Lecture 1: Introduction to Software Engineering

Software Engineering (CSM 31212)

MM. Mohamed Mufassirin

Lecturer in Computer Science

Dept. of Computer Science

FAS/SEUSL

Overall Objectives:

After successfully completing this module, you should be able to:

- Explain the principles of software engineering and appreciate the need for engineered process in software development
- Understand and be fluent in the use of software **engineering terminology**
- Be capable of intelligently communicating with most members in a software development organization (management, analysts, architects, developers, ...)

Overall Objectives (Cont…)

- Be able to create and use planning, requirements analysis, domain analysis and design objects and carry them into code.
- Be capable of taking on the role of systems analyst in a software development organization
- Be able to document all phases of the software development processes

Syllabus Outline

- Introduction to software engineering
- Software Process
- Software requirement specification – different models for identifying software requirements
- Software design methodologies
- Coding and documentation
- Software Testing
- Software Project Management
- Software Configuration Management

Recommended Reading (References)

- Ian Sommerville, **Software Engineering**, 10th Edition, Addison Wesley, 2016.

- R Pressman, **Software Engineering - A Practitioners Approach**, 9th Edition, McGraw Hill., 2019

Evaluation Method

Two Credits Hours (30Hrs)

- 2/3 Quizzes – 10%
- In-class Assessment – 10%
- Assignment – 10%

- End Semester Exam – 70%

Contact Info: Mobile: 077 666 0000
E-mail: mufassirin@seu.ac.lk

Software Engineering

Introduction

Learning Outcomes

- Appreciate the problems associated with developing software
- Understanding the need for a managed approach to software development
- Be able to define the term ‘Software Engineering’
- Identify software quality attributes and their classification

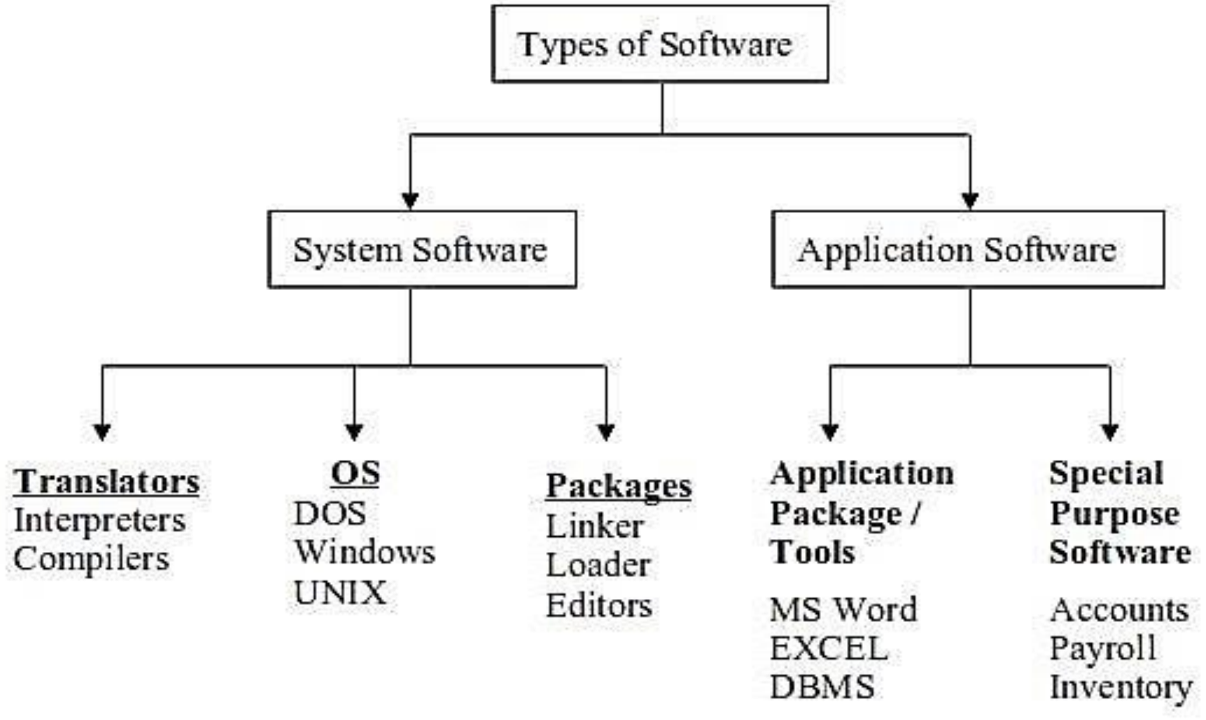
Software

What is Software?

