# MODULE-2

# MODULE-2

## 2.1. PROCESS FRAMEWORK MODELS

- 2.1.1 Capability maturity model (CMM)
- 2.1.2 ISO 9000

## 2.2 PHASES IN SOFTWARE DEVELOPMENT

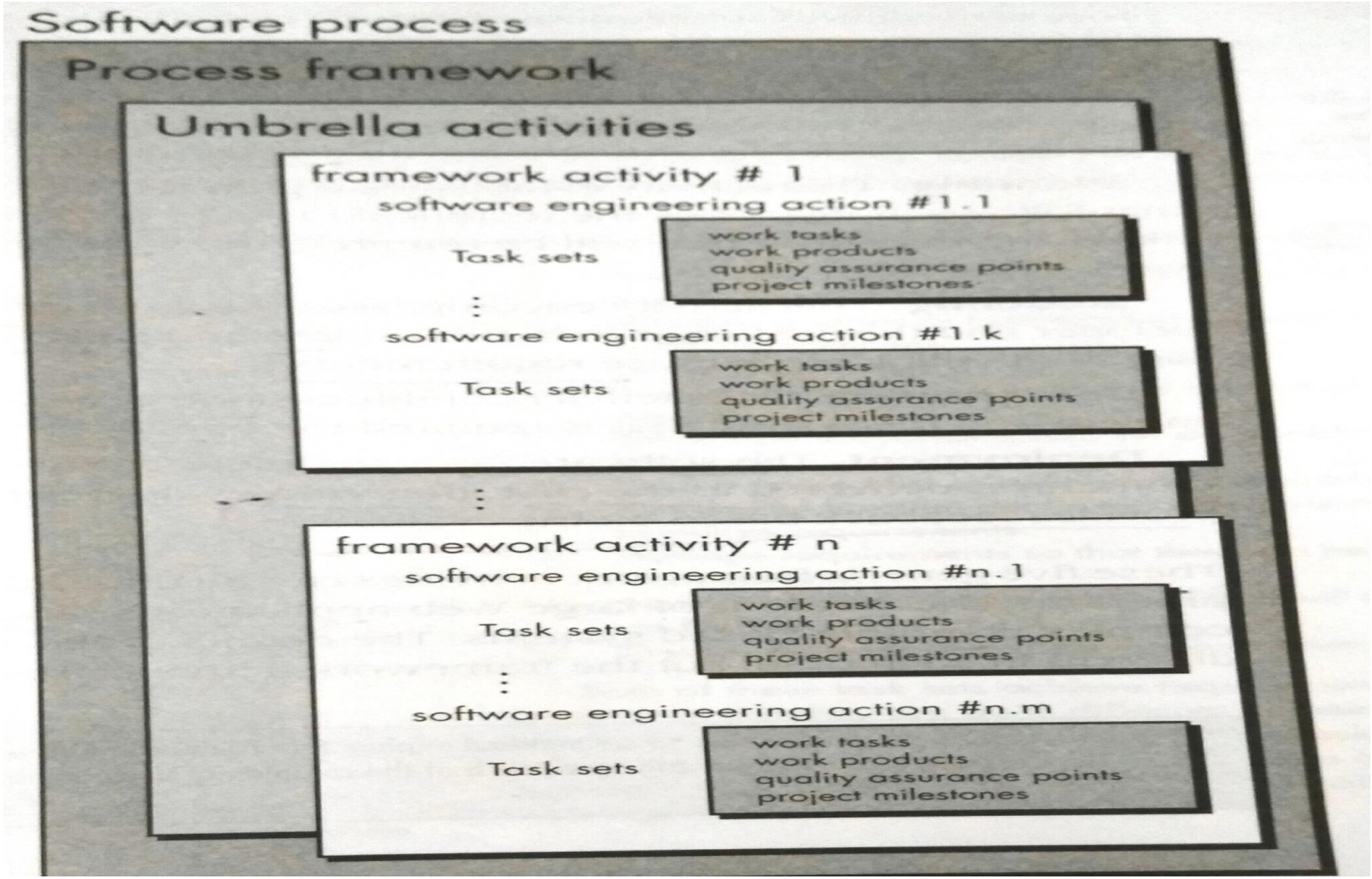
- 2.1 Requirement analysis
- 2.2 Requirements elicitation for software
- 2.3 Analysis principles
- 2.4 Software prototyping
- 2.5 Requirement Specification

# 1. PROCESS FRAMEWORK

# INTRODUCTION

- Process framework establishes the foundation for a complete software process
- It identifies a small no: of framework activities that are applicable to all software projects
- It encompasses a set of umbrella activities, that are applicable to entire s/w process

# SOFTWARE PROCESS FRAMEWORK



# UMBRELLA ACTIVITIES

- **Umbrella activities**
  - It encompasses a set of framework activities
- **Framework activities**
  - Each framework activity consist of a set of software engineering actions
- **Software engineering action:**
  - It is a collection of related tasks set that produces a major s/w engineering work product
  - Eg: Design, Requirement gathering
  - Each engineering action is populated with individual work tasks

- **Task set**
  - It defines the actual work to be done to accomplish the objectives of software engineering action
  - It is a collection of
    - Software engineering work tasks
    - Related work products
    - Quality assurance points
    - Project milestones

# EXAMPLE

- Eg: If Requirement gathering is a software engineering action, then task set of requirement gathering is
  - Make a list of stakeholders of project
  - Invite stakeholders to an informal meeting
  - Ask each stakeholders to make a list of features & functions required
  - Discuss requirements and build final list
  - Prioritize requirements
  - Note the areas of uncertainty

# 5 GENERIC PROCESS FRAMEWORK ACTIVITIES

- These activities are applicable to majority of software projects
- Can be used during development of small pgms, large web applications, complex computer based s/m
- Details of s/w process will be different but the framework activities remains the same
  - **[A] . Communication**
  - **[B] . Planning**
  - **[C] . Modeling**
  - **[D] .Construction**
  - **[E] .Deployment**

# [A]. COMMUNICATION

- This activity involves heavy communication & collaboration with customer & other stakeholders
- Encompasses requirement gathering & other related activities

# [B]. PLANNING

- This activity establishes a plan for the software engineering work
- It describes
  - Technical tasks to be conducted
  - The risks that are likely to happen
  - Resources required
  - Work product to be produced
  - Work schedule

# [C]. MODELING

- This activity encompasses the creation of models
- It allows the developer & customer to understand the software requirements & design
- It is composed of 2 software engineering actions
  - Analysis
  - Design

# [C]. MODELING

- **Analysis**
  - It consists of set of work tasks that lead to the creation of analysis model or requirement specification
  - Set of tasks include
    - Requirement gathering
    - Elaboration
    - Negotiation
    - Specification
    - Validation

