ISSN No. 0976-5697

Volume 8, No. 1, Jan-Feb 2017

International Journal of Advanced Research in Computer Science

RESEARCH PAPER

Available Online at www.ijarcs.info

A Study of Software Development Life Cycle Process Models

Kazim Ali
MS Computer Science
Department of Computer Science, Lahore Leads University
Main Campus, Kalma Chouk Lahore, Pakistan

***Abstract*:** The software development life cycle (SDLC) is used to design, develop and produce high quality, reliable, cost effective and within time software products in the software industry. This is also called software development process model. There are different SDLC process models are available. In this paper I have tried to describe different SDLC models according to their best use. There are many papers which have written in this regard. I will also use their knowledge or findings in this paper. The main purpose of this paper is to explain some of important SDLC models like Waterfall Model, Iterative Model, Spiral Model, V-Model, Big Bang Model, Agile Model, Rapid Application Development Model and Software Prototype. The main purpose of this paper is to explain advantages and disadvantages of these SDLC models. I will also describe which SDLC model is best fit for which type of software applications.

***Keywords*:** Waterfall Model, Agile Model, RAD, Software Prototype

I. INTRODUCTION

All SDLC processes consist of a set of finite activities which are used to develop a software product. A SDLC process contains a complete plan for describing how to design, develop, maintain and increase the efficiency of a software product [1]. The SDLC process describes the methodologies which improves overall software quality and development process [2]. The following figure shows the different phases of a typical software development life cycle.

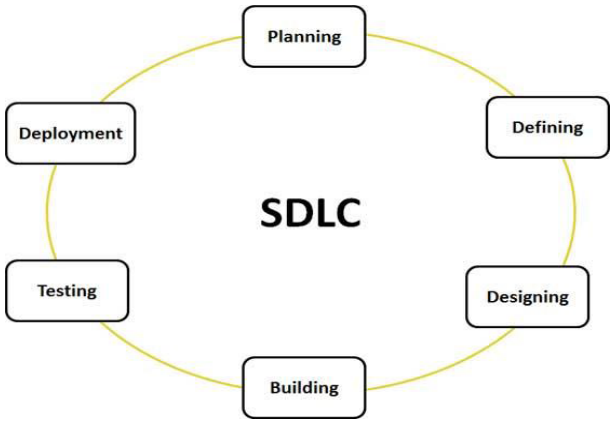

Fig. 1

I explain these phases of a typical SDLC process in a short and precise manner which is give below.

1) Planning

Planning and requirement analysis is the most vital and basic phase of every life cycle process. It is completed by the senior members after held meeting with customer or owner of the software system. Quality assurance requirement, risk identification and feasibility report are the main outcomes of this phase.

2) Defining Requirement

When planning and requirement analysis is completed then the next phase is the detailed definition of the requirements, document these requirements and get verification from the customer. The output of this phase is the software requirement specification (SRS) document which contains all the requirements of product to de designed and developed [6].

3) Designing the Software Architecture

Software requirement specification is used as input to design the architecture of the software product which is being developed. Initially more than one architecture are designed and then reviews by important stakeholders according to criteria like risk assessment, robustness, design modularity, time and cost, the best design is selected [6].

4) Building or Developing the Product

In this phase, actual development of the product starts according to the designed architecture. If designing is done successfully then this phase is not much difficult. Developers use different tools such as compilers, interpreters and debuggers are used to generate code. Different programing languages like C, C++, Pascal, Java and PHP are used. Programing language depends upon type of software being developed.

5) Testing

In this phase developed product is tested whether it meets user' requirements which are documented in software requirement specification document. Software defects are reported, tracked, fixed and retested so that the product has gained high quality.

Deployment and Maintenance

After testing the product, it is deployed in its actual environment. Where customer verify his requirements which is called acceptance testing. After the feedback of customer, the further enhancement is done [6].

II. SOFTWARE DEVELOPMENT LIFE CYCLES MODELS

The most important and popular SDLC models are given below.

