CHAPTER -4 (Requirement Analysis and Specification)

1) What is Requirement Engineering? Explain Requirement Engineering tasks in detail .

- "The broad spectrum of tasks and techniques that lead to an understanding of requirement is called requirement engineering."
- Requirement engineering begins with communication and continues to modeling activity. it builds a bridge to design and construction.
- Requirements engineering provides the appropriate mechanism for understanding what the customer wants, analyzing need, assessing feasibility, negotiating a reasonable solution
- Specifying the solution unambiguously, validating the specification, and managing the requirements as they are transformed into an operational system.

Requirement Engineering tasks.

Inception

- Inception means specifying beginning of project. Most of software project gets started because of business need or when new market is discovered.
- Here stakeholders from business community define business case for idea and try to find breadth and depth of market.
- Intent is to establish a basic understanding of the problem and find nature of solution.

Elicitation

- In Elicitation requirements are elict from customers, users and others. Also it will be find out from customers, users and others what are the product objectives what is to be done to accomplish these objectives.
- Moreover it is necessary to know how product fits into business needs, and how the product is used on a day to day basis.
- Requirement elicitation is difficult due to following reasons.
  Problems of scope.
  Problems of understanding.
  Problems of volatility.

Elaboration

- The information obtained from the customer during inception and elicitation is expanded and refined during elaboration.
- Elaboration Focuses on developing a refined technical model of software functions, features and constraints.
- It is driven by the creation and refinement of user scenarios.

Negotiation

- In negotiation phase reconciling of conflicting requirement need to be carried out. Here customer asked to rank their requirements according to their priority.
- Using iterative approach requirements are prioritized.
- At the end there's no winner and loser after completion of negotiation activity.

Specification

- Specification means different thing to different people.
- It can be written document, a set of graphical models, a formal mathematical model, a collection of usage scenarios, a prototype, or any combination of these.
- It is the final work product produced by the requirements engineer.

- It serves as the foundation for subsequent software engineering activities.
- It describes the function and performance of a computer-based system and the constraints that will govern its development.

Validation

- Work products produced are assessed for quality in this step.
- A task which examines the specification to ensure that all software requirements have been stated unambiguously.
- That inconsistencies, omissions and errors have been detected and corrected.
- That work products conform to the standards established for the process, project and the product.

Requirements Management

- During requirements management, the project team performs a set of activities to identify, control, and track requirements and changes to the requirements at any time as the project proceeds
- Each requirement is assigned a unique identifier
- The requirements are then placed into one or more traceability tables
- These tables may be stored in a database that relates features, sources, dependencies, subsystems, and interfaces to the requirements.

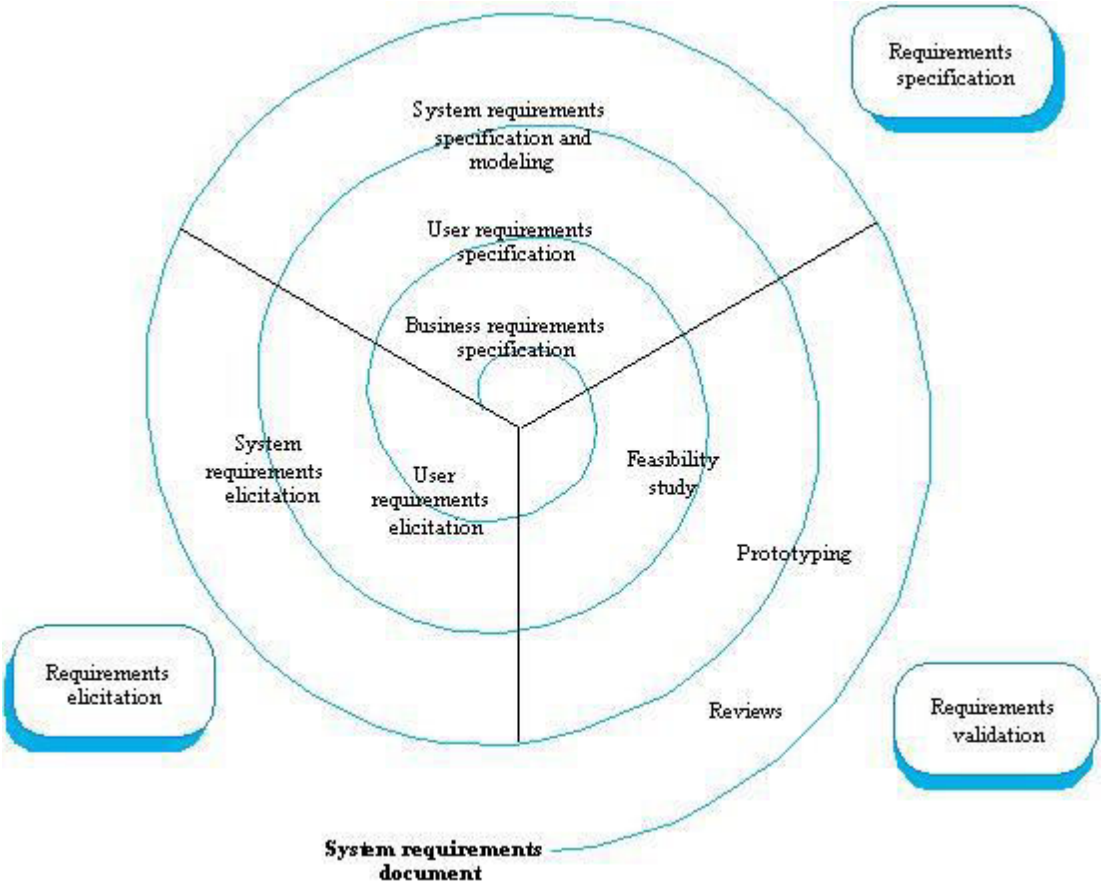
2) Explain System Requirement Specification.

