AGILE-PARADIGM SHIFT IN SDLC

Session - 1

Software Characteristics

- Define software

- Explain the need for software

- Describe the characteristics of software as a process

- Describe software as a product

- List the characteristics of a software

- List and describe the different types of software

◆ Electronic computers first evolved in 1940s.

Early efforts were focused on designing the hardware, but with advent of new technologies, hardware became **cheaper** and **more powerful**.

With more user-friendly operating systems and high-level languages, **software development** rapidly grew.
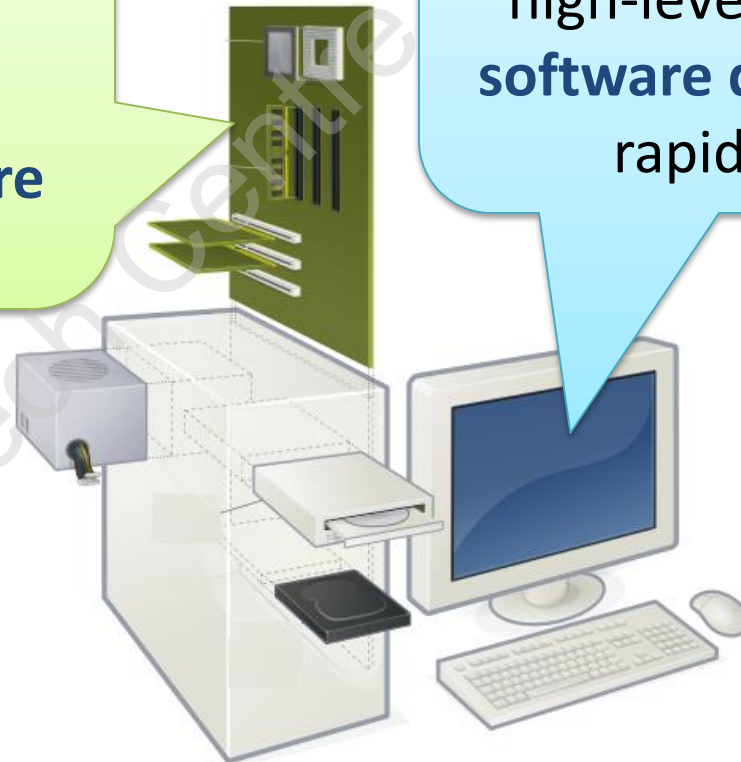
The techniques for writing simple programs could not be scaled up for developing software systems, and the computing world found itself in the midst of a 'software crisis'.

- Software Engineering:
    - Was coined in the conference sponsored by NATO Science Committee in Europe in 1960.
- IEEE defines software engineering as:

"The application of a systematic, disciplined, quantifiable approach to the development, operation, and maintenance of software".

Software

- Today, software plays dual role:
  - It is a product and the vehicle to deliver a product.

- Software is defined by IEEE as, '**a collection of computer programs, procedures, rules, and associated documentation and data**.'

- Components include:

Instructions or computer programs	• Execution of desired features, functions, and performance.
Data structures	• Store information required by programs to work upon or process.
Documentation	• Describe the operation and use of programs.

Need for Software

- Software either automates or speeds up existing manual task with increased accuracy.

- Some of the reasons for need for software are:

A market trend or a requirement

A business essential

A client requirement

A legal requirement

An upgrade from manual to automated operations

Systems such as transportation, medical, telecommunications, military, industrial, entertainment, and so on use software

Development of Software

- Development of software is an ongoing process.

- As the business needs change, the software may become less useful with the new changes in the business requirement.

- Hence, it constantly needs to be updated or replaced.

Software as a Process [1-6]

A software process is a set of activities, together with ordering constraints among them, such that if the activities are performed properly and in accordance with the ordering constraints, the desired result is produced.

◆ The fundamental objectives of a process are optimality and scalability.

Optimality — Ability of a process to produce high-quality software at low cost.

Scalability — Ability of being applicable for large software projects.

◆ A few important properties are:

    ◈ Predictability

    ◈ Maintainability

    ◈ Defect Removal and Prevention

    ◈ Process Improvement

Predictability

- Is a process that determines how accurately the outcome of following process in a project can be predicted before the project is completed.

- If not predictable, then the process is of limited use.

- With low predictability, the experience gained through projects is of little value.

