

**INDIAN INSTITUTE OF INFORMATION TECHNOLOGY**

**VADODARA**

**IT-301 SOFTWARE ENGINEERING**

Team IT - 05

Mentor : Dr. Asim Banerjee

**Web Development**

SDLC Models for Software Project

Version 1.0

**Version History:**

|  |  |  |  |
| --- | --- | --- | --- |
| Version | Date | Author | Reviewer |
| v1.0 | **27-09-2017** | **Niranjan Choudhary**  **Anshit Kumar** | **Manikanta, vamsi** |

**Team Members:**

|  |  |
| --- | --- |
| Name | E-mail |
| Akriti Bhadoriya | [201552074@iiitvadodara.ac.in](mailto:201552074@iiitvadodara.ac.in) |
| Anshit Kumar | [201552056@iiitvadodara.ac.in](mailto:201552074@iiitvadodara.ac.in) |
| Niranjan Choudhary | [201552060@iiitvadodara.ac.in](mailto:201552074@iiitvadodara.ac.in) |
| Chetanya Shrimali | [201552064@iiitvadodara.ac.in](mailto:201552074@iiitvadodara.ac.in) |
| J Manikanta Swamy | [201552068@iiitvadodara.ac.in](mailto:201552074@iiitvadodara.ac.in) |
| Vamshi Krishna | [201552079@iiitvadodara.ac.in](mailto:201552074@iiitvadodara.ac.in) |

Table of Contents

[Purpose](#_oryi5n42utfr)

[Stage 2: Requirements Gathering and Analysis](#_lae6g8a65k24)

[Stage 3: Designing](#_eyhv76kkysnx)

[Stage 4: Developing (coding) and Unit Testing](#_mthmaa695702)

[Stage 5: Testing](#_71kd8lnlbsll)

[Stage 6: Deployment and Maintenance](#_ken63zuhx27t)

# 

# **Purpose**

The purpose of making this document is to understand SDLC (Software Development Life Cycle). Thereafter choosing which model will be best for our project.

**SDLC – Overview**

It is abbreviated as Software Development Life Cycle. SDLC is a framework which defines different tasks to be performed at different step in the software development process. The SDLC aims to produce a high-quality software that meets client expectations. And gets completed within given time and cost.

A typical Software Development Life Cycle consists of the following stages :–

**Stage 1: Planning**

The Planning phase is the most crucial step in creating a successful software. During this phase you decide exactly what you want to do and the problems you’re trying to solve, by:

* Defining the problems, the objectives and the resources such as time and costs.
* The ability of proposing alternative solutions after meeting with clients.
* Studying how to make your product better than the client’s competitors.

After analyzing this data you will have three choices: develop a new system, improve the current system or leave the system as it is.

In our project once we agreed on doing the project of developing a website for Green Apple Restaurant, we met with our client and discussed the problem. We analyze the problem and then proposed the solution, the alternatives and new things that they can do in that. Our client told us to develop a new system. We then started working on next stage.

### **Stage 2: Requirements Gathering and Analysis**

The client’s requirements should be determined and documented, what their expectations are for the system, and how it will perform i.e. making a proposal for client. A feasibility study will be made for the project as well, involving determining whether it’s organizationally, economically, socially, technologically feasible i.e. making feasibility study. Then developing an SRS (Software Requirement Specification) document which consists of all the product requirements to be designed and developed during the project life cycle. It’s very important to maintain strong communication level with the clients to make sure you have a clear vision of the finished product and its function.

### **Stage 3: Designing**

How will we get what we want?

This phase of SDLC starts by turning the software specifications into a design plan called the Design Specification. All stakeholders then review this plan and offer feedback and suggestions. It’s crucial to have a plan for collecting and incorporating stakeholder input into this document. Failure at this stage will almost certainly result in cost overruns at best and total collapse of the project at worst.

### 

### **Stage 4: Developing (coding) and Unit Testing**

Let’s create what we want.

This SDLC stage develops the software by generating all the actual code. In particular, when a programmer, web developer, database developer are brought on to do the major work on the project. If the design is performed in a detailed and organized manner, code generation can be accomplished without much hassle. Developers must follow the coding guidelines defined by their organization and programming tools like compilers, interpreters, debuggers, etc. are used to generate the code. The programming language is chosen with respect to the type of software being developed.

Unit Testing is done by developer on a regular basis.Unit testing is a form of coding that breaks your software down into specific functions then tests each individually for any logic flaws. The goal is to find bugs before your customers find them.

### 

### **Stage 5: Testing**

Did we get what we want?

In this stage, we test for defects and deficiencies. Bringing different components and subsystems together to create the whole integrated system, and then Introducing the system to different inputs to obtain and analyze its outputs and behavior and the way it functions. Testing is becoming more and more important to ensure customer’s satisfaction, and it requires no knowledge in coding, hardware configuration or design.

Testing can be performed by real users, or by a team of specialized personnel, it can also be systematic and automated to ensure that the actual outcomes are compared and equal to the predicted and desired outcomes.

### **Stage 6: Deployment and Maintenance**

Let’s start using what we got and get closer to what we want.

This stage involves the actual installation of the newly-developed system. Once the product is tested and ready to be deployed it is released formally in the appropriate market. This part of the SDLC process happens in a limited way at first. Depending on feedback from end users, more adjustments can be made.

The plan almost never turns out perfect when it meets reality. Further, as conditions in the real world change, we need to update and advance the software to match. After the product is released in the market, its maintenance is done for the existing customer base.

**1. ITERATIVE WATERFALL MODEL**

**Introduction:-**

* Iterative Waterfall Model carries less risk than a traditional Waterfall approach.

