**Program**: Set of instructions we are giving to system.

**Software**: group of programs that are developed together whatever be the language.

What is SDLC? Anything which is developed will follow certain life cycle.

Life Cycle means it’s the process of going through set of phases. Ex. Constructing a house coming up with plan, finalized the plan, implement the plan , finally house is ready, after few years will destroy it and start again the same process…so everyone / everything has a cycle.

SDLC : Software Development Life Cycle…set of phases that will continue in a cycle to complete the process within the software organization.

A typical Software Development life cycle consists of the following stages:

1. **Requirement Gathering /Planning** – business manager / project manager or owner

2. **Analysis** – product owner / BA

3. **Design** – architect / ui designer

4. **Implementation/Develop/coding** – developers—frontEnd/backend/application developers and DBA team

5. **Testing** – test engineers and team

6. **Maintenance** --

**Design:**

**Use case diagrams**: Uniform defines about actors and their responsibilities or functionality according to actors.

**Activity Diagrams**: like flow charts. Every steps are broken down into flow depending upon the execution continues.

**Sequence diagrams**: depicted in form of a sequence diagram…step by step sequence.

**Class diagrams**: defining variables and methods/properties and action for given behavior. Are very technical.

**Wireframes:** pictorial representation of the requirements, position of the elements. For mobile testing we get the requirements documents are called wireframes.

SDLC models can be broadly classified as below.  
  
1. Sequential models  
a.    Waterfall model  
b.    V-Model

 2. Incremental/Iterative models  
a.    Prototype model  
b.    Spiral Model  
c.    Agile Model

[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:

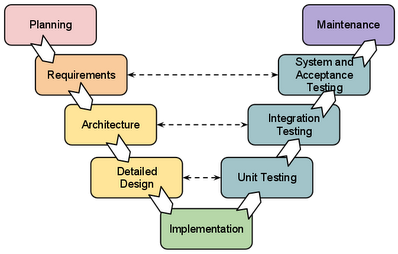
The waterfall Model is a linear sequential flow. In which progress is seen as flowing steadily downwards (like a waterfall) through the phases of software implementation. This means that any phase in the development process begins only if the previous phase is complete. The waterfall approach does not 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://melsatar.files.wordpress.com/2012/03/waterfall.jpg)

### 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.



**Waterfall model:** in this model the process flow goes from top to down, step by step. Next phase of the project can only be started once the first phase is completed. The output of the first phase is the input for the next phase.

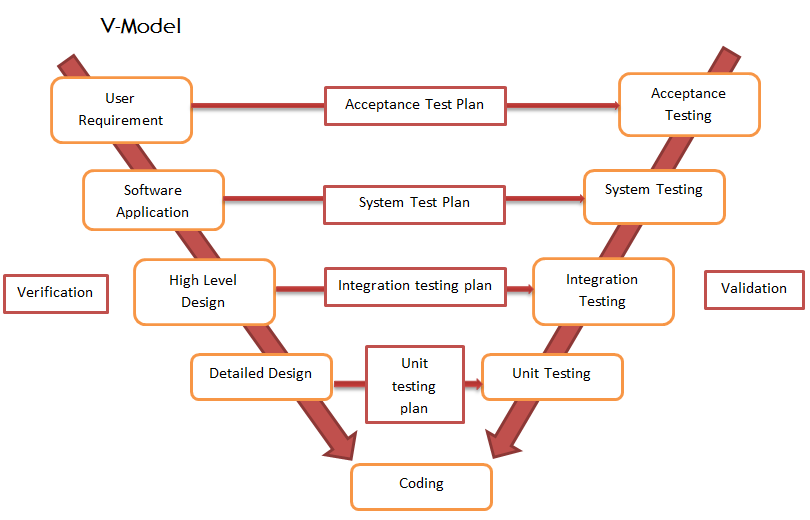
Advantages of waterfall model  
  
i.    The main strength of the waterfall model is its simplicity.  
ii.    Easy to manage due to the rigidity of the model. Each phase has specific deliverables and a review process.  
iii.    Phases are processed and completed one at a time.  
iv.    Works well for smaller projects where requirements are very well understood.

