Chapter-1

(1) What is software? Write its application.

Ans: Software is –

(1) instruction that provide desired features, function & performance.
(2) Data structure that enables Program to manipulate information.
(3) Documentation that describes operation & use of the Program.

Application:

(1) System Software.
(2) Application software.
(3) Web/mobile application.
(4) Engineering software.
(5) Embedded software.
(6) Product line Software.
(7) Artificial & Intelligence software.

(2) How software is different from other traditional Engineering branches?

Ans:

(1) Software is logical rather than physical system element.

(2) Software is developed or engineered. It is not manufactured in classical sense.

(3) Software doesn't wear out.

(4) Although industry is moving towards component based construction, most software continues to be custom built.

(3) Difference bet^n:

Hardware	Software
(1) Manufactured	(1) Developed or Engineered
(2) Wear out	(2) Deteriorate.
(3) Maintanance simple	(3) Complex.
(4) Component built construction	(4) Custom built construction
(5) Relatively simple	(5) Relatively complex.

(4) Software doesn't wear out but it does deteriorate - Explain.

ব্যবহারের কারণে ক্ষয় হওয়া

অবনতি

Ans:

Hardware:

(1) Hardware exhibits high failure rate early in its life.

(2) Then the defects are corrected and failure rate drops quite low.

(3) As time passes, the failure rate rise again as hardware component suffer from the cumulative effects of dust, vibration, abuse etc. So, hardware begins to wear out.

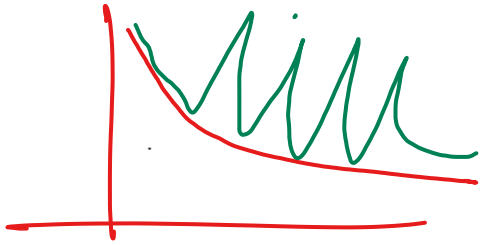
Software:

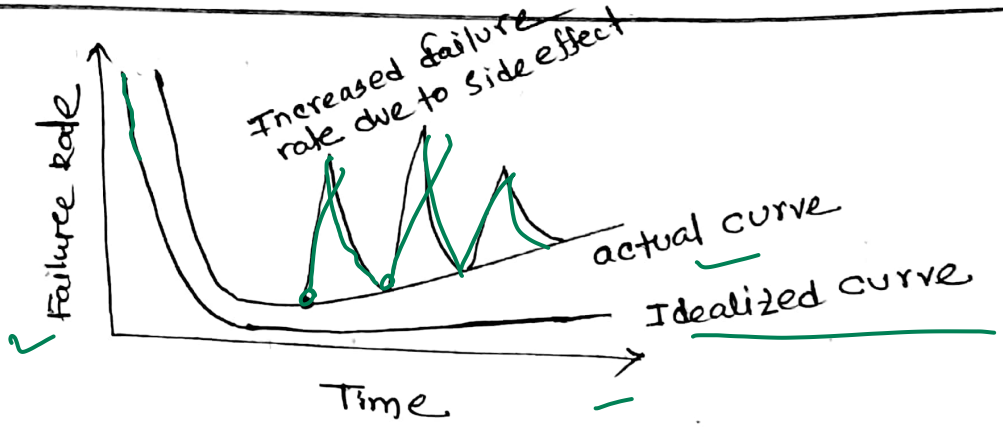
(1) Software is not susceptible to environmental maladies.

রোগ

(2) Due to undiscovered defects it exhibits high failure rate early in its life.

(3) Then defects are corrected and failure rate drops. But actual picture is not like this.





(4) Due to changes and errors there are spike in the failure rate curve.

(5) Before the curve can return to the original steady state, another change is requested that causing another spike.

Thus, the minimum failure rate begins to rise due to change software is deteriorating.

(5) What is legacy system? Write down its characteristics.

legacy software: Legacy software systems were developed decades ago and have been continually modified to meet the changes in business requirements & computing Platforms.

Characteristics:

- longivity.
- business criticality.
- costly to maintain.
- risky to evolve.
- Poor quality.

** what type of changes are made to legacy systems?

Ans:

(1) The software must be adopted to meet the needs of new computing environment.

