T. Y. B. Sc. (Computer Science) Semester-V
DSC (UG-CS-503) Software Engineering

1. Introduction to Software Engineering

1.1 Software and Software Engineering
1.2 Evolution of Software
1.3 Software Characteristics
1.4 Software Applications
1.5 Software Myths
1.6 Software Process
1.7 Software Development Life Cycle (SDLC)

1.2 Evolution of Software

- Software-engineering principles have evolved over the past more than 50 years from art to an engineering discipline.

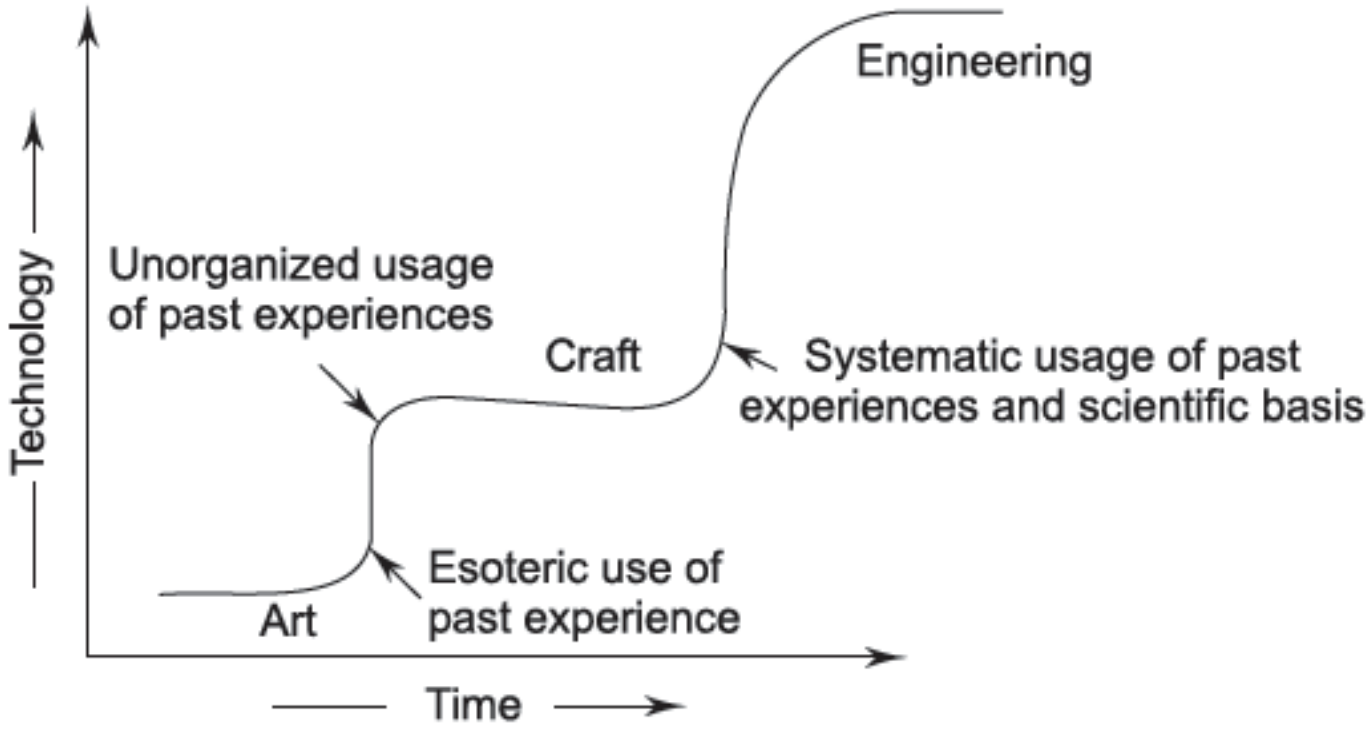


Figure : Software Evolution

Four eras of software development process :

Early Era : (1950 to 1960)

- The general-purpose hardware became commonplace
- Software were Customized (Custom-designed)
- Software were Limited in distribution
- Batch orientation (ultimately used by the same person or organization)

Four eras of software development process :

Second Era : (1960 to 1972)

- Due to new concepts of human-machine interaction, Real-time software deals with the changing environment
- Changing environment (multi-users environment) can work with software at the same time
- Multi-programming
- Databases
- Product software

Four eras of software development process :

Third Era : (1973 to 1985)

- The software was consumer designed
- (Distributed systems) distribution was not limited
- Low-cost hardware
- Embedded intelligence

Four eras of software development process :

Fourth Era : (1985 to till present)

The evolution of computer systems moves us away from individual computers and computer programs and toward the collective impact of computers and software.