# [C]. MODELING

- **Design**
  - Encompasses work tasks that create a design model
  - Tasks are:
    - Data design
    - Architectural design
    - Interface design
    - Component level design

# [D] CONSTRUCTION

- This activity combines the code generation & testing, which is required to uncover the errors in the code

# [E]. DEPLOYMENT

- Software is delivered to the customer who evaluates the delivered product
- Provides feedback based on the evaluation

# GENERIC UMBRELLA ACTIVITIES

1. Software project tracking & control
2. Risk management
3. Software quality assurance
4. Formal technical reviews
5. Measurement
6. Software configuration management
7. Reusability management
8. Work product preparation & production

# SOFTWARE PROJECT TRACKING & CONTROL

- Allows the software team to assess progress of the project, against project plan
- Take necessary action to maintain schedule

# RISK MANAGEMENT

- Assess the risks that may affect the outcome or the quality of the project

# SOFTWARE QUALITY ASSURANCE

- Defines & conducts the activities required to ensure software quality

# FORMAL TECHNICAL REVIEWS

- Assess software engineering work products to uncover & remove errors, before they are propagated to next action or activity

# MEASUREMENT

- This activity defines & collects
  - Process
  - Project and
  - Product measures
- That assist the team in delivering software, that meets customers needs

# SOFTWARE CONFIGURATION MANAGEMENT

- Manages the effects of change throughout the software process

# REUSABILITY MANAGEMENT

- Defines criteria for work product reuse
- Establish mechanisms to achieve reusable components

# WORK PRODUCT PREPARATION & PRODUCTION

- Encompasses the activities required to create work products such as
  - Models
  - Documents
  - Logs
  - Forms
  - Lists

# 1.1. CAPABILITY MATURITY MODEL (CMM)

# INTRODUCTION

- **CMM** is not a software life cycle model
- It is a strategy for improving software process
- Developed by SEI (Software Engineering Institute) in 1986
- **CMM** is used to judge the maturity of the software processes of an organization
- CMM helped organization to improve the quality of software they developed

# INTRODUCTION

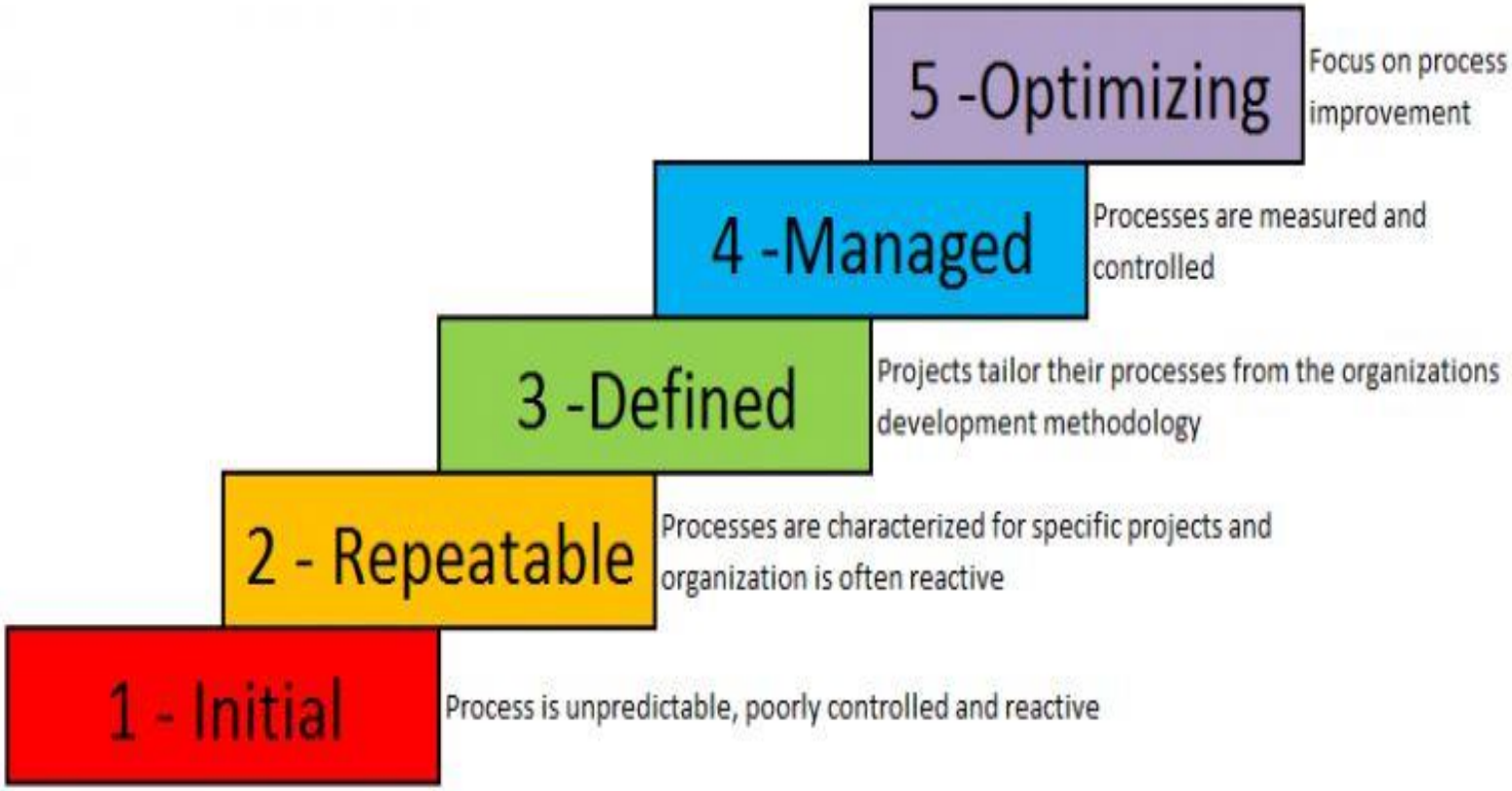
- SEI-CMM can be used in two ways
  - Capability evaluation
  - Software process assessment

- **Capability evaluation**
  - Provide a way to access the software process capability
- **Software process assessment**
  - It is used by an organization with the objective to improve the process capability

# CMM MATURITY LEVELS

- 5 maturity levels
  - **Initial** (maturity level 1)
  - **Repeatable** (maturity level 2)
  - **Defined** (maturity level 3)
  - **Managed** (maturity level 4)
  - **Optimizing** (maturity level 5)

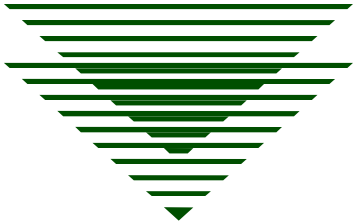
# CMM MATURITY LEVELS



# INITIAL LEVEL-1

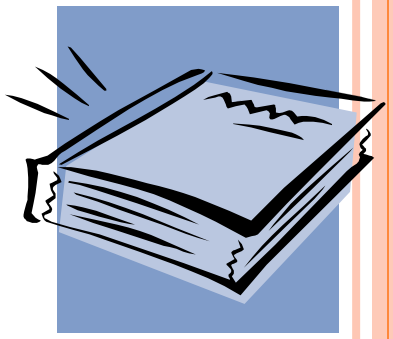
- A software development organization at this level is characterized by ad hoc activities.
- Very few or no processes are defined and followed. Since software production processes are not defined,
- Different engineers follow their own process and as a result development efforts become chaotic. Therefore, it is also called **chaotic level.**
- The success of projects depends on individual efforts and heroics.
- When engineers leave, the successors have great difficulty in understanding the process followed and the work completed.

