why tap

Paper 1

Introduction to Full Stack Web Development

Unit 3

Software Development Life Cycle and Methodologies

Module Leader

Sathishkumar Kannan, MS (UK)

Table of Contents

List of Figures

1. Introduction

According to Zhu (2005), the goal of both software development and system development methodologies is to act like a frame work for planning, structuring and controlling the process of development in software engineering. Many methodologies have been implemented for development process. In this, we will discuss about Build and Fix Model, Waterfall model, Spiral Model,

2. Build and Fix Model

The model called build-and-fix was adopted from simper and earlier age of the hardware product development. Those of us who bought early Volkswagen automobiles in the l950s and ’6os remember it well. When new models were bought and the old models updated, cars were used to sold frequently without meeting the benefits of testing, and the testing can be done only the customer. The vehicles are cheerfully and apparently serviced by the dealers all the times without making any charges to the owners. The dealers usually charge the owners for inconvenience and also for occasional risk of the breakdown. These model works completely well but it merely depends on having patient and faithful consumer set totally depend of using of product. This case is same with the software. Some well-known vendors are much familiar for number of free upgrades and rapid proliferation of some new versions. This works always best in case of monopolistic or semi- monopolistic environment by which the customers have only very limited access to the alternative vendors. Unfortunately, in this approach, the overall quality of the product will never address, even though some development issues are corrected. There is no chance to feedback to design process to make any proactive improvement. But the corrections or errors are put back into the market place as error fixes, upgrades or the service packs as soon as possible as means of the marketing damage control. Due to this, the build-and-fix model is reactive totally and recent standards are not really development approach at all.

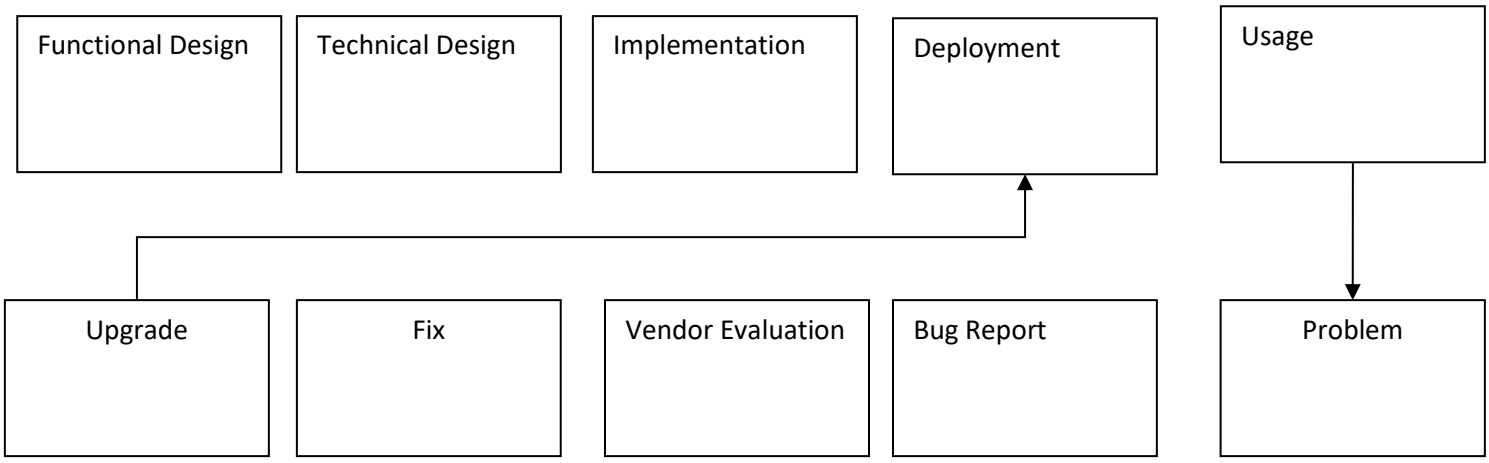

Figure 1 Build and Fix Model

3. Waterfall Model

In l97o's, waterfall model was introduced by Win Royce at the Lockheed. This model called by its name because it is represented as cascade from gathering requirements, creating design, program implementation, system test and deliver to the customer which is clearly shown in the diagram. It is step forward in the software development as engineering discipline. The diagram also depict single level feedback path which was not actually a part of original approach but that is added to every subsequent improvement of the approach. The waterfall model has no feedback between the stages like water will not flow or reverse uphill but it is drawn ever downward by the gravity.

