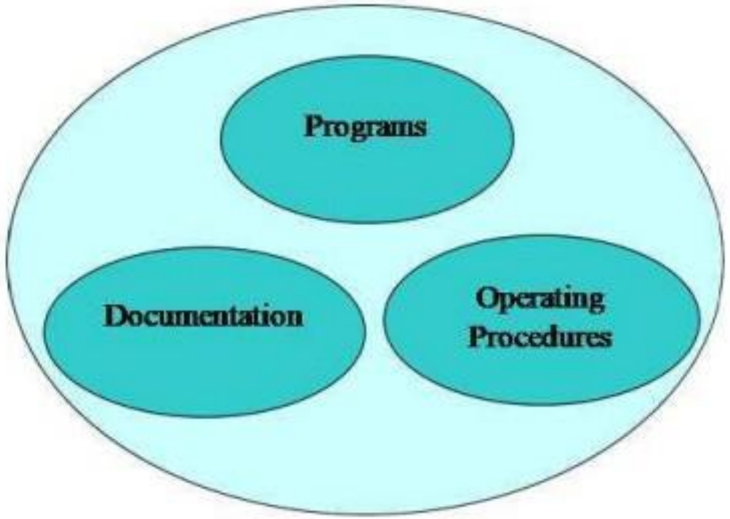
Software Engineering

BCA IV SEMESTER-TU

What is software?

- **Computer programs** and **associated documentation**

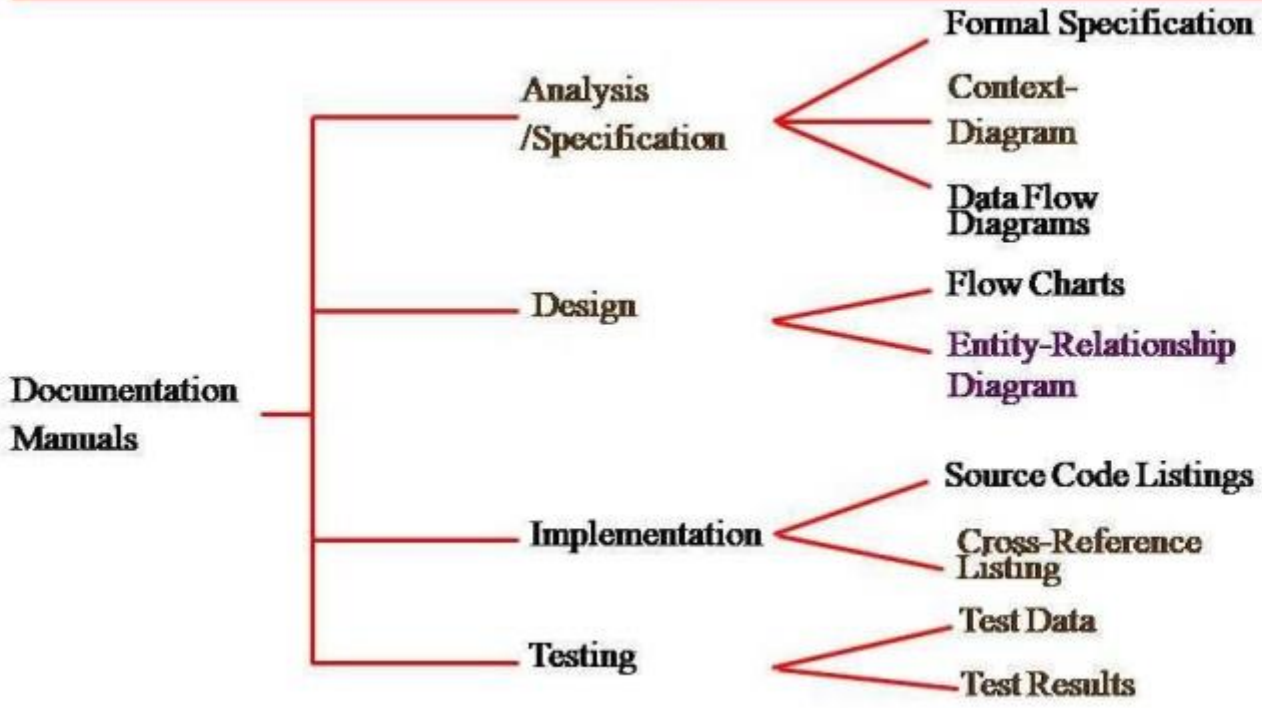
What is software?



