Software Engineering
B.Tech CS/Year-III Sem-VI

Term: 2025-2026

UNIT-1

Text Books:

1.Software Engineering, A practitioner's approach Roger s. Pressman 6th edition McGraw-Hill

2.Software Engineering, Pankaj Jalote, Wiley.

3.Software Engineering, New Age International, K.K Agarwal and Yogesh Singh, 3rd edition

4.Fundamentals of Software Engineering, Rajib Mall, PHI Publication

Software Crisis

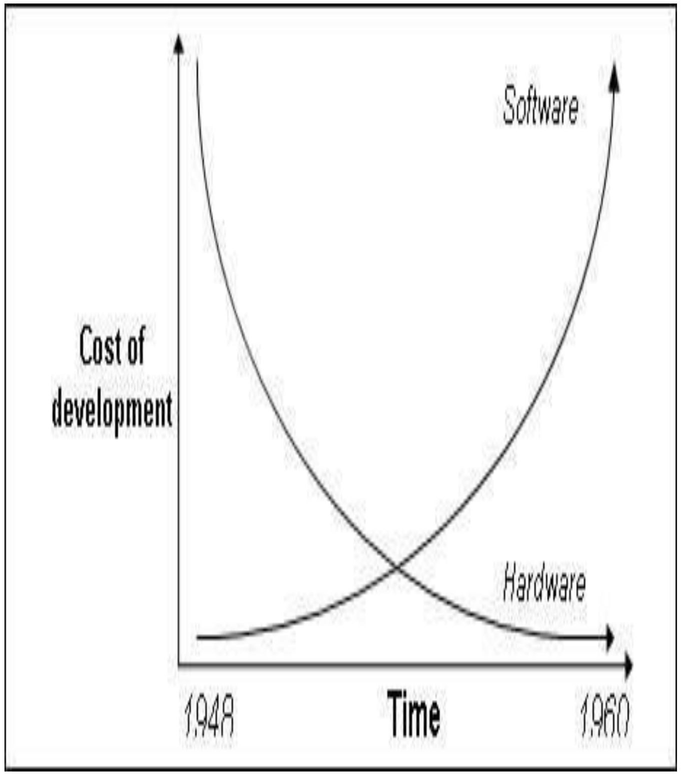
A software crisis is a mismatch between what software can deliver and the capacities of computer systems, as well as expectations of their users.

This became a growing problem in the 20th century

Pressure to produce complex, advanced code can be a significant contributor to a software crisis.

Software Crisis

By the end of the 1960s, hardware costs had fallen exponentially, and were continuing to do so, while the cost of software development was rising at a similar rate



Factors contributing to the Software Crisis

- ❖ Late delivery, over budget,
- ❖ Product does not meet specified requirements
- ❖ Lack of adequate training in software engineering
- ❖ Low productivity improvements.

