Overview of Agile Management & Development Methods

Addicam.V.Sanjay

Abstract – Agile Development

This paper presents an overview of Agile Project Management & Development Methods. The paper will describe

- The deficiency of normal project development processes
- Who started Agile Management & development methods
- What is it made up of and who should use it
- Differences between Agile development processes and normal development processes
- Analogies of Agile project development & management methods in contrast with Normal development Processes
- The effectiveness of Agile development processes
- Give examples of Agile project development methods, methods like Extreme Programming and Scrum
- The drawbacks of Agile development methods
- Suggest ways in which the limitations of these Agile development methods can be overcome.

Current Status Agile Development

What is the current Software Project Management & development environment???

> "The Problem for engineers is that change translates into chaos, especially when a single error can potentially bring down an entire system. But, change also translates into opportunity. It's as simple as this: if there is time to put a certain amount of functionality into the product easily, then there is time to put in more functionality at the price of a certain amount of disruption and risk. Thus does madness creep into our projects – we will tend to take on as much risk as we possibly can."

This statement by James Bach in "American Programmer" effectively sums up the current Software Management method. There is a constant need for delivering "more" in a given amount of time. The other attributes of the current development environment can be described by the following attributes.

- Availability of skilled professionals - the newer the technology, tools, methods, and domain, the smaller the pool of skilled professionals
- Stability of implementation technology - the newer the technology, the lower the stability and the greater the need to balance the technology with other technologies and manual procedures

- Stability and power of tools - the newer and more powerful the development tool, the smaller the pool of skilled professionals and the more unstable the tool functionality
- Effectiveness of methods - what modelling, testing, version control, and design methods are going to be used, and how effective, efficient, and proven are they
- Domain expertise - are skilled professionals available in the various domains, including business and technology

Modelling these Software Development Process Attributes

Can we create a Model to take care of these attributes???

Attempts to model this development process have encountered the following problems.

- Many of the development processes are uncontrolled
- The inputs and outputs are either unknown or loosely defined
- The transformation process lacks necessary precision
- Quality control is not defined. Testing processes are an example.

Managing Current Development Processes

And, the current Management & development processes handle all these problems ….. Do they???

Currently, most software management & development is considered a "chaotic" activity, better known as "code and fix". This means software is written without much of an underlying plan, and the design of the software system is cobbled together. However, there has always been an alternative to this chaotic development by using methodologies (Fowler, 2000).

These methodologies are typically known as "heavy" or "monumental". Examples of them include the waterfall, spiral, etc. They impose a strong emphasis on process especially upfront planning. Even though they have been around for quite some time, they are not noted for being very successful or popular (Fowler, 2000).

The unpopularity of these "heavy" methodologies is a result of the massive effort required throughout the process which can actually slow down development.

Introducing Agile Management and Development Method

And, Now Ladies And Gentlemen, the Distinguished Agile Management & Development Method!!

What is Agile Management and Development Method

Agile Management & Development Method is one of a growing number of alternatives to traditional, process-centric software management methods with a focus on people, results, minimal methods and maximum collaboration. It is geared to the high speed and high change of today's ebusiness projects. (Highsmith, 2000).

History of Agile Development

Who is behind this "Agile" thing???

In February of 2001, a group of people, frustrated with the existing heavy software methodologies met in Utah to find some common ground in alternate software development. They came up with this manifesto:

> We are uncovering better ways of developing software by doing it and helping others do it. Through this work we have come to value:
>
> - Individuals and interactions over processes and tools
> - Working software over comprehensive documentation
> - Customer collaboration over contract negotiation
> - Responding to change over following a plan

That is, while there is value in the items on the right, we value the items on the left more. (Agile Alliance, 2000)

This manifesto is the cornerstone of all the different Agile Software Management & Development methods.

Agile Development Characteristics

What are the characteristics of an Agile Software Management & Development???

Today's time-sensitive business climate requires that we quickly accommodate requirements changes during development and, after development, be equally adept a delivering the upgrades caused by software's rapid software evolution and the customer's ever-increasing requirements (Aoyama, 2000).