Software=Program+Documentation+Operating Procedures

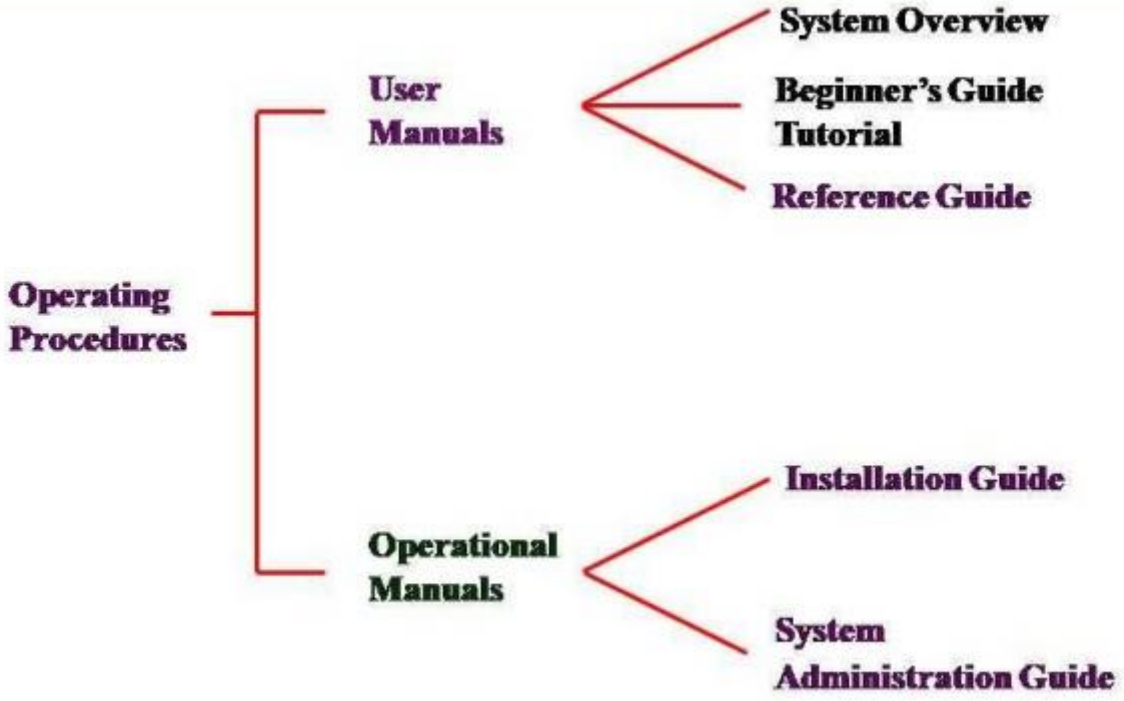
Components of software

Documentation consists of different types of manuals are



List of documentation manuals

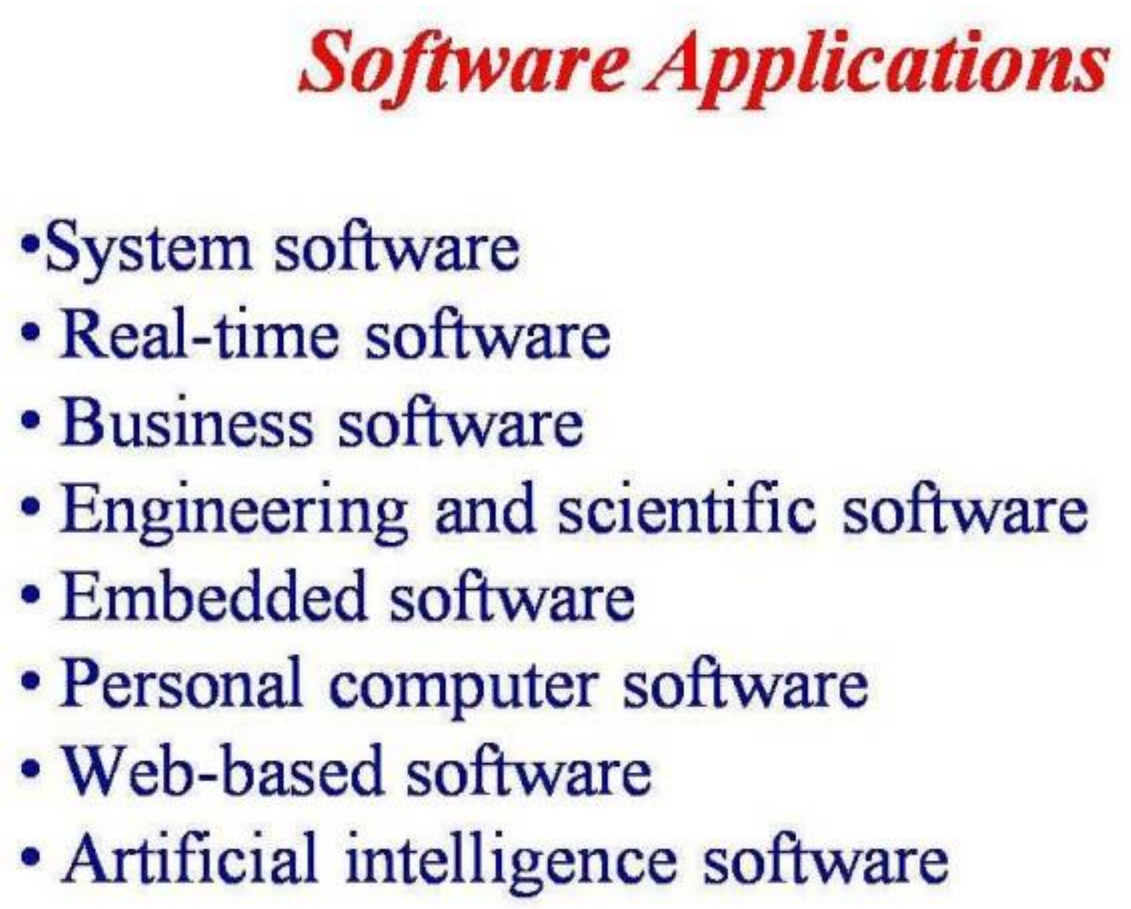
Documentation consists of different types of manuals are



List of operating procedure manuals.