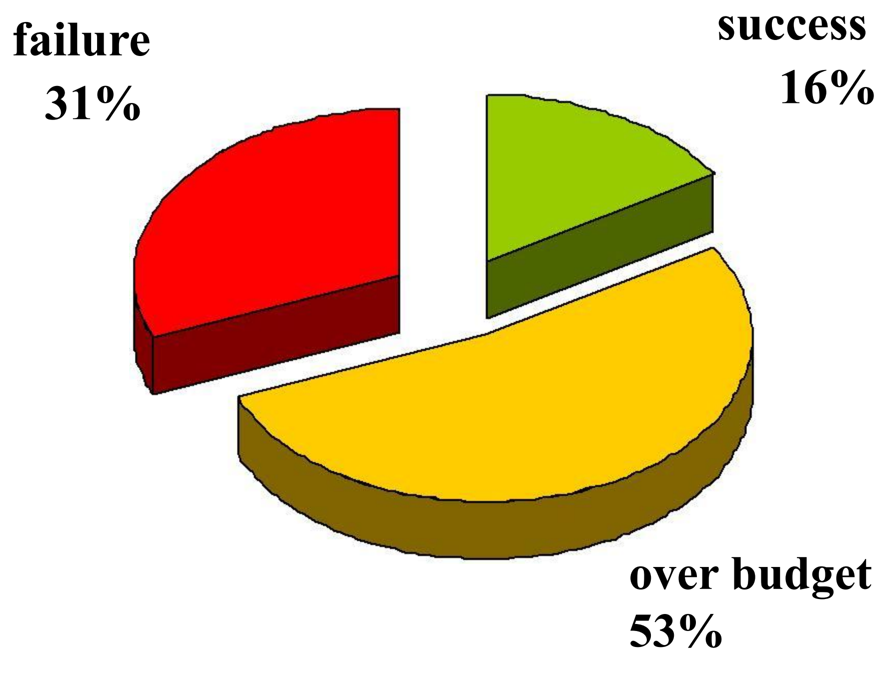
Software Crisis

As per the IBM report,

"31%of the project get cancelled before they are completed, 53% over-run their cost estimates by an average of 189% and for every 100 projects, there are 94 restarts".

Software Crisis

Software industry is in Crisis!



Some Software Failures

Ariane 5

It took the European Space Agency **10 years and $7 billion** to produce Ariane 5, a giant rocket capable of hurling a pair of three-ton satellites into orbit with each launch and intended to give Europe overwhelming supremacy in the commercial space business.

The rocket was destroyed after 39 seconds of its launch, at an altitude of two and a half miles along with its payload of four expensive and uninsured scientific satellites.

Some Software Failures

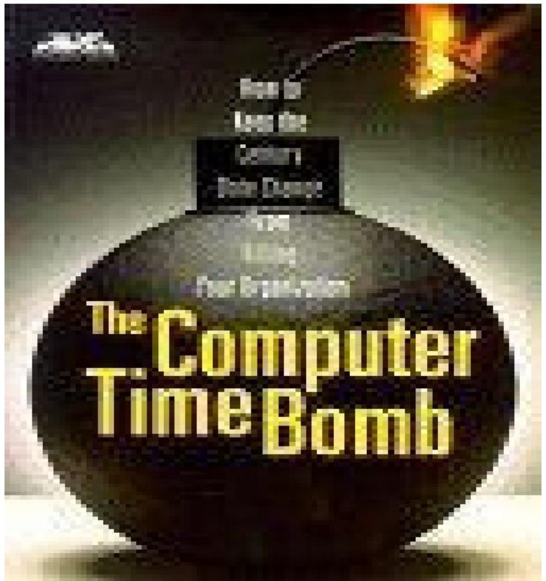
When the guidance system’s own computer tried to convert one piece of data the sideways velocity of the rocket from a 64 bit format to a 16 bit format; the number was too big, and an overflow error resulted after 36.7 seconds. When the guidance system shutdown, it passed control to an identical, redundant unit, which was there to provide backup in case of just such a failure. Unfortunately, the second unit, which had failed in the identical manner a few milliseconds before.

Some Software Failures

Y2K Problem

It was simply the ignorance about the adequacy or otherwise of using only last two digits of the year.

The 4-digit date format, like 1964, was shortened to 2-digit format, like 64.



Some Software Failures

<u>The Patriot Missile</u>

- First time used in Gulf war
- Used as a defense from Iraqi Scud missiles
- Failed several times including one that killed 28 US soldiers in Dhahran, Saudi Arabia.

Reasons:

A small timing error in the system's clock accumulated to the point that after 14 hours, the tracking system was no longer accurate. In the Dhahran attack, the system had been operating for more than 100 hours.

Some Software Failures

Financial Software

Many companies have experienced failures in their accounting system due to faults in the software itself. The failures range from producing the wrong information to the whole system crashing.

