International Journal of Social Science Exceptional Research

# Comparative Analysis of Agile and Waterfall Methodologies in Modern Software Development

**Hareesh Kumar Rapolu**
Independent Researcher, USA

* Corresponding Author: **Hareesh Kumar Rapolu**

**Article Info**

**ISSN (online):** 2583-8261
**Volume:** 03
**Issue:** 01
**January-February** 2024
**Received:** 20-12-2023
**Accepted:** 23-01-2024
**Page No:** 172-176

**Abstract**
Agile and waterfall methodologies are prominent techniques used in the software industry. Both deliver projects from different perspectives. This paper presents a deep analysis of both methodologies, their strengths, weaknesses, challenges, and roles in different projects. Many software industries used traditional methods in the earlier days of software development. The use of traditional methods has various limitations and unstructured code. The latest techniques and procedures have recently made development simpler, faster, and more efficient. The analysis of these traditional and modern methods of the software industry is explained in detail in this paper.

**DOI:** https://doi.org/10.54660/IJSSER.2024.3.1.172-176

**Keywords:** Microservices, Spring Security, Cloud, API, Authentication and Authorization, JWT token, Encryption, Infrastructure, Spring Annotations

## 1. Introduction

Software development methods provide a guideline or framework to build software in that manner and deliver the project on time. All stages that the software product goes through during both production and customer use are called the software development life cycle1. The waterfall method is a sequential, linear, or straightforward approach invented in the 1970s. It is a detailed step-by-step procedure that goes from requirement gathering to maintenance. It is a popular approach in the initial stages of software development. There are various methodologies other than waterfall and Agile, but these methods became popular due to their performance. The waterfall is still a commonly adopted solution for many businesses due to its structured phase, but it still holds a few drawbacks like client collaboration, no-time delivery, etc. In contrast, more than 70% of businesses adopt agile methodology because it has good client collaboration, faster task engagement, and timely delivery. Agile frameworks include Kanban, Scrum and Extreme Programming, Dynamic Systems development method, FDD, etc. These agile methods are taking over traditional methods due to their adaptability.

## 2. Background and related work

The waterfall model is the earliest recognized software method introduced by Dr. Winston in 1970. It is a linear or sequential approach with a step-by-step process performed one after another following a one-directional flow. He also gave the risks and limitations of this sequential process. The waterfall model has a failure rate of 30% and lengthens the development process. Initially, this model is used to design large-scale industries following detailed procedures. This model has seven steps, and one should start moving when the previous step is completed and reviewed. From the limitations and drawbacks of the waterfall model, a new methodology emerged called the agile methodology. The slow delivery process, demanding faster solutions, and the linear approach of the waterfall model became the roots of the Agile model. A group of people, Kent Beck, Martin Fowler, Ward Cunningham, Ken Schwaber, and others, published the Agile Manifesto. Scrum was invented in 1995, DSDM (Dynamic Systems Development Method) in 1994(Dynamic Systems Development Method), XP by Kent Beck (Extreme Programming) in 1996, and FDD by Jeff De Luca and Peter Coad in 1997. Scrum became the widely used framework for Agile software.

development. This paper will further explain the detailed view and comparison of all the models.

### 3. Overview of the waterfall model

It is a widely used software strategy that is the first and most popular for linear or sequential development flow. It is a document-driven approach working on a clear set of goals. Each phase must be done completely and reviewed before the next stage begins. The waterfall model consists of Seven phases or steps. These stages are considered the earliest model, yet they are viewed as vital techniques in the conventional approach to software development2.

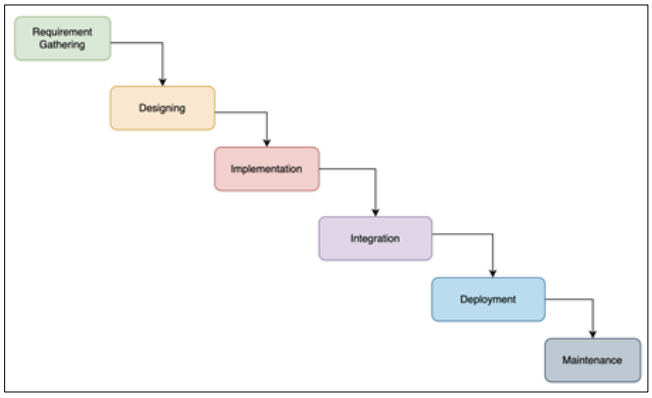


**Fig 1:** Waterfall Model

**A) Requirement gathering:** Gathering requirements is the initial stage of any SDLC model. It deals with requirements like price and analyses business-related aspects. Here, all nonfunctional and functional requirements from stakeholders are estimated. The requirements should mandatorily specify required resources, assign tasks to the various teams, and detail the goals.

