A Comparative study of Various Models in Software Development Life Cycle

Ms. Shweta Adlak
MCNUJC, Bhopal

Mr. Praveen Kumar Singh
G.S. College, Jabalpur

Mr. Amit Gupta
Extol College, Bhopal

Abstract – Software Development Life Cycle (SDLC) is also known Software Process model, is structured and systematic approach to develop software product. The various software Model are available in market, all are used in different situations. The process involves in one model used in one organization may or may not be followed by in another organization. This paper covers SDLC seven phases and comparison between eight models. Each model has some advantages, disadvantages and Preferred Application Area. We compare theses model on factors like requirements specification, understanding requirements, user involvement, budget, risk analysis, reusability, understandability etc.

Key word: SDLC (Software Development Life cycle), Development Models, Comparative Analysis of Models, RAD (Rapid application development).

Introduction –

Software development life cycle (SDLC) is used to design, develop & produce high quality, reliable, cost effective software in specified time as per the need of customer. Quality of product is maintained using SDLC. SDLC is a collection of various steps which followed for the systematic development design & maintenance of the software and ensure that all users’ requirement is fulfilled with least amount of resource consumption [1].
Phases of SDLC:-

(i) **Planning** - Planning is the first and most important 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.

(ii) **Analysis** – In the requirement analysis phase of SDLC (Software Development Life Cycle) where discussion with client about his needs regarding software development. The aim of this phase to grab out all the details of the project or we can say that requirement analysis phase is to capture the detail of each requirement and to make sure everyone understands the scope of the work and how each requirement is going to be fulfilled.

(iii) **Design** - 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 [2].

(iv) **Development** - 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.

(v) **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 [3].
(vi) **Implementation** - On receiving system design documents, the work is divided in modules. 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 [3].

(vii) **Maintenance** - If some problems are faced in the customer’s environment then solves these problems in maintenance phase. Also some enhancement can do in this phase if user is not fully satisfied [4].

**SDLC MODELS:**

This paper represents a systemic review on following models.

A. Waterfall Model
B. V- Model
C. Incremental Model
D. Spiral model
E. Rapid Application Development Model (RAD)
F. Prototyping Model
G. Big Bang Model
H. Agile Model

A. **Waterfall model** - The Waterfall Model was first Process Model is derived from more general system engineering process by Royce,1970[13]. introduced. It is also referred to as a linear-sequential life cycle model. It is very simple to understand and use. In a waterfall model, each phase must be completed fully before the next phase can begin. This type of software development model is basically used for the for the project which is small and there are no uncertain requirements. At the end of each phase, a review takes place to determine if the project is on the right path and whether or not to continue or discard the project. In this model software testing starts only after the development is complete. In **waterfall model phases** do not overlap [5].

![Waterfall Model](image-url)
Advantage :-

(i) It is simple and easy to understand and use.
(ii) Each phase has a specific deliverable and review process. [6]
(iii) Phases are processed and completed one at a time.
(iv) Waterfall model works well for smaller projects where requirements are very well understood [5].

Disadvantage :-

(i) High amounts of risk and uncertainty.
(ii) Iteration problem (it is very difficult to go back from one phase to another phase.
(iii) A little flexibility and adjusting scope is difficult and expensive [7].
(iv) Late delivery of the software
(v) High amounts of risk and ambiguity.
(vi) Not feasible for complex and object oriented projects.
(vii) Not fit for long and ongoing projects.
(viii) Not good where high risk chances occur and the requirements are moderate.

B. V- Model: -The V-Model [12] is an extension to the Waterfall Model in that it does not follow a sequential mode of execution rather it bends upward after the coding phase to form V shape. In this model testing functionality is associated with each phase this is also called verification & validation model. In this model, we cannot move to next phase until the previous phase is not completed.

Advantages :-

(i) This highly disciplined model and phases are completed one at a time.
(ii) This is suitable for tiny and small projects where requirements are well understood.
(iii) Simple and easy to use.
(iv) Proactive defect tracking i.e. defects are found at early stage.
(v) Easy to manage because each phase provide a deliverable with review process.
Disadvantages:

(i) Rigid and least flexible.
(ii) Not suitable where requirement are not fixed.
(iii) Working software is only available at the end of life cycle.
(iv) Not suitable for bigger and complex projects.
(v) Risk and uncertainty is high.

C. Incremental model:- Developing systems through incremental release requires first providing essential operating functions, then providing system users with improved and more capable versions of a system at regular intervals (Basili 1975)[16]. In this model, each module passes through the requirements, design, implementation and testing phases. A working version of software is produced during the first module, so you have working software early on during the software life cycle. Each subsequent release of the module adds function to the previous release. The process continues till the complete system is achieved [5].

