Basic Concept of Software Engineering

Sofware Engineering is a discipline in which methods, theories and tools are applied to develop a professional software

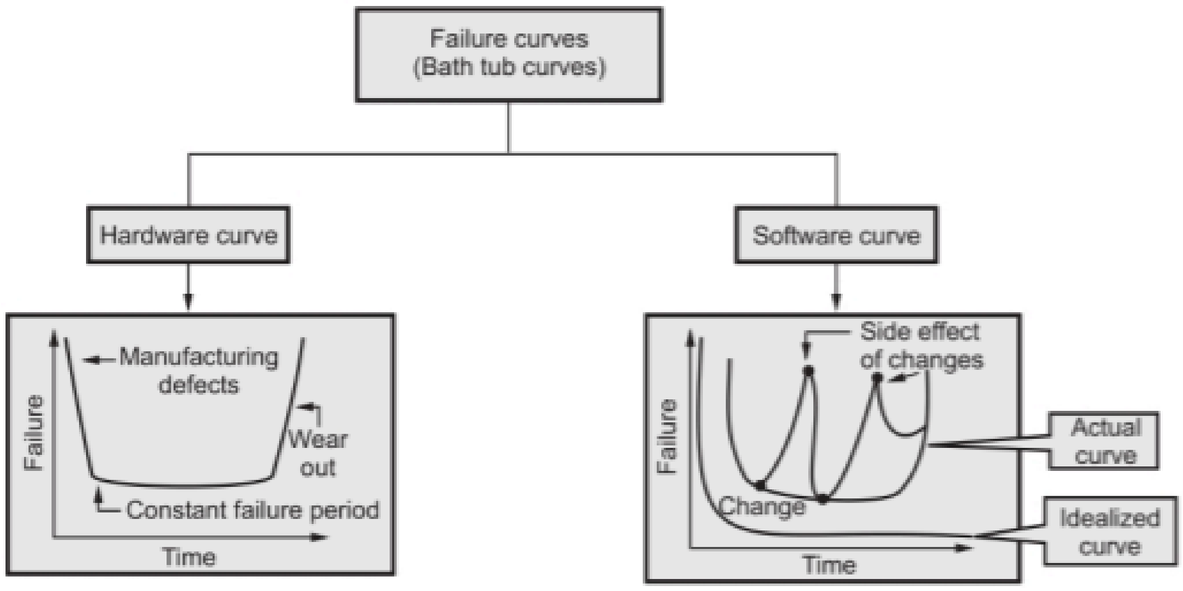
Software is a collection of computer programs and related documents that are intended to provide desired features, functionalities and better performance. Software products may be :

1. Generic - That means developed to be sold to a range of different customers.
2. Custom - That means developed for a single customer according to their specification.

Characteristics of Software

1) **Software is engineered, not manufactured**: Software development and hardware development are two different activities. A good design is a backbone for both the activities. Quality problems that occur in hardware manufacturing phase can not be removed easily. On the other hand, during software development process such problems can be rectified. In both the activities, developers are responsible for producing qualitative products.

2) **Software does not wear out:** In the early stages of the hardware development process, the failure rate is very high because of manufacturing defects. But after correcting such defects the failure rate gets reduced. The failure rate remains constant for some period of time and again it starts increasing because of environmental maladies (extreme temperature, dusts, and vibrations).
On the other hand software does not get affected by such environmental maladies. Hence ideally it should have an "idealized curve". But due to some undiscovered errors the failure rate is high and drops down as soon as the errors get corrected.
Hence in failure rating of software the "actual curve" is as shown below:



During the life of software if any change is made, some defects may get introduced.

This causes the failure rate to be high. Before the curve can return to its original steady state another change is requested and again the failure rate becomes high.
• Thus the failure curve looks like a spike. Thus frequent changes in software cause it to deteriorate.
Another issue with software is that there are **no spare parts for software.**
If a hardware component wears out it can be replaced by another component but it is not possible in case of software.
Therefore software maintenance is more difficult than hardware maintenance.

3) Most software is custom built rather than being assembled from components • While developing any hardware product firstly the circuit design with desired functioning properties is created.
Then required hardware components such as ICs, capacitors and registers are assembled according to the design, but this is not done while developing software products.
Most of the software is custom built.
However, now the software development approach is getting changed and we look for reusability of software components.
It is practised to reuse algorithms and data structures.
Today the software industry is trying to make a library of reusable components.

