Page No.
Date

Software Engg.

Software Engg. is the establishment and use of sound engg. principles in order to obtain economically software that is reliable and works efficently on real machines.

Software myths

Software myths i.e erroneous belives about software and process i.e used to build it. Myths have no. of attributes that make them insidious for instance they appear to be reasonable statement of facts (sometimes containing elements of truth), they have a initiative feel and they are often propologated by experience practicioners

Today mast loyal software engg. professionals recognize myths for what they thought i.e missleading attitudes that

has cause serious problems for managers and practioners alike. However, old attitudes and habits are difficult to modify and elements of software which

managment myth → manages the software responsibility like like ~~as~~ managers with most disciplin are often under pressure to maintain budget if schedules from slimingshaping and improve quality

myths -

① we already have a book i.e full of standards and procedures for boilding software would not that provide with everything they need to know?

The book of standards may very well exists but is it used. are software practioners are aware of its existence. Does it reflect modern software engg. practice? Is it complete? It it adaptible? Is it streamlined to improve time to delivery while still maintaining focus on quality in many cases the ans to all of this que is no.

② if we get behind the schedule, we can add more programmers and catch up, €

Reality → Software development is not a mechanistic process like manufacturing. In the words "Brooks adding people to ~~late~~ let software project makes it later." At first This statement may seem counter initiative. However as new peoples are added people who were working must spend time educating the new commers there by

reducing the amount of time spend on productive development effort. People can be added but only in a plan and well coordinated manner.

III If I decide to outsource a software project to a third party, I can just relax and let that firm built it.

Reality :- If an organisation does not understand how to manage and control software projects internally, it will inevitably struggle when outsources projects.

Customer's myths.

A customer who request Computer software may be a person at next desk, a technical group down the hall, the marketing / sales department, or an outside company that has requested software under contract. In many cases, the customer believes mix about software because software managers and practioners do literal to correct information. Myths lead to false expectation (by the customer) and ultimately dissatisfaction with developer.

I A General statement of objectives is sufficient to begin writting programs and we can fill details later.

Although a comprehensive and stable statement of requirments is always not possible, and ambiguous statement of objectives is a recipe for disaster. Unambigious requirments are develop only to effective and continous communication between customer and developer.

II Software requirements continually change, but change can be easily accomodated coz software is flexible, ~~it is true~~

Reality - It is true that Software req^s ~~m~~ change but the impact of change varies with time ~~in~~ at which it is introduce. when req^s changes requested early (before design or code has been started), the cross impact is relatively small, however as time passes, the cost impact grows rapidly, the resources have been commited, the design framework has been establish and change can cause upheavel that requires additional resources and major design modification.

→ Practicener's myth

myth's that are still belive by Software practicener's have been fostered by over 50 years of programming culture during the early days programming was viewed as art form.

i] once we write the program & get it to work, our job is done.

Reality: Someone once said "The sooner u begin writing code, the longer it will take u to get done". Industry data indicate that bet^n 60 - 80 % of all effort expended on software will be expended after it is delivered to customer for the first time.

ii] until I get program running, I have no way of assessing its quality

Reality :- One of the most effective software quality assurance mechanisms can be applied from the inception of a project i-e the technical review. Software reviews are quality filter that have been found to be more effective than testing for finding certain classes of software defects.

iii] The only deliverable work product for successful project is the working program.

Reality :- A working program is only one part of software configuration that includes many elements. Veriety of work products (for ex. models, documents, plans) provide a foundation for successful engg. and more Imp. Guidance for software support

iv] Software engg. will make us create voluminous and unnecessary documentation and with Invariably slow us down

Reality : Software engg. is not about creating documents it is about creating quality product. Better quality leads to reduced rework. faster delivery times.

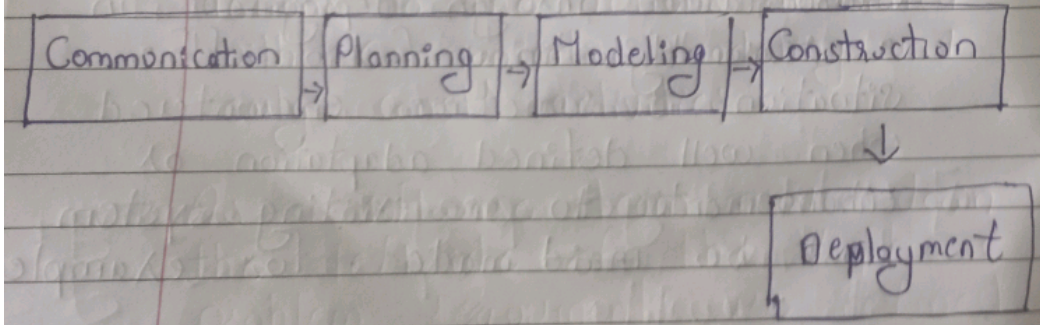
→ Process models / paradigm

1] waterfall model

There are times when the requirements for problem are well understood that is when work flows from communication through deployment in a reasonably linear fashion. The situation is sometimes encountered when well defined adaptation or enhancements to an existing system must be ~~maid~~ made. for example

an adaptation to accounting software that has been mandated by govt of changes to govement regulations. It may also occur in a limited no of new development efforts, but only when req^t are well defined and reasonably stable.

