Research of Contrast between Waterfall Model and Prototype Model

Rishika Srivastava

Student of MCA, MCA Department, SRMCEM, Lucknow, UP, India
Email: *rishika_mc21037[at]srmcem.ac.in*

Abstract: ***The software development process is complex and requires a structured approach to ensure successful completion. There are various models that can be used for software development, with the Waterfall Model and the Prototype Model being two of the most commonly used. The waterfall model follows a sequential and linear approach, while the prototype model involves building a preliminary version of the software that can be refined and improved upon. This paper compares and contrasts the waterfall model and the prototype model, examining their advantages and disadvantages, and suitability for different types of projects.***

Keywords: Software, software development model, waterfall model, prototype model, phases, requirement

1. Introduction

Software development models are methodologies used to guide the software development process from start to finish. These models provide a structured approach to software development, ensuring that projects are completed efficiently, on time, and within budget. There are many different software development models, each with its own set of advantages and disadvantages.

The waterfall model is well-suited for projects with clear and well-defined requirements. It provides a structured approach to software development, where each phase of the process must be completed before moving on to the next phase. However, it is inflexible and does not allow for changes once a phase has been completed, which can be costly and time-consuming.

In contrast, the prototype model involves building a preliminary version of the software that can be refined and improved upon based on feedback from stakeholders. This model allows for greater flexibility and customer involvement, and can lead to a final product that better meets the needs of the customer. However, the prototype model can be less structured and may require more time and resources for development.

The choice of a software development model depends on the nature of the project and its requirements. Some projects may require a more structured approach, while others may require greater flexibility. It is important to choose the right software development model to ensure that the project is completed successfully.

2. The Waterfall Model

The waterfall model is a widely used software development methodology that involves a linear and sequential approach to software development. This model is used to manage software development projects and was first introduced by Dr. Winston W. Royce in 1970. The waterfall model is a process that follows a strict set of steps, where each step must be completed before moving on to the next step. This model has been the foundation for many software development methodologies, including agile and iterative methodologies.

The phases of the waterfall model include requirements gathering and analysis, system design, implementation, testing, deployment, and maintenance. The waterfall model provides a structured approach to software development, but it can be inflexible and does not allow for changes once a phase has been completed. While it may be well-suited for projects with clear and well-defined requirements, it may not be the best approach for projects that require a lot of customer involvement or where changes are likely to be required. Overall, the waterfall model has both advantages and disadvantages, and its suitability depends on the nature of the project and its requirements.

1) Phases of waterfall model

The waterfall model is a sequential process where the software development process is divided into a series of phases. The phases are:

a) Requirements Gathering

The first phase of the waterfall model is requirements gathering. This involves gathering and defining the requirements of the software. This phase is critical because it sets the stage for the entire software development process. During this phase, the development team works closely with the stakeholders to understand their needs and requirements.

b) Analysis

The second phase of the waterfall model is analysis. During this phase, the development team analyzes the requirements gathered in the previous phase and identifies any potential problems. The team also develops a detailed plan for the software development process.

c) Design

The third phase of the waterfall model is design. During this phase, the development team creates a detailed design of the software. This includes developing system architecture, creating data models, and defining the user interface.

d) Implementation

The fourth phase of the waterfall model is implementation. During this phase, the development team creates the actual software based on the design created in the previous phase. This phase involves writing code, integrating software components, and conducting unit testing.

e) Testing

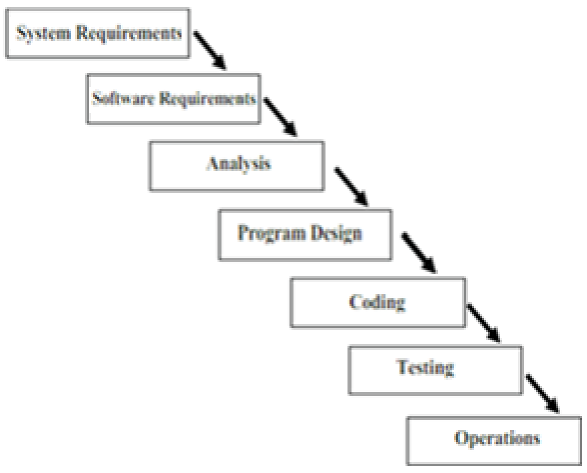
The fifth phase of the waterfall model is testing. During this phase, the development team tests the software to ensure that it meets the requirements and works as expected. This phase includes system testing, integration testing, and user acceptance testing.

f) Maintenance

The final phase of the waterfall model is maintenance. During this phase, the development team maintains the software and provides support to the end-users. This includes fixing any bugs, addressing user feedback, and updating the software as necessary.