**Adv**: the model is simple and easy to understand

1. Requirements are well documented , clear and fixed.
2. .    Easy to manage due to the rigidity of the model. Each phase has specific deliverables and a review process.
3. Works well for the smaller projects where requirements are very well understood.
4. Phases are processed and completed at a time

**Dis Adv**:   
i.    Once an application is in the testing stage, it is very difficult to go back and change something that was not well-thought out in the concept stage.  
ii.    No working software is produced until late during the life cycle.  
iii.    The sequential nature of communication among the phases can introduce inordinate delays in resolving the problems.  
iv.    High amounts of risk and uncertainty.  
v.    Poor model for long and ongoing projects.  
vi.    Not suitable for the projects where requirements are at a moderate to high risk of changing.  
  
When to use waterfall model  
  
i.    Requirements are very well known, clear and fixed.  
ii.    Product definition is stable.  
iii.    Technology is understood.  
iv.    There are no ambiguous requirements  
v.    Ample resources with required expertise are available freely  
vi.    The project is short.

V-Model: V-model means Verification and Validation model. V-model may be considered as extension of Waterfall model, because just like Waterfall model it is a well structured    model in which the different phases progress in a sequential or linear way.   
In the Waterfall model, testing is considered as a post-development (after coding) activity. In V-Model, the corresponding testing phase of the development phase is planned in parallel. So, there are Verification phases on one side of the ‘V’ and Validation phases on the other side.



V- Model application is almost the same as the waterfall model, as both the models are of sequential type. Requirements have to be very clear before the project starts, because it is usually expensive to go back and make changes.

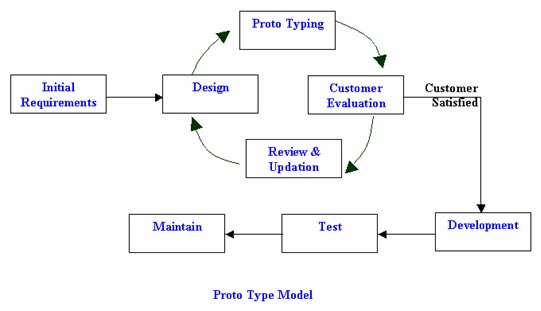
The advantages :

* This is a highly-disciplined model and Phases are completed one at a time.
* Works well for smaller projects where requirements are very well understood.
* Simple and easy to understand and use.
* Easy to manage due to the rigidity of the model. Each phase has specific deliverables and a review process.

The disadvantages :

* High risk and uncertainty.
* Not a good model for complex and object-oriented projects.
* Poor model for long and ongoing projects.
* Not suitable for the projects where requirements are at a moderate to high risk of changing.
* Once an application is in the testing stage, it is difficult to go back and change a functionality.
* No working software is produced until late during the life cycle.

**Prototype Model:** The Prototyping Model is a systems development method (SDM) in which a [prototype](http://searchcio-midmarket.techtarget.com/definition/prototype) (an early approximation of a final system or product) is built, tested, and then reworked as necessary until an acceptable prototype is finally achieved from which the complete system or product can now be developed. This model works best in scenarios where not all of the project requirements are known in detail ahead of time. It is an iterative, trial-and-error process that takes place between the developers and the users.



Adv & Dis Adv:

Improved and increased user involvement.

Insufficient analysis· User confusion of prototype and finished system.

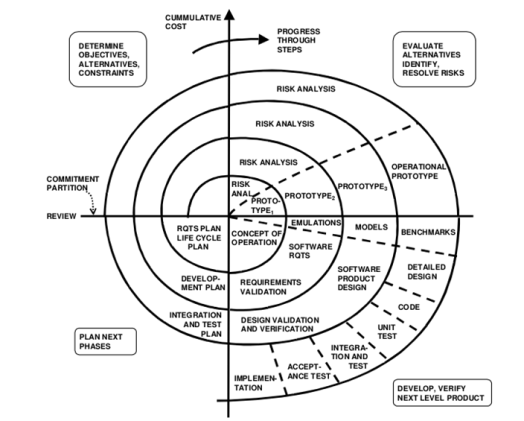
Developer misunderstanding of user objectives.

Excessive development time of the prototype.

