

IJEAST

INTERNATIONAL JOURNAL
OF ENGINEERING APPLIED SCIENCE
AND TECHNOLOGY

VOLUME : 2 ISSUE : 4 Print / Issue Publication Date: 31-Mar-2017

ISSN : 2455-2143

Indexed In

Google SCHOLAR OPEN ACCESS doi Crossref

WWW.IJEAST.COM

editor@ijeast.com

International Journal of Engineering Applied Sciences and Technology, 2017
Vol. 2, Issue 4, ISSN No. 2455-2143, Pages 294-299
Published Online February-March 2017 in IJEAST (http://www.ijeast.com)



A REVIEW ON VARIOUS SOFTWARE DEVELOPMENT LIFE CYCLE (SDLC) MODELS

Kumari Mamta
Department Of Computer Science
GVIET
Banur, Punjab-India

Shivani
Department Of Computer Science
GVIET
Banur, Punajab-India

***Abstract*—:- To satisfy the need of automate the different activities and satisfy the need of some standard and structural procedure or methodology which to be introduced in the industry so that the transition from manual to automated system became easy. This leads to the concept of System Development Lifecycle (SDLC) Models came into existence that emphasized on the need to follow some structured approach towards building new or improved system. There are five development models namely Linear Sequential Model, Iterative Model, Prototype Model, Spiral Model, V-shaped Model. All these models have cons as well as prons. The main objective of this research is to represent the different models of software development and make a comparative study of them to show the features and defects of each model.**

Keywords: Software Development Model (SDLC) and application, Linear Sequential Model, Incremental Model, Prototype Model, Spiral Model, V-Shape Model.

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.

Software life cycle models describe phases of the software cycle and the order in which those phases are executed. Each phase produces deliverables required by the next phase in the life cycle. Requirements are translated into design. Code is produced according to the design which is called development phase. After coding and development the testing verifies the deliverable of the implementation phase against requirements. The testing team follows Software Testing Life Cycle (STLC) which is similar to the development cycle followed by the development team.

There are following six phases in every Software development life cycle model:

Requirement gathering and analysis
Design
Implementation or coding
Testing
Deployment
Maintenance

1) Requirement gathering and analysis: Business requirements are gathered in this phase. This phase is the main focus of the project managers and stake holders. Meetings with managers, stake holders and users are held in order to determine the requirements like; Who is going to use the system? How will they use the system? What data should be input into the system? What data should be output by the system? These are general questions that get answered during a requirements gathering phase. After requirement gathering these requirements are analyzed for their validity and the possibility of incorporating the requirements in the system to be development is also studied.

Finally, a Requirement Specification document is created which serves the purpose of guideline for the next phase of the model. The testing team follows the Software Testing Life Cycle and starts the Test Planning phase after the requirements analysis is completed.

2) Design: In this phase the system and software design is prepared from the requirement specifications which were studied in the first phase. System Design helps in specifying hardware and system requirements and also helps in defining overall system architecture. The system design specifications serve as input for the next phase of the model.

International Journal of Engineering Applied Sciences and Technology, 2017
Vol. 2, Issue 4, ISSN No. 2455-2143, Pages 294-299
Published Online February-March 2017 in IJEAST (http://www.ijeast.com)

In this phase the testers comes up with the Test strategy, where they mention what to test, how to test.

3) Implementation / Coding: On receiving system design documents, the work is divided in modules/units and actual coding is started. Since, in this phase the code is produced so it is the main focus for the developer. This is the longest phase of the software development life cycle.

4) Testing: After the code is developed it is tested against the requirements to make sure that the product is actually solving the needs addressed and gathered during the requirements phase. During this phase all types of functional testing like unit testing, integration testing, system testing, acceptance testing are done as well as non-functional testing are also done.

5) Deployment: After successful testing the product is delivered / deployed to the customer for their use.

As soon as the product is given to the customers they will first do the beta testing. If any changes are required or if any bugs are caught, then they will report it to the engineering team. Once those changes are made or the bugs are fixed then the final deployment will happen.

6) Maintenance: Once when the customers starts using the developed system then the actual problems comes up and needs to be solved from time to time. This process where the care is taken for the developed product is known as maintenance.

II. SDLC MODELS

SDLC, Software Development Life Cycle is a process used by software industry to design, develop and test high quality softwares. The SDLC aims to produce a high quality software that meets or exceeds customer expectations, reaches completion within times and cost estimates.

- SDLC is the acronym of Software Development Life Cycle.
- It is also called as Software development process.

- The software development life cycle (SDLC) is a framework defining tasks performed at each step in the software development process.

- ISO/IEC 12207 is an international standard for software life-cycle processes. It aims to be the standard that defines all the tasks required for developing and maintaining software.