- Set of instructions given to a computer (computer programs) to perform particular task/tasks

  - Systems software
  - Application software

Types of software

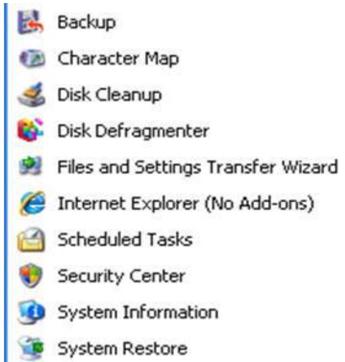


Types of Software (Cont..)

- System Software:

  System software is a collection of programs written to service the other programs.

- Operating systems, (eg. MS Windows, Mac OS, Linux)
- Programming Language Translator, (eg. Interpreter, compilers)
- Utility Software, (eg. Antivirus, Defragmentation agent)


Backup
Character Map
Disk Cleanup
Disk Defragmenter
Files and Settings Transfer Wizard
Internet Explorer (No Add-ons)
Scheduled Tasks
Security Center
System Information
System Restore

Types of Software (Cont..)

- Application software
  - Computer software designed to help the user to perform a singular or multiple related specific tasks.

    2 types of application Software.

    - i) General purpose application software
    - ii) Specific purpose application software

- **General purpose application software**
  - This software is much broader in use

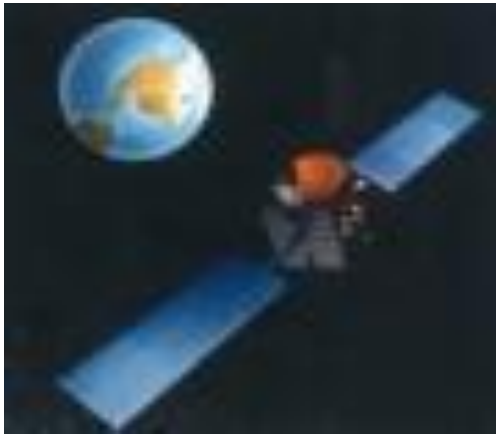
    Eg. Word processors, Database

- **Specific purpose application software**
  - This software is very specific in its use.

    Eg. Engineering programs, Mathematics Software

Types of Software (Cont..)

- Web-based software:
- The network becomes a massive computer providing an almost unlimited software resources that can be accessed by anyone with a modem.



