COMPARISON BETWEEN RAD AND WATERFALL MODELS

For

Dr. Paul J. Mc Evoy

By

Sheraz Hussain 10592123

Fardeen Ahmed 10587424

Hamze Shekh Najeeb Gazal 10576081

Daniel Amler 10587694

December 10, 2023

COMPARISON BETWEEN RAD AND WATERFALL MODEL

[Table of Contents 1](#_Toc153195271)

[1. Introduction 3](#_Toc153195272)

[2. History of SDLC: Evolution and Early Methodologies 3](#_Toc153195273)

[2.1 1950s-1960s: Early software development without a formalized process 4](#_Toc153195274)

[2.2 1970s: Emergence of SDLC methodologies 5](#_Toc153195275)

[3. Evolution of the Waterfall Model: History and Key Points 5](#_Toc153195276)

[3.1 History of Waterfall Model 6](#_Toc153195277)

[3.2 Key Points of the Waterfall Model: 6](#_Toc153195278)

[3.3. Evolution of the Waterfall Model: 7](#_Toc153195279)

[4. History of RAD: Evolution and Key Concepts 8](#_Toc153195280)

[4.1. Rapid Application Development (RAD): 8](#_Toc153195281)

[4.1.1. Agile Approach with Emphasis on Quick Development and Iteration: 8](#_Toc153195282)

[4.1.2. Key Component: Prototyping and Early User Feedback: 8](#_Toc153195283)

[4.1.3. Collaboration between Developers and Users: 9](#_Toc153195284)

[5. RELEVANCE 9](#_Toc153195285)

[5.1. History and Evolution of SDLC: 9](#_Toc153195286)

[5.2. Relevance of the Waterfall Model: 9](#_Toc153195287)

[5.3. Relevance of RAD in Contemporary Software Development: 10](#_Toc153195288)

[6. DESIGN OF RAD AND WATERFALL MODEL: 10](#_Toc153195289)

[6.1. Waterfall Model: 10](#_Toc153195290)

[6.2. Advantages: 12](#_Toc153195291)

[6.3. Disadvantages: 13](#_Toc153195292)

[6.4. RAD Model: 14](#_Toc153195293)

[**6.5. Advantages** 16](#_Toc153195294)

[**6.6. Disadvantages** 16](#_Toc153195295)

[7. Waterfall Model Implementation 17](#_Toc153195296)

[7.1. Requirements Analysis: 18](#_Toc153195297)

[7.2. Design: 18](#_Toc153195298)

[7.3. Development: 19](#_Toc153195299)

[7.4. Evaluation and testing: 19](#_Toc153195300)

[7.5. RAD Model Implementation 20](#_Toc153195301)

[7.5.1. Planning: 21](#_Toc153195302)

[7.5.2. Analysis: 21](#_Toc153195303)

[7.5.3. Design: 22](#_Toc153195304)

[7.5.4. Implementation: 22](#_Toc153195305)

[**8. Advocates and Detractors** 23](#_Toc153195306)

[**8.1. Advocate** 23](#_Toc153195307)

[8.2. Detractors 23](#_Toc153195308)

[8.3. Unified CX Matrix 24](#_Toc153195309)

[8.4. Omnichannel Approach 24](#_Toc153195310)

[8.5. Human Talent 24](#_Toc153195311)

[References 27](#_Toc153195312)

# 1. Introduction

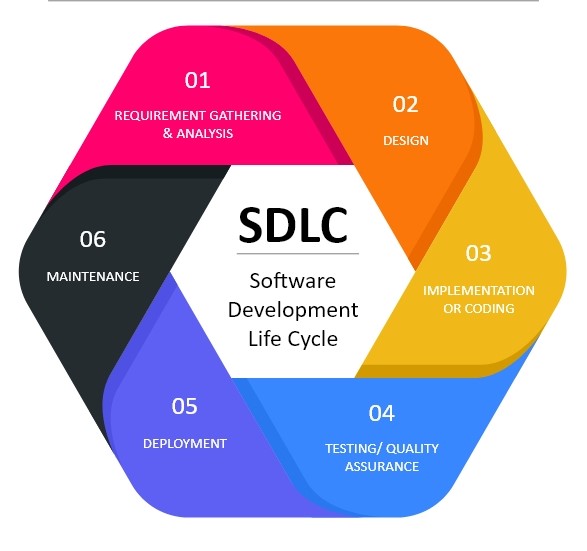
When developing a new piece of software, There are many methodologies that can be conducted to achieve a well-structured and fully operational software, Developers need to choose the correct methodology before starting any work, These methodologies or as they are called System Development Life Cycle models (SDLC), Are the starting pillar to any software engineering project and are a set of instructions and steps that must be followed to develop any kind of software, These steps can include: Planning, Analysis, Design, And implementation respectively, Thus, It is crucial to identify the methodology that will suit your application or business needs, There are multiple options of SDLC models that can be implemented, Each with unique set of phases and approaches to the relevant procedures and each one serves for a particular goal you want to achieve from your software, Moreover, Depending on the model chosen, The testing techniques and approaches will vary, Types of SDLC models include the following:

1. RAD Model
2. Waterfall Model
3. Spiral Model
4. V Model

Therefore, in this research, we will conduct a comparison between RAD And Waterfall models respectively, the comparison will be in the context of: Design, History, Relevance, Implementation, Advantages and Disadvantages of each model, And their Advocates and Detractors.

# 2. History of SDLC: Evolution and Early Methodologies

The concept of Software Development Life Cycle (SDLC) has evolved over time with the growth and advancements in computing and software development. This project provides a brief overview of the history of SDLC, highlighting its emergence, and focusing on early methodologies that laid the foundation for modern software development practices.



## 2.1 1950s-1960s: Early software development without a formalized process

During the 1950s and 1960s, software development was an evolving field, where developers focused on writing code without a formalized process or methodology. This period witnessed remarkable progress in computing and software development, with the emergence of programming languages and the development of large-scale software systems.

As software projects grew in complexity, developers faced challenges in managing and maintaining software due to the lack of structured processes. Without a defined approach, software development projects often encountered difficulties in meeting deadlines, controlling costs, and ensuring quality.

## 2.2 1970s: Emergence of SDLC methodologies

The 1970s marked a significant turning point in software development, as industry practitioners recognized the need for formalized methodologies to address the challenges encountered during the development process. Several SDLC methodologies began to emerge, aiming to provide a systematic and structured approach to software development.

One of the earliest and most influential SDLC models that emerged during this period was the Waterfall model. Proposed by Winston W. Royce in 1970, the Waterfall model adopted a linear and sequential approach to software development. It consisted of distinct phases, including requirements gathering, system design, implementation, testing, deployment, and maintenance. Each phase was completed before moving onto the next, with minimal opportunities for feedback or iteration.

# 3. Evolution of the Waterfall Model: History and Key Points

The waterfall model is a sequential software development process that follows a linear and structured approach. This paper aims to provide a comprehensive overview of the history, key points, and evolution of the waterfall model, referencing credible sources from the Harvard style. This information is tailored for an 8th-level student, offering a clear understanding of this widely used software development methodology.

## 3.1 History of Waterfall Model

The waterfall model was first introduced by Dr. Winston W. Royce in his seminal paper, "Managing the Development of Large Software Systems," published in 1970 at the Symposium on Advanced Programming Methods for Digital Computers. The purpose of Royce's paper was to propose a solution to manage the complexity of large-scale software development projects.

### 3.2 Key Points of the Waterfall Model:

3.2.1. Sequential Approach:

The waterfall model follows a linear and sequential structure, where each phase of development begins only after the previous phase is completed. The characteristic phases of the waterfall model include requirements gathering, system design, implementation, testing, deployment, and maintenance.

3.2.2 Rigidity:

Once a phase is completed, it seldom allows for revisiting previous stages, making it a rigid model. Each stage is considered as a distinct entity.

3.2.3 Documentation:

Emphasis is placed on extensive documentation at each stage to ensure a clear understanding of the system under development. This documentation includes requirements specifications, architectural designs, test plans, and user manuals.

### 3.3. Evolution of the Waterfall Model:

Over time, the waterfall model has undergone several modifications to address its limitations and accommodate changing software development practices. Here is an overview of the major adaptations:

3.3.1 V-Model:

The V-model, introduced in the late 1980s, adds a testing phase for each corresponding development phase. Unlike the linear nature of the waterfall model, testing is given equal importance, creating a symmetric V-shaped structure.

#### 3.3.2 Waterfall with Feedback Loops:

Acknowledging the need for feedback and iteration, this modified version integrates feedback loops between phases, allowing for minimal adjustments and enhancements during the development process.

#### 3.3.3 Agile Methodologies:

As software development became more iterative and adaptive, agile methodologies emerged. Agile methods, such as Scrum or Kanban, prioritize flexibility, collaboration, and iterative development, in contrast to the sequential rigidity of the waterfall model.

