IARJSET

ISSN (O) 2393-8021, ISSN (P) 2394-1588

International Advanced Research Journal in Science, Engineering and Technology

Vol. 8, Issue 9, September 2021

DOI: 10.17148/IARJSET.2021.8978

Study of Software Development Process & Exploration of SDLC Models

Barkha Gupta[1]

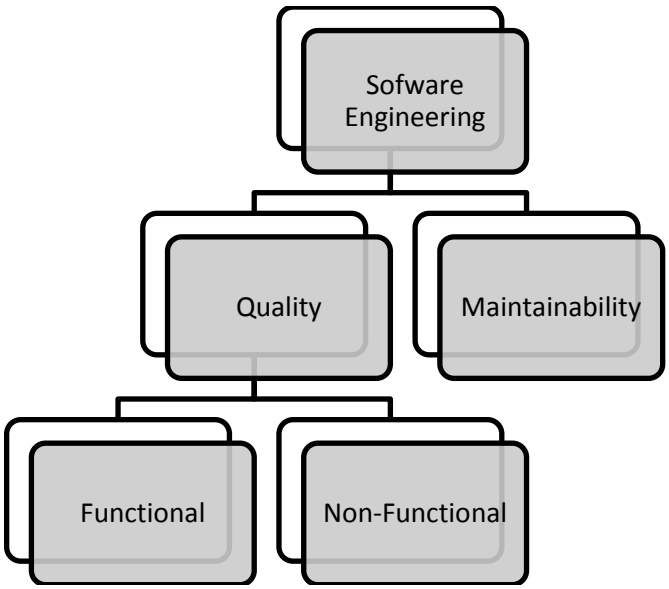
Programme Assistant, Department of Computer, Agriculture University, Jodhpur, Rajasthan, India[1]

Abstract: Software is a program, set of instructions or routine written for the computer to perform particular task. Software consists of programs, procedure used to setup and operate that particular software system and documentation of that program. Software engineering in broader term is a collection of software designing, developing, testing and maintaining the software. It shall act consistently with the public interest. It covers all the aspects which is mandatory in software production.

Keywords: SDLC, Software engineering, Software development cycle, Application development process, SDLC Models.

I. INTRODUCTION

Software engineering is a combination of two different words software and engineering. Software here means program, collection of codes that are written to perform a particular task and that fulfills particular requirements. Engineering on the other hand conveys that process of developing a product, using the appropriate methods and well-defined principles. In combination it becomes the process of analysing the user requirement then gathering the information as per need, designing the software, building, testing and then maintaining the software. Software Engineering is a discipline whose aim is the production of quality software within the timeline provided and within the budget given. It is a systematic approached that is used to develop a software. Software engineering focuses on two aspects that are Quality and Maintainability. Quality consists of Functional and Non-Functional. Functional represents degree to which correct software is produced or degree to which user requirement is correctly fulfilled. Non-Functional is also termed as Structural which contains features other than function like robustness, testing etc. Maintainability represents post-development software and this is considered after the software has been developed and delivered to the user.



II. VARIETY OF SOFTWARE

(i) **System Software**- System Software are that software which are designed to execute the computer's hardware. This software is necessary for the efficient working of a system. They also provide platform for the other software to

IARJSET
ISSN (O) 2393-8021, ISSN (P) 2394-1588
International Advanced Research Journal in Science, Engineering and Technology
Vol. 8, Issue 9, September 2021
DOI: 10.17148/IARJSET.2021.8978

run and execute. They also communicate with the computer hardware. Few examples are operating system like Microsoft Windows, Linux, Android and MacOS etc. and Compiler.

(ii) **Application Software**- Application software is that software which are meant to perform specific task that depends on personal needs and demands based on customer requirements as well. They are directly used by end users. This software can be in field of education, personal or business sector. Few examples are accounting software like tally, photo-editing software like Adobe Photoshop, CorelDRAW, Word processor or Media Player Software like VLC, KMPlayer etc.

(iii) **Engineering Software**- Engineering software is also known as scientific software. These are that software that are used to solve scientific problems or having complex numerical problems.

