
Airo
NATIONAL JOURNAL
ISSN : 23213914
Volume : 7
A Multidisciplinary Indexed National Research Journal

A STUDY ON VARIOUS MODELS OF SOFTWARE DEVELOPMENT LIFE CYCLE

G. RAJESH

Research Scholar, Satya sai University

Abstract

This paper reviews software development life cycle models that are used in the area of software development. It elucidates about various advantages and disadvantages of each model, according to which, it can be decided which model should be used under which conditions. These models are of the following two types: traditional models and contemporary models. The Waterfall model, Incremental Model, Spiral Model and V-Shaped Model are traditional models which follow a set of prescribed steps. The Contemporary models widely used in industries are Rapid Application Development Model, Agile Development Model and Extreme Programming Model.

Keywords

Software development life cycle (SDLC), Software models, Traditional Models, Contemporary Models and Agile teams.

I. INTRODUCTION

Software engineering is a coherent, methodical and structured approach used for development, performance and maintenance of software products. This implies that when different group of people apply same methodologies of software, similar software will be produced. It is the application of engineering to software to develop efficient software products that are able to meet the constraints of cost, quality and time. System Development Life Cycle in Software engineering refers to the process of constructing or fabricating systems, models and techniques used for their development. Software Development Life Cycle provides sequence of operations for software developers to develop the software in a manner such that it is completed within

deadlines and quality of the product of software is maintained as per the standards laid down by the developers. It is often considered as a subset of System Development Life Cycle.

II. PHASES OF SDLC

The various activities carried out for software development can be divided into manageable sections called as phases. These phases may be carried out in different way as per the need. Generally, there are five common phases in SDLC: A. Requirement Gathering & Analysis B. Designing C. Coding D. Testing E. Maintenance and Support

The requirement gathering and analysis phase helps to understand the problem. This is followed by deciding a plan to solve the problem in the designing phase. The planed solution is then implemented during coding. The Solution program is tested against various test cases. Deployment and maintenance follows this stage. There are various software development approaches designed to develop the software products. These are known as Software Life Cycle Models.

Fig. 1 Phases of SDLC

III. SDLC MODELS

These models describe the various phases and the order of their execution to be

followed to develop the software product. Each phase produces deliverable that are fed as input to the next phase in the life cycle. The product generated by the last phase serves as the final product called as software product. Different models proposed so far are:

A. Traditional models

1) Waterfall Model 2) Incremental Model 3) Spiral Model 4) V-shaped Model 5) RAD Model

B. Contemporary Models

1) Agile Model 2) Extreme Programming Model

IV. TRADITIONAL MODELS

Traditional software development models are those that are characterized by linear nature, that is, the various phases of SDLC are carried out sequentially. Building the software product initiates with the visualization of the final software, and continues working through to build the final visualized software product. A few traditional models are discussed below.

A. WATERFALL MODEL

Waterfall model was proposed by Royce in 1970. It is known as the classical and the basic model of Software Engineering. It is a linear sequential SDLC model because the various phases are carried out in a sequence. In this model, development is seen as flowing downwards through the following phases:

