Software Engineering

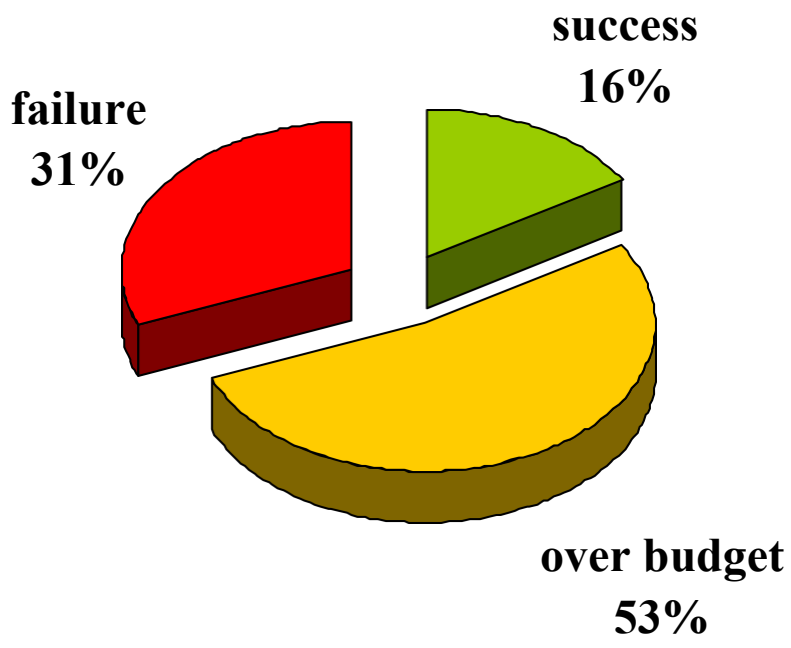
Why Software Engineering ?

- Change in nature & complexity of software
- Concept of one "guru" is over
- We all want improvement

Ready for change

The Evolving Role of Software

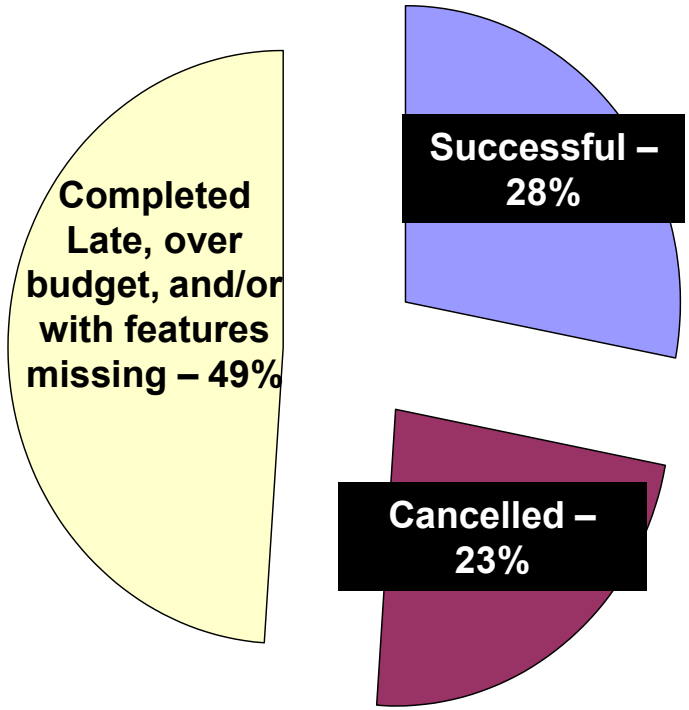
- Software industry is in Crisis!



Source: The Standish Group International, Inc. (CHAOS research)

The Evolving Role of Software

This is the SORRY state of Software Engineering Today!