# REPEATABLE LEVEL-2



- At this level, the basic project management practices such as tracking cost and schedule are established.
- Size and cost estimation techniques like function point analysis, COCOMO, etc. are used.
- The necessary process discipline is in place to repeat earlier success on projects with similar applications.
- Please remember that opportunity to repeat a process exists only when a company produces a family of products.

# DEFINED LEVEL-3

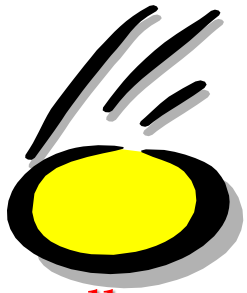


- At this level processes for both management & development activities are defined & documented
- Defining and implementing proven practices throughout the organization
- Increased productivity, efficiency and effectiveness using these practices
- Emergence of training group to provide organization-wide knowledge
- ISO 9000 aims at achieving this level.

# MANAGED LEVEL-4

- Management can effectively control the software development effort using precise measurements.
- At this level, organization set a quantitative quality goal for both software process and software maintenance.
- At this maturity level, the performance of processes is controlled using statistical and other quantitative techniques, and is quantitatively predictable.

# OPTIMIZING LEVEL-5



- The Key characteristic of this level is focusing on continually improving process performance
- through both incremental and innovative technological improvements.
- For example, if from an analysis of the process measurement results, it was found that the code reviews were not very effective and a large number of errors were detected only during the unit testing,
- then the process may be fine tuned to make the review more effective.

# 1.2 ISO 9000

# INTRODUCTION

- International standards organization (ISO) is a association of 60 countries
- It published a 9000 series standards in 1987
- ISO 9000 std specifies the guidelines for maintaining quality of the system
- It addresses operational aspects & organizational aspects
  - Responsibilities
  - Reporting
- It specifies a set of recommendations for quality product development

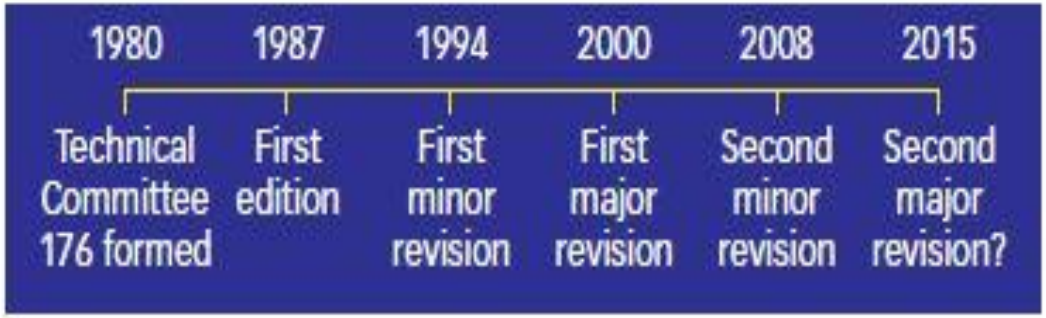
# ISO 9000 STANDARD SERIES

- ISO 9000 is a series of 3 standards
  - ISO 9001
  - ISO 9002
  - ISO 9003
- These series of standards are based on the premise that,
  - If proper process is followed for production, then good quality products are bound automatically

- **ISO 9001**
  - This std is applied to organizations engaged in
    - **Design**
    - **Development**
    - **Production**
    - **Servicing of goods**
  - This std is applicable to most software development organizations
- **ISO 9002**
  - This std is applied to those organizations which do not design products but only involved in production
    - Eg: car manufacturing industries
  - Not applicable to s/w development organizations

- **ISO 9003**
  - Applied to organizations involved in installation & testing of the products

## ISO 9001 timeline / FIGURE 1



# MEA ISO CERTIFIED

UKcert
Certification & Inspection

Certificate of Registration

This is to certify that the
Quality Management System
of

**MEA ENGINEERING COLLEGE**

at

**VENGOOR (PO), PERINTHALMANNA, MALAPPURAM-679 325, KERALA, INDIA.**

has been independently assessed and
is compliant with the requirements of:

**ISO 9001:2015**

For the following scope of activities:

**PROVIDING PROFESSIONAL EDUCATIONAL SERVICES IN ENGINEERING AND TECHNOLOGY AT GRADUATE AND POST GRADUATE LEVEL.**

**Certificate Number: UQ-201812032131**

**Validity of this certificate can be verified at www.ukcertifications.co.uk/verify**

| | |
|---|---|
| Date of Certification | 3rd December 2018 |
| 1st Surveillance Audit Due | 2nd December 2019 |
| 2nd Surveillance Audit Due | 2nd December 2020 |
| Certificate Expiry | 2nd December 2021 |

Authorised Signatory

This certificate is the property of UK Certification & Inspection Limited and shall be returned immediately on request.

