The Challenges of Software Development: Waterfall and Agile

Mohamed Ben-Zahia, Ali Aburas, and Miloud Ghuwar

Computer Science Department-University of Tripoli

Tripoli-Libya

m.ben-zahia@uot.edu.ly

الملخص

تلعب البرمجيات دورًا حيويًا في حياتنا اليومية، ومع تطور تكنولوجيا المعلومات، اصبحت مشاريع البرمجيات أكثر تعقيدًا وأصبح استخدام منهجيات الشلّال وآجايل في اعداد البرمجيات أكثر شيوعا. ومع ذلك، فهذه المنهجيات ليست مناسبة لكل مشاريع البرمجيات. ان استخدام منهجية الشلّال او آجايل كنموذج لإعداد البرمجيات لا يكفي والحل لهذه المعضلة هو تبني منهجية جديدة تسمى المنهجية الهجينة التي تستفيد من ميزات وقوة كلتا المنهجيتين.

تقدم هذه الورقة معايير محددة لاختيار المنهجية المناسبة اعتمادًا على مجموعة عوامل محددة، وعلاوة على ذلك، فالورقة تقدم ايضا تحسين لنموذج الشلال لإعداد البرمجيات لمعالجة اية نواقص او عيوب فيه.

الكلمات المفتاحية: تخطيط الاتجاه، الشلال، الهجين، اجايل، سكرم

Abstract

Software plays a vital role in our daily life, and with information technology evolving, software projects are getting more complex. Waterfall and agile are commonly used for software development. However, they are not sufficient for every software project.

A pure waterfall or agile as a software development model is not enough. The solution to this dilemma is adopting a hybrid methodology that takes advantage of the strength of both methodologies. This research paper uses criteria for selecting an appropriate methodology depending on some factors. Moreover, enhancing the waterfall model is presented to improve its deficiency.

Keywords: Plan Driven, Waterfall, Hybrid, Agile, Scrum

Introduction

The waterfall model (plan-driven) of the traditional methodology used as a lifecycle for software development is well-known and straightforward. It is based on dependent phases of the software development life cycle (SDLC). Each phase depends on the completion of the previous phase. However, it has some problems due to today's dynamic market and the need to deliver software products fast. This is true, especially in the software industry, where requirements change rapidly due to changes in services, laws, and technologies. Another problem with the waterfall is that it requires clear documented requirements before going to the next phase of design. The 2020 Standish Group Chaos Study shows that about 70% of any failing waterfall project is attributed to requirements [1]. In addition, the waterfall lacks the user's participation during the SDLC, and this participation and involvement are becoming the main factor to the success of any software project.

The agile methodology comes into existence to solve some of the software development problems of the waterfall model, such as late delivery time and incomplete requirements. Unlike waterfall, agile models like Scrum usually divide the project into small, iterative increments. Each iterative increment produces a new working feature. The 2016 Standish Group Chaos Study shows that agile projects are more successful than waterfall projects and have fewer challenges and failures [2]. However, the Scrum faces some problems and challenges. For example, Scrum meetings are often skipped or postponed due to some circumstances, and this skipping would negatively affect the project. Another problem with Scrum is the competence of team members is hard to obtain. To overcome the previous problems of both methodologies, researchers and practitioners devised another methodology, the hybrid that combines the strengths of both methodologies.

The aim of this research paper is to offer two solutions. First, for those who choose the waterfall model as the preferred development model, an enhanced waterfall is provided to overcome some of the waterfall's problems. The second solution is the hybrid model that combines waterfall and agile models' strengths. The selection of any methodology would depend on some factors such as time of delivery and team size.

The paper is organized as follows: Section 2 contains a literature review, introducing the traditional (plan-based), agile, and hybrid methodologies. Section 3, methodologies and models. Section 4 presents the factors for selecting a software development model. In section 5, the solutions to software development challenges are presented as the enhancement of the waterfall model and the hybrid approach as another solution.

