ADI Journal on Recent Innovation (AJRI)
Vol 3 No. 1 September 2021

p-ISSN : 2685-9106
e-ISSN : 2686-0384

Discussion of Agile Software Development Methodology and its Relevance to Software Engineering

Amitkumar Dudhat[1], Muhammad Ali Abbasi[2]
Master of Science in information and communication technology, Veer Narmad South Gujarat University[1], master's in computer engineering, Istanbul Aydin University[2]
India[1], Turkey[2]
e-mail: amit000790@gmail.com, Alyabbasi93@gmail.com

Author Notification
27 August 2019
Final Revised
28 August 2019
Published
03 September 2019

To cite this document:
Dudhat, A., & Ali Abbasi, M. . (2021). Discussion of Agile Software Development Methodology and its Relevance to Software Engineering. *ADI Journal on Recent Innovation*, *3*(1), 105–114.
DOI: 10.34306/ajri.v3i1.536
Hash: ABCCsznr0xMpX5Cb1Nw4y6Vjkix4sD93QG9txZIOFEtyCWAtqCuWIAUAFT6KrVSD

Abstract

Agile Software Development Methodology is a lesser-known and infrequently utilized methodology in academia. In reality, though, content developer software practitioners frequently employ this technique. This journal was created to give readers an overview of agile techniques and their use at various stages of software development in general.

Keywords: Agile Methodology, Software Engineering

I. INTRODUCTION

Over the last few years, agile software development methods have grown in popularity[1]. Agile practices were created with the goal of delivering software quicker and ensuring that the program satisfies changing customer demands. Some argue that agile software development is the "contemporary" alternative for waterfall software development [2].

Traditional plan-driven software development techniques (such as waterfall) have the drawback of being too mechanistic to be applied in detail. As a result, industrial software developers are wary of new solutions that are difficult to understand, and as a result, they go underutilized[3].
Agile software development encompasses a wide range of techniques:

- Adaptive Software Development
- Extreme Programming (XP)
- Dynamic System Development Methodology,
- Feature Driven Development,
- SCRUM,
- Agile Modelling, and
- Crystal,

Various writers focused on various areas of software development. Some were concerned with methods to planning and requirements, while others were concerned with ways to build software that could be modified more readily, and yet others were concerned with the human interactions that allow software engineers to adapt more quickly to changing client demands[4], [5]. These different initiatives served as a focal point for a community that advocated a set of practices that succeed without requiring many of the activities that more defined methods need.

Literature Review

Agile Software Development Overview

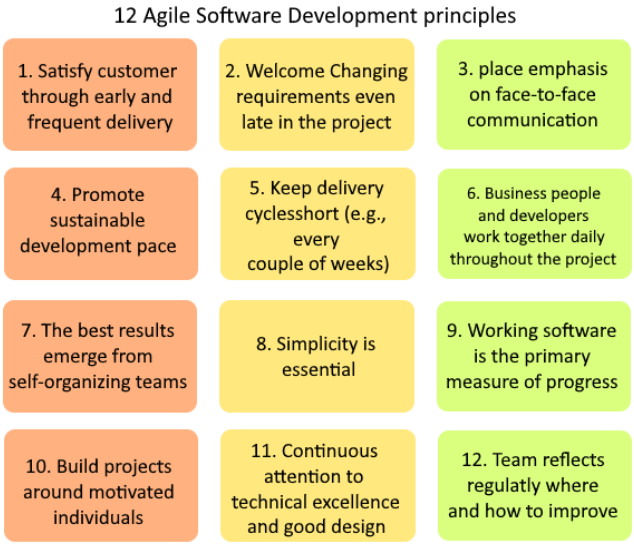
Addison-Wesley Longman is a good place to start. According to dictionary definitions, the term "Agile" refers to the ability to move swiftly and effortlessly[6]. Thus, for a software development organization, " agility" refers to the ability to adapt and respond quickly, effectively, and properly to changes in its environment as well as the expectations imposed by this environment[4], [7].

Extreme Programming, Embrace Change1 was released in the fall of 1999, and the trend had found its spark. In early 2001, a gathering of innovators working on various agile approaches gathered and drafted the "Agile Manifesto for Software Development."