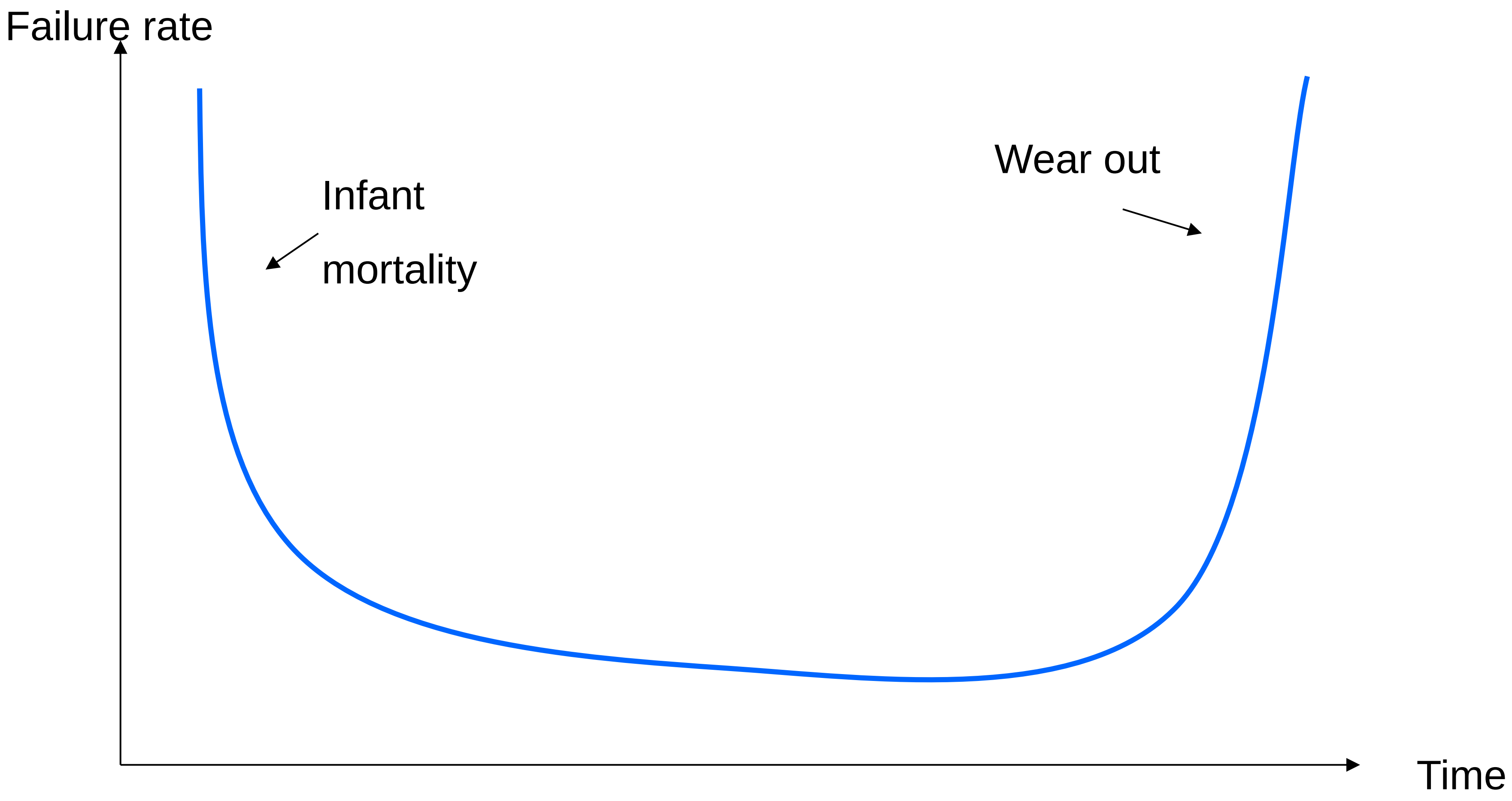
•Artificial intelligence software:

All software makes use of non numerical algorithms to solve the complex problems that are not amenable to computing or straightforward analysis.