This model may work satisfactorily when the design requirements can be addressed perfectly before moving into the design creation, and when the design creation was done perfectly then the program implementation began, and when the code was developed completely then testing part were began, and when the testing was done without any errors before the customers using it. This could be possible only when the user never changing the requirements. Only simple hardware products can be designed and manufacturing by this method, but this approach is become unsatisfactory for the software development due to the some complex issues. It is not possible to guarantee correcting of program for more than l69 lines of code. Providing the program functionality was useful and advantage during early stage of embedded control system, if the

program was tiny, but now-a days the multifunction cell phone requires millions of codes or even more.

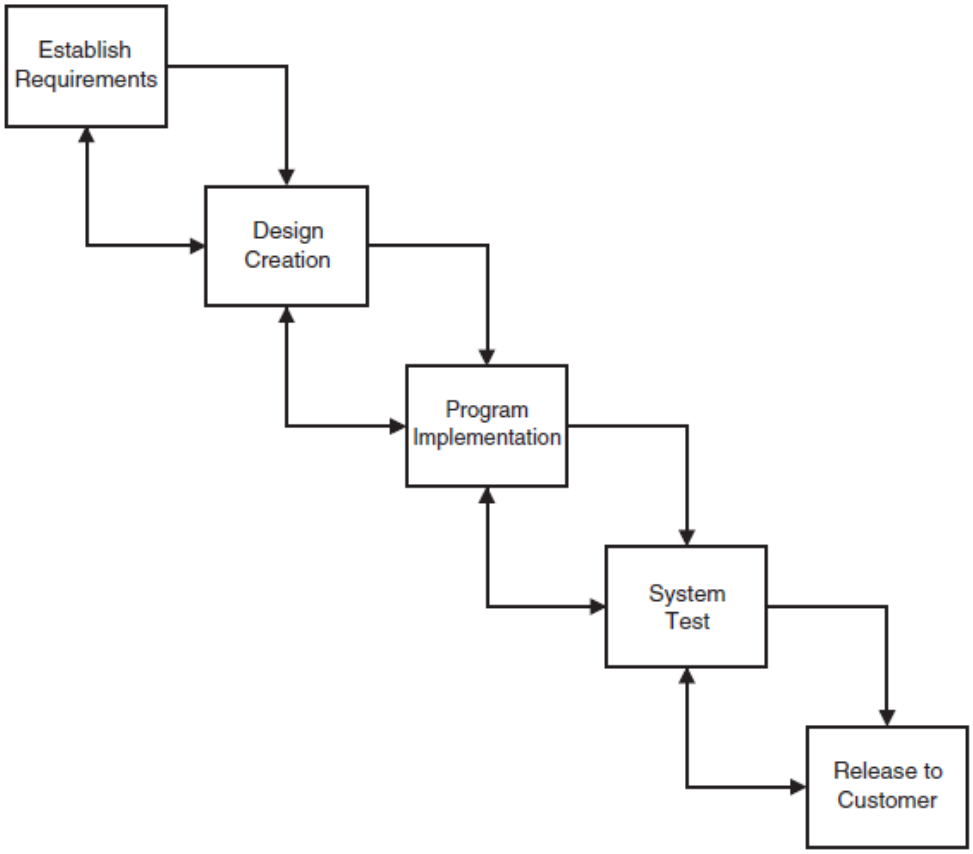

Figure 2 Waterfall Model

4. Rapid Prototyping Model

Rapid prototyping has long been used in the development of one-off programs, based on the familiar model of the chemical engineer's pilot plant. More recently it has been used to prototype larger systems in two variants—the "throwaway" model and the "operational" model, which is really the incremental model to be discussed later. This development process produces a program that performs some essential or perhaps typical set of functions for the final product. A throwaway prototype approach is often used if the goal is to test the implementation method, language, or end-user acceptability. If this technology is completely viable, the prototype may become the basis

