**Software Development Life Cycle (SDLC)**

**Definition - What does *Software Development Life Cycle (SDLC)* mean?**

The software development life cycle (SDLC) is a framework defining tasks performed at each step in the software development process. SDLC is a structure followed by a development team within the software organization. It consists of a detailed plan describing how to develop, maintain and replace specific software. The life cycle defines a methodology for improving the quality of software and the overall development process.

The software development life cycle is also known as the software development process.

***Software Development Life Cycle (SDLC)***

SDLC consists of following activities:

1. Planning: The most important parts of software development, requirement gathering or requirement analysis are usually done by the most skilled and experienced software engineers in the organization. After the requirements are gathered from the client, a scope document is created in which the scope of the project is determined and documented.
2. Implementation: The software engineers start writing the code according to the client's requirements.
3. Testing: This is the process of finding defects or bugs in the created software.
4. Documentation: Every step in the project is documented for future reference and for the improvement of the software in the development process. The design documentation may include writing the application programming interface (API).
5. Deployment and maintenance: The software is deployed after it has been approved for release.
6. Maintaining: Software maintenance is done for future reference. Software improvement and new requirements (change requests) can take longer than the time needed to create the initial development of the software.

There are several software development models followed by various organizations:

* Waterfall Model: This model involves finishing each phase completely before commencing the next one. When each phase is completed successfully, it is reviewed to see if the project is on track and whether it is feasible to continue.
* V-Shaped Model: This model focuses on the execution of processes in a sequential manner, similar to the waterfall model but with more importance placed on testing. Testing procedures are written even before the commencement of writing code. A system plan is generated before starting the development phase.
* Incremental Model: This life cycle model involves multiple development cycles. The cycles are divided up into smaller iterations. These iterations can be easily managed and go through a set of phases including requirements, design, implementation and testing. A working version of the software is produced during the first iteration, so working software is created early in the development process.

Software Development Life Cycle Models and Methodologies

# Introduction

Software development life cycle (**SDLC**) is a series of [phases](http://melsatar.blog/2017/06/13/what-do-you-need-to-know-about-the-eight-software-development-phases/) that provide a common understanding of the software building process. How the software will be realized and developed from the business understanding and requirements elicitation phase to convert these business ideas and requirements into functions and features until its usage and operation to achieve the business needs. The good software engineer should have enough knowledge on how to choose the SDLC model based on the project context and the business requirements.

Therefore, it may be required to choose the right SDLC model according to the specific concerns and requirements of the project to ensure its success. I wrote another article on how to choose the right SDLC, you can follow this [link](http://melsatar.blog/2012/03/21/choosing-the-right-software-development-life-cycle-model/) for more information. Moreover, to learn more about [Software Testing life cycles](http://melsatar.blog/2018/04/02/software-testing-fundamentals/) and [SDLC phases](http://melsatar.blog/2017/06/13/what-do-you-need-to-know-about-the-eight-software-development-phases/) you follow the links highlighted here.

In this article, we will explore the different types of SDLC models and the advantages and disadvantages of each one and when to use them.

# Types of Software developing life cycles (SDLC)

* [Waterfall Model](http://melsatar.blog/2018/02/16/the-waterfall-model-a-different-perspective/)
* V-Shaped Model
* [Evolutionary Prototyping](http://en.wikipedia.org/wiki/Software_prototyping) Model
* [Spiral](http://en.wikipedia.org/wiki/Spiral_model) Method ([SDM](http://en.wikipedia.org/wiki/Software_development_methodology))
* [Iterative and Incremental](http://en.wikipedia.org/wiki/Iterative_and_incremental_development) Method
* [Agile development](http://en.wikipedia.org/wiki/Agile_software_development)

## Waterfall Model

### Description

The [Waterfall Model](http://melsatar.blog/2018/02/16/the-waterfall-model-a-different-perspective/) is a linear sequential flow. In which progress is seen as flowing steadily downwards (like a waterfall) through the phases of software implementation. This means that any phase in the development process begins only if the previous phase is complete. The waterfall approach does not define the process to go back to the previous phase to handle changes in requirement. The waterfall approach is the earliest approach and most widely known that was used for software development.