Maintainability

- Development costs exceed during the life of the software.

- To reduce the cost of software, the goal of development should be to minimize the maintenance effort.

❖ Table shows the effort distribution within phases of a software process.

| Activity	Percentage
Requirements	10%
Design	20%
Coding	20%
Testing and Maintenance	50%

❖ Table shows the distribution of how programmers spend their time on different activities.

Programmer Activity	Time Percentage
Writing Programs	13%
Reading Programs and manuals	16%
Job Communication	32%
Others (including personal)	39%

Defect Removal and Prevention

◈ Errors can occur at any stage of the development cycle.

◈ The cost of correcting the errors of different phases is not the same, and depends on when the error is detected and corrected.

◈ If the error were corrected after coding, both design and code would have to be changed, thereby increasing the cost of correction.

◈ Figure shows the error detection in every phase.
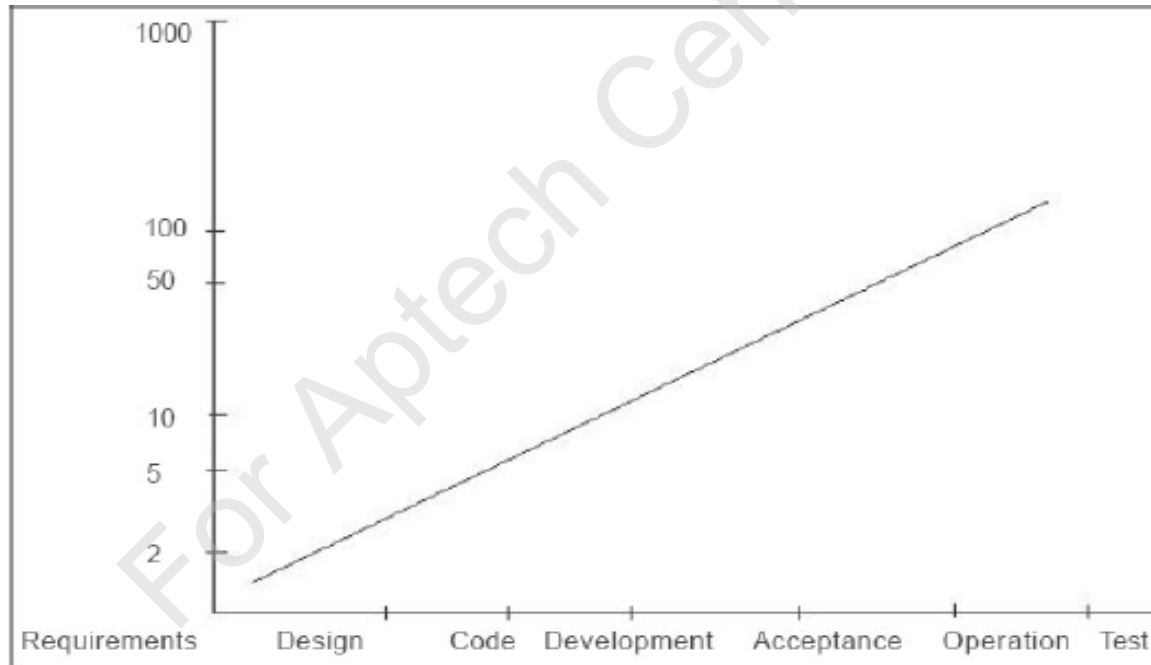
Process Improvement

◈ A process is not a static entity.

◈ A process must be improved to satisfy goals as quality improvement and cost reduction.

◈ Each project done using the existing process must feed information back to the process itself, which can then use this information for self-improvement.

Software as a Product [1-8]

- The goal of any engineering activity is to build something - a product.

- For example:
  - The civil engineer builds a dam.
  - The aerospace engineer makes a plane.
  - The electrical engineer makes a circuit.
  - The product of software engineering is a '**software system**'.

Software as a Product

- Is not a physical entity. One cannot touch or feel it to get an idea about its quality.

- Is a logical entity, and therefore, it is different from other engineered products.

Software as a Product [2-8]

◆ The characteristics of software that makes it different from other products are:

  ◈ Software is developed or engineered, it is not manufactured

  ◈ Software is malleable

  ◈ Software does not 'wear out'

  ◈ Most software are custom-built, rather than being assembled from existing components

Software as a Product [3-8]

Software is developed or engineered, it is not manufactured

- ⬥ Software does not automatically roll out of an assembly line.