**B) Designing:** This phase is started only when the requirement-gathering section is completely understood and executed. In this phase, the architectural design is based on the requirements from the previous phase. System design documents, such as high-level design, model, data flow, data structure, and various information related to project design and programming languages, are performed here. Hence, we can get an overall architecture in a diagram, which is a pictorial presentation of project development.

**C) Implementation:** Implementation is also called the coding phase. The design model is translated to coding for execution. Developers will develop or build the code according to the design using a suitable programming language. The development environment and testing environment are synchronized with the same protocol. The project manager monitors the progress of teams and evaluates it. If something comes up in the initial development, debugging starts, and the code is re-uploaded to the test environment to lessen the bugs.

**D) Integration and Testing:** Developmental errors are identified at this stage. Initially, individual components are integrated into a complete system. Test cases are developed based on the system design documents. Testing documents are prepared with the input data and expected output for each test case. Various levels of testing are performed, including unit testing, integration testing, performance testing, security testing, and acceptance testing. If any defects are found, the bugs are discussed with the development team and resolved before proceeding to the next phase.

**E) Deployment:** The implemented code is deployed for the end users whenever the testing process is successful. This phase ensures that the deployed code runs correctly without bugs. Before deploying code, a production environment is prepared for smooth transitions with timelines. All the components, such as databases, configurations, and data, are migrated to the production server to prepare the live production.

**F) Maintenance:** After production, the deployed software is monitored for its performance, and for any modifications required, the software will be redeployed. This phase ensures that all deliverables are met before handing over to the client.

### 4. Challenges in waterfall model

The waterfall model is the finest approach of Software development, but there are many limitations, as described below:

**A) Feedback:** Performance is only seen at the end of the project, so if expectations are not met, there will be a complete rework, and one should begin from the first phase of development.

**B) Risk:** The waterfall model is a sequential approach. If there is any error in one phase, the next phase will continue to have the same error, which is a risk of failure of the entire application.

**C) Time and cost management:** If the bugs are identified late, it is hard to fix them immediately, and all the steps must be followed, which leads to an increase in time and cost.

**D) Linear flow:** The model itself is a sequential or linear flow. Hence, each phase is completely reviewed before going to the next phase. Waiting for the approvals leads to increased development time and cannot overlap between phases.

**E) Delivery:** This Linear flow of the model cannot provide faster deliveries, and stakeholders cannot visualize the product and confirm the changes.

### 5. Overview of agile methodology

The main goal of agile methodology is to overcome the limitations of the waterfall model. Agile focuses on flexibility, adaptability, and continuous software development delivery. As discussed earlier in this paper, the Agile Manifesto provides various software development principles. This paper will discuss extreme programming, scrum, and DSDM in detail. These methods have become more popular in agile methodologies. At the conclusion of every iteration, the project stakeholders, alongside the customer representative, assess the progress and reassess priorities to enhance the return on investment (ROI) and confirm alignment with customer needs and company objectives. Hence, agile is considered as a customer-centric approach3.

**A) XP:** Extreme Programming originated from the challenges posed by traditional software methodologies. It emphasizes continuous integration, test-driven development, and high-quality code with rapid deliverables4.

**1) Lifecycle:**

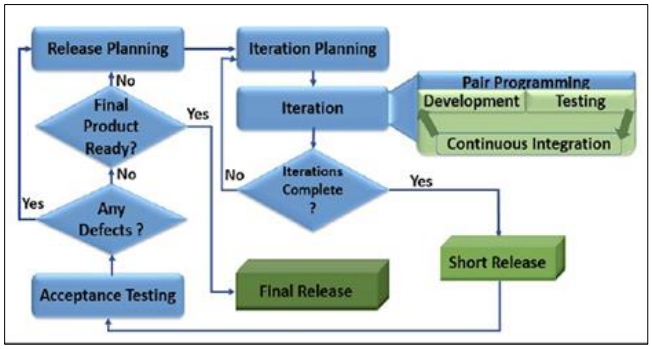


**Fig 2:** Lifecycle of XP

**2) Working procedure:**

- A planning phase is conducted where the customers provide a deep description of what they need, and then the team collaborates for release planning based on time and risk.
- XP divides conventional software into smaller divisions. Each phase is divided into a user story, which is then broken down into tasks and assigned to respective team members.
- The programmers estimate the effort for each task in hours. This model uses CRC cards, which are grouped as separate cards for each model in the system.
- This model uses the pair programming strategy, in which two developers work on the same task to develop and review the code, improving its quality.
- XP suggests a better testing environment with a TDD approach where the test cases are continuously written and executed. These test cases are written before the code is developed. Collective code ownership is developed, and programmers have the right to modify the code and maintain easy readability.
- Testing can be done in multiple steps, such as unit testing (done by developers), acceptance testing, integration testing, and frequent testing to detect bugs early.
- After the testing is complete, a customer representative will review it and work with development to make any necessary adjustments for the release.

