ASSIGNMENT 1

SOFTWARE CHALLENGES AND FAILURES IN THE WORLD

SOME SOFTWARE FAILURES INTERNATIONALLY

BREAK PRISON

In 2002 a software was designed to monitor the behaviour of prisoners. The software calculates a prisoner's sentence depending on good or bad behaviour his worked very well and data were collected to monitor the behaviours of all prisoners. After 13 years of monitoring when a new IT boss was appointed and informed the governor's office of this glitch. In December 2015 a glitch occurred which led to over 3,200 US prisoners been released before their declared dates.

WELSH NHS IT FAILURE

In 2018, doctors and hospitals staffs of the Wales NHS experienced

Wide spread software failure that led them being unable to access patient files or folders. According to the National Cyber Security Centre, the failure caused the GPs not be able to access blood and x-ray results. This also caused a backlog as patients not been able to be contacted to cancel appointments and notes could not be typed up and saved NHS systems

WANNACRY

In May 2017, a large **ransom ware** attack called **WannaCry (WannaCrypt or WCry**) hits HNS England in various organisations

in UK and around the world. This was as a result of vulnerabilities found in Microsoft operating system. According to Microsoft, the Widows version of that software were vulnerable and were no longer supported by Microsoft such as Windows8 and Windows XP.

FACE BOOK, INSTAGRAM AND WHATSUP

Since Mark Zuckerberg brought WhatsApp and Instagram into his social media empire, users have increasingly had to deal with not one but three of their favourite platforms suffering from outages whenever Facebook experiences a technical issue.

In the first week of July 2019, users across the globe found themselves unable to load photos in the Facebook News Feed, view stories on Instagram, or send messages in WhatsApp. Although Facebook didn't detail the reason for the outage, it did release a statement claiming the issue had been accidentally triggered during "routine maintenance".

This isn't the first time these three platforms have experienced problems with downtime. Facebook and Instagram were disrupted for around 14 hours whilst all three of Zuckerberg's digital platforms suffered further outages again.

SOME SOFTWARE FAILURES IN GHANA

Software Failures In Ghana

OLX

A classified forum which saw a decline of interest by people with the up rise of social media advertisement. People preferred to advertise on social media as it was less expensive and they have control over it. It was eventually bought by JiJi. Over the years, OLX became a common place for fraud, especially sellers who post fake advertisements to dupe buyers upon receiving advance payment, and fraudulent buyers who engage in UPI scams, phishing and sending fake SMSs and emails of bank transfer confirmation.

Easy Taxi

Easy Taxi was one of the first ride hailing apps that folded up months after entering the market, it was new to the Ghanaian Market as most Ghanaians weren't accustomed to this service and to top it up, it was very expensive compared to the traditional taxis that were already known. They also collapsed in Nigeria around the same period of their fold up in Ghana.

OMG VOICE

OMG voice became rather popular in their early days with creative content and young staff but folded up due to unrealistic targets. Techmoran.com reported how the “buzz feed of Africa” had one of its founders resigning with lots of unpaid salaries due to the struggling nature of the company eventually leading to its collapse. Most of the brands cited above did not only operate and fail in Ghana but in other African Countries like Nigeria, South African and Kenya.

Online payment companies have not been left out either. Some of these companies that used to be very vibrant during their launch stage have been very quiet over the years. Examples of such are Slydepay and Zeepay who have been silent for some time now. With that said, it cannot be ignored that some of these online platforms have remained and continued to flourish, an example is ExpressPay.

Other similar cases were Ghana Port and Harbour Authority (GHAPHA) verses GC net, ECG verse PDS.

ASSIGNMENT 2

THE 4Ws and the 1H

Application of the first W “what”

What do the users do?

What are the objectives they have to achieve?

You have to understand their tasks all in all and also the assignments that relate to the software testing or framework that you're planning.