- ⬥ There exist a number of tools for automating the process, especially for the generation of code.

- ⬥ The development depends on the individual skills and creative abilities of developers.

- ⬥ In most engineering disciplines, the manufacturing process is considered carefully, because it determines the final cost of the product.

- ⬥ For software, on the other hand, manufacturing is a trivial process of duplication.

- ⬥ The software production process deals with design and implementation, rather than manufacturing.

Software is malleable

◈ The characteristic that sets software apart from other engineering products is that software is malleable.

◈ The product itself can be modified easily. This makes it quite different from other products such as cars and ovens.

◈ The property of malleability of software can be used to advantage –

　◈ Provided it is done with a lot of discipline; this is where procedures and quality standards for making modifications become important.

Software does not 'wear out'

◈ Figure shows the relationship between failure rate and time for hardware.
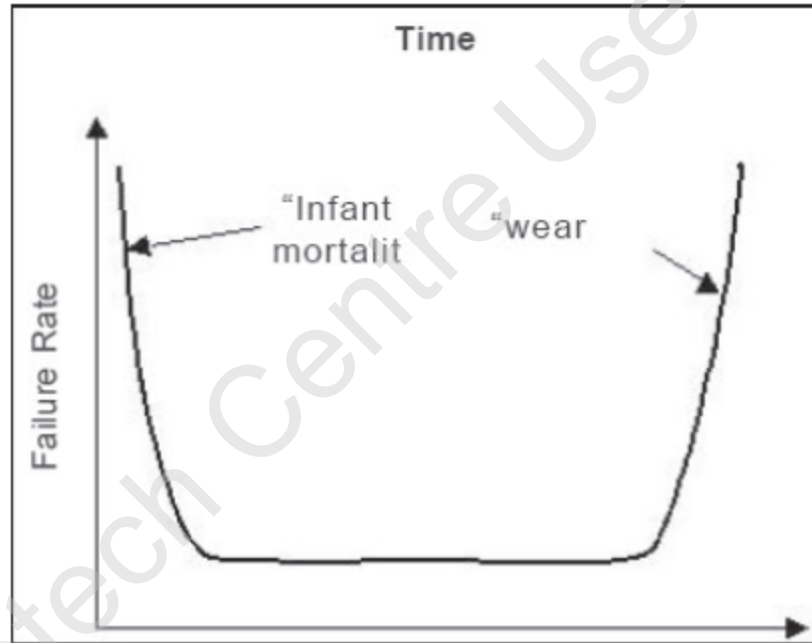
◈ The relationship depicts that the hardware shows high failure rate in its early life cycle.

◈ As time passes by, the failure rate rises again as hardware begins to wear out due to cumulative effects of dust, vibration, abuse, temperature extremes, and many other environmental maladies.

◈ Figure shows the curve for software failure against time.
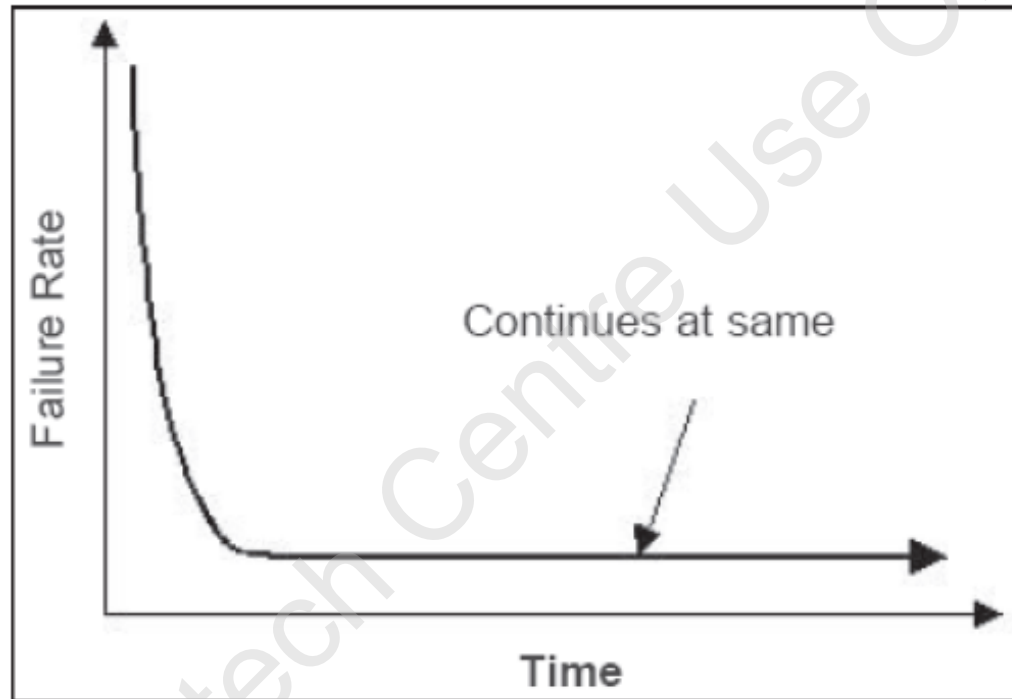
◈ The defects in the early stages will cause high failure rates in the early life cycle.

