International Journal of New Technology and Research (IJNTR)
ISSN:2454-4116, Volume-3, Issue-5, May 2017 Pages 30-33

Evolutionary Models in Software Engineering

Gowtham V., Manoj Y., Pooventhiran G., Praveen A., Shivaram R., and Kathiresan A.

***Abstract*— Software development life cycle models play a vital role in developing a software application. This research deals with such advanced models which are the evolutionary models namely: incremental model, and spiral model. Both these models have their own advantages and disadvantages as well. The main objective of this research paper is to represent the two evolutionary models' features and limitations.**

***Index Terms*—Evolutionary Models, Increments, Risk analysis, Software engineering, Staffing.**

I. Introduction

Computers nowadays have become the essential part of human beings. To aid computer serve humans, software applications were developed, as a solution to a particular problem. In the development of such software, we used to face several issues in terms of requirements, errors, cost and time etc. The software sustains only if it meets out the entire requirements to solve the problem for which it is developed, and if fails, the software becomes obsolete, and so the cost, time, and effort put together to develop the software become useless. But since the software does not wearout like hardware, the failure rate curve flattens in due course of time. Nevertheless, the companies which develop software rely on the type of software development model employed to develop the software. To develop such software efficiently, like other products, the software is also engineered with predefined structures and process flows. Software engineering is actually a layered technology. The layers in software engineering enables the programmers design the software easily and in a well-understood manner. The base layer is process layer which holds the technology layers together and enables the timely development of the software. The next layer is method which describes the technical methodologies to develop the software. The topmost layer is the toolsproviding either automated or semi-automated supports for the processes and methods. Software engineering thus paves the way for developing the software in a well-defined manner thereby making it more reliable. It itself tells the importance and need for software engineering. Software engineering mainly deals with the effective development and sustainable maintenance of such softwares.

Pooventhiran G, B.Tech Information Technology student of PSG College of Technology, Coimbatore-641004.
Gowtham V, B.Tech Information Technology student of PSG College of Technology, Coimbatore-641004.
Manoj Y, B.Tech Information Technology student of PSG College of Technology, Coimbatore-641004.
Praveen A, B.Tech Information Technology student of PSG College of Technology, Coimbatore-641004.
Shivaram R, B.Tech Information Technology student of PSG College of Technology, Coimbatore-641004.
Kathiresan A, B.Tech Information Technology student of PSG College of Technology, Coimbatore-641004.

II. Software Life Cycle Models

The generic software development process consists of five phases: communication, planning, modelling, implementation, and testing. Apart from these, it also has supporting activities like risk management, quality assurance, reviews etc. Any softwaredevelopmentlifecycle(SDLC) model, irrespective of its process flow, follows these phases. The process flow makes the SDLC models differ in functionalities.

The classical Waterfallmodel makes the processes flow linearly which causes trace backing difficult and at high cost. So, any errors in previous stages which are left unresolved, makes the following stages erroneous. It is worth using only if the programmer knows the complete requirements and understands at the very beginning of development. Since, the entire requirements for some software cannot be known prior to the release of software, waterfall model does not suit for those.

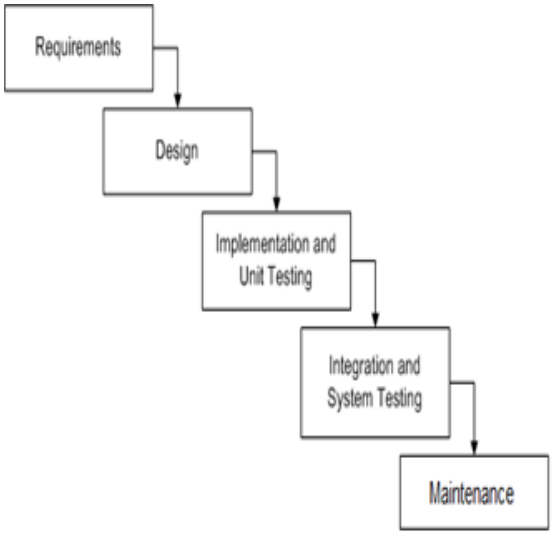

Fig. 1 Waterfall Model

III. Evolutionary Models

Here comes into picture, the evolutionary models. They are basically iterative. Once the requirements are analysed, they pass through a series of iterations till the complete software is developed. The evolutionary models mainly support the programmer to develop the complete version of a software. After each release, based on the review given by the reviewers, further iterations are performed. It's mostly

employed to make the reliable version of the software. It involves more user interaction in every iteration, and thereby increasing reliability. The main two evolutionary models are:

1. Incremental model
2. Spiral model

A. *Incremental Model*

In incremental model, the project is divided into smaller parts. After requirement analysis, the iterations start. Each iteration follows the classical waterfall model. In each iteration, communication, planning, modelling, construction, and delivery are performed. Here, each iteration is released to the customer as increments. The first increment contains the basic functionalities of the software, and therefore called the core product. The software is updated whenever a new feature is to be added and built on top of the core product. It continues till the complete version of the product is developed. So, it takes very long period to release a complete software.

