INDIAN INSTITUTE OF ENGINEERING, SCIENCE AND TECHNOLOGY, SHIBPUR

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

B. TECH (CST), 6TH SEMESTER, MID TERM EXAMINATION, 2022

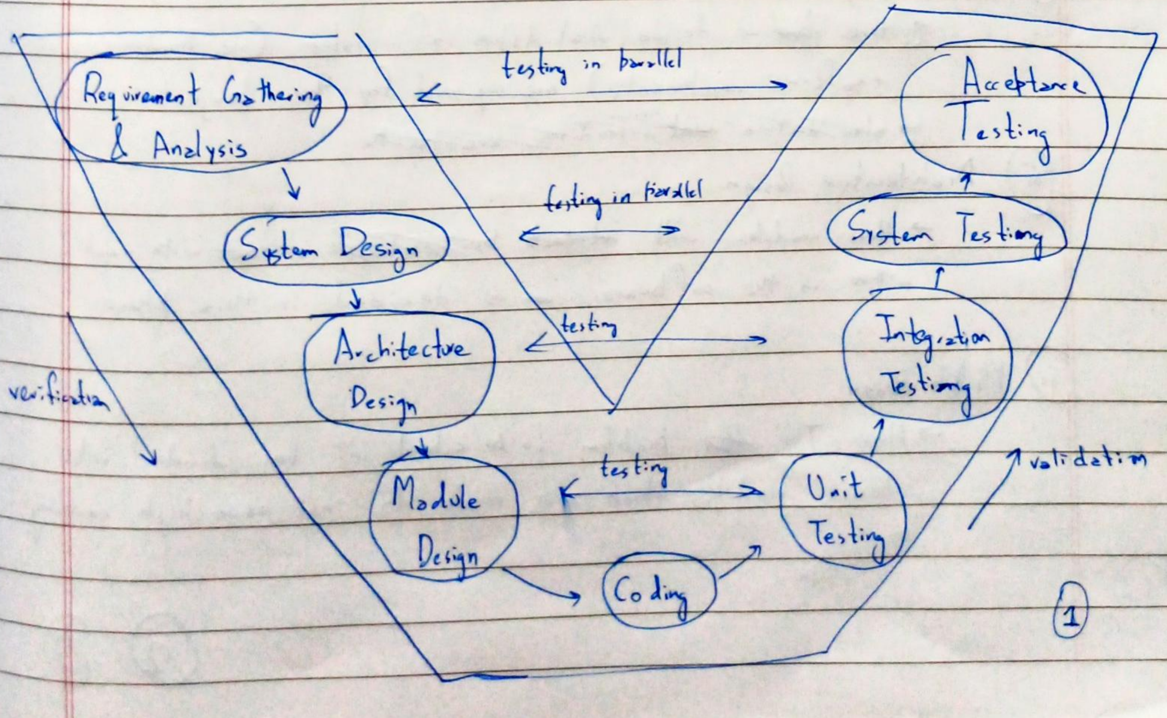
SOFTWARE ENGINEERING (CS 3203)

Date: 09/03/2022
Name: Abhiroop Mukherjee
Examination Roll No. : 510519109
G Suite ID : 510519109.abhirup@students.iiests.ac.in
No. of sheets uploaded : 7

(Q.1) V-Shaped SDLC Model

(a) → V-Shaped SLDC is a variant of the waterfall model, which emphasises on the verification & validation of the product being developed.

→ In this model, testing of product is planned in parallel with it's corresspbonding development phase.



(1)

→ Two major phases of SDLC

(i) Verification Phase

→ here we verify that the system requirements are being verified or not

(ii) Validation

→ here we verify that the code we have written to verify the requirements are actually valid or not

(b) Phases of SDLC

(i) Requirement Gathering & Analysis

→ In this phase System Analyst Analyze what users want and documents the things that the project should achieve in form of an Software Requirement ~~Sheet~~ Specification (SRS) Sheet

(ii) System Design

→ Here we decide what types of system (i.e hardware + software components) are required for the project

→ ~~also define what how modules are to be~~

(iii) Architecture Design

→ How modules will ~~interact~~ ~~be made &~~ interact with each other in the software is decided in this phase.

(iv) Module Design

→ Here The problem to be solved will be divided into cohesive modules which are made to not have high coupling

(v) Coding Phase

→ now all planning is done ~~thorouly~~ thoroughly and the software code gets written

(vi) Unit Testing

→ Developed Module are unit tested to make sure they are running as specified during Module Design Phase.