A. *Waterfall Model*

It is the first software development process model. Waterfall model is a sequential process model which does not overlap. It means that until the one phase is not completed then next phase cannot start. It is simple and easy to understand [6]. The graphical representation of water fall model is give below.

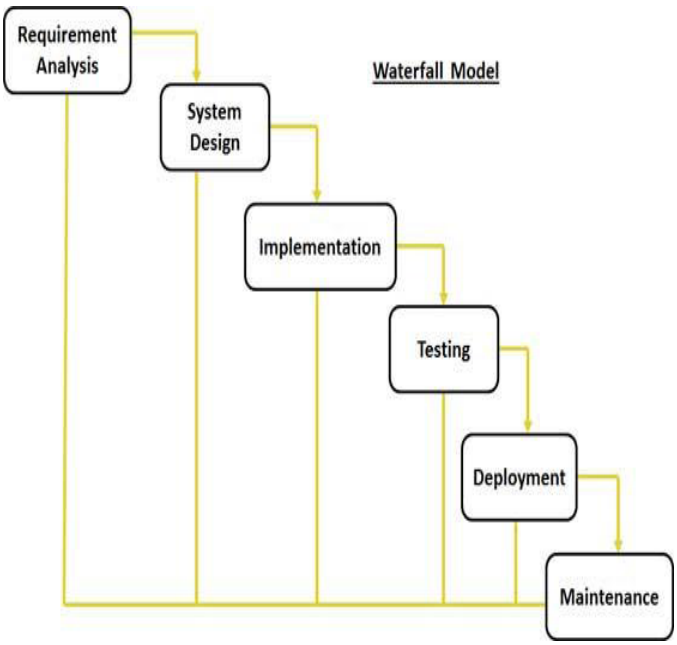

Fig. 2

The phases in this process model are:

1) *Requirement Analysis*

In this phase, all requirements of the software product are gathered in this phase and documented in software requirement specification document [4].

2) *System Design*

In this phase, overall structure of the software product is designed on the basis of requirement analysis phase.

3) *Implementation*

The development of software starts in this phase. It develops in small programs which are called units. These units are tested according to their functionality and integrated in the next phase.

4) *Testing*

In this phase, integrate the all units which are developed in implementation phase. After integration the whole product is tested to check whether it meets its goals. Software defects and bugs are reported, if they are available then fix and retested.

5) *Deployment*

When function and non-functional requirements are tested and validated then the software is deployed in the customer's environment.

6) *Maintenance*

If some problems are faced in the customer's environment then solve these problems in maintenance phase. Also some enhancement can do in this phase if user is not fully satisfied. [4]

7) *Waterfall Model Application*

It is impossible that one SDLC model is fit for all types of software application. So it is very important to choose best software process model to specific software. I am tried to list some situations where waterfall model is best fitted which are

- Where software requirements are well understood, documented, cleared and fixed.
- Software definition is stable.
- Technology is understood and not dynamic.
- There is no ambiguity in software requirements.
- The required resources are available.
- The Project is small.

8) *Advantages and Disadvantages of Waterfall Model*

The advantage and disadvantage of waterfall model are given in the following table.

Table I

Advantages	Disadvantages
• It is very easy to understand and use.[4] • Each phase has a specific deliverable and review process.[5] • Phases are processed and completed one at a time. • Works well for small projects where requirements are well understood. • Clearly defined phases. • Well understood deliverables. • Arranging tasks easily. • Process and results are documented.	• Working software is available late during the life cycle. • It has a lot of risks. • A poor model for large projects. • It is not fit for projects where requirements are changed frequently.[4] • Difficult to measure progress during phases. • It cannot manage changing requirements. • Adjusting scope during the process model can end a project.

B. *ITERATIVE MODEL*

In iterative model, requirements are not completed and started iterative process with a small set of requirements. Each iteration evolves a small version of product and it is repeated until the final version is developed. Iterative process model starts implementation with a subset of requirement specifications. Each iteration is added new functionality in the

process and continuous until it is completed [6]. The graphical representation of iterative model is given below.