A dominant idea in agile development is that the team can be more effective in responding to change if it can

- Reduce the cost of moving information between people, and
- Reduce the elapsed time between making a decision to seeing the consequence of that decision
- Place people physically closer
- Replace documents with talking in person and at whiteboards, and
- Improver the team's amicability-its sense of community and morale- so that people are more inclined to relay valuable information quickly
- Make user experts available to the team or, even better, part of the team and
- Work incrementally

(Cockburn et al.,2001)

Where to use Agile Development

Where do you use Agile Management & Development Methods???

Agile Management & development methods are used under the following circumstances:

1. Your customers/users are active participants in your requirements and/or analysis modelling efforts
2. Changing requirements are welcomed and acted upon accordingly – there is no "requirements freeze"
3. You are working on the highest priority requirements first, as prioritized by your project stakeholders, and in turn focusing on highest risk issues as work progresses
4. You are taking an iterative and incremental approach to modelling
5. Your primary focus is on the development of software, not documentation or the models themselves
6. You are modelling as a team where everyone's input is welcome
7. You are actively trying to keep things as simple as possible – You are using the simplest tools available to you and creating the simplest model(s) that do the job
8. You are discarding most, if not all, of your models as development progresses
9. Customers/business owners make business decisions, developers make technical decisions
10. The content of your models is recognized as being significantly more important than the format/representation of that content
11. How you will test what you are describing with your model(s) is a critical issue being continually considered as you model (Ambler, 2001)

Where not to use Agile Development

Where do you not use Agile Management & Development Methods???

Agile development methods aren't used under the following circumstances:

1. Your goal is to produce documentation, such as a requirements document, for sign-off by one or more project stakeholders
2. You are using a case tool to specify the architecture and/or design of your software BUT not using that specification to generate part or all of your software
3. Your customers/users have limited involvement with your efforts. For example they are involved with initial development of requirements, perhaps are available on a limited basis to answer questions, and at a later date will be involved in one or more acceptance reviews of your work
4. You are focusing on a single model at a time. Common examples are "use case modelling sessions", "class modelling sessions", or "data modelling sessions." The root cause of this problem is typically "one artefact developers" such as people specialized in data modelling or user interface modelling – with Agile Method generalists should be leading the effort.
5. You are working towards a freeze of one or more of your models – In other words you are taking a serial approach.

6. You are delivering models and/or documentation to another team who will then evolve the system further. In other words you are "handing off" your work in a serial manner

(Ambler, 2001)

Agile Management Method Analogy

Er… Can you give me an analogy between Agile management methods and the normal heavy weight software development method ….

Let us assume that you are designing a custom coffee maker. With Agile Development methods you will

1. Meet with the customer; create a high level list of features the coffee maker will do. Get sign-off on the requirements
1. Chunk these into features that could be delivered in six-week intervals
2. Ask the customer to prioritize the list
3. Create a detailed plan to implement the first feature including a detailed user acceptance test case
4. Implement the plan for the first feature (for example, make coffee using coffee grounds and cold water). Deliver the feature to the customer, involving them with questions whenever necessary
5. First feature is delivered, and approved by the customer. The time it took to deliver is used for a baseline to predict the sizing of future features. The customer is asked again to reprioritize, and pick the next feature to be delivered in six weeks based on the new estimates
6. Repeat steps 3, redoing existing features as necessary to add new features until the entire solution is in place

(Russell, 2002)

In the case of a normal heavy weight software Management & development process, you will

1. Meet with the customers
2. Model the processes required for the custom coffee maker
3. Get sign-off on the requirements to ensure that the customer does not change their minds
4. Create a detailed project plan of the entire project, and assign resources to tasks. Begin
5. Project progresses. Different individuals working on different pieces may contact customer with similar or different questions. Difficult to assess overall status as plan is challenged with normal disruptions and surprises. Schedule problems early are addressed by shortening future tasks, like testing
6. Complete project. Customer requests many modifications. Project becomes a maintenance project rather than a development project

(Russell, 2002)

O.k., fine…what are the currently available Agile development processes, which are available for an user?