# ISO 9000 FOR SOFTWARE INDUSTRY

- It is a generic std applied to variety of industries
- Many clauses in ISO 9000 are written using generic terminologies
- It is very difficult to interpret for Software development organizations
- Reason
  - Development of software products are different from development of other products

# SOFTWARE DEVELOPMENT VS OTHER PRODUCT DEVELOPMENT

- Software is intangible and so its difficult to control
  - It is difficult to control & manage something which cannot be seen
  - Other products can be seen during its development
  - So it is easy to determine how much work is completed & can estimate how much more can be done
- Raw material consumed during software development is data
  - Other products development involve large consumption of raw materials

# ISO 9000-PART3

- Standard used to interpret clauses of ISO 9000 to software industry
  - Clauses of ISO 9000 are concerned with raw material control
  - This has no relevance in software development
- Due to the differences of software development & other pdt development, it is difficult to interpret clauses of ISO in context of software industry
- So ISO released a separate document called ISO 9000 part 3
- This is to interpret ISO std to software industry

# BENEFITS OF ISO CERTIFICATION

- Confidence of customers to an organization increases when organization qualify ISO certification
  - True in international markets
  - Organizations involved in software export takes this certification
- ISO requires a well documented software production process
  - This increases the quality of the product
- ISO makes development process focused, efficient & cost effective
- Points out the weak points of organizations & recommends remedial action

# HOW TO GET ISO CERTIFICATION?

- Organization intending to get ISO certification applies to ISO 9000 registrar for registration
- ISO registration process have following stages
  - Application stage
  - Pre-assessment
  - Document review and adequacy audit
  - Compliance audit
  - Registration
  - Continued surveillance

# HOW TO GET ISO CERTIFICATION?

- **Application stage**
  - Organization applies to registrar for registration
- **Pre-assessment**
  - In this stage, registrar makes a rough assessment of the organizations
- **Document review & adequacy audit**
  - In this stage registrar reviews the documents submitted by the organization
  - Makes suggestions for possible improvements
- **Compliance audit**
  - In this stage, registrar checks whether the suggestions made by it during review have been satisfied or not
- **Registration**
  - Registrar awards ISO 9000 certificate after successful completion of all previous phases
- **Continued surveillance**
  - Registrar continues monitoring the organization periodically

# FEATURES OF ISO 9001 REQUIREMENTS

- **Document control**
  - All document concerned with development of s/w pdt is properly
    - Managed
    - Authorized
    - Controlled
  - This requires a configuration management s/m
- **Planning**
  - Proper plans must be prepared
  - Progress of against this plans must be monitored
- **Review**
  - Important document on all phases must be independently checked & reviewed for correctness & effectiveness
- **Testing**
  - Product is tested against specification
- **Organizational aspects**
  - Organizational aspects like management reporting etc are conducted

# SHORTCOMINGS OFISO 9000 CERTIFICATION

- It does not guarantee that the process is of high quality
  - No guideline is given for defining an appropriate process
- ISO certification is not fool proof
  - No international accreditation agency
  - There can be variations in the norms of awarding certificates among different accreditation agencies
- It does not automatically lead to continuous process improvement (TQM)

# DIFFERENCE BETWEEN ISO AND CMM

| ISO 900(INTERNATIONAL STANDARD ORGANISATION) | CMM (CABABILITY MATURITY MODEL) |
|---|---|
| It applies to any type of industry . | CMM is specially developed for software industry |
| ISO 9000 addresses corporate business process | CMM focuses on the software Engineering activities. |
| ISO 9000 specifies minimum requirement. | CMM gets into technical aspect of software engineering. |
| ISO 9000 restricts itself to what is required. | It suggests how to fulfill the requirements. |
| ISO 9000 provides pass or fail criteria. | It provides grade for process maturity. |

| | |
|---|---|
| ISO 9000 has no levels. | CMM has 5 levels:<br>Initial<br>Repeatable<br>Defined<br>Managed<br>Optimization |
| ISO 9000 does not specifies sequence of steps required to establish the quality system. | It reconnects the mechanism for step by step progress through its successive maturity levels. |

# 2. PHASES IN SOFTWARE DEVELOPMENT

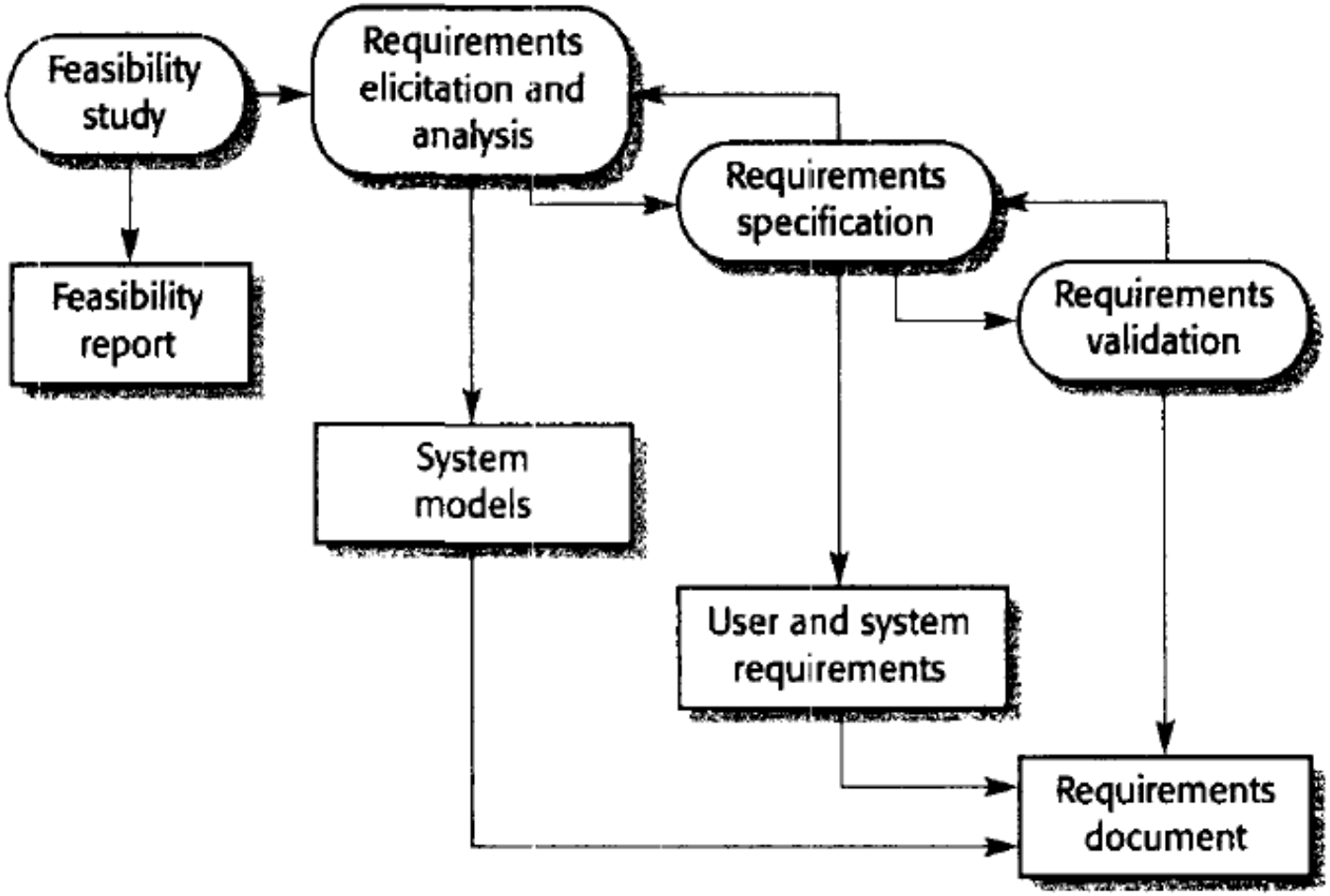
# 2.1 REQUIREMENTS ENGINEERING

# INTRODUCTION

- The goal of the requirements engineering process is to create and maintain a system requirements document.
- The overall process includes four stages
  - **Feasibility study**
  - **Requirement elicitation and analysis**
  - **Requirement specification**
  - **Requirement validation**

# INTRODUCTION

- Feasibility study
  - Checking whether the system is useful to the business
- Requirement elicitation & analysis
  - Discovering requirements
- Requirement specification
  - Converting these requirements into some standard form
- Requirement validation
  - Checking that the requirements actually define the system that the customer wants
- Documents are produced at each stage of the requirements engineering process.


