1

Software Engineering
&
Software Development Life Cycle

Objectives

After completing this lesson, you should be able to do the following:

- Understand software engineering importance,concepts
- Definition of Software & software Engineering
- Characteristics of software engineering
- Practitioner’s view of software engineering
- Software applications
- Overview of SDLC
- Phases of software Development ,purpose and deliverables
- Describe Requirements Engineering Process
- To describe functional and non-functional requirements

Brief History - Evolution of Software

1950s	The Early Years	Batch Orientation Custom Software
1970s	The Second Era	Multi User – Real Time Database – Product Software
1980s	The Third Era	Distributed Systems Low Cost Hardware Consumer Impact
1990s	The Fourth Era	Powerful Desktop Systems OO Technology Neural Networks GUI Application

Software Industry

- Emphasis has rightly shifted from hardware to software.
- In India itself, software industry growth has been phenomenal.
- IT field has enormously grown in the past 50 yrs.....
- IT industry in India is expected to touch 10,000 crores of which software share is dramatically increasing.

Software Crisis

Software costs/schedules are grossly inaccurate

- Cost overruns of several times, schedule slippage's by months, or even years are common.

Productivity of people has not kept pace with demand

- ... added to it is the shortage of skilled people…

Quality of software is less than desired

- Error rates of released software leave customer dissatisfied.. Threatening the very business.

Why this Crisis ?

Characteristic of software.

- Is Intangible.
- The logical nature provides challenge to the developer, gets personalized.

What is Software Engineering?

- Definition of Software
- Definition of Software Engineering (SE)
- Characteristics of SE
- Practitioner's view of SE
- Software Applications

Definition of Software…

Definition of Software (Product)

- Instructions (Computer Program) that when executed provide desired functions and performance
- Data structures that enable the instructions to adequately manipulate information
- Documents that describe the operation and use of the instruction .

Software is a set of items or objects that form a "configuration" that includes:

- Programs
- Documents
- Data

Definition of SW Engineering

IEEE definition

- The application of systematic, disciplined and quantifiable approach to the development, operation, and maintenance of software; that is, the application of engineering to software.

Sommerville definition

- Software engineering is concerned with the theories, methods and tools for developing, managing and evolving software products

Characteristics of Software Engineering

- Greater emphasis on systematic development
- Computer assistance for software development (CASE)
- A concentration on finding out the user's requirements
- Formal specification of the requirements of a system
- Demonstration of early version of a system (prototyping)
- Greater emphases on trying to ensure error free code

Practitioner's view of SW Engineering

Is software art or is it engineering?

- It is a combination
- Creativity of art
- Discipline of engineering
- Coming out with multiple options in user interface (art)
- Managing the development process with metrics (engineering)

Software Applications…

Approaches to developing software:

- System Software – Controls operating system behavior
- Real-time Software – Nuclear reactor temperature control
- Business Software – All applications like Billing, Payroll
- Engineering and Scientific software – Simulations and statistical packages (uses input historical data)