This manifesto emphasizes the following aspects of development:

- Individuals and interactions over procedures and tools
- Working software against thorough documentation
- Collaboration with customers rather than contract negotiations
- Adapting to change in a planned manner

They announced the 12 Agile Software Development principles, which are depicted in Figure 1.



Motivation

Traditional / heavyweight / plan-oriented techniques suffer difficulties and failure, therefore agile or lightweight development methodologies are a solution to the problem. With the characteristics of agile development techniques, developers are attempting to build software fast while maintaining the required quality[8].

Developers in the heavyweight environment have the difficulty of bringing a seemingly endless backlog of software projects to fruition while staying on top of the newest advancements. Most software projects fail against some measure of success, according to study after survey. Software is supplied late, beyond budget, and fails to satisfy quality standards[9]. Furthermore, determining the reasons of these failures is challenging. However, most projects fail for one or more of the reasons listed below:

- Requirements that are not clearly communicated
- Requirements that do not solve the business problem
- Requirements that change before the project is completed
- Software (code) that has not been tested
- Software that has not been tested in the way that the user will use it
- Software that has been developed in such a way that it is difficult to modify
- Projects that are not staffed with the resources necessary in the project plan
- Schedule and scope commitments are made before fully comprehending the needs or the technical hazards

At the same time, several projects that did not need binders of paperwork, comprehensive drawings, or project plans were highly effective[10]. Without all these extra stages, many experienced programmers had excellent success. The individuals working on the project, rather than the technology or procedures utilized, proved to be the decisive factor in project success. These events serve as a catalyst for engineers to abandon heavyweight techniques in favour of agile approaches to software development[11].

II. METHODE

Characteristics

The following are some of the most frequent characteristics of agile development methodologies:

a. Lightweight

Agile techniques are easier to utilize than traditional (heavyweight) approaches because they need fewer steps to analyse, develop, and execute software requirements.

b. Adaptable

If the requirements are properly defined and do not change, heavyweight techniques for software estimating and project planning perform effectively. However, most projects' needs vary over time, necessitating the use of methods that can adapt to changing requirements [12]. Because changes are the rule, not the exception, agile methodologies allow for a quick reaction to requirement changes.

c. incremental / iterative

Developers only require short project cycles in agile development. A system that can be executed is not created at the end of a project. Instead, it is created quickly and provided to the customer for testing.

d. Collaborative

Agile development is collaborative because clients and developers collaborate often and in close communication. The client is supposed to be present at the construction site and to be actively participating in the construction process.

e. Simple and straightforward

The technique is simple to understand and adapt, as well as extensively documented.

f. People-oriented

Agile development techniques provide developers a lot of control. Developers make all the technical decisions, estimate the amount of work to be done, sign up for iteration tasks, and decide how much procedure to follow in a project. As a result, it is more concerned with people than with processes.

Despite their similarities, plan-oriented and agile techniques are not exactly competitors. Both have their own set of applications and ‘home turf.' They're utilized in a variety of projects, including the following:

• In big projects with predictable needs, plan-oriented approaches are used.an extremely important application domain

• Agile techniques for small teams and unpredictable settings generating applications that aren't important

The table below summarizes the spectrum of application (home ground) for agile and plan-driven methods:

Homeground area	Agile methods	Plan-driven methods
Requirements	Largely emergent , rapid change	Knowable early, largely stable
Refactoring	Inexpensive	Expensive
Primarily objective	Rapid value	High assurance
Customers	Dedicated, knowledgeable, collocated, collaborative, representative, and empowered	Access to knowledgeable, collaborative, representative, and empowered customers.
Development	Agile, knowledgeable, collocated, and collaborative	Plan-oriented, adequate skills, access to external knowledge
Architecture	Designed for current requirements	Designed for current and foreseeable requirements
Size	Smaller teams and products	Larger teams and products

Demonstrates one facet of these disparities. The two decreasing curves in this diagram indicate the possible damage to a project if not enough time and effort is put into planning. The two increasing curves depict the project's potential harm because of excessive planning time and effort.