Literature Review

Software is becoming the backbone of any IT project. Moreover, traditional (plan-driven) and agile methodologies are considered the two famous approaches used for software development. Each of these methodologies has pros and cons. The plan-driven approach has many software development models, such as waterfall and spiral, whereas the Agile approach has models such as Scrum and XP.

The waterfall model was introduced by Winston Royce in 1970 [3]. It is a sequential process using a fixed plan and has detailed documentation. The model consists of five separate phases: requirement, design, implementation, testing, and maintenance. It faces some problems such as the absence of a representative of the client throughout the process groups, and that the requirements must be known at the initial stages of the project [4]. The waterfall method has fallen short in today's dynamic marketplace owing to its lack of flexibility and speed. This is true, especially in the software industry where requirements change rapidly and a turn-around time of several months will simply not cut it [5]. The model is still in use, and some researchers are even convinced that it will be around for a much longer period. However, some modifications are needed to enhance the model.

In 2001 the Agile methodology was introduced to solve some of the problems of the waterfall model. The term "Agile" came into world with agile manifesto in 2001 at Utah, USA to discuss software development models [6]. This methodology focuses on iteration cycles, early delivery, and reacting positively to changing requirements during the iteration cycle. The Agile software development methodology is one of the simplest and most effective process to turn a vision for a business need into software solutions. This methodology is used to describe software development approach that employs continual planning, learning, improvement, team collaboration, evolutionary development, and early delivery. It encourages flexible responses to change [7].

In 2017, almost 87% software houses followed agile to accomplish their software development projects successfully [8]. The Agile method serves best in huge dynamic business. The method has become popular and project managers and project engineers started creating quality software with a less cost [9]. In projects utilizing agile methodology, some challenges are encountered such as uncertainty in initial requirements, and documentation is minimal and generally in the form of user stories which can be considered as light weight requirements [10].

Agile methodology faces other challenges, such as a lack of dedicated cross functional teams, and a lack of skilled product owners from the business side [11]. The main reasons agile project failures are: organizational culture balanced with agile values and less understanding of the required wide organizational change [12]. There are many different models or frameworks that come under agile methodology such as Scrum, XP, and Kanban.

Scrum is considered a framework for software project development and management. It is a continuous iteration method that can be done to improve the product life cycle [13]. The Scrum development process consists of a series of iterative sprint processes. A sprint is a series of development activities in a limited time, including analysis, design, coding, and testing [14]. It is an excellent agile methodology to increase software product accurately, correctly, and quickly give all team members new management responsibilities [14]. Scrum is a simple, easy to understand and a good method to manage processes and sustains critical products to develop software that satisfies business requirements [15]. However, Scrum faces some obstacles and problems that are needed to be dealt with.

Traditional and agile methodologies have weaknesses and strengths. For example, the shortcomings of agile methodology are lack of necessary documentation, and difficulty of measuring the progress of the entire project [16]. The hybrid approach in product development combines the strengths of agile and plan-driven approaches [17]. Practitioners combine various approaches to software creation to improve productivity, product quality, and adoptability of the process to react to change [18].

Methodologies and Models

When developing software, a developer needs to choose a methodology from the two well-known methodologies. These methodologies are plan-driven and agile. The choice of one of these methodologies would depend on some factors, such as system size, delivery time, and documentation.

The waterfall model of the traditional methodology is still popular and used. However, it faces some problems and shortcomings. Another methodology, called agile, has been introduced to solve some of the previous problems. This methodology consists of many development models such as Scrum and XP.

The waterfall model is used as a life cycle for software development. A software project consists of sequential phases such as requirements, design, implementation, testing, and maintenance. However, there is no going back to previous phases, and the requirements are frozen till the end of the project. The advantages of the waterfall model are as follows:

- Easy and famous.
- The developer knows where s/he is in the life cycle.
- It has detailed documentation to be used in the maintenance phase.

However, it has some disadvantages as follows:

- Presumes the requirements to be completed at the beginning of the project.
- Does not adopt changes of requirements.
- Takes a long time to complete a project (months or even years).

