Spiral Model - A Risk-Driven Approach to Software Development

Chau Ying Jia

Department of IT

Ipoh,Perak

[chau@graduate.utm.my](mailto:chau@graduate.utm.my)

Afif Shaqir

Department of CS

Shah Alam,Selangor

[afifshaqirirfan@graduate.ut](mailto:afifshaqirirfan@graduate.ut)m.my

Adib Zikri

Department of CS

Besut, Terengganu

[ahmadadibzikri@graduate.utm.m](mailto:ahmadadibzikri@graduate.utm.my)y

***Abstract* - Software Development Life Cycle(SDLC) is a process to develop software or applications by fulfilling users requirements. Various software process models can be used according to the suitability of the situation expressed by users. We will focus on the spiral model and waterfall model in this paper. The spiral model is an incremental model and has gained significant attention due to its emphasis on risk management and continuous evaluation. It creates a risk-driven approach to the software process but it is resolved throughout the process.While a waterfall uses sequential steps, one process after the other. Those processes can't be revisited after their completion. The purpose of this paper is for a deeper understanding of the spiral and waterfall model and about their differences**

***Keywords - spiral model, waterfall model, incremental model, risk management, sequential process***

*I - INTRODUCTION*

To develop, design, build and maintain a software or industrial system, we are always suggested to follow the order of phases developing a software to ensure a complete and perfect system is being produced.

Spiral model was initially proposed by Barry Boehm in 1988. It is an iterative incremental development process that combines elements of the waterfall model and prototyping. Unlike the other linear progressions, it develops in a series of spiral phases. It has 5 phases which are objectives defined, risk analysis, engineering, evaluation and planning.[1] This iterative approach allows for continuous evaluation, risk assessment, and maintenance throughout the development lifecycle in each spiral.It is well-suited for large and complex projects due to its risk management capabilities.[2] The spiral model is represented with the help of spiral concept but the number of loops depends purely upon how complex the project is.

While the waterfall model is defined by its linear and rigid structure. Each phase should be complete before moving on to the next, there is no returning back to the front phases. It requires thorough documentation and up-front planning, which can provide a clear project roadmap.[3] It consists of seven phases and each phase is completed sequentially, the process is easy to manage, with clear checkpoints to monitor progress.

*II-DESIGN OF SPIRAL MODEL*

The model that Boehm proposed has been used for several years, the spiral model has been developing as a result of experience with several waterfall model improvements that are mainly used by many software projects. In contrast to the waterfall approach, there are no set stages for software specification ,design , or implementation. Loops in the spiral as shown in Fig.1 are chosen depending on the requirements and each loop in spiral represents a phase in the software development process.The spiral model was created with iterative development in mind; each loop represents a project cycle, with the focus on continuously improving and developing the software. In contrast to conventional linear methods, the spiral model includes stakeholder input and risk analysis at each stage, allowing for continuous assessment and revisions to the project's goals, specifications, and solutions in response to potential risks or changes.

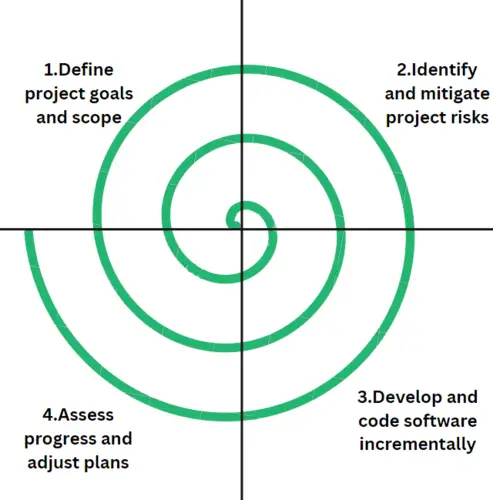


Fig. 1: Spiral model

Each sector of each loop in the spiral development model is selected based on the requirements, and each loop represents a phase in the software development process. The sectors for defining project goals and scope specify the system objectives, such as system functionality and performance, which are determined for the phase. Determining the system's main objectives and concentrating on what the project is intended to accomplish are key components of defining project goals. These objectives frequently have to do with the functioning of the system, including certain features and capabilities as well as performance standards like scalability, reliability, and response times.