- The Software Requirements Specification is produced at the culmination of the analysis task.
- The National Bureau of Standards, IEEE (Standard No. 830-1984), and the U.S. Department of Defense have all proposed candidate formats for software requirements specifications.
- The Introduction of the software requirements specification states the goals and objectives of the software, describing it in the context of the computer-based system.
- Actually, the Introduction may be nothing more than the software scope of the planning document.
- The Information Description provides a detailed description of the problem that the software must solve. Information content, flow, and structure are documented. Hardware, software, and human interfaces are described for external system elements and internal software functions.
- A description of each function required to solve the problem is presented in the Functional Description. A processing narrative is provided for each function, design constraints are stated and justified, performance characteristics are stated, and one or more diagrams are included to graphically represent the overall structure of the software and interplay among software functions and other system elements.
- The Behavioral Description section of the specification examines the operation of the software as a consequence of external events and internally generated control characteristics.
- Validation Criteria is probably the most important and, ironically, the most often neglected section of the Software Requirements Specification.
- Finally, the specification includes a Bibliography and Appendix. The bibliography contains references to all documents that relate to the software. These include other software engineering documentation, technical references, vendor literature, and standards.
- The appendix contains information that supplements the specifications. Tabular data, detailed description of algorithms, charts, graphs, and other material are presented as appendixes.
- In many cases the Software Requirements Specification may be accompanied by an executable prototype or a Preliminary User's Manual. The Preliminary User's Manual presents the software as a black box.

3) Explain Requirement Engineering Processes.

A Requirement Engineering is a process in which various activities such as discovery, analysis and validation of system requirements are done.

- It begins with feasibility study of the system and ends up with requirement validation. Finally the requirement document has to be prepared.
- This process is a three stage activity where the activities are arranged in the iterative manner. In the early stage of this process most of the time is spent on understanding the system by understanding the high-level non functional requirements and user requirements.
- The spiral model of requirement engineering process is shown here.



The generic activities that are common to all processes are

- Requirement elicitation
- Requirement analysis.
- Requirement validation
- Requirement management.
- Requirement engineering process can also be viewed as structured analysis method in which the system is analyzed fully some system model are prepared.
- Particularly use cases are developed which help in exposing the functionalities of the systems.
- Along with creation of system models some additional information is also provided in the requirement engineering.

4) Explain Requirement Validations

- When any model of software is created at that time it is examined for inconsistency, ambiguity etc. the

requirements can be prioritized by the stake holders and grouped within requirement package that will be implemented by software packages.

- It is one process in which will check about gathered requirements whether it represents the same system or not.
- Here any kind of generated errors can be fixed. If any person can be successful to fix the requirement error then it is more helpful because fixing an requirement errors after delivery may cost up to 100 times the cost of fixing an implementation error.
- Requirement checking can be done in following manner.

Validity checks

- A user may think that a system is needed to perform certain functions. However, further thought and analysis may identify additional or different functions that are required. Systems have diverse stakeholders with distinct needs, and any set of requirements is inevitably a compromise across the stakeholder community.

Consistency checks

- Requirements in the document should not conflict. That is, there should be no contradictory constraints or descriptions of the same system function.

Completeness checks

- The requirement document should include requirements, which define all functions, and constraints intended by the system user.

1. **Realism checks** –Using knowledge of existing technology, the requirements should be checked to ensure that they could actually be implemented. These checks should also take account of the budget and schedule for the system development.
2. **Verifiability** – To reduce the potential for dispute between customer and contractor, system requirements should also be written so that they are verifiable this means that you should be able to write a set of tests that can demonstrate that the delivered system meets each specified requirement.

Requirement validation techniques

– 1. **Requirements reviews**

- Requirement review is a systematic manual analysis of the requirements the requirement review should be taken only after formulation of requirement definition. And both the customer and contractor staff should be involved in reviews. Reviews may be formal (with completed documents) or informal.
- Good communications should take place between developers, customers and users such a healthy communication helps to resolve problems at an early stage.