Some Software Failures

Windows XP

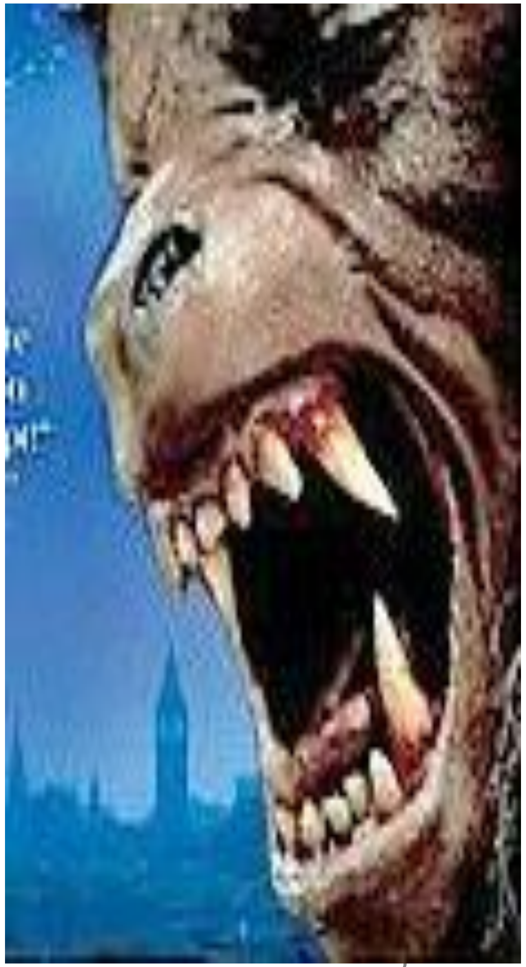
- ❖ Microsoft released Windows XP on October 25, 2001.
- ❖ On the same day company posted 18 MB of compatibility patches on the website for bug fixes, compatibility updates, and enhancements.
- ❖ Two patches fixed important security holes.

“No Silver Bullet”

The hardware cost continues to decline drastically.

However, there are desperate cries for a silver bullet something to make software costs drop as rapidly as computer hardware costs do.

But as we look to the horizon of a decade, we see no silver bullet. There is no single development, either in technology or in management technique.



“No Silver Bullet”

The hard part of building software is the specification, design and testing of this conceptual construct, not the labour of representing it and testing the correctness of representation.

We still make syntax errors, to be sure, but they are trivial as compared to the conceptual errors (logic errors) in most systems. That is why, building software is always hard and there is inherently no silver bullet.

“No Silver Bullet”

The blame for software bugs belongs to:

- Software companies
- Software developers
- Legal system
- Universities

Problem Domain

The Problem Domain is the software that solves some problems of some users where larger systems or businesses may depend on the software, and where problems in the software can lead to significant direct and indirect loss.

Software

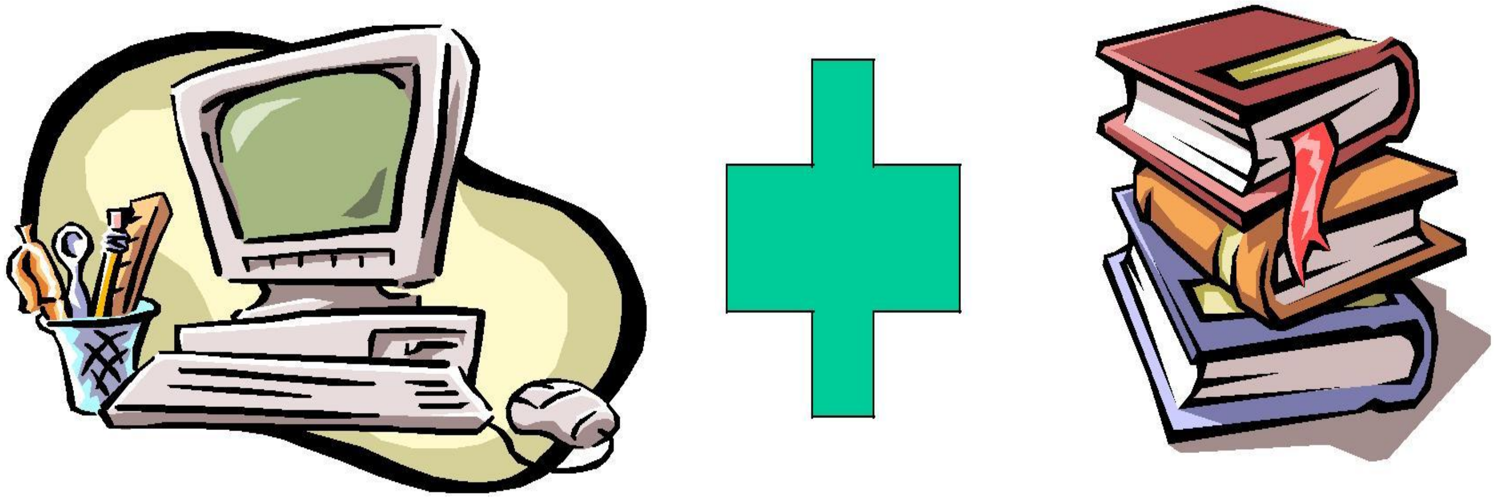
Misconception : Software is Program

Various differences between Program and Software are given in the tabular form as follows:

	Program	Software
1	Programs are developed by individuals for their personal use.	A software is usually developed by a group of engineers working in a team.
2	Usually small in size	Usually large in size
3	Single user	Large number of users
4	Lacks proper documentation	Good documentation support
5	Lack of user interface	Good user interface
6	Have limited functionality	Exhibit more functionality

Software

Software is the collection of programs, documentation, and operating procedures by which computers can be made useful to people .

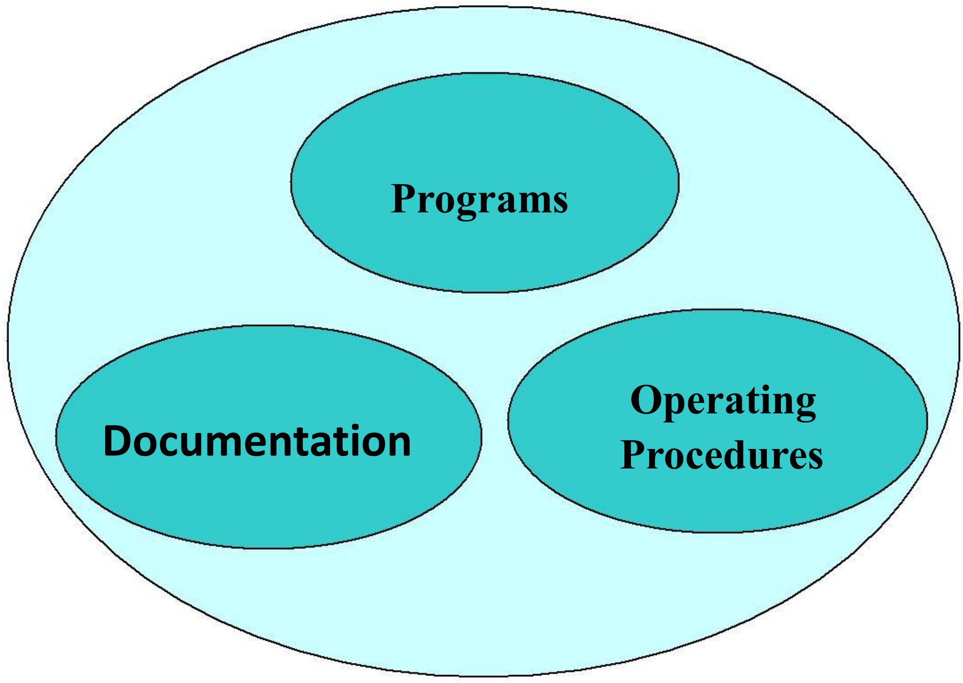


Software Components

Program : source code, object code

Operating Procedures : user manual, operational manual. (instructions / scripts to set up and use the program ; instructions on how to treat failures ; instructions/scripts on how to test the program

Documentation : requirement specification document, design document, test document etc.



Components of a software system

Software Product

Software products may be developed for a particular customer or may be developed for a general market