The waterfall model has played a significant role in the history of software development, setting the foundation for subsequent methodologies. While the model has proven useful in certain contexts, its limitations led to the development of more flexible approaches, such as the V-model and agile methodologies. By understanding the history, key points, and evolution of the waterfall model, software developers and organizations can make informed decisions about the most suitable methodology for their projects.

# 4. History of RAD: Evolution and Key Concepts

Agile Methodology and Rapid Application Development (RAD) are two prominent software development approaches that have emerged to address the limitations of traditional models like Waterfall. This project provides a brief history of RAD, highlighting its evolution and key concepts.

## 4.1. Rapid Application Development (RAD):

### 4.1.1. Agile Approach with Emphasis on Quick Development and Iteration:

Rapid Application Development (RAD) is a specific type of Agile methodology that places a strong emphasis on rapid development and iteration. It aims to deliver high-quality software solutions quickly and iteratively.

### 4.1.2. Key Component: Prototyping and Early User Feedback:

RAD incorporates prototyping as a crucial component. By creating prototypes at an early stage, developers can gather feedback from users and stakeholders, ensuring that the final product meets their needs and expectations more effectively.

### 4.1.3. Collaboration between Developers and Users:

RAD recognizes the importance of collaboration between developers and users throughout the entire development process. It encourages active involvement and frequent communication to ensure that software solutions align with user requirements.

# 5. RELEVANCE

From the history, key points and the evolution of the waterfall is very important for software developers and organisations. While the waterfall model is the main role for the software development, its drawbacks prompted the creation of more adaptable strategies like the V-Model and Agile Methodologies. The importance is in using information to decide which approach is best for a certain project. Rapid application development (RAD), an Agile technique that emphasises rapid development, iteration, and active user collaboration, is an advancement over the Waterfall Model.

# 5.1. History and Evolution of SDLC:

The development of approaches in response to the difficulties experienced by developers is reflected in the history of the SDLC. The efficiency of software development was destroyed in the 1950s and 60s by the lack of a clear goal, which made it impossible to fulfil deadlines, keep expenses under control, and guarantee quality. An ideal shift occurred in the 1970s with the introduction of formalised SDLC approaches, most famously Winston W. Royce's Waterfall model. Although it added difficulties, this is following, and linear method attempted to handle the complexity of large-scale software projects.

# 5.2. Relevance of the Waterfall Model:

The Waterfall approach is still suitable in some situations, despite its drawbacks. A full focus of the system under development is ensured by its significance on the documentation. In order to show how flexible the model is to be changing the development processes, it has been changed to include the testing stages of the (V-Model) and feedback loops. Making good decisions about the Waterfall model's is relevant in modern software development projects is made possible by having a solid understanding of its history and essential keys to the waterfall approach.

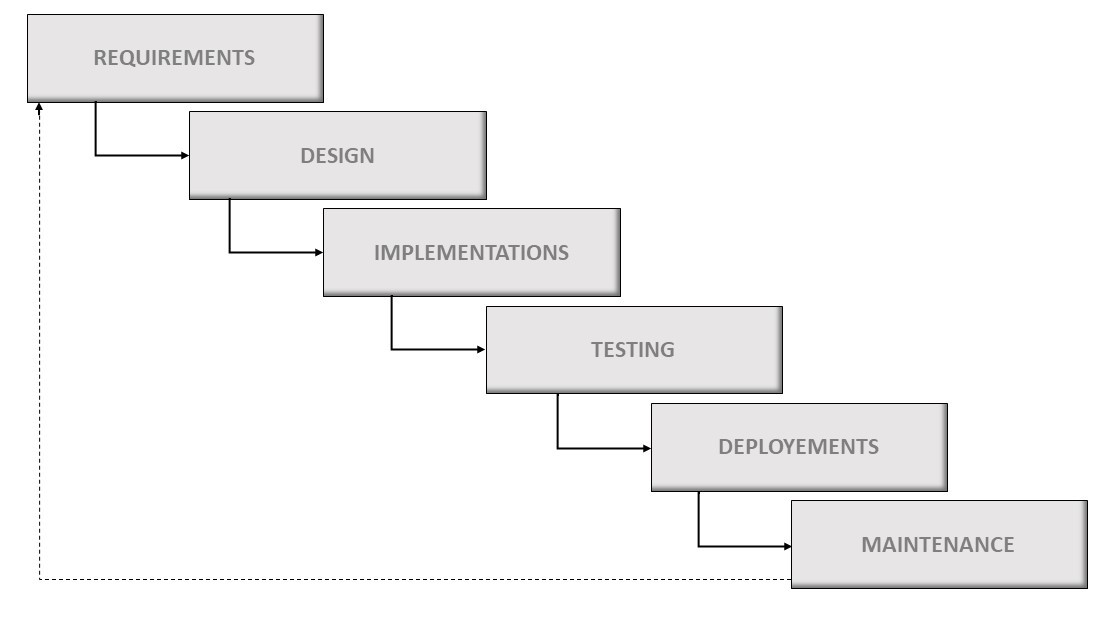
# 5.3. Relevance of RAD in Contemporary Software Development:

RAD is still relevant today because of its technology and importance on its quick growth. Early customer information is made easier by testing and getting results that the finished product lives up to consumer expectations. In order to match software solutions with changing user desires to it needs, developers and users should actively collaborate to achieve a good development process for the organisation like consumer user and developers. Futhermore, of its focus on rapid development and iteration, Rapid Application Development (RAD) is particularly well-suited for projects with dynamic or rapidly changing specifications. contemporary efforts in software development.

# 6. DESIGN OF RAD AND WATERFALL MODEL:

## 6.1. Waterfall Model:

The Waterfall model represents a structured and serialized methodology for software development that entails progressively advancing through a sequence of discrete phases, with each stage being fulfilled prior to transitioning to the subsequent one. This model encompasses a series of well-defined stages, including requirement gathering, system design, implementation, testing, deployment, and maintenance.



#### 6.1.1. Requirement Gathering:

The initial phase involves gathering and documenting the software requirements, including functional and non-functional requirements, user expectations, and project constraints.

#### 6.1.2. System Design:

In this phase, system architects and designers transform the requirements into a detailed system design. It includes defining the system architecture, software modules, interfaces, and data structures.

#### 6.1.3. Implementation:

Also known as coding, this phase involves translating the system design into actual code using programming languages and development tools.

#### 6.1.4. Testing:

Following the successful deployment of the software solution, a rigorous testing process ensures that all components function optimally and free from errors. In this stage, diverse forms of testing are conducted including unit tests, integration tests, system-level assessments, and acceptance testing to uncover and remedy any flaws or bugs that may have been introduced during development.

#### 6.1.5. Deployment:

After successful testing, the software is deployed or installed in the target environment, which could be production or a customer's location.

#### 6.1.6. Maintenance:

The final phase of the Waterfall model involves ongoing maintenance and support of the software. It includes activities such as bug fixes, updates, and enhancements.

The Waterfall model aimed to address the challenges of managing complex software projects by organizing the development process into well-defined stages **(ROYCE)**. However, it had limitations, particularly when requirements changed, or errors were discovered later in the process.

### 6.2. Advantages:

1. The waterfall model provides a clear structure and a well-defined set of stages for the development process. This makes it easier to plan and manage the project as it progresses through each phase.

2. If the project requirements are stable and well-understood from the beginning, the waterfall model can be a good fit. It allows for a linear and sequential approach where requirements are gathered upfront, and the subsequent phases build upon them.

3. The simplicity of the waterfall model makes it easy to understand and manage. Each phase has specific deliverables and milestones, facilitating better project tracking and control.

### 6.3. Disadvantages:

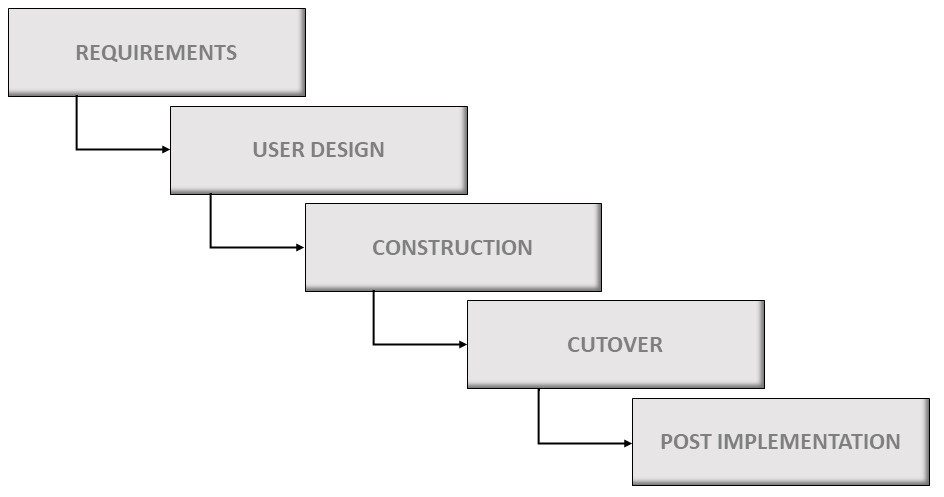
1. One significant drawback of the waterfall model is its lack of flexibility when it comes to changing requirements. Once a phase is completed, it becomes difficult to incorporate changes without going back and repeating previous stages. This can lead to challenges if the requirements evolve or if important changes are identified later in the project.