![Incremental Model Diagram]

Figure 4: Incremental Model

Advantage:

(i) Lowers initial delivery cost.
(ii) In this model customer can respond to each built.
(iii) This model is less costly compared to others
(iv) It is flexible and less expensive to change requirements and scope
(v) Errors are easy to be identified[8]
Disadvantage :-

(i) Needs good planning and design.
(ii) Each iteration phase is rigid and does not overlap each other.
(iii) The complete cost of this model is higher than waterfall model.

D. Spiral model:- The spiral model was proposed by Boehm in 1988,[14] is recommended where the requirements and solution call for developing full- fledged, large, complex system with a lot of features and facilities from scratch. It is used when experimenting on technology, trying out new skill and when the user not able to offer requirement in clear terms. It combines characteristics of both prototype and waterfall process model. The model is divided into some task regions, which are as follows: Customer Communication, Planning, Risk Analysis, and Engineering, Construction and release and Customer evaluation. The distinctive feature of this model is that each stage is controlled by a specific risk management criteria ensuring decision making using critical factors.

![Spiral Model Diagram](image)

**Figure 5: Spiral Model**

Advantage :-

(i) Users see the system early.
(ii) Changing requirements can be accommodated.
(iii) Allows for extensive use of prototypes
(iv) Suitable for medium and large sizes of products.

Disadvantage :-

(i) Process is complex.
(ii) Complex management.
(iii) Spiral can go indefinitely.
(iv) Large number of phases requires heavy documentation.
(v) Project completion duration is not known.
(vi) Not suitable for low risky and small projects.

E. **Rapid application development model (RAD):** - The RAD model is proposed when requirements and solution can be modularized as independent system or software components, each of which can be developed by different teams [15][18]. After these smaller system components are developed, they are integrated to produce the larger software system solution. 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 [9].

![Image](image_url)

**Figure 6: Incremental Model**

**Advantage:**

(i) Reduced development time.
(ii) Increases reusability of components
(iii) Encourages customer feedback
(iv) Progress can be measured
(v) Productivity with fewer people in short time
(vi) Changing requirements can be accommodated

**Disadvantage:**

(i) High dependency on modeling skills
(ii) Requires highly skilled developers/designers.
(iii) Only system that can be modularized can be built using RAD
(iv) Inapplicable to cheaper project as cost

F. **Prototyping model:** - The basic idea here is that instead of freezing the requirements before a design or coding can proceed, a throwaway prototype is built to understand the requirements. Prototyping is a technique for providing a reduced functionality or a limited performance version of a software system early in its development (Balzer 1983, Budde 1984, Hekmatpour 1987)[17]. This prototype is developed based on the currently known requirements. By using this prototype, the client can get an
“actual feel” of the system, since the interactions with prototype can enable the client to better understand the requirements of the desired system.

![Figure 7: Prototyping Model](image)

**Advantage :-**

(i) Missing functionality can be identified easily.
(ii) Confusing or difficult functions can be identified.
(iii) Users are actively involved in the development.
(iv) Errors can be detected much earlier.
(v) Increased user involvement in the product even before implementation.

**Disadvantage :-**

(i) After seeing an early prototype end users demand the actual system to be delivered.
(ii) Poor documentation.
(iii) Users may get confused in the prototypes and actual systems.
(iv) Too many changes can disturb the rhythm of the development team.

**G. Big Bang Model**:- This model is a SDLC model where there is no particular process and development methods followed and customer also not sure about his requirement. The development process starts with only money and efforts as input and output is the developed software, which may or may not be as per customer requirement. The entire effort is spent on software development and coding.

![Figure 8: Prototyping Model](image)
Advantages:
(I) Very little or no planning required.
(II) It provides flexible approach to developers.
(III) Management is very easy.
(IV) Least resources are required.
(V) Ideal for small projects.

Disadvantages:
(I) Highly risky.
(II) Not suitable for complex and object oriented model.
(III) Not suitable for large projects.
(IV) Very expensive if requirements are misunderstood.
(V) Completion date is not given.

H. Agile model: The agile model is hybrid model it is uses advantages of the both iterative and incremental model by dividing software product breaking a product into apparatus where on each cycle or iteration, a working model of a component is delivered. This model delivers updated releases and each Release contains some incremental updates and after completion of each iteration product is tested to ensure that the iteration is acceptable or not. The Agile model emphasizes association, as the clients, developers and testers effort mutually all through the project. A benefit of the Agile model is that it rapidly deliver an operational product and is measured a very practical development approach [11].

Figure 9: Agile Model
Advantage :-

(i) Early delivery.
(ii) Good model for changing environment.
(iii) This is a realistic approach to develop software project.
(iv) Face-to-face conversation is the best form of communication.
(v) Working software is delivered frequently
(vi) In agile process mostly meeting arranged before product release
(vii) Easy manageable.
(viii) Little planning required.