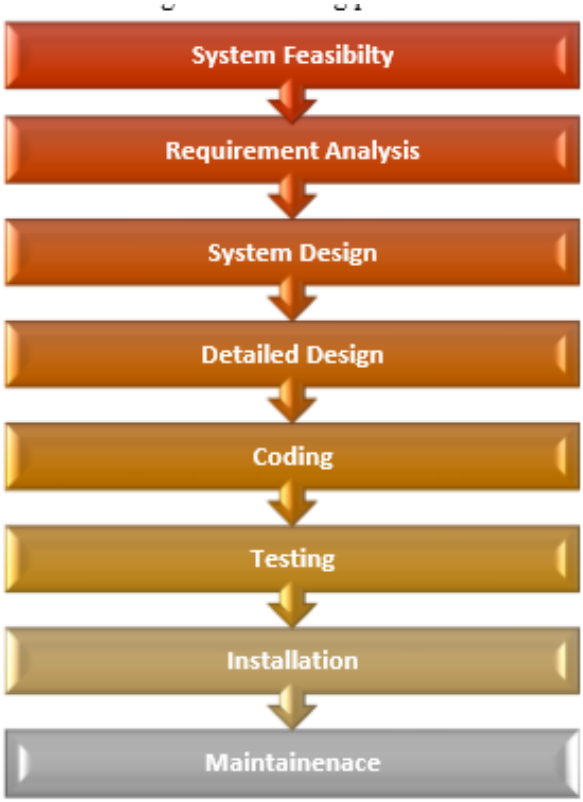

Fig. 2 Waterfall Model

1) Each phase must be completed before the next phase can begin.

2) Extensive documentation along with validation and verification is involved.

B. INCREMENTAL MODEL

The Incremental Model combines the elements of linear model with iterative prototyping. In prototyping, incomplete versions of the software program are created which are either discarded or developed further into final products based upon user response. The Incremental Model implements prototyping repeatedly in a sequential manner. This model prioritizes system requirements and then implements them subsequently to the previously developed prototype. This process is repeated until the desired functionality is achieved by software product.

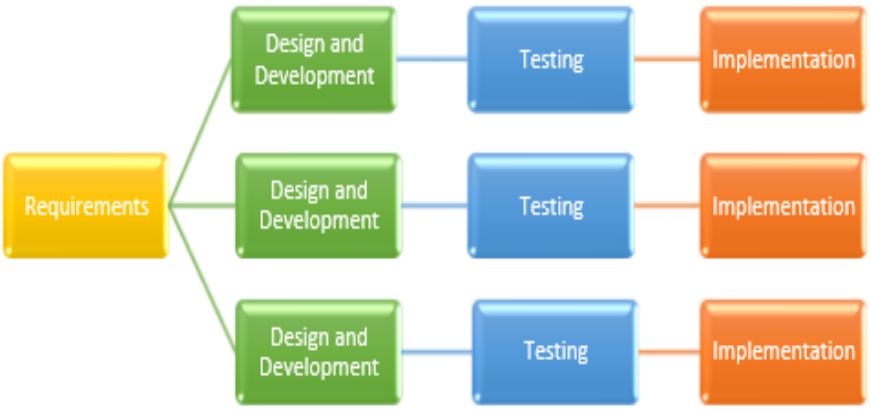

Fig. 3 Incremental Model

C. SPIRAL MODEL

Spiral model was defined by Barry Boehm in 1988. It is similar to Incremental Model with more focus on risk analysis. The Spiral Model involves four phases: planning, risk analysis, engineering and evaluation. A software project passes through these phases repeatedly in iterations corresponding to various spirals in the model. The baseline spiral starts in the planning phase. The requirements are gathered and risk assessment is done. Subsequent spirals are built on the baseline spiral. During risk analysis phase, risks are identified and alternative solutions are suggested. Prototype is produced at the end of risk analysis phase. At the end of engineering phase, tested software is produced. During evaluation phase, the customer evaluates output of project before it continues to next spiral. In this model angular component measures progress while radii represent cost.

