SOFTWARE ENGINEERING

(15B11CI513)

Credits :- 4 **Contact Hours :-** 3-1-0

Course Outcomes

At the completion of the course, students will be able to:

COURSE OUTCOMES		COGNITIVE LEVELS
C314.1	Explain software engineering principles and software process models for project development.	Remembering (Level 1)
C314.2	Identify functional and non-functional requirements of a software project and design document software requirements specification.	Understand (Level 2)
C314.3	Design, represent and document software requirements specification. Plan and execute activities for a software project.	Create (Level 6)
C314.4	Apply UML modeling for software design from software requirements specification.	Apply (Level 3)
C314.5	Analyze code checklist. Perform code Reviews, Code Refactoring, and Code optimization, design pattern	Analyze (Level 4)
C314.6	Apply testing principles, develop and implement various manual and automated testing procedures, formal methods	Apply (Level 3)
C314.7	Evaluate software in terms of general software quality attributes and possible trade-offs presented within the given problem.	Evaluate (Level 5)

Course Description

Module No.	Topics in the Module	No. of Lectures for the module
1.	Introduction to Software Engineering: Introduction to software engineering Principles, Software process models (build and fix model, waterfall model, Incremental process model, Evolutionary- Prototype and Spiral models, Agile Models (tools study), PSP, TSP, Software Reengineering. Project planning, Project Scheduling: network diagram, Gant Chart, CPM and PERT.	7
2.	Requirement Engineering: Types of requirement, Requirement Elicitation, Analysis, Specification, SRS, Requirement Verification and Validation.	4
3.	Software Design: Use case diagram, State diagram, Activity Diagram, Class Diagram, Sequence diagram, Collaboration diagram, Deployment Diagram, Component Diagram and Package diagram. Design Modularity: Coupling Cohesion.	7
4.	Software Construction: Coding standards and guidelines, Code checklist, Code Reviews, Code Refactoring, Code optimization. Design pattern, Modern programming environments (Code search, Programming using library components and their APIs), Program comprehension; Program correctness, Defensive programming.	8
5.	Software Metrics: Size-Oriented Metric, Function-oriented Metric, Halstead's Software Metric, Information Flow Metric, Object-oriented Metric, Class-Oriented Metric, COCOMO Model.	7
6.	Software Testing: White-Box Testing, Basis Path Testing, Control Structure Testing: Condition Testing, Data Flow Testing, Loop Testing, Black-Box Testing: Equivalence class partitioning, Boundary Value Analysis, Decision table testing, Cause effect graphing, Mutation Testing and regression Testing, formal methods.	9
	Total number of Lectures	42

Recommended Reading material

Text Book(s):	
1.	Roger S. Pressman, "Software Engineering: A practitioner approach", Fifth Edition-TMH International .
2.	Sommerville , "Software Engineering" , Seventh Edition - Addison Wesley.
Reference Book(s):	
3.	Grady Booch, James Rumbaugh, Ivar Jacobson, The Unified Modeling Language User Guide, Addison Wesley, Reading, Massachusetts, May 2005
4.	Richard Thayer , "Software Engineering Project Management", Second Edition -Wiley-IEEE Computer Society Press.
5.	B. Bezier, "Software Testing Techniques", Second Edition- International Thomson Computer Press.
6.	Pankaj Jalote, "An Integrated Approach to Software Engineering" Third addition , Springer Press

Evaluation Criteria

Components	**Maximum Marks**
T1	20
T2	20
End Semester Examination	35
TA	25 (Assignments, Tutorial, Mini Project, Attendance)
Total	**100**

SOFTWARE ENGINEERING

(15B11CI513)