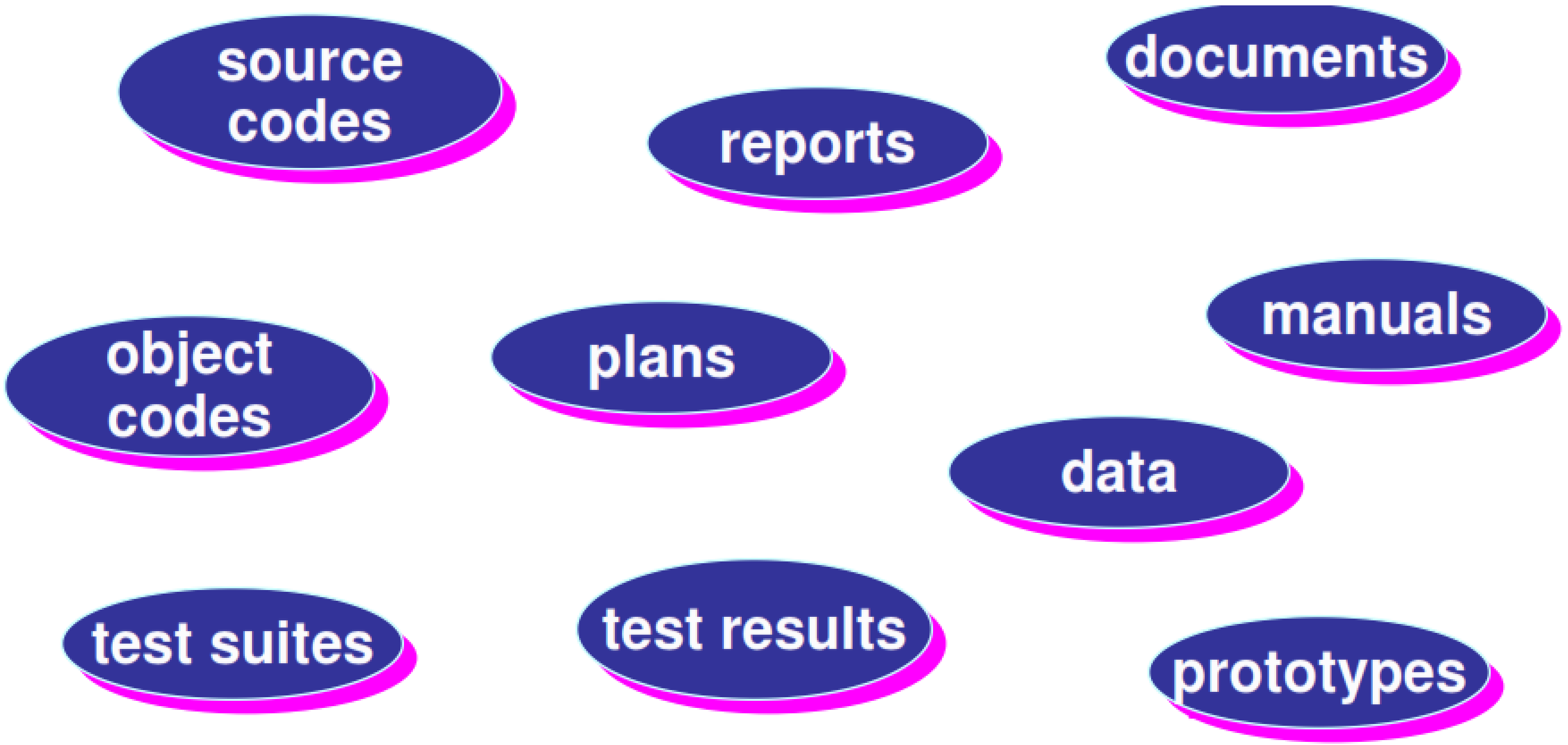
Software products may be

–**Generic Products** - developed to be sold to a range of different customers

–**Customized Products** - developed for a single customer according to their specification

Software Product

Software product is a product designated for delivery to the user



Difference between Student Software & Industrial Strength Software

Student Software is not used for solving any real problem of any organization.

An Industrial Strength Software is used by the client's organization for operating some part of **business** e.g, inventories, finances etc.

This kind of software system needs to be of **high quality.**

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

Software Engineering

It is a systematic, disciplined, cost effective techniques for software development. It’s a engineering approach to develop a software.

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.**

Goal of Software Engineering

- To improve quality of software products.
- To increase the productivity, and
- To give job satisfaction to the software engineers.

Software Characteristics

1. **Software is not manufactured, it is developed.**

2. **Reusability of components.**

3. **Software is flexible.**

Software Characteristics

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	When a program fails, the reasons are often unavailable or even deliberately

SE Challenges

The problem of producing software to satisfy user needs drives the approaches used in SE.