Meaning If you have done some mistake in earlier stage of the project you can

go back and correct it.

* 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.
* Major risk are taken in the initial iteration itself,so that at the end of project the risk is minimal.
* In iterative model we are building and improving the product step by step. Hence we can track the defects at early stages. This avoids the downward flow of the defects.
* Works well for smaller projects where the team know the requirements of the client.
* We can make changes in previous phase according to our need.
* It supports redesign, acceptance and review of any new requirement or any different mistake.



**Why We are Choosing Iterative Waterfall Model :-**

* As we are not expert and have not done many other project earlier, Mistakes can happen. So rather choosing Waterfall Model we chose iterative waterfall Model so that we can revert back and make changes.
* Our Project is a small scale project therefore Iterative Waterfall Model is appropriate for it because it best supports small scale projects.
* As we understood our Project Requirement and is very well Clear, Iterative Waterfall Model is best for it.
* Also we as a team is not large in number and our project is also simple and client-based,Iterative Waterfall Model is good for it.
* As our group has a introductory knowledge of Web Development,Some of them have a good hold on it. Therefore this model will help us to get on track if we stumble on something.

**2. WATERFALL MODEL:**

**Introduction:-**

* The waterfall model is simplest and oldest model, which states that the phases are organized in a linear order and straightforward.
* Waterfall model is step by step phase dependent model there the large and

complex project is broken into smaller tasks of specifying requirements. Separating the concerns and focusing on a select few in a phase gives better handle to the engineers and managers in dealing with the complexity of the problem .

* In this model, a project begins with feasibility analysis. Upon successfully demonstrating the feasibility of a project, the requirements analysis and project planning begins. The design starts after the requirements analysis is complete, and coding begins after the design is complete. Once the programming is completed, the code is integrated and testing is done. Upon successful completion of testing, the system is installed
* Waterfall model is approach when one step completely finished then further proceed to other step

**Why We are rejecting Waterfall Model :-**

* In waterfall model when we are in design phase but we have change in requirement phase, demand on client it not possible to change in this model.
* In this model we have to well exprience and good knowledge of technology, otherwise high risks and uncertainty
* In finding error take long time until testing phase

**3. Spiral Model :-**

**Introduction**

* This Spiral model is a combination of iterative development process model and sequential linear development model i.e. the waterfall model with a very high emphasis on risk analysis.
* The spiral model has four phases. A software project repeatedly passes through these phases in iterations called **Spirals**.
  + Identification
  + Design
  + Construct or Build
  + Evaluation and Risk Analysis
* Based on the customer evaluation, the software development process enters the next iteration.
* The process of iterations along the spiral continues throughout the life of the software.

**Why We are rejecting Spiral Model :-**

* The models work best for large projects only, where the costs involved are much higher and system prerequisites involves higher level of complexity. As our project is smaller & simpler and have a fixed time period, we cannot use this model.
* End of the project may not be known early. But we have a fixed time period.
* It is not suitable for low risk projects. Our’s is simpler project which have less risk.
* Large number of intermediate stages requires excessive documentation.More number of document and iteration will increase the complexity of our project,which we don’t have to do.

**4. PROTOTYPE MODEL:**

**Introduction:**

* Prototype model which displays the functionality of the product under development, but may not actually hold the exact logic of the original software.
* Prototyping is an attractive idea for complicated and large systems for which

there is no manual process or existing system to help determine requirements.

* When client is not confident about the developer’s capabilities, he asks for a small prototype to be built. Based on this prototype model, he can judge capabilities of developer.
* Stepwise approach to design a software prototype.
* *Basic Requirement Identification*
* *Developing the initial Prototype*
* *Review of the Prototype*
* *Revise and Enhance the Prototype*

**Why We are rejecting Prototype Model :-**

* Users may get confused in the prototypes and actual systems.Here we are developing a software for a client. So we cannot take this risk.
* It is a slow process. As we have have fixed time, we cannot choose a model which is slow.
* The effort invested in building prototypes may be too much if it is not monitored properly. We have fixed time and people working on project, we cannot have that much workforce to complete it on time. So we cannot take this model.

**5.INCREMENTAL MODEL:**

**Introduction:**

* Incremental development is based on the idea of developing an initial implementation, exposing this to user comment and evolving it through several versions until an adequate system has been developed
* Each increment or version of the system incorporates some of the functionality

that is needed by the customer

* It is easier to get customer feedback on the development work that has been

done. Customers can comment on demonstrations of the software. Customers find it difficult to judge progress from software design documents.

* More rapid delivery and deployment of useful software to the customer is possible, even if all of the functionality has not been included.

**Why We are rejecting Incremental Model :-**

* The problems of incremental development is where different teams develop different parts of the system. Large systems need a stable framework or architecture and the responsibilities of the different teams working on parts of the system need to be clearly defined with respect to that architecture.
* For our project we have to do all functionality need to finished on deployment time
* Rectifying a problem in one unit requires correction in all the units and consume a lot of time.

**6. AGILE MODEL:**

**Introduction:**

* Agile development model is also a type of Incremental model. Software is developed in incremental, rapid cycles. This results in small incremental releases with each release building on previous functionality
* People and interactions are emphasized rather than process and tools. Customers, developers and testers constantly interact with each other.
* Regular adaptation to changing circumstances.
* Even late changes in requirements are welcomed

**Why We are rejecting Agile Model :-**

* Depends heavily on customer interaction, so if customer is not clear, team can be driven in the wrong direction.
* There is a very high individual dependency, since there is minimum documentation generated.
* Transfer of technology to new team members may be quite challenging due to lack of documentation.