ResearchGate

See discussions, stats, and author profiles for this publication at: https://www.researchgate.net/publication/353324450

Waterfall Methodology, Prototyping and Agile Development

Technical Report · June 2021

DOI: 10.13140/RG.2.2.17918.72001

CITATIONS
2

READS
6,346

1 author:

Udesh S. Senarath
Rajarata University of Sri Lanka
11 PUBLICATIONS **2** CITATIONS

SEE PROFILE

Some of the authors of this publication are also working on these related projects:

Mobile Applications in China View project

Software Development Life Cycle View project

All content following this page was uploaded by Udesh S. Senarath on 18 July 2021.

The user has requested enhancement of the downloaded file.

Waterfall Methodology, Prototyping and Agile Development

By

Udesh S. Senarath

22nd June 2021

CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ACRONYMS

FTP - File Transfer Protocol

HRM – Human Resource Management

HTTP - Hypertext Transfer Protocol

SDLC – Software Development Life Cycle

SQL - Structured Query Language

SRS - Software Requirements Specification

SSH - Secure Shell

Answer for the Question One

1.1 Question

There are different approaches to software development. Discuss the similarities and differences between the **waterfall and prototyping** lifecycle models.

1.2 Answer

1.2.1 Background of Software Development

Each project holds a scheme that follows a system to achieve project completion. Such as a software project, that also has many systems or development methods according to the nature or size of the project. However, a difficult decision that takes time for the project manager is selecting the most suitable method for the project. The project manager selects the development technique, considering about the time and budget of the project in specific for success. Many software development methodologies have been introduced to the world each day and collecting all the knowledge regarding software development practices is difficult task for a project manager. The technological age is rapidly changing year after year. There are many methods for developing computer programs and the appropriate method for the desired project is problematic to regulate. The Developer, Project Manager deals with the most imperative challenges and solutions for software development. There are benefits of a project management process software development that can help to manage the burden of management. The area of software development is a broad area and propagates with new and future standards and technologies every day. Programming languages are introduced on the market almost every month using new versions and frameworks. New programming language changes also offer new features and technologies that facilitate and occasionally change your project everywhere. For this purpose, the project manager must monitor future varieties of the programming languages in order to meet the requirements of the project owner and to coordinate with the project team. Designed with strong technical proficiency by vastly trained and trained software development teams. Highly trained and trained people also need the highest return from their work and are based on hourly or daily work. The project manager will, therefore, take into account the price costs for each business day and the estimated time to complete the project by budget control. People with high qualifications do not like to join a better team because they tend to cherish their own work and self-absorption. Although low - skilled people mean that it takes more time to complete the project and less synchronization in relation to the project, but with lower employment rates. The project manager is therefore prepared to hire highly trained

people in less time to prepare an effective project. The project manager must be able to use a highly competent person's skills in the best way to complete the project by coordinating it with the team with any differences between them. Members of the program development team are often coordinated from all over the world. To communicate face - to - face, software development is not required. Many online management tools are offered to accomplish tasks and track tasks, such as base camp, Click-up, Zendesk, and asana. Great exchange of files, such as Live Drive, Google Drive, Dropbox, etc. Meetings can be held online using Skype, Zoom or MS Teams and instant messaging using WhatsApp, Viber, Telegram, etc.

1.2.2 Stages of Software Development

1.2.2.1 Preliminary Investigation (Research)

It is the beginning of almost all software development projects. At this stage, the project owner, project manager and project team meet and exchange information about the project. The entrepreneur fulfils his or her objectives by searching for markets for those persons or organizations with similar goals to help with the budget or any other purpose, then formulating them in documentary form and then investigating a company that has the ability to achieve this goal in the time and time required budget. The project owner delivers his / her exact objectives to the project manager. The project manager is responsible for receiving the project owner's requirements in full, evaluating them and moving to the project team with technical specifications. The project manager must take care of both the business and technical perspective. The project team is responsible for meeting the requirements from a technical perspective. The project team should investigate the programming language, framework, libraries, version creation tools and infrastructure needed to build the software project according to requirements.

1.2.2.2 System Analysis (Planning)

This is the stage in which elements are assembled and organized in a way to complete the program product. The large project must be divided into a small flow and easy management of subgroups. The project manager places all subsets, functions, and database at the front of the project team to focus on the appropriate technology to achieve the objective and should decide on the best management methodology for use and the protocol to be followed to complete the project, In the budget and in the range.