### [Waterfall](https://i1.wp.com/melsatar.blog/wp-content/uploads/2012/03/waterfall.jpg)The usage

Projects which not focus on changing the requirements, for example, projects initiated from a request for proposals ([RFPs](http://en.wikipedia.org/wiki/Request_for_proposal)), the customer has a very clear documented requirements

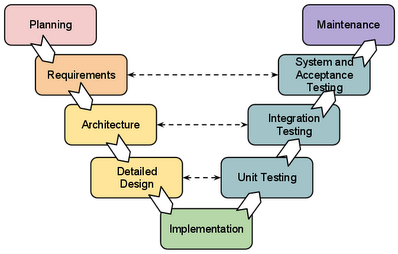
### Advantages and Disadvantages

|  |  |
| --- | --- |
| Advantages | Disadvantages |
| * Easy to explain to the users. * Structures approach. * Stages and activities are well defined. * Helps to plan and schedule the project. * Verification at each stage ensures early detection of errors/misunderstanding. * Each phase has specific deliverables. | * Assumes that the requirements of a system can be frozen. * Very difficult to go back to any stage after it finished. * A little flexibility and adjusting scope is difficult and expensive. * Costly and required more time, in addition to the detailed plan. |

## V-Shaped Model

### Description

It is an extension of the waterfall model, Instead of moving down in a linear way, the process steps are bent upwards after the implementation and coding phase, to form the typical V shape. The major difference between V-shaped model and waterfall model is the early test planning in the V-shaped model.



### The usage

* Software requirements clearly defined and known
* Software development technologies and tools are well-known

### Advantages and Disadvantages

|  |  |
| --- | --- |
| Advantages | Disadvantages |
| * Simple and easy to use * Each phase has specific deliverables. * Higher chance of success over the waterfall model due to the development of test plans early on during the life cycle. * Works well for where requirements are easily understood. * Verification and validation of the product in early stages of product development. | * Very inflexible, like the waterfall model. * Adjusting scope is difficult and expensive. * The software is developed during the implementation phase, so no early prototypes of the software are produced. * The model doesn’t provide a clear path for problems found during testing phases. * Costly and required more time, in addition to a detailed plan |

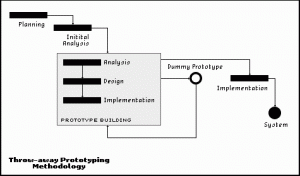
## Prototyping Model

### Description

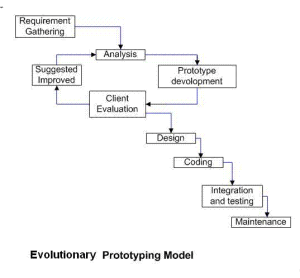
It refers to the activity of creating prototypes of software applications, for example, incomplete versions of the software program being developed. It is an activity that can occur in software development and It used to visualize some component of the software to limit the gap of misunderstanding the customer requirements by the development team. This also will reduce the iterations may occur in waterfall approach and hard to be implemented due to the inflexibility of the waterfall approach. So, when the final prototype is developed, the requirement is considered to be frozen.

It has some types, such as:

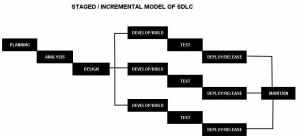
* Throwaway prototyping: Prototypes that are eventually discarded rather than becoming a part of the finally delivered software

[](https://i2.wp.com/melsatar.blog/wp-content/uploads/2012/03/throwaway-prototyping.gif)

* Evolutionary prototyping: prototypes that evolve into the final system through an iterative incorporation of user feedback.

[](https://i0.wp.com/melsatar.blog/wp-content/uploads/2012/03/ev-proto.png)

* Incremental prototyping: The final product is built as separate prototypes. In the end, the separate prototypes are merged in an overall design.

[](https://i1.wp.com/melsatar.blog/wp-content/uploads/2012/03/stagedmodelofsdlc.jpg)

* Extreme prototyping: used in web applications mainly. Basically, it breaks down web development into three phases, each one based on the preceding one. The first phase is a static prototype that consists mainly of HTML pages. In the second phase, the screens are programmed and fully functional using a simulated services layer. In the third phase, the services are implemented

### The usage

* This process can be used with any software developing life cycle model. While this shall be chosen when you are developing a system has user interactions. So, if the system does not have user interactions, such as a system does some calculations shall not have prototypes.

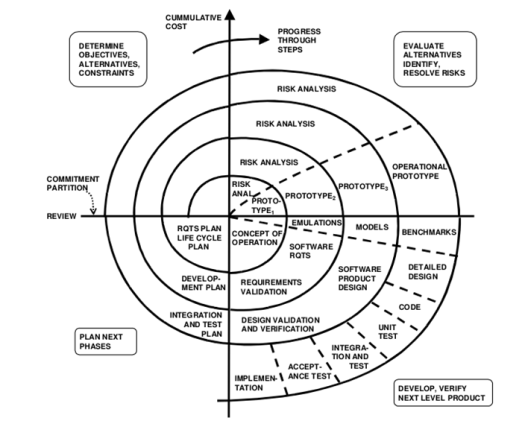
### Advantages and Disadvantages

|  |  |
| --- | --- |
| Advantages | Disadvantages |
| * Reduced time and costs, but this can be a disadvantage if the developer loses time in developing the prototypes. * Improved and increased user involvement. | * Insufficient analysis. User confusion of prototype and finished system. * Developer misunderstanding of user objectives. * Excessive development time of the prototype. * It is costly to implement the prototypes |

## Spiral Model (SDM)

### Description

It is combining elements of both design and prototyping-in-stages, in an effort to combine advantages of top-down and bottom-up concepts. This model of development combines the features of the prototyping model and the waterfall model. The spiral model is favored for large, expensive, and complicated projects. This model uses many of the same phases as the waterfall model, in essentially the same order, separated by planning, risk assessment, and the building of prototypes and simulations.