Application of the second W 'Why'

The task team may ask 'Why' questions to get a more granular comprehension of the problem and look to clarify triggers or drivers that may have added to the problem. A portion of the questions for software testing that can be asked are:

Application of the third W "When"

By utilizing 'When' questions, project teams can time stamp the events and understand the connections among different events that may have affected the rise of the incompatibility problem.

Application of the fourth W "Who"

Who are the users?

What are their characteristics?

What learning and experience do they convey to their tasks?

Are there any other groups of users?

Assuming this is the case, what separates them from one another, and which client bunches are generally vital? The subject of whose desires to meet dependably emerge.

Would it be those of the Developers or of the Users?

Users and engineers have their very own assumptions regarding the application and the

codes. The desires from the two sides during software testing ought to be weighted legitimately. The project team can make questions to distinguish the people involved in contributing to the particular problem. A few questions that might be inquired.

Application of the fifth W "Where"

By asking 'Where' questions, the project team can improve the handle of the source(s) of the problem. A few questions that can be inquired during this stage of software testing are.

Now Application of the H "How"

In the metric reporting of 5Ws and 1H, anything that disappoints the client is a defect, thus understanding the client and client prerequisites is the most critical problem in building up a metric reporting software testing culture. This is a critical thinking administration philosophy that can be connected to a business procedure to recognize and take out the main drivers of defects within software development, eventually enhancing the key software features and sparing expense for the association. In such a manner, the fundamental objective of metrics reporting is that any software development in an association should be monetarily suitable.

ASSIGNMENT 3

Myths of Software Engineering

Product must be perfect at the first attempt – it is an inherently a false believe. Nobody knows what '**perfec**t' looks like until people start using the product. It better to build a Minimum Variable Product (**MVP**) with the essential functionality rather than require (and wait for) the nine yards from the developers. Put the MVP on the market, start earning money, take feedback and improve the product.

Remote Developers are worse than in-house developers – some clients believe than if developers are out of sight, they're unmotivated, uncontrollable, and unaccountable. This is hardly true, having modern project management practices, excellent communication opportunities and project management systems.

ASSIGNMENT 4

SOFTAWRE ENGINEERING MODELS

SDLC (Software Development Life Cycle) Methodologies, Process, and Models and Phases

Software Development Life Cycle Process

SDLC is a process which defines the various stages involved in the development of software for delivering a high-quality product. SDLC stages cover the complete life cycle of a software i.e. from inception to retirement of the product.

Adhering to the SDLC process leads to the development of the software in a systematic and disciplined manner.

Purpose:
Purpose of SDLC is to deliver a high-quality product which is as per the customer's requirement.

SDLC has defined its phases as, Requirement gathering, Designing, Coding, Testing, and Maintenance. It is important to adhere to the phases to provide the Product in a systematic manner.