(vii) Integration Testing.

→ Here we ~~check~~ test that different modules are communicating together as described in Architecture Design Phase.

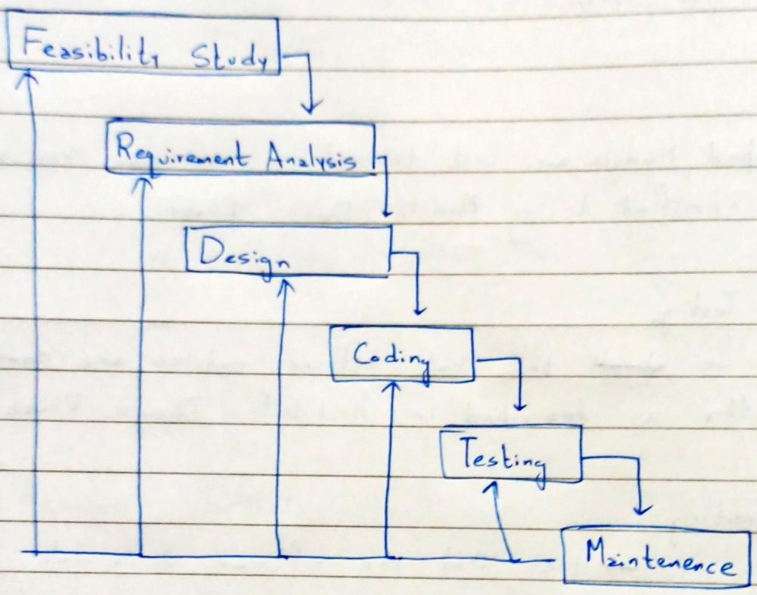
(viii) Systen Testing

→ Here we ~~che~~ test that the software as a whole is ~~runnig~~ running as intended in all systems that were decided during System Design Phase.

(ix) Acceptance Testing

→ Here we test that the software is beingt accepted by end users or not.

Q2 Waterfall Model (Iterative)



I → Iterative waterfall model Applicability

→ Iterative Waterfall model is applicable when

(i) Major requirements are defined but may slightly ~~change~~ change/~~alter~~ evolve over time.

(ii) We have experienced team of developers which can adapt to changing requirement

(iii) New Technologies is to be learnt by the developers.

(II) Advantages of Iterative Waterfall Model

(i) Feed back path from maintenence phase allows room for correcting any error which might creep up during ~~a~~ product development.

(ii) Model is sutable for comparitively large and complex project

(iii) Model is similar enough to traditional waterfall model & ~~is~~ hence ~~easy to~~ simple & easy to understand.

(iv) Customer Involvement is not required during software development.

(III) Disadvantages of Iterative Waterfall Model

(i) Once the Product development is done, there is no scope for evolution/incremental delivery

(ii) There is no overlapping of phase, ~~it~~ next phase can start only if previous phase has ended.

(iii) There is no mechanism to handle risk in this model

(iv) This model has limited customer interaction and hence ~~may~~ have a ~~risk~~ ~~a~~ risk of acceptance by customer.

Q4 I CLI : Command Line Interface

→ Allows user to enter command to the terminal to assign task to the software

→ Advantages

(i) Allows composing of complex commands
(ii) Fast Interaction with Software with respect to GUI
(iii) Easy to develop as we need to care about only texts for User Interface.
(iv) less memory usage (as only text UI) ~~or~~ compared to GUI

→ Disadvantages

(i) Absence of GUI make the software harder to learn
(ii) User needs to memorize commands
(iii) Keyboard only interface, can also cause typing error for users
(iv) Composing of complex commands is suitable for technical people only.

Eg: command line programs like bash shell, apt, ls, ifconfig, winget, etc

(II) GUI: Graphical User Interface

→ Allows user to use graphical widgets to interact with software

→ **Advantages**

(i) → Very easy to learn and use as graphical interface using metaphoric icons and pictures are ~~st~~ simple to understand, even beginners can understand a good GUI with a glance

(ii) Graphical Element can have customizable options to make the software more personal to customers

(iii) supports variety of input devices like mouse, keyboard, touch screen, voice typing, etc.

(iv) Typing error experienced in CLI is ~~[illegible]~~ mostly avoided

→ **Disadvantage**

(i) Slower than similarly made CLI software as it uses more computing and memory to handle graphical widgets.

(ii) composing complex commands ~~2~~ in CLI is not possible here

(iii) Harder to develop as ~~[illegible]~~ developer has more widgets to add.