuing mentoring role to assist with follow-up implementation and Agile Retrospectives at key project milestones.

Michael Stange, Agile Coach and Project Manager, Renewtek

Michael started coaching and developing teams in the defence force in 1994 and, since joining the IT industry, has worked on both traditional and Agile projects in various roles. He has been employing Agile and XP practices and principles for five years, and has been focusing on process improvement in all areas of the software development lifecycle.

In his previous role, Michael was Agile practice lead for a large Australian consulting organisation, implementing many successful Agile-based client projects. Michael joined Renewtek in early 2008 as both an Agile and Scrum mentor and a project manager, providing guidance to internal staff and delivering client software projects. As a Certified Scrum Master and Prince2 practitioner, Michael employs a pragmatic approach – maintaining the right balance of agility and process governance to deliver business value to Renewtek's clients.

About Renewtek

Renewtek is a technical consultancy and project delivery specialist. We help organisations minimise their business risk through the provision of commonsense consulting advice and our in-depth understanding of how technology can best be utilised to achieve the desired business results.

Renewtek has strategic partnerships with Adobe, BEA Systems, FileNet, IBM, Microsoft and Oracle. Our extensive experience can mitigate the business challenges associated with implementing new technology, upgrading existing technology assets and the integration of disparate technologies.

MELBOURNE & SYDNEY **www.renewtek.com**

Renewtek Pty Ltd ABN 11 110 232 623
4/410 Collins Street, Melbourne VIC 3000 Australia T +61 3 9670 7790 F +61 3 9670 6663
8/58 Pitt Street, Sydney NSW 2000 Australia T +61 2 9252 4999 F +61 2 9252 4139
9th Floor N'orch Building, 170-172 Bui Thi Xuan, District 1, Ho Chi Minh City, Vietnam
email **info@renewtek.com** **www.renewtek.com**