Each of these phases must be completed before moving on to the next phase. In the waterfall model, the output of one phase becomes the input for the next phase. For example, the output of the requirements gathering phase becomes the input for the system design phase.

2) Diagram of waterfall model



3) Feature of waterfall model

- Sequential Phases: The waterfall model consists of a series of sequential phases, where each phase must be completed before moving onto the next phase. These phases typically include requirements gathering, design, implementation, testing, deployment, and maintenance.
- Document-Driven: Each phase of the waterfall model is typically driven by a set of documents, such as requirements documents, design documents, and test plans. These documents are used to define the scope of the project, identify the stakeholders, and track progress.
- Emphasis on Planning: The waterfall model places a heavy emphasis on planning. Before any development work begins, a comprehensive project plan is created that outlines all of the project's goals, timelines, and deliverables.
- Limited Iterations: The waterfall model is characterized by a limited number of iterations. Once a phase is complete, it is difficult to make changes without going back to the beginning of the project.
- Emphasis on Testing: Testing is a critical part of the waterfall model. Each phase includes a testing component, which helps to ensure that the final product meets all of the specified requirements.
- Emphasis on Quality: The waterfall model places a strong emphasis on quality. Each phase of the model is designed to produce a high-quality deliverable that meets all of the specified requirements.
- Linear Progression: The waterfall model follows a linear progression, where each phase must be completed before moving onto the next phase. This approach is often criticized for being inflexible and limiting creativity.
- Well-Defined Deliverables: The waterfall model produces well-defined deliverables at the end of each phase. This makes it easier to track progress and ensure that all of the project goals are being met.
- Limited Customer Involvement: The waterfall model is typically characterized by limited customer involvement. Customers are involved in the initial requirements gathering phase, but their input is typically not solicited during the later phases of the project.
- Limited Scope for Changes: Once a phase is complete in the waterfall model, it is difficult to make changes without going back to the beginning of the project. This can make it challenging to accommodate changing customer requirements or unforeseen issues that arise during development.

4) Advantages of waterfall model

- Clear and well-defined requirements: The waterfall model is well-suited for projects where the requirements are clear and well-defined. This allows the development team to plan and execute each phase of the project with a clear understanding of what needs to be done.
- Structured approach: The waterfall model provides a structured approach to software development. This allows the development team to work in a systematic and organized manner, which can improve the efficiency of the development process.
- Easy to understand: The waterfall model is easy to understand and can be easily communicated to stakeholders. This can help ensure that everyone involved in the project has a clear understanding of what needs to be done and when it needs to be done.
- Emphasis on Documentation: The waterfall model places a strong emphasis on documentation. Each phase of the development process is documented, which helps to ensure that all team members are aware of what is expected of them. This documentation also makes it easier for new team members to understand the project.
- Clear Deliverables: The waterfall model requires that each phase of the development process has clear deliverables. This ensures that the project stays on track, and that progress can be measured and

monitored. Clear deliverables also help to ensure that the project is completed on time and within budget.

5) Disadvantages of waterfall model

- Inflexibility: The waterfall model is inflexible and does not allow for changes once a phase has been completed. This means that if a change is required, it can be difficult and expensive to implement.
- Limited customer involvement: The waterfall model does not allow for a lot of customer involvement during the development process. This can result in a final product that does not meet the needs of the customer.
- Testing at the end: The waterfall model involves testing at the end of the development process. This can result in errors and bugs that are not identified until the end of the process, which can be costly to fix.

3. The Prototype Model

The prototype model is a software development methodology that involves building a preliminary version of the software that can be refined and improved upon. This model is used to manage software development projects and was first introduced by IBM in the 1960s. The prototype model is a process that involves creating a preliminary version of the software, called a prototype, and using feedback from stakeholders to refine and improve upon it.

The prototype model is an iterative process where the software development process is divided into a series of iterations. Each iteration involves building a prototype of the software, which is used to gather feedback from stakeholders. This feedback is then used to refine and improve upon the prototype in subsequent iterations. The prototype model is an iterative process that allows for greater flexibility and customer involvement in the software development process.

1) Phases of prototype model

- **Requirements Gathering:-**

In the requirements gathering phase of the prototype model, the development team works with the client to understand the software requirements. This phase is similar to the requirements gathering phase in the waterfall model. However, in the prototype model, the focus is on understanding the user's needs and expectations, rather than on documenting every requirement.