The waterfall model sometimes called the classic ~~tyf~~ lifecycle suggest a systematic a sequential approach through software development that begins with customer specification of requirment and progress through planning, modeling, construction and deployment.



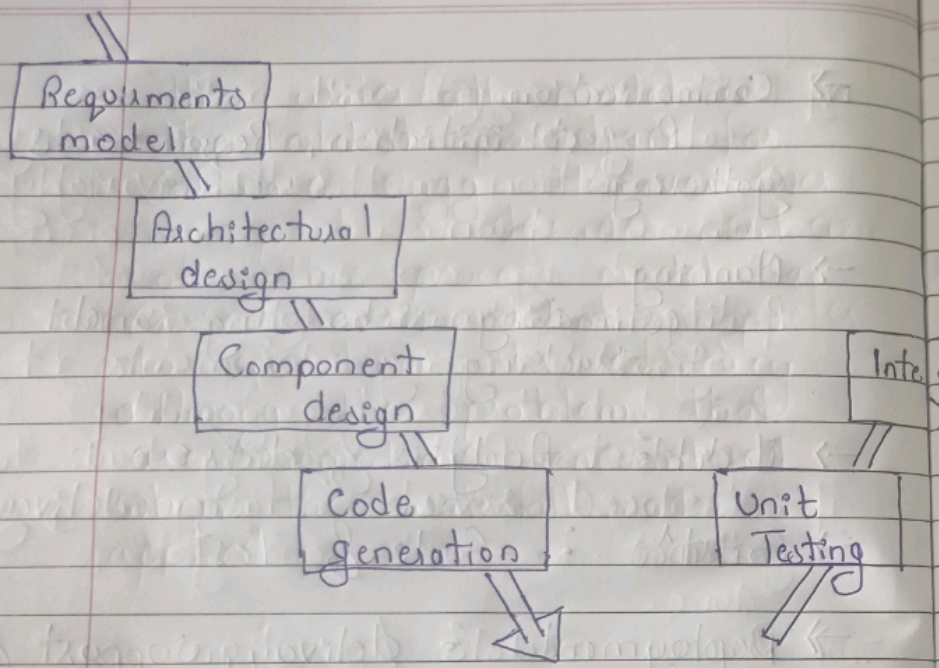
→ Communication
Project initiation, Requirments gathering

→ Planning
estimating, schedoling and tracking

→ Model : Analysis & design

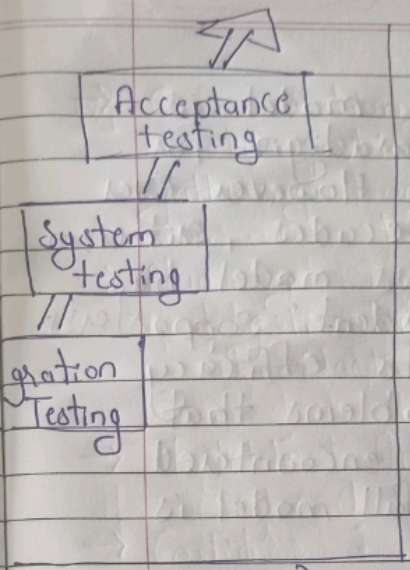
→ Const^n : Code & Test

→ Deployment : delivery, support & feedback.



A variation in the representation of the waterfall model is called V model represented in above fig the V-model depicts the relationship of quality assurance actions to the actions accosiated with communication modeling and early const^n activities

As a software team moves down the leftside of the V, basic problem requirments are refined



into progressively more detail and technical representation of the problem and its soln. Once code has been generated The team moves up the right side of V essentially performing a series of test (i.e QAS) that validate each of the models created as the team moved down the left side. In reality there is no fundamental diff bet^n the classic lyfecycle and the V model. The V model provides a way of visualizing how verification and validation actions are applied to earlier engg. work.

Page No.
Date

The waterfall model is the oldest paradigm for software engg. However over the past 30 decades, criticism of this process model has caused even ardent supporters to question its efficacy.

- Among the problems that are sometimes encountered when the waterfall model is applied.