Introducing Agile Development

Agile software development refers to a group of software development methodologies based on similar principles. It is not a single approach, but rather a family of processes. In 2001, a group of prominent members of the "lightweight methods" community came together[1], adopting the term "agile methods" and publishing the "Agile Manifesto"[2], describing the main principles of these methods. The manifesto spawned a movement in the software industry, known as agile software development.

Agile principles

The Agile Manifesto lists best practices according to Agile development strategies. It states:

"We are uncovering more effective ways of developing software by engaging in it ourselves and helping others do it as well. 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*

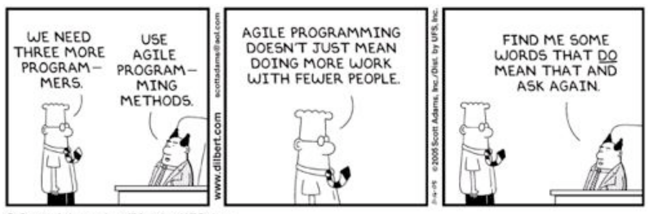
That is, while there is value in the items on the right, we value the items on the left more."
These statements are further explained in the Manifesto, resulting in 12 principles of Agile development:

- Customer satisfaction is achieved through rapid, continuous delivery of useful software
- Working software is delivered quickly (weeks rather than months)
- Working software is the principal measure of progress
- Even late changes in requirements are welcomed
- Close, daily cooperation between business people and developers
- Face-to-face conversation is the best form of communication (Co-location)
- Projects are built around motivated individuals, who should be trusted
- Continuous attention to technical excellence and good design is crucial
- Simplicity, - the art of maximizing the amount of work not done--is essential.
- Self-organizing teams must be emphasized
- Regular adaptation to changing circumstances is vital

Functional principles of Agile are reflected in concepts such as error proofing, eliminating waste, creating flow, adding customer value, and empowering workers.

Goal and advantages

From a project management perspective, Agile principles promote frequent inspection and adaptation, while encouraging teamwork, self-organization, and accountability of team members. From an engineering perspective, Agile aims to rapidly deliver high-quality software, while the business approach remains focused on aligning development with customer needs and optimizing return-on-investment.


WE NEED THREE MORE PROGRAM-MERS.
USE AGILE PROGRAM-MING METHODS.
www.dilbert.com
scottadams@aol.com
AGILE PROGRAMMING DOESN'T JUST MEAN DOING MORE WORK WITH FEWER PEOPLE.
© 2005 Scott Adams, Inc./Dist. by UFS, Inc.
FIND ME SOME WORDS THAT DO MEAN THAT AND ASK AGAIN.

© Scott Adams, Inc./Dist. by UFS, Inc.

How it works

There is little if any consensus on what types of software projects are best suited for Agile development. It has been widely documented as working well for small (< 10 developers), co-located teams, although scale or geography by themselves are not necessarily barriers to success.

Small increments

Agile methods favor an iterative approach with small increments and minimal planning rather than long-term planning. Iterations are completed in short time frames called "timeboxes" which typically last from one to four weeks. The goal is to have an available release at the end of each iteration (though the iteration may not add enough functionality to warrant a release). An important detail is that each iteration offers additional functionality over the last iteration's result and the product becomes more full-featured as it grows closer to its final release.

Team composition

Agile project teams are (ideally) cross-functional, self-organizing teams. In this context, "cross-functional" does not imply that each team member should be able to handle all tasks in an iteration, but rather indicates a correct mix of required skills. To encourage teamwork and individual accountability, there should be no consideration for any existing corporate hierarchy, and team structure will ideally arise naturally. Each Agile team will contain a customer representative, appointed by the stakeholders to act on their behalf. The customer representative should be available to answer problem-domain questions during the entire iteration.