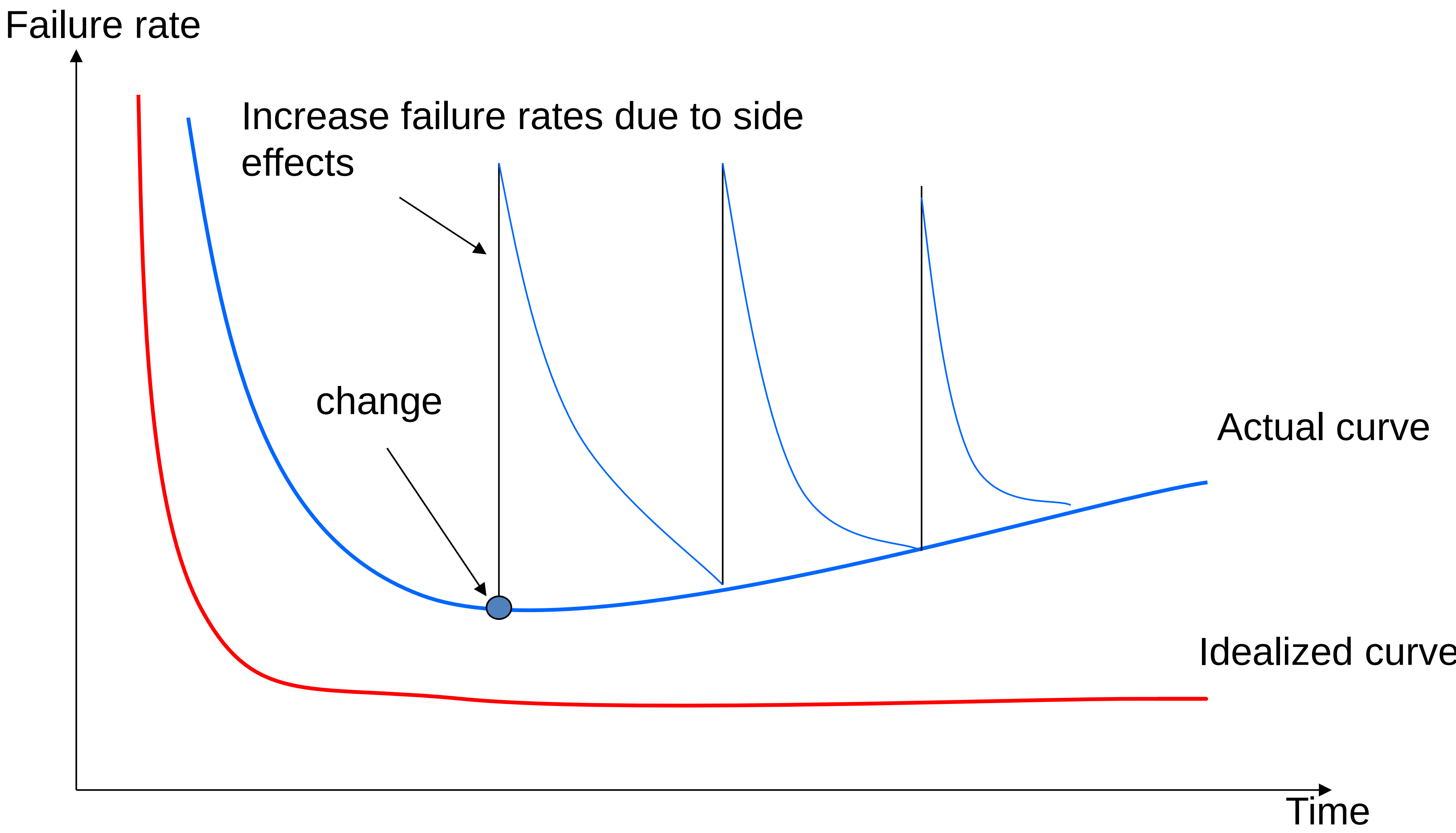
Software/ Hardware Failure

The inability of a program/ hardware to continue processing due to erroneous logic

Failure curve for hardware [Pressman]