Conclusively... ☺

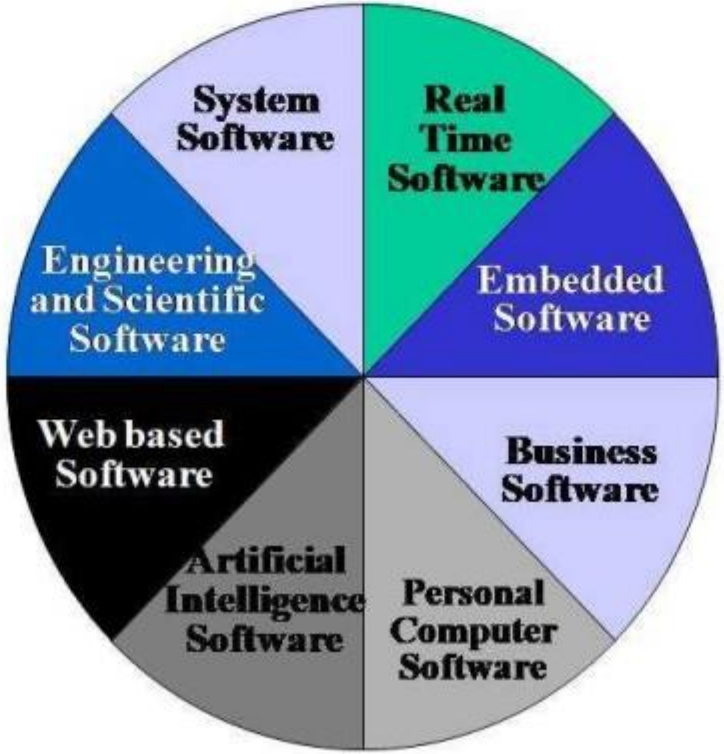
Software is a computer program, its associated documents and configuration data which is needed to make these programs operate correctly. A software system usually consists of a **number of separate programs, configuration files** which are used to set up these programs, **system documentation** which describes the structure of the system and **user documentation** which explains how to use the system.



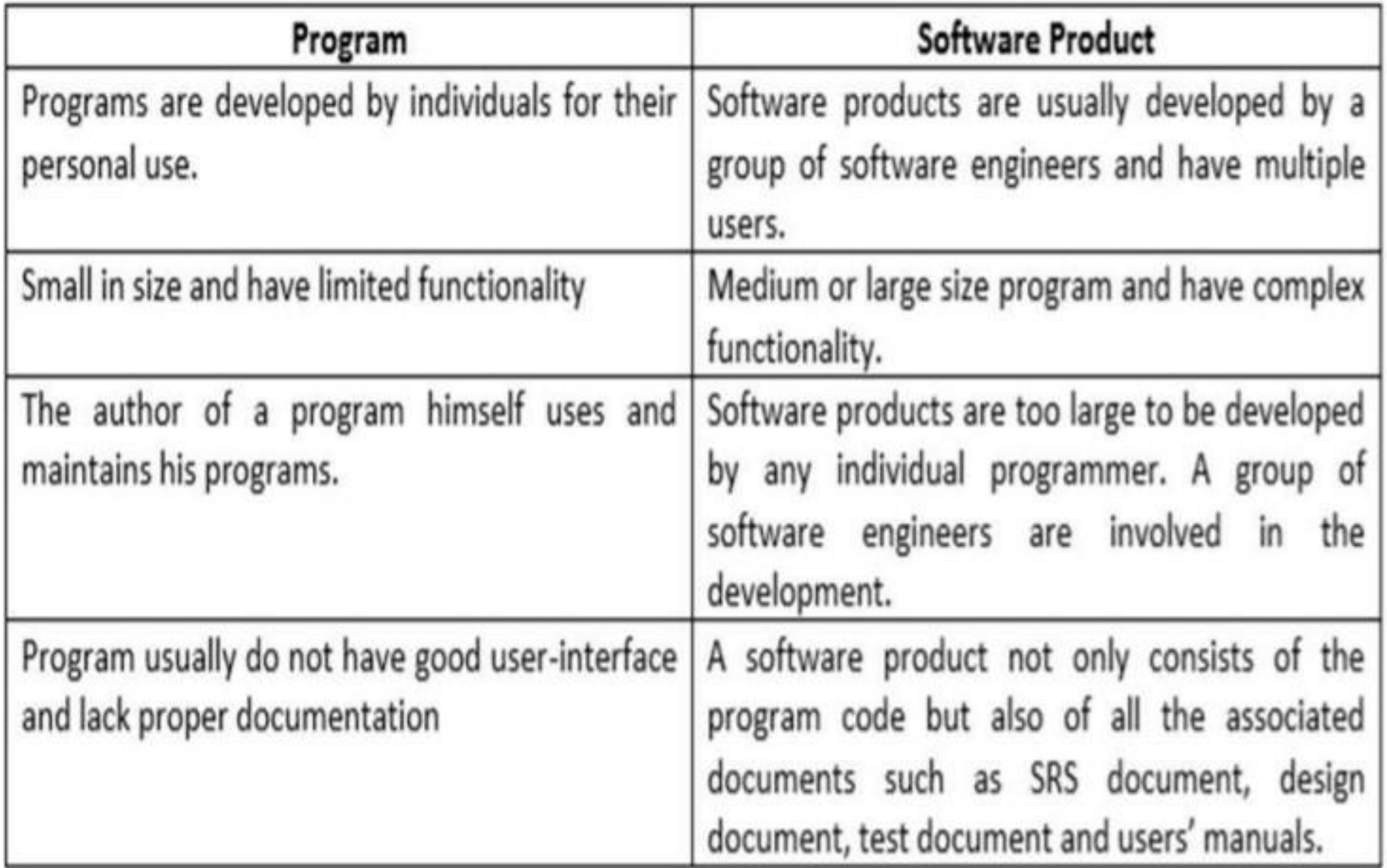
Software Applications

- System software
- Real-time software
- Business software
- Engineering and scientific software
- Embedded software
- Personal computer software
- Web-based software
- Artificial intelligence software

