Chapter (1) (P)
Introduction to software Engineering

What is software?

Software is —

(1) instructions (computer programs) that when executed provide desired features, functions and performance.

(2) data structures that enables the program to adequately manipulate information.

(3) documents that describe the operation and use of the program.

How it is different from other traditional engineering branches

As software is logical rather than a physical system element, its characteristics that are considerably different are —

1. Software is developed or engineered; it is not manufactured in the classical sense.
2. Software does not "wear out".
3. Although industry is moving toward component-based construction, most software continues to be custom built.

Hardware Vs. Software:

#	Hardware	Software
1.	Manufactured	Developed / engineered
2.	Wears out	deteriorates
3.	Component-built construction	Custom built construction
4.	Relatively simple	complex
5.	Maintenance is also simple	complex.

Wear Vs. Deterioration:

Bathtub curve depicts failure rate as a function of time for hardware. This curve indicates that hardware exhibits relatively high failure rates on defects early in its life. Defects are the corrected and failure rate drops to a steady state level. for some period of time.

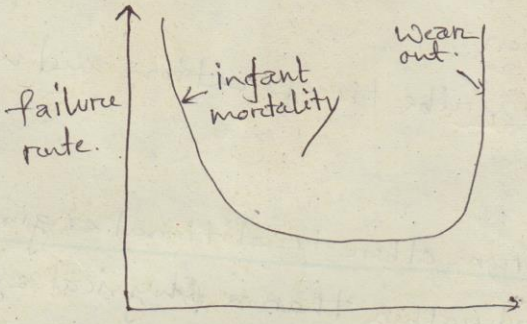
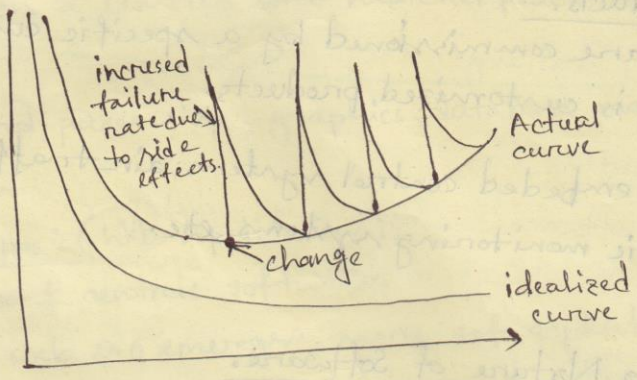


Fig: Failure curve for hardware (bathtub curve)

But with time, failure rate rises again as hardware components suffer from cumulative affects of dust, vibration, abuse, temparature extremes, and many other environmental maladies.

This rise of failure rate due to environmental effects is called "wear out".

On the other hand, software is not subpectible to environmental maladies. Idealised curve depicts failure rate for software. failure to software is caused due to undiscovered ~~errors~~ defects at the early stage. These are corrected and curve flattens as shown. But actual picture is not like that-



Actual curve depicts ~~shows~~ this contradiction. During its life software will undergo changes. As changes are made, it is likely that errors will be introduced, causing the failure rate curve to spike. Before the curve can return to original steady-state failure rate, another change is requested, causing the curve to spike again.

Slowly, the minimum failure rate level begins to rise, - the software is deteriorating due to change.

② Software Products: (What do you mean by service and customized software products?)

☐ Generic Products: (serve software products):

Stand-alone systems that are marketed and sold to any customer who ~~not~~ wishes to buy them. is generic products.

Examples - Pc softwares such as editing, graphics program etc.

☐ Customised Products:

Software that are commissioned by a specific customer to their own needs is customized products.

examples – embeded control systems, air traffic control software, traffic monitoring systems, etc.

1.3 The Changing Nature of Software:

Software Application:

(1) System Software:

→ system software is a collection of programs written to service other programs.

compilers, editors, file management utilities, etc.

(2) Application Software:

→ Application software consists of standalone programs for specific needs.

(3) Engineering/scientific software:

→ Characterized by "number crunching" algorithms.

automotive stress analysis, molecular biology, orbital dynamics etc

(4) Embeded software:

→ resides within a product or system.

key pad control of a microwave oven, digital function of dashboard display in a car etc.

(5) Product line Software:

→ Designed to provide a specific capabilities for use by many different customers, f

→ focus on a limited ~~and~~ market place to address mass consumer market.

word processing, graphics, database management etc.

6. WebApps: (Web Applications):

→ network centric software.

→ As web 2.0 emerges, more ~~sof~~ sophisticated computing environment is supported integrated with remote database and business applications.

7. Artificial Intelligence Software:

→ makes use of non-numerical algorithms to solve complex problems that are not amenable to computation or straight-forward analysis.

Robotics, expert systems, pattern recognition, game playing etc.

Software new categories:

→ Open world Computing: pervasive, distributed computing

→ Ubiquitous Computing: wireless network.

→ Net sourcing: Web as a computing engine.

→ Open Source: "free" source code open to the computing community

→ Data mining;

→ Grid computing

→ Cognitive machines

→ Software for nanotechnologies.

- ~~What is legacy software?~~
- List out the key challanges faced by software engineer.

☒ Ubigitous computing:

Challenges for software engineer is to develop systems and applicati- software that will allow small devices, personal computers and enterprise system to communicate accross vast networks.

☐ Netsourcing:

Challenges is to ~~ant~~ architect simple and sophisticated applicat- -ons that provide benefit to targited end-user markets worldwide.

☐ Open Source:

-b to build source-code that is self-descriptive, but more importan- -ly to develop techniques that will enable both customers and developers to know what changes have been made and how those changes manifest themselves within the software.

☐ The "new economy":

- to build applications that will facilitate mass communication and mass product distribution using concepts that are only now forming.

What is legacy software?

Legacy Software:

Legacy software systems ... were developed decades ago and have been continually modified to meet changes in business requirements and computing platforms. The proliferation of such systems is causing headaches for large organizations who find them costly to maintain and risky to evolve.

— Dayani-Fraad & his colleagues.

* Characteristics:
  - longevity and
  - business criticality.
  - poor quality (can be present)

Why must it change?

What type of changes are made to legacy systems?

→ Software must be adapted to meet the needs of new computing environments or technology.

→ Software must be enhanced to implement new business requirements.

→ Software must be extended to make it interoperable with more modern systems or databases.

→ Software must be re-architected to make it viable within a network environment.

When any of these modes of evolution occur, a legacy system must be ~~re~~reengineered so that it remains viable & into the ~~fu~~future.