Scrum Model

The Scrum model, Figure 1, is the most popular model in the Agile methodology. The model consists of a set of iterations called sprints. The life cycle of each sprint consists of tasks: plan, design, code, test, and delivery of some features. And after finishing a sprint, new features are added to be coded and tested in the next sprint. The model encourages collaboration between customers and the team to reduce the errors that come because of inefficient communication. The communication also provides user satisfaction about the software product.

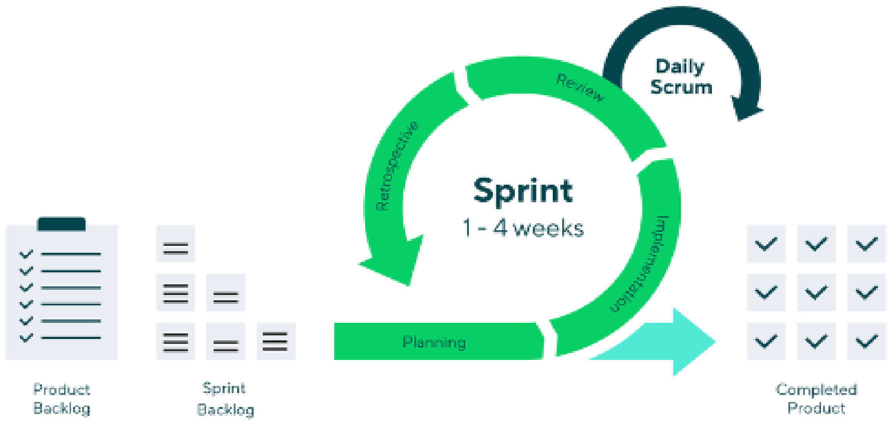

Figure 1: Agile software development with Scrum.

The Scrum model is a self-organizing process in that decisions are made by the whole team. Moreover, the team of the Scrum model is cross-functioning, where each member takes a feature from planning to delivery of that feature. The Scrum master role acts only as a facilitator or a coach to remove any obstacles encountered during the sprint. The product owner is a representative of the user side. He guides the team to a successful end. The typical scrum team is 5-9. At the end of each sprint, a sprint review meeting is held to show what a team member accomplished during the sprint. Another meeting called retrospective is held to discuss any positive or negative circumstances that happened so to be good lessons for the next sprints in the future. The model has the following advantages:

- Short development cycle.

- Accepts changes in requirements.
- High customer satisfaction.
- Involves user in the development team.

However, the Scrum model has the following disadvantages:

- Can be successful only with talented developers.
- Not appropriate for large-scale projects.
- Needs full commitment of team members.
- Lacks documentation which is needed in maintenance phase

Factors For Selecting Software Development Model

There are many software development models used in the software market. However, each of them has a chance to either succeed or fail in software projects. The selection of the unstable model might lead to the failure of the project.

There are many factors that are needed to consider when selecting the appropriate development model, namely, waterfall, agile or hybrid. The dilemma and challenges of software development are still in existence. Software companies use Agile development approach because of its lightweight nature to solve some problems of the plan-driven approach. Moreover, the Agile approach is designed for small size teams, continuous delivery in a short period of time, close communication with team members and collaboration and feedback from the customer side.

Table 1: Factors that influence the choice of the proper model

Factor	Waterfall	Agile	Hybrid
User Involvement	**Not Required**	**Required**	**Required**
Budget	**Fixed**	**Variable**	**fixed**
Team Size	**Large**	**Small**	**Medium**
Technology	**Low**	**High**	**High**
Requirement	**Clear, Complete**	**Unclear, Incomplete**	**Unclear, Incomplete**
Change of Requirements	**minimal**	**High**	**High**
Documentation	**Detailed**	**Light**	**detailed**
Delivery Time	**Late**	**Early**	**Early**
Developer Experience	**Low**	**High**	**High**
Plan	**General plan**	**Short plan**	**General plan**

The Agile methodology faces some challenges, such as a lack of essential documents, and measuring the progress of the entire project is difficult due to the absence of a general plan for the whole project. And for this reason, companies often merge Agile and traditional methods to form the hybrid method. The Agile methodology is iterative, incremental, and self-organizing.