The Changing Nature of Software


System Software
Real Time Software
Engineering and Scientific Software
Embedded Software
Web based Software
Business Software
Artificial Intelligence Software
Personal Computer Software

Program Vs. Software Product



Program	Software Product
Programs are developed by individuals for their personal use.	Software products are usually developed by a group of software engineers and have multiple users.
Small in size and have limited functionality	Medium or large size program and have complex functionality.
The author of a program himself uses and maintains his programs.	Software products are too large to be developed by any individual programmer. A group of software engineers are involved in the development.
Program usually do not have good user-interface and lack proper documentation	A software product not only consists of the program code but also of all the associated documents such as SRS document, design document, test document and users' manuals.

Software products

Software Products may be developed for a particular customer or may be developed for a general market. There are **two** types of software products:

- **Generic products:**

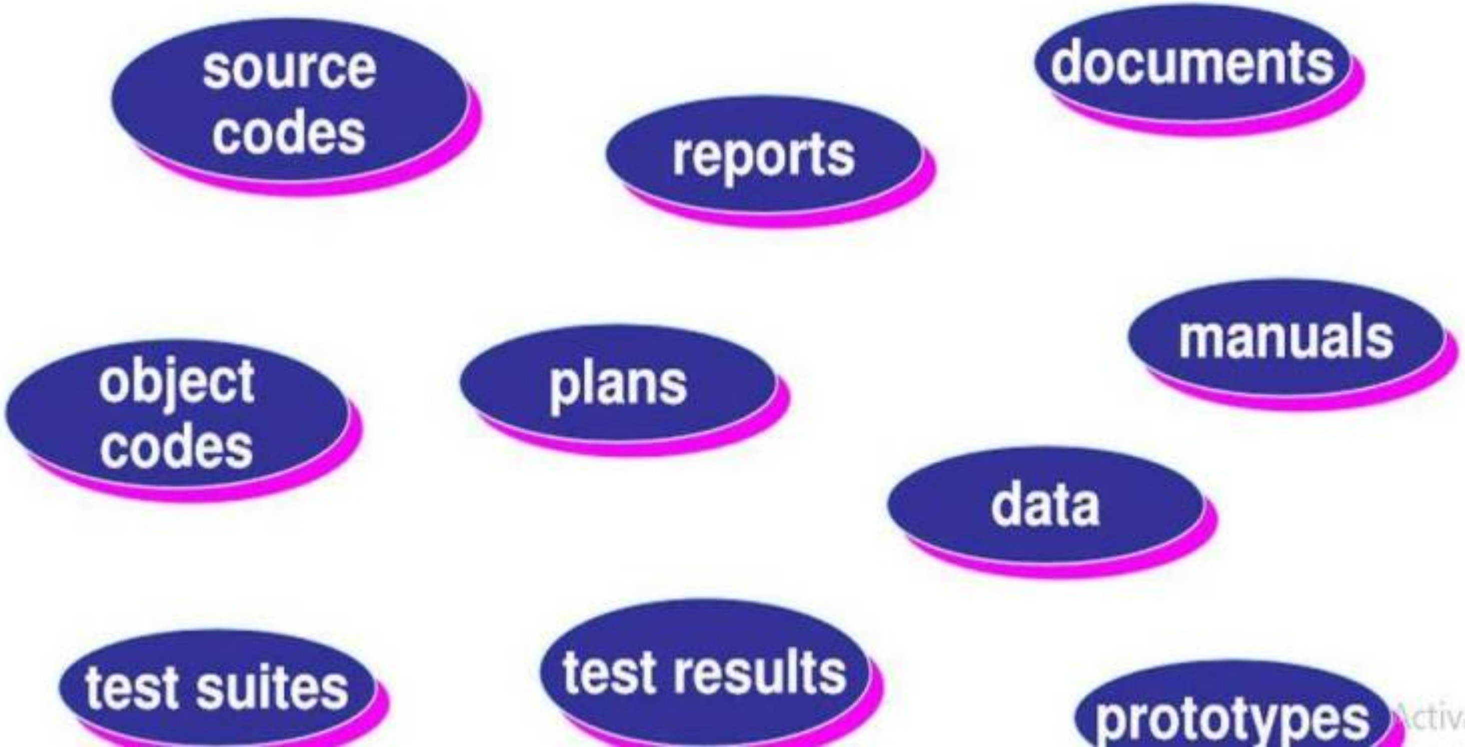
These are stand-alone systems which are produced by a development organization and sold on the open market to a range of different customers. Sometimes they are referred to as shrink-wrapped software. Examples: databases, word processors, drawing packages and project management tools.

- **Bespoke (or customised) products:**

These are systems developed for a single customer according to their specification. Examples: control systems for electronic devices, systems written to support a particular business process, air traffic control systems etc.