- **Prototyping:-**

The prototyping phase is where the actual prototype is developed. This phase is unique to the prototype model and is not present in the waterfall model. The prototype is usually developed quickly, with a focus on functionality rather than aesthetics. The prototype is then tested with the client to identify any issues or changes that need to be made.

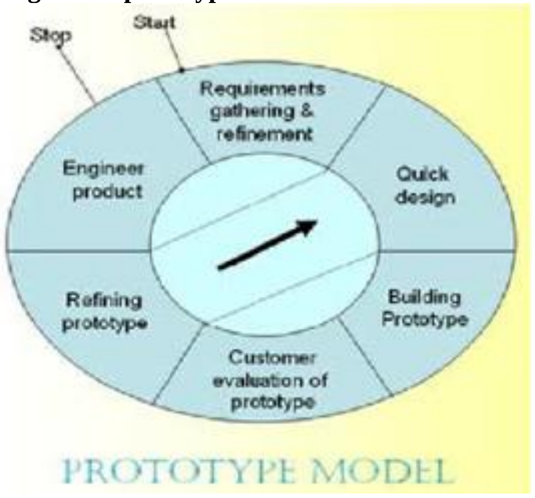
- **Refining:-**

Once the prototype is developed, the development team works on refining it. This phase is similar to the design and development phases of the waterfall model. The focus is on improving the functionality of the prototype, based on the feedback received during the testing phase. The prototype is iteratively refined until it meets the client's needs.

- **Final Product Development:-**

Once the prototype is refined and approved by the client, the development team works on developing the final product. This phase is similar to the implementation and testing phases of the waterfall model. However, in the prototype model, the development team has a better understanding of what the client wants, which reduces the risk of changes or revisions during the development process.

2) Diagram of prototype model



a) Features of prototype model

- Iterative Development: The prototype model is characterized by an iterative development process, where the product is developed in stages and each stage is refined based on user feedback.
- Quick Feedback: The prototype model emphasizes quick feedback from users, which helps to ensure that the product meets the needs of its intended audience.
- Emphasis on User Interface: The prototype model places a strong emphasis on user interface design, since the product's user interface is a critical component of its success.
- Limited Scope: The prototype model typically has a limited scope, since the goal is to create a working prototype that can be used to gather user feedback and refine the product.
- Rapid Development: The prototype model emphasizes rapid development, since the goal is to create a working prototype as quickly as possible.
- User Involvement: The prototype model emphasizes user involvement, since the goal is to gather feedback from users throughout the development process.
- Functionality Focused: The prototype model is focused on developing the product's core functionality first, and then refining its user interface and other features based on user feedback.
- Risk Reduction: The prototype model is designed to reduce risk, since the product's design and functionality can be refined based on user feedback before it is released to the public.

- Limited Documentation: The prototype model typically has limited documentation, since the focus is on creating a working prototype rather than a comprehensive set of design documents.
- Flexible: The prototype model is flexible, since it allows for changes to be made based on user feedback and other factors that may emerge during the development process.

b) Advantages of prototype model

- Greater customer involvement: The prototype model involves greater customer involvement in the software development process. This can lead to a final product that better meets the needs of the customer.
- Flexibility: The prototype model is flexible and allows for changes to be made throughout the development process. This can reduce the risk of errors and bugs in the final product.
- Cost-effective: The prototype model can be cost-effective, as changes can be made early in the development process when they are less expensive to implement.
- Early detection of issues: A prototype model allows stakeholders to test the system and identify issues early on in the development process, before significant time and resources have been invested. This can save time and money in the long run.
- Reduced development time: The feedback gathered from the prototype model can be used to refine the system's design and requirements, which can help reduce development time and avoid costly rework.

c) Disadvantages of prototype model

- Lack of structure: The prototype model can be less structured than other software development methodologies. This can lead to a lack of clarity and organization in the development process.
- Time-consuming: The prototype model can be time-consuming, as multiple iterations are required to refine and improve upon the prototype.
- Scope creep: The prototype model can lead to scope creep, where the project expands beyond its original scope. This can result in delays and increased costs.

4. Comparison

The waterfall model is well-suited for projects with clear and well-defined requirements. It provides a structured approach to software development, where each phase of the process must be completed before moving on to the next phase. However, it is inflexible and does not allow for changes once a phase has been completed, which can be costly and time-consuming.

In contrast, the prototype model involves building a preliminary version of the software that can be refined and improved upon based on feedback from stakeholders. This model allows for greater flexibility and customer involvement, and can lead to a final product that better meets the needs of the customer. However, the prototype model can be less structured and may require more time and resources for development.