◈ As the defects are corrected, the failure rate drops, and the curve flatten.

◈ This contradicts the earlier curve because software undergoes maintenance during its life.

◈ As changes are made, new defects are introduced, causing the failure rate to spike.

◈ However, before the curve can return to the original steady state, a new change is requested, causing the curve to spike again.

◈ Slowly, the minimum failure rate level begins to rise; thus software deteriorates due to continuous change requests.

◈ Figure shows failure rate for software.
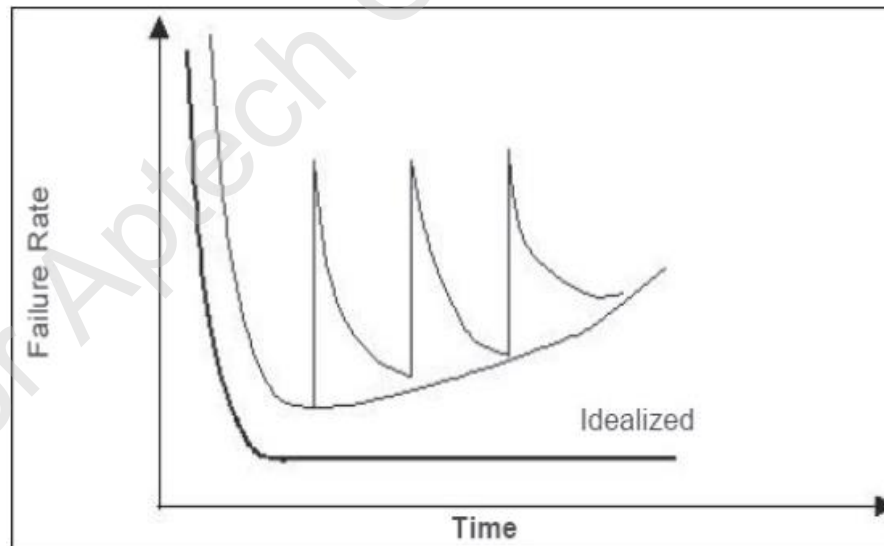
Software as a Product [8-8]

Most software are custom-built, rather than being assembled from existing components

- For building hardware, the digital components are assembled to achieve proper functioning.

- These digital components can be ordered 'off the shelf'.

- Earlier, software can be ordered off the shelf, but as a complete unit, not as a software component that can be assembled into software programs.

- Now, the widespread use of object-oriented technology has resulted in the creation of software engineering components that:

  - Can be assembled individual components into software programs.
  - Promoting '**software reusability**'.

Types of Software [1-2]

◆ Software can be classified into two categories:

◈ Application Software

◈ System Software

Application Software

- Are designed to allow the user of the system to complete a specific task or set of tasks.
- Include programs such as Web browsers, office software, games, and so on.

System Software

- Sits directly on top of computer's hardware components.
- Includes the range of software you would install to your system that enables it to function.
- For example, the operating system drivers for hardware devices, linkers, and debuggers.