- Powerful desktop systems
- Expert systems
- Artificial intelligence
- Network computers
- Parallel computing
- Object-oriented technology

1.3 Software Characteristics

- Software is often the single largest cost item in a computer-based application

- It is a product differs from other physical products

1. Most software is custom-built, rather than assembled from existing components

- Most software continues to be custom built, although recent developments tend to be component-based. Modern reusable components encapsulate both data and the processing applied to data, enabling the software engineer to create new applications from reusable parts. For example, today a GUI is built using reusable components that enable the creation of graphics windows, pull-down menus, and a wide variety of interaction mechanisms.
- The data structures and processing details required to build the interface are contained in a library of reusable components for interface construction.
- Software designers are not afforded the luxury of obtains components off the self. Generally it can be obtained as a complete unit and not as components that can be reassembled into new program. The situation is changing rapidly and software reusability is beginning to emerge**.**

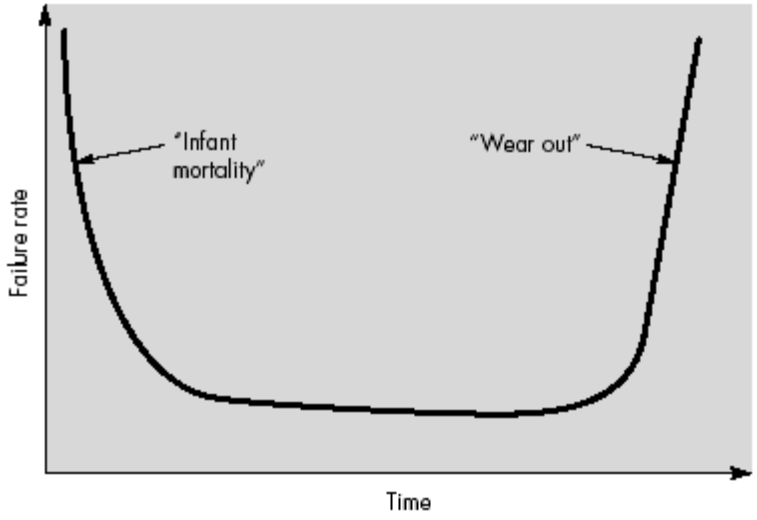
2. **Software is developed or engineered, it is not manufactured in the classical sense**

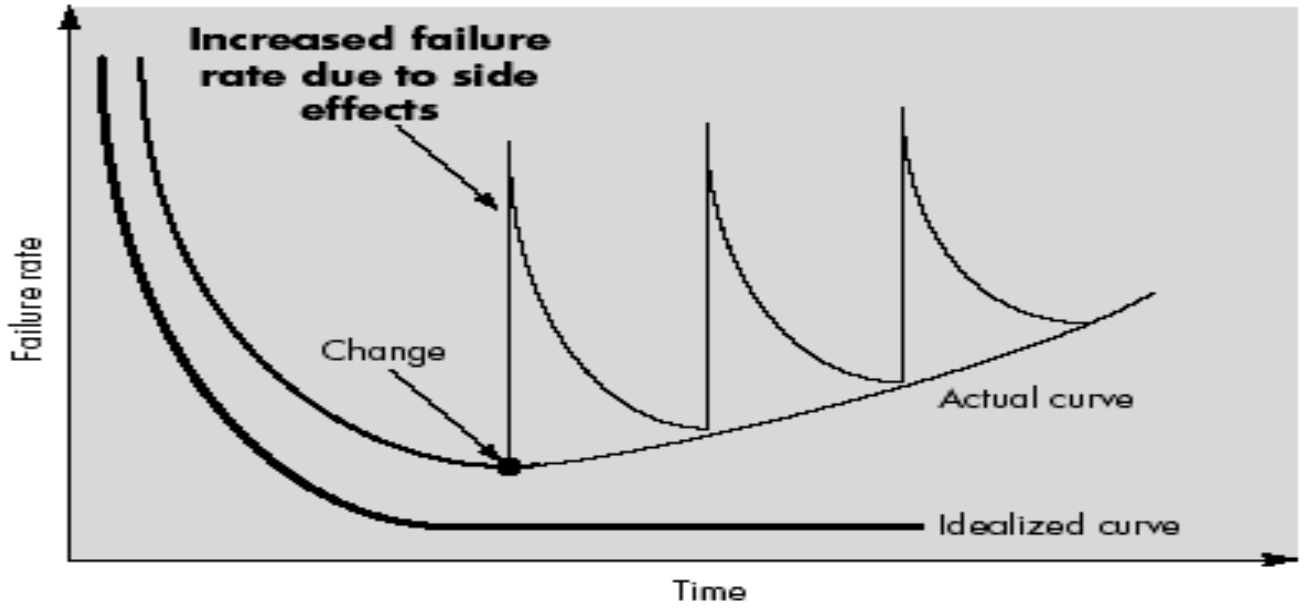
- Since the process of development of software is art as well as science, it is not manufactured like other physical product.