2. The waterfall model follows a sequential approach, where feedback from users or stakeholders is limited until the later stages, such as testing or deployment. This reduces the chances of incorporating valuable feedback early on and may result in a product that doesn't fully meet user needs.

3. The waterfall model is linear and progresses in a top-down manner, significant changes identified in the later stages can be costly and time-consuming to implement. This is because changes may require reworking previous phases or even starting from scratch, which can impact project timelines and budgets.

## 6.4. RAD Model:

Rapid Application Development (RAD) is a specific type of Agile methodology that places a strong emphasis on rapid development and iteration. It aims to deliver high-quality software solutions quickly and iteratively **(BOHEM)**.



The RAD model follows the following key stages:

#### **6.4.1. Requirements Planning:**

In this stage, the project's objectives and requirements are identified and analysed. The focus is on defining the scope of the project, identifying key functionalities, and setting development priorities.

#### **6.4.2. User Design:**

The user design stage involves close collaboration between developers and users. Developers create prototypes, mock-ups, or wireframes to visualize and discuss the intended user interface (UI) and user experience (UX). Early user feedback is incorporated to refine the design.

#### **6.4.3. Construction:**

Once the user interface design is finalized, development begins. RAD emphasizes rapid development through the use of various techniques, such as code reuse, component-based development, and rapid prototyping. The software is developed in small iterations or increments.

#### **6.4.4. Cutover:**

The cutover stage involves deploying the developed software into the target environment. This may include installation, configuration, data migration, and user training. It aims to ensure a seamless transition from the old system to the newly developed software.

#### **6.4.5. Post-Implementation:**

After the software is deployed, the RAD model emphasizes continuous evaluation and improvement. Feedback from users and stakeholders is gathered and incorporated into future iterations. This stage may involve bug fixes, performance optimizations, and feature enhancements.

The RAD model stands out for its strong reliance on prototyping as a key component. By creating prototypes at an early stage, developers can gather feedback from users and stakeholders, ensuring that the final product meets their needs and expectations more effectively (Bohem).

### **6.5. Advantages**

1. The Rapid Application Development (RAD) model emphasizes iterative development and prototyping, resulting in faster delivery of working software. This can be beneficial when there is a need for quick time-to-market.

2. The RAD model encourages active customer participation throughout the development process. This leads to better alignment with customer requirements and ensures that the final product meets their expectations.

3. RAD allows for frequent iterations and feedback loops, enabling developers to quickly address issues, incorporate changes, and adapt to evolving project requirements. This flexibility helps reduce the risk of delivering an unsatisfactory end product.

4. By involving stakeholders early and employing a collaborative approach, the RAD model helps identify potential issues and changes before significant resources are invested. This can mitigate costly rework and reduce overall project costs.

### **6.6. Disadvantages**

1. The rapid development cycles and continuous iterations of the RAD model can lead to complex project management. It requires effective coordination, communication, and skilled resources to ensure proper synchronization between different development teams.

2. RAD relies heavily on the availability of experienced and skilled developers. If such resources are not readily accessible, it can become challenging to implement the RAD model effectively.

3. Due to the iterative and flexible nature of RAD, there is a risk of scope creep, where additional requirements keep getting added during the development process. This can result in project delays and budget overruns if not managed properly.

4. While the RAD model is suitable for small to medium-sized projects, it may encounter difficulties when scaling up to larger projects with complex requirements. The rapid development cycles may not allow for sufficient planning and coordination required for larger-scale endeavors.

# 7. Waterfall Model Implementation

Technology quickly became an crucial part of our life especially in the past 5 years, Technology was able to turn our lives upside down in many aspects, One of the heavily influenced aspects by technology is digital business, Many companies and industries have now moved to E-commerce solutions in order to achieve more profits, According to (Putra *et al*., 2022), Indonesia was able to get a total income of about USD 27 billion in 2018 due to many business migrating to an online shop, Which shows the amount of opportunities that Indonesia has in the E-commerce concept, As outlined by (Putra *et al*., 2022), Google is estimating that in 10 years the E-commerce market will skyrocket to USD 100 billion, However, There are many business that are still to make the jump into the digital world, To illustrate, In Indonesia, There are plenty of Micro, Small, and Medium enterprises (MSMEs) that don’t really understand how to enhance their businesses by moving into the internet, And therefore, Missing out on maximizing their profits, One example of this is the “Dua Ruang” Coffee shop in Malang Indonesia, They still only have a physical store and are yet to enter the digital world, Therefore, (Putra *et al.*, 2022) decided to make a website for this Coffee shop that will use the Waterfall model as its methodology for developing the site.