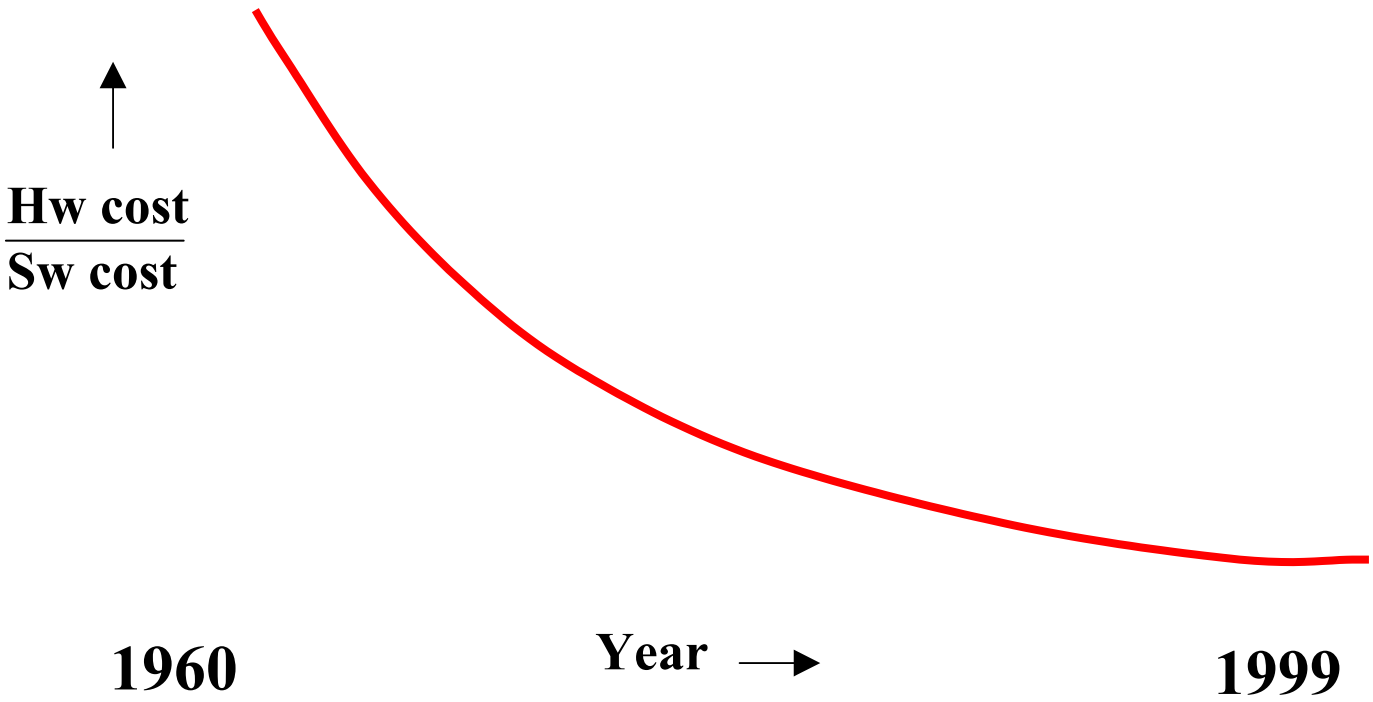


- **Data on 28,000 projects completed in 2000**

The Evolving Role of Software

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”.

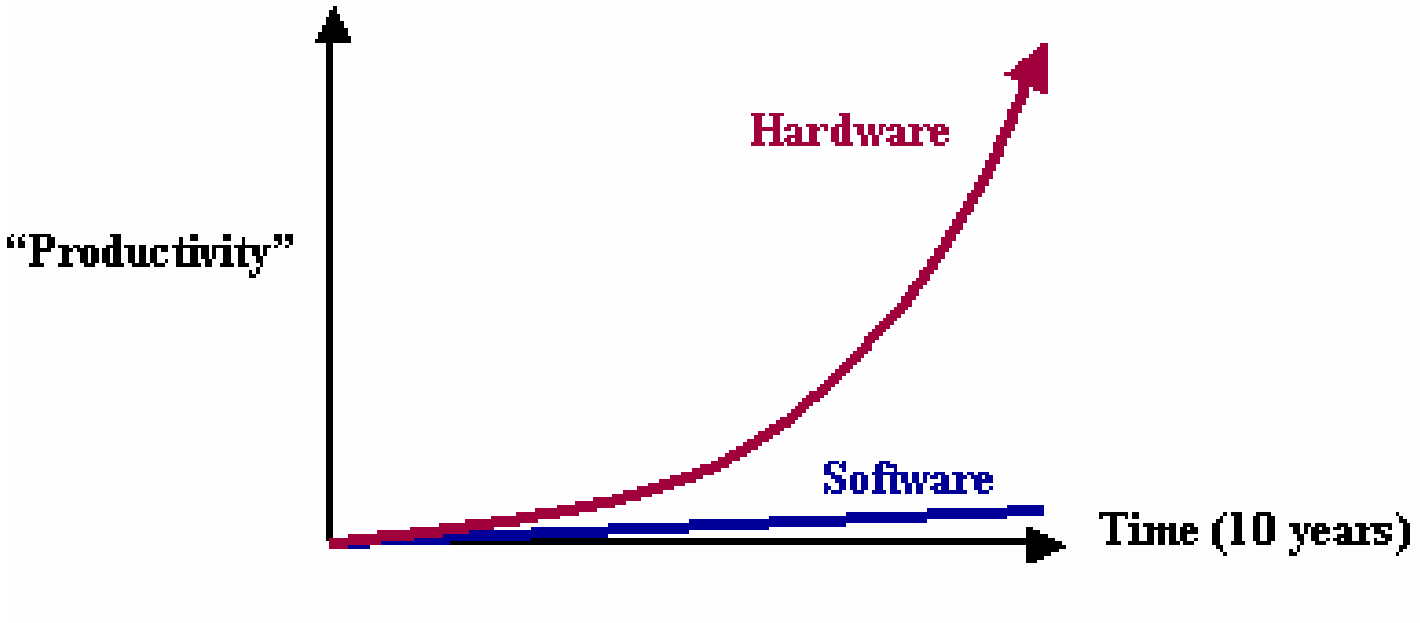
The Evolving Role of Software



Relative Cost of Hardware and Software

The Evolving Role of Software

- Unlike Hardware
  - Moore's law: processor speed/memory capacity doubles every two years