(iv) **Embedded Software** – Embedded software is a computer program written on firmware or a chip that is embedded on any machine, typically not the computer. It is used to give instructions to that machine and to control the functions of various hardware or machine. It provides limited functionality and features. Few examples are Microwave oven, Washing machine etc.

(v) **Artificial Intelligence Software**- This software possesses hum like intelligence into a machine. This software is capable of intelligent behavior. They are used to learn from the human behavior and by various data pattern and then they act or mimics like human being. They are capable of solving problems as well. Few examples are Alexa, Siri and various robots in gaming etc.

(vi) **Mobile Application Software**- This software which designed to run on a mobile having a particular operating system (Android, iOS etc.) installed on that device. This software is commonly known as apps. Few examples are Play Store, MX Player etc.

(vii) **Legacy Software** – This software is developed years ago and became outdated now. They are very old software and frequent updation and modification is required by the organization to meet the business requirements which leads to huge expenses and waste on time as well.

III. SOFTWARE EVOLUTION & ITS LAW

Software evolution is a process of developing a software product using software engineering principles and methods. Software evolution law is given by Lehman. Software evolution law has three categories: -

(i) **S-type** – S-type is an acronym of static type which implies fixed or least subject to changes are allowed and hence it is simplest type among all. This type strictly according to the defined specifications and solutions. Example of S-type is calculator.

(ii) **P-type** – P-type is an acronym of practical type. It is collection of procedures where specifications can be described but the solution is not that obvious instantly. Example of P-type is gaming software.

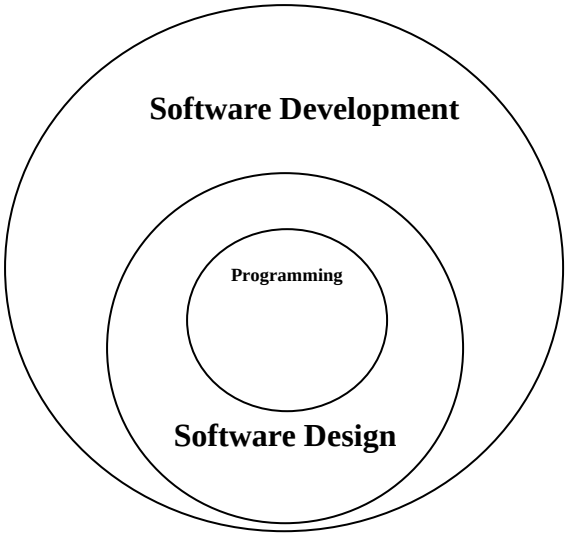
(iii) **E-type** – E-type is an acronym of embedded type which works according to the requirement of the real world. In this type, high degree of changes is allowed. Example of E-type is online trading software. The E-type software evolution has 8 laws which are as follows: -

a) Continuing changes
b) Increasing complexity
c) Continuing growth
d) Conservation of familiarity
e) Reducing cost
f) Feedback system
g) Self-regulation
h) Organisation stability

IARJSET
ISSN (O) 2393-8021, ISSN (P) 2394-1588
International Advanced Research Journal in Science, Engineering and Technology
Vol. 8, Issue 9, September 2021
DOI: 10.17148/IARJSET.2021.8978

IV. SOFTWARE PARADIGM

Software paradigm is the methods or steps that is to be followed while designing and developing a software.



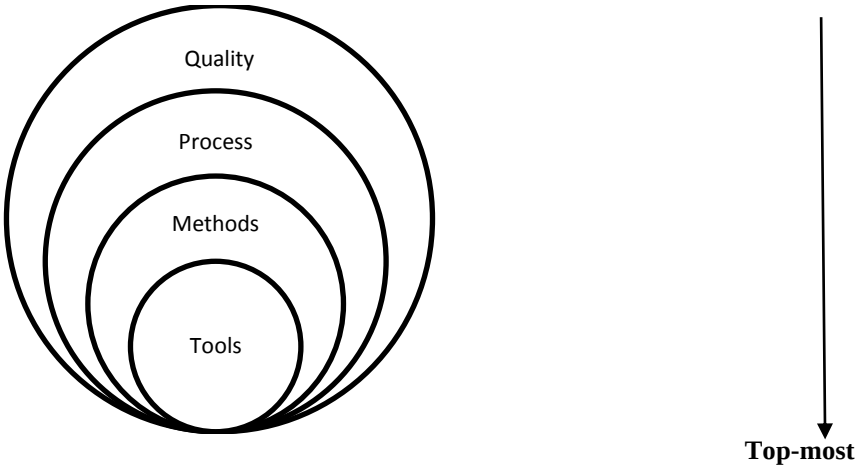
(i) **Software Development Paradigms** – Software development paradigm is also known as software engineering paradigm. It includes requirement gathering, software design and programming of building a software.