of the final product development, but normally it is merely a vehicle to arrive at a completely secure functional specification, as shown in below figure. From that point on the process is very similar to the waterfall model. The major difference between this and the waterfall model is not just the creation of the operational prototype or functional subset; the essence is that it be done very quickly— hence the term rapid prototyping.

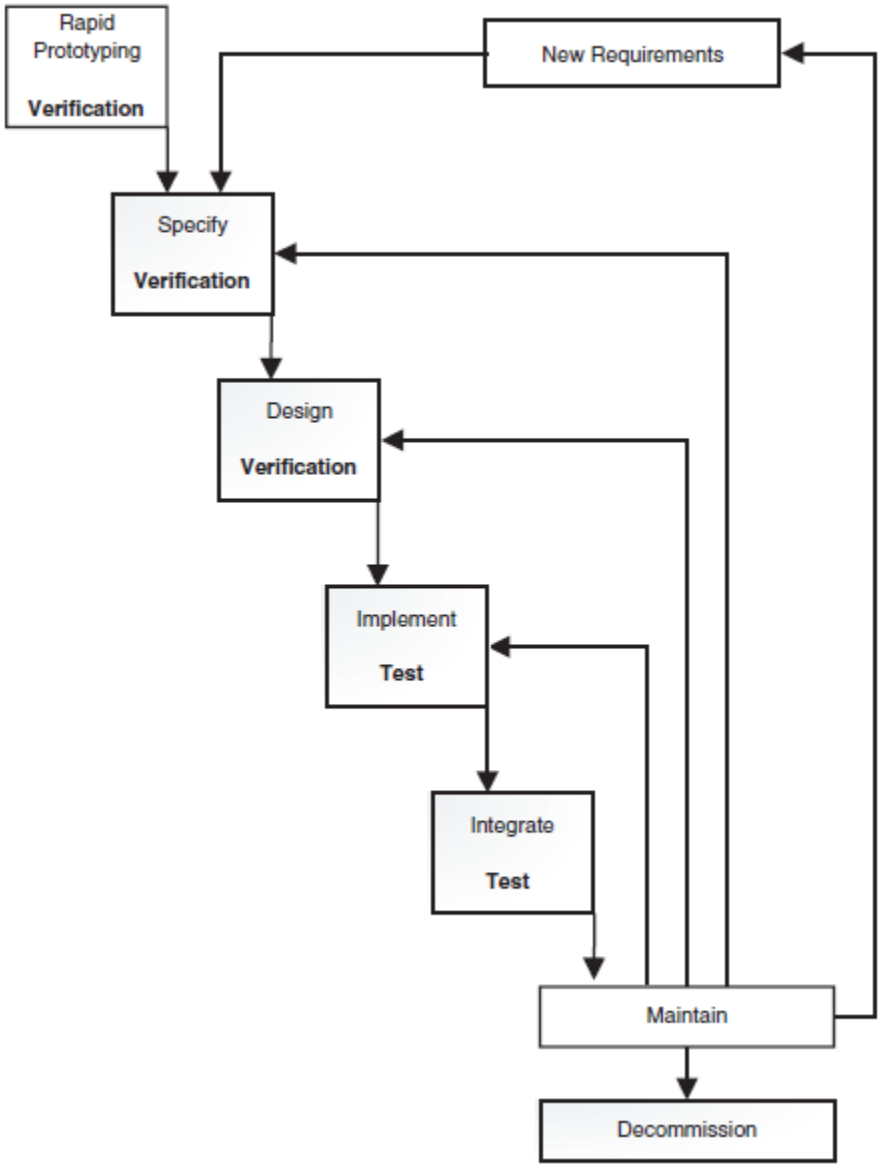

Figure 3 Rapide Prototyping Model

5. Spiral Model

The spiral model, developed by Dr. Barry Boehm at TRW, is an enhancement of the waterfall/rapid prototype model, with risk analysis preceding each phase of the cascade. You can imagine the rapid prototyping model drawn in the form of a spiral, as shown Figure. This model has been successfully used for the internal development of large systems and is especially useful when software reuse is a goal and when specific quality objectives can be incorporated. It does depend on being able to accurately assess risks during development.

