

Agile Methodology

Changing ways of software development...

Beneficial For

Persons who would like to gain knowledge about software development processes/methodologies, project team member practitioners of project management field, programmers, product managers, business analysts and developers.

➤ Summary

Agile Software Development is an introduction to lightweight software development philosophy, thought, and recommended practices.

There are a number of agile software development methods, such as those espoused by The Agile Alliance. Most agile methods attempt to minimize risk by developing software in short time boxes, called iterations, which typically last one to four weeks. Each iteration is like a miniature software project of its own, and includes all of the tasks necessary to release the mini-increment of new functionality: planning, requirements analysis, design, coding, testing, and documentation. While iteration may not add enough functionality to warrant releasing the product, an agile software project intends to be capable of releasing new software at the end of every iteration. At the end of each iteration, the team reevaluates project priorities.

Introduction:

Agile software development is a conceptual framework for undertaking software engineering projects. The goal of Agile Methods is to allow an organization to be agile, but what does it mean to be Agile? Jim Highsmith says that being Agile means being able to Deliver quickly. Change quickly. Change often [Highsmith, et. al., 2000]. While Agile techniques vary in practices and emphasis, they share common characteristics, including iterative development and a focus on interaction, communication, and the reduction of resource intensive intermediate artifacts. Developing in iterations allows the development team to adapt quickly to changing requirements. Working in close location and focusing on communication means teams can make decisions and act on them immediately, rather than wait on correspondence. Reducing intermediate artifacts that do not add value to the final deliverable means more resources can be devoted to the development of the product itself and it can be completed sooner.

There are a number of agile software development methods, such as those espoused by The Agile Alliance. Most agile methods attempt to minimize risk by developing software in short time boxes, called iterations, which typically last one to four weeks. Each iteration is like a miniature software project of its own, and includes all of the tasks necessary to release the mini-increment of new functionality: planning, requirements analysis, design, coding, testing, and documentation. While iteration may not add enough functionality to warrant releasing the product, an agile software project intends to be capable of releasing new software at the end of every iteration. At the end of each iteration, the team reevaluates project priorities.

Key Features

➤ Agile Software Development:

What is Agile software development? What is the mean to be Agile?

➤ Principles behind agile methods:

Some of the principles behind the Agile Manifesto.....

➤ Comparison with other methods:

Agile Methods will probably not win over traditional methods but live in symbiosis with them.

➤ Is Your Organization Ready for Agile Methods?

Agile methods are a family of development processes, not a single approach to software development, to select from, each bringing practices that will change the daily work of the organization according to the need and size of project.

➤ Criticism of Agile development:

Every coin has two sides. Agile development also has some limitations as believed by some practitioners.

There are a number of agile software development methods, such as those espoused by The Agile Alliance. Most agile methods attempt to minimize risk by developing software in short time boxes, called iterations, which typically last one to four weeks. Each iteration is like a miniature software project of its own, and includes all of the tasks necessary to release the mini-increment of new functionality: planning, requirements analysis, design, coding, testing, and documentation. While iteration may not add enough functionality to warrant releasing the product, an agile software project intends to be capable of releasing new software at the end of every iteration. At the end of each iteration, the team reevaluates project priorities.

Agile – denoting “the quality of being agile; readiness for motion; nimbleness, activity, dexterity in motion” – software development methods are attempting to offer an answer to the eager business community asking for lighter weight along with faster and nimbler software development processes. This is especially the case with the rapidly growing and volatile Internet software industry as well as for the emerging mobile application environment.

Agile methods emphasize real-time communication, preferably face-to-face, over written documents. Most agile teams are located in a bullpen and include all the people necessary to finish software. At a minimum, this includes programmers and their "customers" (customers are the people who define the product; they may be product managers, business analysts, or actual customers). The bullpen may also include testers, interaction designers, technical writers, and managers.