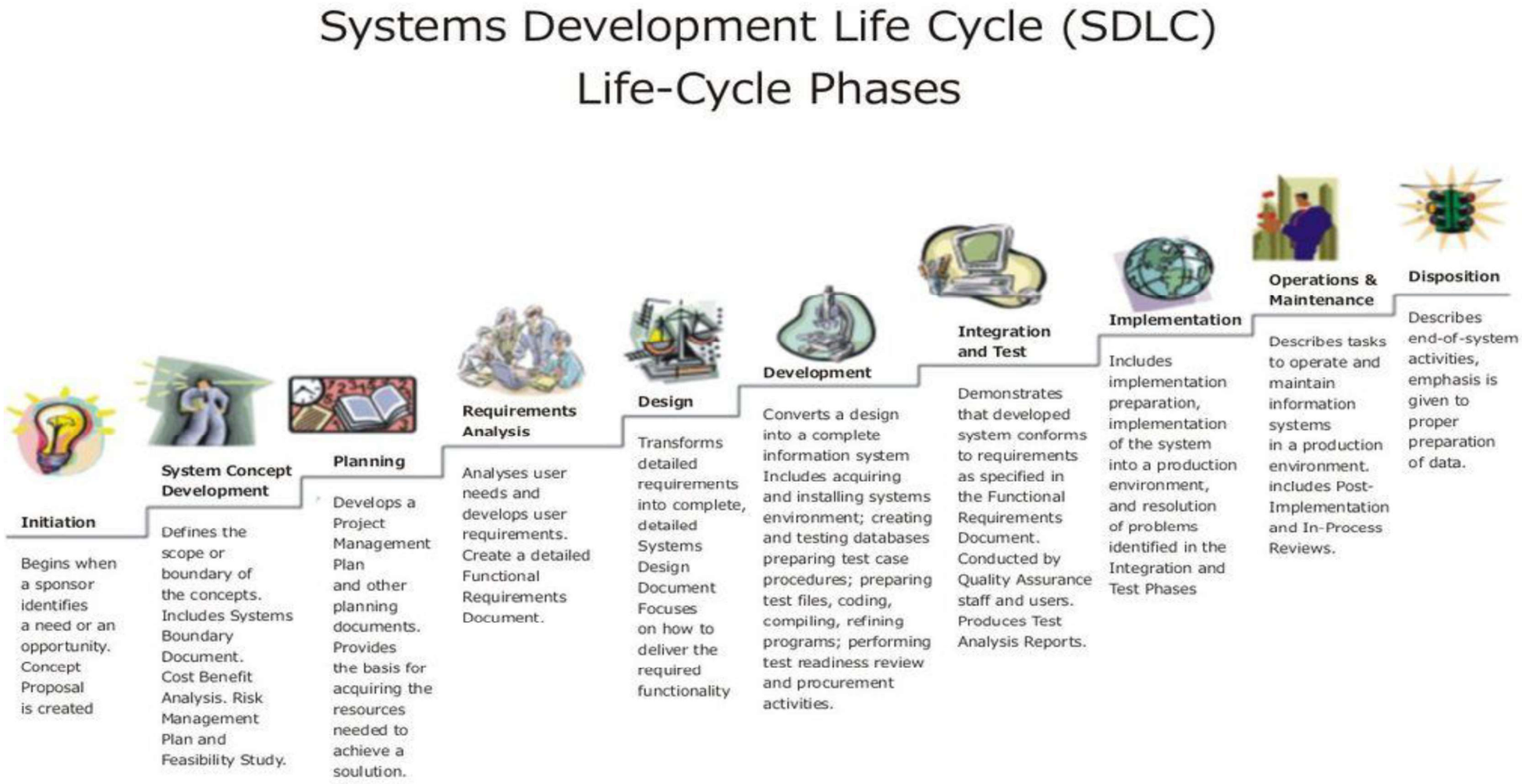
Software Applications Examples

- Embedded Software – Cruise Controls in car
- Personal Computer Software – MS Office
- Artificial Intelligence Software – Robot to develop Photographic Images

What is SDLC ?

The systems development life cycle (SDLC) is a conceptual model used in project management that describes the stages involved in an information system development project, from an initial feasibility study through maintenance of the completed application.

SDLC Life Cycle



SDLC Phases

Planning and Control	This Phase will produce a high-level overview document of the proposed project. It will contain information relating to the project's requirements and will enable the formalization and definition of the scope of the project .	**Input:** Identified Business Need **Deliverables:** Project Charter
Requirement Analysis	This Phase of the SDLC is required to understand and document the users' needs for the system.	**Input:** Project Charter **Deliverables:** Detailed System Analysis (SRS,SFS)
Design	This Phase of the SDLC continues from the Detailed Requirement Analysis and describes how the proposed system will be built.	**Input:** Detailed System Analysis **Deliverables:** Detailed System design (SDS)

SDLC Phases

Build

This Phase addresses the preparation and establishment of the technical environment for development.