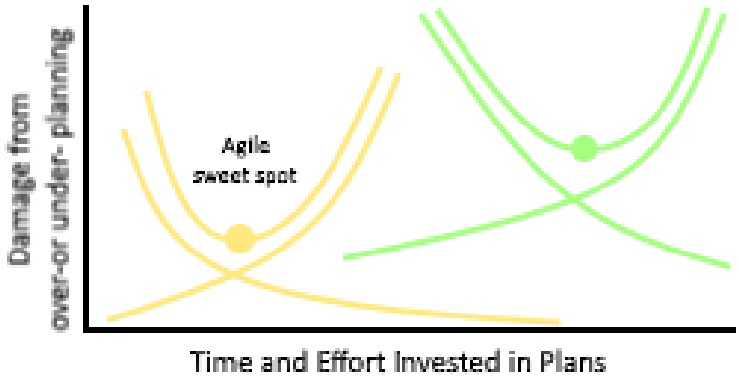


Figure 2: Using the Spiral Model and MBASE to Balance Discipline and Flexibility

The lines crossing on the left show a project where potential harm is low with under-planning and high with over-planning. This category includes a lot of commercial software, including Web services [13]. The lines crossing on the right represent a project for which potentially harm from under-planning is quite large, and for which much more planning would be required before damage from delays due to planning would occur. This category includes safety-critical software projects.

The Most Important Advantages of Agile Software Development

Agile software development is less document-oriented and more code-oriented than traditional software development. This, however, is not its most distinguishing feature, but rather a reflection of two more fundamental distinctions between the two styles. In this section, we will attempt to describe the key advantages of agile software development, which are difficult to achieve in traditional software development.

1. It's easier to plan and keep track of what's going on.
    a. Customer and developer cooperation is emphasized in agile software development. Developers can plan and manage projects more easily using this practice.

Furthermore, agile approaches often suggest iterations and provide a new software version every one to three months.

b. Early input is essential.

Agile techniques are particularly excellent at enabling communication between consumers and developers because of their tight collaboration [14]. Developers can obtain early feedback from consumers because of their habit of often providing functioning software.

c. Gives the customer value right away.

Again, because the customer is involved throughout the software development process, it has the potential to substantially enhance customer satisfaction, particularly if the client truly knows their requirements and is ready to participate.

d. Allows the creative process to take place.

Agile approaches are more concerned with people than with processes. To develop high-quality software, they depend on people's experience, competency, and direct collaboration rather than rigid, document-centric processes.

e. Adaptable to changes

Most of the software process is planned in detail over a longer time period using traditional approaches [15]. This works well if not much changes and the development team is familiar with both the application domain and software technology [16]. Agile approaches truly demonstrate their power in an application area where changes are frequent. The development is quite responsive to changes, thanks to an on-site customer who is always ready to offer answers to any clarifications developers want.

The Most Important Issues in Agile Software Development

While it appears that many software developments projects have been successful thanks to agile procedures, most of these success tales are based solely on personal experiences. We aim to highlight the key challenges and limitations surrounding agile methods in this section:

1. There is a scarcity of funding for large-scale development projects.

The control and communication mechanisms employed in agile techniques are suitable for small to medium-sized teams [17]. If a large team is involved in the software development project, less agile techniques will be required to address challenges that are unique to large management. Failure to offer enough structure and documentation.

2. It's difficult to create huge, sophisticated software

It is thought that refactoring may be used to eliminate the requirement to plan for change in agile software development. However, it may not operate effectively in big complicated systems since essential design features may be difficult to modify [18]. Models/designs play a critical role in such systems, whose functionality is so tightly linked and interwoven that incremental software development may not be viable.

3. There is a lack of support for reusability.

Extreme Programming, for example, focuses on creating software products that answer a specific problem [19]. While there appears to be a justification for using agile processes to create reusable artifacts, it is unclear how agile techniques can be properly applied.

4. Non-functional needs are difficult to manage.

When customers or users discuss what they want the system to accomplish, they rarely consider resources, maintainability, portability, security, or performance [20]. Some needs for the user interface or safety can be solicited and implemented throughout the development phase. Non-functional requirements should be handled more explicitly in agile techniques since they can influence the database, programming language, or operating system chosen.

5. Support for distributed development environments is limited.

Face-to-face communication may not be possible in development contexts when team members and consumers are physically separated. An overview of Agile Software Development Methodology and Its Relevance to Software Engineering Communication Agile procedures enhance communication in software engineering.

6. Agile development can have an impact on an organization's structure.

The decision-making authority in agile software development is distributed.