1.2.2.3 System Design (Design)

At this point, the application design is created. Mobile and web applications make the design more effective than desktop applications. The design is entirely dependent on the nature of the project, project, function, and purpose. Like the banking application, they have less design and

specific design, whereas the museum's web application needs excellent graphic designs to attract people. This stage is very important because at this stage the application design preview is displayed to the project owner so that he decides to finish it or change it. The entrepreneur comes with some will, and they must be added to the research and planning as well and after the implementation of this function in the project.

1.2.2.4 Software Development (Development)

The implementation of the program is already evolving this stage. This stage has two surroundings. The development and test environment always simultaneously use the same protocol. The code needs to be written in the development environment and these codes need to be loaded into the test environment using the same synchronization protocol. The main aspect of this phase is the monitoring of progress and the project manager's implementation. The project manager monitors the update progress and updates the project owner on the project progress. Developers always carry out a debugging process to help remove project errors and load error-free test environment codes. Developers also write comments during encryption and make it easier for other developers to understand them.

1.2.2.5 System Testing (Testing)

At this stage of programming and design, errors were found and fixed. Testing the function of each function and seeing the result find programming errors. If the output results are assumed to be incorrect or the applications fail or behave in a way that is not supposed to be, they are programming errors. Data from the application or hackers are easily stolen access to the application and a programming error also occurs. Although the project owner should be notified of the design error, the project owners know the requirements that project managers must meet and what the project team does. If an error occurs, these errors occur during the planning phase. To determine the design error, the project owner must be involved because he is the one who formulated the requirements.

1.2.2.6 System Implementation (Setup)

The application is installed in the direct environment during the configuration phase. The actual setup includes source code, database, etc., almost everything used to compile programs where applications from third parties are required, APIs, etc. The application also undergoes a different test cycle when it is fully installed in the real environment. After testing, content is added to the application.

1.2.2.7 System Maintenance (Maintenance)

This phase covers the development of programs after training and the implementation of the schedule. By monitoring the firewall, mail logs, HTTP, FTP, MySQL, and SSH errors make sure the application is running properly. Monitor traffic and input data.

These early stages represent the cornerstone of all software development projects agreed upon by software development communities. Depending on the software development methodology, since phase names are changed in some methodologies, others are mixed and others overlap. Waterfall Software Development Methodology, Prototyping Software Development Methodology, Iterative and Incremental Software Development Methodology, Spiral Software Development Methodology, Rapid Application Development Software Development Methodology, Extreme Programming Software Development Methodology, V-Model Software Development Methodology, Scrum Software Development Methodology, Cleanroom Software Development Methodology, Dynamic System Development Methodology, Rational Unified Process of Software Development, Lean Software Development Methodology, Test-Driven Development Methodology, Behavior-Driven Development Methodology, Feature-Driven Development Methodology, Model-Driven Engineering Methodology, Crystal Methods of Software Development, Joint Application Development Methodology, Adaptive Software Development Methodology, Open Source Software Development, and Microsoft Solutions Framework can be figured as different approaches to software development used by the all around the world from the very beginning – up until now.

Among the aforementioned software development methodologies Waterfall Software Development Methodology and Prototyping Software Development Methodology can be found as widely acknowledged software development approaches and below is the comparative analysis about the Waterfall Software Development Methodology and Prototyping Software Development Methodology.

1.2.3 Waterfall Software Development Methodology

The Waterfall Model was first Process Model to be introduced. It is also referred to as a linear sequential life cycle model. It is very simple to understand and use. In a waterfall model, each phase must be completed before the next phase can begin and there is no overlapping in the phases. The waterfall Model illustrates the software development process in a linear sequential flow; hence it is also referred to as a linear-sequential life cycle model. This means that any phase in the development process begins only if the previous phase is complete. In waterfall

model phases do not overlap. This model is used widely in Software Engineering to ensure success of the project. In "The Waterfall" approach, the whole process of software development is divided into separate phases and typically, the outcome of one phase acts as the input for the next phase sequentially. Following is a diagrammatic representation of different phases of waterfall model.