- Basically we cannot say software is manufactured but it is a development process.

3. **Software doesn't "wear out"**

- The hardware components suffer from the cumulative effect of dust, vibration abuse temperature extremes and many other environmental problems. So that hardware exhibits relatively high failure rate early in its life.
- After some days or some specific time hardware begins to wear out.
- However software has no risk to the environmental problems that cause hardware to wear out.





Software is not susceptible to the environmental maladies that cause hardware to wear out. In theory, therefore, the failure rate curve for software should take the form of the “idealized curve” like a zig-zag form shows in figure.

Undiscovered defects will cause high failure rates early in the life of a program. However, the implication is clear software doesn't wear out. But it does deteriorate. Before the curve can return to the 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.

The difference between hardware and software regarding to wear out are :

- When a hardware component wears out, it is replaced with another new component and software has a code.

- Software maintenance is more complex than hardware maintenance.

4. Software is flexible

- A program can be developed to do almost anything.
- Sometimes, this characteristic may be the best and may help us to accommodate any kind of change.
- Reuse of components from libraries help in reduction of effort.
- Now-a-days, we reuse not only algorithms but also data structures.
- Whatever the need required by customer changes will be reflected in the software.

1.4 Software Applications

System Software:

- The software developed for especially for computer system or effective utilization hardware is known as system software. They are compiler, OS, editor etc.
- The system software area is characterized by heavy interaction with computer hardware; heavy usage by multiple users scheduling, resource sharing ,etc.

1.4 Software Applications

1. System Software:

- The software developed for especially for computer system or effective utilization hardware is known as system software. They are compiler, OS, editor etc.
- The system software area is characterized by heavy interaction with computer hardware; heavy usage by multiple users scheduling, resource sharing ,etc.

1.4 Software Applications

2. Real time software:

- Software that monitors/analyzes /controls real world events as they occur is called real time. Elements of real time software include data gathering, component that collects and format information from external environment.
- A analysis component that transforms information as required by the application.
- A control/output component that responds to the external environment, and a monitoring component that coordinates all other components so that real-time response can be maintained.

1.4 Software Applications

3. **Business software**: Business information processing is the largest single software application area. Discrete system is payroll, inventory, accounts etc. Applications in this area restructure existing data in a way that facilitates business operations or management decision making. In addition to that conventional data processing application, business software applications also encompass interactive computing.

4. **Engineering and Scientific software**: This software has been characterized by number crunching algorithm. Application range from astronomy to volcano logy,from automotive stress analysis to space shuttle orbital dynamics. Simulation ,CAD etc

1.4 Software Applications

5. **Embedded software**: Intelligent product have become commonplace in nearly every consumer and industrial market. Embedded software resides in real only memory and is used to control product and system for the consumer and industrial kits.

6. **Personal Computer Software**: The personal computer software market has burgeoned over the past two decades. Word processing , spreadsheets , computer graphics, database management, etc.

1.4 Software Applications

7. Web-based software: The web pages retrieved by a browser are software that incorporates executable instruction eg. Perl ,java and HTML and data may be in the form of hypertext and a variety of visual and audio formats. In essence, the network becomes a massive computer providing an almost unlimited software resource that can be accessed by anyone with a modem.

8. Artificial intelligence software : AI software makes use of nonnumeric algorithms to solve complex problems that are not easy ambltocomputation or straight forward analysis.

1.5 Software Myths

The following are different myths about software:

- If we get behind schedule, we can add more programmers and catch up.

  Reality : Software development is not the mechanical process in which by adding more people it gets completed. The development process is one of the planned process and every member of development team knows the role and task to be completed. Newly added person must have the activity knowledge of development.

- If we decide to outsource the software project to a third party, we can just relax and let that firm build it.

- Project requirement continuously changes, but changes can be easily accommodated because software is flexible.

- **Reality** : Project requirement continuously changes is exactly true but whenever any changes required in software developer should collect all the concern information from relevant source. Changes will be performed without disturbing any other activity or module of the software.

1.5 Software Myths Cont….

- The only deliverable work product for a successful project is the working program.
- Software with more features is better software.
- Once we write the program and get it to work, our job is done.
- Until we get the program running, we have no way of assessing its quality.
- Software engineering will make us create voluminous and unnecessary documentation and will invariably slow us down
- A general statement of objectives is sufficient to begin writing programs; we can fill in the details later.
- We already have a book that’s full of standards and procedures for building software. Won’t that provide my people with everything they need to know?