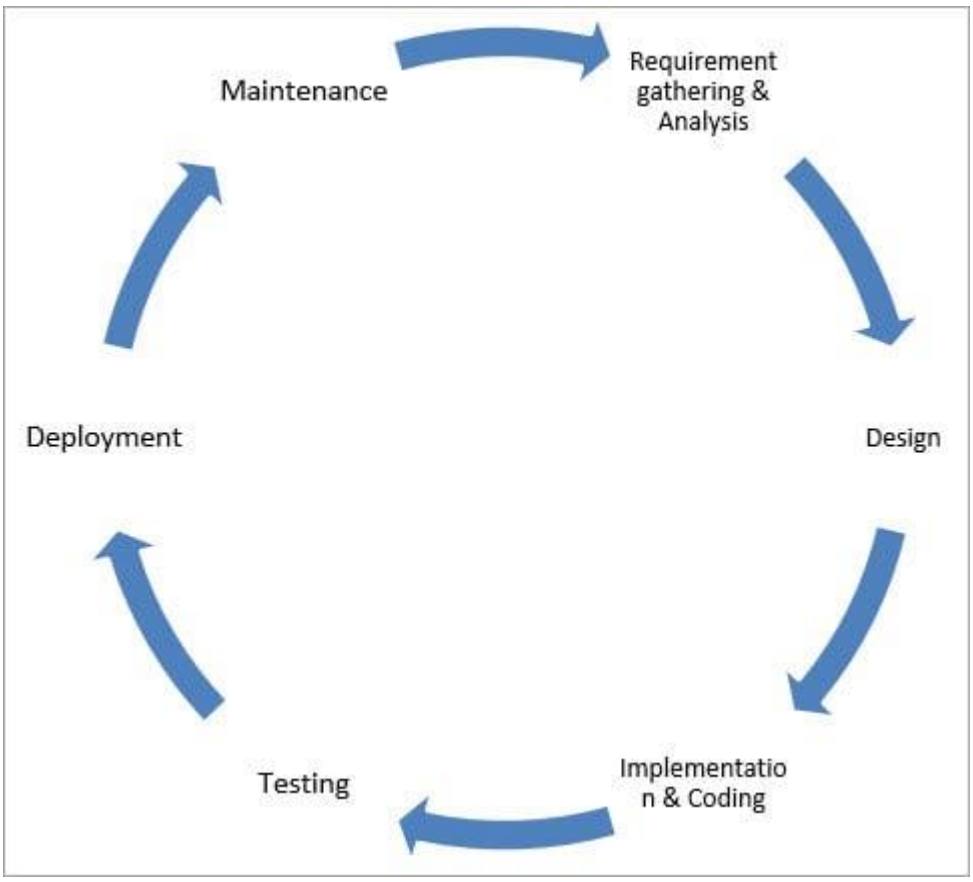
For Example,
A software has to be developed and a team is divided to work on a feature of the product and is allowed to work as they want. One of the developers decides to design first whereas the other decides to code first and the other on the documentation part.

This will lead to project failure because of which it is necessary to have a good knowledge and understanding among the team members to deliver an expected product.

SDLC Cycle

SDLC Cycle represents the process of developing software.

Below is the diagrammatic representation of the SDLC cycle:



SDLC PHASES

Given below are the various phases:

- Requirement gathering and analysis
- Design
- Implementation or coding
- Testing
- Deployment
- Maintenance

Requirement Gathering and Analysis

During this phase, all the relevant information is collected from the customer to develop a product as per their expectation. Any ambiguities must be resolved in this phase only.

Business analyst and Project Manager set up a meeting with the customer to gather all the information like what the customer wants to build, who will be the end-user, what is the purpose of the product. Before building a product a core understanding or knowledge of the product is very important.

For Example, A customer wants to have an application which involves money transactions. In this case, the requirement has to be clear like what kind of transactions will be done, how it will be done, in which currency it will be done, etc.
Once the requirement gathering is done, an analysis is done to check the feasibility of the development of a product. In case of any ambiguity, a call is set up for further discussion.

1) Requirement gathering

Once the requirement is clearly understood, the SRS (Software Requirement Specification) document is created. This document should be thoroughly understood by the developers and also should be reviewed by the customer for future reference.

2) Design

In this phase, the requirement gathered in the SRS document is used as an input and software architecture that is used for implementing system development is derived implementation of coding.

3**) Implementation or Coding**

Implementation or Coding starts once the developer gets the Design document. The Software design is translated into source code. All the components of the software are implemented in this phase.

4) Testing

Testing starts once the coding is complete and the modules are released for testing. In this phase, the developed software is tested thoroughly and any defects found are assigned to developers to get them fixed.

Retesting, regression testing is done until the point at which the software is as per the customer's expectation. Testers refer SRS document to make sure that the software is as per the customer's standard.

5) Deployment

Once the product is tested, it is deployed in the production environment UAT (User Acceptance Testing) is done depending on the customer expectation.
In the case of UAT, a replica of the production environment is created and the customer along with the developers does the testing. If the customer finds the application as expected, then sign off is provided by the customer to go live.