Expense of implementing prototyping

## Spiral Model (SDM):

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.

Agile SDLC Model: is a combination of iterative and incremental models focusing on the process adaptability and customer satisfaction by rapid delivery of working software product(updating the status of the project every time).

Agile method breaks the projects into small incremental builds. These builds are provided in iterations.( **Iteration** is the act of repeating a process, either to generate an unbounded sequence of outcomes, or with the aim of approaching a desired goal, target or result. Each repetition of the process is also called an "iteration", and the results of one iteration are used as the starting point for the next iteration.)

Each iteration lasts for one or three weeks of duration. Every iteration involves teams working simultaneously working on various area like planning, requirement analysis, design, coding, unit testing, acceptance testing.

At the end of the each iteration working project/product is displayed to the customer and important stake holders.

What is Agile: Agile is a principle or methodology. In agile the tasks are divided into small time frames to deliver specific feature for a release. Iterative approach is taken and working software build is released after each iteration. Each iteration is incremental in terms of features, the final build hold all the features required by the customer.



The advantages of the Agile Model are as follows −

* Is a very realistic approach to software development.
* Promotes teamwork and cross training.
* Functionality can be developed rapidly and demonstrated.
* Resource requirements are minimum.
* Suitable for fixed or changing requirements
* Delivers early partial working solutions.
* Good model for environments that change steadily.
* Minimal rules, documentation easily employed.
* Enables concurrent development and delivery within an overall planned context.
* Little or no planning required.
* Easy to manage.
* Gives flexibility to developers.

The disadvantages of the Agile Model are as follows −

* Not suitable for handling complex dependencies.
* More risk of sustainability, maintainability and extensibility.
* An overall plan, an agile leader and agile PM practice is a must without which it will not work.
* Strict delivery management dictates the scope, functionality to be delivered, and adjustments to meet the deadlines.
* 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.

Agile methods support a broad range of the [software development life cycle](https://en.wikipedia.org/wiki/Software_development_life_cycle).[[36]](https://en.wikipedia.org/wiki/Agile_software_development#cite_note-Abrahamsson2002-36) Some focus on the practices (e.g., XP, pragmatic programming, agile modeling), while some focus on managing the flow of work (e.g., Scrum, Kanban). Some support activities for requirements specification and development (e.g., FDD), while some seek to cover the full development life cycle (e.g., DSDM, [RUP](https://en.wikipedia.org/wiki/Rational_Unified_Process)).

Different terms used in Agile:

**Scrum**: is an iterative and incremental agile software development framework for managing the product development. It defines the strategy where a development team works as a unit to reach a common goal.

**Scrum Master**: The Scrum Master is not a traditional [team lead](https://en.wikipedia.org/wiki/Team_leader) or [project manager](https://en.wikipedia.org/wiki/Project_manager), but acts as a buffer between the team and any distracting influences. The Scrum Master ensures that the Scrum framework is followed. The Scrum Master helps to ensure the team follows the agreed processes in the Scrum framework, often facilitates key sessions, and encourages the team to improve.

Scrum master assigns user stories to scrum team and monitor team and coordinates

**Scrum Team** : team consists of developers, testers,

**Sprint**: a sprint is a set period of time during which the specific works has to be completed and made ready for the review. Each sprint begins with a planning meeting. The duration of the sprint is determined by the scrum master. Normally each sprints lasts for 3 to 4 weeks or 30days. At the end of the Sprint, team submits the completed work to the project owner and project owner can take the decision to accept or reject the work during the planning meeting. Each iteration is divided into approx. 3 to 4 weeks duration which are called as Sprints.

Backlogs: Whole set of requirements finalized by the product owner. The Product Backlog is a prioritized list of everything that might be included in a product. The Product Owner creates, maintains, and regularly re-orders the Product Backlog. The Product Owner uses the Product Backlog to adapt to emerging requirements, customer feedback, and market changes.

Sprint Backlogs: sprint requirements chosen from the product backlog which are required for that sprint.

User Stories: A user story is a very high-level definition of a requirement, containing just enough information so that the developers can produce a reasonable estimate of the effort to implement it

**User stories are individual stories, divided to team members**.it tells to create a simplified description of a requirement