For example: In today's software, GUI is built using reusable components such as message windows, pull down menus and many more of such components. The approach is getting developed to use in-built components in the software. This stream of software is popularly known as ***component engineering.***

Various Categories of Software 3 Feb 2022

1) System software - It is a collection of programs written to service other programs. Typical programs in this category are compiler, editors, and assemblers. The purpose of the system software is to establish communication with the hardware.

2) Application software - It consists of standalone programs that are developed for specific business needs. This software may be supported by database systems.

3) Engineering/Scientific software: This software category has a wide range of programs from astronomy to volcanology, from automotive stress analysis to space shuttle orbital dynamics and from molecular biology to automated manufacturing. This software is based on complex numeric computations.

4) Embedded software: This category consists of programs that can reside within a product or system. Such software can be used to implement and control features and functions for the end-user and for the system itself.

5) Web applications: Web application software consists of various web pages that can be retrieved by a browser. The web pages can be developed using programming languages like JAVA, PERL, CGI, HTML, DHTML

6) Artificial Intelligence software: This kind of software is based on knowledge based expert systems. Typically, this software is useful in robotics, expert systems, image and voice recognition, artificial neural networks, theorem proving and game playing.

Goals and Objectives of Software Engineering

While developing software following are common objectives:

1. Satisfy users requirements - Many programmers simply don't do what the end user wants because they do not understand user requirements. Hence it becomes necessary to understand the demand of the end user and accordingly software should be developed.

2. High reliability - Mistakes or bugs in a program can be expensive in terms of human lives, money, and customer relation. For instance, Microsoft has faced many problems because earlier releases of Windows had many problems. Thus software should be delivered only if high reliability is achieved.

3. Low maintenance costs - Maintenance of software is an activity that can be done only after delivering the software to the customer. Any small change in software should not cause restructuring of the whole software. This indicates that the design of software has poor quality.

4. On time delivery - It is very difficult to predict the exact time on which the software can be completed. But a systematic development of software can lead to meeting the given deadline.

5. Low production costs - The software product should be cost effective.

6. High performance - The high performance software is expected to achieve optimization in speed and memory usage.

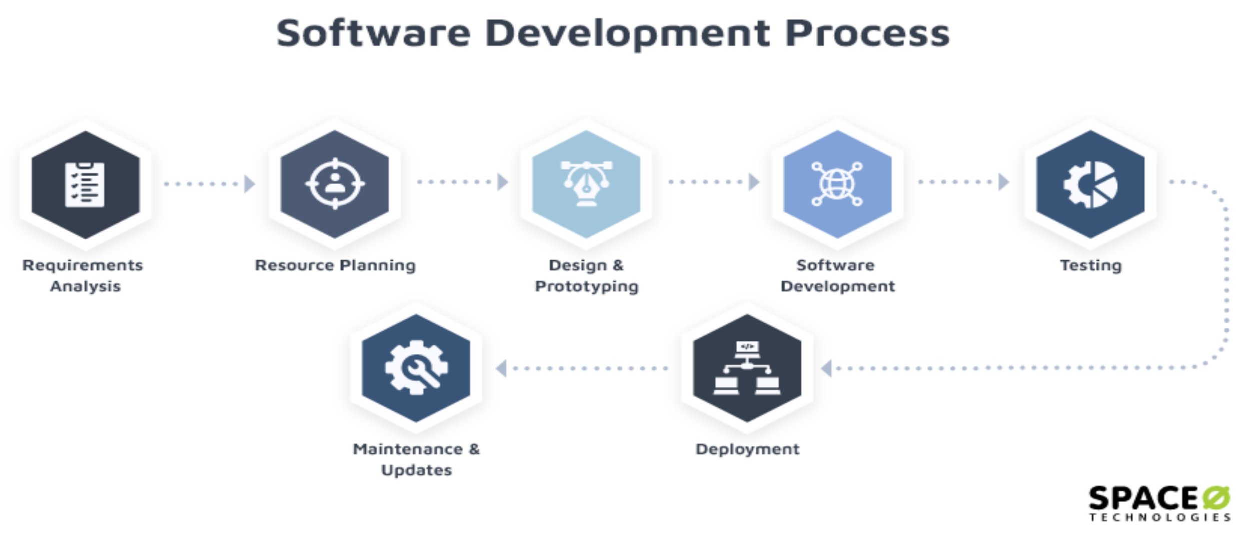
7. Ease of reuse - Use the same software in different systems and software. Environments reduce development costs and also improve reliability. Hence reusability of developed software is an important property.