This approach to decision-making differs from that of many organizations [21]. In certain cases, programmers make a lot of important decisions, including those that are directly related to business, process, and system requirements, while their supervisors either accept it or not.

III. RESULT AND DISCUSSION

Using Agile Methodology in Software Development

The general procedure of Agile Methodology is like that of traditional software engineering methodology [4]. It begins with a requirement analysis, then moves on to the design, implementation, and testing phases. The specifics that occur in each phase, on the other hand, are extremely different since the focus is not on the model but on the quickly changing needs. This section explains how agile techniques differ from one another yet may be integrated into the software engineering process.

8. Analysis of Requirements

The ability to reach out to customers is critical in agile software development [22]. This provides the foundation for quick feedback and communication, resulting in a better knowledge of requirements and the development process. In the Extreme Programming approach, for example, it is considered that the customer is an ideal user representative who can correctly answer all questions and has the authority to make sound judgments.

All agile methodologies highlight that interacting with customers is the best way to obtain information for development and avoid misunderstandings. Customers and developers should try to speak with the person in charge if something is unclear or loosely stated to avoid indirect knowledge transfer.

However, there is still an issue, even though agile techniques are built on incorrect client participation throughout the whole development process. Agile techniques often require several clients with diverse backgrounds, but sometimes just one customer is available to work with the development team. As a result, not all the questions that arise can be answered in sufficient depth.

9. Modelling and design

Agile Modelling is one of the agile approaches that heavily incorporates modelling (AM). Although modelling is utilized in AM, it serves a distinct function. Models are used in AM to express understanding about a tiny component of a system under development, for example. Most of the models do not make it into the final system model since they are primarily throw-away models that are created on a whiteboard or paper and wiped after serving their function [23].

The agile technique is appropriate for small to medium-sized projects when used in this way. Because it integrates a full model of software in the design process, the plan-driven technique is better suited for big and complicated projects.

8. Incorporation

A method utilized in Extreme Programming is the most interesting and extreme method when it comes to implementation. It's known as "pair programming." In XP, programmers collaborate in pairs and as a group, using a basic design and rigorously tested code, constantly updating the design to meet current demands. Pair programming, on the other hand, does not imply that all code is written by two programmers working together on a single computer. Some code is better done alone, while others are best implemented in pairs.

10. Testing

Before writing the core code, developers in Extreme Programming perform unit tests for everything. They develop unit tests for all the common scenarios as well as any unique ones. Then they run all the unit tests, and if they discover a bug in the code, they write more unit tests to isolate the issue. After that, they repair the problem in the code and repeat all the tests.

11. Methodology Tools for Agile

The agile method may be used with a variety of technologies available on the market [7]. VersionOne is one of the programs. We can plan and manage our agile software development projects with the help of a tool like this. VersionOne is a software development project management company that specializes in the planning, management, and execution of quickly changing, unexpected software development projects [5], [24], [25]. It also offers numerous template projects in a web-based environment for Scrum, Extreme Programming, and DSDM techniques.

Another tool, known as TP, was created to make planning, tracking, and quality assurance tasks easier.

III. CONCLUSION

To summarize, the agile approach is a viable alternative to plan-driven methodologies (e.g., UML or Waterfall Model) in the creation of high-quality software. It's also ideal for small to medium-sized projects because developers don't generally focus on models as the major component of the final program. An overview of Agile Software Development Methodology and Its Relevance to Software Engineering Agile techniques feature a distinct approach to each software engineering phase that focuses on feedback and change. As a result, it is thought that this technique can increase programmers' productivity.

REFERENCE

[1] P. Abrahamsson, J. Warsta, M. T. Siponen, and J. Ronkainen, "New Directions on Agile Methods: A Comparative Analysis." [Online]. Available: http://dictionary.oed.com

[2] G. G. Wiguna, K. Darkun, and K. Sulistyadi, "SAST & AHP METHOD IN DETERMINING THE BEST STRATEGY OF OFFICE ERGONOMICS PROGRAM IMPROVEMENT TO PREVENT RISK OF MUSCULOSKELETAL DISORDERS AT XYZ COMPANY QATAR," *ADI Journal on Recent Innovation (AJRI)*, vol. 2, no. 1, pp. 186–193, Jan. 2020, doi: 10.34306/ajri.v2i1.28.