### The usage

It is used in the large applications and systems which built-in small phases or segments.

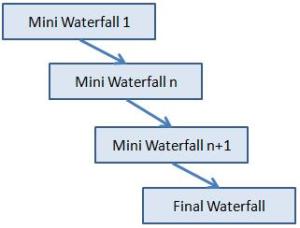
### Advantages and Disadvantages

|  |  |
| --- | --- |
| Advantages | Disadvantages |
| * Estimates (i.e. budget, schedule, etc.) become more realistic as work progressed because important issues are discovered earlier. * Early involvement of developers. * Manages risks and develops the system into phases. | * High cost and time to reach the final product. * Needs special skills to evaluate the risks and assumptions. * Highly customized limiting re-usability |

## Iterative and Incremental Model

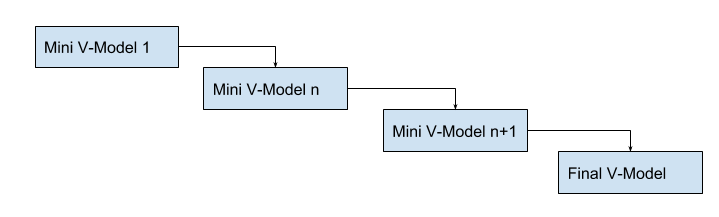
### Description

It is developed to overcome the weaknesses of the waterfall model. It starts with an initial planning and ends with deployment with the cyclic interactions in between. The basic idea behind this method is to develop a system through repeated cycles (iterative) and in smaller portions at a time (incremental), allowing software developers to take advantage of what was learned during the development of earlier parts or versions of the system.

It can consist of mini waterfalls or mini V-Shaped model[](https://i0.wp.com/melsatar.blog/wp-content/uploads/2012/03/incremental-sdlc.jpg)

### The usage

It is used in shrink-wrap application and large system which built-in small phases or segments. Also, can be used in a system has separated components, for example, ERP system. Which we can start with the budget module as a first iteration and then we can start with inventory module and so forth.



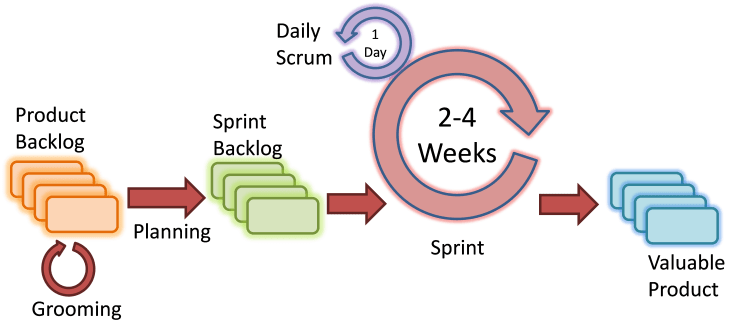
### Advantages and Disadvantages

|  |  |
| --- | --- |
| Advantages | Disadvantages |
| * Produces business value early in the development lifecycle. * Better use of scarce resources through proper increment definition. * Can accommodate some change requests between increments. * More focused on customer value than the linear approaches. * We can detect project issues and changes earlier. | * Requires heavy documentation. * Follows a defined set of processes. * Defines increments based on function and feature dependencies. * Requires more customer involvement than the linear approaches. * Partitioning the functions and features might be problematic. * Integration between iteration can be an issue if this is not considered during the development. |

## Agile Model

### Description

It is based on iterative and incremental development, where requirements and solutions evolve through collaboration between cross-functional teams.



Scrum Agile Model

### The usage

It can be used with any type of the project, but it needs more engagement from the customer and to be interactive. Also, it can be used when the customer needs to have some functional requirement ready in less than three weeks and the requirements are not clear enough.

### Advantages and Disadvantages

|  |  |
| --- | --- |
| Advantages | Disadvantages |
| * Decrease the time required to avail some system features. * Face to face communication and continuous inputs from customer representative leaves no space for guesswork. * The end result is the high-quality software in the least possible time duration and satisfied customer. | * Scalability. * The ability and collaboration of the customer to express user needs. * Documentation is done at later stages. * Reduce the usability of components. * Needs special skills for the team. |