This depends on controlling all factors and eliminating or at least minimizing exogenous influences. Like the other extensions of and improvements to the waterfall model, it adds feedback to earlier stages. This model has seen service in the development of major programming projects over a number of years, and is well documented in publications by Boehm and others.

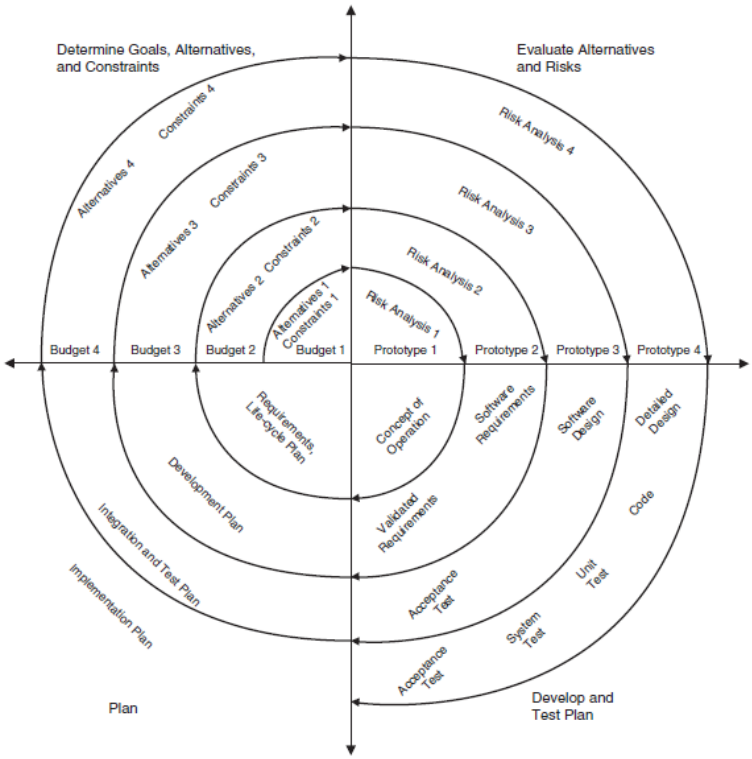

Figure 4 Spiral Model

6. Agile Methodology

6.1 Overview

Agile methodology is considered to be the most vital approach for managing time and cost in the software development process. According to Kong (2007), in this method many processes are broken and only effective team is involved in for delivering the product. The team never faces any compulsory segregation.

- **Definition:** Agile is an iterative and flexible software development approach that emphasizes collaboration, adaptability, and customer feedback.
- **Manifesto:** Values individuals and interactions, working solutions, customer collaboration, and responding to change.

6.2 Agile Principles

- Individuals and Interactions: Prioritize communication and collaboration.
- Working Solutions: Deliver functional software in short iterations.
- Customer Collaboration: Involve customers throughout the development process.
- Responding to Change: Embrace changing requirements.

6.3 Agile Frameworks

- Scrum: Organizes work into time-boxed iterations called sprints.
- Kanban: Focuses on visualizing work and maintaining a continuous flow.

6.4 Agile Process

- **Iterations:** Develop the software in small, incremental cycles.
- **Feedback:** Gather feedback from users and stakeholders after each iteration.
- **Adaptation:** Adjust the project based on feedback and changing requirements.

6.5 Advantages of Agile:

- Flexibility: Embraces changing requirements and priorities.
- Customer Involvement: Regular customer feedback throughout the process.