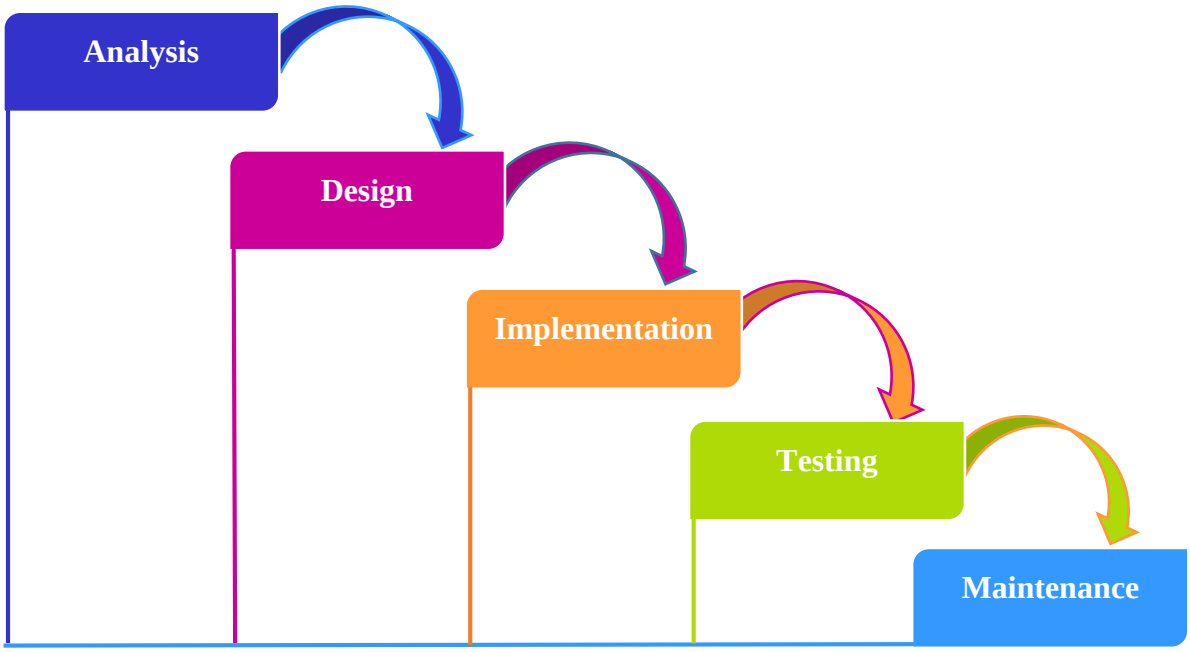

Figure 1 Waterfall Software Development Methodology

Essentially, the Waterfall model comprises five phases: Analysis, design, implementation, testing, and maintenance.

1.2.3.1 Phases of the Waterfall Model

1.2.3.1.1 Analysis Phase

Often known as Software Requirements Specification (SRS) is a complete and comprehensive description of the behavior of the software to be developed. It implicates system and business analysts to define both functional and non-functional requirements. Usually, functional requirements are defined by means of use cases which describe the users' interactions with the software. They include such requirements as purpose, scope, perspective, functions, software attributes, user characteristics, functionalities specifications, interface requirements, and database requirements. In contrast, the non-functional requirements refer to the various criteria, constraints, limitations, and requirements imposed on the design and operation of the software rather than on particular behaviors. It includes such properties as reliability, scalability, testability, availability, maintainability, performance, and quality standards.

1.2.3.1.2 Design Phase

It is the process of planning and problem solving for a software solution. It implicates software developers and designers to define the plan for a solution which includes algorithm design, software architecture design, database conceptual schema and logical diagram design, concept design, graphical user interface design, and data structure definition.

1.2.3.1.3 Implementation Phase

It refers to the realization of business requirements and design specifications into a concrete executable program, database, website, or software component through programming and deployment. This phase is where the real code is written and compiled into an operational application, and where the database and text files are created. In other words, it is the process of converting the whole requirements and blueprints into a production environment.

1.2.3.1.4 Testing Phase

It is also known as verification and validation which is a process for checking that a software solution meets the original requirements and specifications and that it accomplishes its intended purpose. In fact, verification is the process of evaluating software to determine whether the products of a given development phase satisfy the conditions imposed at the start of that phase; while, validation is the process of evaluating software during or at the end of the development process to determine whether it satisfies specified requirements. Moreover, the testing phase is the outlet to perform debugging in which bugs and system glitches are found, corrected, and refined accordingly.

1.2.3.1.5 Maintenance Phase

It is the process of modifying a software solution after delivery and deployment to refine output, correct errors, and improve performance and quality. Additional maintenance activities can be performed in this phase including adapting software to its environment, accommodating new user requirements, and increasing software reliability.