The Evolving Role of Software

***Managers* and *Technical Persons* are asked:**

- ✓ Why does it take so long to get the program finished?
- ✓ Why are costs so high?
- ✓ Why can not we find all errors before release?
- ✓ Why do we have difficulty in measuring progress of software development?

Factors Contributing to the Software Crisis

- Larger problems,
- Lack of adequate training in software engineering,
- Increasing skill shortage,
- Low productivity improvements.

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

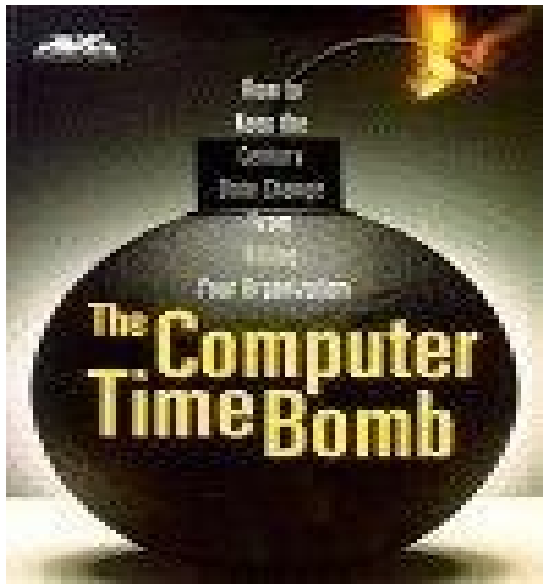
When the guidance system's own computer tried to convert one piece of data the sideways velocity of the rocket from a 64 bit format to a 16 bit format; the number was too big, and an overflow error resulted after 36.7 seconds. When the guidance system shutdown, it passed control to an identical, redundant 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

The Patriot Missile

- 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

The Space Shuttle

Part of an abort scenario for the Shuttle requires fuel dumps to lighten the spacecraft. It was during the second of these dumps that a (software) crash occurred.

...the fuel management module, which had performed one dump and successfully exited, restarted when recalled for the second fuel dump...

Some Software failures

A simple fix took care of the problem…but the programmers decided to see if they could come up with a systematic way to eliminate these generic sorts of bugs in the future. A random group of programmers applied this system to the fuel dump module and other modules.

Seventeen additional, previously unknown problems surfaced!

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.

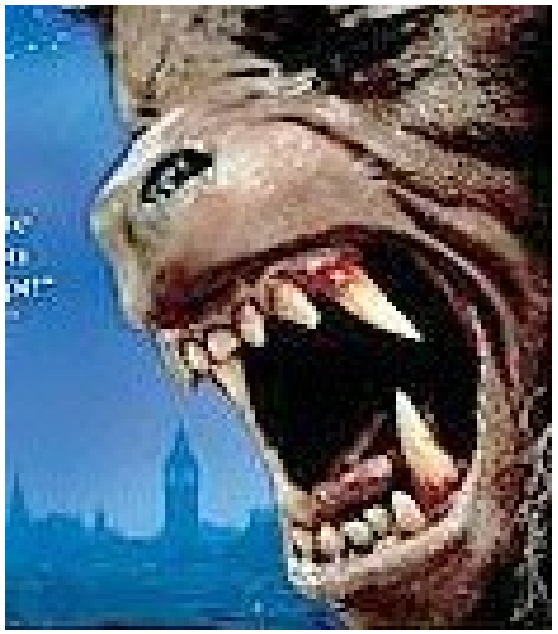
This is **Software Engineering.**

"No SilverBullet"

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, that by itself promises even one order of magnitude improvement in productivity, in reliability and in simplicity.



"No SilverBullet"

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.

While there is no royal road, there is a path forward.

Is reusability (and open source) the new silver bullet?