A typical Software Development life cycle consists of the following stages:

- Stage 1: Planning and Requirement Analysis
- Stage 2: Defining Requirements
- Stage 3: Designing the product architecture
- Stage 4: Building or Developing the Product
- Stage 5: Testing the Product
- Stage 6: Deployment in the Market and Maintenance

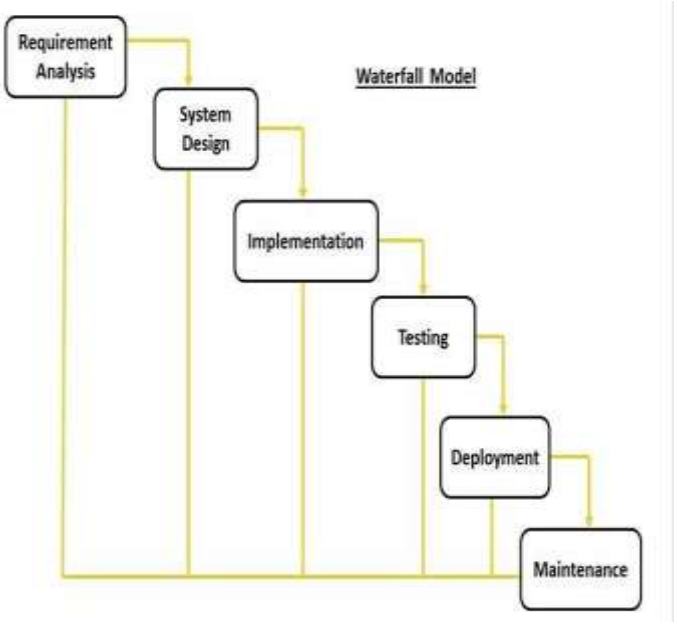
III. SDLC MODELS

There are various software development life cycle models defined and designed which are followed during software development process. These models are also referred as "Software Development Process Models". Each process model follows a Series of steps unique to its type, in order to ensure success in process of software development.

Following are the most important and popular SDLC models followed in the industry:

- Waterfall Model
- Iterative Model
- Spiral Model
- V-Model
- Big Bang Model

The other related methodologies are Agile Model, RAD Model, Rapid Application Development and Prototyping Models.



The sequential phases in Waterfall model are:

International Journal of Engineering Applied Sciences and Technology, 2017
Vol. 2, Issue 4, ISSN No. 2455-2143, Pages 294-299
Published Online February-March 2017 in IJEAST (http://www.ijeast.com)

- **Requirement Gathering and analysis:** All possible requirements of the system to be developed are captured in this phase and documented in a requirement specification doc.
- **System Design:** The requirement specifications from first phase are studied in this phase and system design is prepared. System Design helps in specifying hardware and system requirements and also helps in defining overall system architecture.
- **Implementation:** With inputs from system design, the system is first developed in small programs called units, which are integrated in the next phase. Each unit is developed and tested for its functionality which is referred to as Unit Testing.
- **Integration and Testing:** All the units developed in the implementation phase are integrated into a system after testing of each unit. Post integration the entire system is tested for any faults and failures.
- **Deployment of system:** Once the functional and non functional testing is done, the product is deployed in the customer environment or released into the market.
- **Maintenance:** There are some issues which come up in the client environment. To fix those issues patches are released. Also to enhance the product some better versions are released. Maintenance is done to deliver these changes in the customer environment.

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.

IV. APPLICATIONS OF WATERFALL MODEL:-

1) Whenever we are developing a small application we go for waterfall model.
2) When we have a short term project.
3) When we are sure that the requirement never changes. These are the stages in which we adopt the waterfall model.

V. ITERATIVE MODEL

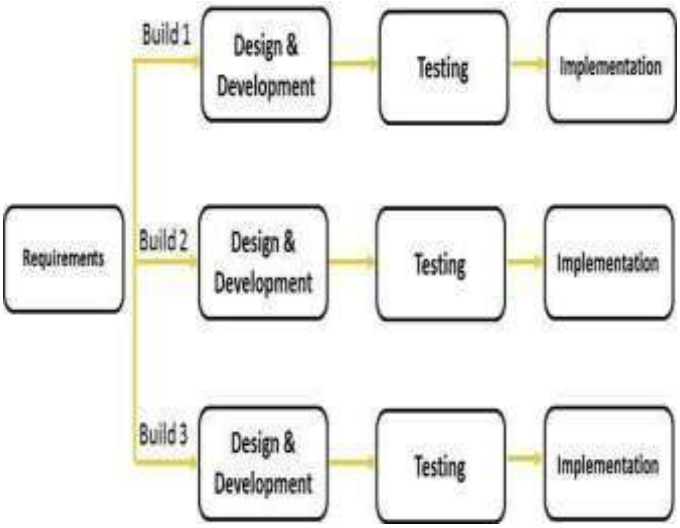
In Iterative model, iterative process starts with a simple implementation of a small set of the software requirements and iteratively enhances the evolving versions until the complete system is implemented and ready to be deployed.