All these phases are cascaded to each other in which progress is seen as flowing steadily downwards like a waterfall through the phases. The next phase is started only after the defined set of goals are achieved for previous phase and it is signed off, so the name "Waterfall Model". In this model phases do not overlap.

1.2.3.2 Waterfall Model Application

Every software developed is different and requires a suitable SDLC approach to be followed based on the internal and external factors. Some situations where the use of Waterfall model is most appropriate are:

- Requirements are very well documented, clear and fixed.
- Product definition is stable.
- Technology is understood and is not dynamic.
- There are no ambiguous requirements.
- Ample resources with required expertise are available to support the product.
- The project is short.

1.2.2.3 Advantages and Disadvantages of Waterfall Model

An advantage of waterfall development is that it allows for departmentalization and control. A schedule can be set with deadlines for each stage of development and a product can proceed through the development process model phases one by one. Development moves from analysis, through design, implementation, testing, and ends up at maintenance. Each phase of development proceeds in strict order. While the waterfall development model does not allow for much reflection or revision. Once an application is in the testing stage, it is very difficult to go back and change something that was not well-documented or thought upon in the concept stage. That is a main disadvantage. In addition to those main advantages and disadvantages following can also be identified;

Table 1 Advantages & Disadvantages of waterfall model

Advantages	Disadvantages
• Simple and easy to understand and use.	• No working software is produced until late during the life cycle.
• Easy to manage due to the rigidity of the model. Each phase has specific deliverables and a review process.	• High amounts of risk and uncertainty. Integration is done as a "big-bang" at the very end, which doesn't allow identifying any technological or business bottleneck or challenges early.
• Phases are processed and completed one at a time.	• Not a good model for complex and object-oriented projects.

Table 1 Advantages & Disadvantages of waterfall model (continued)

• Works well for smaller projects where requirements are very well understood.	• Poor model for long and ongoing projects.
• Clearly defined stages, well understood milestones, easy to arrange tasks and process and results are well documented.	• Not suitable for the projects where requirements are at a moderate to high risk of changing. So risk and uncertainty is high with this process model.

1.2.4 Prototyping Software Development Methodology

The Prototyping model is also a popular software development life cycle model in which prototype is built, tested, and reworked until an acceptable prototype is achieved. It also creates base to produce the final system or software. It works best in scenarios where the project's requirements are not known in detail. It is an iterative, trial and error method which takes place between developer and client. The prototyping model can be considered to be an extension of the Iterative Waterfall model. This model suggests building a working Prototype of the system, before the development of the actual software. A prototype is a toy and crude implementation of a system. It has limited functional capabilities, low reliability, or inefficient performance as compared to the actual software. A prototype can be built very quickly by using several shortcuts by developing inefficient, inaccurate or dummy functions. Rapid Throwaway prototypes, Evolutionary prototype, Incremental prototype, and Extreme prototype are the popular types of prototyping methods.

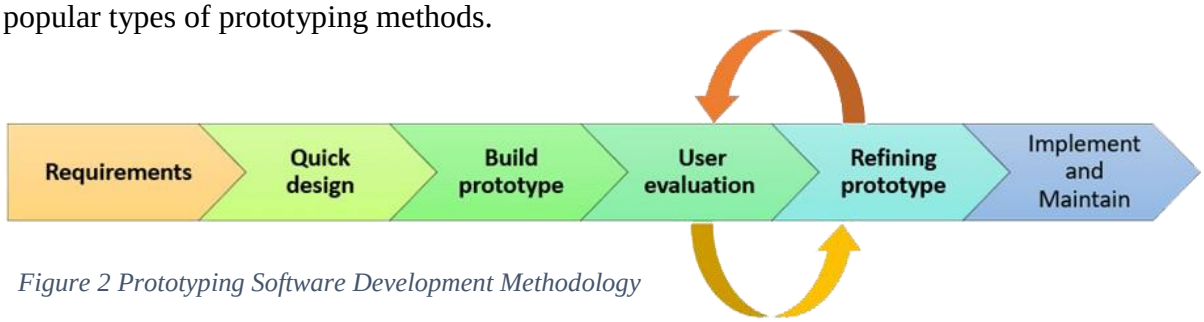

Figure 2 Prototyping Software Development Methodology

1.2.4.1 Phases of the Prototyping Model

1.2.4.1.1 Requirements gathering and analysis

A prototyping model starts with requirement analysis. In this phase, the requirements of the system are defined in detail. During the process, the users of the system are interviewed to know what is their expectation from the system.