**A) Scrum:**

Scrum is a well-used Agile framework and sprint-based methodology for software development. It consists of Roles, Artifacts, and events. In Scrum, team members self-organize tasks to reach a common goal. Scrum maintains quality and productivity and adapts to changing requirements.

**1) Key features:**

- Scrum is a lightweight framework that is simple to understand.
- It is a structured framework which emphasizes self-organization
- It allows team members to work collaboratively, which promotes creativity.

**2) Roles and Responsibilities:**

Scrum follows the incremental process and defines roles and responsibilities between team members.

- Product owner: A product owner is responsible for the design, management, control, creation, and prioritization of the Product Backlog. He owns a right to accept or reject the work delivered by the team. A product owner prioritizes the risks and tasks based on business value. He decides what should be included in the sprint and decides on product backlog.
- Scrum Master: The Scrum Master maintains the team and serves as a coach to ensure the progress of the tasks according to plan. He checks whether the team adheres to Scrum rules. The Scrum Master is responsible for communicating the team's progress to management and listens to his team for any impediments, helping team members by providing necessary resources. He ensures the team remains focused on continuous improvement.
- Developer: The Developer is responsible for completing the tasks according to the task effort. The developers achieve sprint goals and will be responsible for any backlogs. Every team will have at least 5 developers.

**3) Lifecycle:**

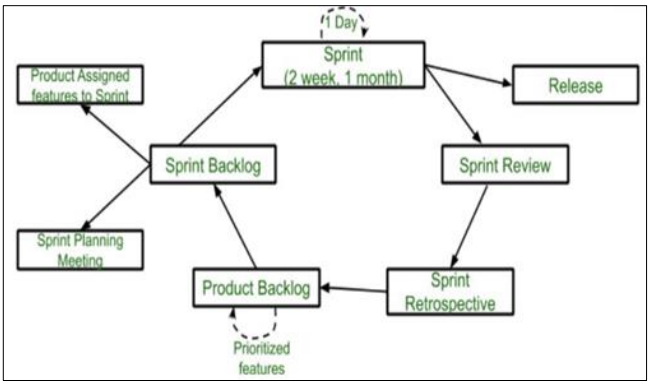


**Fig 3:** Lifecycle of Scrum

**4) Working procedure:**

Throughout sprint planning, a few artifacts are followed at different stages for the scrum lifecycle to manage sprint goals and maintain transparency.

- Product Backlog: The Product Owner is responsible for developing and updating a prioritized list of tasks and needs.
- Sprint Planning: Everyone in the project collaborates to plan the sprint and analyze the goals to be met
- Effort Estimations: Efforts for each task are estimated in sprint planning by the scrum master along with the team, and selects the tasks to work on
- Scrum Execution: The team begins working on assigned tasks according to the efforts required for each task and executes the sprint, showing progress in daily scrum calls.
- Sprint review: The team reviews the completed product with stakeholders and collects their feedback.
- Sprint Retrospective: The internal team discusses the completed project's feedback and analyzes what didn't go well, what went well, and any improvements for the future.

**B) DSDM:**

- DSDM is a widely used agile methodology that efficiently delivers high-quality outcomes and uses heavier approaches. It follows the principle of "fitness for business purposes." The stakeholders are involved throughout the project and communicate openly. DSDM performs various iterative cycles for new improvements or updates. Throughout its development, DSDM follows a few principles.
- On-time Delivery: As an agile methodology, DSDM utilizes predetermined time periods and iterative cycles to develop a product. A significant focus is placed on timely delivery to manage evolving requirements.
- Maintains Quality: Agile teams must optimize processes and eliminate waste. They must establish quality criteria from the beginning and maintain these standards throughout the project. To comply with this principle, DSDM teams should consistently test and evaluate their outputs.
- Focus on the business requirements: DSDM Proponents assert that a business case should be established for the project before any development work commences. The team must have a complete and clear understanding of the project's priorities so that every choice contributes to reaching the specified objectives.
- Collaboration: Every team member should work collaboratively to achieve success in DSDM. The team should work as a single unit to be more creative toward the goal.
- Communication in the team: Frequent communication is more important when working collaboratively on the same goal. Teams must be transparent about their needs.
- On-time Delivery: As an agile methodology, DSDM utilizes established timeboxes and iterations for product development. A significant focus is placed on timely delivery to manage evolving requirements.