(ii) **Software Design Paradigms** – This includes design, maintenance and programming of a software.

(iii) **Programming Paradigm** – It includes coding, testing and integration of a software

V. LAYERED TECHNOLOGY

Bottom - most



(i) **Quality**

Quality is a bottom-most layer whose focus is on the quality of a software. This is the major issue faced while developing the software to meets its quality which is demanded by its customer. A product should meet its specification in terms of its quality and this layer focus on this parameter.

IARJSET
ISSN (O) 2393-8021, ISSN (P) 2394-1588
International Advanced Research Journal in Science, Engineering and Technology
Vol. 8, Issue 9, September 2021
DOI: 10.17148/IARJSET.2021.8978

(ii) **Process**

This is second layer of the layered technology. This layer is core of the software engineering layer as process determines the way in which software will be produced. It is collection of steps or activities that leads to a part of a product. Process is also a collection of methods.

(iii) **Methods**

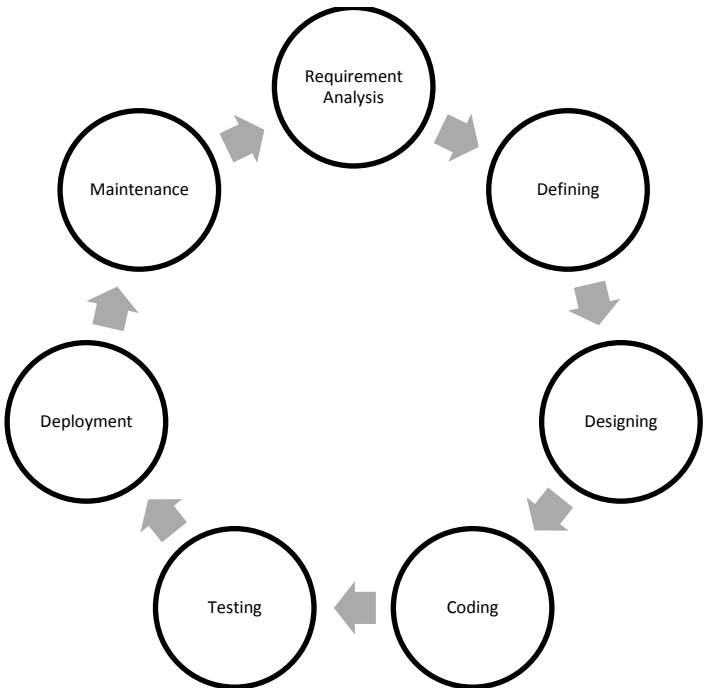
Methods is the third layer of the layered technology. It is broad array of stack which consists of several things. It includes requirement analysis, design, program construction, testing and support. It shows "how-to" perform a task.

(iv) **Tools**

Top-most layer of layered technology is tools. These tools here refer to automated or semi-automated task of the process which is available readymade. When these tools are integrated, they make the task easy as the information which is created by one tool can directly be used by the other tools instead of creating it again. So, this also provide the features of reusability.

VI. SOFTWARE DEVELOPMENT LIFE CYCLE

SDLC is the acronym of Software Development Life Cycle is a diagrammatic representation of software life cycle. It consists of all the activities or steps that are mandatory for the development of a software. This is a process used by the organisation to design, develop and test the software and generate high quality software. SDLC is also termed as process model or Application Development Life Cycle or Software Development Process. The period of software development starts when a software product is conceived and its ends with when then software is no longer required by the user or not available to use. Thus, SDLC is a framework defining each smaller task performed at each level to develop a software. The following figure shown below is a graphical representation of the various stages of a typical SDLC.