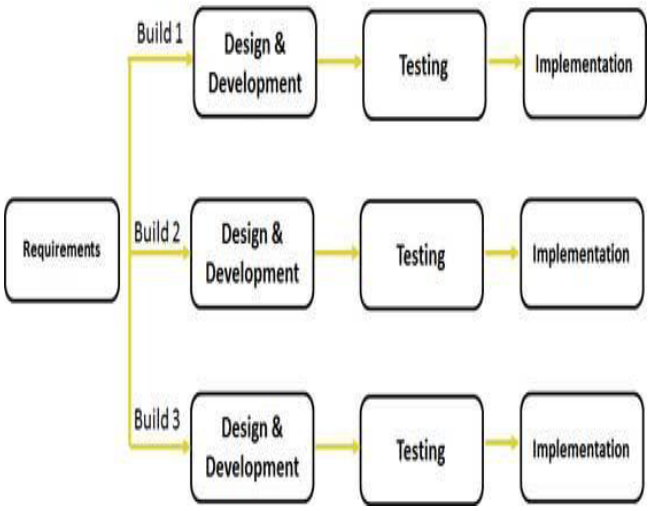

Fig. 2

*1) **Iterative Process Model Application***

Like other SDLC model, Iterative model is also not fit for every application. However, this model is often used in the following scenario:

- When complete requirements of the system are cleared and well defined.
- Important and major requirements must be completed however some functionality may evolve with time.
- There is a time to market constrain.
- A new technology is being used while working on the project.
- Resources with needed skill set are not available and planned to be used on contract basis for specific iterations.

*2) **Advantages and Disadvantages of Iterative Model***

Advantages and disadvantages of iterative SDLC model are shown in table given below.

Table2

Advantages	Disadvantages
• Some of important working functionality is available quickly. • Results are derived soon and quickly. • Parallel development can possible. • Project progress can be measured. • Requirement change process within budget. • In small iteration testing and debugging is easy. • Risk identification and milestone management is easy. • Most risky part is done first there easy manage high risks. • After every iteration functional product is delivered. • Issues, challenges and risk which get from one iteration are applied to the next iteration. • Better risk analysis. • It supports changing requirements environment. • Minimum initial operations. • It is best for large and critical products. • During iterative model software product is developed early which facilitates customer evaluation and feedback report.	• Many resources are required. • Although requirement changing cost is low but not much appropriate for changing requirement. • More management is required. • System design issues are raised because not all requirements are available in starting. • Complete system definition is required for iterations. • Not best for small and tiny projects. • At the risk is unknowable. • For risk analysis and identifications, skilled persons are required. • Project progress depends upon risk analysis.

C. SPIRAL PROCESS MODEL

This SDLC model is the combination of iterative model and sequential model like waterfall model. The spiral model combines the idea of iterative and waterfall development in a very systematic and controlled way [6]. The diagram of spiral model is given below:

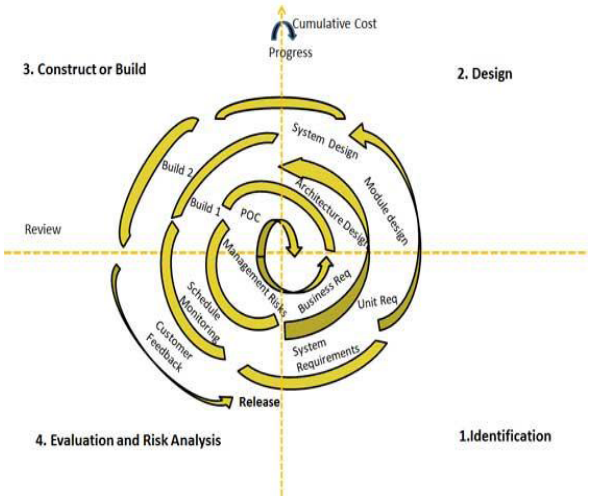

Fig. 3

Spiral model consists of four models which are:

1) ***Identification***

The purpose of this phase is to collect business requirements. The system requirements, subsystem requirement and unit requirements are all done in this initial phase. For this purpose continuous communication with customer is done and at the end system is deployed in the customer‘s environment.