On the other hand, the waterfall model is one of the most used development models. It is a linear structure. The model has the advantage of being easy and simple. And its detailed documentation is valuable and needed for the maintenance phase. However, users would not see the final software product till the end of the process, and this would add more risk to the project.

Scientists and developers have started to adopt the hybrid approach. However, there is still confusion about the use of hybrid and how to integrate the two methodologies effectively. Though using the iterative and incremental would ease the rigidity of plan-driven approach, future research will focus on the demand-oriented combination of Agile and plan-driven approaches.

Solution

Software development is a challenging task and faces many obstacles. The software development life cycle models are used to achieve high-quality projects and meet customer expectations. However, the chosen model among many recognized SDLC models is tedious. The following sections offer some improvements to the existing waterfall model.

Enhanced Waterfall Model

The waterfall model, Figure 2, is the most used and oldest model for software development. However, many criticisms are becoming apparent due to the bad performance toward the increasing demands from the customer side. These demands are related to changing technology, productivity, services, and laws. The need to release software products in a short time is also a demand.

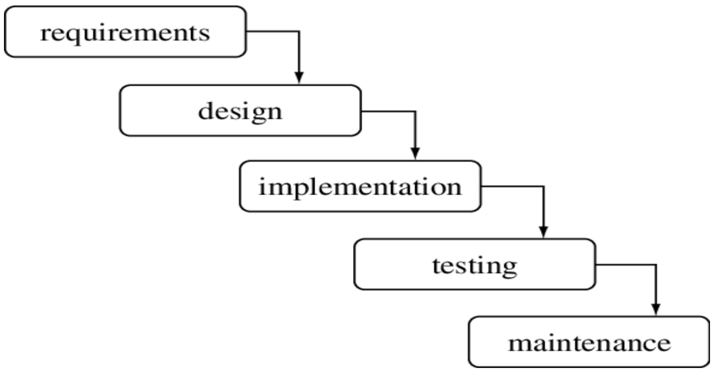

Figure 2: A standard waterfall model for software development.

There are many failing projects that used the waterfall model. The cure to these problems and the advantages of waterfall becomes a necessity. The following are some of the Improvements to the waterfall model. These improvements aim to develop an ideal model for most software systems and be competitive with other methodologies such as Agile.

Requirements

Some elicitation and molding of requirements involve natural language, which is vague and error-prone. This trend leads to ambiguous and unclear requirements, and the requirements are considered costly and would affect the design phase.

Copyright © ISTJ
حقوق الطبع محفوظة
للمجلة الدولية للعلوم والتقنية

The solution is to use graphic tools to specify requirements such as use case diagrams, data flow diagrams, and decision trees. Another solution is to involve the customer in most life cycle phases of the project. Validation is considered another technique to obtain high-quality requirements and should come as early as possible during the execution of the software development project. The approach of software requirement engineering is another remedy for requirements faults. Moreover, negotiation and prioritization of requirements with the customers are techniques for effective and efficient requirements.

Changes of Requirements
As mentioned earlier, the traditional waterfall model resists changes of requirements during the life cycle phase. These changes happen because of new services, laws, and technologies. Failing to deal with the changes is a sign of failure. The solution is to let the customer be a team member of the development to add feedback from the user side. The classical waterfall should be converted to an iterative and incremental model. Big systems also should be decomposed into subsystems and components, and each component should be developed and delivered to the customer gradually and as soon as possible. Moreover, returning to the previous phase should be accepted.

Delivery Time
Developing big systems using the traditional waterfall model takes a long time (months or years). The delivery time becomes a factor in the success or failure of any software product. There are ways to decompose big systems into chunks or components, and in this way, the customer can see a working product in a short time. The MOSCOW rules should be used to what requirements should be developed first. The letter M means the requirements MUST be done first, then other requirements SHOULD be done after the MUST. The C letter for COULD, and the W letter for WOULD. The software is delivered in increments (components) one after another to reduce the time of development.