Input:

- Detailed System Design

Deliverables:

- System Build
- Developer Guidelines
- Application Forms and Reports.

Test

This Phase of the SDLC is to prepare for and carry out the implementation of the developed system through user and acceptance testing to a full production system

Input:

- System Build

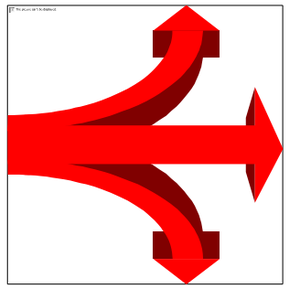
Deliverables:

- User procedures
- Test Plan

Software Development life cycle

Three Identifiable Phases:

1. Definition



2. Development

3. Maintenance

Definition Phase

Focuses on WHAT

- What information to be processed?
- What functions and performances are desired?
- What interfaces are to be established?
- What design constraints exists?
- What validation criteria are required to define a success system

Development Phase

Focuses on

- How the database should be designed ?
- How the software architecture to be designed ?
- How the design will be translated in to a code ?
- How testing will be performed ?

Maintenance Phase

Maintainability is defined as the ease with which software can be understood, corrected, adapted and enhanced”

Maintenance phase focuses on CHANGE that is associated with

- Error correction
- Adaptation required as the software environment evolves
- Enhancements brought about by changing customer requirements
- Reengineering carried out for performance improvements

Software Requirements Analysis …

The process of identifying requirements, current problems, constraints ,Opportunities for improvement , timelines and Resources costs .

- Defining a problem
- Gathering the requirements
- Developing an analysis model representing those requirements
- Describing the problem

Software Requirements Analysis …

It typically includes two types of activity :

Eliciting Requirements: The task of communicating with customers and users to determine what their requirements are. Sometimes called Requirements Gathering

Analyzing Requirements: Determining whether stated requirements are unclear, incomplete, ambiguous or contradictory, and then resolving theses issues

Several Elicitation techniques used are :

- Interviewing
- Meeting
- Questionnaire and Surveys
- Prototyping
- Workshop(JAD)
- Responsive listening
- Use Cases
- QFD
- Brainstorming