Feasibility study
Requirements elicitation and analysis
Requirements specification
Requirements validation
Feasibility report
System models
User and system requirements
Requirements document

# FEASIBILITY STUDY

- The input to the feasibility study is
  - A set of preliminary business requirements,
  - An outline description of the system
  - How the system is intended to support business process

# AIM OF FEASIBILITY STUDY

- It answers the following questions
  1. Does the system contribute to the overall objectives of the organization?
  2. Can the system be implemented using current technology and within given cost and schedule constraints?
  3. Can the system be integrated with other systems which are already in place?

- Answer to the first question is critical
  - If s/m does not support business objectives, then it has no value in business

# PHASES OF FEASIBILITY STUDY

- **3 phases**
  - Information collection
  - Information assessment
  - Report writing

# INFORMATION COLLECTION

- Information's are collected from various sources
- Information sources can be
  - Managers of the department where the system will be used,
  - Software engineers who are familiar with the type of system that is proposed,
  - Technology experts and end-users of the system

# INFORMATION ASSESSMENT PHASE

- Identifies the information that is required to answer the three questions set out above.
- Talk with information sources to discover the answers to these questions.
- Information sources can be
  - Managers of the department where the system will be used,
  - software engineers who are familiar with the type of system that is proposed,
  - technology experts and end-users of the system

# REPORT WRITING

- After obtaining the information, feasibility study report is written.
- Make a recommendation about whether or not the system development should continue.
- Propose changes to the scope, budget and schedule of the system
- Suggest further high-level requirements for the system.

# 2.2 REQUIREMENT ELICITATION & ANALYSIS

# INTRODUCTION

- After an initial feasibility study, the next stage of the requirements engineering process is requirements elicitation and analysis
- In this activity, software engineers work with customers and system end-users
- Communication is done to find out about
  - Application domain
  - What services the system should provide
  - The required performance of the system
  - Hardware constraints

# STAKEHOLDERS

- Requirements elicitation and analysis may involve a variety of people in an organization
  - The term *stakeholder is used to refer to any person or group who will be affected* by the system, directly or indirectly.
- **Stakeholders include**
  - End-users who interact with the system
  - Everyone else in an organization that may be affected by its installation
  - Engineers who are developing or maintaining related systems
  - Business managers
  - Domain experts
  - Trade union representatives.

# REASONS FOR POOR ELICITATION OF REQUIREMENTS FROM STAKEHOLDERS

- Stakeholders often don't know what they want from the computer system
  - They may find it difficult to articulate what they want the system to do
  - make unrealistic demands because they are unaware of the cost of their requests.
- They express requirements in their own terms with implicit knowledge of their own work.
  - Requirements engineers, without experience in the customer's domain, find these difficult to understand these requirements.

- Different stakeholders have different requirements, & they may express in different ways.
  - Requirements engineers have to consider all potential sources of requirements
  - discover commonalities and conflict.

- Political factors may influence the requirements of the system.
  - managers may demand specific system requirements that will increase their influence in the organization.
- The economic and business environment in which the analysis takes place is dynamic.
  - This changes the analysis process.
  - Hence the importance of particular requirements may change.
  - New requirements may emerge from new stakeholders who were not originally consulted.

# ACTIVITIES OF REQUIREMENT ELICITATION& ANALYSIS

- Requirement discovery
- Requirement classification & organization
- Requirement prioritization & negotiation
- Requirement Documentation

# ACTIVITIES OF REQUIREMENT ELICITATION& ANALYSIS

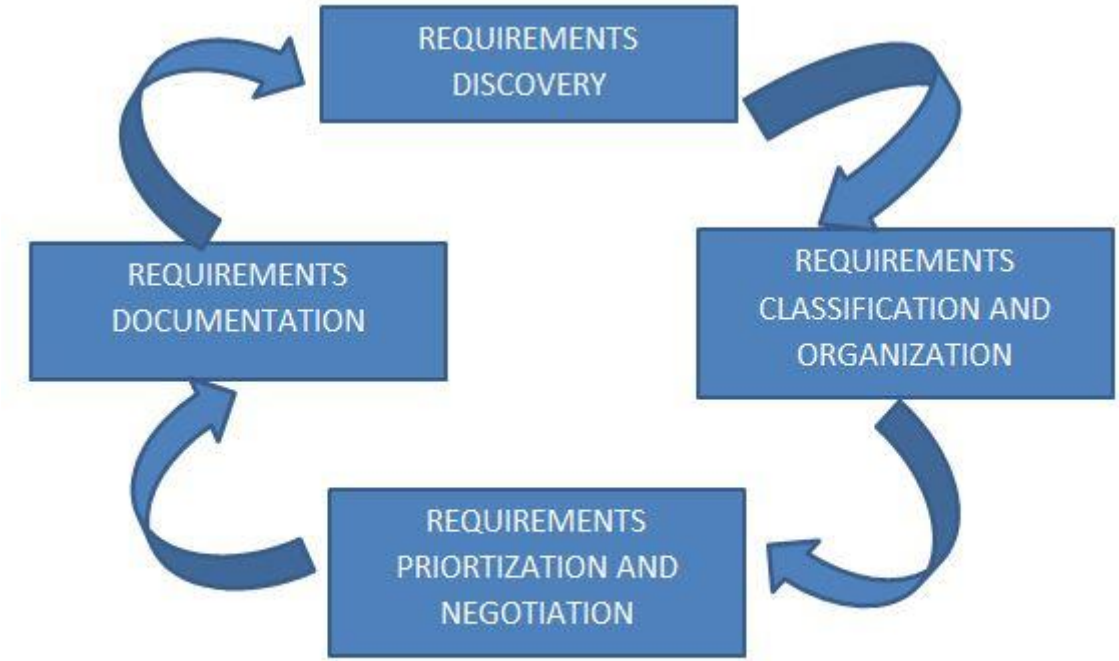



**Fig: Requirement elicitation and analysis process**

# REQUIREMENT DISCOVERY

- This is the process of interacting with stakeholders in the system
- Interaction is done to collect their requirements.
- It is the process of gathering information about the proposed and existing systems
- Analyze the user and system requirements from these information.