6.6 Disadvantages of Agile:

- Less Predictable: Project timelines and deliverables may be less predictable.
- Requires Collaboration: Success depends on strong collaboration within the team.

6.7 When to Use Agile:

- Dynamic Requirements: When requirements are expected to change.
- Large Projects: For large and complex projects

7. Rational Unified Process (RUP)

Rational Software Corporation is considered to be the cyclic software development process. By using this process an ordered mechanism can be seen in the activities of the team. According to Kruchten (2004), the aim of this methodology is to generate effective software that helps to satisfy the demands of the client. This method better suits a project where implementation is done in the present software. RUP involves four main phases that maximizes the requirement for a long-term project. According to Kroll (2003), the phases shown below are performed with a perfect objective:

- Inception phase
- Construction phase
- Elaboration phase
- Transition phase

Inception phase

Here predictions on budgets, market recognition, profits are well planned. Here the team analyzes whether the project would be beneficial or not. The team in the development of software well focuses on the requirement of the client. The team has the option to reject some project if it is found to be non-beneficial for the team.

Elaboration phase

In this phase a better planning is made on the project. This phase helps to remove the risk involved in the project. Here the team gets a chance to differentiate both functional and the non-functional part. This phase is found to be the most vital phase when compared with other phases involved.

Construction phase

Here development of the product and effective testing is done. This phase is also named as manufacturing phase where the resources develop a effective product that is both time and cost effective. Here the total team focuses on product development. The plan framed in the elaboration phase will be designed, integrated etc in this phase.

Transition phase

This phase helps to achieve the transition of final product to user interface. Once the product is delivered to the customer the customer does a beta testing for testing the product in order to understand whether it satisfies the demands of the customers. In this phase there are chances for modifying the product, clarifying and cancelling the project.

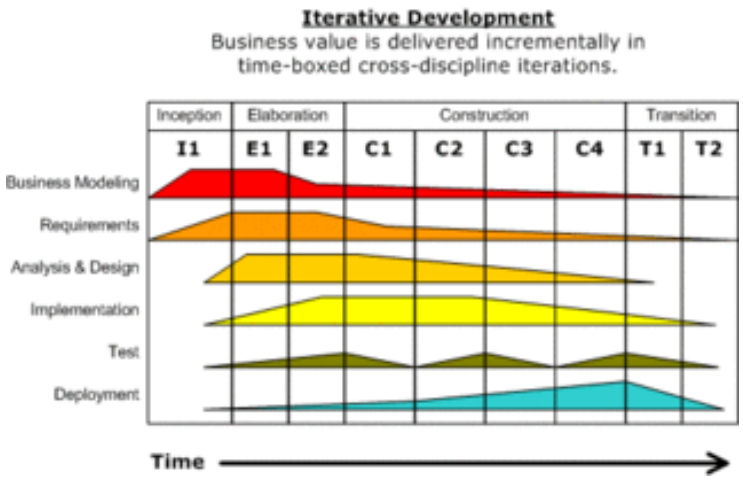

Figure 5 Rational Unified Process Work Flow

The figure shown above clearly describes the different software implementation in any of the four phases involved in RUP. Here the vertical line depicts various disciplines whereas horizontal line denotes the time. In the inception and elaboration phase business modeling is performed. In the inception phase the requirement of the client are well analyzed and its is designed accordingly in the elaboration phase. Implementation takes place in the elaboration phase where as progress in the development happens in the construction phase. In every phase except the inception phase testing happens at the end of the process. Deployment of the product happens in the construction phase that continues in the transition phase.

8. Choosing Between Agile and Waterfall

Considerations

- Project Size: Small projects may benefit from Waterfall, while larger projects may favor Agile.
- Requirements Stability: Stable requirements may align with Waterfall, while evolving requirements fit Agile.
- Client Involvement: Frequent client involvement favors Agile.

Hybrid Approaches

- Agile-Waterfall Hybrid: Combining aspects of both methodologies to suit project needs.