Structure of the Life Cycle
The life cycle of the waterfall model consists of only five phases: requirements, design, implementation, testing, and maintenance. These phases have shortcomings in terms of quality and quantity. For example, the requirements phase produces vague and incomplete requirements and should be replaced with the requirements engineering phase. Another example, the software product, after testing, should be considered a beta version and treated as an early version of the final product. This beta version is an excellent development for the first release of the required software. And with feedback from the user, the requirements become clearer.

There are several essential phases that are missing in the waterfall life cycle, which are considered vital in the software development process. The proposed missing phases: communication, planning, deployment, conversion, and operation are more effective in successful software projects. Moreover, the deployment phase should include the activities: installation, data conversion, and training. Big software projects also should be divided into subprojects to ease the development process.

Iterative Waterfall

The iterative waterfall model, Figure 3, is proposed to solve some of the shortcomings of the waterfall model. The model starts with the feasibility phase and ends with the maintenance phase. The phases of the life cycle are: feasibility study, requirements analysis, design, coding, testing, and maintenance. Moreover, the model consists of repeated cycles throughout the process. The cycles are repeated, and the incremental happens in a short period of time. And each cycle involves some set of features.

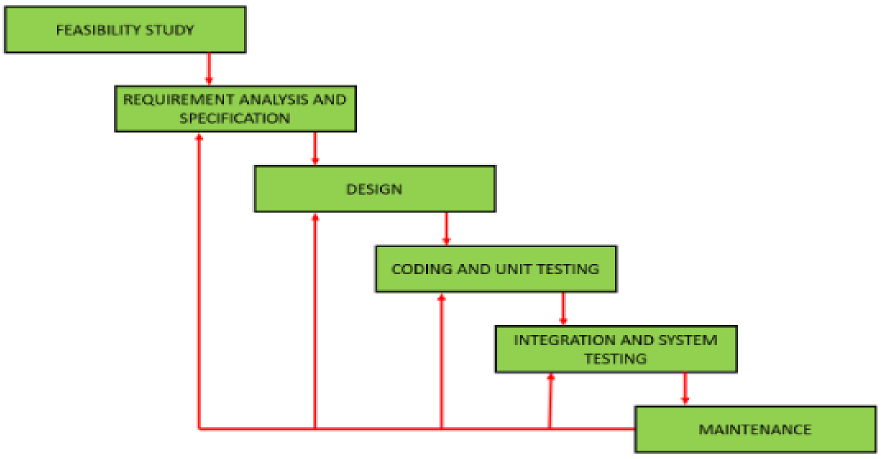
FEASIBILITY STUDY
REQUIREMENT ANALYSIS AND SPECIFICATION
DESIGN
CODING AND UNIT TESTING
INTEGRATION AND SYSTEM TESTING
MAINTENANCE

Figure 3: Iterative Waterfall Model.

Hybrid Methodology

The evolution of software is increasing day by day, and combining traditional and agile methodologies is a trend. The goal of merging the advantages of the two approaches is to produce a new software development approach known as a hybrid.

One of the reasons for adopting the hybrid methodology is the changing of requirements due to the change in services, laws, and technology. The time pressure to market is another reason.

The hybrid methodology is used to enrich the waterfall model with agile principles and vice versa. And get benefits from both traditional and agile approaches. Recent research contends that both agile and waterfall approaches can be valid, depending on project complexities and risks. And a combined approach, hybrid, could prove optimal.

The hybrid methodology uses the waterfall model as a base, but the implementation is done in an iterative manner (2 weeks cycle), and a demo is done in each cycle or iteration. Each developer is assigned to a chunk where design, coding, testing, and delivery to the customer as working software. The software product is gradually delivered.

We propose a Hybrid model that merges Waterfall and Agile approaches. The waterfall contributes the requirement, planning and maintenance phases. whereas, the Agile contributes with series of iterations where each iteration consists of design, coding, , testing and delivery tasks. The finished chunk would be delivered to the customer as working software in a short period of time.

Conclusion

There are many methodologies used to develop software. The waterfall is the most widely used model, however, many problems are reported related to this model. Some of these problems are related to requirements, delivery time, and user involvement. Agile methodology comes into existence to solve some of the previous problems. The Scrum model is considered one of the most used models of agile methodology, which was initiated in 2001.