2) ***Design***

In this phase, architectural design, logical design of modules, physical design and final design of subsequent spirals are included.

3) ***Construction***

In this phase, actual development is started. In baseline phase only proof of concept is developed to get customer feedback. When requirements and design details are cleared then a working model is developed which is called build and send this build to the customer for further feedback.

4) ***Evaluation and Risks***

In this phase, identifying, estimating and monitoring technical feasibility and management risk for example schedule slippage and cost overrun. The build developed in first iteration is sent to customer for evaluation and feedback. The next development starts based on the customer feedback in the next iterations. The iteration process continuous through whole life cycle.

5) ***Spiral Model Application***

Spiral model is widely used in the industry. The situations in which spiral model are use are given below:

- Constraints on budget and risk evaluation are important.
- Medium to high projects.
- Where long project commitment due to changing requirement environment.
- When customer does not know his complete requirements.
- Requirements are complexe and need evaluation to get clarity.
- Where changes are expect in during life cycle.

6) ***Advantages and Disadvantages of Spiral Model***

Table consists of advantages and disadvantages of spiral model is:

Table 3

Advantages	**Disadvantages**
• Continuous changing requirements can be managed. • Much use of prototypes is allowed. • Requirements collect more accurately. • Users see the system early. • Development process is divided into parts and risky parts are developed early which help better management.	• Complex management. • Project completion duration is not known. • Not suitable for low risky and small projects. • Process is complex. • Spiral can go indefinitely. • Large number of phases requires heavy documentation.

D. ***V-Model***

In V-Model, the processes occur in a sequential order in V-shape. It is an expansion of waterfall model, with each phase of V-Model, a testing phase is associated. This is also called verification and validation model. Its means every single phase in this connect with a testing phase. Therefore it is very controlled and disciplined model. The validation phases are the one side of V and the verification on the other, coding phase joins these phases which are shown in the figure below:

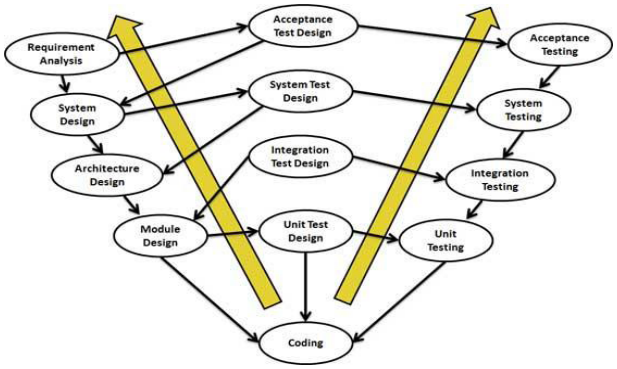

Fig. 4

The verification phases in V-Model are given below:

1) ***Requirements Analysis***

This is the initial phase of V-Model. Detailed communication with the customer is done and determined exact requirements and expectation of the customer. Some customer cannot know or describe their needs and requirements so this phase is very important and well managed. Therefore acceptance test is done and customer’ needs used as the inputs of acceptance test.

2) ***System Design***

When we have captured exact and detailed requirements of the product, it is time to design the complete system. System test plan is developed depends on the system design.

3) ***Architectural Design***

In this phase, architectural specifications are explained and designed. Mostly many technical approaches are chosen but final decision is done based on financial and feasibility report. System designing is decomposed into modules according to functionality. This is also called High Level Design.

4) **Module Design**

In this phase the internal designed of the modules is specified, referred as Low Level Design. The design should

compatible with other system modules. Also unit testing is done on the base of internal design.

*5) **Coding Phase***

In the phase, the coding of the system modules which are designed during design phase, is started. The best programming language is designed according to system and architectural requirements. The code based on the some guidelines and standards. The coding passes through many review process before the completion.

The validation phases in V-Model are:

*6) **Unit Testing***

The unit tests which are designed in the designing phase are executed on coding during this validation phase. Unit testing can find defects and bugs at early level but all defects cannot find by unit testing.

