Fundamentals of Software Engineering

Lecture 1
Introduction

Module Outline

Credit	3
Content	30 hours Lectures and 15 Hours Tutorials
Evaluation	40% for Assessments 60% for Final examination
Assessments	Two Assessments; - CA1 (25) - Small Software Development - CA2 (15) - Online Quiz

Learning Outcomes

On successful completion of this module, students are able to:
1. Elucidate the engineering approach of software development industry.
2. Utilize knowledge on requirement engineering in commercial applications.
3. Map suitable development methodology based on the requirement analysis.
4. Demonstrate the ability to improve quality of a software using reverse engineering.
5. Appraise the best code restructuring method based on the development environment.
6. Apply software engineering knowhow to develop & deploy basic software solutions with requisite compromising the budget and the estimated time.

What is Software?

- Software specifies the set of computer programs, procedures and associated documents (Flowcharts, manuals, etc.) that describe the program and how they are to be used.

- Software encompasses:

(1) instructions (computer programs) that when executed provide desired features, function, and performance;

(2) data structures that enable the programs to adequately store and manipulate information and

(3) documentation that describes the operation and use of the programs.

Why Software is important?

- Software engineering is important, because specific software is needed in almost every industry, in every business, and for every function.
- Business benefits of new software
    - Cut costs by automating routine tasks.
    - Improve the efficiency of staff.
    - Increase or measure office productivity.
    - Streamline business operations and accounts.
    - Replace paper processes.
    - Communicate more effectively with customers, suppliers or partners.

Types of Software

- There are two types of software,
  - System Software
  - Application Software

Application Software

- Application software products are designed to satisfy a particular need of a particular environment. All software applications prepared in the computer lab can come under the category of Application software.
- Application software may consist of a single program, such as Microsoft's notepad for writing and editing a simple text.
- It may also consist of a collection of programs, often called a software package, which work together to accomplish a task, such as a spreadsheet package.
- Examples of Application software are the following –
  - Payroll Software
  - Student Record Software
  - Inventory Management Software
  - Income Tax Software

System Software

- The system software is a collection of programs designed to operate, control, and extend the processing capabilities of the computer itself.

- System software is generally prepared by the computer manufacturers. These software products consist of programs written in low-level languages, which interact with the hardware at a very basic level.

- System software serves as the interface between the hardware and the end users. Some examples of system software are Operating System, Compilers, Interpreter, Assemblers, etc.

Exercise 1

What are the differences between system software and application software

Software Engineering Definition

- IEEE defines software engineering as: (1) The application of a systematic, disciplined, quantifiable approach to the development, operation and maintenance of software; that is, the application of engineering to software.

- Fritz Bauer, a German computer scientist, defines software engineering as: (2) Software engineering is the establishment and use of sound engineering principles in order to economically obtain software that is reliable and work efficiently on real machines.

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.

Recap from lesson learnt

Mark the following True/False

1. Software is a set of instructions, data or programs used to operate computers and execute specific tasks. (True/False)
2. Software is often divided into two categories. Those applied software and system software (True/False)
3. Application software includes operating systems and any program that supports system software. (True/False)
4. Each programme in application software is designed to assist the user with a particular process, which may be related to productivity, creativity, and/or communication. (True/False).
5. Software engineering is a process of analyzing user requirements and then designing, building, and testing software applications which will satisfy those requirements. (True/False).
6. Software Engineering helps to improve software crises. (True/False)
7. When we develop a software we should follow software engineering disciplines.(True/False)