[3] P. Abrahamsson, O. Salo, J. Ronkainen, and J. Warsta, "Agile Software Development Methods: Review and Analysis." [Online]. Available: http://www.vtt.fi/inf/pdf/publications/2002/P478.pdf.

[4] S. F. Meilana, "Development Professionalism Strategy In Lecturers Improve The Competitiveness Of The Nation Through The Development Of Science And Technology," *ADI Journal on Recent Innovation (AJRI)*, vol. 2, no. 1September, pp. 222–226, Jan. 2020, doi: 10.34306/ajri.v2i1.64.

[5] F. W. Ramadhan, H. T. Sukmana, L. K. Oh, and L. K. Wardhani, "ANALYSIS OF WARGANET COMMENTS ON IT SERVICES IN MANDIRI BANK USING K-NEAREST NEIGHBOR (K-NN) ALGORITHM BASED ON ITSM CRITERIA," *ADI Journal on Recent Innovation (AJRI)*, vol. 1, no. 1, pp. 14–19, Sep. 2019, doi: 10.34306/ajri.v1i1.91.

[6] J. De, "What Is Agile Software Development? The Agile Ecosystem: Chaordic, Collaborative, and Streamlined An Agile Case Story," 2002. [Online]. Available: www.stsc.hill.af.mil5

[7] Z. Fauziah and D. Supriyanti, "Influence of Business Process Maturity Model as a Business Architecture Planning Proposal Parts the Business Process Assessment model, and 3 Parts for IT Application Readiness," *ADI Journal on Recent Innovation (AJRI)*, vol. 2, no. 2, pp. 2685–9106, 2021, doi: 10.34306/ajri.v2i2.288.

[8] J. Leonard, D. M. Damanik, and O. G. Amrikhasanah, "APPLICATION OF INFORMATION SESSION INFORMATION SYSTEM AS MEDIA SUBMISSION OF FINAL RESULTS COMPREHENSIVE SESSION," *ADI Journal on Recent Innovation (AJRI)*, vol. 1, no. 1, pp. 62–70, Sep. 2019, doi: 10.34306/ajri.v1i1.16.

[9] K. Prihandani, "Tinjauan kualitas pengembangan sistem informasi dengan metode agile."

[10] W. Fadilah, "PROTOTYPE OF E-OFFICE ADMINISTRATION LETTER SYSTEM IN GENERAL PART OF GOVERNMENT OF TANGERANG REGENCY," *ADI Journal on Recent Innovation (AJRI)*, vol. 2, no. 1, pp. 212–221, Jan. 2020, doi: 10.34306/ajri.v2i1.35.

[11] R. Rosdiana, P. Padeli, R. S. S. Handayani, and R. Alfian, "DESIGN AND DEVELOPMENT OF POPULATION SERVICE ADMINISTRATION SYSTEM WITH PIECES METHOD IN KEMIRI VILLAGE HEAD OFFICE BANTEN," *ADI Journal on Recent Innovation (AJRI)*, vol. 1, no. 1, pp. 33–45, Sep. 2019, doi: 10.34306/ajri.v1i1.98.

[12] J. Pramono, I. M. Sumartaha, and B. Purwantoro, "DESTINATION SUCCESSES FACTORS FOR MILLENNIAL TRAVELERS CASE STUDY OF TANAH LOT TEMPLE, TABANAN, BALI," *ADI Journal on Recent Innovation (AJRI)*, vol. 1, no. 2, pp. 136–146, Jan. 2020, doi: 10.34306/ajri.v1i2.44.

[13] S. Rahayu, H. Haris, and Y. Candrah, "WEB-BASED CLASSIFICATION INFORMATION SYSTEM FOR WEB-BASED AT PT. SINTECH BERKAH ABADI," *ADI Journal on Recent Innovation (AJRI)*, vol. 2, no. 2, pp. 98–105, May 2020, doi: 10.34306/ajri.v2i2.62.

[14] K. Sulistyadi, S. Ramli, and S. Uddin, "Factors Influencing MCI Preparedness of Paramedic in XYZ Industrial City," *ADI Journal on Recent Innovation (AJRI)*, vol. 2, no. 2, pp. 223–231, Feb. 2021, doi: 10.34306/ajri.v2i2.24.