Extreme Programming

eXtreme Programming is a new approach to software management which takes a code-centric view of the activity. XP offers several compelling features:

- Comprehensive unit tests,
- Short release cycles,
- Adding only what's needed for the current task,
- Collective code ownership,
- Continual improvement, and
- Adding features in the order of importance

(Chromatic, 2001)

The development environment in an Organization which uses XP is characterized by these procedures:

- Customer lists the features that the software must provide
- Programmers break the features into stand-alone tasks and estimate the work needed to complete each task
- Customer chooses the most important tasks that can be completed by the next release
- Programmers choose tasks, and work in pairs
- Programmers write unit tests
- Programmers add features to pass unit tests
- Programmers fix features/tests as necessary, until all tests pass
- Programmers integrate code
- Programmers produce a released version
- Customer runs acceptance tests
- Version goes into production
- Programmers update their estimates based on the amount of work they've done in release cycle

(Chromatic, 2001)

SCRUM:

- Scrum is an agile, lightweight process to manage and control development work.
- Scrum is a wrapper for existing engineering practices.
- Scrum is a team-based approach to iteratively, incrementally develop systems and products when requirements are rapidly changing

- Scrum is a process that controls the chaos of conflicting interests and needs.
- Scrum is a way to improve communications and maximize co-operation.
- Scrum is a way to detect and cause the removal of anything that gets in the way of developing and delivering products.
- Scrum is a way to maximize productivity.
- Scrum is scalable from single projects to entire organizations. Scrum has controlled and organized development and implementation for multiple interrelated products and projects with over a thousand developers and implementers.
- Scrum is a way for everyone to feel good about their job, their contributions, and that they have done the very best they possibly could

(Control Chaos, 2002)

Crystal

The crystal family of lightweight SDLC methodologies was created by Alistair Cockburn. Crystal is a family of human-powered and adaptive, ultralight, "shrink-to-fit" software Management & development methodologies.

- "Human-powered" means that the focus is on achieving project success through enhancing the work of the people involved (other methodologies might be process-centric, or architecture-centric, or tool-centric, but Crystal is people-centric).
- "Ultralight" means that for whatever the project size and priorities, a Crystal-family methodology for the project will work to reduce the paperwork, overhead and bureaucracy to the least that is practical for the parameters of that project.
- "Shrink-to-fit" means that you start with something possibly small enough, and work to make it smaller and better fitting.

Crystal is non-jealous, meaning that a Crystal methodology permits substitution of similar elements from other methodologies (Cockburn, 2001)

DSDM

The Dynamic Systems Development Method (DSDM) is a lightweight software methodology which has its origins in the U.K. In traditional approaches the focus has been on satisfying the contents of a requirements document and conforming to previous deliverables, even though the requirements are often inaccurate. The previous deliverables may be flawed and the business needs may have changed since the start of the project. In addition, time and resources are often allowed to vary during development.

In DSDM, the exact opposite is true, time is fixed for the life of a project, and resources are fixed as far as possible. This means that the requirements that will be satisfied are allowed to change (DSDM, 2001).

DSDM has underlying principles that include active user interaction, frequent deliveries, empowered teams, testing throughout the cycle. Like other agile methods

they use short time boxed cycles of between two and six weeks. There's an emphasis on high quality and adaptivity towards changing requirements (Fowler, 2001).

Wisdom

The Whitewater Interactive System Development with Object Models (Wisdom) addresses the needs of small development teams who are required to build and maintain the highest quality interactive systems (Nunes & Cunha, 2000). The Wisdom methodology has three key components:

1. A software process based on user-centered, evolutionary, and rapid-prototyping model
2. A set of conceptual modeling notations that support the modeling of functional and nonfunctional components
3. A project management philosophy based on tool usage standards and open documentation

(Wolak, 2001)

Users of Agile Management Methods

Are there any companies which are using Agile Management methods ???

I found these companies making use of Agile development methods:

- Northrop Grumman Commercial Aircraft division
- Texas Instrument
- Marlow
- Loral Vought Systems
- Center for Collaborative Manufacturing
- E-systems
- Tracor, Inc (UTA, date not mentioned).

Escrow.com also uses agile development methods. Escrow.com has documented the results of their success with agile development methods in this table:

	Before Adopting Agile	After Adopting Agile	% change
Total code size	45,773	15,048	-67%
Average Methods per class	6.30	10.95	+73%
Average lines per method	11.36	5.86	-48%
Average cyclometric complexity	3.44	1.56	-54%

(Agilelogic, 2000)

The data in the table speaks for the success of the company with Agile development methods.