Commonly used Requirements Elicitation Techniques - Prototype

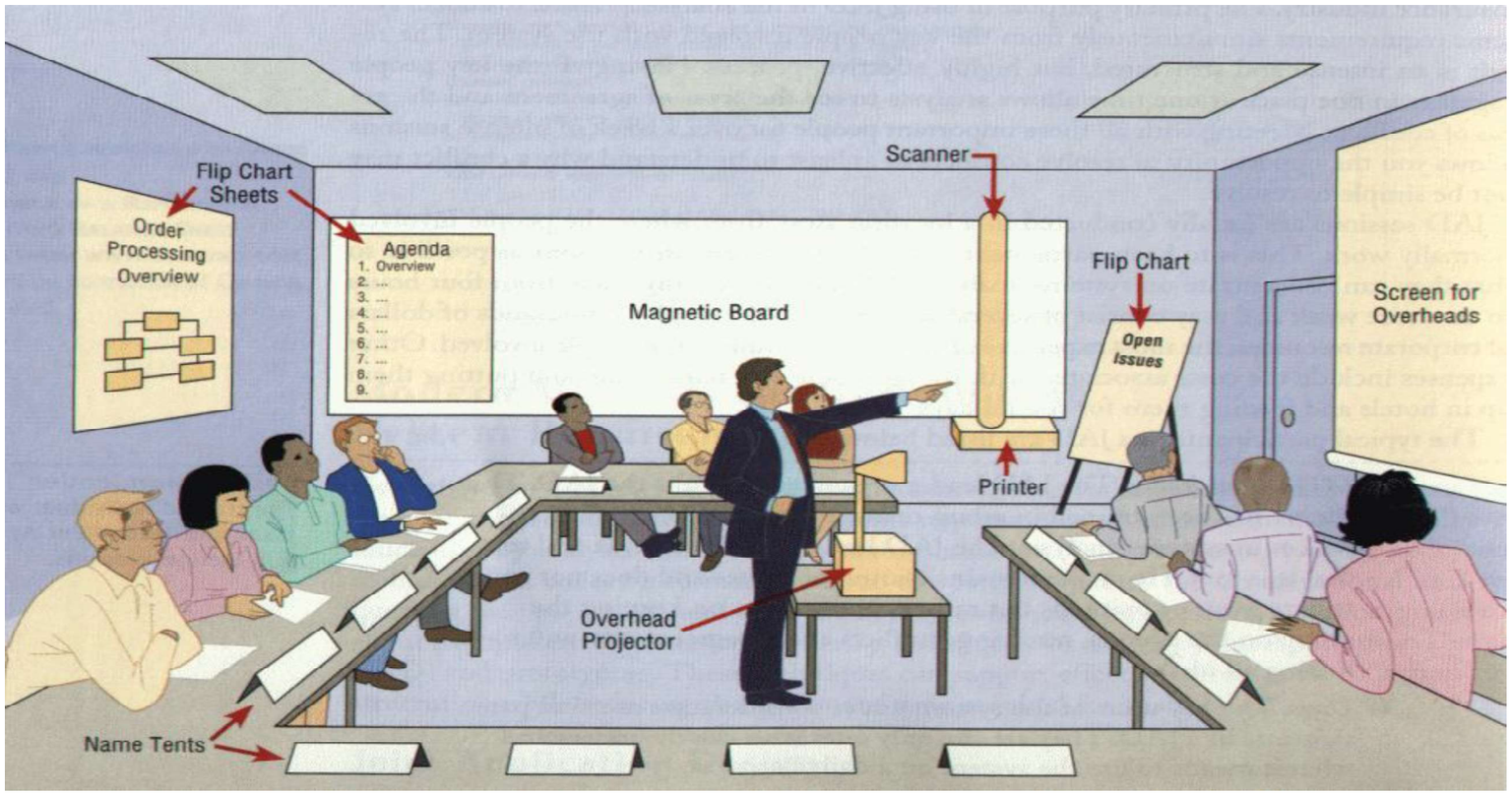
- A software requirements prototype is a partial implementation of a software system, built to help developers, Users, and Customers better understand system requirements
- Used to verify and validate the User Interface design and to confirm usability
- Helps users get an idea what the system will look like,, and makes it easier to make design decisions before the system is built
- Facilitates the communication between users and developers

Commonly used Requirements Elicitation Techniques - FAST

- **F**acilitated **A**pplication **S**pecific **T**echniques (**FAST**) is a technique for requirements elicitation
- The objective is to close the gap between what the developers intend and what users expect
- It is a team oriented approach for gathering requirements
- Meeting between developers and customers.
- A “facilitator” appointed to control the meeting.
- Rules for preparation and participation are established.
- “Definition Mechanisms” (worksheets, flip charts, wall boards...) are used.
- The goal is to identify the problem, propose solution, negotiate different approaches and specify a preliminary set of requirements.

Commonly used Requirements Elicitation Techniques - JAD

- Joint Application Development (JAD) is a modern information-gathering technique for analysis that brings together the key users, managers, and systems analysts in a JAD Location



JAD

- JAD is used to cut the time required by personal interviews, to improve the quality of the results, and to increase end-user participation
- JAD has been used as a technique to accomplish requirements analysis and to design the user interface with users in a group setting
- JAD requires specialized skills by the analyst
- JAD requires a commitment by the organization and users

Typical JAD Session Agenda

- Project leader : Introduce all JAD team members
- Discuss ground rules, goals and objectives for the JAD session
- Top Management (Sponsor): explain the reason for the project and express top management authorization and support
- Users: Provide operational level input on current operations, input and output requirements, user interface issues and how the project will support day to day tasks
- JAD team members: Discuss and document all system requirements, develop models and Prototypes
- Group leaders : Report on results and assigned tasks and topics, present issues that should be addressed by the overall JAD team
- Recorder : Documents results of JAD sessions and works with system analysts to build system models and develop case tool documentation