1.2.4.1.2 Quick design

The second phase is a preliminary design or a quick design. In this stage, a simple design of the system is created. However, it is not a complete design. It gives a brief idea of the system to the user. The quick design helps in developing the prototype.

1.2.4.1.3 Build a prototype

In this phase, an actual prototype is designed based on the information gathered from quick design. It is a small working model of the required system.

1.2.4.1.4 Initial user evaluation

In this stage, the proposed system is presented to the client for an initial evaluation. It helps to find out the strength and weakness of the working model. Comment and suggestion are collected from the customer and provided to the developer.

1.2.4.1.5 Refining prototype

If the user is not happy with the current prototype, you need to refine the prototype according to the user's feedback and suggestions.

This phase will not over until all the requirements specified by the user are met. Once the user is satisfied with the developed prototype, a final system is developed based on the approved final prototype.

1.2.4.1.6 Implement product and maintain

Once the final system is developed based on the final prototype, it is thoroughly tested and deployed to production. The system undergoes routine maintenance for minimizing downtime and prevent large-scale failures.

1.2.4.2 Software Prototyping Application

Software Prototyping is most useful in development of systems having high level of user interactions such as online systems. Systems which need users to fill out forms or go through various screens before data is processed can use prototyping very effectively to give the exact look and feel even before the actual software is developed.

Software that involves too much of data processing and most of the functionality is internal with very little user interface does not usually benefit from prototyping. Prototype development could be an extra overhead in such projects and may need lot of extra efforts.

1.2.4.3 Advantages and Disadvantages of Prototyping Model

Software prototyping is used in typical cases and the decision should be taken very carefully so that the efforts spent in building the prototype add considerable value to the final software developed.

Table 2 Advantages & Disadvantages of prototyping model

Advantages	Disadvantages
• Increased user involvement in the product even before implementation.	• Risk of insufficient requirement analysis owing to too much dependency on prototype.
• Since a working model of the system is displayed, the users get a better	• Users may get confused in the prototypes and actual systems.

Table 2 Advantages & Disadvantages of prototyping model (continued)

understanding of the system being developed.	
• Reduces time and cost as the defects can be detected much earlier.	• Practically, this methodology may increase the complexity of the system as scope of the system may expand beyond original plans.
• Quicker user feedback is available leading to better solutions and Confusing or difficult functions can be identified.	• Developers may try to reuse the existing prototypes to build the actual system, even when it is not technically feasible.
• Confusing or difficult functions can be identified.	• The effort invested in building prototypes may be too much if not monitored properly.

1.2.5 Comparison of Waterfall and Prototyping Lifecycle Models

1.2.5.1 Similarities of Waterfall Model and Prototype Model

- Each lifecycle is divided into phases where specific objectives are to be met.
- Both models have an objective to reduce the development and maintenance cost.
- Both models are suitable for short-term program or short-life span system.
- In many cases the modes are used interchangeably such as to identify the customer requirement project managers and system analysts use prototyping model and once the requirements are clearly identified, they use waterfall model to build the actual system.
- Bothe modes are simple and easy to understand the phases involve in the lifecycle.
- There is no very big difference in the flexibility of the models because once the requirements are identified and system specifications are set, they cannot be changed once the development is started.
- Fully functional system is delivered in the final stage of each methodology.
- System testing is done after the implementation of the system.
- Customer can see the final, actual product at the end of the life cycle.
- Prototyping Software Development Methodology can be named as an extended version of Waterfall Software Development Methodology because in the prototyping the requirements analysis is done by implementing a prototype while in the waterfall model it is done referring to document or by employing data collection techniques. But once the

requirements are fixed both follows similar types of steps up until implementation and maintenance.