Important Factors that affect the SE approaches selected to solve the problem are-

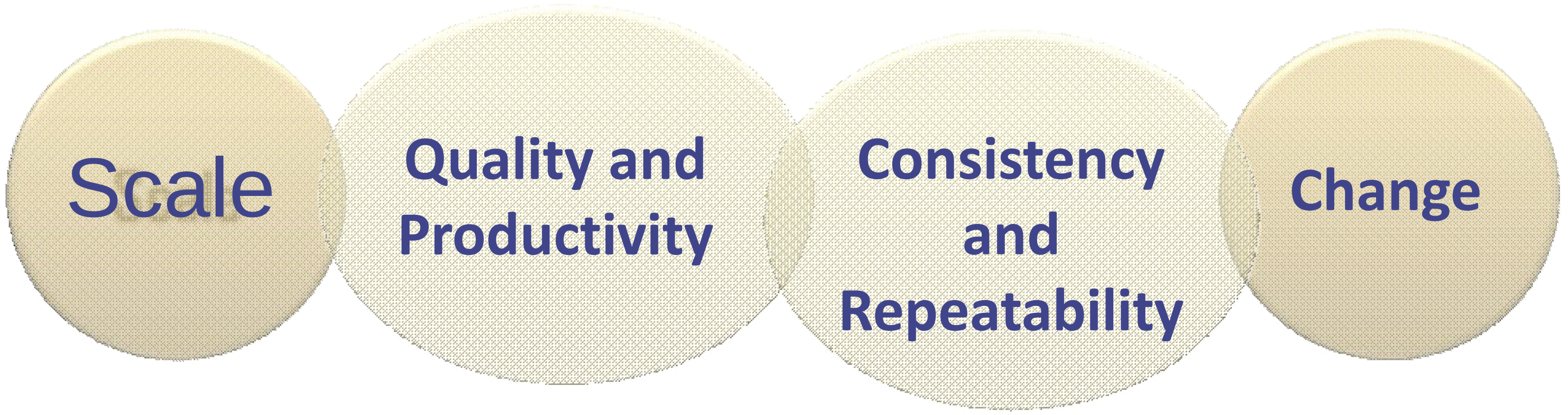


Fig: Primary Challenges for Software Engineering

Scale

- ❖ SE must deal with problem of scale
  - methods for solving small problems do not scale up for large problems
  - industrial strength S/W problems tend to be large

  e.g, Counting the number of people in a room vs taking a census

- ❖ Two clear dimensions in this
  1. **engineering methods**
  2. **project management**

- ❖ For small, both can be informal or ad-hoc, for large both have to be formalized

