Chapter 3

Technology review

3.1. Introduction

Previous chapter covers detail description about problem domain. In this chapter I will discuss the technologies currently available to solve a problem in this nature. My effort is to bring all the available technologies and there potentials in solving a problem in this nature.

3.2. Software process models

There are many ways of developing software. According to the requirement, nature, time, and so many other factors it can be vary from one software to another. There are three main software process models we can distinguish by their unique process models.

3.2.1. The waterfall model

As per the below figure 3.1 the main activities in software development are placed as steps of a waterfall. Similar to the pattern of water streaming down from one step to another step in waterfall the software development process is also proceeded.

According to the Sommerville the principal stages of the model map onto fundamental development activities. [9] Based on that we can say by this model software development life cycle has been broken into different faces. It can be recon as the widely used software process model. This could be useful in large system designs when the requirements are not change during the process.

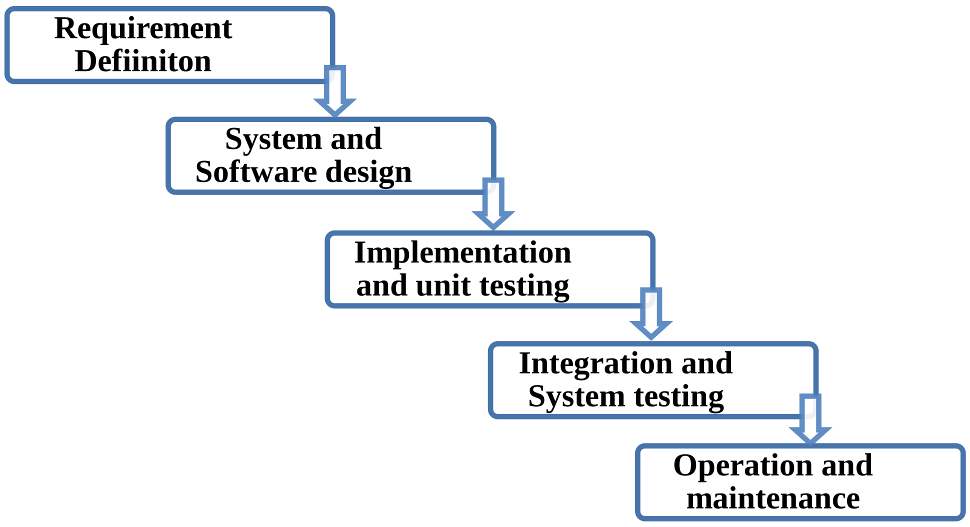
Requirement Defiiniton
System and Software design
Implementation and unit testing
Integration and System testing
Operation and maintenance

Figure 3.1 Waterfall model

3.2.2. Evolutionary development

This is methodology is also called as prototype methodology. There a prototype of the software is developed initially with the initial requirements given by customer. Then with the feedback of the customer the complete requirement can be done.

By following this approach the key steps in waterfall method can be performed simultaneously. Therefore we can identify 3 main process steps that cover the full cycle in waterfall method. Figure 3.2 depicts key steps.