Software Processes

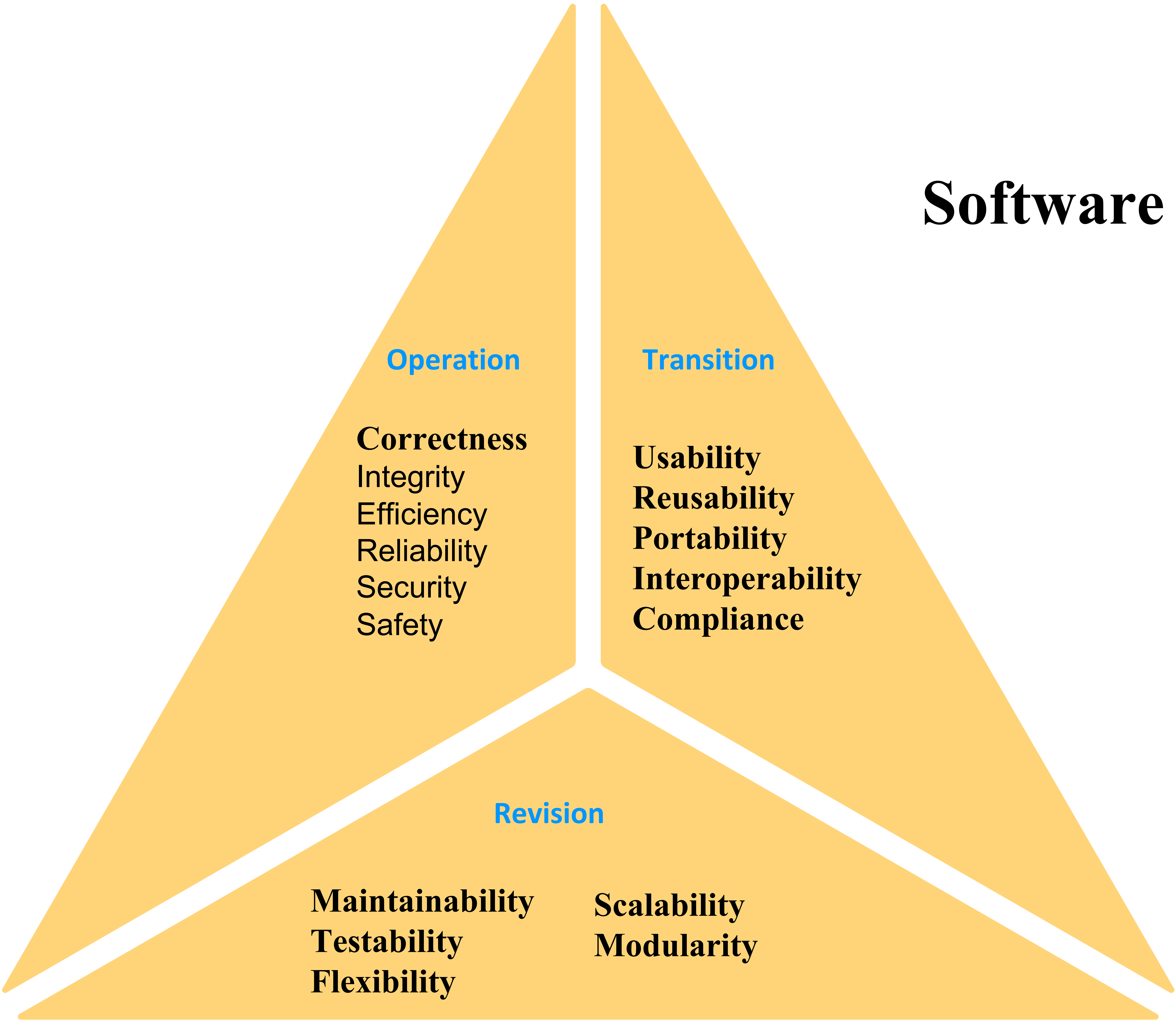
Software specifications: The functionality of the software and constraints on its operation must be defined.

Software development: The software to meet the requirement must be produced.

Software validation: The software must be validated to ensure that it does what the customer wants.

Software evolution: The software must evolve to meet changing client needs.