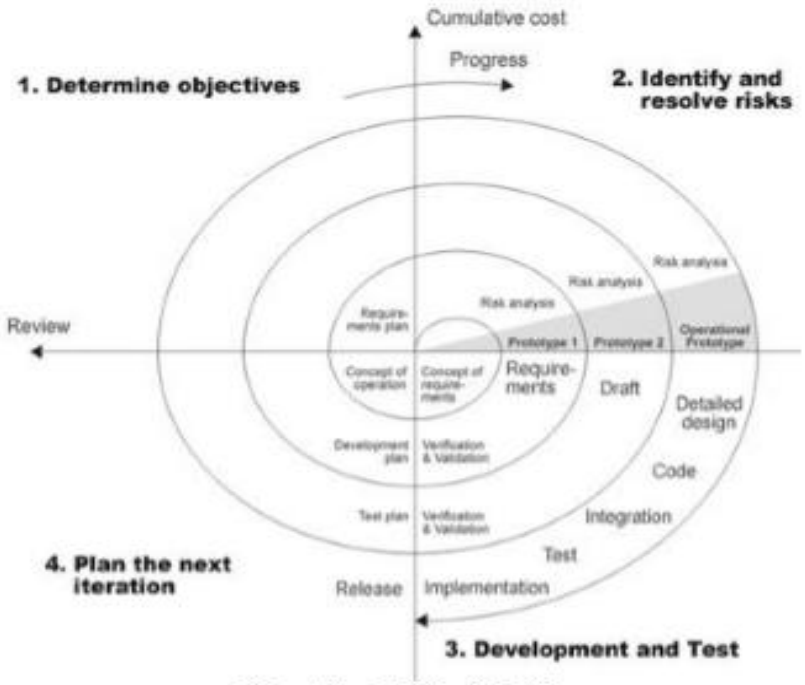

Fig. 4 Spiral Model [9]

D. V-SHAPED MODEL

V Model refers to verification and validation model. Like the waterfall model, the V-Shaped life cycle model is a sequential model that is each phase must be completed to begin the next phase. In this model, testing is done simultaneously with the development phase which means the earlier tasks are verified later.

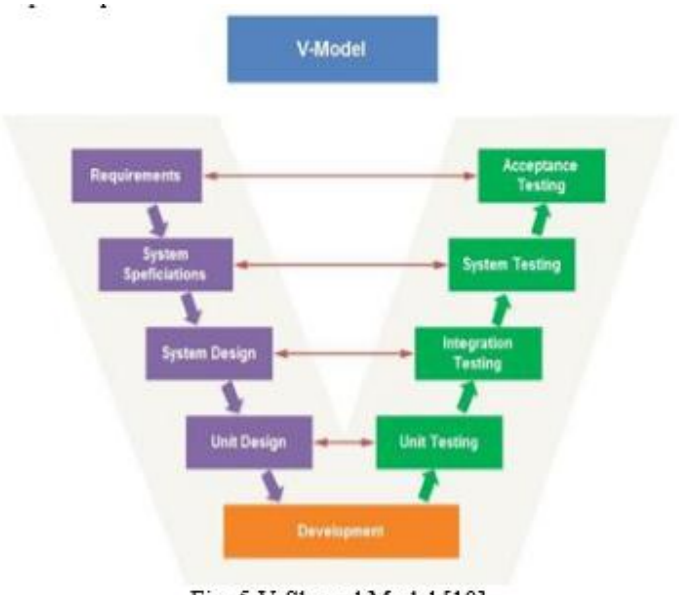

Fig. 5 V-Shaped Model [10]

V. CONTEMPORARY MODELS

The contemporary software development models are based upon the principle of iteration of SDLC processes to give more importance to adaptability during development of the software. A few contemporary models are discussed below.

A. RAPID APPLICATION DEVELOPMENT MODEL

Rapid Application Development model is a type of incremental model. In this model, focus is on developing quality product in less time. In RAD Model, components are developed in parallel like several mini projects and are quickly delivered. This gives the customers something to see and use and also provide feedback regarding the delivered product and any change in their requirements can be handled. Thus, the development team can deliver a fully functional system within a very short period.