Challenges in Software Engineering

The key challenges facing software engineering are:

1. <u>Coping with Legacy Systems:</u> Old, valuable system must be maintained and updated. Hardware is evolved faster than softwares. If original developer has moved on, managing, maintaining or integrating of software becomes a critical issue.
2. <u>Heterogeneity Challenge:</u> Sometimes system are distributed and includes a mix of hardware and software. This imlies that software systems must neatly integrate with other software systems built by different organisations and teams using different hardware and software
3. Delivery Time Challenge: There is increasing pressure for faster delivery of software. As the complexity of systems that we develop increases, this challenge becomes harder.

As software is an integral part of computer based systems, it is essential to apply software engineering principles and practices while developing software. The main objective of software engineering is to adopt systematic, development approach while building a high quality softwarei



What is SDLC (Software Development Life Cycle)?

Software development process is dividing the software development into tiny, sequential steps to enhance the product, project, and design altogether. The iterative logical process for software program development or application development to cater to the needs of any business or personal objectives is known as 'Software Development'.
We know that software developers use app development software to create a specific code for any software development or application development. However, coding is only a part of a big process called the software development process or life cycle. The entire SDLC consists of ideation, research, coding, documenting, testing, debugging, deployment, and updating, used by the software industry.

Stages of Software Development Process

1) Requirements Analysis and Resource Planning:
   The first step to any process is always planning. Being a project manager, you might have done a requirement analysis of your project, but you are going to need software engineering experts to create a

software development plan for your project. You need to analyse if the software you are planning to develop aligns with your business or personal goals. This is a requirements analysis.
The purpose of any software is to make the tasks easier. So, you must check which tasks you are trying to optimise and how the software will help you in this. After this, you need to allocate resources for the software development process. You need to decide what kind of resources you will need in order to complete it.

2) Design and Prototyping:
After the analysis and planning part is over, it is time to start creating a software architecture for the product. This architecture or design will define the complete workflow of the software. In terms of software, the design doesn’t only have to do about the look but also about the overall functioning and user experience of the software.
You can play an important part in the design process as you need to explain to the software designers what you want from the program. You can define how the users will interact with the software application/product. The designers will design simple wireframes to show these interactions using various tools like Adobe and InVision. If needed, you can also have complete prototypes that display each and every functionality of the product.
In this stage, you can check if there are any drawbacks or lack of any features. You can easily make changes in this stage and start with development when everything is finalised

3) Software Development:
Development in software-process only begins when you are completely sure of the requirements and onboard with the design and features. The development team starts working on the development of a program by writing the necessary code.
Development in software-process only begins when you are completely sure of the requirements and onboard with the design and features. The development team starts working on the development of a program by writing the necessary code.
Now, the development is carried out in different manners based on the type of software requirements.
This is the riskiest phase of the software development process. However, being an experienced software development company, we easily understand the requirements and develop a product up to the expectations of the customer.