What is a Requirements

- It may range from a high level abstract statement of a service or of a system constraint to a detailed mathematical functional specification
- Comprehensible from many viewpoints
- Expressed in the terms that the stakeholders understand

Types of Requirements

User Requirements : Are typically those requirements that the end user of the system, manager of the development process, the contractor can understand

- They are typically written in a natural language
- They don't include computer jargons of any kind
- Typically used by the users for sign off, before the development process is to begin

System Requirements(Functional Specification): Is more from a developer's perspective

- Defines a systems function, the services in the system and the constraints.

Example: User Vs System requirement Specification

Requirement: System that is being built has to provide a means of accessing external files created by other tools (software's) which means the operating system in this case, has to be able to provide a means of accessing, opening and viewing external files (jif, mpg, pdf) that are created by other tools.

Example :User Vs System requirement specification (cont...)

User Requirements Definition:

- The software must provide a means of accessing external files created by other tools

System Requirements Specification :

- The user should have a facility to define the type of the external file
- Each external file should have an associated tool which can be applied to view the file
- Each external file must be represented as an icon on the desktop
- When the user selects an icon representing an external file, the effect must be to apply the pre-defined tool and open the file

Functional and Non-functional Requirements

- **Functional Requirement**: Statements of services the system should provide, how the system should react to particular inputs and how the system should behave in particular situations

- **Non Functional Requirements**: Constraints on the services or functions offered by the system such as timing constraints on the development process, standards, etc

Requirement Types

Mainly based on the functionality of the system it is classified as Functional & Non-functional Requirements

Functional Requirements :

- Business Requirements
- User Requirements
- Functional Specifications
- Business Rules
- System Requirements

Non Functional Requirements :

- Quality Attributes
- Interoperability requirements
- constraints

Examples of Functional Requirements

- System date and time should be displayed in the application

- The Application should be able to Add new Accidents when Accident Enter New Accidents button is pressed and when valid data are entered. The Main screen should be navigated when Main screen button is pressed

- In the H&S Application, when "Update Personal Details" Button is clicked, “Add/Amend Dates, Amend Personal Details” screen should be Displayed and the records should be navigated properly when appropriate (Previous and next) buttons are Clicked.