Software Product

Software product is a product designated for delivery to the user



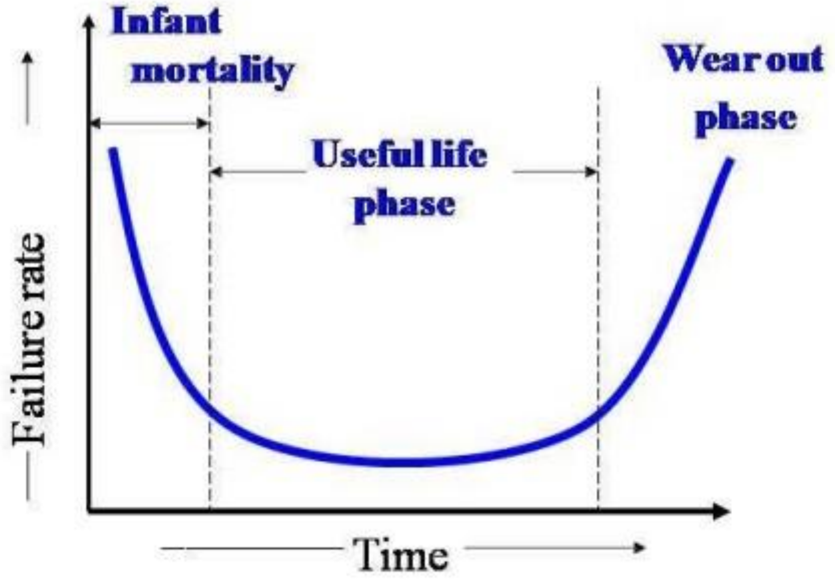
Characteristics of Software

Software is a logical rather than a physical system element. Therefore, software has characteristics that are considerably different than those of hardware. When hardware is built, the human creative process (analysis, design, construction, testing) is ultimately translated into a physical form.

- **Software is developed or engineered; it is not manufactured in the classical sense.**
- **Software does not "wear out" but it does deteriorate.**
- **Most software continues to be custom-built. (Flexible & Reusable)**

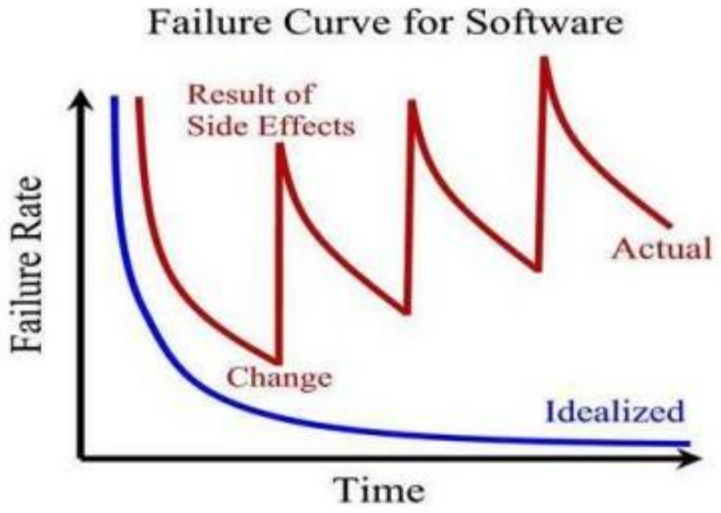
Software Characteristics:

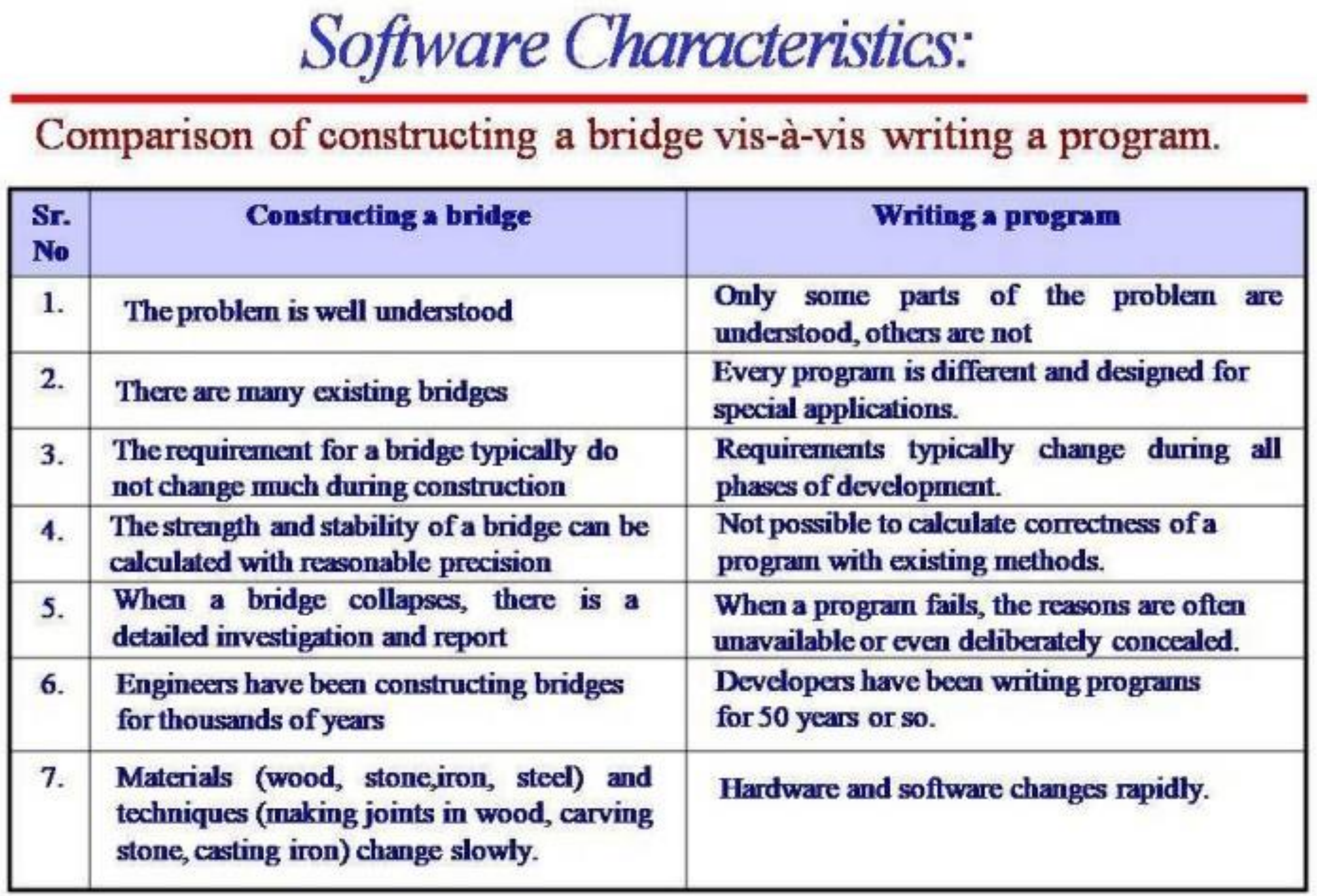
✓ Software does not wear out.