Face-to-face communication

Since Agile focuses on direct communication, many methods encourage teams to be located in a single open office. Teams should have regular, formal face-to-face (or voice/video conference) meetings, in which the customer representative and (optionally) observing stakeholders are present. The goal of these meetings is to ensure that problems are discovered early and progress is frequently inspected.

Progress measurement

Agile methodologies are sometimes said to be the opposite of disciplined, plan-driven methods. This is a misleading distinction, however, since it would imply that Agile methods are unplanned or undisciplined. Envisioning them on a continuum from "adaptive" to "predictive" presents a better comparison between Agile and other methodologies. Traditional ("Waterfall") methodologies can be said to be on the predictive end, while agile methods lie on the adaptive side. Since adaptive methods focus on responding to changing realities, describing exactly what will happen in the future becomes harder. The further away a date is, the less clear an adaptive method will be about what exactly will happen on that date. This is why features that are yet to be realized are set at the beginning of a time box and progress is measured by the functionality and quality of the iteration's release.

Agile methodologies

In spite of the fact that Agile is a relatively new methodology, there are several well-known methods that implement Agile strategy, each with their own focus. Some well-known Agile software development methods are:

- *Agile modeling* – involve stakeholders in modeling
- *Agile Unified Process (AUP)* – adaptation of IBM's Rational Unified Process (RUP)
- *Extreme Programming (XP)* – prescribes daily stakeholder practices
- *Feature Driven Development (FDD)* - combining best practices for client-valued functionality (features).
- *Scrum* - process skeleton including practices and predefined roles.

Comparing Agile

Agile methods have much in common with Rapid Application Development (RAD) techniques. The iterative approach and timeboxing principles play an important role in both types of development. But there are also several important differences:

1. Agile focuses on actual solutions rather than prototypes and mock-ups
2. Feature list is broken down and implemented incrementally
3. Quality of the iteration end products is important from the first iteration
4. Agile uses self-managed teams
5. Daily communication: all team members are aware of obstacles and achievements

Much like RAD, Agile is often described as a reaction to the "heavyweight" development methods, where the waterfall model is implemented using heavily regulated, planned, and micro-managed principles. The main difference is Agile's emphasis on obtaining the smallest workable piece of functionality to deliver business value early, continually improving it and adding functionality throughout the project.

Cowboy coding is the absence of a defined method. Because of the frequent re-evaluation of plans and emphasis on face-to-face communication rather than creating documentation, Agile is sometimes confused with cowboy coding. This is understandable, but incorrect. Agile methods follow structured, disciplined, and often rigorous processes.

Popularity

Today's fast-paced business environment requires an organization's development process to be flexible and adaptable to changing needs. As a result, increasingly more companies are using Agile methodologies in their projects to reduce risk and deliver value to the business sooner than traditional development methods were used. Recent yearly surveys[3] by The Agile Alliance show that the key reason people are adopting Agile is twofold: managing changing requirements and priorities and accelerating time-to-market. Although it is hard to measure popularity of Agile development, there is no denying that the use of agile methods has been on the rise since its official naming in 2001.

Agile Novulo; bringing theory to practice

As with all software methodologies, the choice of development platform plays an important role in the success of its implementation. For Agile, this means that that the chosen development platform should help uphold the four main values in agile development throughout all phases of the project, in order to fully take advantage of its envisioned benefits. The Novulo platform delivers on these values and is aligned with the Agile strategy.

When looking at the software demands of today's business community, it is easy to understand why the various Agile approaches are becoming increasingly more popular. Software developers combine best practices in an effort to deliver maximum quality software in a timely fashion while remaining flexible with regard to changing requirements. So, how can these developers implement the four Agile precepts using Novulo?

- **Individuals and interactions over processes and tools**
  Organization, teams, and groups are made up of individuals. In order for the team, group, or organization to 'work together' and meet common goals, they have to interact. The nature of how they interact, and the efficiency of the interaction, can severely affect the performance of that team. Consequently, tools and processes should exist to improve the efficiency of interaction. Novulo can provide an efficient format for interaction between team members, where demonstrating a design choice by example can save lots of time by avoiding misunderstandings (this is one of the focal points of Extreme Programming).