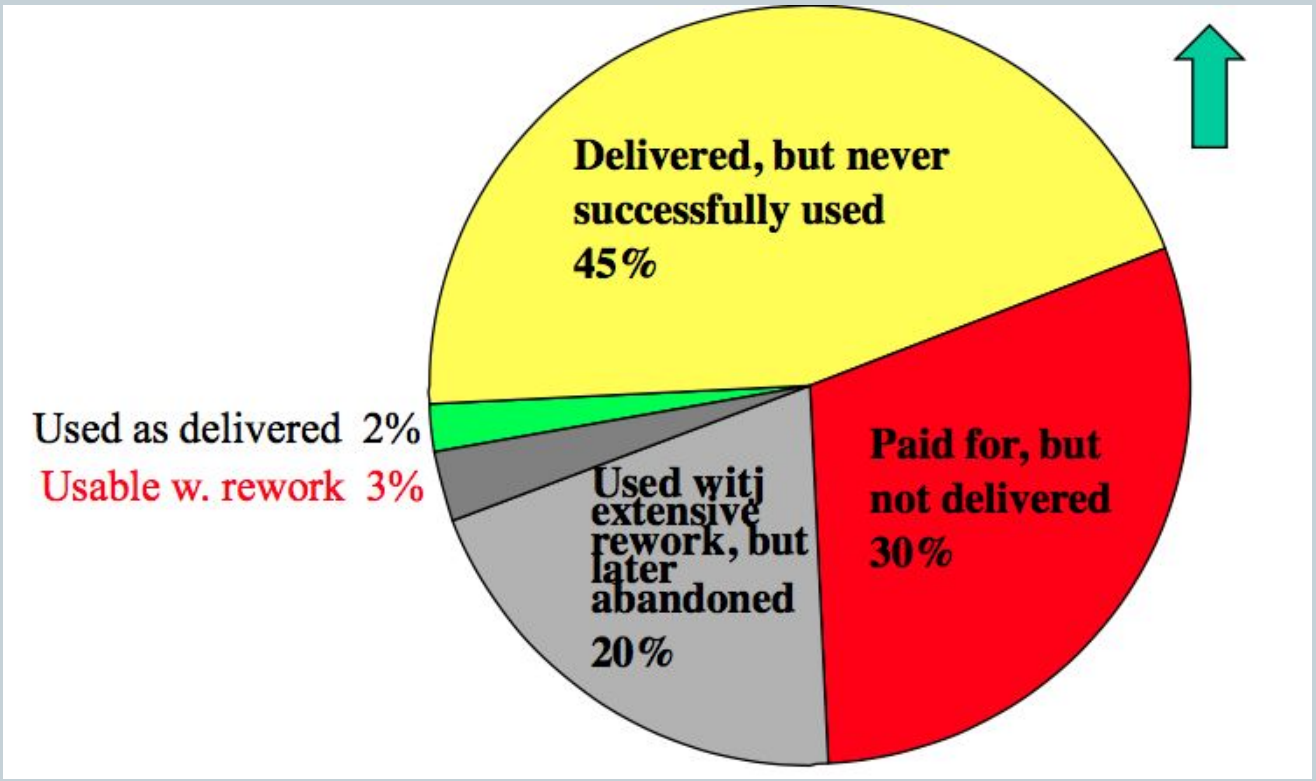
Module 1: *Introduction to Software Engineering*

Topics in Module 1

- Introduction to software engineering Principles
- Software process models
  - Build and fix model
  - Waterfall model
  - Incremental process model
  - Evolutionary
  - Prototype
  - Spiral models
- Agile Models (tools study)
- PSP, TSP
- Software Reengineering
- Project planning
- Project Scheduling
  - Network diagram
  - Gant Chart
  - CPM
  - PERT.

Why Software Engineering?