Software Characteristics:

✓ Software does deteriorate





Software Characteristics:

Comparison of constructing a bridge vis-à-vis writing a program.

Sr. No	Constructing a bridge	Writing a program
1.	The problem is well understood	Only some parts of the problem are understood, others are not
2.	There are many existing bridges	Every program is different and designed for special applications.
3.	The requirement for a bridge typically do not change much during construction	Requirements typically change during all phases of development.
4.	The strength and stability of a bridge can be calculated with reasonable precision	Not possible to calculate correctness of a program with existing methods.
5.	When a bridge collapses, there is a detailed investigation and report	When a program fails, the reasons are often unavailable or even deliberately concealed.
6.	Engineers have been constructing bridges for thousands of years	Developers have been writing programs for 50 years or so.
7.	Materials (wood, stone,iron, steel) and techniques (making joints in wood, carving stone, casting iron) change slowly.	Hardware and software changes rapidly.

Attributes of good software

The software should deliver the required functionality and performance to the user and should be maintainable, dependable and usable.

•Maintainability

Software must evolve to meet changing needs of the customers. This is a critical attribute because software change is an inevitable consequence of a changing business environment.

•Dependability

Software must be trustworthy. Dependability has a range of characteristics, including reliability, security and safety. Dependable software should not cause physical or economical damage in the event of system failure.

• Efficiency

It refers to the ability of the software to use system resources in the most effective and efficient manner. software should make effective use of storage space and executive command as per desired timing requirement. Efficiency therefore includes responsiveness, processing time, and memory utilization, etc.

•Usability

Software must be usable by the users for which it was designed, this means it

Other Attributes [Related ☺] of good software

•Functionality

It refers to the degree of performance of the software against its intended purpose.

•Reliability

A set of attribute that bear on capability of software to maintain its level of performance under the given condition for a stated period of time.

•Portability

A set of attribute that bear on the ability of software to be transferred from one environment to another, without or minimum changes.

•Robustness

It refers to the degree to which the software can keep on functioning in spite of being provided with invalid data.

•Integrity

It refers to the degree to which Unauthorized Access to the software data

What is software engineering?

Software engineering is an engineering discipline which is concerned with all aspects of software production.

Software engineers should

- adopt a systematic and organised approach to their work
- use appropriate tools and techniques depending on
  - the problem to be solved,
  - the development constraints and
- use the resources available

What is software engineering?

At the first conference on software engineering in 1968, Fritz Bauer defined software engineering as "The establishment and use of sound engineering principles in order to obtain economically developed software that is reliable and works efficiently on real machines".

Stephen Schach defined the same as "A discipline whose aim is the production of quality software, software that is delivered on time, within budget, and that satisfies its requirements".

Both the definitions are popular and acceptable to majority. However, due to increase in cost of maintaining software, objective is now shifting to produce quality software that is maintainable, delivered on time, within budget, and also satisfies its requirements.

Differences between software engineering and computer science

Computer science is concerned with theory and fundamentals; software engineering is concerned with the practicalities of developing and delivering useful software.

Computer science theories are currently insufficient to act as a complete underpinning for software engineering.

Differences between software engineering and system engineering

System engineering is concerned with all aspects of computer-based systems development including hardware, software and process engineering. Software engineering is part of this process.

System engineers are involved in system specification, architectural design, integration and deployment.

Generic phases of Software Engineering

1. **Definition phase**
2. **Development Phase**
3. **Support Phase**

Definition phase

- It focus on WHAT i.e. to identify what is to be processed
- What functions and performance are desired?
- What design constraints exist?
- What validation criteria are required to define a successful system?
- Identify the key requirements of the systems and the software?

❑ It perform three major tasks as

- Information engineering
- Software project planning
- Requirement analysis

CTD…

Development phase

- It focus on HOW i.e. to define how data are to be structured.
- How function is to be implemented within a software architecture?
- How procedural details are to be implemented?
- How interface are to be characterized?
- How the design will be translated into a programming language
- How testing will be performed?

❑ It perform three specific technical tasks as

- **Software design**
- **Code generation**
- **Software testing**

CTD...

Support

It focus on change associated with error correction, adaptations required as the software environment evolves, and changes due to enhancements brought by the changing customer requirements.