Failure curves for software -Pressman



Development Failures

Examples

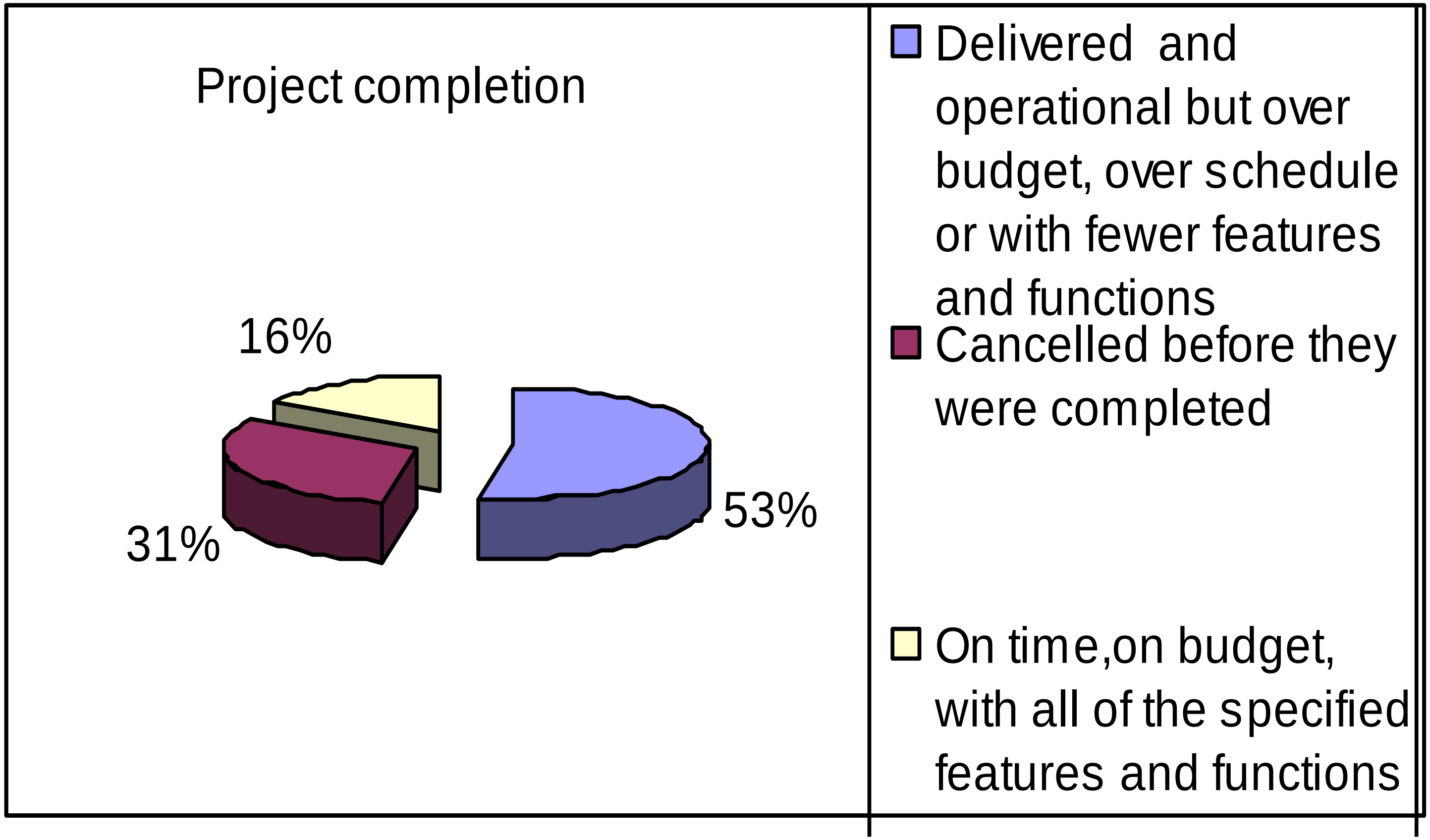
IBM Survey, 2000

- 55% of systems cost more than expected
- 68% overran the schedules
- 88% had to be considerably redesigned

Bureau of Labour Statistics, (2001)

- for every 6 new systems put into operation, 2 cancelled
- probability of cancellation is about 50% for large systems
- average project exceeds schedule by 50%

The Statistics



Distribution of software cost over life cycle

$$\left.\begin{array}{l}\text{1. Requirements capture}\\ \text{2. Requirement specification 14\%}\\ \text{3. Design}\\ \text{4. Implementation}\\ \text{5. Testing 6\%}\end{array}\right\}\ \text{40\%}$$

6. Maintenance 60%

Software Problems

1. Time Schedules and cost estimates of many software projects are grossly inaccurate.
2. Software is costly.
3. The quality of software is not satisfactory.
4. Software is difficult to maintain.
5. The productivity of software people is not satisfactory to meet the demand.

Software Engineering

What makes software special?

The main difference in software engineering compared to other engineering disciplines are listed below.

1. It is difficult for a customer to specify requirements completely.
2. It is difficult for the developer to understand fully the customer needs.
3. Software requirements change regularly.
4. Software is primarily intangible; much of the process of creating software is also intangible, involving experience, thought and imagination.
5. It is difficult to test software thoroughly.

A Solution – Software Engineering

- Greater emphasis on systematic , scientific development.
- Computer assistance in software development (CASE)
- A concentration on finding out the user’s requirements
- Formal/Semi Formal specification of the requirements of a system.
- Demonstration of early version of a system (prototyping)
- Grater emphasis on development of error free easy to understand code.

Evolution of software engineering

1. Software development began as a single person activity in 1940s and 1950s.
2. Software engineering was considered a new scientific discipline in 1960s and 1970s.
3. In 1980s and 1990s engineering ideas dominated software development

Software Engineering

Simple definition :

- Designing, building and maintaining large software systems.