An iterative life cycle model does not attempt to start with a full specification of requirements. Instead, development begins by specifying and implementing just part of the software, which is then reviewed in order to identify further requirements. This process is then repeated, producing a new version of the software at the end of each iteration of the model.

Iterative Model design

Iterative process starts with a simple implementation of a subset of the software requirements and iteratively enhances the evolving versions until the full system is implemented. At each iteration, design modifications are made and new functional capabilities are added. The basic idea behind this method is to develop a system through repeated cycles (iterative) and in smaller portions at a time (incremental).

Following is the pictorial representation of Iterative and Incremental model:



Iterative and Incremental development is a combination of both iterative design or iterative method and incremental build model for development. "During software development, more than one iteration of the software development cycle may be in progress at the same time." and "This process may be described as an "evolutionary acquisition" or "incremental build" approach."

In incremental model the whole requirement is divided into various builds. During each iteration, the development module goes through the requirements, design, implementation and testing phases. Each subsequent release of the module adds function to the previous release. The process continues till the complete system is ready as per the requirement.

International Journal of Engineering Applied Sciences and Technology, 2017
Vol. 2, Issue 4, ISSN No. 2455-2143, Pages 294-299
Published Online February-March 2017 in IJEAST (http://www.ijeast.com)



The key to successful use of an iterative software development lifecycle is rigorous validation of requirements, and verification & testing of each version of the software against those requirements within each cycle of the model. As the software evolves through successive cycles, tests have to be repeated and extended to verify each version of the software.

Iterative Model Application

Like other SDLC models, Iterative and incremental development has some specific applications in the software industry. This model is most often used in the following scenarios:

- Requirements of the complete system are clearly defined and understood.
- Major requirements must be defined; however, some functionalities or requested enhancements may evolve with time.
- There is a time to the market constraint.
- A new technology is being used and is being learnt by the development team while working on the project.
- Resources with needed skill set are not available and are planned to be used on contract basis for specific iterations.
- There are some high risk features and goals which may change in the future.

SPIRAL MODEL

The **spiral model** is a risk-driven process model generator for software projects. Based on the unique risk patterns of a given project, the spiral model guides a team to adopt elements of one or more process models, such as incremental, waterfall, orevolutionary prototyping.

This model was first described by Barry Boehm in his 1986 paper "A Spiral Model of Software Development and Enhancement".[2] In 1988 Boehm published a similar paper[3] to a wider audience. These papers introduce a diagram that has been reproduced in many subsequent publications discussing the spiral model.

[R]isk-driven subsetting of the spiral model steps allows the model to accommodate any appropriate mixture of a specification-oriented, prototype-oriented, simulation-oriented, automatic transformation-oriented, or other approach to software development.

Spiral Model design

The spiral model has four phases. A software project repeatedly passes through these phases in iterations called Spirals.

- **Identification:**This phase starts with gathering the business requirements in the baseline spiral. In the subsequent spirals as the product matures, identification of system requirements, subsystem requirements and unit requirements are all done in this phase.

  This also includes understanding the system requirements by continuous communication between the customer and the system analyst. At the end of the spiral the product is deployed in the identified market.

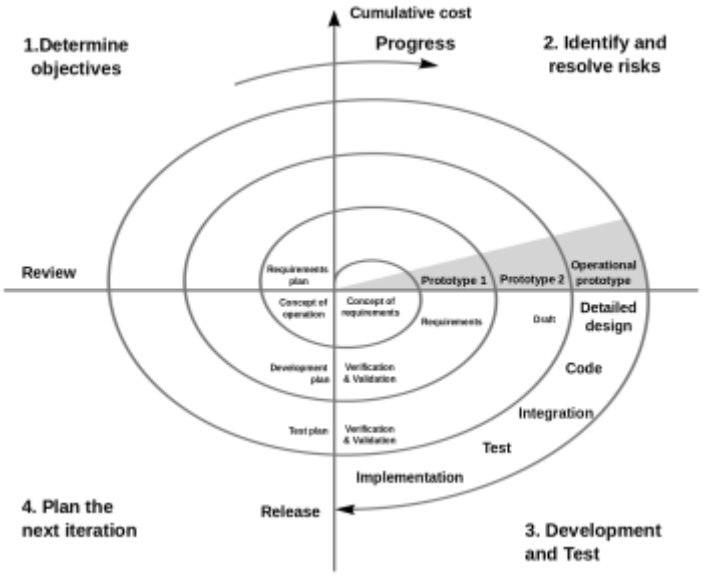
- **Design:**Design phase starts with the conceptual design in the baseline spiral and involves architectural design, logical design of modules, physical product design and final design in the subsequent spirals.