◆ Figure shows the system software.
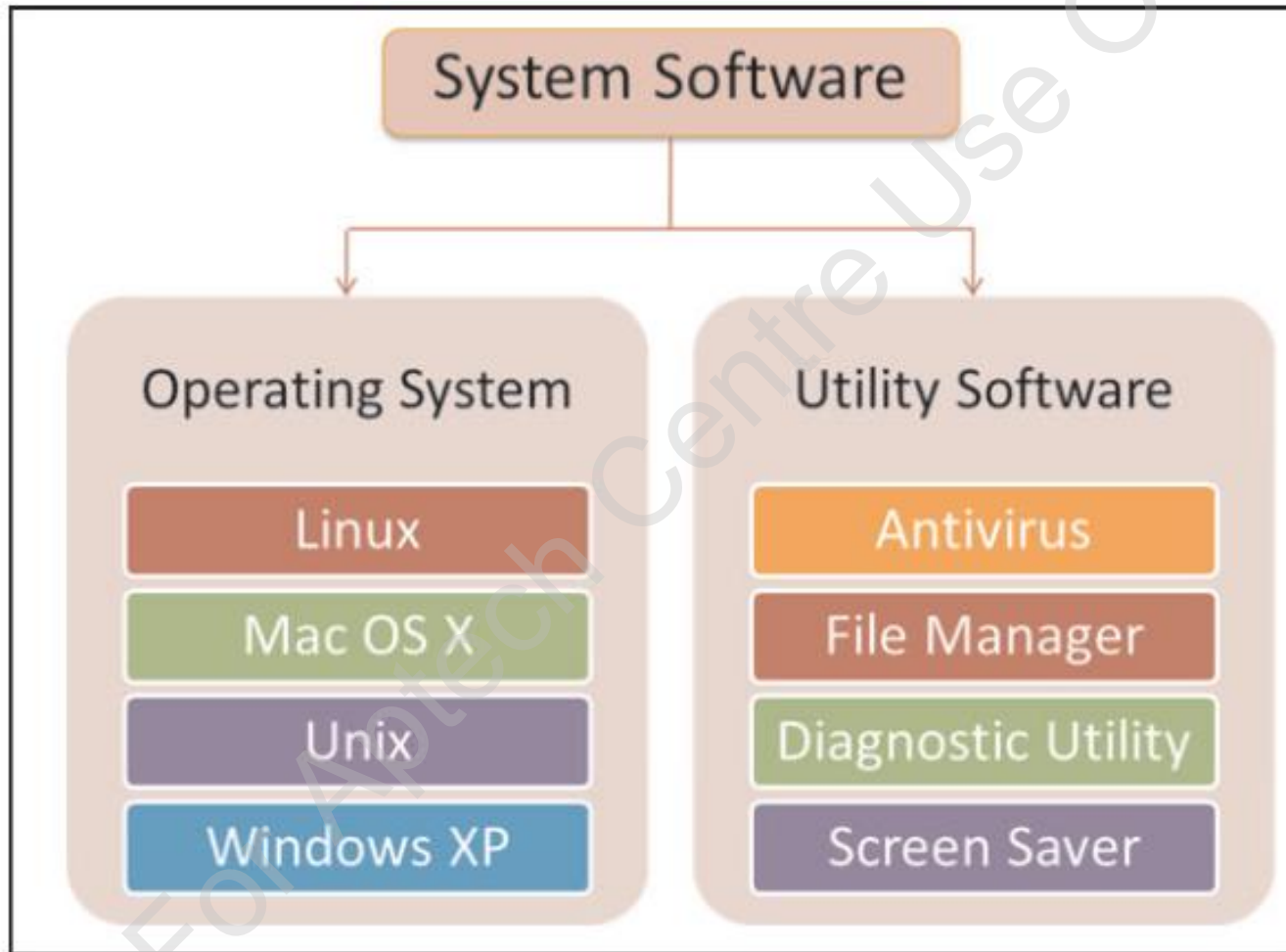
Summary

- Software is a product, and at the same time, the vehicle to deliver a product.

- IEEE defines software as a collection of computer programs, procedures, rules, and associated documentation and data.

- Software is developed to either automate or speed up an existing manual task with increased accuracy.

- A software process may be defined as 'a framework for the tasks that are required to build high-quality software.'

- The fundamental objectives of a process are optimality and scalability.

- The product of software engineering is a 'software system'. It is not as tangible as the other products, but it is a product nonetheless.

- Software can be classified into two categories namely, Application software and System software.