The next phase is about identifying project risk, the risks are identified and the related activities are put in place in order to reduce the key risks. The risk evaluation activities such as a more detailed analysis, prototyping, or simulation can take part to identify and resolve the possible risks.[5]The next phase involves incremental software development and coding. This process involves developing the website, coding it, testing its acceptability, and hosting it. We do this using either the basic waterfall model or the incremental model, depending on which is best for coding. Each build is assigned a corresponding version number in the following spirals once we have a better understanding of the requirements.[6]

Following the spiral model, "assess progress and adjust plans" is a crucial step in maintaining project consistency with objectives and adjusting to developing obstacles or changes. The team gathers input from stakeholders, assesses deliverables, identifies new risks, and reviews progress after each iteration. Project plans are modified in light of this evaluation; this may entail reworking schedules, redistributing resources, or improving requirements. The project may remain adaptable and responsive thanks to this iterative strategy, which lowers risks and raises the likelihood of success.[7]

*III-THE WATERFALL MODEL*

The Waterfall model is one of the earliest software development methodologies in the field of software engineering introduced by WinstonRoyce in 1970. It is most widely used in Europe and North America.[13] It is a sequential and linear methodology, which separates the development process into stages like analysis, design, implementation, testing, deployment, and maintenance. Each phase must be completed before the next phase can begin, without any overlap.[10, 12] This structured progression is similar to the flow of a waterfall, where each phase seamlessly transitions into the next process. Fig. 2 shows the graphical representation of the waterfall model.

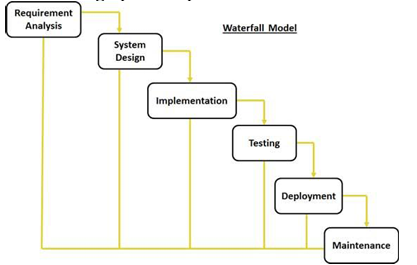


Fig. 2 Process of a waterfall model

The Waterfall model follows a direct and linear approach to problem-solving, making it relatively simple to manage and understand. However, its rigid structure can limit the flexibility, as there are still changes to requirements or design elements that often require returning to earlier stages. In addition, the waterfall model has limited opportunities to get the clients feedback for maintenance as clients may have extra or changes of request during the project execution and the waterfall model would not revert back the process once the requirements phase has ended.[3] The project is out of the hands of the client which has led to the requirements of the client should have not been met. These are the downsides of the waterfall model.

In the end, waterfall methodology is more suitable for projects with well-defined and straightforward features that are unlikely to change during development.[11] When the project’s requirements are well documented and fixed before the implementation, enabling the team to define the entire project scope, create a complete schedule, and design the overall application, then it is suitable to use the waterfall methodology. This approach enhances resource utilization by allowing tasks to be executed in parallel or grouped strategically to match team members' skills. With a thorough understanding of all requirements and deliverables, it also leads to a more effective application design. Additionally, a detailed schedule and resource plan make it easier to assess project progress and status.[14]

*IV-STRENGTH AND WEAKNESS*

The Spiral Model integrates risk analysis and quality assurance into every cycle. This means that quality objectives such as performance benchmarks, user experience, and security standards are consistently prioritized and checked. By focusing on these aspects at every iteration, the model supports continuous improvement and helps deliver a more reliable and quality final product.[5]

The Spiral Model's dependence on risk assessment knowledge is one of its main weaknesses. A lack of expertise or proficiency in risk management can have a detrimental effect on the project's success because the model places a strong emphasis on identifying and reducing risks at every iteration. Because of this, the Spiral Model is less appropriate for teams that need more knowledge or resources to evaluate risks effectively.[5]

*V-COMPARISON OF WATERFALL MODEL AND SPIRAL MODEL*

As we know, both Waterfall model and Spiral model have their own strengths and weaknesses. Waterfall excels in projects that have well-defined requirements and a clear scope, while Spiral shines in projects that are more complex, require continuous risk management, and benefit from iterative refinement. To be more of it, here is the comparison between Waterfall and Spiral.

To begin with, Waterfall model is considered simple and easy.[15] Each phase is completed one at a time such as requirement, design, development, testing and deployment in specific order without. The progress moves from phase to phase without overlap. Meanwhile, Spiral model is more complex due to the integration of iterative development with risk analysis. [15] The cycle involves multiple phases repeated over several iterations. Despite being more flexible and adaptable to changes, it added more complexity in terms of project management.

Furthermore, early-stage planning in Waterfall model must be completed in detail before moving on to the next phase.[8] Since Waterfall follows a linear and sequential process, the success of each phase relies on the well-defined requirements from the start. The fail or incomplete requirements, there will be problems later during the development process, as changes are difficult once the implement phase finishes. [9] On the other hand, it is more flexible for early-stage planning in Spiral model.[8] Since Spiral uses an iterative approach, requirements can be revisited and refined during each cycle or iteration. If the project doesn’t have clear or complete requirements at the beginning, the Spiral model provides the flexibility to address them over time, focusing on risk management and feedback loops to refine the project gradually. [9]