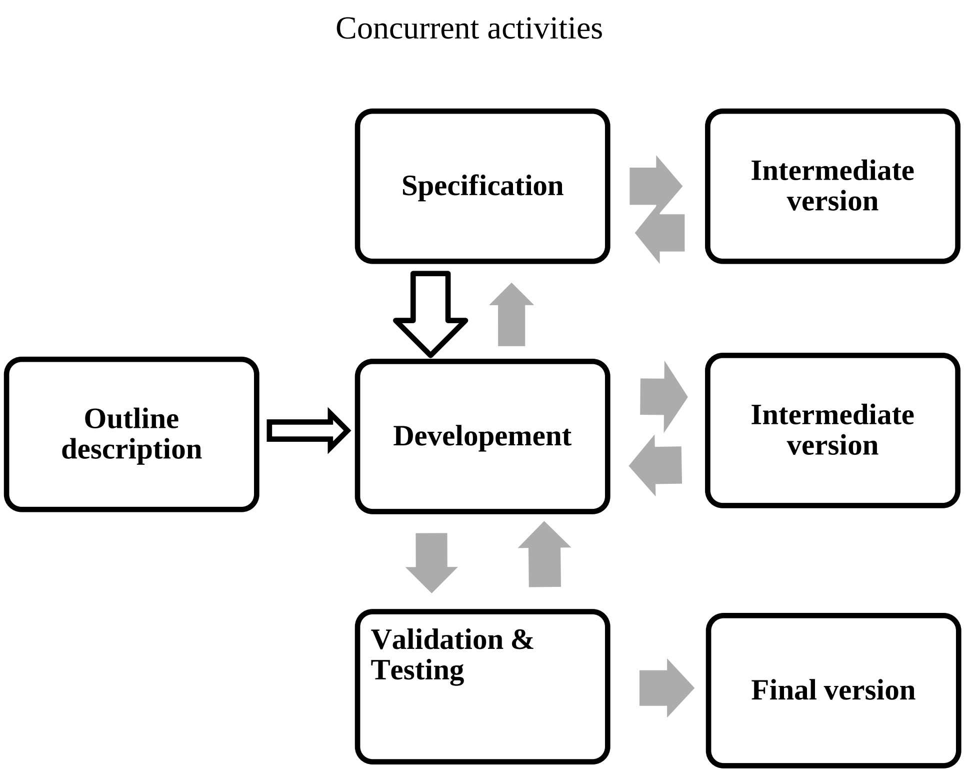

Figure 3.2 Evolutionary development model

There are two types of evolutionary development methods,

1. Exploratory development

   Design a part of the requirement first, that is more understandable and easy. Then with the input of customer the full requirements are addressed by adding parts to that.

2. Throwaway prototyping

   In this methodology a prototype is design with the poorly understood requirements. And the customer's feedback on that will be helped to created full customer requirements.

Evolutionary approach seems more appropriate and effective compare to waterfall approach since customer involves in the design stage. As users are more aware about the software and the developers can also more through about the requirements of the customers/users during the development stage. The software requirement specification is design in incrementally.

This kind of approach is suitable when the requirements are not clear and there is no time to follow an extensive analysis design phase. Even after a proper analysis for components like user interface design this approach is very effective.

3.2.3. Component-base software engineering.

This methodology is fairly new to software industry. When there are similar requirements catered previously, developers tent to get those components and reuse with some adjustments to the new system. This behavior is expanded now up to the reusable components that can be taken off the shelf.

When there is a large collection of reusable components this approach can be exercise. With the reusable components and interfaces that con linked each component this approach is fairly easy and fast. Due to the reusable components new steps call ‘Component analysis’ is adding to the development life cycle. Refer the figure 3.3.

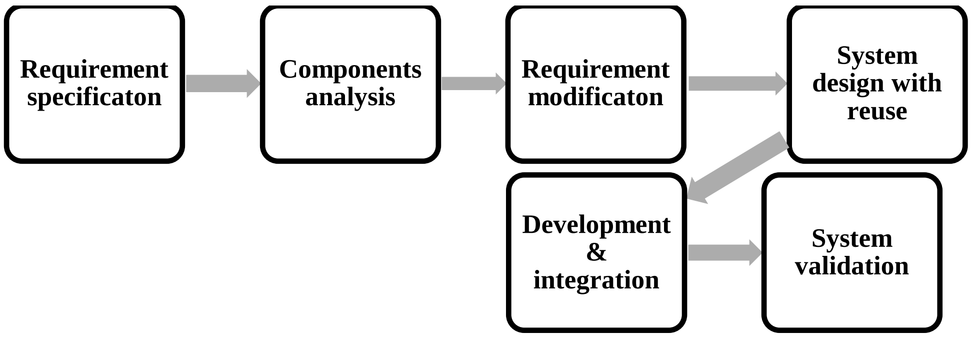

Figure 3.3 Component base development methodology

This methodology cut down the development cost drastically. And it reduces the risk as well. Since the most of the off the shelf product components are thoroughly tested and may have proven its capabilities in previous projects. The problem here is weather these components can cater the requirements of the customers or users. Most of the time requirement may have to be change or adjust according to the available features of the components that are available. When there is systems upgrade due to change in the requirement the whole system can be obsolete due to the components inability of supporting new features.[9]

3.3. Design and development approaches

There are number of software design methodologies practicing in the software development process. Depending on the environment and the requirement the suitable methodology is selected. Below are the available development methodologies.