Waterfall model and Prototype model are two different software development models that have unique features and characteristics.

- Approach: The Waterfall model follows a sequential approach where each phase is completed before moving on to the next one, whereas the Prototype model follows an iterative approach where each phase is repeated until the final product is achieved.
- Requirements: In the Waterfall model, the requirements are documented at the beginning of the project, whereas in the Prototype model, the requirements are refined and modified throughout the development process.
- Feedback: The Waterfall model does not provide feedback until the end of the project, whereas the Prototype model provides feedback throughout the development process.
- Testing: In the Waterfall model, testing is done at the end of the project, whereas in the Prototype model, testing is done throughout the development process.
- Time and Cost: The Waterfall model is a linear process that requires a lot of time and cost, whereas the Prototype model is an iterative process that can save time and cost by identifying and fixing issues early in the development process.
- Flexibility: The Waterfall model is not flexible and does not allow for changes once the development process has started, whereas the Prototype model is flexible and allows for changes to be made throughout the development process.
- Risk: The Waterfall model is high-risk as it does not provide feedback until the end of the project, whereas the Prototype model is low-risk as it provides feedback throughout the development process.

5. Conclusion

In this paper, the theory of waterfall model and prototype model with their phases, advantages and disadvantages were discussed. Also diagram of both the models has been shown. Also feature are compared of both the models.

Overall, the choice between the waterfall model and the prototype model depends on the nature of the project and its requirements. Projects with clear and well-defined requirements may be better suited for the waterfall model, while projects with more fluid requirements and greater customer involvement may benefit from the prototype model. A combination of both models may also be suitable for certain projects, with the initial development following the prototype model and subsequent phases following the waterfall model.

References

[1] Royce, Winston W.Managing the development of large software systems. Proceedings of IEEE WESCON, August 1970, pages 1-9.
[2] Boehm, Barry W. Software Engineering Economics. Prentice Hall1981.
[3] Basili, Victor R., and Rombach, H. Dieter. The TAME project: towards improvement-oriented

software environments. IEEE Transactions on Software Engineering, vol. 14, no. 6, June 1988, pages 758-773.

[4] Davis, Alan M. Software Prototyping and Productivity: A Quantitative Study. IEEE Transactions on Software Engineering, vol. 14, no. 2, February 1988, pages 211-216.

[5] Sommerville, Ian, and Sawyer, Pete. Requirements engineering: a good practice guide. John Wiley & Sons 1997.

[6] Sommerville, Ian. Software engineering. Pearson Education Limited 2016.

[7] Pressman, Roger S. Software engineering: a practitioner's approach. McGraw-Hill Education 2014.

[8] Kettunen, Petri, and Soini, Juhani. Comparing software development life cycle models. International Journal of Software Engineering and Knowledge Engineering, vol. 8, no. 5, October 1998, pages 591-616..

[9] Boehm, Barry W., and Ross, Ronald T. Theory-W Software Project Management: Principles and Examples. IEEE Transactions on Software Engineering, vol. 1, no. 1, March 1975, pages 2-18.

[10] Nielsen, Jakob. Usability Engineering. Academic Press 1993.

[11] Vliet, Hans van. Software Engineering: Principles and Practice. John Wiley & Sons 2008.

[12] Ramesh, Balasubramaniam, and Jarke, Matthias. Toward reference models for requirements traceability. IEEE Transactions on Software Engineering, vol. 28, no. 7, July 2002, pages 580-593.

[13] Wilfred van Casteren. The Waterfall Model and the Agile Methodologies: A comparison by project characteristics. Academic Competences in the Bachelor 2assignment: Write a scientific article on 2 Software Development Models March 7, 2017.

[14] Agarwal, M., & Bhardwaj, A. (2012). An analysis of the waterfall model of software development. International Journal of Computer Science and Information Technologies, 3(4), 4829-4833.

[15] Zalewski, J. (2013). The waterfall model in software development. Journal of Systems and Software, 86(6), 1498-1511.

[16] Basu, A., & Pal, T. (2017). An evaluation of the waterfall model for software development. International Journal of Advanced Research in Computer Science, 8(4), 1-5.

[17] Konda, S., & Mishra, A. (2008). A review of software prototyping techniques. Journal of Computing, 2(3), 8-13.

[18] Budgen, D. (2003). Software design. Addison-Wesley.

[19] Hailpern, B., & Tarr, P. (1999). Using prototypes to develop software: A practitioner's guide. Addison-Wesley.