Software Quality Triangle



Operation

Correctness
Integrity
Efficiency
Reliability
Security
Safety

Transition

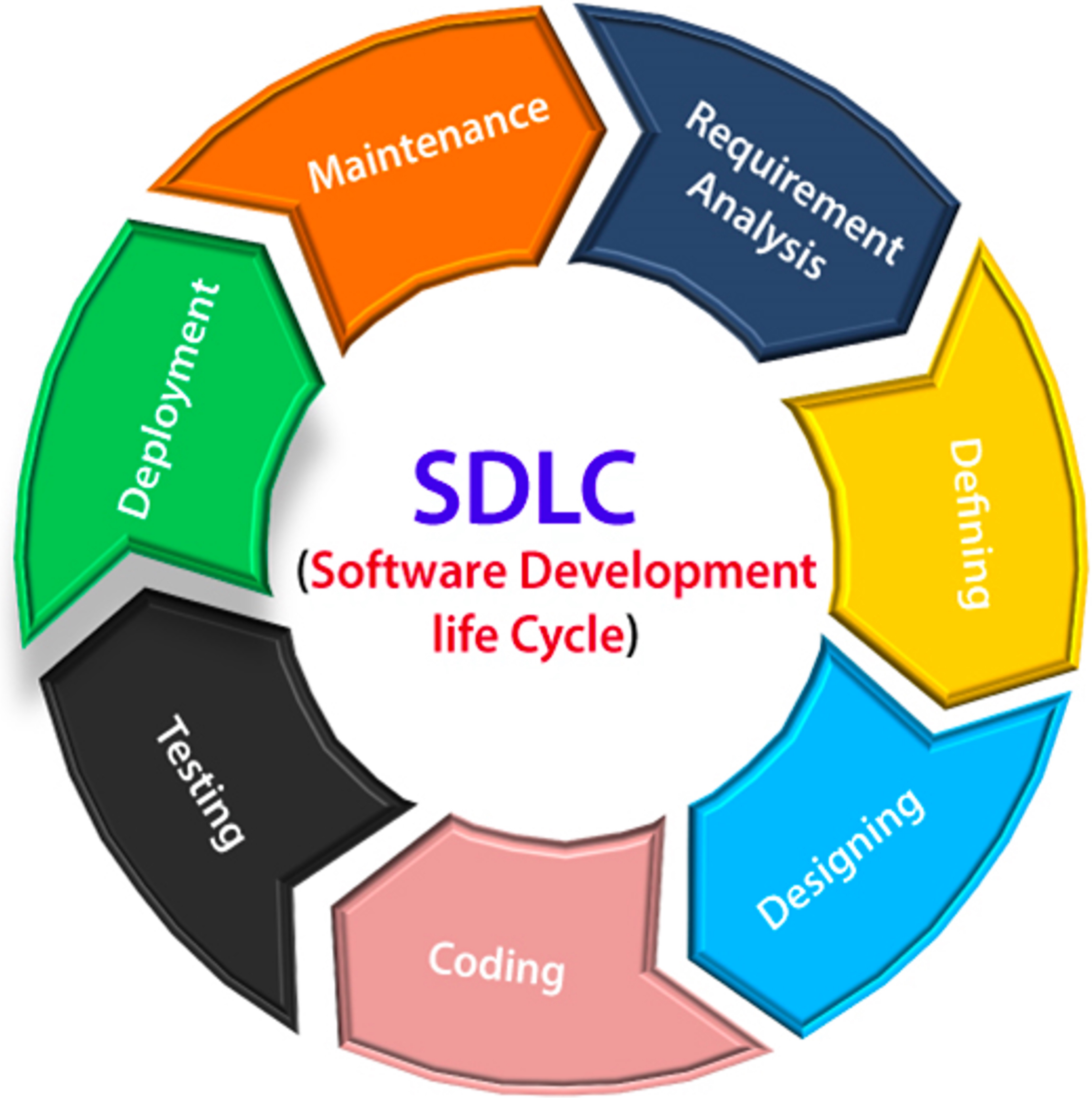
Usability
Reusability
Portability
Interoperability
Compliance

Revision

Maintainability
Testability
Flexibility

Scalability
Modularity

Software Development Lifecycle



Recap from lesson learnt

Mark the following True/False

1. Reliability means the software product should not have any defects.(True/False)
2. Reusability is the ability of software to exchange information with other applications and make use of information transparently. (True/False)
3. The ultimate purpose of SDLC is to deliver a high-quality product which is as per the customer's requirement.(True/False)
4. Software Development Life Cycle (SDLC) is a framework that defines the steps involved in the development of software at each phase. (True/False).
5. During implementation phase, all the relevant requirements are collected from the customer to develop a product as per their expectation. (True/False).
6. Testing starts once the coding is complete and the modules are released for testing. (True/False).
7. Source Code Management applications are used in requirement analysis phase in order to define the requirements. (True/False)
8. Software Requirement Specification should include all functional and non-functional requirements. (True/False)
9. Software validation is the process of converting a system specification into an executable system through the design of system. (True/False)
10. Software evolution is a process of developing software initially, then timely updating it for various reasons. (True/False)

Introduction to CA1

Student is required to implement small software application and maintain all the phases in software development from requirements gathering to deployment .

Initially, student should identify the major project idea and submit complete proposal.

Ex: Issue tacking application

Main system features are Login, registration, Create/edit/delete issues, view issue list, assign user roles for identified task to resolve issue/s, check the status (open/in progress/waiting on client/resolved), set deadlines to resolve issue, notifications, etc.

However, at the development stage student is required to develop only three/four main functionalities in one feature.

Ex: Manage issue with CRUD operations.

Introduction to CA1 cont….

There will be mainly four submissions : Proposal submission, requirements definition of the SRS , source code submission only for the developed feature, Unit test cases submission.

At the final presentation student is required to demonstrate the implmented functionalities.