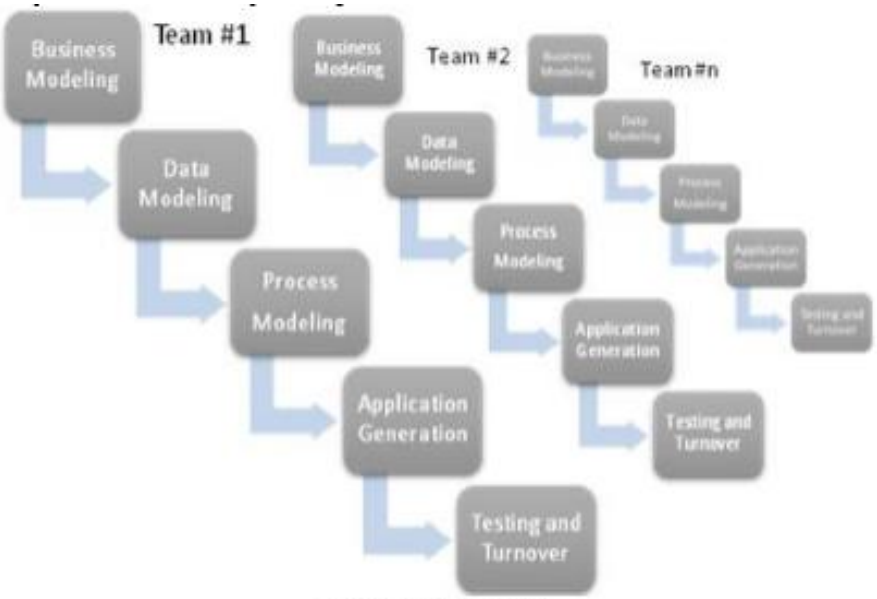

Fig. 6 RAD Model

B. AGILE SOFTWARE DEVELOPMENT MODEL

The agile software development model was introduced by the agile team in 2001 through the agile manifesto. It focuses on early and continuous delivery of software, through iterative and incremental development, so as to achieve customer satisfaction. The main attributes of the agile model are:

1) Incremental: Small software releases, accompanied by rapid development cycles

2) Co-operative: Closer customer-developer interaction

3) Adaptive: flexible enough to adapt to last moment changes

Agile software development is largely dependent upon the agile team, which is a cross-functional group of individuals responsible for defining, building and testing the solution software. These teams are highly jelled and their working is based upon trust and interaction. They have the ability and the authority to develop the agile software.

C. EXTREME PROGRAMMING MODEL

The Extreme Programming (XP) is an agile development methodology used for

developing software in an unstable environment. It allows flexibility to changes during the development process, thereby lowering the cost of change in requirements during the later development phases.

VI. CONCLUSION

There are many software development life cycle models, each claiming to be the best. This paper provides an overview of few of these models. No model is superior to any other and these models are used depending upon the requirements and prevailing conditions. Each model has its own advantages and disadvantages; therefore a hybrid of these methodologies is developed and chosen for different projects according to the requirements. Recent time has shown a shift from traditional models to contemporary models of software development. Selecting the correct life cycle model can help to deliver the required software product within deadline, meeting quality and cost constraints. Further, more changes and evolutions are continuously being performed to increase the quality of software being developed by improving the methodologies used.

REFERENCES

[1] Mishra A., Dubey D.," A Comparative Study of Different Software Development Life Cycle Models in Different Scenarios", IJARSMS, Volume 1,Page 64-69,October 2013.

[2] Maheshwari S., Jain Dinesh Ch.,"A Comparative Analysis of Different types of Models in Software Development Life Cycle", IJARSMS, Volume 2, Issue 5, May 2012.

[3] Kute S., Thorat S. ," A Review on Various Software Development Life Cycle(SDLC) Models", IJRCCT, Volume 3, Issue 7, July – 2014

[4] Abrahamsson P., Warsta J., Siponen M. and Ronkainen J., "New Directions on Agile Methods: A Comparative Analysis", Proceedings of the International Conference on Software Engineering, May 3-5, 2003, Portland, Oregon, USA.

[5] Singh G., Tamanna, "An Agile Methodology Based Model for Software development", IJARCSSE, Volume 4, Issue 6, June 2014

[6] Wallin C., Land R., "Software Development Lifecycle Models The Basic Types"

[7] Ragunath P., Velmourougan S., Davachelvan P., Kayalvizhi S., Ravimohan R., "Evolving A New Model (SDLC Model-2010) For Software Development Life Cycle (SDLC)", IJCSNS, VOL.10 No.1, January 2010

[8] Dora S., Dubey P., "Software Development Life Cycle (SDLC) Analytical comparison and survey on Traditional and agile methodology", NMRJRST, VOLUME NO.2, ISSUE NO.8

[9] http://testingart.com/software-development-life-cycle-models/

[10] http://leansoftwareengineering.com/2008/05/05/boehms-spiral-revisited/