[15] I. B. Rahardja and M. Masnia, "THE OPTIMIZATION OF CAPACITY BOILER EFFICIENCY 26 TONS/HOURS WITH FUEL ALUMINATION AND STATISTICAL PRODUCT AND SERVICE SOLUTIONS (SPSS) ANALYSIS," *ADI Journal on Recent Innovation (AJRI)*, vol. 2, no. 2, pp. 127–187, May 2020, doi: 10.34306/ajri.v2i2.70.

[16] M. Danaci, F. Koylu, and Z. A. B Al-Sumaidaee, "Identification of Dynamic Models by Using Metaheuristic Algorithms," *ADI Journal on Recent Innovation (AJRI)*, vol. 3, no. 1, pp. 36–48, 2021, doi: 10.34306/ajri.v3i1.492.

[17] T. Dingsøyr and N. B. Moe, "Research challenges in large-scale agile software development," *ACM SIGSOFT Software Engineering Notes*, vol. 38, no. 5, pp. 38–39, Aug. 2013, doi: 10.1145/2507288.2507322.

[18] I. A. M. Gayatri and I. N. Suriata, "CHALLENGES AND OPPORTUNITIES OF BLIND MASSEURS IN INCREASING COMPETENCY THROUGH IMPLEMENTATION BUSINESS STANDARDS OF MASSAGE PARLOR," *ADI Journal on Recent Innovation (AJRI)*, vol. 1, no. 2, pp. 107–120, Jan. 2020, doi: 10.34306/ajri.v1i2.40.

[19] A. Fatoni and D. Dwi, "RANCANG BANGUN SISTEM EXTREME PROGRAMMING SEBAGAI METODOLOGI PENGEMBANGAN SISTEM," vol. 3, no. 1, 2016, [Online]. Available: http://developdottxt.

[20] M. Siponen, R. Baskerville, and T. Kuivalainen, "Integrating Security into Agile Development Methods," 2005.

[21] P. W. Novika and G. Batam, " Building a Professionalism a Lecturer Strengthening the Competitiveness of the Nation," *ADI Journal on Recent Innovation*, doi: 10.34306/ajri.v2i2%20Maret.54.

[22] R. Rojali and D. I. Sari, "RELATIONSHIP OF INDIVIDUAL CHARACTERISTICS, PHYSICAL HOME ENVIRONMENT AND BEHAVIOR WITH THE INCIDENCE OF PULMONARY TB IN CIJORO PASIR VILLAGE, MUARA VILLAGE EAST CIUJUNG AND WEST RANGKASBITUNG VILLAGE, RANGKASBITUNG SUBDISTRICT, LEBAK REGENCY 2019," *ADI Journal on Recent Innovation (AJRI)*, vol. 1, no. 2, pp. 167–179, Jan. 2020, doi: 10.34306/ajri.v1i2.36.

[23] D. N. Khasanah, S. Wulandari, and R. Dwi, "WEB-BASED LOGISTIC DEMAND INFORMATION SYSTEM DESIGN AT RAHARJA UNIVERSITY," *ADI Journal on Recent Innovation (AJRI)*, vol. 1, no. 1, pp. 79–84, Sep. 2019, doi: 10.34306/ajri.v1i1.19.

[24] N. L. Diary, L. P. Perdanawati, A. M. Adiandari, and B. A. Wijaya, "ANALYSIS OF THE EFFECT OF LEADERSHIP AND ORGANIZATIONAL CULTURE ON ORGANIZATIONAL CITIZENSHIP BEHAVIOR WITH JOB SATISFACTION AS AN INTERVENING VARIABLE AT UBUD WANA RESORT," *GIANYAR. ADI Journal on Recent Innovation (AJRI)*, vol. 1, no. 2, pp. 121–129, 2020, doi: 10.34306/ajri.v1i2.42.

[25] I. M. Kartika, K. Wahyudi, I. M. A. Suwandana, and I. B. Suteja, "ANALYSIS OF MARKETING STRATEGY OF BRAND NONMIN DRINKING WATER OF OXYGEN IN PT TAMANBALI TIRTA BANGLI," *ADI Journal on Recent Innovation (AJRI)*, vol. 1, no. 2, pp. 147–157, Jan. 2020, doi: 10.34306/ajri.v1i2.45.