Agile methods also emphasize working software as the primary measure of progress. Combined with the preference for face-to-face communication, agile methods produce very little written documentation relative to other methods. This has resulted in criticism of agile methods as being undisciplined.

Principles behind agile methods — The Agile Manifesto

Agile methods are a family of development processes, not a single approach to software development. In 2001, 17 prominent figures in the field of agile development (then called "light-weight methodologies") came together at the Snowbird ski resort in Utah to discuss ways of creating software in a lighter, faster, more people-centric way. They created the Agile Manifesto, widely regarded as the canonical definition of agile development, and accompanying agile principles.

Some of the principles behind the Agile Manifesto are:

- Customer satisfaction by rapid, continuous delivery of useful software
- Working software is delivered frequently (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.
- Projects are built around motivated individuals, who should be trusted
- Continuous attention to technical excellence and good design.
- Simplicity
- Self-organizing teams
- Regular adaptation to changing circumstances

The publishing of the manifesto spawned a movement in the software industry known as agile software development.

Key Features

→ Agile Software Development:

What is Agile software development? What is the mean to be Agile?

→ Principles behind agile methods:

Some of the principles behind the Agile Manifesto.....

→ Comparison with other methods:

Agile Methods will probably not win over traditional methods but live in symbiosis with them.

→ Is Your Organization Ready for Agile Methods?

Agile methods are a family of development processes, not a single approach to software development, to select from, each bringing practices that will change the daily work of the organization according to the need and size of project.

→ Criticism of Agile development:

Every coin has two sides. Agile development also has some limitations as believed by some practitioners.

Comparison with other methods

Agile methods are sometimes characterized as being at the opposite end of the spectrum from "plan-driven" or "disciplined" methodologies. This distinction is misleading, as it implies that agile methods are "unplanned" or "undisciplined." A more accurate distinction is to say that methods exist on a continuum from "adaptive" to "predictive". Agile methods exist on the "adaptive" side of this continuum.

Adaptive methods focus on adapting quickly to changing realities. When the needs of a project change, an adaptive team changes as well. An adaptive team will have difficulty describing exactly what will happen in the future. The further away a date is, the more vague an adaptive method will be about what will happen on that date. An adaptive team can report exactly what tasks are being done next week, but only which features are planned for next month. When asked about a release six months from now, an adaptive team may only be able to report the mission statement for the release, or a statement of expected value vs. cost.

Predictive methods, in contrast, focus on planning the future in detail. A predictive team can report exactly what features and tasks are planned for the entire length of the development process. Predictive teams have difficulty changing direction. The plan is typically optimized for the original destination and changing direction can cause completed work to be thrown away and done over differently. Predictive teams will often institute a change control board to ensure that only the most valuable changes are considered.

Agile methods have much in common with the "Rapid Application Development" techniques from the 1980's as espoused by James Martin and others

Contrasted with iterative development:

Most agile methods share iterative development's emphasis on building releasable software in short time periods. Agile methods differ from iterative methods in that their time period is measured in weeks rather than months and work is performed in a highly collaborative manner. Most agile methods also differ by treating their time period as a strict time box.