- 9 software projects totaling $96.7 million: Where The Money Went [Report to Congress, Comptroller General, 1979]



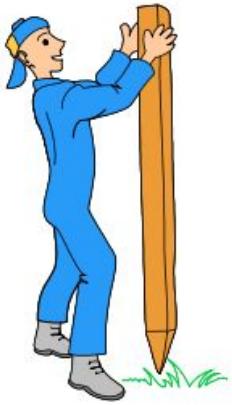
Scientist vs Engineer

- Computer Scientist
  - Proves theorems about algorithms, designs languages, defines knowledge representation schemes
  - Has infinite time…
- Engineer
  - Develops a solution for an application-specific problem for a client
  - Uses computers & languages, tools, techniques and methods
- Software Engineer
  - Works in multiple application domains
  - Has only 3 months…
  - …while changes occurs in requirements and available technology

Software is Complex

- Complex ≠ Complicated
- Complex=composed of many simple parts related to one another
- Complicated = not well understood, or explained

Complexity Example: Scheduling Fence Construction Tasks



Setting posts
[3 time units]

Cutting wood
[2 time units]

Nailing
[2 time units for unpainted;
3 time units otherwise]

Painting
[5 time units for uncut wood;
4 time units otherwise]

Setting posts < Nailing, Painting

Cutting < Nailing

…shortest possible completion time = ?

[⇒ "simple" problem, but hard to solve without a pen and paper]

Software Engineering: A Problem Solving Activity

- Analysis: Understand the nature of the problem and break the problem into pieces
- Synthesis: Put the pieces together into a large structure

For problem solving we use

- Techniques (methods):
  - Formal procedures for producing results using some well-defined notation
- Methodologies:
  - Collection of techniques applied across software development and unified by a philosophical approach
- Tools:
  - Instrument or automated systems to accomplish a technique

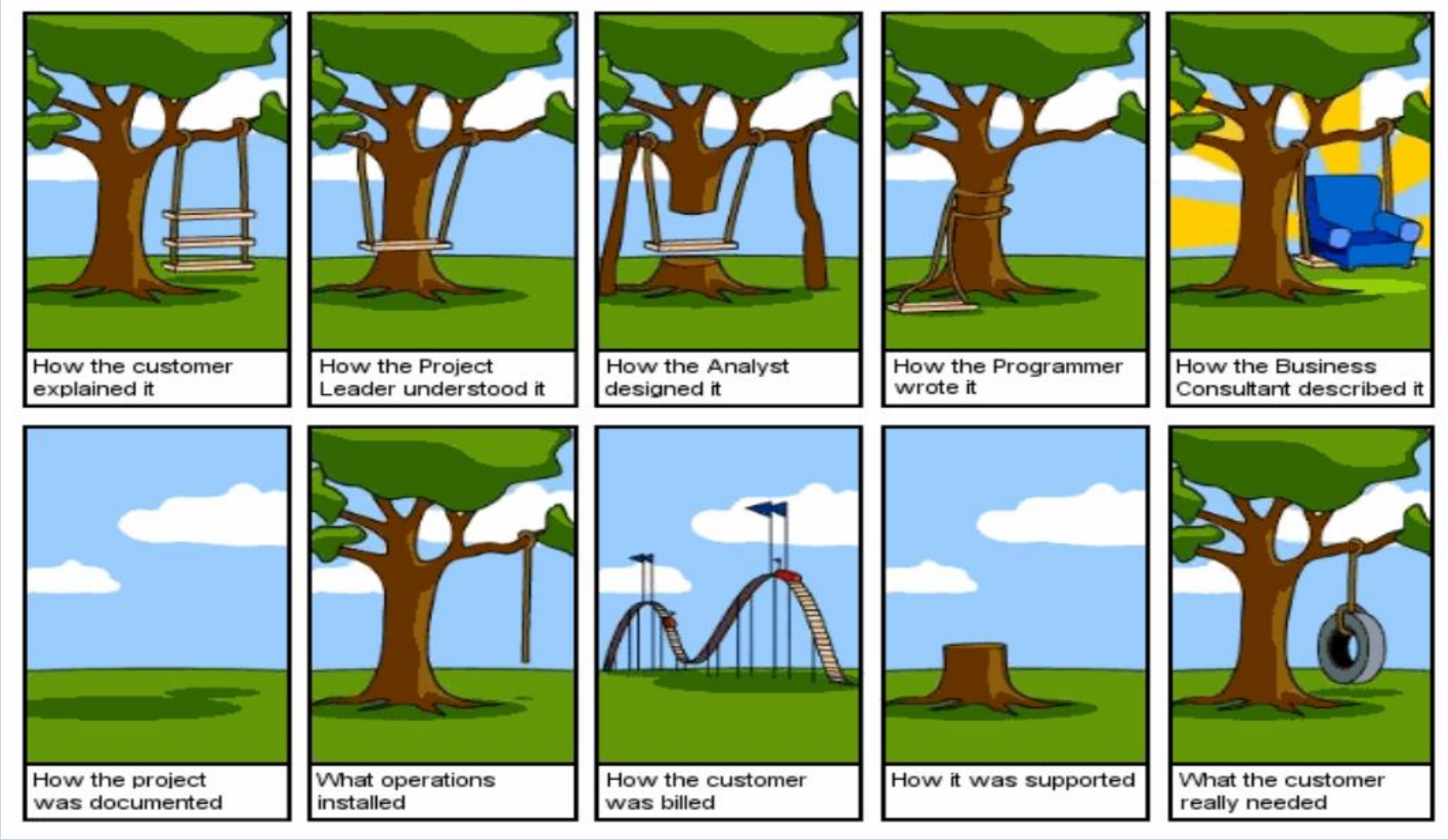
Factors affecting the quality of a software system