Disadvantage :-

(i) In case of some software deliverables, especially the large ones, it is difficult to assess the effort required at the beginning of the software development life cycle.
(ii) Documentation could get lengthy.
(iii) It is not useful for small projects.
(iv) Not appropriate for complex dependencies.
(v) Technology transfer to new team is difficult due to minimum documentation.

Table 1 Comparison between various type software development models

<table>
<thead>
<tr>
<th>Factor</th>
<th>Waterfall</th>
<th>V- Model</th>
<th>Incremental</th>
<th>Prototype</th>
<th>RAD</th>
<th>Spiral</th>
<th>Big Bang</th>
<th>Agile</th>
</tr>
</thead>
<tbody>
<tr>
<td>Requirement specification</td>
<td>beginning</td>
<td>beginning</td>
<td>beginning</td>
<td>Frequently changed beginning</td>
<td>beginning</td>
<td>Frequently changed beginning</td>
<td>Frequently changed</td>
<td></td>
</tr>
<tr>
<td>Understanding Requirement</td>
<td>Well understand</td>
<td>Well understand</td>
<td>Not well understand</td>
<td>Not well understand</td>
<td>Well understand</td>
<td>Understood</td>
<td>Not well understand</td>
<td>Understand</td>
</tr>
<tr>
<td>Project Size</td>
<td>Large</td>
<td>Small, medium</td>
<td>small</td>
<td>Large</td>
<td>small</td>
<td>Large</td>
<td>small</td>
<td>small</td>
</tr>
<tr>
<td>User Involvement</td>
<td>Only beginning</td>
<td>Only beginning</td>
<td>Intermediate</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Budget</td>
<td>Low</td>
<td>High</td>
<td>Low</td>
<td>High</td>
<td>Very high</td>
<td>High</td>
<td>Very High</td>
<td>Very High</td>
</tr>
<tr>
<td>Risk analysis</td>
<td>Only at beginning</td>
<td>Only at beginning</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Reusability</td>
<td>Limited</td>
<td>Limited</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Limited</td>
</tr>
<tr>
<td>Understandability</td>
<td>Simple</td>
<td>Simple</td>
<td>Intermediate</td>
<td>Intermediate</td>
<td>Intermedia te</td>
<td>Complex</td>
<td>Simple</td>
<td>Much Complex</td>
</tr>
<tr>
<td>Overlapping</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>Change incorporated</td>
<td>Difficult</td>
<td>Difficult</td>
<td>Easy</td>
<td>Easy</td>
<td>Easy</td>
<td>Easy</td>
<td>Easy</td>
<td>Easy</td>
</tr>
</tbody>
</table>

Table -2 SDLC Model and Preferred Application Area

<table>
<thead>
<tr>
<th>SDLC Model</th>
<th>Preferred Application Area</th>
</tr>
</thead>
<tbody>
<tr>
<td>Waterfall model</td>
<td>Large Mainframe based or Transaction Oriented Batch system</td>
</tr>
<tr>
<td>V- model</td>
<td>Large Mainframe based or Transaction Oriented Batch system(Medical Development)</td>
</tr>
<tr>
<td>Incremental model</td>
<td>Web Information System and Event Driven System</td>
</tr>
<tr>
<td>Spiral model</td>
<td>Real Time and Safety Critical Systems</td>
</tr>
<tr>
<td>RAD model</td>
<td>Quick small and medium Business Application</td>
</tr>
<tr>
<td>Prototyping model</td>
<td>Online system and Decision Support System</td>
</tr>
<tr>
<td>Big bang model</td>
<td>useful for very small academic or practice project</td>
</tr>
</tbody>
</table>
Conclusion- In this study we discussed a variety of software development process models. Each model has its unique feature and specialty. Waterfall model is one of its kind and its main focus on sequential process which does not overlap. It is a simple and easy model to understand. Second model we discussed V- model is an expansion of waterfall model, with each phase of V-model, a testing phase is associated. Easy to manage each phase. Next model we discussed in this paper is Incremental model which based on iterative prototype model suitable for small and medium size projects. It is flexible than water fall model. Spiral Model is combination of systematic and structured development. In this Model continuous changes can be adopted easily.RAD Model does not need specific or particular planning. This model easily accommodates users changing requirements and reduces the development time. Prototyping Model is useful for project which is involved in using new technology. Prototype provides user’s an idea about final product. Big bang model is not follow specific process and development method of SDLC. The Last model we discussed in our study is Agile model, which mostly used in these days. This Model is a concentrate on rapid development of software product and customer satisfaction. All These software models works on providing working model to the customer and stakeholders for their software development.

References: -