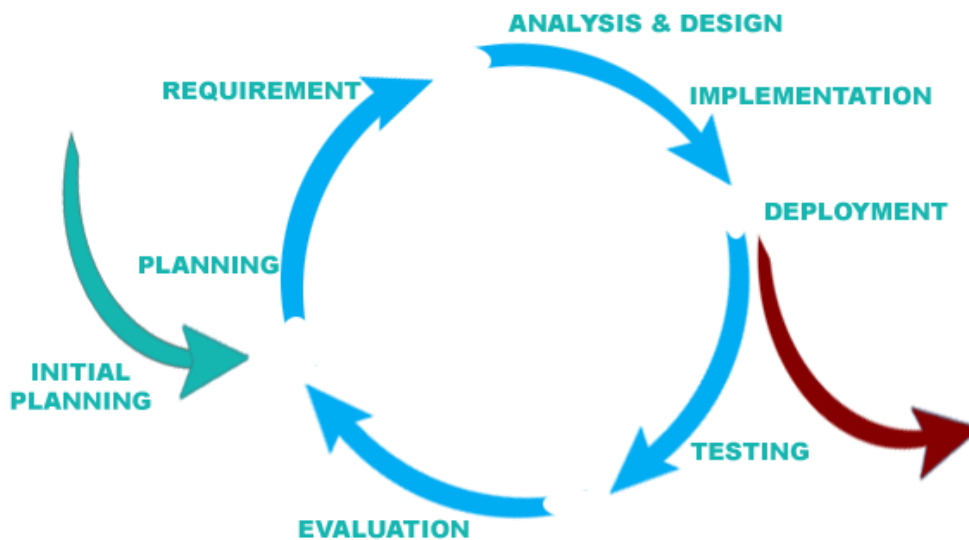


Fig: Iterative & Incremental Development

Key Features

→ Agile Software Development:

What is Agile software development? What is the mean to be Agile?

→ Principles behind agile methods:

Some of the principles behind the Agile Manifesto.....

→ Comparison with other methods:

Agile Methods will probably not win over traditional methods but live in symbiosis with them.

→ Is Your Organization Ready for Agile Methods?

Agile methods are a family of development processes, not a single approach to software development, to select from, each bringing practices that will change the daily work of the organization according to the need and size of project.

→ Criticism of Agile development:

Every coin has two sides. Agile development also has some limitations as believed by some practitioners.

Contrasted with the waterfall model:

Agile development has less in common with the waterfall model. In the eyes of some the waterfall is discredited, but as of 2004, this model is still in common use.[5] The waterfall model is the most predictive of the methodologies, stepping through requirements capture, analysis, design, coding, and testing in a strict, pre-planned sequence. Progress is generally measured in terms of deliverable artifacts—requirement specifications, design documents, test plans, code reviews and the like. The waterfall model can result in a substantial integration and testing effort toward the end of the cycle, a time period typically extending from several months to several years. The size and difficulty of this integration and testing effort is one cause of waterfall project failure.[citation needed] Agile methods, in contrast, produce completely developed and tested features (but a very small subset of the whole) every few weeks or months. The emphasis is on obtaining the smallest workable piece of functionality to deliver business value early, and continually improving it/adding further functionality throughout the life of the project.

Some agile teams use the waterfall model on a small scale, repeating the entire waterfall cycle in every iteration.[citation needed] Other teams, most notably Extreme Programming teams, work on activities simultaneously.