6) Maintenance

After the deployment of a product on the production environment, maintenance of the product i.e. if any issue comes up and needs to be fixed or any enhancement is to be done is taken care by the developers.

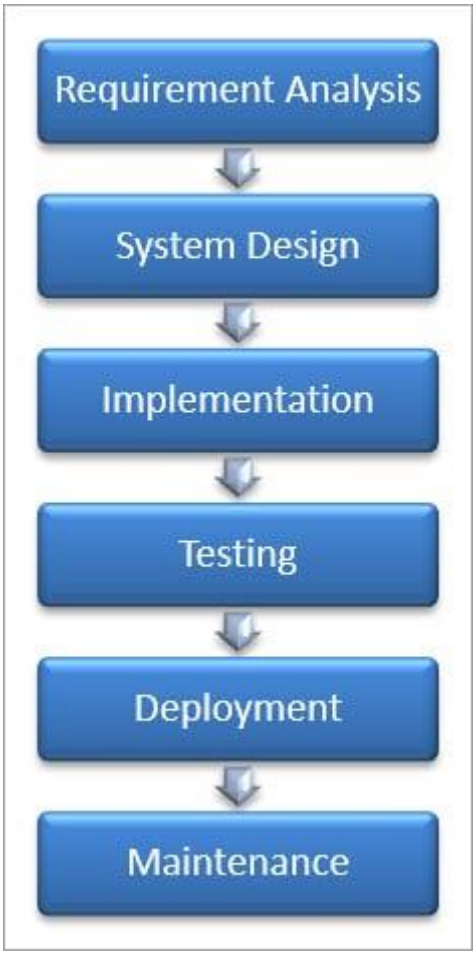
SUMMARY

Software Development Life Cycle Models

A software life cycle model is a descriptive representation of the software development cycle. SDLC models might have a different approach but the basic phases and activity remain the same for all the models.

- First, Requirement gathering and analysis is done. Once the requirement is freeze then only the System Design can start. Herein, the SRS document created is the output for the Requirement phase and it acts as an input for the System Design.

- In System Design Software architecture and Design, documents which act as an input for the next phase are created i.e. Implementation and coding.
- In the Implementation phase, coding is done and the software developed is the input for the next phase i.e. testing.
- In the testing phase, the developed code is tested thoroughly to detect the defects in the software. Defects are logged into the defect tracking tool and are retested once fixed. Bug logging, Retest, Regression testing goes on until the time the software is in go-live state.
- In the Deployment phase, the developed code is moved into production after the sign off is given by the customer.
- Any issues in the production environment are resolved by the developers which come under maintenance.