1.2.5.2 Differences Between Waterfall Model and Prototype Model

Table 3 Differences Between Waterfall Model and Prototype Model

Waterfall Model	Prototype Model
All the requirements are specified in the first phase of the lifecycle.	Requirements are identified through out a separate process by using prototypes.
User only see the product as final output.	User can have a rough idea about the final output during the prototyping process.
Users only involve the process at the beginning and therefore the user involvement is comparatively low.	The user involvement is very high in identifying and developing the project.
There can be relatively high maintenance after delivering the final software as a result of lower user involvement.	The maintenance can be relatively in a minimum level because the user involvement is considerably high.
Waterfall model is a software development model and works in sequential method.	Prototype model is a software development model where a prototype is built, tested and then refined a s per customer needs.
It is best suited when the customer requirements are clear.	It is best suited when the requirement of the client is not clear and supposed to be changed.
The complexity of an error increase as the nature of the model each phase is sequential of the other.	The complexity of an error is low as the prototype enables the developer to detect any deficiency early at the process.
More experienced and knowledge employees are need to carry out the methodology.	Medium level of knowledge and experience are required.
Risk of system failure is comparatively high.	Risk of system failure is relatively low as a result of better identification or user requirement.
More documents are required for the process and at the end of the process more documents will be produced.	Document requirement is medium to low at the beginning of the process and the end of the process because the priority is given for the prototype.
Resource and cost control can be achieved hence the wastage is low because everything is pre decided and working according to a well prepared plan.	The entire prototyping can add more and more costs to the process therefore the cumulative cost can be high at the end of the process and the resource wastage is also comparatively high because each wrong prototype is a west of resource.

Answer for the Question Two

1.1 Question

Your organization has decided to use agile approach having previously used waterfall approach in systems development. Discuss the possible impact of this decision upon systems developers and users in the organization.

1.2 Answer

1.2.1 Agile Software Development Methodology

Agile means the ability to produce and respond to the dynamic environment. It is a way of dealing with, and ultimately succeeding in, an uncertain and raging environment. The authors of the Agile Manifesto chose "Agile" as the label for this whole idea because that word represented the adaptability and response to change which was so important to their approach. Agile development refers to any development process that is aligned with the concepts of the Agile Manifesto. The Manifesto was developed by a group of fourteen leading figures in the software industry, and reflects their experience of what approaches do and do not work for software development. Agile Methodology is really about thinking through how you can understand what is going on in the environment that you are in today, identify what uncertainty you are facing, and figure out how you can adapt to that as you go along.

Agile software development refers to a group of software development methodologies based on iterative development, where requirements and solutions evolve through collaboration between self-organizing cross-functional teams. The meaning of the concept is to build a software incrementally using short iterations of 1 to 4 weeks so that the development process is aligned with the changing business needs. Agile methods or Agile processes generally promote a disciplined project management process that encourages frequent inspection and adaptation, a leadership philosophy that encourages teamwork, self-organization and accountability, a set of engineering best practices intended to allow for rapid delivery of high-quality software, and a business approach that aligns development with customer needs and company goals. Instead of a single-pass development of 6 to 18 months where all the requirements and risks are predicted upfront, Agile adopts a process of frequent feedback where a workable product is delivered after 1 to 4 weeks' iteration. A graphical representation of the Agile Methodology is given bellow.

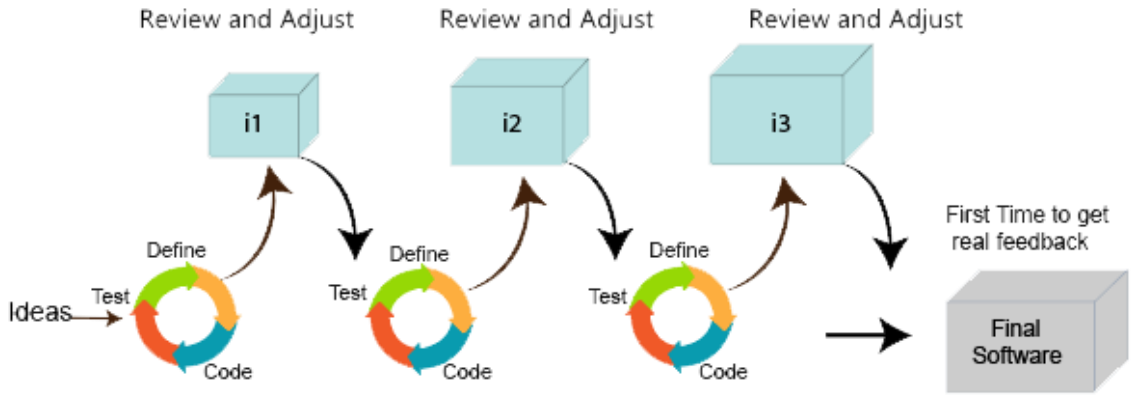

Figure 3 A graphical representation of the Agile Methodology

1.2.1.1 Scrum Method

Scrum is an agile method for project management or framework used primarily for software development projects. Its goal is to dramatically improve productivity in teams and to deliver new software every 2-4 weeks. Agile Scrum methodology involves forming teams with diversity, strong communicative teamwork, frequent feedbacks from clients, a lot of participates of users etc. It is a lightweight process framework for agile development, and the most widely-used one. Scrum is most often used to manage complex software and product development, using iterative and incremental practices. Scrum significantly increases productivity and reduces time to benefits relative to classic "waterfall" processes. Scrum processes enable organizations to adjust smoothly to rapidly-changing requirements, and produce a product that meets evolving business goals.