Advantages of Agile Development

What are the advantages of Agile Management & development methods???

The benefits are significant and include:

- Shortened development cycle-time of 75%
- Higher stability of work-loads
- Higher utilization of work-load, that is, developing large-scale, software systems with a fixed number of developers,
- Higher flexibility to change of Management & development plans,
- Higher quality by earlier feedback from the customers.

(Ayoma, 2000)

Limitations of Agile Development

What are the limitations of Agile development methods???

The limitations of Agile Management methods are:

- Agile development methods do not scale. Due to the integrative approach, it is hard for some to understand exactly where the project stands. In a typical environment, upper management wants to know when each phase is completed such as design, code, or test. Thus, due to the various iteration steps, it can be hard to understand if the project is on track
- Agile management methods do not handle large teams well. The approach only works for small to medium-sized teams
- Agile development requires highly skilled and highly motivated individuals, which may not always be existent. Agile places a premium on having premium people

Personal View on Agile Development Methods

My two cents on Agile development methods!!!

I immensely like the short development cycles prescribed in all of the Agile development methods. This is very practical and is true in many organizations. I have also observed in my professional experience in the software industry that having short development cycles of 15 days to one month, and going for a review of the entire project after that really is very pragmatic.

Now for the darker side of Agile development methods. Agile development methods place too much premium on people. It expects highly motivated people. It also expects colleagues to interact in a very close manner, everyday, without disagreement, emotions or bias. This will just not happen. It is very tough to work very closely with a group or people and get along with them. People will have disagreements everyday, which will affect the way in which future interactions with clients and fellow developers will be affected. If the heavy emphasis on people is not sorted out, Agile may end up being a very bold experiment.

Conclusion

Agile methods are light weight software methods. Agile development methods are very pragmatic in understanding the fact that requirements in a business environment changes constantly. Highly creative people who have understood the shortcomings of normal software management processes are using agile development methods in organizations. Many organizations all around the world are trying out the various available Agile development methods.

References:

AgileAlliance .(2001, February). History: The Agile Manifesto. Retrieved Sept 22 2004, from the World Wide Web: http://agilemanifesto.org/history.html

Amber, Scott. (2002). When and when aren't you Agile Modeling? Retrieved Sept 22 2004, from the World Wide Web: http://www.agilemodeling.com/essays/whenAreYouAgileModeling.html

Aoyama, Mikio. (1998, November). IEEE Software: Web-based Agile Software Development. Retrieved Sept 22 2004, from the World Wide Web: http://rockfish-cs.cs.unc.edu/COMP290-S02/Aoyama-98.pdf

Aoyama, Mikio. (????). Summary of Progress-Implementation Projects. Retrieved Sept 22 2004, from the World Wide Web: http://delivery.acm.org/10.1145/310000/302164/p3-aoyama.pdf?key1=302164&key2=1550017101&coll=portal&dl=ACM&CFID=2074172&CFTOKEN=30174701

Chromatic. (2001, May). O'Reilly Open Source Convention: An Introduction to Extreme Programming. Retrieved Sept 22 2004, from the World Wide Web: http://linux.oreilly.net.com/pub/a/linux/2001/05/04/xp_intro.html

Cockburn, Alisair., Highsmith, Jim . (2001, September). Agile Software Development: The people Factor. Retrieved Sept 22 2004, from the World Wide Web: http://www.adaptivesd.com/Articles/IEEEArticle2Final.pdf

Cockburn, Alistair. (2001, October). Philosophy of crystal Methodologies. Retrieved Sept 22 2004, from the World Wide Web: http://crystalmethodologies.org/philosophy.html

Control Chaos. (2001). SCRUM software Development process. Retrieved Sept 22 2004, from the World Wide Web: http://www.controlchaos.com/scrumwp.htm

Control Chaos. (2002). What is Scrum? Retrieved Sept 22 2004, from the World Wide Web: http://www.controlchaos.com/scrumo.htm

Disaster. (2001, July). Recipes for Disaster. Retrieved Sept 22 2004, from the World Wide Web: http://www.cio.com/archive/070101/secret_sidebar_2_content.html