Others

- => ‘Software engineering is concerned with the theories, methods and tools for developing, managing and evolving software products’ **- I Sommerville**

=> ‘The practical application of scientific knowledge in the design and construction of computer programs and the associated documentation required to develop, operate and maintain them’ **- B.W.Bohem**

=> ‘ The establishment and use of sound engineering principles in order to obtain economically software that is reliable and works efficiently on real machines’ **- F.L. Bauer**

=> ‘The application of systematic, disciplined, quantifiable approach to the development, operation, and maintenance of software’ **- IEEE Standard 610.12**

Software Engineering – Definition

‘Use of systematic, engineering approach in all stages of software development and project management to develop high quality and economical software using appropriate software tools.’

What is the difference between Software Engineering and Computer Science?

What is the difference 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

What is the difference between Software Engineering and System Engineering?

What is the difference 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

Software Quality Attributes

Bohem's Classification

- **Current usefulness**

  - The qualities expected from a software system in user's point of view.

- **Potential Usefulness**

  - The qualities expected from a software system in developer's point of view.

Current usefulness

- ✓ **Efficiency** - Software should not make wasteful use of system resources
- ✓ **Reliability** – is that consistently performs according to its specifications
- ✓ **Usability** - Software must be usable by the users for which it was designed
- ✓ **Correctness** - developed under contract, purchasers and developers iron out specifications for the software
- ✓ **User friendliness** - easily operated and understood by means of a straightforward guide.
- ✓ **Robustness** - is the ability to handle exceptional conditions.

Potential usefulness

- Maintainability - Software must evolve to meet changing needs
- Modularity - is a software design technique that emphasizes separating the functionality of a program into independent, interchangeable modules
- Reusability - is the use of existing assets/ resources in some form within the software product development process
- Portability - can be used in an operating systems other than the one in which it was created without requiring major rework.

Software Maintenance

Any software system needs to be changed. Maintenance is the costliest operation in the software development process (about 60%). Software need to be changed due to various reasons.

- Errors in the system.
- Changes in the user requirements
- Availability of new technology
- Changes in the enterprise or Govt. policy.

Why software engineering?

- Early days system development is an individual work or a work by two or three people
- Such projects ran into failures
  - Exceed the budget
  - Took more time
  - Difficulty in maintain
- So an alternative to such ad hoc method is needed.
- Engineering techniques are successfully applied to software development
  - Development becomes a team work rather than an individual work
  - Only experts involved in the process
  - Quality is guaranteed

Why Software engineering

- The economies of ALL developed nations are dependent on software.
- More and more systems are software controlled
- Expenditure on software represents a significant fraction of Gross National Product (GNP) in all developed countries.

Software costs

- Software costs often dominate computer system costs. The costs of software on a PC are often greater than the hardware cost.
- Software costs more to maintain than it does to develop. For systems with a long life, maintenance costs may be several times the development costs.
- Software engineering is concerned with cost-effective software development.

Software products

- Generic products
  - Stand-alone systems that are marketed and sold to any customer who wishes to buy them.

  Examples – PC software such as graphics programs, project management tools; CAD software; software for specific markets such as appointments systems for dentists.
- Customized products
  - Software that is commissioned by a specific customer to meet their own needs.

  Examples – embedded control systems, air traffic control software, traffic monitoring systems.

Importance of software engineering

- More and more, individuals and society rely on advanced software systems. We need to be able to produce reliable and trustworthy systems economically and quickly.
- It is usually cheaper, in the long run, to use software engineering methods and techniques for software systems rather than just write the programs as if it was a personal programming project.
- For most types of system, the majority of costs are the costs of changing the software after it has gone into use.

Software process activities

- Software specification, where customers and engineers define the software that is to be produced and the constraints on its operation.
- Software development, where the software is designed and programmed.
- Software validation, where the software is checked to ensure that it is what the customer requires.
- Software evolution, where the software is modified to reflect changing customer and market requirements.

General issues that affect most software

- Heterogeneity
  - Increasingly, systems are required to operate as distributed systems across networks that include different types of computer and mobile devices.
- Business and social change
  - Business and society are changing incredibly quickly as emerging economies develop and new technologies become available. They need to be able to change their existing software and to rapidly develop new software.
- Security and trust
  - As software is intertwined with all aspects of our lives, it is essential that we can trust that software.

Questions

Multiple choice quiz

http://highered.mcgraw-hill.com/sites/0073375977/student_view0/chapter1/multiple_choice_quiz.html

Thank You

?