Lastly, least maintenance needed for Waterfall model.[15] Since the requirements are gathered and finalized at the beginning, and the development follows a strict linear process, the assumption is that the product is fully functional once delivered. Because of this, there are fewer post-deployment modifications or adjustments, which translates into less continuous maintenance. Meanwhile, more typical maintenance involves in Spiral model.[8] Each cycle involves feedback which means the possibility of evolving products. So, as result, the system may require adjustments, updates, or fixes to address any issues identified during risk assessments or feedback loops after each iteration

*VI-CONCLUSION*

Both the Spiral and Waterfall models provide unique methods for developing software, each having strengths and weaknesses. Spiral is dynamic and adaptable. It enables progressive development, where requirements can be revisited and improved throughout the project lifecycle. Meanwhile, Waterfall model is more straightforward and structured. It is an effective approach for projects with well-defined and stable requirements, where the scope is clear from the start and unlikely to change. Maybe in the future, we can discover new models that combine Spiral and Waterfall. They might be offering the structure of Waterfall for certain project phases and the flexibility of Spiral for others. Hence, Spiral and Waterfall will continue to influence how teams approach problem-solving, risk management, and project execution, adapting to the growing demands of future technologies and industries.

*VII-REFERENCE*

[1]GeeksforGeeks. (2024b, July 30). *What is Spiral Model in Software Engineering?* [online] GeeksforGeeks. Available at:

<https://www.geeksforgeeks.org/software-engineering-spiral-model/>

[2]Potok, T. E. (1992). Extensions to the spiral model to support joint development of complex software systems. *ACMSE ’92: Proceedings of the 30th Annual ACM Southeast Regional Conference*, [online] 124. Available at: <https://dl.acm.org/doi/10.1145/503720.503762>

[3]Atlassian, B. (n.d.). *Waterfall Methodology for Project Management | Atlassian*. [online] Atlassian. Available at: <https://www.atlassian.com/agile/project-management/waterfall-methodology>

[4]GeeksforGeeks. (2023, December 2). *Spiral pattern image*. [online] GeeksforGeeks. Available at: <https://media.geeksforgeeks.org/wp-content/uploads/20231202112409/spiral_77.png>

[5]Ali Selamat, Md Hafiz Selamat, & Nurulhuda Zainuddin. (2021). Software engineering (1st ed.). Penerbit UTM Press.

[6]Boehm, B. W. (1988). A spiral model of software development and enhancement. *Computer*, *21*(5), 61-72.Available at: <https://www.cse.msu.edu/~cse435/Homework/HW3/boehm.pdf>

[7]International Journal of Scientific & Engineering Research (IJSER). (n.d.). *Analysis of spiral model in software projects for the software houses of Pakistan*.Available at:

<https://www.ijser.org/researchpaper/Analysis-of-Spiral-Model-in-Software-Projects-for-the-Software-Houses-of-Pakistan.pdf>

[8]GeeksforGeeks. (2024, May 23). *Difference between Waterfall Model and Spiral Model*. [online] GeeksforGeeks. Available at: <https://www.geeksforgeeks.org/difference-between-waterfall-model-and-spiral-model/>

[9]*ChatGPT - Spiral vs Waterfall Models*. [online] (n.d.). ChatGPT. Available at: <https://chatgpt.com/share/6720598c-d320-800e-948d-7da7b8919353>

[10]S, S. (2017). A Study of Software Development Life Cycle Process Models. *SSRN Electronic Journal*.[online] Available at: <https://papers.ssrn.com/sol3/papers.cfm?abstract_id=2988291>

[11]Petersen, K., Wohlin, C., & Baca, D. (2009). The Waterfall Model in Large-Scale Development. In *Lecture notes in business information processing* [online] (pp. 386–400). Available at: [https://link.springer.com/chapte/10.1007/978-3-642-02152-7\_29](https://link.springer.com/chapter/10.1007/978-3-642-02152-7_29)

[12]*SDLC - Waterfall Model*. [online] (n.d.). Available at: <https://www.tutorialspoint.com/sdlc/sdlc_waterfall_model.htm>

[13] Ruël, H.J.M., Bondarouk, T. and Smink, S. (2010). The Waterfall Approach and Requirement Uncertainty. *International Journal of Information Technology Project Management*, [online] 1(2), pp.43–60. doi: <https://doi.org/10.4018/jitpm.2010040103>

[14]Sherman, R. (2015). *Waterfall Methodology - an overview | ScienceDirect Topics*. [online] Sciencedirect.com. Available at: <https://www.sciencedirect.com/topics/computer-science/waterfall-methodology>.

[15]Zaman, S. (2023). *Spiral Vs Waterfall Model - The Ultimate Choice For Software Development*. [online] Impala Intech. Available at: <https://impalaintech.com/blog/spiral-vs-waterfall/>