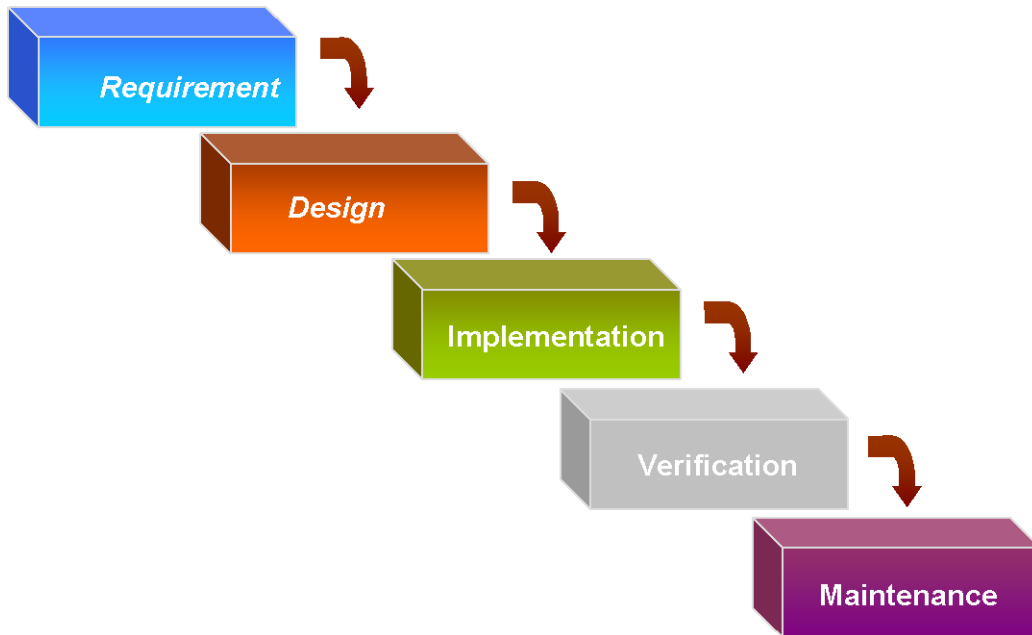


Fig: Waterfall Model

Agile Methods

Some of the well-known agile software development methods are as follows:

- Extreme Programming (XP)
- Scrum
- Agile Modeling
- Adaptive Software Development (ASD)
- Crystal Clear and Other Crystal Methodologies
- Dynamic Systems Development Method (DSDM)
- Feature Driven Development (FDD)
- Lean software development
- Agile Unified Process (AUP)

Key Features

Agile Software Development:

What is Agile software development? What is the mean to be Agile?

Principles behind agile methods:

Some of the principles behind the Agile Manifesto.....

Comparison with other methods:

Agile Methods will probably not win over traditional methods but live in symbiosis with them.

Is Your Organization Ready for Agile Methods?

Agile methods are a family of development processes, not a single approach to software development, to select from, each bringing practices that will change the daily work of the organization according to the need and size of project.

Criticism of Agile development:

Every coin has two sides. Agile development also has some limitations as believed by some practitioners.

A Selection of Agile Methods

Agile Methods have much in common, such as what they value, but they also differ in the practices they suggest. In order to characterize different methods, we will examine the following Agile Methods: Extreme Programming, Scrum, Crystal Methods, Feature Driven Development, Lean Development, and Dynamic Systems Development Methodology. We will attempt to keep the depth and breadth of our discussion consistent for each method, though it will naturally be limited by the amount of material available. XP is well-documented and has a wealth of available case studies and reports, while DSDM is subscription-based, making it much more difficult to find information. The 12 other methods lie somewhere in between.

The table below presents the collected prescriptive characteristics of the discussed methods.

Table 1. Prescriptive Characteristics

	XP	Scrum	Crystal	FDD	LD	DSDM	AM
Team Size	2-10	1-7	Variable	Variable	N/A		
Iteration Length	2 weeks	4 weeks	< 4 months	< 2 weeks			
Distributed Support	No	Adaptable	Yes	Adaptable			
System Criticality	Adaptable	Adaptable	All types	Adaptable			

As we have seen, different Agile Methods have different characteristics. A brief comparison of Crystal Clear and XP resulted, for example, in the following [Highsmith, et. al., 2000]:

- XP pursues greater productivity through increased discipline, but it is harder for a team to follow
- Crystal Clear permits greater individuality within the team and more relaxed work habits in exchange for some loss in productivity
- Crystal Clear may be easier for a team to adopt, but XP produces better results if the team can follow it
- A team can start with Crystal Clear and move to XP; a team that fails with XP can move to Crystal Clear

Key Features

→ Agile Software Development:

What is Agile software development? What is the mean to be Agile?

→ Principles behind agile methods:

Some of the principles behind the Agile Manifesto.....

→ Comparison with other methods:

Agile Methods will probably not win over traditional methods but live in symbiosis with them.

→

→ Criticism of Agile development:

Every coin has two sides. Agile development also has some limitations as believed by some practitioners.

Is Your Organization Ready for Agile Methods?