(2) the software must be enhanced to implement new business requirements.

(3) the software must be extened to make it work with more modern system.

(4) The software must be re-architected to make it Viable within an evolving computing ~~sys~~ environment.

→ ব্যবহার

(6) Software take dual ~~rate~~ role - 'Product & Vehicle' — Explain.

Ans:

As a Product software ~~engineering~~ delivers Computing potentials by computer hardware or by network of computer.

As a vehicle, software acts as the basis of or the control of the OS computer, the communication & Creation & control of other Program.
Software development tool.

(7) What is software engineering? Elements of S/W Process.

Software engineering is the application of systematic, disciplined and quantifiable approach to the development, operation and maintenance of a software.

A Software process is a collection of activities, action and task that are Performed when some work Products are to be created.

Elements of S/W Process:

(1) **Activity:** It strive to achieve broad objective & its applied regardless of application domain, size of product etc.

(2) **Action:** Encompasses a set of work that produces major work product.

(3) **Task:** Accomplishes some part of work implemented by the action.

AAT
- Task
- Action
- Activity

** The Process framework : (CPMCD)

CPM CD

A generic process framework for S/W engineering encompasses five activities :

(1) Communication :

- involves communication & collaboration with user & other stakeholders.
- Encompases requirement gathering & other related activities.

(2) Planning :

- Includes technical tasks are to be conducted, risks that are likely, required resources, work product to be Produces.

(3) Modeling :

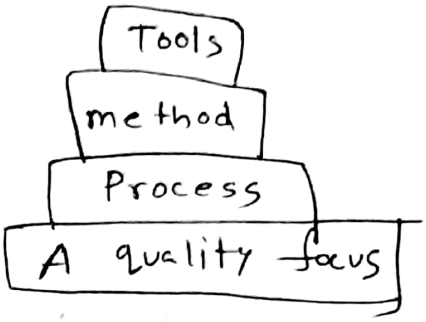
- modeling is an important framework to better understand software requirements and the design to fulfill requirements.

(4) Construction : Includes — Code generation
— Testing

(5) Deployment : — software is delivered to customer.

- Customer evaluate the delivered Product and Provide feedback.

** SE is a layered technology? Explain it.



(1) A quality focus:

- It is the bed rock of SE.
- It leads to the development of various approach to SE.

(2) Process:

- foundation of SE.
- holds technology layer together.
- defines a framework that must be establish for effective delivery of software.

(3) Method:

- Provides the way to build software.
- encompases a broad array of task that includes communication, requirement analysis, design modeling, testing & support.

(4) Tools:

- Provides automated & semi-automated support for Process & methods.
- Tools are integrated so that information created by one tool can be used by another.

** Umbrella activities:

Umbrella activities are applied throughout a software project & help software team to manage and control progress.

Typical Umbrella activities are —

(1) Software project tracking and control.
(2) Risk management.
(3) Software quality assurance.
(4) Technical reviews,
(5) Measurement.
(6) Software configuration management.
(7) Reusability management.
(8) Work Product Preparation & Production.

** Essence of s/w engineering: (George Polya)

(1) Understand the problem (communication & analysis)
(2) Plan a solution (modeling & software design)
(3) Carry out the plan (code generation)
(4) Examine the result for accuracy (testing & QA)

** General Principles: (David Hooker)

① The Reason It All Exists:

A software system exists for one reason: to provide valve to its users. All decisions should be made with this in mind.

② KISS (keep it simple, stupid!):

- all design should be as simple as possible.

③ Maintain the Vision:

- A clear vision is essential to the success of a software project.

- Compromising the architectural vision of a software system weakens & will break the well designed system.

④ What you produce, others will consume. :

- Always specify, design, document and implement knowing someone else will have to understand what are you doing.

⑤ Be open to the future :

- Never design yourself into a corner. Always ask 'what if' and prepare all possible answers by creating systems that solve the general problem, not just specific one.

⑥ Plan ahead for reuse :

- Reuse saves time and effort.

⑦ Think ! :

- Placing clear, complete thought before action almost always produces better result.