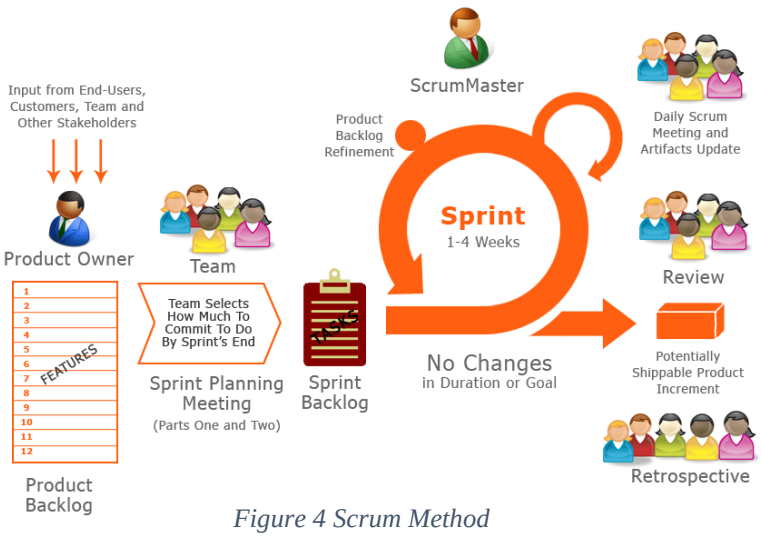

Figure 4 Scrum Method

1.2.1.2 Roles in Agile Development

With the adaptation of the agile methodology to a software development company, the human resource requirement and formations need to be replaced. In traditional software development frameworks such as waterfall methodology each software development phase is controlled by different type of people having separate set of skills but when it comes to agile development, different kind of people who are having different types of skills have to work collaboratively in a same project. Therefore, the human skills of each employee is very important while Human skill referred to the ability to work and deal with human resource. However, there are mainly tree leading roles involve in agile (scrum) methodology. They are: Scrum Master, Product Owner and Cross-Functional Team.

1.2.1.2.1 Scrum Master

A scrum master is a team leader and facilitator who helps the team members to follow agile practices so that they can meet their commitments. The responsibilities of a scrum master are,

- Enable close co-operation between all roles and functions.
- Remove any block.
- Shield the team from any disturbances.
- Work with the organization to track the progress and process of the company.
- Facilitate team meetings and decision-making process.
- Ensure the agile inspects and adapt processes are leverage properly which includes:
    - Daily Stand-ups
    - Planned Meetings
    - Demo
    - Review
    - Retrospective Meetings

1.2.1.2.2 Product Owner

Product owner is the one who drives the product from business perspective. The responsibilities of a product owner are as follows:

- Identify the requirements and prioritize their values.
- Determine the relies date and contents.
- Take an active role in iteration planning and releasing planning meetings.
- Ensure the team is working on the most valued requirements.

- Represent the voice of the team.
- Accept the user stories that meets the definition of done and defined acceptance criteria.

1.2.1.2.3 Cross-Functional Team

Every agile team should be a self-sufficient team with 5 to 9 members and an average experience ranging from 6 to 10 years. Typically, an agile team comprises of 3 to 4 developers, 1 tester, 1 technical lead, 1 product owner and 1 scrum master.

Product owner and Scrum master are a part of team interface, where other members are part of technical interface.

1.2.1.3 Impact of Adopting the Agile Methodology

1.2.1.3.1 Impact on the Software Developers of the Organization

Team members enjoy development work, and like to see their work used and valued. Scrum benefits team members by reducing non-productive work such as writing specifications or other artifacts that no one uses, and giving them more time to do the work they enjoy. Team members also know their work is valued, because requirements are chosen to maximize value to customers. Another aspect is that the whole development team is not meeting the customer directly and they only do what the product owner planned. The product owner is the person who should do all the meetings with customers and that helps to developers to focus more on the development work rather than meeting customers and working on requirement analysis and all the other non-development activities. In fact, developers are not very much good at dealing with customers and doing the Human Resource Management (HRM) part. They are more focus to development environment. With this agile development methodology, developers are not facing to the customers and instead the product owner does that for them. Developers are getting well identified, and carefully managed requirements and customer feedbacks which makes the development process more effective and accurate. However, if the organization is having some new developers, they might need to be replaced because the cross functional teams are looking for more experienced set of developers in order to make the iteration (sprint) fast and accurate. Rather than replacing the existing developers, organization can train the new developers and make them more qualified and that can be a pulse point to both organization point of view and developers perspective to increase the productivity of the developers and through that the company can take the 100% advantages from the adaptation of Agile Methodology.