As we have seen, there are many Agile Methods to select from, each bringing practices that will change the daily work of the organization. Before an organization selects and implements an Agile Method, it should ponder whether or not it is ready for Agile or not. Scott Ambler discusses factors affecting successful adoption in his article *When Does (n t) Agile Modeling Make Sense*. Number one on his list, Agile adoption, will be most successful when there is a conceptual fit between the organization and the Agile view. Also important for adoption are your project and business characteristics. Is your team already working incrementally? What is the team's motivation? What kind of support can the team expect? Are there adequate resources available? How volatile are the project requirements? Barry Boehm suggests using traditional methods for projects where requirements change less than 1% per month.

Ambler also suggests the importance of an Agile champion — someone to tackle the team's challenges so they can work easily. Boehm stresses the importance of having well-trained developers, since Agile processes tend to place a high degree of reliance on a developer's tacit knowledge. The customer also needs to be devoted to the project, and must be able to make decisions. Poor customers result in poor systems [Cockburn and Highsmith, 2001a]. Boehm adds, unless customer participants are committed, knowledgeable, collaborative, representative, and empowered, the developed products typically do not transition into use successfully, even though they may satisfy the customer.

Alistair Cockburn lists a few caveats when adopting an Agile process:

- As the number of people on a project grows, there is an increased strain on communications
- As system criticality increases, there is decreased tolerance for personal stylistic variations

If Agile Methods do not seem to be a good fit for your project or organization right off the bat, Ambler suggests partial adoption. Look at your current development process, identify the areas that need the most improvement, and adopt Agile techniques that specifically address your target areas. After successful adoption of the chosen practices and, even better, a demonstrated improvement to your overall process, continue selecting and implementing Agile techniques until you have adopted the entire process.

Criticism of Agile Development

Every coin has two sides. Agile development also has some limitations as believed by some practitioners, some of the criticisms include following:

- lack of structure and necessary documentation
- only works with senior-level developers
- incorporates insufficient software design
- requires too much cultural change to adopt
- can lead to more difficult contractual negotiations

The criticisms regarding insufficient software design and lack of documentation are addressed by the Agile Modeling method which can easily be tailored into agile processes such as XP.

Key Features

→ Agile Software Development:

What is Agile software development? What is the mean to be Agile?

→

→ Comparison with other methods:

Agile Methods will probably not win over traditional methods but live in symbiosis with them.

→ Is Your Organization Ready for Agile Methods?

Agile methods are a family of development processes, not a single approach to software development, to select from, each bringing practices that will change the daily work of the organization according to the need and size of project.

→ Criticism of Agile development:

Every coin has two sides. Agile development also has some limitations as believed by some practitioners.

Agile and CMM (I)

As mentioned above, Agile is a reaction against traditional methodologies, also known as rigorous or plan-driven methodologies. One of the models often used to represent traditional methodologies is the Capability Maturity Model (CMM).

The goals of CMM are to achieve process consistency, predictability, and reliability. Its proponents claim that it can be tailored to also fit the needs of small projects even though it was designed for large projects and large organizations.

Most Agile proponents do not, however, believe CMM fits their needs at all. If one were to ask a typical software engineer whether the Capability Maturity Model for Software and process improvement was applicable to Agile Methods, the response would most likely range from a blank stare to a hysterical laughter. One reason is that CMM is a belief in software development as a defined process that can be defined in detail, algorithms can be defined, results can be accurately measured, and measured variations can be used to refine the processes until they are repeatable within very close tolerances. For projects with any degree of exploration at all, Agile developers just do not believe these assumptions are valid. This is a deep fundamental divide — and not one that can be reconciled to some comforting middle ground.

Many Agile proponents also dislike CMM because of its focus on documentation instead of code. A typical example is the company that spent two years working (not using CMM though) on a project until they finally declared it a failure. Two years of working resulted in 3,500 pages of use cases, an object model with hundreds of classes, thousands of attributes (but no methods), and, of course, no code [Highsmith, 2002b]. The same document-centric approach resulting in documentary bloat that is now endemic in our field is also reported by many others.