- **Construct or Build:**Construct phase refers to production of the actual software product at every spiral. In the baseline spiral when the product is just thought of and the design is being developed a POC (Proof of Concept) is developed in this phase to get customer feedback.

  Then in the subsequent spirals with higher clarity on requirements and design details a working model of the software called build is produced with a version number. These builds are sent to customer for feedback.

- **Evaluation and Risk Analysis:**Risk Analysis includes identifying, estimating, and monitoring technical feasibility and management risks, such as schedule slippage and cost overrun. After testing the build, at the end of first iteration, the customer evaluates the software and provides feedback.

International Journal of Engineering Applied Sciences and Technology, 2017
Vol. 2, Issue 4, ISSN No. 2455-2143, Pages 294-299
Published Online February-March 2017 in IJEAST (http://www.ijeast.com)



Based on the customer evaluation, software development process enters into the next iteration and subsequently follows the linear approach to implement the feedback suggested by the customer. The process of iterations along the spiral continues throughout the life of the software.

Spiral Model Application

Spiral Model is very widely used in the software industry as it is in synch with the natural development process of any product i.e. learning with maturity and also involves minimum risk for the customer as well as the development firms. Following are the typical uses of Spiral model:

- When costs there is a budget constraint and risk evaluation is important.
- For medium to high-risk projects.
- Long-term project commitment because of potential changes to economic priorities as the requirements change with time.
- Customer is not sure of their requirements which is usually the case.
- Requirements are complex and need evaluation to get clarity.
- New product line which should be released in phases to get enough customer feedback.
- Significant changes are expected in the product during the development cycle.

Spiral Model Pros and Cons

The advantage of spiral lifecycle model is that it allows for elements of the product to be added in when they become available or known. This assures that there is no conflict with previous requirements and design.

This method is consistent with approaches that have multiple software builds and releases and allows for making an orderly transition to a maintenance activity. Another positive aspect is that the spiral model forces early user involvement in the system development effort.

On the other side, it takes very strict management to complete such products and there is a risk of running the spiral in indefinite loop. So the discipline of change and the extent of taking change requests is very important to develop and deploy the product successfully.

The following table lists out the pros and cons of Spiral SDLC Model:

Pros	Cons
• Changing requirements can be accommodated. • Allows for extensive use of prototypes • Requirements can be captured more accurately. • Users see the system early. • Development can be divided into smaller parts and more risky parts can be developed earlier which helps better risk management.	• Management is more complex. • End of project may not be known early. • Not suitable for small or low risk projects and could be expensive for small projects. • Process is complex • Spiral may go indefinitely. • Large number of intermediate stages requires excessive documentation.



VI. REFERENCE

1. Lean Softwere Development : An Agile Tookit For Software Development Manager – By Mary Poppendieck , Tom Poppendieck , Ken Schwabr.
2. The Art of Software Testing – By Glenford J Mayers.
3. Angile Project Management With Scrum – By Ken Schwaber.
4. http://en.wikipedia.org/wiki/system_Development_Life_Cycle
5. http://en.wikipedia.org/wiki/Agile_software_development
6. http://agilemanifesto.org
7. http://www.agilealliance.org

IJEAST

INTERNATIONAL JOURNAL OF ENGINEERING APPLIED SCIENCE AND TECHNOLOGY

AUTHORS

Kumari Mamta

Shivani

ABOUT IJEAST

International journal of Engineering Applied Science and Technology (IJEAST) is a peer-reviewed, open access journal that publishes high - quality research paper in the field of Engineering, Applied Science and Technology.

IJEAST aims to provide a platform for researchers, academicians, and professionals to share their innovative ideas, research findings, and practical experiences with the global scientific community.

JOURNAL METRICS

H-INDEX	i10-INDEX	IJEAST CITATIONS	PEER-REVIEWED & OPEN ACCESS
33	154	9,647 (IJEAST CITATIONS)	PUBLISHED MONTHLY

PEER REVIEWED

All submission are rigorously peer reviewed to ensure quality.

AUTHORS ARE OUR PRIORITY

We value our authors and recognize their contribution to the advancement of research and knowledge.

OPEN ACCESS

Free and unrestricted access to research for all.

GLOBAL REACH

Connecting researchers and professionals worldwide.

TIMELY PUBLICATION

We ensure a swift and efficient publication process.

For more information, visit our website
www.ijeast.com

editor@ijeast.com

www.ijeast.com

India

2455-2143