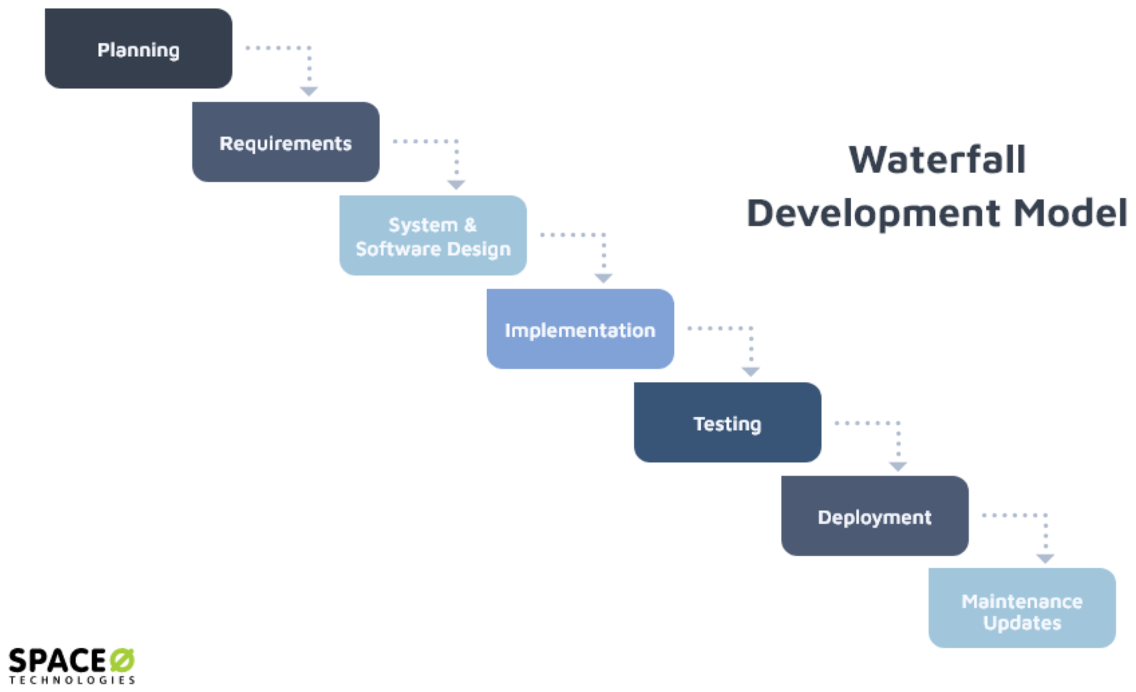
4) Testing:
   This is actually a continuous process of software development, and testing is performed alongside development. Testing is done to check the functionality, usability, and stability of the product under the rapid development process.
   We have a team of quality assurance testers or QA testers. This team tests every piece of code created by the software development team. This is done both manually as well with automated tools to find out if there are any bugs or glitches.
   Later, bugs are fixed by changing or adding new code to the original code. We make sure that your final product runs smoothly on the preferred devices and has all the required features and functionalities as discussed.
5) Deployment Phase:
   This is a crucial stage in the software development life cycle. After coding and testing are done, the next development phase is to deploy your software on the necessary servers and devices. This is only done after you have approved of the product functionality and the stability of the product is proved.
   Many times, the product is given an Alpha release. A small bunch of users use the product and give their feedback. After analysing the feedback, modifications are made to the software and then released as a Beta release. Now, more users have access to the software program.
6) Maintenance and Updating:
   Software development is a cycle. It is an iterative process of software development. After launching the product, the process is not complete. You need to keep a track of software maintenance and keep upgrading it. You need to consistently monitor software development and suggest changes whenever required.
   This is done because technology keeps advancing and in order to keep up with these changes, the software products need to be updated. As time passes, users have different requirements that are uncovered. Further, user feedback also plays an important role in devising future updates for any software product.
   Finally, we also create software that is easily scalable for future scale-ups or scale-down according to the changing trends and requirements.

Software Development Life Cycle Models

1. Waterfall Model

2. Prototyping Development Model
3. Evolutionary Model
4. Spiral Model

<u>Waterfall Model</u>



The waterfall model is also called the linear sequential model or Classic life cycle model.
It is the oldest software development paradigm. This software development model suggests a systematic, sequential approach to software development. The software development starts with the requirement gathering phase. Then progressing through analysis, design, coding, testing, maintenance and update.

<u>Phases of The Waterfall Model</u>

1. Planning
2. Requirement
3. Software System Design
4. Implementation
5. Testing
6. Product Release
7. Maintenance and Updates

1) <u>Requirement Gathering/Planning:</u> The basic requirements of the system must be understood by the software engineer who is also called analyst.

The information, domain, function, behaviour, requirements of the system are understood. All these requirements are then well documented and discussed with the customer for reviewing

2) Design: The design is an intermediate step between requirement analysis and coding. It focuses on program attributes such as:
    a) Data Structure
    b) Software architecture
    c) Interface representation
    d) Algorithmic details.

   The requirements are translated in some easy to represent form using coding can be done effectively and efficiently. The design needs to be documented for future use.
3) Coding: This is a step in which software design is translated into machine readable form. If design is done in sufficient detail, then coding can be done effectively. Programs are created in this phase.
4) Testing: Testing begins when coding is done. While performing tests, the focus is on logical internals of the software The testing ensures execution of all parts and functional requirements. The purpose of testing is to uncover errors, fix the bugs and meet the customer requirements.
5) Maintenance: It is the longest life cycle phase. When the system is installed and put in practical use, then errors may get introduced. Correcting those errors is the major purpose of maintenance activities. Similarly, enhancing system services as new requirements are discovered in the maintenance phase.