(i) **Planning and Requirement Analysis**– This is first phase and fundamental phase of SDLC. This is performed by top level managers and experts with the input given by the customer. The information here is used to define the very basic approach to develop a software. Identification of risk associated with the software and its quality assurance is also determined at this phase.

IARJSET

ISSN (O) 2393-8021, ISSN (P) 2394-1588

International Advanced Research Journal in Science, Engineering and Technology

Vol. 8, Issue 9, September 2021

DOI: 10.17148/IARJSET.2021.8978

(ii) **Defining** – Once the previous stage is completed, then this is second stage of SDLC. Once the requirements of the user are understood and information is gathered Software Requirement Specification (SRS) has been developed here. Once the SRS is developed it has to be approved by the customer which contains all the product requirements in detailed manner.

(iii) **Designing** –Based on the SRS, the design of the software in terms of its architecture is proposed here. Usually more than one design with its documentation which is known as Design Document Specification (DDS) is developed here and given for the approval to senior authority and the user. Based on various parameters like user friendly, product robustness, time and money constraint the decision has been taken.

(iv) **Coding or Implementation** – This is the main programming stage of SDLC where the main task of building the software begins. The programmer starts here the coding part as per mentioned in DDS. Developers has to follow the guidelines given by their organisation. There are different programming languages available like C, C++, Java, Phyton, PHP etc. Choice can be made by the programmer as per the requirement of the software.

(v) **Testing** – Once the Programming part is finished, then the testing phase begins. This includes testing of each and every module and ensure that they are working fine as mentioned by the user and fulfilling their requirements. This phase is done to find out the errors, bugs, defeats in the software. Once the bugs are fixed, re-testing is done until and unless bug-free software is produced and product reaches the quality standard defines in the SRS.

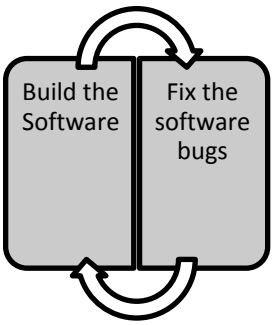
(vi) **Deployment** – Once the software has become error free, then it is ready to deploy which means ready to release in the official market where the common users can use the software. In some cases, depending on the requirement, software may first be released in limited number of users to take the feedback and then do the changes accordingly and then re-launch it to all the user or in open market. This type of limited launch is also known as user acceptance testing.

(vii) **Maintenance** – Once the software is deployed or launched in the live environment and used my ample of users then there comes the requirement of maintaining the software. While using the software in real-time, there may be possibility of certain bugs or error may appear or software may outdate as per the platform. So, developers have to maintain that software and keep them working on changing environment.

VII. SDLC MODELS

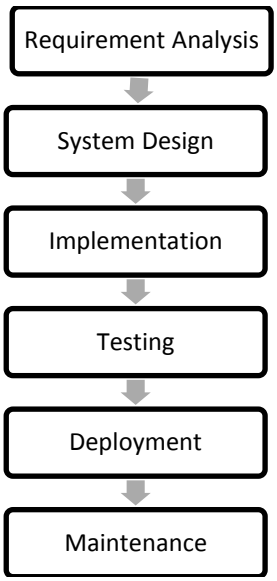
There are different SDLC models given and each having different phases that has to be followed in a particular sequence to develop a software. Each model is different from other models on basis of its characteristics.

(i) **Build and Fix Model** – This model is the simplest model with not well-defined structure. It consists of two phases –Build and Fix. The first phase is to build the software which means do the programming as per the requirements and the next phase is to fix its error or bugs and then repeat this process until the final accepted software is build.