While Agile proponents see a deep divide between Agile and traditional methods, this is not the case for proponents of traditional methods. Mark Paulk, the man behind CMM, is surprisingly positive about Agile Methods and claims that Agile Methods address many CMM level 2 and 3. So according to some, XP and CMM can live together, at least in theory. One reason is that we can view XP as a software development methodology and CMM as a software management methodology. CMM tells us what to do, while XP tells us how to do it. Others agree that there is no conflict. Siemens, for example, does not see CMM and Agility as a contradiction. Agility has become a necessity with increasing market pressure, but should be built on top of an appropriately mature process foundation, not instead of it.

CMMI is the latest effort to build maturity models and consists of Process Areas (PA) and Generic Practices (GP). In an attempt to compare Agile and CMMI, Turner analyzed their values and concluded that their incompatibilities are overstated and that their strengths and weaknesses complement each other.

The question whether Agile is hacking is probably less important than whether Agile and CMM (I) can co-exist. This is due to the fact that many organizations need both to be Agile and show that they are mature enough to take on certain contracts. A model that fills that need and truly combines the Agile practices and the CMM key processes has not, that we are aware of, been developed yet.

Key Features

→ Agile Software Development:

What is Agile software development? What is the mean to be Agile?

→ Principles behind agile methods:

Some of the principles behind the Agile Manifesto.....

→ Comparison with other methods:

Agile Methods will probably not win over traditional methods but live in symbiosis with them.

→ Is Your Organization Ready for Agile Methods?

Agile methods are a family of development processes, not a single approach to software development, to select from, each bringing practices that will change the daily work of the organization according to the need and size of project.

→ Criticism of Agile development:

Every coin has two sides. Agile development also has some limitations as believed by some practitioners.

Conclusion

Agile Methods are here to stay, no doubt about it. Agile Methods will probably not win over traditional methods but live in symbiosis with them. While many Agile proponents see a gap between Agile and traditional methods, many practitioners believe this narrow gap can be bridged. Glass even thinks that traditionalists have a lot to learn from the Agile folks and that traditional software engineering can be enriched by paying attention to new ideas springing up from the field.

Why will Agile Methods not rule out traditional methods?

Agile Methods will not out rule traditional methods because diverse processes for software engineering are still needed. Developing software for a space shuttle is not the same as developing software for a toaster. Not to mention that the need to maintain software, typically a much bigger concern than development, also differs according to the circumstances. Software maintenance is, however, not an issue discussed in Agile circles yet, probably because it is too early to draw any conclusions on how Agile Methods might impact software maintenance.

So what is it that governs what method to use?

One important factor when selecting a development method is the number of people involved, i.e., project size. The more people involved in the project, the more rigorous communication mechanisms need to be. According to Alistair Cockburn, there is one method for each project size, starting with Crystal Clear for small projects and, as the project grows larger, the less Agile the methods become.

Other factors that have an impact on the rigor of the development methods are application domain, criticality, and innovativeness. Applications that may endanger human life, like manned space missions, must, for example, undergo much stricter quality control than less critical applications. At the same time, a traditional method might kill projects that need to be highly innovative and are extremely sensitive to changes in market needs.

In conclusion, the selection of a method for a specific project must be very careful, taking into consideration many different factors, including those mentioned above. In many cases, being both Agile and stable at the same time will be necessary. A contradictory combination, it seems, and therefore extra challenging, but not impossible. As Siemens states, We firmly believe that agility is necessary, but that it should be built on top of an appropriately mature process foundation, not instead of it.

Key Features

→ Agile Software Development:

What is Agile software development? What is the mean to be Agile?

→ Principles behind agile methods:

Some of the principles behind the Agile Manifesto.....