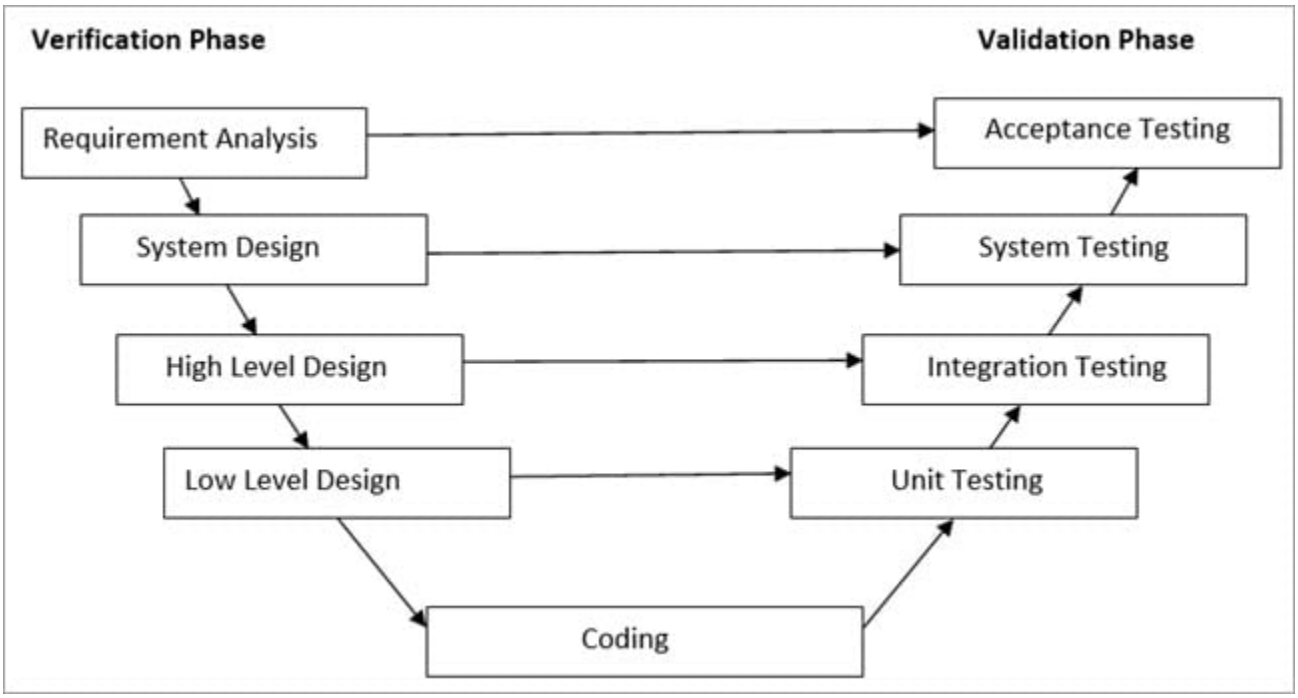
ASSIGNMENT 5

Software Engineering Models

V-SHAPE MODEL

V – SHAPE MODEL is also known as Verification and Validation Model. In this model Verification & Validation goes hand in hand i.e. development and testing goes parallel. V model and waterfall model are the same except

that the test planning and testing start at an early stage in V-Model.



a) Verification Phase:

(i)Requirement Analysis:
In this phase, all the required information is gathered & analyzed. Verification activities include reviewing the requirements.

(ii) System Design:
Once the requirement is clear, a system is designed i.e. architecture, components of the product are created and documented in a design document.

(iii) High-Level Design:
High-level design defines the architecture/design of modules. It defines the functionality between the two modules.

(iv) Low-Level Design:
Low-level Design defines the architecture/design of individual components.

(v) Coding:
Code development is done in this phase.

b) Validation Phase:

(I) Unit Testing:
Unit testing is performed using the unit test cases that are designed and is done in the Low-level design phase. Unit testing is performed by the developer itself. It is performed on individual components which lead to early defect detection.
(ii) Integration Testing:
Integration testing is performed using integration test cases in High-level Design phase. Integration testing is the testing that is done on integrated modules. It is performed by testers.
(iii) System Testing:
System testing is performed in the System Design phase. In this phase, the complete system is tested i.e. the entire system functionality is tested.
(iv) Acceptance Testing:
Acceptance testing is associated with the Requirement Analysis phase and is done in the customer's environment.