*7) **Integration Testing***

Integration testing is connected with the architectural design phase. Integration tests are executed to check coexistence and communication of the internal modules within the system.

*8) **System Testing***

System testing are directly associated with design phase. System tests check the functionality of the whole system and communication of the system with external systems. Most of the hardware and software issues are found during the system testing.

*9) **Acceptance Test***

Acceptance testing is done in the user's environment and it is associated with the business requirement of the customer. It founds compatibility issues with other system at user side and also non-functional defects like load and performance.

*10) **V-Model Applications***

The conditions in which v-Model is much useful are:

- Where software requirements clear, defined and fixed.
- Stable project definition.
- Technology is stable and well trained by development team.
- No ambiguity in requirements.
- Where project is short.

*11) **Advantages and Disadvantages of V-Model***

The following table list out the advantages and disadvantages of V-Model:

Table 4

Advantage	Disadvantages
• This is a highly disciplined model and phases are completed one at time. • Well suitable for small and tiny projects where requirements are well understood. • Simple and easy to use. • Easy to manage because each phase provide a deliverable with review process.	• Risks and uncertainty is high. • Not suited for complex and object oriented projects. • Poor model for large and ongoing projects. • Not suitable where requirements are changed. • If application is in testing phase, difficult to go back for changing functionality. • Working software is available at the end of life cycles.

*E. **Big Bang Model***

The Big Bang model is a SDLC model where there is no specific process and development method is followed. The development process starts with only money and efforts as input and the output is the developed software which may or may not be as per customer requirement.

*1) **Big Bang Model Applications***

The following scenario is best for Big Bang model:

- Where very little or no planning.
- Requirements are understood.
- Ideal for small projects.
- Useful for academic projects.
- Completion date is not given.

2) ***Advantages and Disadvantages of Big Bang Model***

The following table shows the advantages and disadvantages of this model:

Table 5

Advantages	Disadvantages
• It is very simple model. • Little or no planning is required. • Easy to manage. • Very few resources are required. • Provide flexibility to developers.	• Very highly risky. • Not good for complex and object oriented model. • Poor model for large project. • Very expensive if requirements are understood.

*F. **Agile Model***

Agile SLDC model is a combination of the iterative and incremental processes. It concentrates on process adaptability and customer satisfaction by rapid delivery of working software product. Agile method break software product into small incremental build. These incremental builds are completed in iterations. In each iteration development team works on the planning, requirement analysis, design, coding, unit testing and acceptance testing. At the end of iteration a working build is submitted to the customer and stockholder. Agile model ensures that every software project handled differently and different exiting methods apply on it, based on product requirements. In agile tasks are divided into different time boxes based features. Then iterative approach is used and a working product is delivered at the end of iteration. These iterations continuous till the final software version with all the features, is delivered. The graphical representation of agile process models is:

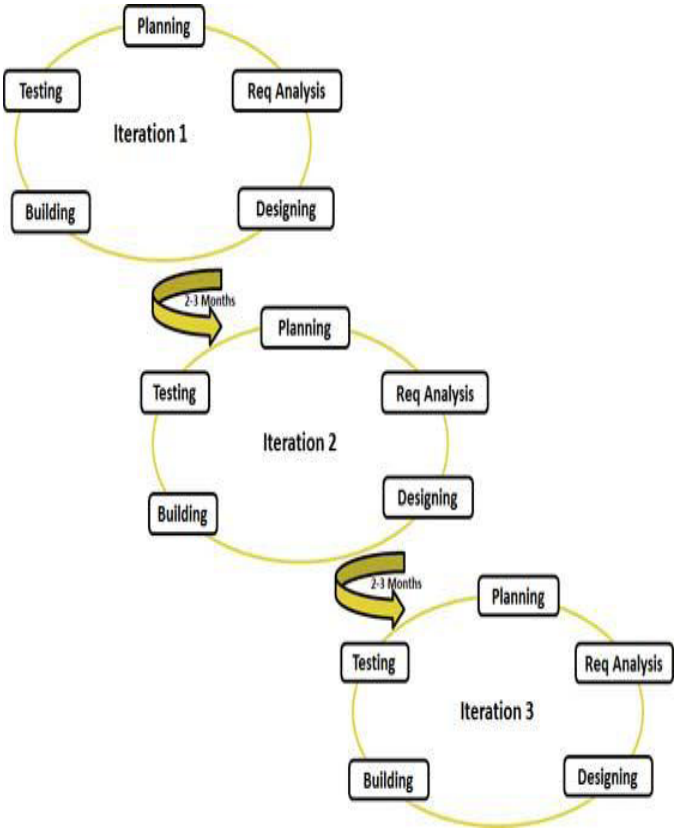

Fig. 5

The most popular agile methods are Rational Unified Process (1994), Scrum (1995), Crystal clear, Extreme Programming (1996), Adaptive Software Development, Feature Driven Development and Dynamic Systems Development (1995). After the publication of Agile Manifesto in 2001, these methods are now called agile methodologies. The Agile Manifesto principles are:

1) Individuals and Interactions

Self-organization, motivation and pair programming are important in agile development.

2) Working Software

Demo working software is presented to customer for understanding requirements instead of documentation.

3) Customer Collaboration

Since requirements gathering is not completed at initial stage so continuous communication to the customer is very important.

4) Responding to Change

The basic purpose of agile development is to rapid response and continuous development.

5) Advantages and Disadvantages of Agile Model

The following table list out the advantages and disadvantages of agile model:

Table 6

Advantages	Disadvantages
• This is a realistic approach to develop software project. • It encourages team work and cross training. • Features and functionality are developed quickly. • Minimum resources are required. • Appropriate for fixed and changing requirement. • Early delivery. • Good model for changing environment. • Minimal rules and documentation is easily employed. • Concurrent development occur and delivery within plan time. • Little planning is required. • Easy manageable. • Flexible for developers.	• Not appropriate for complex dependencies. • Risks for maintainability, extensibility and sustainability. • Strict delivery dictates scope, functionality to be delivered and adjustments to meet the deadlines. • Much depends on customer communication if customer is not available then developing team can go in wrong direction. • Individually dependency is very high since documentation is gathered. • Technology transfer to new team is difficult due to minimum documentation.

G. Rapid Application Model

The RAD (Rapid Application Development) model depends upon prototyping and iterative development, no need for specific planning. The process of writing the software itself involves the planning required for developing the product. A prototype is a working model which is functionally equal to a component of the product.

The phases of RAD are:

1) Business Model

The business model for the product under development is designed in terms of flow of information and the distribution of information between various business channels. A complete business analysis is performed to find the vital information for business, how it can be obtained, how and when is the information processed and what are the factors driving successful flow of information.

2) Data Modelling

The information gathered in the Business Modelling phase is reviewed and analysed to form sets of data objects vital for the business. The attributes of all data sets is identified and defined. The relation between these data objects are established and defined in detail in relevance to the business model.

3) Process Modelling

The data object sets defined in the Data Modelling phase is converted to establish the business information flow needed to achieve specific business objectives as per the business model. The process model for any changes or enhancements to the data object sets is defined in this phase. Process descriptions for adding, deleting, retrieving or modifying a data object are given.

4) Application Generation

The actual system is built and coding is done by using automation tools to convert process and data models into actual prototypes.

5) Testing and Turn Over

The overall testing time is reduced in RAD model as the prototypes are independently tested during every iteration. However the data flow and the interfaces between all the components need to be thoroughly tested with complete test coverage. Since most of the programming components have already been tested, it reduces the risk of any major issues.
The diagram of RAD model is given below:

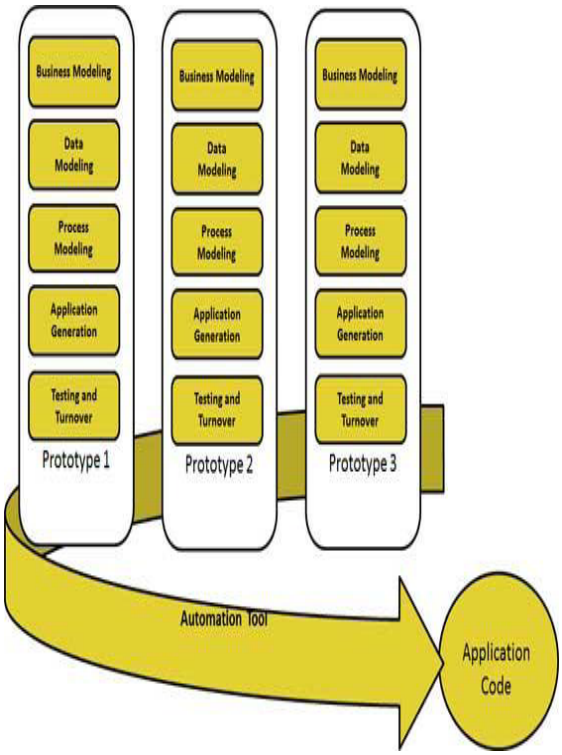

Fig 7

6) Rad Model Application

Following are the typical scenarios where RAD can be used:

- RAD should be used only when a system can be modularized to be delivered in incremental manner.
- It should be used if there's high availability of designers for modeling
- It should be used only if the budget permits use of automated code generating tools.
- RAD SDLC model should be chosen only if domain experts are available with relevant business knowledge
- Should be used where the requirements change during the course of the project and working prototypes are to be presented to customer in small iterations of 2-3 months.

7) *Advantages and Disadvantages of RAD Model*

The following table list out the advantages and disadvantages of RAD model:

Table 7

Advantages	**Disadvantages**
• Changing requirements can be accommodated. • Progress can be measured. • Iteration time can be short with use of powerful RAD tools. • Productivity with fewer people in short time. • Reduced development time. • Increases reusability of components.	• Dependency on technically strong team members for identifying business requirements. • Only system that can be modularized can be built using RAD. • Requires highly skilled developers/designers. • High dependency on modeling skills. • Inapplicable to cheaper projects as cost.

H. Software Prototyping

The Software Prototyping refers to building software application prototypes which display the functionality of the product under development but may not actually hold the exact logic of the original software. The phases of software prototype are:

1) Basic Requirement Identifications

This step involves understanding the very basics product requirements especially in terms of user interface. The more intricate details of the internal design and external aspects like performance and security can be ignored at this stage.

2) Developing the Initial Prototype

The initial Prototype is developed in this stage, where the very basic requirements are showcased and user interfaces are provided. These features may not exactly work in the same manner internally in the actual software developed and the workarounds are used to give the same look and feel to the customer in the prototype developed.

3) Review of the Prototype

The prototype developed is then presented to the customer and the other important stakeholders in the project. The

feedback is collected in an organized manner and used for further enhancements in the product under development.

*4) **Revise and Enhance the Prototype***

The feedback and the review comments are discussed during this stage and some negotiations happen with the customer based on factors like , time and budget constraints and technical feasibility of actual implementation. The changes accepted are again incorporated in the new Prototype developed and the cycle repeats until customer expectations are met.

Software Prototyping Types

There are different types of software prototypes used in the industry. Following are the major software prototyping types used widely:

*1) **Rapid Prototyping***

This type of prototyping uses very little efforts with minimum requirement analysis to build a prototype. Once the actual requirements are understood, the prototype is discarded and the actual system is developed with a much clear understanding of user requirements [6].

*2) **Evolutionary Prototyping***

Evolutionary prototyping also called as breadboard prototyping is based on building actual functional prototypes with minimal functionality in the beginning. The prototype developed forms the heart of the future prototypes on top of which the entire system is built. Using evolutionary prototyping only well understood requirements are included in the prototype and the requirements are added as and when they are understood [6].

*3) **Incremental Prototyping***

Incremental prototyping refers to building multiple functional prototypes of the various sub systems and then integrating all the available prototypes to form a complete system [6].