# TECHNIQUES/METHORDS FOR REQUIREMENT DISCOVERY

1. Interview
2. Surveys
3. Questionnaires
4. Brainstorming
5. Prototyping
6. Use cases
7. Observation

# 1. INTERVIEW

- Interviews are strong medium to collect requirements. Organization may conduct several types of interviews such as:
- Structured (closed) interviews, where every single information to gather is decided in advance, they follow pattern and matter of discussion firmly.
- Non-structured (open) interviews, where information to gather is not decided in advance, more flexible and less biased.
- Oral interviews
- Written interviews
- One-to-one interviews which are held between two persons across the table.
- Group interviews which are held between groups of participants. They help to uncover any missing requirement as numerous people are involved.

# 2. SURVEYS

- Organization may conduct surveys among various stakeholders by querying about their expectation and requirements from the upcoming system.

# 3. QUESTIONNAIRES

- A document with pre-defined set of objective questions and respective options is handed over to all stakeholders to answer, which are collected and compiled.
- A shortcoming of this technique is, if an option for some issue is not mentioned in the questionnaire, the issue might be left unattended.

# 4. BRAINSTORMING

- An informal debate is held among various stakeholders and all their inputs are recorded for further requirements analysis.

# 5. PROTOTYPING

- Prototyping is building user interface without adding detail functionality for user to interpret the features of intended software product.
- It helps giving better idea of requirements

# 6. USE CASES

- Use cases are a requirements discovery technique that were first introduced in the objectory method and they have now become a fundamental feature of the Unified Modeling Language.(UML)
- In their simplest form, a use case identifies the actors involved in an interaction and names the type of interaction.
- Actors in the process, who may be human or other systems, are represented as stick figures.
- Each class of interaction is represented as a named ellipse.
- Lines link the actors with the interaction.
- Use cases identify the individual interactions between the system and its users or other systems.

# USE CASES

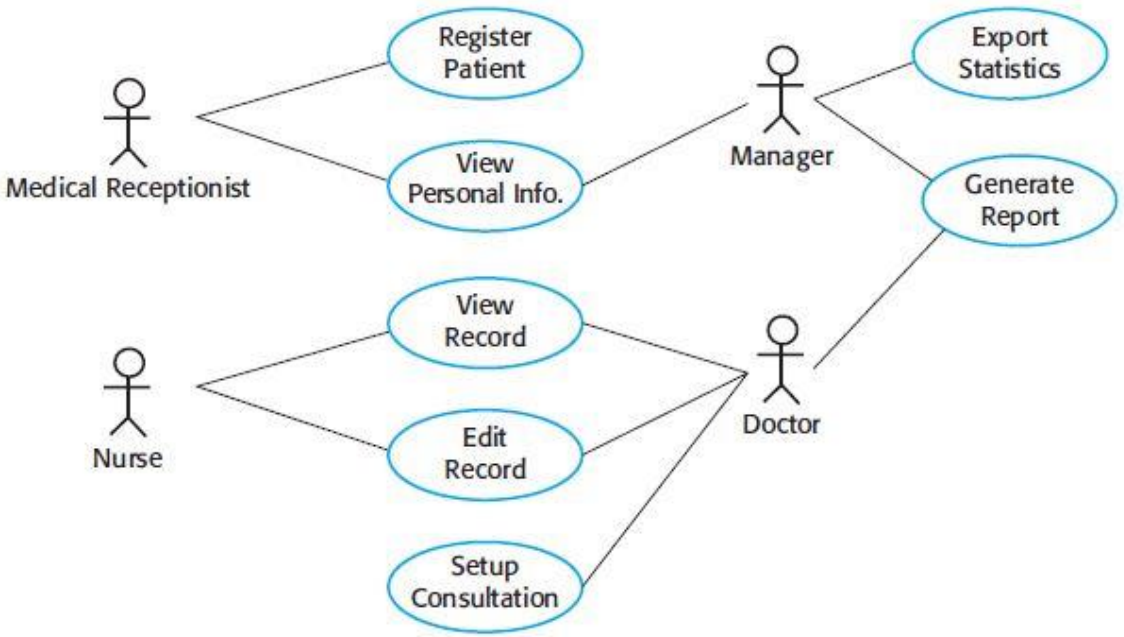



**Figure 4.15** Use cases for the MHC-PMS

# 7. OBSERVATION

- Team of experts visit the client's organization or workplace.
- They observe the actual working of the existing installed systems.
- They observe the workflow at client's end and how execution problems are dealt.
- The team itself draws some conclusions which aid to form requirements expected from the software.

# REQUIREMENT CLASSIFICATION & ORGANIZATION

- This activity takes the unstructured collection of requirements
- Groups the related requirements
- Organizes these requirements into coherent clusters.

# REQUIREMENT PRIORITIZATION & NEGOTIATION

- When multiple stakeholders are involved, requirements will conflict.
- This activity is concerned with prioritizing requirements,
- Find out and resolve requirements conflicts through negotiation.

# REQUIREMENTS DOCUMENTATION

- The requirements are documented
- They are inputted into the next round of the spiral.
- Formal or informal requirements documents may be produced.

# 2.4 SOFTWARE PROTOTYPING

# INTRODUCTION

- The Software Prototyping refers to building software application prototypes which displays the functionality of the product under development, but may not actually hold the exact logic of the original software.
- Software prototyping is becoming very popular as a software development model, as it enables to understand customer requirements at an early stage of development.
- It helps get valuable feedback from the customer and helps software designers and developers understand about what exactly is expected from the product under development.
- Prototype is a working model of software with some limited functionality.

# SOFTWARE PROTOTYPING-TYPES

- Throwaway/Rapid Prototyping
- Evolutionary Prototyping
- Incremental Prototyping

# THROWAWAY/RAPID PROTOTYPING

- Throwaway prototyping is also called as rapid or close ended prototyping.
- This type of prototyping uses very little efforts with minimum requirement analysis to build a prototype.
- Once the actual requirements are understood, the prototype is discarded and the actual system is developed with a much clear understanding of user requirements.

# EVOLUTIONARY PROTOTYPING

- Evolutionary prototyping also called as breadboard prototyping is based on building actual functional prototypes with minimal functionality in the beginning.
- The prototype developed forms the heart of the future prototypes on top of which the entire system is built.
- By using evolutionary prototyping, the well-understood requirements are included in the prototype and the requirements are added as and when they are understood.