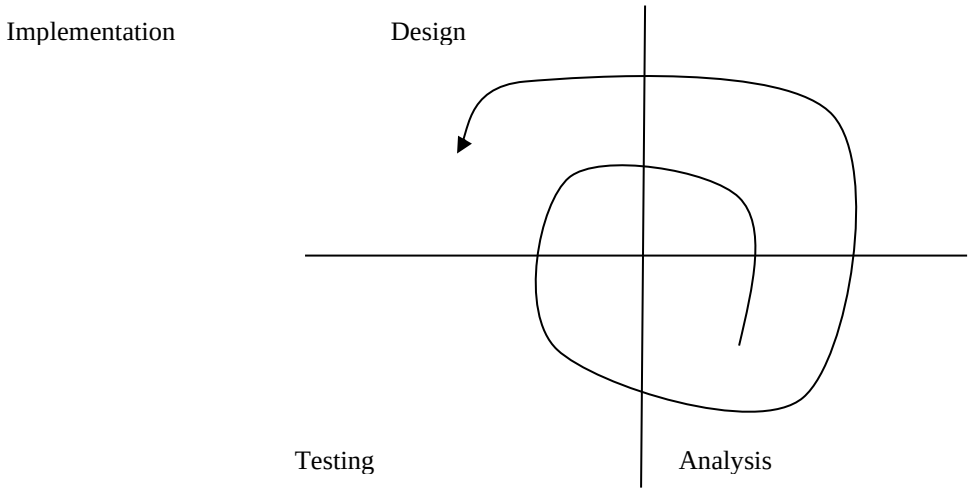


(ii) **Waterfall Model** – This model is also referred as Classic Life Cycle Model or Traditional Model or Linear Sequential Life Cycle Model. This is most commonly and widely used model in SDLC. It is a sequential model with no overlapping of any two stages which moves steadily downwards just like waterfall. The below mentioned diagram specify the same and various phases included in waterfall model. Each phase is explained in detail above in the topic SDLC cycle. Waterfall model maintains that one should move to next phase only when its previous phase is completed and there is no jumping or skipping of any phase is allowed.

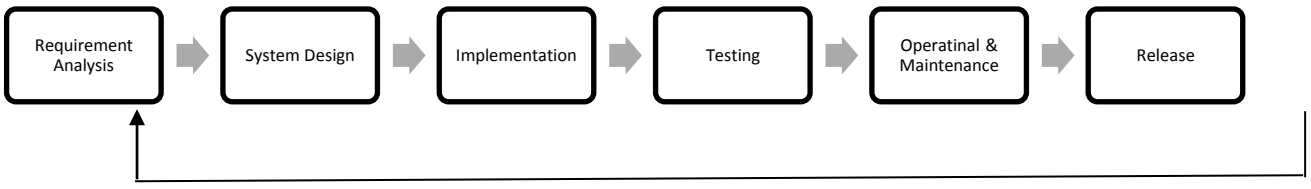
IARJSET
ISSN (O) 2393-8021, ISSN (P) 2394-1588
International Advanced Research Journal in Science, Engineering and Technology
Vol. 8, Issue 9, September 2021
DOI: 10.17148/IARJSET.2021.8978



(iii) **Spiral Model** – This model is proposed by 'Barry Boehm'. In this model activities are arranged in spiral manner. This model mainly focusses on handling risk and uncertainty. Spiral starts from inward and move toward outward in an anti-clock wise direction. The basic idea behind this model is to developed in increments, where each increments adds some additionality functionality to the previous ones and this keep repeating until final software developed.



(iv) **Incremental Model** – In this model requirements are divided into different smaller parts and each part is separately developed and released. This process repeats in loop and keep adding to its functionality and at the end of the cycle will give completed software. In this multiple cycle are used and each cycle gives one release. Each previous cycle acts as base for the next cycle.