DSDM. (2001). Overview: Why is DSDM different. Retrieved Sept 22 2004, from the World Wide Web: http://www.dsdm.org/en/about/overview.asp

Fowler, M. (2000, December). Put your process on a diet software development. Retrieved Sept 22 2004, from the World Wide Web: http://www.sdmagazine.com/articles/2000/0012/0012a/0012a.htm

Fowler, Martin. (2001, November). The New Methodology. Retrieved Sept 22 2004, from the World Wide Web: http://www.martinfowler.com/articles/newmethodology.html

Fowler, Martin., Highsmith, Jim. (2001, August). The Agile Manifesto. Retrieved Sept 22 2004, from the World Wide Web: http://www.sdmagazine.com/documents/x=844/sdmo108a/0108a.htm

Highsmith, Jim. (2001). Adaptive Software Development. Retrieved Sept 22 2004, from the World Wide Web: http://www.chc-3.com/cs511/fall2001/talks/team2_asd.ppt

Hodgetts, Paul., Phillips, Denise. (2000). Extreme Adoption experiences of a B2B startup. Retrieved Sept 22 2004, from the World Wide Web: http://www.agilelogic.com/files/extremeAdoptioneXperiencesofaB2Bstartup.pdf

Interaction Design. Using XP to develop context-sensitive Adverts for the web. Retrieved Sept 22 2004, from the World Wide Web: http://www.id-book.com/casestudy-xp.html

Marick, Brian. (2001, Febraury). Agile Methods and Agile Testing. Retrieved Sept 22 2004, from the World Wide Web: http://www.testing.com/agile/agile-testing-essay.html

Maurer, Frank. (2002). XP in detail: University of Calgary. Retrieved Sept 22 2004, from the World Wide Web: http://sern.ucalgary.ca/courses/SENG/609.24/W2002/slides/XP2.ppt

Nelson, Elden. (2002, January). Extreme Programming vs Interaction Design. Retrieved Sept 22 2004, from the World Wide Web: http://www.fawcette.com/interviews/beck-cooper/

Nunes, N., Cunha, J. (2000). Wisdom: A software engineering method for small software development companies IEEE software, September/October 2000, 113- 119.

Pickering, Chris. (2001, June). Agile Is In. Retrieved Sept 22 2004, from the World Wide Web: http://itmanagement.earthweb.com/columns/ecomm/article/0,,2791_7_8031,00.html

Roger, K.J., Whitman, Larry., Underdown,Ryan. (1998). The Enterprise Integration issues Encountered with Agile Process Introduction. Retrieved Sept 22 2004, from the World Wide Web: http://arri.uta.edu/eif/rogersfaim98.pdf

Russell, Lou. (2001). AGILE Development: A Tale of Two Cooks Retrieved Sept 22 2004, from the World Wide Web: http://www.russellmartin.com/articles/agile%20development.htm

UTA. Summary of Progress-Implementation Projects. Retrieved Sept 22 2004, from the World Wide Web: http://arri.uta.edu/aamrc/2_3_1.html

Wolak, Chaelynne. (2001, July). Extreme Programming (XP) uncovered. Retrieved Sept 22 2004, from the World Wide Web: http://www.scisstudyguides.addr.com/papers/cwdiss725paper4.pdf

Wolak, Ronald. (2001, April). DISS 725-System Development: Research Paper1 SDLC on a Diet: Retrieved Sept 22 2004, from the World Wide Web: http://www.scisstudyguides.addr.com/papers/rwDISS725researchpaper1.pdf

The Author:

Addicam Sanjay has over 7 years experience in the Consumer Electronics Industry in Technical as well as Management positions. His primary focus is in producing Home networking software with quick turn around time and with engineers spread in multiple geos. Sanjay received his Masters in Information technology from University of Maryland. His other works are available at http://members.cox.net/asanjay

Project Perfect:

Project Perfect sell "Project Administrator" software, which is a tool to assist organisations better manage project risks, issues, budgets, scope, documentation planning and scheduling. They also created a technique for gathering requirements called "Method H"™, and sell software to support the technique. For more information on Project tools or Project Management visit www.projectperfect.com.au