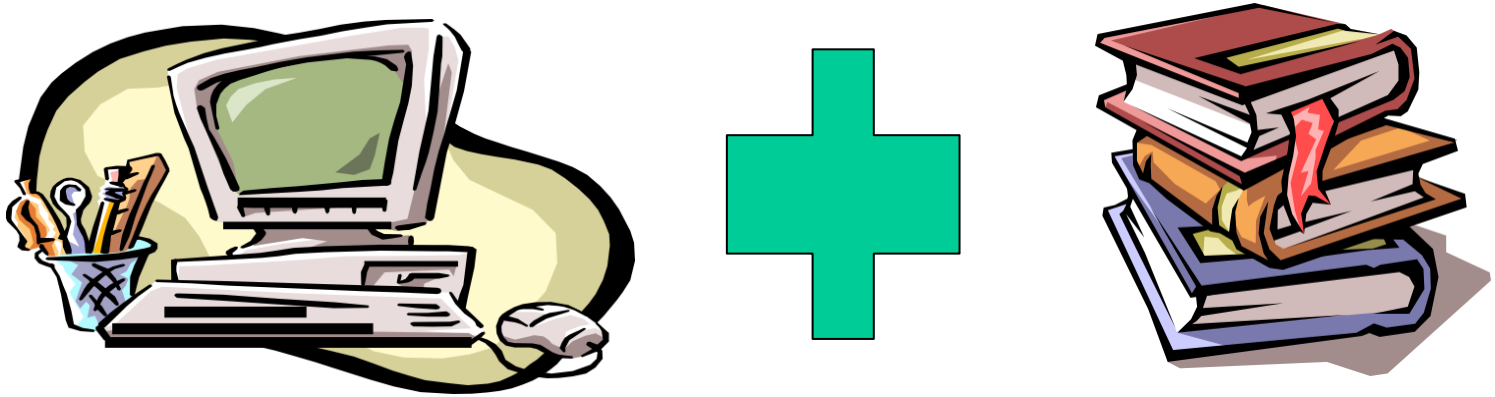
"No SilverBullet"

The blame for software bugs belongs to:

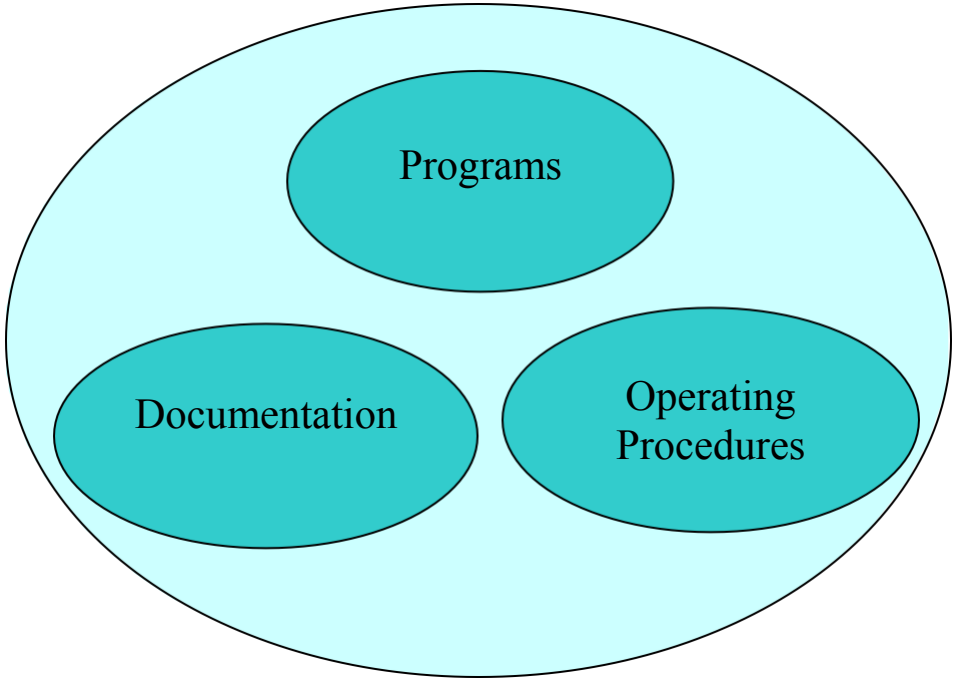
- Software companies
- Software developers
- Legal system
- Universities

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

Documentation Manuals

- Analysis /Specification
  - Formal Specification
  - Context-Diagram
  - Data Flow Diagrams
- Design
  - Flow Charts
  - Entity-Relationship Diagram
- Implementation
  - Source Code Listings
  - Cross-Reference Listing
- Testing
  - Test Data
  - Test Results

List of documentation manuals

Documentation consists of different types of manuals are

System Overview

User Manuals

Beginner's Guide
Tutorial

Reference Guide

Operating Procedures

Installation Guide

Operational Manuals

System Administration Guide

List of operating procedure manuals.

Software Product

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

- **Software products** may be
  - **Generic** - developed to be sold to a range of different customers
  - **Bespoke** (custom) - developed for a single customer according to their specification

Software Product

Software product is a product designated for delivery to the user

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.