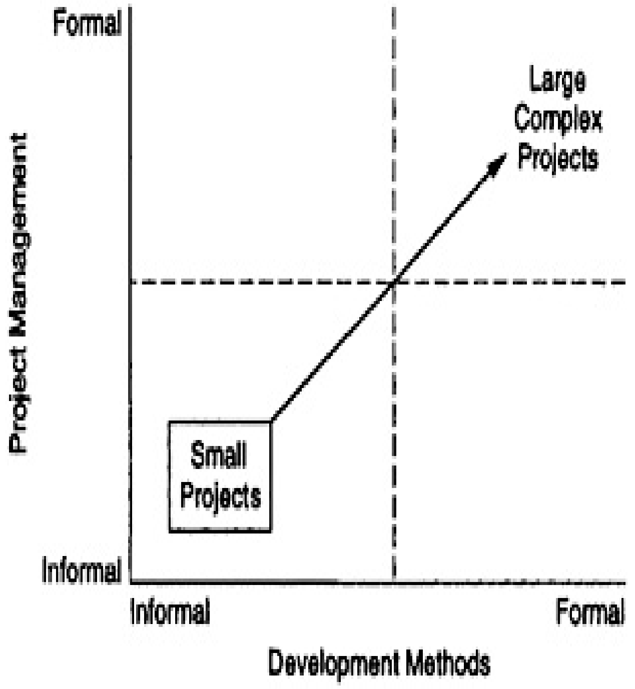


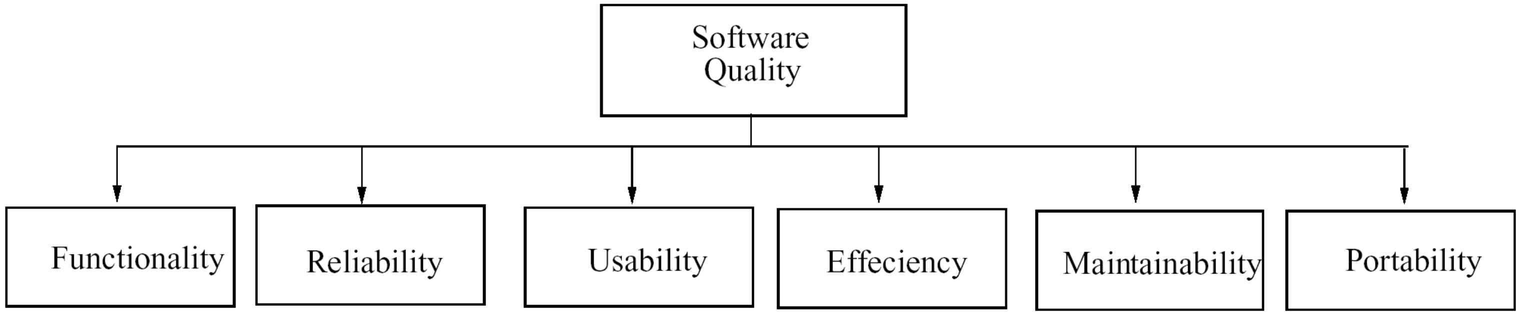
Fig: The problem of scale.

Productivity

- An engineering project is driven by **cost** and **schedule.**
- **Cost:** In s/w, cost is mainly manpower cost; hence, it is measured in person-months
- **Schedule** is in months/weeks – very important in business context
- Productivity captures both Cost and Schedule
    - If P is higher, cost is lower
    - If P is higher, time taken can be lesser
- Approaches used by SE must deliver high Productivity

Quality

- Quality is the other major driving factor
- Developing high Quality S/W is a basic goal
- Quality of S/W is harder to define
- Approaches used should produce a high Quality software
- ISO standard has six software quality attributes -



Consistency & Repeatability

- Sometimes a group can deliver one good software system, but not a second
- **Key SE challenge:** *how to ensure that success can be repeated ?*
- SE wants methods that can consistently produce high Quality SW with high Productivity
- A SW org, wants to deliver high Q&P consistently across projects
- Frameworks like
  - Inter. Org. for Standardization (ISO-9001) and
  - Capability Maturity Model (CMM)

  focus on this aspect

Change

In today’s world change in business is very rapid.

It is therefore expected that software supporting businesses should respond to the changes in order to achieve the task is meant to do. Rapid change has a special impact on software.

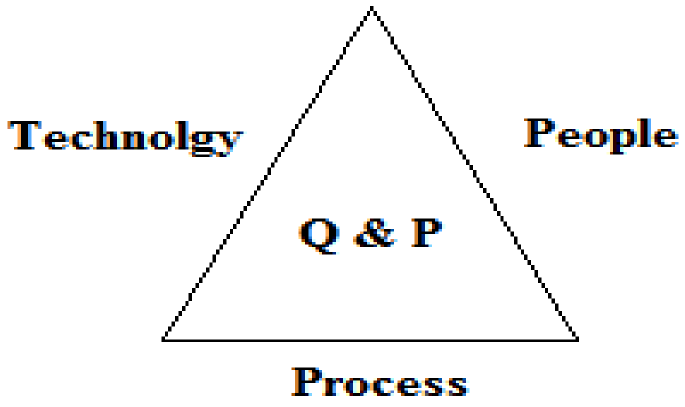
Therefore, one challenge for software engineering is to accommodate and embrace change.

Software Engineering Approach

• Q&P are the basic objectives to be achieved **Consistently for large scale problems**

• Q&P achieved during a project depend upon three main forces - people, processes, and technology,

often called the **IRON TRIANGLE.**



SE Methodology

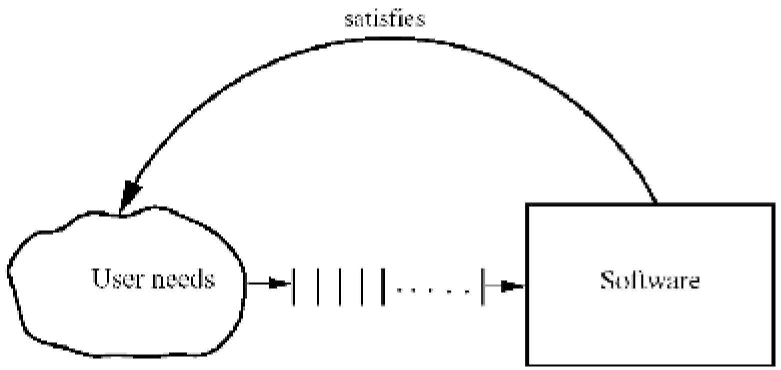
- SE focuses mostly on processes for achieving the goals
- Process must be systematic
- SE separates process for developing s/w from the developed product (i.e the s/w)
- **Premise:** Process largely determines Q&P, hence suitable processes will lead to high Q&P

Software Process

Software Processes include those activities, methods, practices and transformations that are used to create and maintain software products.