Advantages of Waterfall Model

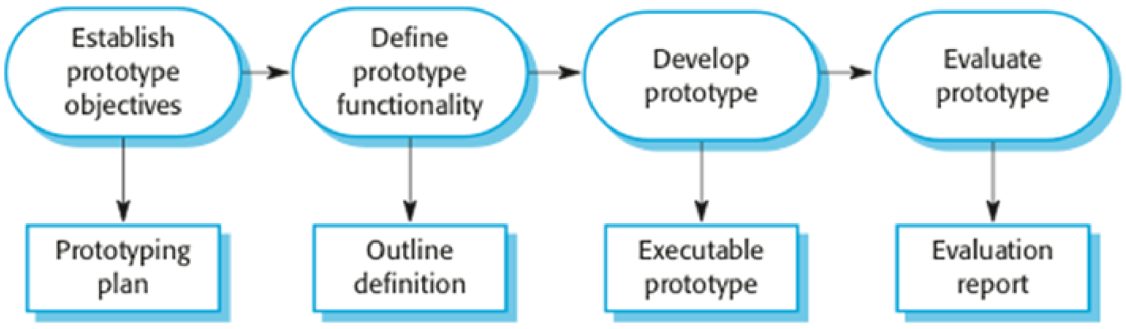
1. Waterfall model works well for smaller projects where requirements are clearly defined and very well understood
2. In this model phases are processed and completed one at a time. Phases do not overlap

Drawbacks of Waterfall Model

1. It is difficult to follow the sequential flow in the software development process. If some changes are made at some phases, it may cause some confusion.
2. The requirement analysis is done initially and sometimes it is not possible to state all the requirements explicitly in the beginning. This causes difficulty in the project.
3. The customer can see the working model of the project only at the end. After reviewing the working model and the customer is not satisfied, it causes serious problems.

4. LInear nature of the waterfall model includes a blocking state because certain tasks may be dependent on some previous task. Hence it is necessary to accomplish all the dependent tasks first. It may cause long waiting times.

Prototyping Development Model



The project team creates a small prototype of the final software for the users. This version is used to get feedback from the users and also check usability and feasibility of the design. If there are any issues that came into light while using the prototype, they are noted down and fixed. Such issues do not make it to the final software product. Such a method is used when you are not sure of what your users might like or not like. This gieves the development team a chance to perform risk analysis before putting the original product out in the market. In prototyping, you are involved throughout the development and design process. This way, the users will clearly understand the requirements of the product. Furthermore, the prototypes are meant to be discarded once they have been tested. Yet it is not uncommon when a prototype after testing and development is upgraded to the final product.

Phases of Prototyping

1. Establish prototype objective
2. Define prototype functionality
3. Develop prototype
4. Evaluate prototype

Advantages of Prototyping

1. Here, errors can be detected easily
2. You can create a better solution with user feedback

Disadvantages of Prototyping

1. The complexity of the product may increase

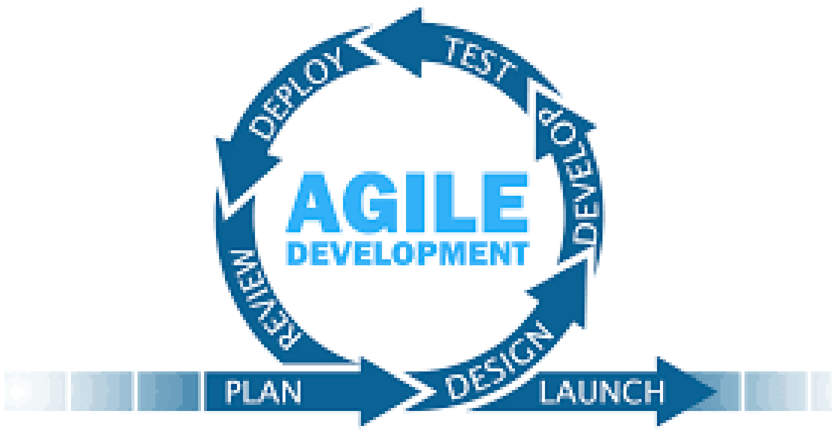
2. The process may become time consuming

Agile Software Development Model

The agile approach is the most popular software development methodology(i.e system development methodology). This is due to the fact that it is highly dynamic and iterative which leads to fewer errors in the final software product. In fact, according to a survey by Goatfilm, agile has been considered the best development model. This model is the exact opposite of the waterfall model. Agile means fast and flexible and the methodology takes on it’s name. The development team can easily make changes to the initial plans. Infact this methodology is for almost all projects. In agile development model, you do not need a complete list of requirements, you can explore and find out more as you go along. You do not have to develop all the functionalities at once, you can develop some features and check the user response before taking a step ahead. This will save you from putting too much effort that might go to waste. Agile processes follow a flexible and iterative approach in which there is constant user involvement to in order to understand their mindset, carry out a project risk assessment and bring about project improvement. Scrum, crystal, agile modelling(AM) extreme programming(EP) are some examples of agile methods. Agile development model is also known as the iterative model.