Incremental development is particularly useful when staffing is unavailable for a complete implementation by the business deadline that has been given for the project[1]. If the core product delivered is a success, then the additional staff(if required) can be added to work in from the next increment[1].
Advantages:

i. Simple and easy
ii. Each iteration releases a working version of software
iii. Promotes maintainability
iv. Easily manageable risks

Disadvantages:

i. Less flexible
ii. Very long time to release the complete version of software

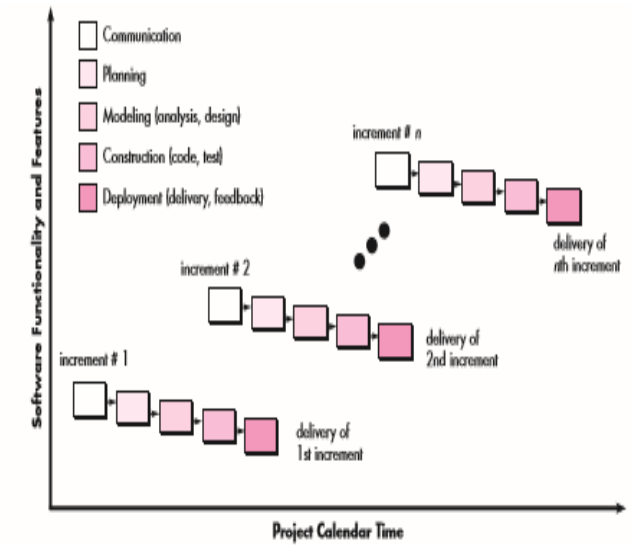

Fig. 2 Incremental Model[1]

B. *Spiral Model*

Spiral modelproposed by Barry Boehm [Boe88], is quite similar to incremental model, but concentrates more on RiskManagement. It has fourmajor phases: planning, riskanalysis, engineering, and evaluation. Here, the software passes through many "spirals" (like increments in incremental models). The base spiral involves requirement gathering and risk assessment i.e. defining the objectives. The subsequent spirals are built on this base spiral. Next, the process is analysed for any risks i.e. risk analysis. Later, the model to develop the software is decided on and at last, based on the review of the software with the project team and customer, the next phase is planned.

The spiral model is a realistic approach to the development of large-scale systems and software[1]. It relies on the risk assessment expert who handles risk management for the success of the software.
Advantages:

i. Better risk analysis
ii. Well-suited for larger projects
iii. Very flexible
iv. High user involvement

Disadvantages:

i. Expensive
ii. Somewhat complicated model

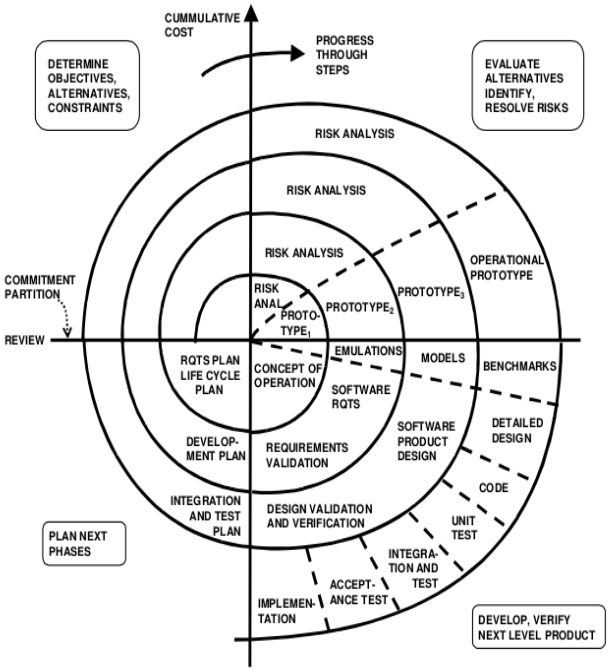

Fig. 3 Spiral Model

IV. Case Study: An Analysis of Railway Reservation System using Spiral Model

The case study is about the railway reservation system which is already existing and developed using the spiral model. This case study justifies the choice of spiral model using the risks involved in developing and maintaining the system. The reservation can be done in two methods:

1. Counter reservation
2. Online reservation

A. *Counter Reservation*

The counter reservation facility uses up the counters

installed in the railway stations to book the tickets. The customers are supposed only to pay the amount for tickets by cash.

B. Online Reservation

The online reservation facility helps the people to reserve and cancel the tickets at their own pace and interest using their credit cards. So, the customers also incur an additional charge in addition to the bill amount.

C. Determine Objectives

This phase identifies the primary objectives and requirements of the software. Business Requirement Specification (BRS), Software Requirement Specification (SRS), alternatives in design etc. All requirements are considered according to the customer's feedback. So, the permanent communication between the customer and the project management is crucial.