The Waterfall model comprises different phases and stages, As known, The Waterfall model’s stages are flowing down like a waterfall and not one stage can be done without the completion of the previous one (Putra *et al*., 2022), The stages of the Waterfall model for this project are as follows:

### 7.1. Requirements Analysis:

This stage involved collecting information for the requirements of the website and analysing the data before moving on to the design stage, Data collection was made through several meetings between the development staff and the executives of the Coffee shop to discuss the requirements, Also the team conducted some studies to evaluate if there are any caps in terms of the development of the website (Putra *et al.*, 2022).

### 7.2. Design:

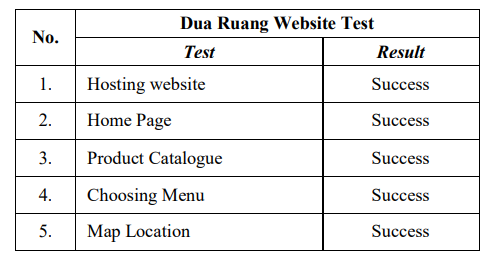
The aim of this stage is to settle for a design for the site, This can include the UI or user interface, And any other visuals and media that would be used to make the website user-friendly, After an agreement is reached for the design, Its time to move to the coding stage, Along with this, The design stage will involve the use of Use case diagrams and class diagrams to further understand how the website will operate and what functions it will have, In addition, There will be some prototypes of the website to show to the executives of the Coffee Shop and have their opinion on that (Putra *et al.*, 2022).

### 7.3. Development:

This stage is where the design of the system will be implemented using coding, As per (Putra *et al*., 2022), The Development team are split into three teams each with different tasks and they are as follows: Front-end, Back-end, And Full stack. The Front-end team is responsible for creating the user interface and the overall layout and style of the site, The Back-end team are responsible for the functionalities and features of the site, And the Full-stack team are people who work on both the Font and Back-end (Putra *et al.*, 2022).

### 7.4. Evaluation and testing:

This is the last step of the development of this web application, here many tests were carried out to check for any possible bugs in the system, or if there are any further enhancements that can be applied to the system before delivering the final product (Putra *et al.*, 2022).



Here we see some tests that were carried out by (Putra *et al*., 2022), We can see that the tests involved a lot of areas and different functionalities of the site, And it looks like all the tests carried out were a success and no faults or bugs were found which is an excellent and this means the website is finally ready for the final delivery (Putra *et al*., 2022)

# 7.5. RAD Model Implementation

(Sasmito *et al*., 2020) wanted to develop an application to enhance the community’s awareness of industrial information and to open up more opportunities for investors to take a look at possible investments in the present industries, This particular app was needed especially in Indonesia due to the unreliable method of collecting industrial information in the country, As well as the vast amount of hidden industries that the public are blinded from, A Geographic Information System (GIS) will be developed as a web application that will assess, analyse, and collect data based on the geographic location, And it will utilize the RAD model as its development method (Sasmito *et al*., 2020), The main reason behind choosing the RAD model is that it offers time-efficiency, cost-effectiveness ,and superb quality assurance of the development phase, To illustrate, the RAD model has also been implemented to develop Library Information Systems, The entire process took about 60 working days to come live, which is a considerably minimal time keeping in mind that it has to go through many phases including: Planning, Analysis, Development, and Testing, And when the application was live to the public, It became evident that business has nourished and the delivery of services has improved drastically (Sasmit *et al*., 2020).

The GIS was implemented across different stages and phases, starting with the development stages and they are as follows:

1. System Design: (Sesmito *et al*., 2020) ended using UML of Unified Modelling Language for the initial design, this phase comprised of using use case diagrams to show the requirements of the application, and also sequence diagrams were used to outline how is the application communicating internally and with the user as well.

1. User Interface: for the GIS application, PHP was the programming language used to develop the application, and it was combined with a Database Management System (DBMS) which is MySQL to handle the storing and retrieving of industrial information (*Sasmito et al*., 2020).

Now the RAD model is implemented for the GIS as follows:

### 7.5.1. Planning:

The main focus of this stage was for them to identify certain obstacles that can arise during the implementation and also to collect all the necessary data from end-users such as the industries themselves in the relevant area, and this was done through multiple meetings between the users and the respective team to outline the objectives and discuss any unexpected setbacks and how to recover from such scenarios (Sasmito *et al*., 2020).