3.3.1. Object oriented analysis and design (OOAD)

Compared with the counter part of OOAD, which is SSAD, OOAD is fairly a new technology. With the growing size and complexity of the software projects OOAD has proved its capabilities to handle such situations.

Always classes represent real world entities. In addition to that there are some other classes for problem solving purposes. Object oriented design all ways make sure to maintain close link between these classes.

UML is using to design the software architecture. By following the proper methodology class diagrams can be designed via UML.

The key feature of the OOAD is, it ensures the security of data that is associate with its functions. Object is defined by a class where it has some functions and local data definitions. In the initialization, the data will be passing into the object and only the methods in the same object can be access those data. This is the main feature of OOAD. Additionally when there is a change required in one functionally in a class it can be easily done without worrying about the other classes or methods in the same class.

Since classes can use to depict a complex problem in very simple form OOAD is very popular among large and complex software solutions. Reusability is another important feature of classes. Classes created to one project can be easily used in other projects as well. By doing that the cost and time can be reduce. And this feature tents to have lot of pre designed classes available. Because of that the development using OOAD becomes more easy and popular. [9]

3.3.2. Structured System Analysis and design (SSAD)

Classical software design methodology is called as SSAD. Before the OOAD is invented this methodology dominated the software industry. Procedural approach is used in the development of software. Therefore procedural languages used in design process.

There are 3 major tools using in SSAD application design.

1) Logical Data Modeling: This is the process of identifying, modeling and documenting the data requirements of the proposed software application.
2) Data Flow Modeling: This is the process of identifying, modeling and documenting what and how the data is flowed in the proposed software application
3) Entity Event Modeling: This is the process of identifying, modeling and documenting interconnection between the business events and entities. The sequence of interaction also taken to consideration.

This model is not using in modern large scale software application developments because it’s difficult in design and maintain such a system designed by this model. Due to the procedural approach the data is prone to any subroutine. If there is an error in sub routine the data is directly get affected. No encapsulation of data. No mechanism to simulate the real world entities in the program. This is vital requirement when designing a large scale system. [3], [4],[5]

3.3.3. Agile software development methodology

This development methodology is immerging as popular development methodology nowadays. It promoted the team work, self-organization and accountability for the assignment.

Iterated software process model is using in most cases in agile software development methodology. This methodology can be applied in rather small scale developments. For a large scale project this can be applied successfully if the project requirements can be broken into collection of separate developments.

In agile development the development team comprise with 5~10 people from all disciplines like architectures, developers, testers, etc. And this is performed with minimum planning and development time bounded 2 to 4 weeks. Full software development life cycle is performed with in that period with the involvement of the whole team allocated to that development. At the end it will be released to customer and based on the customer feedback further enhancements are performed following the same methodology explains above. In the process the discussion within the team is try to keep minimum as possible.

By following this kind of methodology can minimize the risk factor. The full release of the agile development is happen after the customer satisfied with it. That could be reached after a several cycles of testing. Since there is not full software system life cycle is performed there is a possibility of not getting proper project documentation.[6]

3.4. Unified modeling language (UML)

This is graphical development methodology for designing, visualizing and construction of software application. It provides the most necessary artifacts needed in software requirement specifications.

In object oriented analysis and design process UML caters up to the class and database diagrams level. Most of the object oriented concepts are incorporated with in UML.

UML comprise with 3 types of diagrams.

1. Structure diagrams

   Diagrams contain features to be required in software application. Eg: class diagrams, component diagrams, deployment diagrams, object diagrams, etc
2. Behavior diagrams

   Diagrams explain the behavior of the software to be developed. Eg: activity diagrams, use case diagrams, state chart diagrams
3. Interaction diagrams

   Diagrams explain the process/data flow of the software to be developed. Eg: sequence diagrams, communication diagrams, interaction overview diagrams etc.

[7]

Rational Unified Process (RUP) is software development process enables transforming user requirements into software programs. RUP is a generic framework that is independent of the software which is going to developed. It’s agile and component base process that has easy adaptation and interconnection with components. RUP uses UML in its software development process

3.5. Summary

This chapter discusses the candidate technologies/approaches for this development. This provides an overview of the technologies, not a full detail description. Then the next chapter is about my approach. It will covers the technologies and methodologies I’m going to solve the problem.