1.2.1.3.2 Impact on the Customers of the Organization

Customers find that the vendor is more responsive to development requests. High-value features are developed and delivered more quickly with short cycles, than with the longer cycles favored by classic “waterfall” processes. The customer or user’s ability to provide sufficient support to the development team in a system development project has been critical in many instances, and lack of participation and engagement in the project reduces system success. By employing the agile development methodology, the company can address that problem successfully because the agile development is having connections with the employee all the time until the organization release the finale output. In another aspect it is like, the customer is also a part of the agile team. Customer is the ultimate authorized person to decide the sprints are working correctly towards the building the fully accurate and required software solution. All the developments will be make available for customer review and even the customer can change the requirements and find the best out of best. Eventually the customer is the party who is enjoying the ultimate advantage of the agile development because he or she will be able to take the fully functioning software solution sooner and with all the required features in it. And most importantly the software solution will be deviled to the customer before the used technology is obsoleted or the requirements are changed. A common draw back of the traditional methodologies like waterfall is that they take longer time period to build the fully functioning system and by the time the system is completed, the technology used to develop the system may be obsoleted or the requirements of the system is changed. But that problem is successfully addressed by the agile development methodology and that will be the biggest advantage for the customer. In addition to those things customer will be able to see the progress of the system weekly and hence the customer involvement during the development stages are high, customer can see the system before it is fully developed.

Agile development methodology is benefiting for all the parties who are attached with the development process.

REFERANCES

Adel , A., & Abdullah , B. (2015). A Comparison Between Three SDLC Models Waterfall Model, Spiral Model, and Incremental/Iterative Model. *International Journal of Computer Science Issues*, 107-111.

Dima, A. M. (2018). From Waterfall to Agile software: Development models in the IT sector, 2006 to 2018. Impacts on company management. . *Journal of International Studies*, 315-326.

Dingsøyr, T., Dybå, T., & Brede Moe, N. (2010). Agile Software Development: An Introduction and Overview. *Information Systems Research*, 329-354.

Eason, O. K. (2016). Information Systems Development Methodologies Transitions: An Analysis of Waterfall to Agile Methodology. *University of New Hampshire Scholars' Repository*, 504-527.

Gupta, A. K. (2015). A comparison between different types of software development life cycle models in software engineering. *International Journal of Advanced Technology in Engineering and Science*, 324 - 631.

Jinjin , L. (2011). Agile Software Development. *International Journal of Engineering Science and Technology*, 334–341.

Kai , P., Claes , W., & Dejan , B. (2009). The Waterfall Mo del in Large-Scale Development. *International Conference on Product-Focused Software Process Improvement*.

Kruchten, K. B. (2009). Towards agile security assurance. In Proceedings of the 2004 workshop on New security paradigms. *In International Conference on Product-Focused Software Process Improvement*, 386–400.

Manzoor, A. R., & Vivek, B. (2015). a Comparative Study of Software Development Life Cycle Modrls. *International Journal of Application or Innovation in Engineering & Management*, 23-29.

McCormick, M. (2012). *Waterfall vs. Agile Methodology*. New York: HarperCollins Publishers.

Nabil , M. A., & Govardhan, A. (2010). A Comparison Between Five Models Of Software Engineering. *International Journal of Computer Science Issues*, 94-101.

Pawar, R. P. (2015). A Comparative study of Agile Software Development Methodology and traditional waterfall model. *IOSR Journal of Computer Engineering*, 01-08.

Saxena, A., & Upadhyay, P. (2016). Waterfall vs. Prototype: Comparative Study of SDLC. *Imperial Journal of Interdisciplinary Research*, 1012 - 1015.

SoobiaSaeed, Jhanjhi, N., Mehmood, N., & Mamoona, H. (2019). Analysis of Software Development Methodologies. *International Journal of Computing and Digital Systems*, 445-460.

Youssef , B. (2012). A Simulation Model for the Waterfall Software Development Life Cycle. *International Journal ofEngineering & Technology*.

View publication stats