Four types of changes are encountered during the support phases:

- **Correction:** Use of corrective maintenance for any error encountered during support.
- **Adaptation :** Use of adaptive maintenance to accommodate changes in the external environment. Example: Change in the CPU, OS, Business rules, external product characteristics)
- **Enhancement:** Use of perfective maintenance that extends the software beyond its original functional requirements.
- **Prevention :** Use of preventive maintenance by conducting software re-engineering to avoid software deterioration and to correct, adopt or enhance its features

Key Challenges facing software engineering

Software Engineering in the 21st century faces three key challenges, i.e., Coping with legacy systems, coping with increasing diversity and coping with demands for reduced delivery times.

• **Legacy systems**

Old, valuable systems which hold the majority of larger software systems in use must be maintained and updated

•**Heterogeneity**

Systems are distributed and include a mix of hardware and software. The heterogeneity challenge is the challenge of developing techniques to build dependable software, which is flexible enough to cope with this heterogeneity.

• **Delivery**

There is increasing pressure for faster delivery of software unlike time-consuming traditional software engineering techniques. The delivery challenge is the challenge of shortening delivery times for large and complex systems without compromising system quality.

Software Myths

- Software standards provide software engineers with all the guidance they need. The reality is the standards may be outdated and rarely referred to.

- People with modern computers have all the software development tools. The reality is that CASE tools are more important than hardware to producing high quality software, yet they are rarely used effectively.

- Adding people is a good way to catch up when a project is behind schedule. The reality is that adding people only helps the project schedule when is it done in a planned, well-coordinated manner.

- Giving software projects to outside parties to develop solves software project management problems. The reality is people who can't manage internal software development problems will struggle to manage or control the external development of software too.

Myth Ctd…

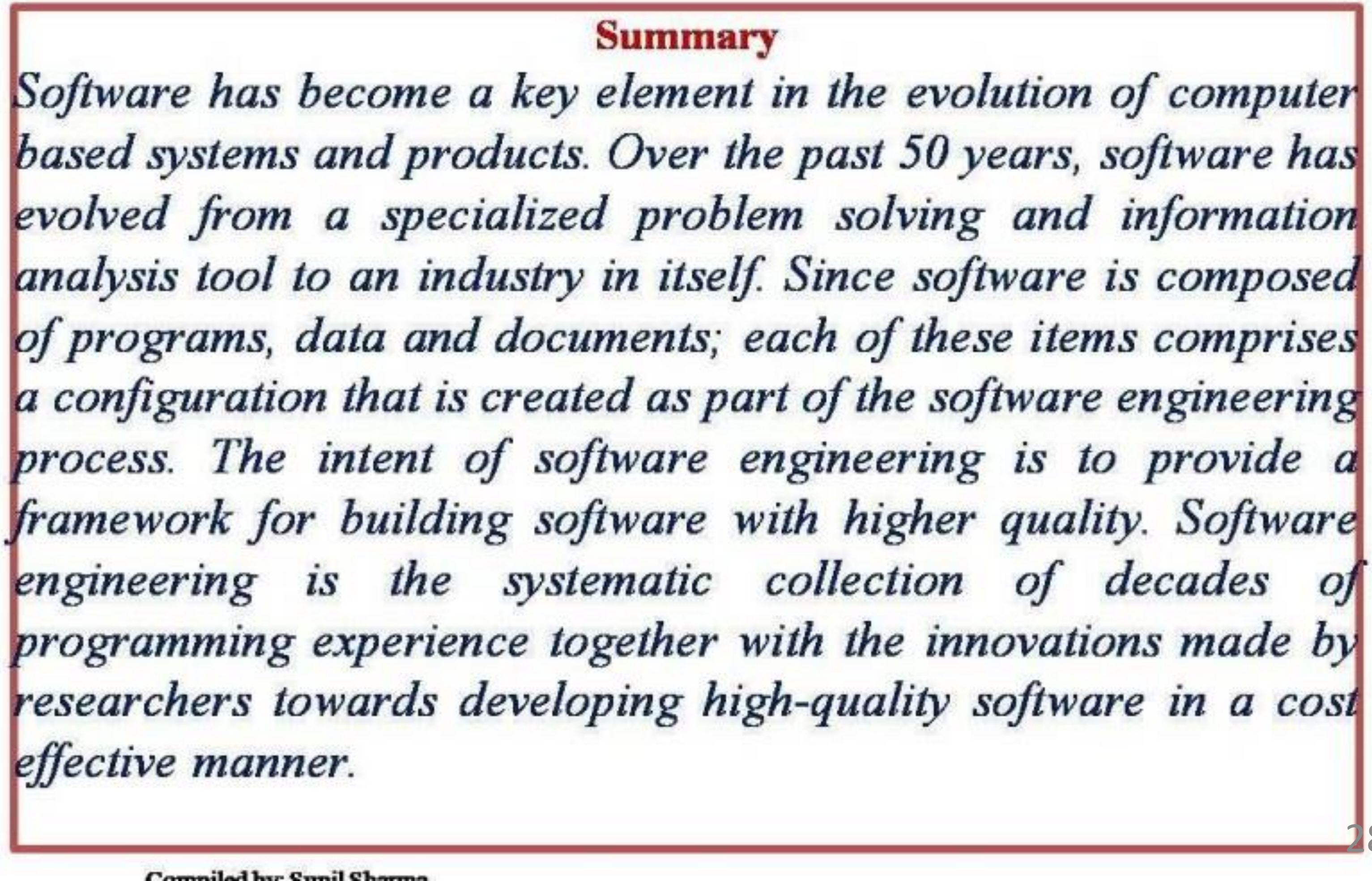
- A general statement of objectives from the customer is all that is needed to begin a software project. The reality is without constant communication between the customer and the developers it is impossible to build a software product that meets the customer's real needs.