Software Process

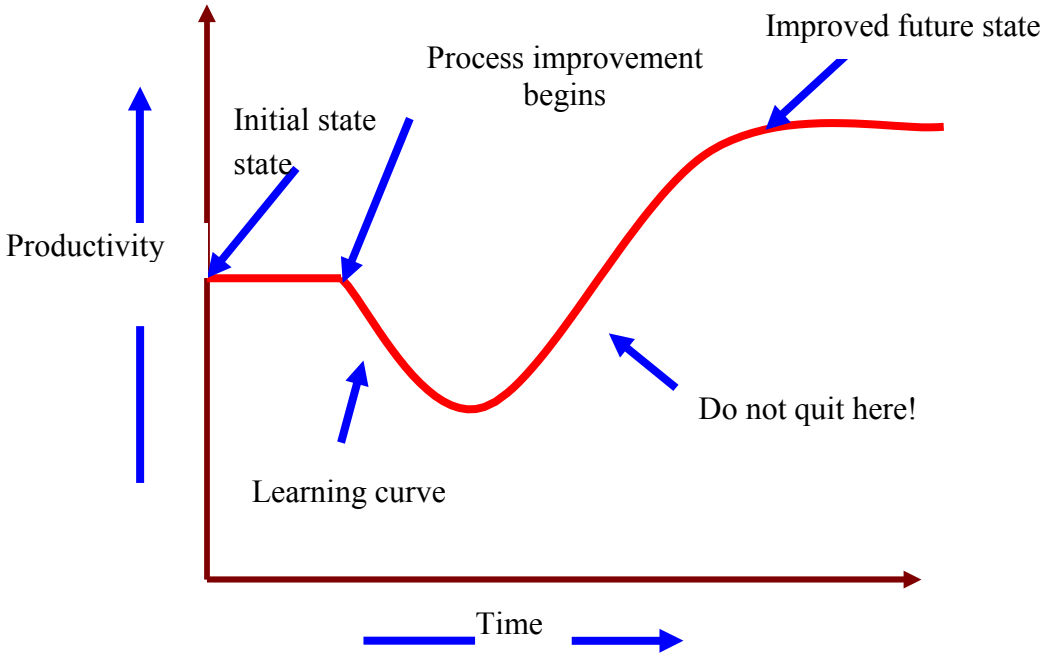
The software process is the way in which we produce software.

Why is it difficult to improve software process ?