### 7.5.2. Analysis:

During this stage, The primary data collected earlier in the planning stage was analysed before going into implementation, The analysis procedure differed from one user to another, To illustrate, The users where categorized into four different users, All of them expect one where given the privileges to control their data on the GIS, and this has influenced the functionalities of certain users of the GIS (*Sasmito et al.*, 2020)

### 7.5.3. Design:

After the analysis phase was finished, based on the analysed data, A design must be created, and what (Sasmito *et al*., 2020) ended up doing was going with basic UML for the design of the GIS, this design was communicated to the users and had approval from all of them (Sasmito *et al*., 2020).

### 7.5.4. Implementation:

For this stage, The design of the system will now be transformed into actual code, As mentioned earlier, (Sasmito *et al*., 2020) are using PHP for the website main programming language and also MySQL for the database, according to (Sasmit *et al*., 2020), The GIS app is able to showcase some data such as: Industry profiles, Number of worker, and production capacity, After that, The application will now move on to the testing phase where (Sasmito *et al*., 2020) did some testing on the overall functionalities of the site including: Sign in, User data, and District data. The aim of this stage was to check that all create, read, update, and delete (CRUD) functions are operating as expected and there are no bugs and issues in performing any of these tasks, Each prototype was shown to the users and any necessary changes to achieve their requirements has been implemented (Sasmit *et al*., 2020).

The GIS web application that was implemented for Tengal city of Indonesia took around 84 days (about 3 months) to finish entirely, As outlined be (Sasmito *et al.*, 2020), The application was implemented 5 week faster than if it was developed with the Waterfall model, This is specifically due to the superb coordination between team leaders and the development team as well as the excellent resource management and the full understanding of completion times of each phase, According to (Saker *et al*., 2015), This is a vital part when it comes to implementing the RAD model as it requires extraordinary communication skills between the team and each individual member must have top notch skills to deliver the project within a strict and relatively short time.

# **8. Advocates and Detractors**

## **8.1. Advocate**

Showing the complexity of managing the Net Promoter Score (NPS), one advocate compares it to a thermometer, highlighting its relationship with customer satisfaction. Despite being careful about directly attributing improvements to the actions taken. This advocate acknowledges the NPS's role in signalling positive changes over time. They distinguish between satisfaction, seen as fulfilling what is expected, and the NPS, which captures the "wow" factor. While also recognizing the limitations it has, this advocate views the NPS as a valuable tool for keeping track of changes in customer perception. (Gerea *et al.,* 2022)

Another advocate acknowledges the limitations of NPS as an actionable metric but stresses its significance in measuring brand experience. They share an example of a serious website change, thus illustrating the impact of a learning curve on initial NPS results. This advocate highlights the importance of revisiting NPS measurements over time to capture the evolving point of views as users adapt to changes. (Gerea *et al.,* 2022)

## 8.2. Detractors

On the other hand, a detractor questions the NPS's ability to truly reflect the impact of actions on customers satisfaction. They express scepticism about claiming responsibility for increasing the NPS, citing instances where users, despite struggling during a test, provided a high score. This detractor truly shows the subjective nature of the NPS, suggesting that it may capture the immediate feeling of users rather than a thorough evaluation of their experience. (Gerea *et al.,* 2022)

## 8.3. Unified CX Matrix

Advocates of united customer experience (CX) metrics propose that establishing common goals across teams is crucial. They envision a future where the objectives of all departments align for a seamless and consistent customer experience. This unified approach is seen as a transformative shift from stored operations to a more collaborative and customer centric organisational structure. (Gerea *et al.,* 2022)

## 8.4. Omnichannel Approach

Highlighting the need for integrated systems, advocates can realise the challenge posed by stored structures in transitioning to an omnichannel approach. The argument they bring to the table is that a modern and integrated Customer Relationship Management (CRM) system is necessary for delivering a united customer experience. (Gerea *et al.,* 2022)

## 8.5. Human Talent

The transition from multichannel to omnichannel introduces challenges that relate to human talent. Leadership and channel ownership are identified as key aspects that demand a shift in organizational and cultural practices. Managers must prioritize omnichannel strategies, while teams need to navigate changing roles and responsibilities in the context of channel integration. (Gerea *et al.,* 2022)

In summary, advocates highlight the NPS's role in keeping track of improvements over time and foresee a future with unified CX metrics, while detractors’ express scepticism about the NPS's ability to reflect real changes in customer satisfaction. Enablers such as integrated systems and a willingness to invest are identified as crucial for successful omnichannel transitions, with challenges primarily centred around the need for a shift in human talent and organizational culture. (Gerea *et al.,* 2022)9. Conclusion

In conclusion, when embarking on the development of new software, choosing the right System Development Life Cycle (SDLC) model is crucial. The SDLC models serve as a set of instructions and steps that must be followed to develop software, with each model offering a unique set of phases and approaches. In this research, a comparison is conducted between the Rapid Application Development (RAD) and Waterfall models.