There is no optimal development model when it comes to selecting the methodology for a software project. However, some factors need to be considered when selecting a certain methodology. These factors include delivery time, team size and change of requirements. Moreover, Agile is preferred when continuous delivery is essential, requirements are not well-defined, and delivery time to market of the software as portions. The Hybrid approach which are proposed in this paper combines the strengths of both waterfall and agile. It becomes more popular now and in the coming years.

References

[1] Standish group – chaos 2020: Beyond infinity. 2020.

[2] Theo Thesing, Carsten Feldmann, and Martin Burchardt. Agile versus waterfall project management: Decision model for selecting the appropriate approach to a project. Procedia Computer Science, 181:746–756, 01 2021.

[3] Harkirat Kaur Aroral. Waterfall process operations in the fast-paced world: Project management exploratory analysis.

[4] Joyce Koi-Akrofi, Akwetey Henry Matey, and Godfred Koi-Akrofi. Understanding the characteristics, benefits and challenges of agile it project management: A literature based perspective. 10:25–44, 09 2019.

[5] https://www.digite.com/blog/waterfall-vs-agile-project-management/.

[6] Kasim Khusanov and .A Mukhamadjonov. Agile software development. 2017:25–28, 08 2017.

[7] https://www.guru99.com/agile-scrum-extreme-testing.html.

[8] Rashina Hoda, Norsaremah Salleh, and John Grundy. The rise and evolution of agile software development. IEEE Software, PP:1–1, 07 2018.

[9] Fawad Ghafoor, Ibrar Ali Shah, and Nasir Rashid. Issues in adopting agile methodologies in global and local software development: a systematic literature review protocol with preliminary results. International Journal of Computer Applications, 160(7), 2017.

[10] Tuna Hacaloˇglu and Onur Demir¨ors. Challenges of using software size in agile software development: A systematic literature review. Academic Papers at IWSM Mensura 2018, 2018.

[11] Amna Batool, Morshed Chowdhury, and Aneeqa Chowdhury. A survey of key challenges of adopting agile in global software development: A case study with malaysia perspective. International Journal of Industrial and Manufacturing Engineering, 13(10):1387–1391, 2019.

[12] Gloria Miller. Agile problems, challenges, failures. 10 2013.

[13] Kresna Dwi Prasetya, Devriady Pratama, et al. Effectiveness analysis of distributed scrum model compared to waterfall approach in third-party application development. Procedia Computer Science, 179:103–111, 2021.

[14] Muhamad Ma'arif, Siti Mariam Shahar, Mohd Fikri Hafifi Yusof, and Nurhizam Mohd Satar. The challenges of implementing agile scrum in information systems project. Journal of Advanced Research in Dynamical and Control Systems, pages 2357–2363, 09 2018.

[15] Valpadasu Hema, Sravanthi Thota, S Naresh Kumar, Ch Padmaja, C Bala Rama Krishna, and K Mahender. Scrum: An effective software development agile tool. In IOP Conference Series: Materials Science and Engineering, volume 981, page 022060. IOP Publishing, 2020.

[16] Kristin Goevert, Jonas Heimicke, Udo Lindemann, and Albert Albers. Interview study on the agile development of mechatronic systems. In Proceedings of the Design Society: International Conference on Engineering Design, volume 1, pages 2287–2296. Cambridge University Press, 2019.

[17] J Heimicke, R Chen, and A Albers. Agile meets plan-driven–hybrid approaches in product development: A systematic literature review. In Proceedings of the Design Society: DESIGN Conference, volume 1, pages 577– 586. Cambridge University Press, 2020.

[18] Rafa l W lodarski, Jean-Remy Falleri, and Corinne Parv´ery. Assessment of a hybrid software development process for student projects: a controlled experiment. In 2021 IEEE/ACM 43rd International Conference on Software Engineering: Software Engineering Education and Training (ICSE-SEET), pages 289–299. IEEE, 2021.