IARJSET

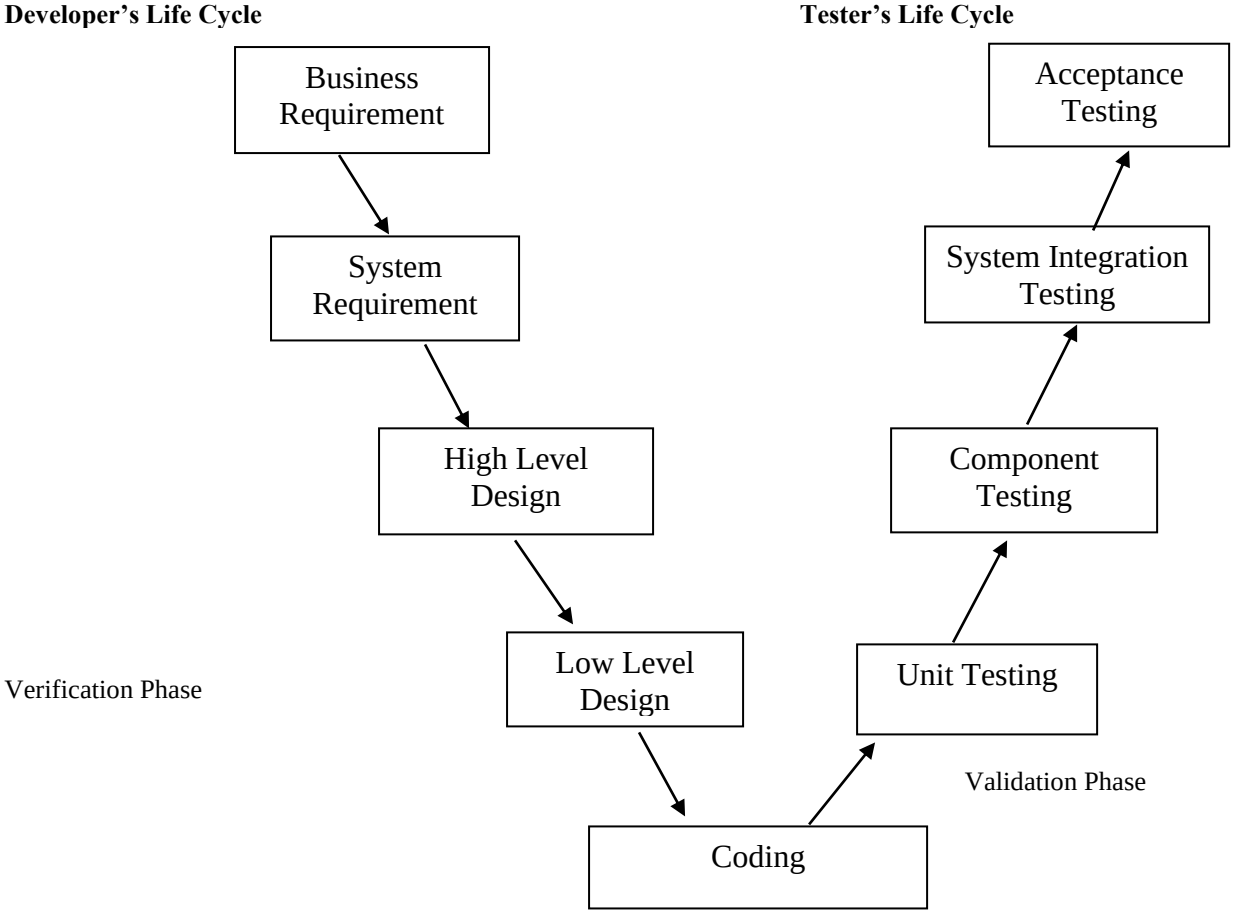
ISSN (O) 2393-8021, ISSN (P) 2394-1588

International Advanced Research Journal in Science, Engineering and Technology

Vol. 8, Issue 9, September 2021

DOI: 10.17148/IARJSET.2021.8978

(v) **V Model** – V model referred as Verification and Validation model where the left-hand side denotes verification phase and right-hand side denotes validation phase. This model works in a parallel fashion and join at coding phase. Verification is a static process where analysis, design and developing take place this side is totally depends on developer's hand. On the other hand, verification phase is a dynamic analysis where the testing take place to determine whether the software meets the customer requirements and expectation.