# INCREMENTAL PROTOTYPING

- Incremental prototyping refers to building multiple functional prototypes of the various **sub-systems** and then integrating all the available prototypes to form a complete system.

# 2.5 REQUIREMENT SPECIFICATION

# INTRODUCTION

- Requirements
  - The requirements for a system are the descriptions of the services provided by the system and its operational constraints.
  - Requirements reflect the needs of customers for a system
  - It is a detailed formal definition of a system function

# CLASSIFICATION OF REQUIREMENTS

- User requirements
- System requirements

# User Requirements Vs. System Requirements

7

| User Requirements | System Requirements |
|---|---|
| ▪ Written for customers .<br>▪ Statements in natural language.<br>▪ Describe the services that system provides and its operational constraints.<br>▪ May include diagrams or tables.<br>▪ Should describe functional and non-functional requirements.<br>▪ Should be understandable by system users who don't have detailed technical knowledge. | ▪ Statements that set out detailed descriptions of the system's functions, services and operational constraints.<br>▪ Defines what should be implemented so may be part of a contract between client and contractor.<br>▪ Intended to be a basis for designing the system.<br>▪ Can be illustrated using system models. |
| **We provide a definition for a user requirement.** | **We provide a specification for a system requirement.** |

# CLASSIFICATION OF SYSTEM REQUIREMENTS

- Functional requirements
- Non functional requirements

# FUNCTIONAL REQUIREMENT VS NON-FUNCTIONAL REQUIREMENT

| Sl.No | Functional Requirement | Non-functional Requirement |
|---|---|---|
| 1. | Defines all the services or functions required by the customer that must be provided by the system | Defines system properties and constraints e.g. reliability, response time and storage requirements. Constraints are I/O device capability, system representations, etc. |
| 2. | It describes what the software should do. | It does not describe what the software will do, but how the software will do it. |
| 3. | Related to business. For example: Calculation of order value by Sales Department or gross pay by the Payroll Department | Related to improving the performance of the business. For example: checking the level of security. An operator should be allowed to view only my name and personal identification code. |
| 4. | Functional requirement are easy to test. | Nonfunctional requirements are difficult to test |
| 5. | Related to the individual system features | Related to the system as a whole |
| 6. | Failure to meet the individual functional requirement may degrade the system | Failure to meet a non-functional requirement may make the whole system unusable. |

# TYPES OF NON FUNCTIONAL REQUIREMENTS

- Product requirements
- Organizational requirements
- External requirements

# PRODUCT REQUIREMENTS

- Requirements which specify that the delivered product must behave in a particular way
- E.g. execution speed, reliability, etc.

# ORGANIZATIONAL REQUIREMENT

- Requirements which are a consequence of organizational policies and procedures
- E.g. process standards used, implementation requirements, etc.

# EXTERNAL REQUIREMENTS

- They are derived from factors external to the system and its development process.
- They include
- Interoperability requirements
  - that define how the system interacts with systems in other organizations
- Legislative requirements
  - They are followed to ensure that the system operates within the law;
- Ethical requirements.
  - requirements placed on a system to ensure that it will be acceptable to its users and the general public.

# SOFTWARE REQUIREMENT SPECIFICATION DOCUMENT(SRS)

# Introduction

- After requirement gathering & analysis, analyst systematically organize requirements in the form of SRS document
- SRS document contains all the requirements in a structured form
  - Most important document & toughest to write
  - SRS is expected to cater needs of a wide variety of audience

# USERS OF SRS DOCUMENT

- Large no: of people need SRS document for various purposes
  - Users & customers
  - marketing personnel
  - Software developers
  - Test engineers
  - User document writers
  - Project managers
  - Maintenance engineers

# USERS OF SRS DOCUMENT

- **Users & customers**
  - They need SRS to ensure that system described in the document will meet their needs or not
- **Marketing professionals**
  - In case of generic products, Marketing personnel need to understand requirements if they want to explain them to customers.

# USERS OF SRS DOCUMENT

- **Software developers**
  - They refer SRS to make sure that they are designing exactly what is required by the customer
- **Test engineers**
  - They use SRS to understand the functionalities of s/w
  - Based on this they write test cases to validate its working
  - i/p and o/p data is identified precisely

# USERS OF SRS DOCUMENT

- **User documentation writers**
  - They need to read the SRS & understand the features, to prepare the user's manual

- **Project managers**
  - They need to read the SRS so that they can estimate the cost of the project easily
  - SRS contains all information to plan the project

# USERS OF SRS DOCUMENT

- **Maintenance engineers**
  - SRS helps the maintenance engineers to understand the functionalities supported by system
  - This helps them to understand code & design
  - This helps them to determine the modifications to s/m functionalities

# OTHER USES OF SRS

- S/w engineers consider SRS as a reference document
- SRS act as a contract b/w customers and development team
- Used to resolve any disagreements b/w developers & customers
- Used as a legal document to settle disputes

# IMPORTANT USES OF WELL FORMULATED SRS DOCUMENT

- Forms an agreement b/w customers & developers
- Reduces future reworks
- Provides a basis for estimating costs & schedules
- Provides a baseline for validation & verification
- Facilitate future extensions

# CHARACTERISTICS OF A GOOD SRS DOCUMENT

- Concise
- Implementation independent
- Traceable
- Modifiable
- Response to undesirable events
- Verifiable

# CONCISE

- SRS should be concise, unambiguous, consistent and complete
- Verbose & irrelevant descriptions reduce readability & increases possibility of errors

# IMPLEMENTATION INDEPENDENT

- SRS should be free of design & implementation decisions
- It should only specify what the s/m should do & should not state how to do these
- SRS specify only the externally visible behavior of s/m, & not discuss implementation issues
- So it is also called as black box specification of s/w

# TRACEABLE

- It should be possible to trace the specific requirement to design elements that implement it & vice versa
- Traceability is important to verify results of a phase wrt previous phase

# MODIFIABLE

- Customers changes requirements frequently during development
- So SRS undergoes several revisions during development
- To cope with this requirement changes, SRS has to be easily modifiable
- For that SRS should be well structured
- Well structured document is easy to modify & understand

# RESPONSE TO UNDESIRED EVENTS

- SRS should discuss the s/m responses to various undesired events & exceptional conditions that may arise

# VERIFIABLE

- All requirements in SRS document must be verifiable
- It should be possible to design test cases based on the description of functionality

# SOFTWARE REQUIREMENT SPECIFICATION DOCUMENT(SRS)

## SOFTWARE REQUIREMENT SPECIFICATION (SRS)

- IEEE defines SRS as a '*document that clearly and precisely describes each of the essential requirements [functions, performance, design constraints and quality attributes] of the software and the external interfaces*'.