*4) **Extreme Prototyping***

Extreme prototyping is used in the web development domain. It consists of three sequential phases. First, a basic prototype with all the existing pages is presented in the html format. Then the data processing is simulated using a prototype services layer. Finally the services are implemented and integrated to the final prototype. This process is called Extreme Prototyping used to draw attention to the second phase of the process, where a fully functional UI is developed with very little regard to the actual services.

Software Prototyping Application

The situations where the best use of software prototyping are:

- High level of user interactions.
- Application like online system.
- Where user go through many screens.
- Applications where users need to fill forms.

Advantages and Disadvantages of Software Prototyping

The following table list out the advantages and disadvantages of Software Prototyping:

Table 8

Advantages	**Disadvantages**
• Increased user involvement in the product even before implementation. • Since a working model of the system is displayed, the users get a better understanding of the system being developed. • Reduces time and cost as the defects can be detected much earlier. • Quicker user feedback is available leading to better solutions. • Missing functionality can be identified easily. • Confusing or difficult functions can be identified.	• Risk of insufficient requirement analysis owing to too much dependency on prototype. • Users may get confused in the prototypes and actual systems. • Practically, this methodology may increase the complexity of the system as scope of the system may expand beyond original plans. • Developers may try to reuse the existing prototypes to build the actual system, even when it's not technically feasible. • The effort invested in building prototypes may be too much if not monitored properly.

III. CONCLUSION

This paper is about the different SDLC model and scenarios or situation in which these SDLC are best used. It can help project manager that which model is best for their projects. Also help developers, teachers, student and any other person which are interested in this topic. I have tried almost all popular SDLC models which are used in the software industry. I explained their advantages and disadvantages. Waterfall and V-Model is traditional and sequential model. Sequential means that the next phase will start only after the completion of first phase. Such models are suitable for projects with very clear product requirements and where the requirements will not change dynamically during the course of project completion. Iterative and Spiral models are more accommodative in terms of change and are suitable for projects where the requirements are not so well defined, or the market requirements change quite frequently. Big Bang model is a random approach to Software development and is suitable for small or academic projects. RAD (Rapid Application

Development) and Software Prototype are modern techniques to understand the requirements in a better way early in the project cycle. These techniques work on the concept of providing a working model to the customer and stockholders to give the look and feel and collect the feedback. This feedback is used in an organized manner to improve the product.

REFERENCES

[1] S. Ian Somerville, Software Engineering, 9th ed.

[2] Silvia T. ACUÑA, Xavier FERRÉ. Software Process Modeling.

[3] Giuseppe Santucci. Software Development Process Models.

[4] Nabil Mohammad Ali Munassar, A Govardhan. A comparison between five Models of Software Engineering. IJCSI International Journal of Computer Science Issues, Vol. 7, Issue 5, September 2010.

[5] S.Thulasee Krishna, Dr. S.Sreekanth, K.Perumal, K.Rajesh Kumar Reddy. Explore 10 Different Types of Software Development Process Models. International Journal of Computer Science and Information Technologies, Vol. 3 (4) , 2012,4580 - 4584 .

[6] Walt Scacchi, Institute for Software Research, University of alifornia, Irvine February 2001. Final Version to appear in, J.J. Marciniak (ed.), *Encyclopedia of Software Engineering, 2nd Edition*, John Wiley and Sons, Inc, New York, December 2001.

[7] Mary Shaw. Writing Good Software Engineering Research Papers**.** Proceedings of the 25th International Conference on Software Engineering, IEEE Computer Society, 2003, pp. 726-736.

[8] A Survey of System Development Process Models. Models for Action Project: Developing Practical Approaches to Electronic Records Management and Preservation. Center for Technology in Government University at Albany / SUNY.

[9] Ms. Shikha maheshwari, Prof.Dinesh Ch. Jain, A Comparative Analysis of Different types of Models in Software Development Life Cycle . International Journal of Advanced Research in Computer Science and Software Engineering ,Volume 2, Issue 5, May 2012.