**1) Lifecycle:**

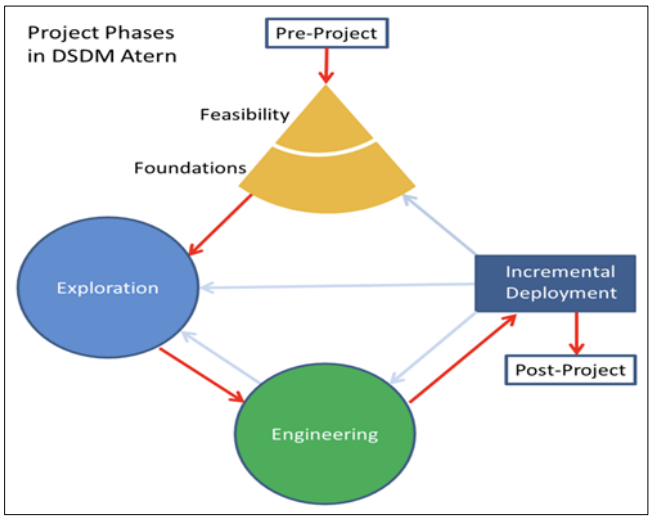


**Fig 4:** Lifecycle of DSDM

**2) Working procedure:**

**Feasibility Phase:** This phase functions as a requirement-gathering phase in the waterfall model. It gathers the essential business needs to design and model And validates the scope and budget estimates.

**Foundations phase:** This is where the initial product backlog will be created. Outline the Project Strategy, which encompasses the schedule, resource distribution, and financial plan.

**Iterative development phase:**

**Exploration phase:** The team analyses and understands the requirements to focus on the goals.

Engineering phase: The team creates a production-ready environment after the development and continuous testing.

**Deployment phase:** After meeting the business needs, deployment is completed in the production environment. After the deployment, a post-implementation review is conducted to ensure all the priorities are met.

**Post-project phase:** The stakeholders and team ensure the project's success and address any outstanding issues. If everything seems good, the project will be closed.

**6. Challenges in agile methodology**

1. Agile necessitates a comprehensive grasp of its core principles and methodologies, including Scrum, Kanban, and XP.
2. Agile prioritizes functional software above detailed documentation, which may pose challenges for projects that demand significant regulatory or technical documentation.
3. If the team is adapted to traditional methods like a waterfall, the transition becomes difficult to understand agile methodologies.

**7. Conclusion**

Traditional methods, like waterfall, are easy to understand and say Design before Code5. Meanwhile, the Agile methodology is more effective and delivers high-quality code. Waterfall can be used for small-scale projects. Also, agile requires an efficient team with good knowledge of the methodologies. Hence, Agile is ideal for R&D work. As there are many methods, it is tough to prioritize one method for one organization6. So, in Today's world, organizations are often interested in both methodologies and choose hybrid approaches.

**Abbreviations and Acronyms**

- XP - Extreme Programming
- API- Application Interface
- JWT- JSON Web Token
- FDD- Feature Driven Development
- DSDM- Dynamic System development Method
- SDLC- Software development Lifecycle Method
- ROI- Return On Investment
- CRC- Class Responsibility Collaborator
- TDD- Test-Driven Development
- R&D- Research And Development

**8. References**

1. Demirag A, Demirkol Öztürk EN, Ünal C. Analysis and comparison of waterfall model and agile approach in software projects. AJIT-E: Academic Journal of Information Technology. 2023;14(54):183-203. doi:10.5824/ajite.2023.03.002.x.

2. Khan S, Mahadik S. A comparative study of agile and waterfall software development methodologies. International Journal of Advanced Research in Science, Communication and Technology. 2022;2(1):399-402. doi:10.48175/IJARSCT-5696.
3. Jain P, Sharma A, Ahuja L. The impact of agile software development process on the quality of software product. 2018 7th International Conference on Reliability, Infocom Technologies and Optimization (Trends and Future Directions) (ICRITO). 2018:812-815. doi:10.1109/ICRITO.2018.8748529.
4. Shrivastava A, Jaggi I, Katoch N, Gupta D, Gupta S. A systematic review on extreme programming. Journal of Physics: Conference Series. 2021;2(1):26-27. doi:10.1088/1742-6596/1969/1/012046.
5. Kumar G, Bhatia PK. Comparative analysis of software engineering models from traditional to modern methodologies. 2014 Fourth International Conference on Advanced Computing & Communication Technologies. Rohtak, India: IEEE; 2014:189-196. doi:10.1109/ACCT.2014.73.
6. Mishra A, Alzoubi YI. Structured software development versus agile software development: A comparative analysis. International Journal of System Assurance Engineering and Management. 2023;14:1504-1522. doi:10.1007/s13198-023-01958-5.