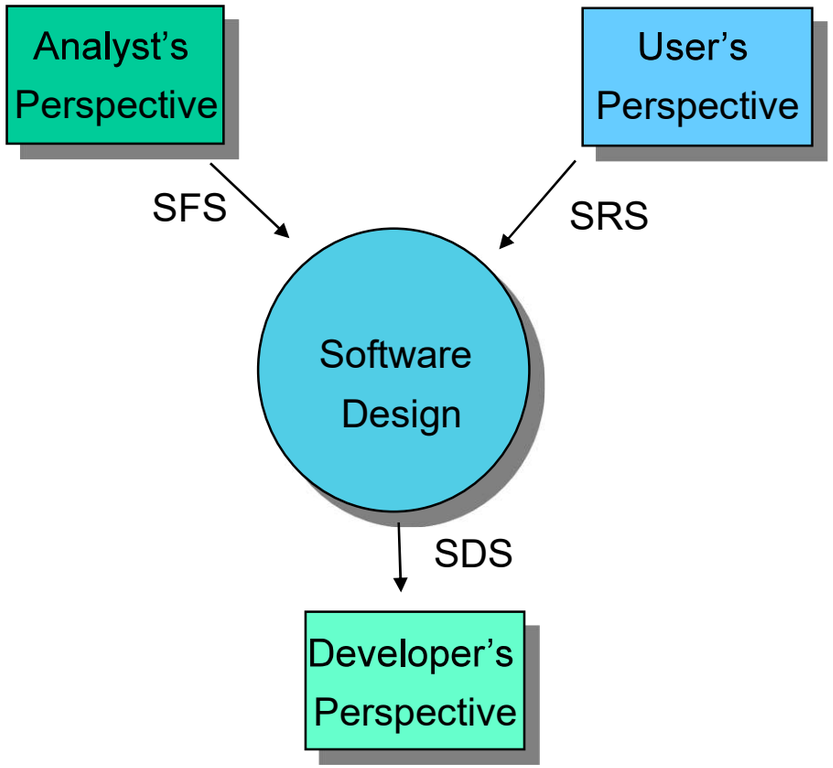
Examples of Non-Functional

- Accidents Menu is displayed when Accidents button is pressed from the Main Menu

- All Web pages generated by the system shall be fully downloadable in no more than 5 seconds over an 80KBps modem connection.

- Responses to queries shall take no longer than 5 seconds to load onto the screen after the user submits the query, while generating reports.

Software Design



- The business of finding a way to meet the functional requirements within the specified constraints using the available technology
- Software Design converts user requirements into a conceptual model which will then be converted into a program..

Design Phase

- Three distinctive aspects of an information system are addressed during its software design
  - Data Design
  - Architectural Design
  - Procedural Design
- Determine the set of instructions that will lead to a solution that meets the customer's requirements.
- This set of instructions is called an algorithm.
- Algorithms are usually represented pictorially (by Flowcharts or UML State Diagrams)
- Architecture, including hardware and software, communication, software design (UML is produced here) are all part of the deliverables of a design phase

Design – HLD (High Level Design)

HLDS (**H**igh **L**evel **D**esign **S**pecifications) contains items in a macro level i.e. from User Point of view

- List of modules and a brief description of each
- Brief functionality of each module
- Interface relationship among modules
- Dependencies between modules
- Database tables identified with key elements
- Overall architecture diagrams along

Design – LLD (Low Level Design)

LLDS (**L**ow **L**evel **D**esign **S**pecifications) / Detailed Design

- Detailed functional logic of the module, in pseudo code
- Database tables, with all elements, including their type and size
- All interface details
- All dependency issues
- Error MSG listing
- Complete input and output format of a module

HLD & LLD Phases put together are called as Design Phase

Coding Phase

- The specifications from the detailed design phase are converted into programs.
- Code is produced from the deliverables of the design phase during implementation
- Design must be translated into a machine-readable form taking input as SRS.

Cross Reference Matrix

- A technique in the form of the matrix to verify the completeness with respect to inputs.
- Typical Cross Reference Matrix is:
  - Between Specs & Design
  - Between Design and Coding
  - Between Specification and Test cases.
- Once the unit functions are identified, it makes sense to prepare Cross Reference Matrix with unit functions as the base.
- CRM keeps growing as one advance in software life cycle

CRM / Traceability Matrix

Identifier	URS/MRS	SRS	SFS	HLD / Design section module	Code Program (LLD)	Functional Test Cases
1	Usecase1	Req1	1.0	module1	program1	testcase1
2	-	-	-	-	-	-
3	usecase3	Req3	3.0	module3	program3	testcase3

Testing Phase

- Finding the hidden defects in developed Product.
- During testing, the implementation is tested against the requirements to make sure that the product is actually solving the needs addressed and gathered during the requirements phase.
- Unit, Integration, system and acceptance tests are done during this phase.
- Unit tests act on a specific component of the system, while system tests act on the system as a whole.

Deployment & Close Phase

- Deploying the product at client side
- Installation and setup of software application at the customer site

Maintenance Phase

Maintaining the software application is:

- Fixing bugs
- Integrate new features requested by customers
- Provide support
- On-going process

Summary

- Definition of Software & software engineering (SE)
- Characteristics of software engineering
- Practitioner’s view of software engineering
- Software applications
- Overview of Life Cycle
- SDLC
- Phases of software Development
- Describe Requirements Engineering Process
- To describe functional and non-functional requirements
- Purpose of each phase and deliverables