(vi) **RAD Model** – RAD Model acronym of Rapid Application Development Model is a linear sequential development model with active user participation in every phase. This model was initially proposed by IBM. In this model, major emphasis is on development process rather than planning. In this model, the final product is developed faster and of higher quality as this model uses powerful development tools and advanced techniques. The various phases in this model are-

a) **Business Modelling** – The information in this phase flow among business functions which is defined by answering questions like what data is to be generated, who generates that data, where does the information go, who will process that information and many others.

b) **Data Modelling** – The data collected from previous stage is been refined here. These data is further divided into set of data objects called entities that are needed to support the business. The attributes which is the characteristics of an entity are identified, and the relationship between these entities are also determined here.

c) **Process Modelling** – The information collected at data modelling phase are transformed to achieve the data flow necessary to implement a business function. Processing descriptions are created for adding, modifying/altering, deleting, or retrieving a data object.

d) **Application Generation** –The tools used here is automated or semi-automated tools of the process which is available readymade. When these tools are integrated, they make the task easy as the information which is created by one tool can directly be used by the other tools instead of creating it again. So, this also provide the features of reusability.

e) **Testing & Turnover –** By suing many ready-made tools, reduce the load of testing the module. But the newly integrated module must be tested for the fully functioning of the software and its all interfaces must be fully exercised.

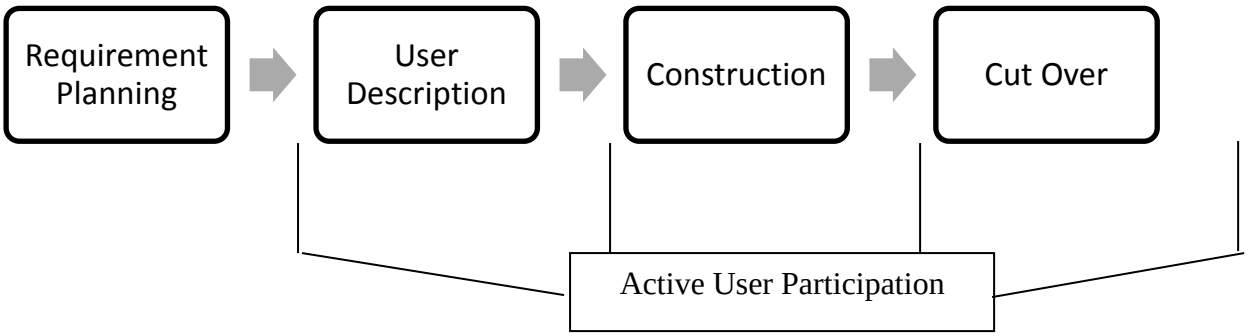
IARJSET

ISSN (O) 2393-8021, ISSN (P) 2394-1588

International Advanced Research Journal in Science, Engineering and Technology

Vol. 8, Issue 9, September 2021

DOI: 10.17148/IARJSET.2021.8978



VIII. CONCLUSION

Software development cycle is study of designing the software, developing it and maintaining it for future use. This cycle starts from the user's requirement and works till the retirement of a software. We have extended our knowledge to know that what a development process would need to build good software. There is no single model which is best in every scenario or gives the best output in every case. It always depends on the type of software building and the need or requirement of the user. However, there must be a disciplined approach to the software development, especially in the case of large projects.

We also emphasis on that the good approach is to divide the larger project into smaller modules. This will help in working with team as well as its modular testing will be easy and bugs can be detected at early stages. wherever possible, make use of replaceable, ready-made and reusable modules or components. The overall software architecture should be constructed around the users' requirements.

IX. ACKNOWLEDGEMENTS

Author is grateful to Agriculture University, Jodhpur for encouraging writing in my field.

X. REFERENCES

[1]. K K Aggarwal. Yogesh Singh. Software Engineering. Third Edition. New Age International Pvt. Ltd.
[2]. Roger S Pressman. Software Engineering: A Practitioner's Approach. McGraw Hill Education Publisher. Seventh Edition.
[3]. Pankaj Jalote. Software Engineering: A Precise Approach. Wiley Publisher.
[4]. Richard Murch. The Software Development Lifecycle: A Complete Guide. Kindle Edition
[5]. Deepak Jain. Software Engineering Process Models. BPB publications.
[6]. Gerardus Blokdyk. Software Development Life Cycle SDLC A Complete Guide. Kindle Edition. 5STARCooks Publisher.