- **Working software over comprehensive documentation**
  In Agile development, creating value for the customer is deemed more important than writing documentation for existing code. This does not mean that documentation is not important; it is a choice that an Agile developer must make between spending time on creating verbose documentation or not. Novulo, however, removes the need to make this choice. Since code is generated from a complete application model, the model itself contains detailed information about structure, processes, and interfaces. The pragmatic interface provides great visual feedback about the inner workings of the system!

  Another principle in agile development related to this value states that the developer should increase customer satisfaction through rapid, continuous delivery of useful software. This principle is also a precise description of one of the primary strengths of the Novulo development platform. The ability to generate high-quality working software, combined with the flexibility in model, allows developers to easily provide customers with working applications with increased functionality at the end of each iteration.

- **Customer collaboration over contract negotiation**
  In a simple supplier consumer model, contract negotiation can be very simple. The consumer wants a product that does *abc*. The

supplier offers to build the product for cost *xyz*. The consumer (hopefully) agrees and the supplier supplies. Unfortunately, when it comes to software development, this is hardly ever the case. In practice, this type of contract negotiation involves risks for both the customer and the developer:

- Requirements (*abc*) are not understood initially or are liable to change – risk to the customer
- Costs (*xyz*) are not fully understood or may change – risk to the developer
- Customer may not know if abc is what is needed until it is delivered and used – risk to the customer
- People delay because of reluctance to sign off on *abc* – cost to both parties
- It costs money to define (or redefine) the contract such that what is to be delivered is understood - risk to both parties

Agile development proposes a different approach, where instead of requirements, an approach is specified. The work to be done is then split into iterations. During each iteration, the customer and developer work together to find out how to best create value for the customer for the money that will be spent during the next iteration. It is a simple proposition that can work well in reducing the listed risks and costs.

Consequently, according to Agile, the customer should not be signing off on a fixed set of requirements, but should be agreeing to an approach. Understandably, a customer might also be hesitant to sign a contract that specifies an approach instead of a set of requirements.

Novulo helps remove this hesitation as the developer can directly clarify the approach. Use the time otherwise spent on comprehensive listing of requirements on a simple initial model in the Novulo Architect. This helps flush out basic requirements while the customer immediately gets a feel for the approach to be used in the development phase. The initial design can then be used as a basis for listing features to be added in the first iterations. Use Novulo to inspire confidence in your development methods and turn skeptics into avid believers!

- **Responding to change over following a plan**
  One of the common misconceptions about this value statement is that Agile advocates unplanned development, leading to chaos. Planning is not a bad thing. In fact, it is so good that you should do it again and again when faced with change.

  Many things may change during the life of a project: requirements, costs, management, priorities, resources, and even understanding of the problem are only a few examples. The first one is the most important one since it has a direct effect on the software to be built. To decide on changes to be made with regard to the software, the developer should collaborate with the customer.

  Novulo provides a system in which you communicate your understanding of the application without creating unnecessary documentation. This means that in order to solve changed requirements, the developer (and customer) can review the proposed solutions in a familiar environment, reducing misconceptions and minimizing efforts spent on documentation that does not provide added value to the business.

Conclusion

Agile was not conceived with the publication of the Agile Manifesto; it is a combination of pre-existing best practices in the software development industry. The manifesto itself proclaims that Agile is an ongoing evolving process that promotes making optimal use of practices that exist and have proven their worth and combining these practices with new methods to better create high-quality software.

It is against this backdrop that Novulo has evolved, and so it is not surprising that many of the principles valued by Agile development methods can be actively supported in the Novulo development platform. The Business Aligned Software Delivery system in Novulo provides a very pragmatic implementation of the most important principle in Agile development:

"Our highest priority is to satisfy the customer through early and continuous delivery of valuable software"[5]