The software process is the way in which we produce software.

Process is what takes us from user needs to the software that satisfies the needs.



Process Specification

- Process is generally a set of phases
- Each phase performs a well-defined task and generally produces an output
- Intermediate outputs – work product
- At top level, typically few phases in a process
- How to perform a particular phase – methodologies have been proposed

ETVX Specification

- ETVX approach to specify a step
  - ❑ Entry criteria: what conditions must be satisfied for initiating this phase
  - ❑Task: what is to be done in this phase
  - ❑Verification: the checks done on the outputs of this phase
  - ❑eXit criteria: when can this phase be considered done successfully
- A phase also produces information for management

Characteristics of Software Process

- Is any process suitable to use?
- As a process may be used by many projects, it needs characteristic beyond satisfying the project goals.

1. Predictability
2. Support Testability and Maintainability
3. Support Change
4. Early Defect Removal
5. Process Improvement and Feedback

Software Engineering Process

- ❖ A **Software engineering process** is the **model chosen for managing the creation of software from initial customer inception to the release of the finished product**. The chosen process usually involves techniques such as -
  - Analysis,
  - Design,
  - Coding,
  - Testing and
  - Maintenance

- ❖ In this engineering process, we trying to prevent the error at the time of software development.

Software Development Life Cycle (SDLC)

- Software-development life-cycle is used to facilitate the development of a large software product in a systematic, well-defined, and cost-effective way.

- An information system goes through a series of phases from conception to implementation. This process is called the Software-Development Life-Cycle.

- Various reasons for using a life-cycle model include:

– Helps to understand the entire process

– Enforces a structured approach to development

– Enables planning of resources in advance

– Enables subsequent controls of them

– Aids management to track progress of the system

Software Development Life Cycle (SDLC)

- **Feasibility study: -** The main aim of feasibility study is to **determine whether it would be financially and technically feasible to develop the product**.

- **Requirements analysis and specification: -** The aim of the requirements analysis and specification phase is to understand the exact requirements of the customer and to document them properly. This phase consists of two distinct activities, namely Requirements gathering and analysis, this phase ends with the preparation of **Software requirement Specification** (SRS)

- **Design: -** The goal of the design phase is **to transform the requirements specified in the SRS document into a structure that is suitable for implementation in some programming language** . Design specification Document is outcome of this phase.

- **Coding and unit testing:-**The purpose of the coding and unit testing phase (sometimes called the implementation phase) of software development is to **translate the software design into source code**. Each component of the design is implemented as a program module. The end-product of this phase is a set of program modules that have been individually tested. Code Listings are generated after this phase.

Software Development Life Cycle (SDLC)

- **Integration and system testing: -** Integration of different modules is undertaken once they have been coded and unit tested During each integration step**, the partially integrated system is tested and a set of previously planned modules are added to it**. Finally, when all the modules have been successfully integrated and tested, system testing is carried out. **The goal of system testing is to ensure that the developed system conforms to its requirements laid out in the SRS document**. Test Reports are generated after this phase.

- **Maintenance: -** Maintenance of a typical software product requires much more than the effort necessary to develop the product itself. Many studies carried out in the past confirm this and indicate that the relative effort of development of a typical software product to its maintenance effort is roughly in the 40:60 ratio. This phase continues till the software is in use.

Software Development Life Cycle (SDLC)

- Different software life cycle models have been proposed so far. Each of them has some advantages as well as some disadvantages. A few important and commonly used life cycle models are as follows:

  1. Classical Waterfall Model
  2. Iterative Model
  3. Prototyping Model
  4. Evolutionary Model
  5. Spiral Model
  6. Rapid Application Development model (RAD)