Modules for Users

The list follows tells the modules or functions used by the users:

i. Login and Logout
ii. Check for PNR status
iii. Train enquiry
iv. Book tickets
v. Payment gateway
vi. Cancel tickets

Modules for administrators

The list follows tells the modules or functions used by the database administrators:

i. Login and Logout
ii. Add/Remove users
iii. Add/Remove train entries
iv. Get passengers list

System and software requirements

The list follows tells the system requirements:

i. Intel Pentium IV
ii. 256 MB RAM
iii. 1GB boot drive
iv. 1GB or more free storage
v. 1 NIC
vi. GSM modem

The list follows tells the software requirements:

i. MS Windows XP or 2000
ii. MS .NET framework 2.0
iii. MS visual studio .NET 2.5
iv. Internet information server
v. MySQL server
vi. Windows Installer 3.1
vii. MBIE browser

The platform requirements be:

i. Windows XP
ii. Windows 98
iii. Linux

D. Risk Analysis

Managing railway reservation database is somewhat a risky task, as it deals with large data. The following could be the risks possible with the system:

i. Deadline given for the project
ii. Incompatibility issues:
    a. Platform incompatibility
    b. Framework incompatibility
    c. Web browser supports
iii. Cost effectiveness

The risks analysed is purely dependent on the risk analysis expert involved in the project. Keeping in mind all these risks, spiral model suits this system the best. Any other evolutionary model like incremental model if used, it uses to release the software as increments that will have additional features in every increment. But the risks involved is more here and so, it cannot produce more reliable software. At the end, a prototype is developed.

E. Construction Of Prototype

This phase develops the prototype designed based on the feedback acquired in the first phase, and the resolved risks in the second phase. The product is tested and reviewed. Anyhow, in the first spiral, a basic version is only developed and further enhanced to a working version called the build with reviews again. It thus gives a higher clarity of requirements.

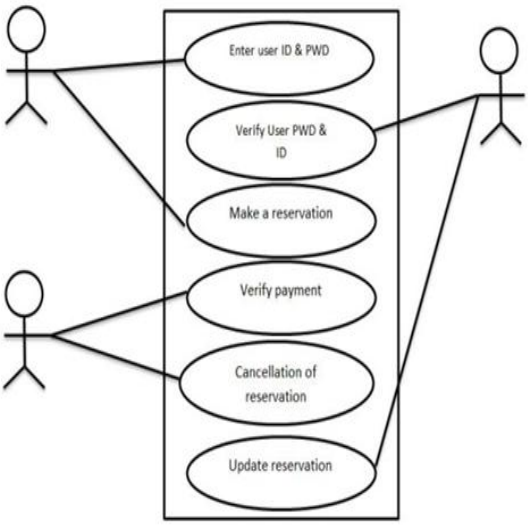

Fig. 4 Use case diagram – Railway Reservation System

The above use case diagram indicates the user interaction with the software.

From the use case diagram and other modelling diagrams, a working version of the application is developed with basic functionalities including booking and cancelling tickets etc. In further spirals, the working version is updated so that a complete version of the software.

F. Planning Next Phase

This small phase evaluates the product so developed before it passes onto next spiral. It's basically done with the customer and the project management team.

The outcome of the first spiral consisting of basic functions are evaluated by the customer, and based on his feedback, the next phase is planned well ahead of time.

V. Conclusion

The evolutionary models provide a better reliability when compared to other models. All evolutionary models are better in each aspect, and useful in where they deserve. The case study has shown that the railway reservation system uses spiral model since it has high risks. Some other softwares can be developed with incremental model like MS Word, which delivers increments as and when a new feature is needed. The evolutionary design models (EVO) also helps in accelerating sales if it is a success. Depending on the process flow of the software, we can choose the appropriate SDLC model. This paper has thus presented some evolutionary models' features and limitations.

References

[1] Roger S. Pressman, "Software engineering – A practitioner's approach", Mc Graw Hill, 7th edition, 2010.

[2] Nabil Mohammed Ali Munassar, A. Govardhan, "A Comparison between Five Models of Software Engineering", International Journal of Computer Science Issues, Vol. 7, Issue 5, September 2010.

[3] Apoorva Mishra, Deepty Dubey, "A Comparative Study of Different Software Development Life Cycle Models in Different Scenarios", International Journal of Advance Research in Computer Science and Management Studies, Vol. 1, Issue 5, October 2013.

[4] Elaine L. May, Barbara A. Zimmer, "The Evolutionary Development Model for Software", Hewlett-Packard Journal, August 1996.

Pooventhiran G, is a B.Tech Information Technology student of PSG College of Technology, Coimbatore-641004.

Gowtham V, is a B.Tech Information Technology student of PSG College of Technology, Coimbatore-641004.

Manoj Y, is a B.Tech Information Technology student of PSG College of Technology, Coimbatore-641004.

Praveen A, is a B.Tech Information Technology student of PSG College of Technology, Coimbatore-641004.

Shivaram R, is a B.Tech Information Technology student of PSG College of Technology, Coimbatore-641004.

Kathiresan A, is a B.Tech Information Technology student of PSG College of Technology, Coimbatore-641004.