# SOFTWARE REQUIREMENT SPECIFICATION DOCUMENT(SRS)

- Parts of a SRS Documents are

  - **Functional requirements**
  - **Non-functional requirements**
  - **Goal of implementation**

# FUNCTIONAL REQUIREMENTS

**Functional Requirements:** Those functionalities required by the users from the system or those functionalities that can be expressed in terms of functions. Functional requirement specify which output should be produced from the given inputs. They describe the relationship between the input and output of the system.

# FUNCTIONAL REQUIREMENTS

**Example 1:** Consider the case of an online calculator: The various functional requirements may be:

i) Addition
ii) Subtraction
iii) Multiplication
iv) Division

**Example 2:** Consider the case of an ATM software: The various functional requirements are

i) Withdrawal
ii) Balance check
iii) Pin change
iv) Fund transfer
v) Print mini statement etc.

# FUNCTIONAL REQUIREMENTS

**Example 3:** Consider the case of a Library management software: The various functional requirements are

1. Register member
2. Register book
3. Issue book
4. Search a book
5. Return book
6. Reserve a book
7. Display available books
8. Display registered members
9. Display the list of borrowed books
10. Display the list of available books etc.

**Example 4:** Consider the case of a Music player software: The various functional requirements are

i) Play a song
ii) Stop playing song
iii) Pause playing song
iv) Shuffle the playlist
v) Fast forward
vi) Reduce volume level etc.

# NON-FUNCTIONAL REQUIREMENTS

- **Non –Functional Requirements:** These are the various abilities / features possessed by a system or those characteristics of a system which cannot be expressed as functions.

  Example: Scalability, maintainability, portability, reliability, availability, capacity, security, usability etc.
- **Goal of implementation:** This part of SRS, documents issues such as revisions to the system functionalities that may be required in future, new devices to be supported in the future, reusability issues etc.

# GOAL OF IMPLEMENTATION

- **Goal of implementation:** This part of SRS, documents issues such as revisions to the system functionalities that may be required in future, new devices to be supported in the future, reusability issues etc.

# SRS Template

Table of Contents
Revision History

**1. Introduction**
1.1 Purpose
1.2 Document Conventions
1.3 Intended Audience and Reading Suggestions
1.4 Project Scope
1.5 References

**2. Overall Description**
2.1 Product Perspective
2.2 Product Features
2.3 User Classes and Characteristics
2.4 Operating Environment
2.5 Design and Implementation Constraints
2.6 User Documentation
2.7 Assumptions and Dependencies

**3. System Features**
3.1 System Feature 1
3.2 System Feature 2 (and so on)

**4. External Interface Requirements**
4.1 User Interfaces
4.2 Hardware Interfaces
4.3 Software Interfaces
4.4 Communications Interfaces

**5. Other Nonfunctional Requirements**
5.1 Performance Requirements
5.2 Safety Requirements
5.3 Security Requirements
5.4 Software Quality Attributes

**6. Other Requirements**
Appendix A: Glossary
Appendix B: Analysis Models
Appendix C: Issues List

# EXAMPLE

- An Engineering college decided to automate their **library** procedures. Prepare **Software Requirement Specification** for the proposed software in IEEE format.

## SRS DOCUMENT FOR LIBRARY MANAGEMENT

**Introduction**

**1.1 Purpose**

The main objective of this document is to illustrate the requirements of the project Easy Library. The document gives the detailed description of the both functional and non-functional requirements of this project. The purpose of this project is to provide a friendly environment to maintain the details of books and library members. The main purpose of this project is to maintain easy circulation system using computers and an android mobile app through which users can easily search and reserve the books they needed without physically visiting the library frequently.

**1.2 Document Conventions**

IEEE standard are followed

**1.3 Intended Audience and Reading Suggestions**

System analysis, Designer, Software architect, Developer, Tester, Maintenance engineer, Customer and all who directly or indirectly involved with this project.

### 1.4 Product Scope

Easy Library is basically updating the manual library system into an internet-based application which has its own mobile (android) application so that the users can know the availability of the books they require, and can reserve them online through their mobile phones. The project

is specifically designed for the use of librarians and library users. The product will work as a complete user interface for library management process and library usage from ordinary users. Easy Library can be used by any existing or new library to manage its books and book borrowing, insertion and monitoring. It is especially useful for any educational institute where modifications in the content can be done easily according to requirements.

The project can be easily implemented under various situations. We can add new features as and when we require, making reusability possible as there is flexibility in all the modules.The language used for developing the project is Java and Android as it is quite advantageous than other languages in terms of performance, tools available, cross platform compatibility, libraries, cost (freely available), and development process.

### 1.5 References

Books, Websites

**Product Perspective**

The main objective of this document is to illustrate the requirements of the project Easy Library. It is actually a replacement of existing software. Certain modifications had been undergone in the existing software. The existing software does not give the privilege of its own mobile application. The graphical user interface of the existing system is not user friendly. It comprises of unwanted functionalities that makes the system complex. The proposed system will provide search functionality to the users to facilitate the search of resources. This search will be based on various categories viz. book name or ISBN. And also provide option to download eBooks. Further the library staff personnel can add/update the resources and the resource users from the system. The users of the system can request issue/renew/return of books for which they would have to follow certain criteria.

**User Classes and Characteristics**

There are two kinds of users for this product. The users include:

Library Members who will be using the above features by accessing the Library online through their mobile app.

Librarian who will be acting as the controller and he will have all the privileges of an administrator.

**System Modules:**

The system consist of two modules:

**Admin module:** Which includes the functionalities of librarian.

**User module:** Which includes the functions of library members

System Features

## 3.1 FUNCTIONAL REQUIREMENTS

**FR1:** Adding a new member

**Description:** This function is handled by admin user by entering the user details.

**Input:** User details

**Output:** Response message

**FR2:** Search a book

**Description:** This function is performed by user to check whether a given book is there or not by entering either book name or author name.

**Input:** Book name or author name

**Output:** Response message

**FR3:** Return a book

**Description:** This function includes both user and admin. The user will return a book with its book id. The admin updates the database and check for due amount.

**Input:** Book id

**Output:** Response message

**NON- FUNCTIONAL REQUIREMENTS**

**NFR1: Reliability**: The system should always perform its intended functionality.

**NFR2: Availability**: The system should be always available (24x7) for performing its intended functions.

**NFR3: Portability**: The system should be able to run indifferent operating environments.

**NFR4: Scalability:** The system should be able to expand its service capabilities.

## Other Requirements

**4.1 Communication:** The system should have a high speed internet connection with a minimum speed of 100MBps.

**4.2 Hardware requirements:**

RAM: 4 GB (minimum)

HDD: 500 GB (minimum)

**4.3 Software requirements**

OS: Windows 7 or higher / Linux

## Appendix