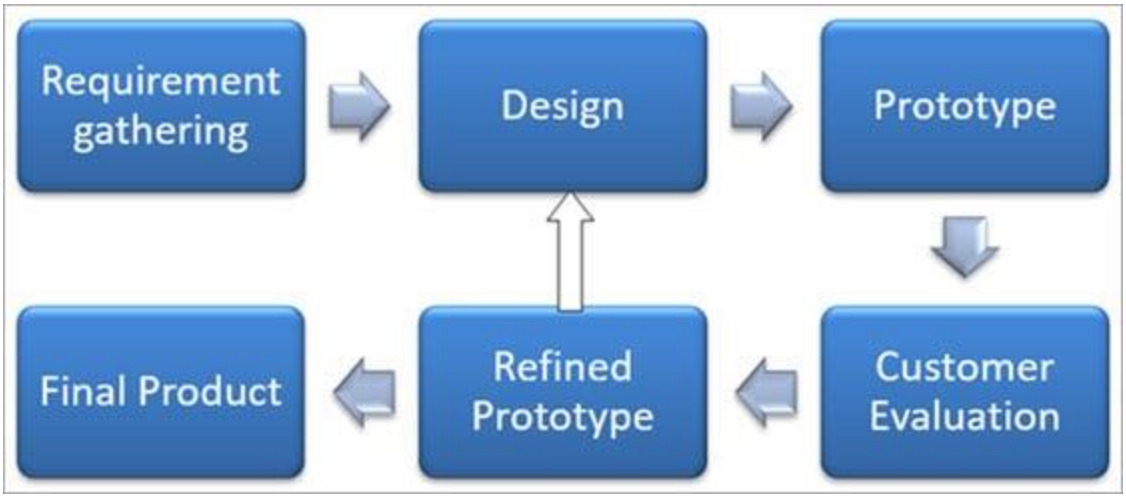
Advantages of V – Model:

- It is a simple and easily understandable model.
- V –model approach is good for smaller projects wherein the requirement is defined and it freezes in the early stage.
- It is a systematic and disciplined model which results in a high-quality product.

Disadvantages of V-Model:

V-shaped model is not good for ongoing projects.

- Requirement change at the later stage would cost too high.



Once the requirement gathering is done, the quick design is created and the prototype which is presented to the customer for evaluation is built.

Customer feedback and the refined requirement is used to modify the prototype and is again presented to the customer for evaluation. Once the customer approves the prototype, it is used as a requirement for building the actual software. The actual software is build using the Waterfall model approach.

Advantages of Prototype Model:

- Prototype model reduces the cost and time of development as the defects are found much earlier.
- Missing feature or functionality or a change in requirement can be identified in the evaluation phase and can be implemented in the refined prototype.
- Involvement of a customer from the initial stage reduces any confusion in the requirement or understanding of any functionality.

Disadvantages of Prototype Model:

- Since the customer is involved in every phase, the customer can change the requirement of the end product which increases the complexity of the scope and may increase the delivery time of the product.

THE BIG BANG MODEL

Big Bang Model does not have any defined process. Money and efforts are put together as the input and output come as a developed product which might be or might not be the same as what the customer needs.

Big Bang Model does not require much planning and scheduling. The developer does the requirement analysis & coding and develops the product as per his understanding. This model is used for small projects only. There is no testing team and no formal testing is done, and this could be a cause for the failure of the project.

Advantages of Big Bang Model:

- It’s a very simple Model.
- Less Planning and scheduling is required.
- The developer has the flexibility to build the software of their own.

Disadvantages of the Big Bang Model:

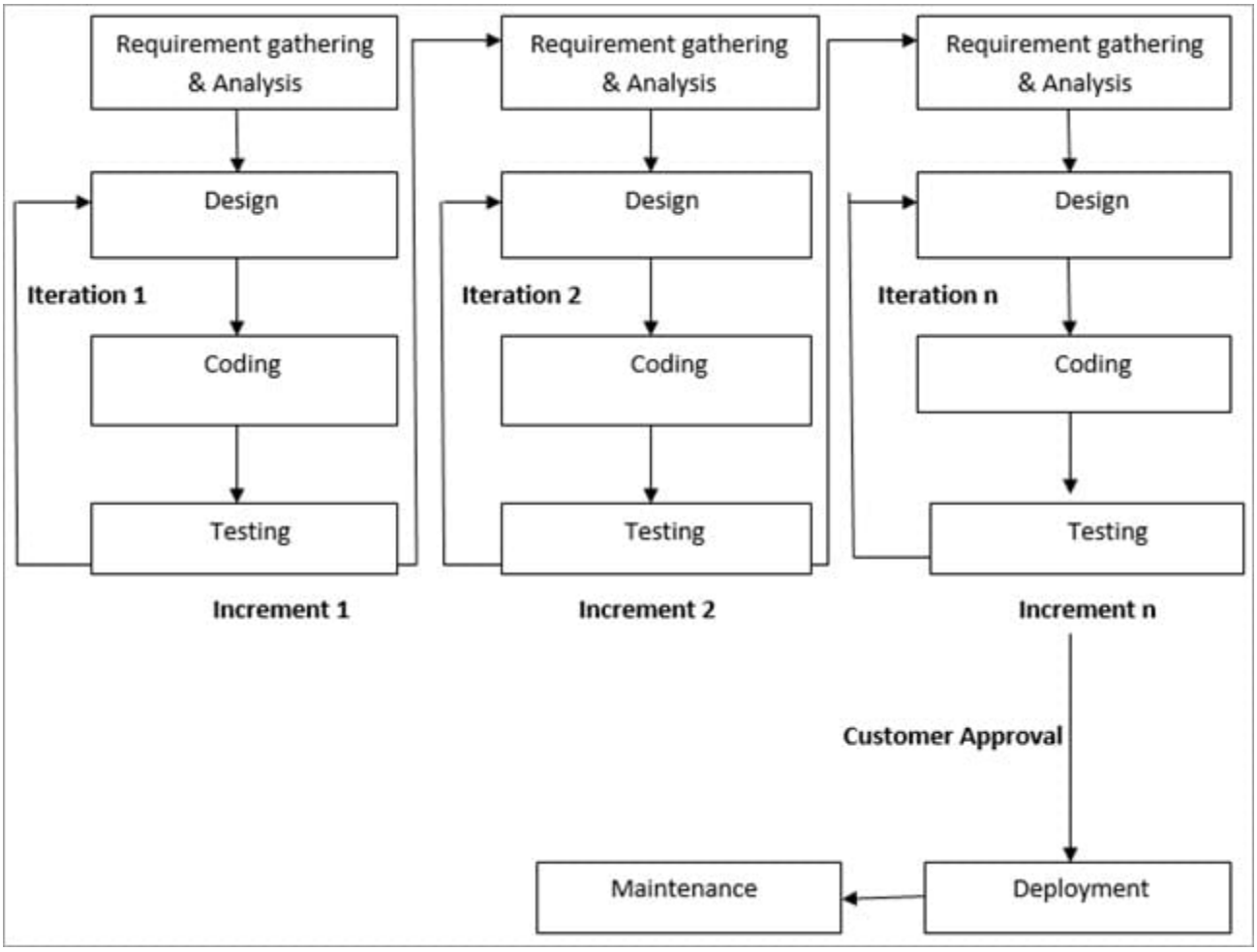
- Big Bang models cannot be used for large, ongoing & complex projects.
- High risk and uncertainty.

THE AGILE MODEL

Agile Model is a combination of the Iterative and incremental model. This model focuses more on flexibility while developing a product rather than on the requirement.

In Agile, a product is broken into small incremental builds. It is not developed as a complete product in one go. Each build increments in terms of features. The next build is built on previous functionality.

In agile, iterations are termed as sprints. Each sprint lasts for 2-4 weeks. At the end of each sprint, the product owner verifies the product and after his or her approval, it is delivered to the customer .Customer feedback is taken for improvement and his suggestions and enhancement are worked on in the next sprint. Testing is done in each sprint to minimize the risk of any failure.



Advantages of Agile Model:

- It allows more flexibility to adapt to the changes.
- The new feature can be added easily.
- Customer satisfaction as the feedback and suggestions are taken at every stage.

Disadvantages:

- Lack of documentation.
- Agile needs experienced and highly skilled resources.
- If a customer is not clear about how exactly they want the product to be, then the project would fail.

Conclusion

Adherence to a suitable life cycle is very important, for the successful completion of the Project. This, in turn, makes the management easier.

Example**,** in case of an unclear requirement, Spiral and Agile models are best to be used.