→ Comparison with other methods:

Agile Methods will probably not win over traditional methods but live in symbiosis with them.

→ Is Your Organization Ready for Agile Methods?

Agile methods are a family of development processes, not a single approach to software development, to select from, each bringing practices that will change the daily work of the organization according to the need and size of project.

→ Criticism of Agile development:

Every coin has two sides. Agile development also has some limitations as believed by some practitioners.

Where is Agile going?

Agile is currently an umbrella concept encompassing many different methods. XP is the most well known Agile Method. While there may always be many small methods due to the fact that their proponents are consultants who need a method to guide their work, we expect to see some consolidation in the near future. We compare the situation to events in the object-oriented world in the 1990s, where many different gurus promoted their own methodology. In a few years, Rational, with Grady Booch, became the main player on the method market by recruiting two of the main gurus: James Rumbaugh (OMT) and Ivar Jacobsson (Objectory). Quickly the three amigos abandoned the endless debates regarding whose method was superior, which mainly came down to whether objects are best depicted as clouds (Booch), rectangles (OMT), or circles (Objectory), and instead formed a unified alliance to quickly become the undisputed market leader for objectoriented methods. We speculate that the same can happen to the Agile Methods, based, for example, on the market-leader XP. Even if the Agile consolidation is slow or nonexistent, what most likely will happen, independent of debates defining what is and is not Agile, and practitioners will select and apply the most beneficial Agile practices. They will do so simply because Agile has proven that there is much to gain from using their approaches and because of the need of the software industry to deliver better software, faster and cheaper.

References

- [1] [Manifesto for Agile Software Development](#)
- [2] [The Agile Alliance](#)
- [3] **Agile Development and Project Management:**
<http://www.ccpa.com/Resources/documents/AgileProjectManagement.pdf>
- [4] **MSF (Microsoft Solutions Framework) for Agile software development:**
<http://msdn2.microsoft.com/en-us/teamssystem/aa718795.aspx>
- [5] [The Agile Journal - An online magazine for agile software development](#)
- [6] [AgileSoftwareDevelopment.com - Public weblogs and news on agile software development](#)
- [7] [Agile Planet weblog aggregator](#)
- [8] **Matt Stephens'** website [SoftwareReality.com - a critical eye on agile development](#)
- [9] Abrahamsson, P.; Salo, O.; Ronkainen, J.; and Warsta, Juhani, "**Agile software development methods**," VTT Publications 478, 2002
<http://www.inf.vtt.fi/pdf/publications/2002/P478.pdf>
- [10] Agile Software Development, A DACS State-of-the-Art Report
<http://www.dacs.dtic.mil/techs/agile/>
- [11] **Ambler, S., Agile Documentation,**
<http://www.agilemodeling.com/essays/agileDocumentation.htm>
- [12] Abrahamsson, P., Warsta, J., Siponen, M.T., & Ronkainen, J. (2003). **New Directions on Agile Methods: A Comparative Analysis**

Key Features

→ Agile Software Development:

What is Agile software development? What is the mean to be Agile?

→ Principles behind agile methods:

Some of the principles behind the Agile Manifesto.....

→ Comparison with other methods:

Agile Methods will probably not win over traditional methods but live in symbiosis with them.

→ Is Your Organization Ready for Agile Methods?

Agile methods are a family of development processes, not a single approach to software development, to select from, each bringing practices that will change the daily work of the organization according to the need and size of project.

→ Criticism of Agile development:

Every coin has two sides. Agile development also has some limitations as believed by some practitioners.

About OTS Solutions

OTS Solutions is one of the leading offshore software development service providers, offering an array of IT related services to its clients across the globe.

Since our inception, we have steadily grown into a reputed provider of high-quality and cost-effective software development services. Our ability to evolve continuously and flexible approach towards the dynamic business world has helped us to gain success in short span of time. Today, OTS provides offshore as well as onsite software development services to its clients across the globe.