- Project requirements change continually and change is easy to accommodate in the software design. The reality is that every change has far-reaching and unexpected consequences. Changes to software requirements must be managed very carefully to keep a software project on time and under budget.
- Once a program is written, the software engineer's work is finished. The reality is that maintaining a piece of software is never done, until the software product is retired from service.
- There is no way to assess the quality of a piece of software until it is actually running on some machine. The reality is that one of the most effective quality assurance practices (formal technical reviews) can be applied to any software design product and can serve as a quality filter very early in the product life cycle.

Evolving role of software

Software takes on a dual role. ***It is a product and the vehicle for delivering a product***. As a product, it delivers the computing potential embodied by computer hardware or, a network of computers that are accessible by local hardware. ***Software is an information transformer-producing, managing, acquiring, modifying, displaying, or transmitting information.*** As the vehicle used to deliver the product, software acts as the basis for the control of the computer (operating systems), the communication of information(networks), and the creation and control of other programs (software tools and environments).

Software delivers the most important product of our time- information. Software transforms personal data (e.g., an individual's financial transactions) so that the data can be more useful in a local context; it manages business information to enhance competitiveness; it provides a gateway to worldwide information networks (Internet) and provides the means for acquiring information in all of its form.



Summary

Software has become a key element in the evolution of computer based systems and products. Over the past 50 years, software has evolved from a specialized problem solving and information analysis tool to an industry in itself. Since software is composed of programs, data and documents; each of these items comprises a configuration that is created as part of the software engineering process. The intent of software engineering is to provide a framework for building software with higher quality. Software engineering is the systematic collection of decades of programming experience together with the innovations made by researchers towards developing high-quality software in a cost effective manner.

Compiled by: Sunil Sharma

Software Myths (Management Perspectives)

Management may be confident about good standards and clear procedures of the company.

But the taste of any food item
is in the eating;
not in the Recipe !

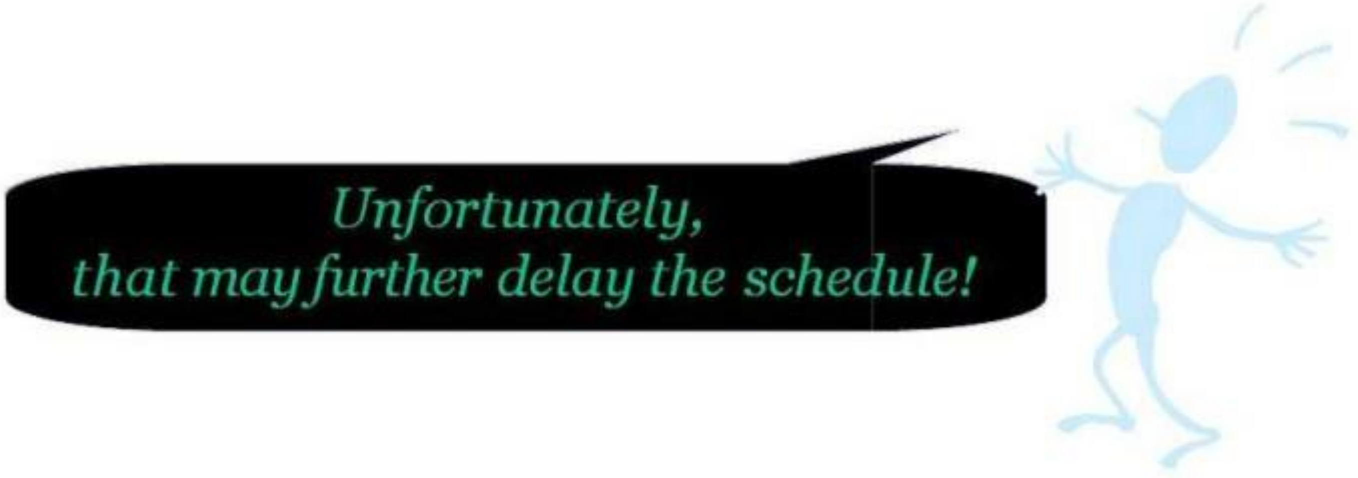
Software Myths (Management Perspectives)

Company has latest computers and state-of-the-art software tools, so we shouldn't worry about the quality of the product.

The infrastructure is only one of the several factors that determine the quality of the product!

Software Myths (Management Perspectives)

Addition of more software specialists, those with higher skills and longer experience may bring the schedule back on the track!



Software Myths (Management Perspectives)

Software is easy to change

The reality is totally different.

Software Myths (Management Perspectives)

Computers provide greater reliability than the devices they replace

This is not always true.

Software Myths (Customer Perspectives)

A general statement of objectives is sufficient to get started with the development of software. Missing/vague requirements can easily be incorporated/detailed out as they get concretized.

If we do so, we are heading towards a disaster.

Software Myths (Customer Perspectives)

Software with more features is better software

Software can work right the first time

Both are only myths!

Software Myths (Developer Perspectives)

Once the software is demonstrated, the job is done.

Usually, the problems just begin!

Software Myths (Developer Perspectives)

Software quality can not be assessed before testing.

However, quality assessment techniques should be used through out the software development life cycle.

Software Myths (Developer Perspectives)

The only deliverable for a software development project is the tested code.

Tested code is only one of the deliverable!

Software Myths (Developer Perspectives)

Aim is to develop working programs

Those days are over. Now objective is to develop good quality maintainable programs!