The Waterfall model is a sequential approach that follows a linear and structured process, with distinct phases such as requirements gathering, system design, implementation, testing, deployment, and maintenance. It is known for its rigidity and emphasis on extensive documentation. Although it has limitations, it has played a significant role in software development and has evolved over time. Modifications, such as the V-Model and integrating feedback loops, have been introduced to address its drawbacks. Additionally, agile methodologies, like Scrum or Kanban, have emerged, prioritizing flexibility and collaboration.

RAD, on the other hand, is a specific type of Agile methodology that focuses on rapid development and iteration. It places a strong emphasis on collaboration between developers and users, incorporating early user feedback and prototyping. RAD has been implemented successfully in various projects, promoting quick time-to-market and improved service delivery.

The history, key points, and evolution of the Waterfall model and RAD provide valuable insights for software developers and organizations. By understanding the strengths and weaknesses of each model and considering the specific requirements and goals of a software project, informed decisions can be made regarding the most suitable methodology to adopt.

References:

1. Sasmito, G. W. Wibowo, D. S. Dairoh. (2020) Implementation of Rapid Application Development Method in the Development of Geographic Information Systems of Industrial Centers. *Journal of information and communication convergence engineering* [online]. **18**(3), pp.194-199. Available from: <https://koreascience.kr/article/JAKO202029565519509.pdf> [Accessed 1st December 2023]
2. Saker, I. H. Faruque, F. Hossen, U. Rahman, A. (2015) A Survey of Development Process Models in Software Engineering. *International Journal of Software Engineering and its Applications* [online]. **9**(11), pp.55-70. Available from: <https://www.researchgate.net/profile/Iqbal-Sarker/publication/298656663_A_Survey_of_Software_Development_Process_Models_in_Software_Engineering/links/57e20f1e08ae9e25307d5214/A-Survey-of-Software-Development-Process-Models-in-Software-Engineering.pdf> [Accessed 2nd December 2023]
3. Royce, W. W. (1970). Managing the development of large software systems. Proceedings of IEEE WESTCON, 26, 1-9. <https://dl.acm.org/doi/10.5555/41765.41801> [Accessed 30th November 2023]

1. Martin, R. C., & Martin, M. (2003). Agile Software Development: Principles, Patterns, and Practices. Prentice Hall. <https://dl.acm.org/doi/abs/10.5555/515230> [Accessed 3rd December 2023]

1. Beck, K., Beedle, M., van Bennekum, A., Cockburn, A., Cunningham, W., Fowler, M., ... &amp; Thomas, D. (2001). Manifesto for Agile Software Development. Agile Alliance. <https://agilemanifesto.org/> [Accessed 1st December, 2023]

1. Boehm, B.W., &amp; Papaccio, P.N. (1988). Understanding and Controlling Software Costs. IEEE Transactions on Software Engineering, 14(10), 1462-1477. <https://doi.org/10.1109/32.6191>
2. Sasmito, B., et al. (2020). The rapid application development model for the development of geographic information system web application in Tegal City. Journal of Physics: Conference Series.
3. Saker, S., et al. (2015). Implementation of RAD model and agile using scrum methodology as a solution to the metacarpal fracture. Journal of Medicine and Life, 8(Spec. Iss. 3), 236-241.
4. Ardhiansyah, A. Y. Putra, D. L. S. Kristanto, J. S. Budhianto, N. P. Maulana, F. I. (2022) ‘Waterfall Model for Desgin and Development of Caffee Shop Website at Malang’, *2022 International Conference on Informatics, Multimedia, Cyber and Information System.* Jakarta Indonesia, 16-17 November 2022. Jakarta: IEEE, pp. 230-233.
5. Carmen Gerea, Valeria Herskovic (2022) ‘Transitioning from Multichannel to Omnichannel Customer Experience in Service-Based Companies: Challenges and Coping Strategies’, Department of Computer Science, Pontificia Universidad Católica de Chile, Santiago 7820436, Chile, 29 March 2022. https://www.mdpi.com/0718-1876/17/2/21