- Not enough time
- Lack of knowledge

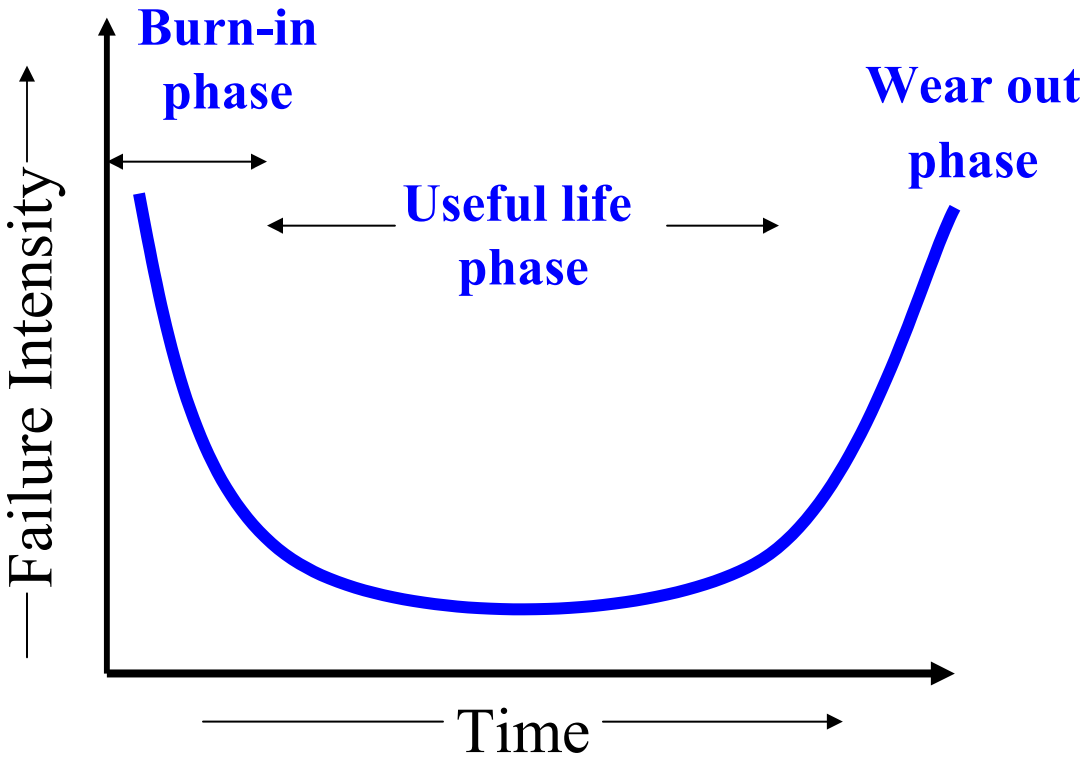
Software Process

- Wrong motivations
- Insufficient commitment



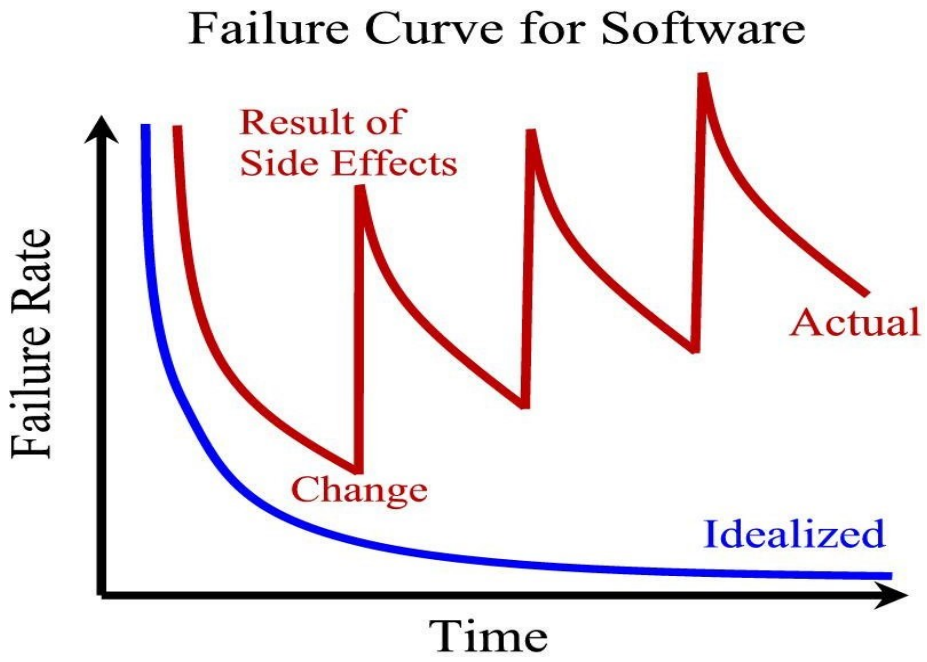
Software Characteristics:

✓ Software does not wear out.



Software Characteristics:

- ✓ Software is not manufactured
- ✓ Reusability of components
- ✓ Software is flexible