Phases of Agile Development Model

1. Requirement
2. Design
3. Development
4. Testing
5. Deployment
6. Feedback and Validation
7. Plan the next phases



Evolutionary Model

It is also called successive version model or incremental model. At first, a simple working model is built. Subsequently, it undergoes functional improvement and we keep on adding more functionalities until the desired system is built.

Applications

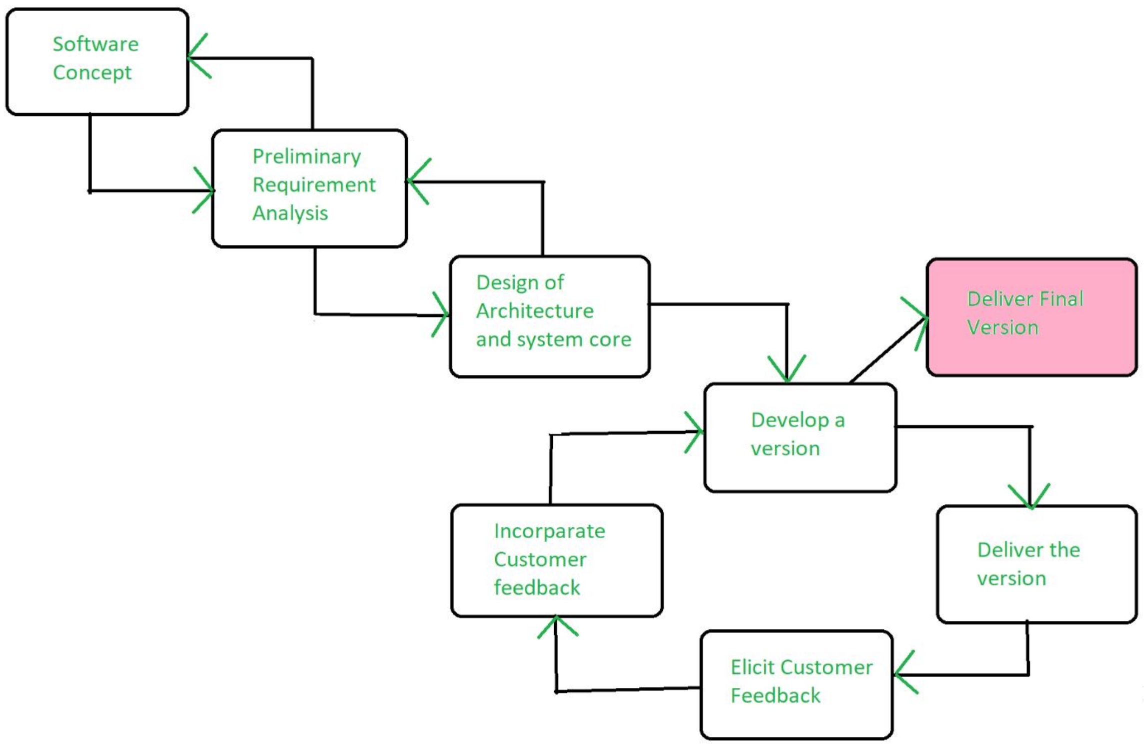
1. Large projects where you can easily find modules for incremental implementation, often used when the customer wants to start using the core features rather than waiting for the full software.
2. Also used in object oriented software development because the system can be easily portioned into units of objects.

Advantages

1. User gets a chance to experiment partially developed software
2. Reduce the errors because the core models gets tested thoroughly
3. The model is less expensive compared to others

Disadvantages

1. It is difficult to divide the problem into several versions that will be acceptable to the customer which can be incrementally implemented and delivered
2. The process can be time consuming
3. It requires planning even for future updates.


Software Concept
Preliminary Requirement Analysis
Design of Architecture and system core
Deliver Final Version
Develop a version
Incorparate Customer feedback
Deliver the version
Elicit Customer Feedback