ASSIGNMENT 6

Difference Between Requirement and Specification in Software Engineering

The key difference between requirement and specification in Software Engineering is that a requirement is a need of a stakeholder that the software should address while a specification is a technical document with the analyzed requirements. A specification describes the features and behaviour of a software.

Software Engineering is the discipline of developing a software methodically. Requirements are the basis of the software. Requirement gathering and analyzing is a major phase of software development. SRS is the document that contains the analyzed requirements. Development phases such as designing, implementation use SRS.

Requirement	**Specification**
Requirements are descriptions of services that a software system must provide and the constrains under which it must operate.	Specification is a technical t document that describes the features and the behavior of software application.
Requirements to help describe what the software should do.	Specification helps to get the clear understanding of the product develop it and to minimize software failures.

Requirement in Software Engineering

The entire project depends on requirements. The first step to develop a software is to do a feasibility study. It focusses on technical aspects of the product. Next process is to gather requirements. It is possible by communicating with clients, end users and system users who will use the product at the end. Interview, surveys and questionnaires are main methods of collecting requirements. Finally, analyzing occurs after requirement gathering.

FUNCTIONAL AND NON-FUNCTIONAL REQUIREMENT
are two types of this requirement. A requirement that specifies a functional aspect of a software is a functional requirement. Hence, it defines a function of a system or a sub system. Furthermore, a library management system should add, edit, delete and search book details. It should also add, edit and delete member details. Moreover, it should calculate the fine for late returns. Those are few functional requirements of that system. A non-functional requirement defines expected characteristics of a software. Security, maintainability, usability, reliability and availability are some examples of nonfunctional requirements. Another type is business requirements. They define the business objectives, vision and goals.

Specification in Software Engineering

First of all, the clients and end-users describe their requirements in natural language. Documenting these requirements happens after analyzing. This document is called the Software Requirement Specification.Then, the system analysts convert them to technical language for the software development team.

This specification works as an agreement between the customer and the development team on what the software product should do. Proper specification helps to prevent software failures. It also helps the development team to

get a clear understanding of Requirement and Specification in Software Engineering

- A specification is a document with analyzed requirements.
- Difference Between Requirement and Specification in **Summary**

The difference between requirement and specification in Software Engineering is that a requirement is a need of a stakeholder that should be solved by the software while a specification is a technical document with the analyzed requirements.

ASSIGNMENT 7

GOOD SOFTWARE

CHARACTERISCTICS OF A GOOD SOFTWARE

For good quality software to be produced, the software design must also be of good quality. Now, the matter of concern is how the quality of good software design is measured? This is done by observing certain factors in software design. These factors are:

1. Correctness
2. Understandability
3. Efficiency
4. Maintainability

1) CORRECTNESS

First of all, the design of any software is evaluated for its correctness. The evaluators check the software for every kind of input and action and observe the results that the software will produce according to the proposed design. If the results are correct for every input, the design is accepted and is considered that the software produced according to this design will function correctly.

2) UNDERSTANDABILITY

The software design should be understandable so that the developers do not find any difficulty to understand it. Good software design should be self- explanatory. This is because there are hundreds and thousands of developers that develop different modules of the software, and it would be very time consuming to explain each design to each developer. So, if the design is easy and self-explanatory, it would be easy for the developers to implement it and build the same software that is represented in the design.

3) EFFIENCY

The software design must be efficient. The efficiency of the software can be estimated from the design phase itself, because if the design is describing software that is not efficient and useful, then the developed software would

also stand on the same level of efficiency. Hence, for efficient and good quality software to be developed, care must be taken in the designing phase itself.

4) MAINTAINABILITY

The software design must be in such a way that modifications can be easily made in it. This is because every software needs time to time modifications and maintenance. So, the design of the software must also be able to bear such changes. It should not be the case that after making some modifications the other features of the software start misbehaving. Any change made in the software design must not affect the other available features, and if the features are getting affected, then they must be handled properly.