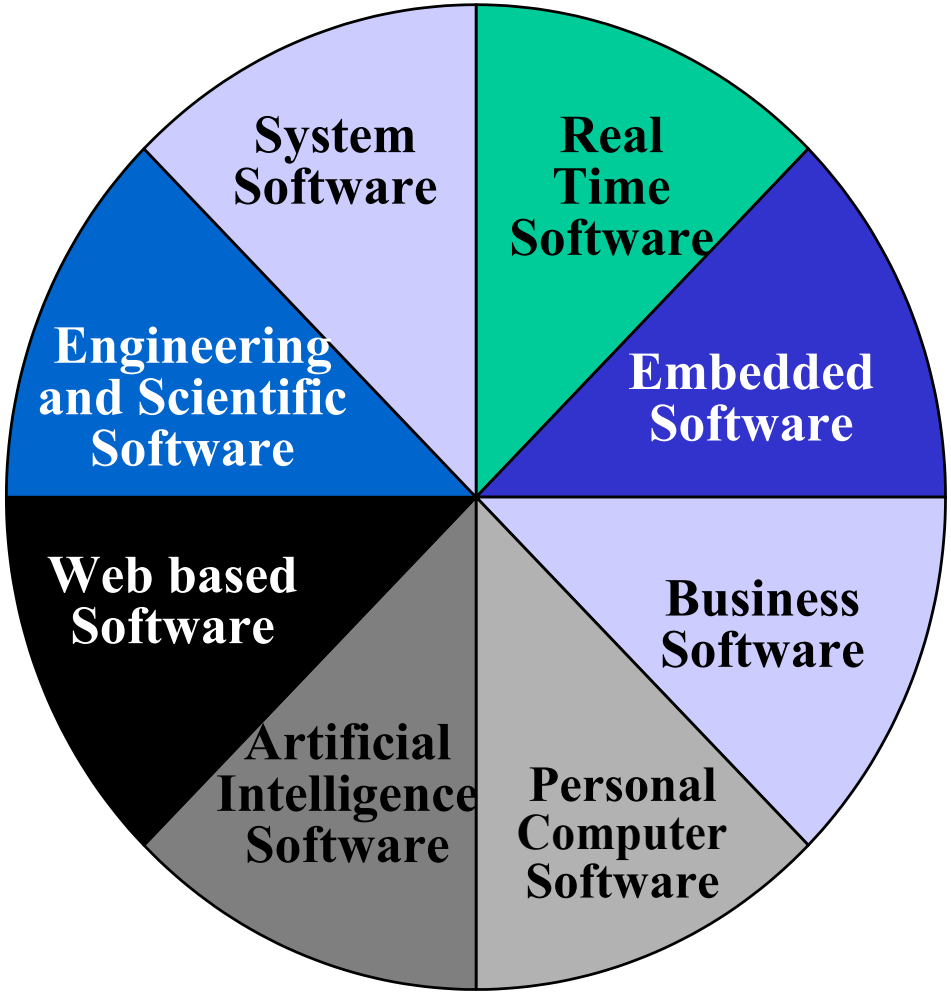


Software Characteristics:

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.

The Changing Nature of Software



The Changing Nature of Software

Trend has emerged to provide source code to the customer and organizations.

Software where source codes are available are known as open source software.

Examples

Open source software: LINUX, MySQL, PHP, Open office, Apache webserver etc.

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!

Unfortunately,
that may further delay the schedule!

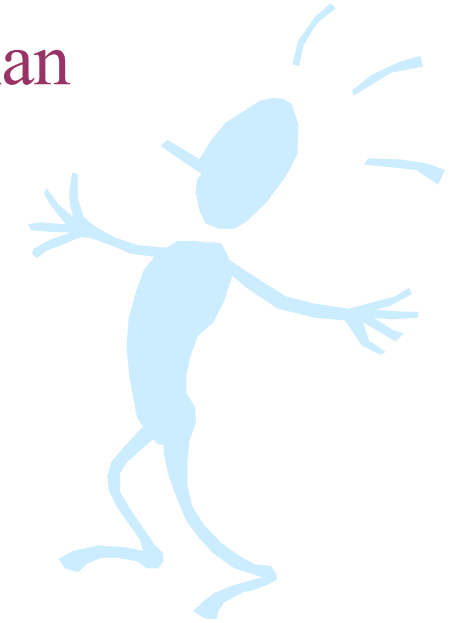
Software Myths (Management Perspectives)

Software is easy to change

The reality is totally different.

Software Myths (Management Perspectives)

Computers provide greater reliability than the devices they replace

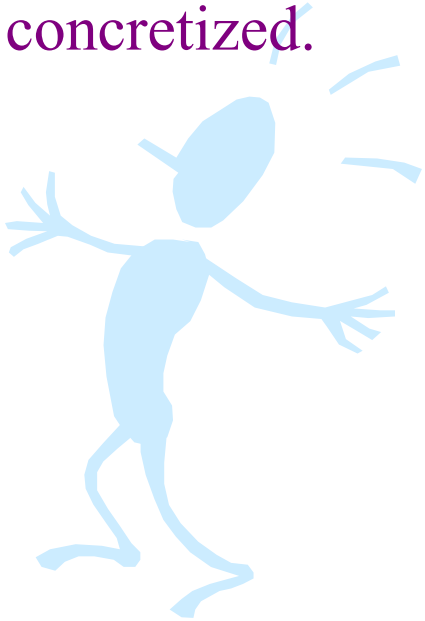


This is not always true.

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)

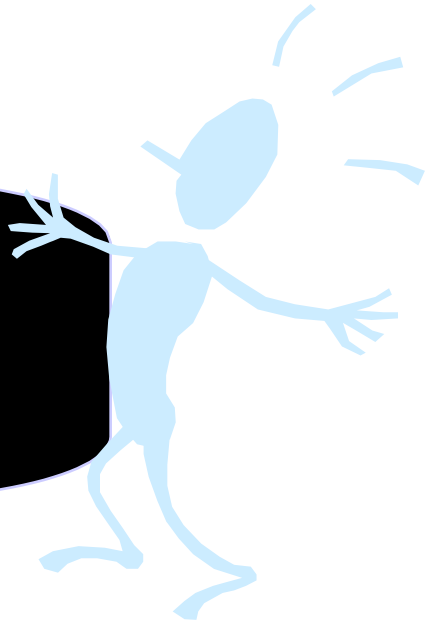
Once the software is demonstrated, the job is done.

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!

Some Terminologies

- Deliverables and Milestones

Different deliverables are generated during software development. The examples are source code, user manuals, operating procedure manuals etc.

The milestones are the events that are used to ascertain the status of the project. Finalization of specification is a milestone. Completion of design documentation is another milestone. The milestones are essential for project planning and management.

Some Terminologies

- Product and Process

Product: What is delivered to the customer, is called a product. It may include source code, specification document, manuals, documentation etc. Basically, it is nothing but a set of deliverables only.