- Complexity:
  – The system is so complex that no single programmer can understand it anymore
  – The introduction of one bug fix causes another bug
- Change:
  – The "Entropy" of a software system increases with each change: Each implemented change erodes the structure of the system which makes the next change even more expensive ("Second Law of Software Dynamics").
  – As time goes on, the cost to implement a change will be too high, and the system will then be unable to support its intended task. This is true of all systems, independent of their application domain or technological base.

What is Software?

- Software (IEEE) is a collection of
  - programs,
  - procedures,
  - rules, and
  - associated documentation and data

The Software Challenge



Software Engineering: Definition

- Software Engineering is a collection of techniques, methodologies and tools that help with the production of
  - a high quality software system
  - with a given budget
  - before a given deadline

while change occurs.

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 (cont…)

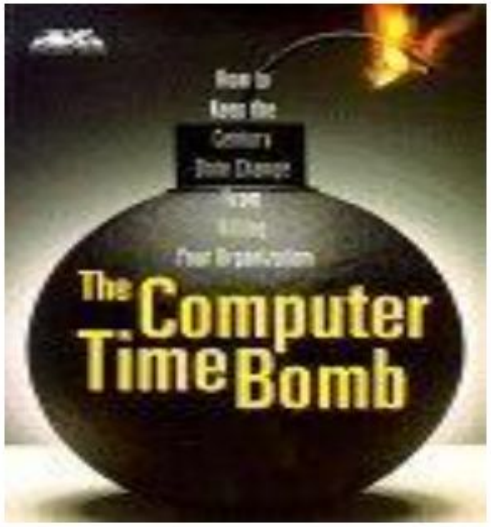
When the guidance system's own computer tried to convert one piece of data the sideways velocity of the rocket from a 64 bit format to a 16 bit format; the number was too big, and an overflow error resulted after 36.7 seconds. When the guidance system shutdown, it passed control to an identical, redundant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

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 (cont…)

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 (cont…)

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 (cont…)

- 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 (cont…)

- 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 (cont…)

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.

Software Applications

- System software
- Application software
- Engineering/scientific software • embedded software
- Product-line software
- WebApps (Web applications)
- AI software

Software—New Categories

- Open world computing—pervasive, distributed computing
- Ubiquitous computing—wireless networks
- Netsourcing—the Web as a computing engine
- Open source—”free” source code open to the computing community
- Data mining
- Grid computing
- Cognitive machines
- Software for nanotechnologies