Process: Process is the way in which we produce software. It is the collection of activities that leads to (a part of) a product. An efficient process is required to produce good quality products.

If the process is weak, the end product will undoubtedly suffer, but an obsessive over reliance on process is also dangerous.

Some Terminologies

- Measures, Metrics and Measurement

A measure provides a quantitative indication of the extent, dimension, size, capacity, efficiency, productivity or reliability of some attributes of a product or process.

Measurement is the act of evaluating a measure.

A metric is a quantitative measure of the degree to which a system, component or process possesses a given attribute.

Some Terminologies

- Software Process and Product Metrics

Process metrics quantify the attributes of software development process and environment;

whereas product metrics are measures for the software product.

Examples

Process metrics: Productivity, Quality, Efficiency etc.

Product metrics: Size, Reliability, Complexity etc.

Some Terminologies

- Productivity and Effort

Productivity is defined as the rate of output, or production per unit of effort, i.e. the output achieved with regard to the time taken but irrespective of the cost incurred.

Hence most appropriate unit of effort is Person Months (PMs), meaning thereby number of persons involved for specified months. So, productivity may be measured as LOC/PM (lines of code produced/person month)

Some Terminologies

- Module and Software Components

There are many definitions of the term module. They range from “a module is a FORTRAN subroutine” to “a module is an Ada Package”, to “Procedures and functions of PASCAL and C”, to “C++ Java classes” to “Java packages” to “a module is a work assignment for an individual developer”. All these definition are correct. The term subprogram is also used sometimes in place of module.

Some Terminologies

"An independently deliverable piece of functionality providing access to its services through interfaces".

"A component represents a modular, deployable, and replaceable part of a system that encapsulates implementation and exposes a set of interfaces".

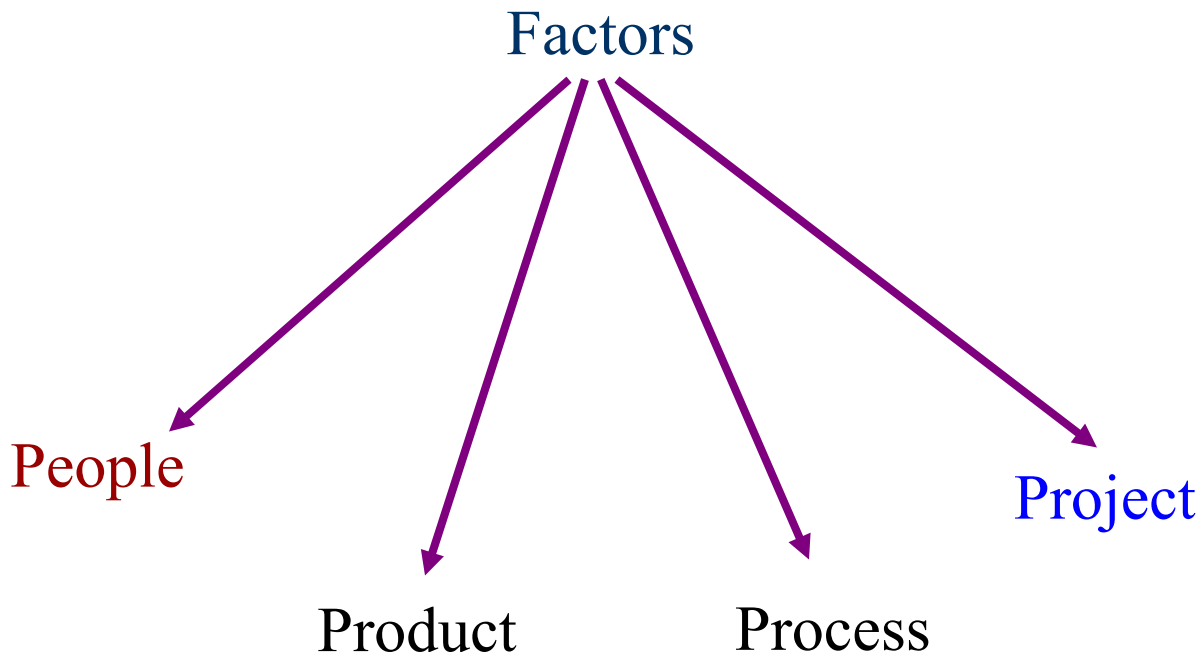
Some Terminologies

- Generic and Customized Software Products

Generic products are developed for anonymous customers. The target is generally the entire world and many copies are expected to be sold. Infrastructure software like operating system, compilers, analyzers, word processors, CASE tools etc. are covered in this category.

The customized products are developed for particular customers. The specific product is designed and developed as per customer requirements. Most of the development projects (say about 80%)come under this category.

Role of Management in Software Development



Role of Management in Software Development