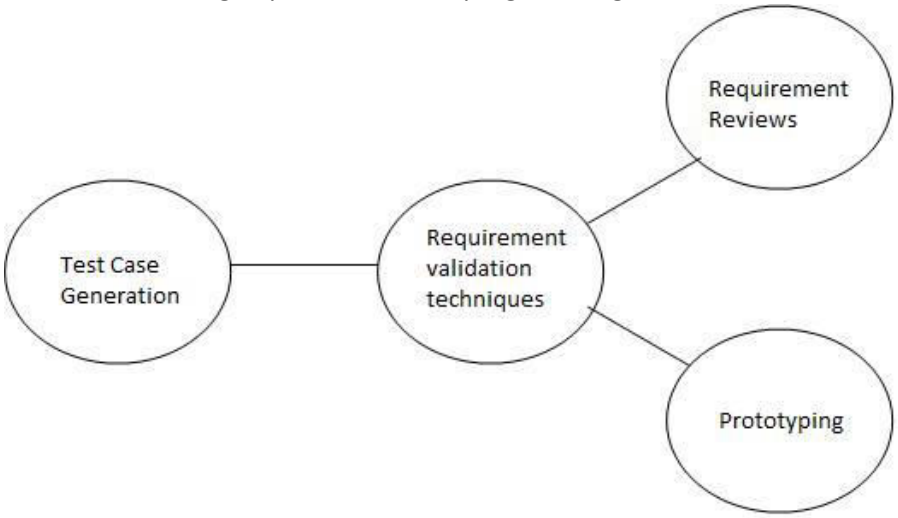
2. **Prototyping**

- The requirements can be checked using executable model of system

3. **Test-case generation**

- Requirements should be testable. If the tests for the requirements are devised as part of the validation process, this often reveals requirements problems.
- If a test is difficult or impossible to design, this usually means that the requirements will be difficult

to implement and should be reconsidered. Developing tests from the user requirements before any code is written is an integral part of extreme programming.



5) How to collect requirement? Explain different methods to collect requirement. What is its importance in Software Engineering?

Requirement Collection

- In requirements engineering, requirements elicitation is the practice of collecting the requirements of a system from users, customers and other stakeholders.
- The practice is also sometimes referred to as requirements gathering.
- The term elicitation is used in books and research to raise the fact that good requirements cannot just be collected from the customer, as would be indicated by the name requirements gathering.
- Requirements elicitation is non-trivial because you can never be sure you get all requirements from the user and customer by just asking them what the system should do.
- Requirements elicitation practices include interviews, questionnaires, user observation, workshops, brain storming, use cases, role playing and prototyping.
- Before requirements can be analyzed, modelled, or specified they must be gathered through an elicitation process.
- Requirements elicitation is a part of the requirements engineering process, usually followed by analysis and specification of the requirements.
- Commonly used elicitation processes are the stakeholder meetings or interviews.
- For example, an important first meeting could be between software engineers and customers where they discuss their perspective of the requirements.

Different methods for collecting the requirements:

1. Collaborative Requirement Gathering

Many different approaches to collaborative gathering have been proposed such as,

- Meetings are conducted and attended by both software engineers and other stakeholders.
- Rules for preparation and participation are established.
- An agenda is suggested that is formal enough to cover all important points but informal enough to encourage the free flow of ideas.

- A “facilitator” (can be a customer, a developer, or an outsider) controls the meeting.
- A “definition mechanism” (can be work sheets, flip charts, or wall stickers or an electronic bulletin board, chat room, or virtual forum) is used.

2. Quality Function Deployment

- Quality Function Deployment (QFD) is a quality management technique that translates the needs of the customer into technical requirements for software. QFD “concentrates on maximizing customer satisfaction from the software engineering process”. So, QFD identifies three types of requirements.

Normal Requirements. The objectives and goals that are stated for a product or system during meetings with customers. If these requirements are present, the customer is satisfied.

Expected Requirements. These requirements are implicit to the product or system and may be so fundamental that the customer does not explicitly state them. Their absence will be cause for significant dissatisfaction.

Exciting Requirements. These features go beyond the customer’s expectations and prove to be very satisfying when present.

3. Usage Scenarios.

- As requirements are gathered, an overall vision of system functions and features begins to materialize.
- However, it is difficult to move into more technical software engineering activities until you understand how these functions and features will be used by different classes of end users.
- To accomplish this, developers and users can create a set of scenarios that identify a thread of usage for the system to be constructed.

4. Elicitation Work Products.

The work products produced as a consequence of requirements elicitation will vary depending on the size of the system or product to be built. For most systems, the work products include.

- A statement of need and feasibility.
- A bounded statement of scope for the system or product.
- A list of customers, users, and other stakeholders who participated in requirements elicitation.
- A description of the system’s technical environment.
- A list of requirements (preferably organized by function) and the domain constraints that applies to each.
- A set of usage scenarios that provide insight into the use of the system or product under different operating conditions.
- Any prototypes developed to better define requirements.

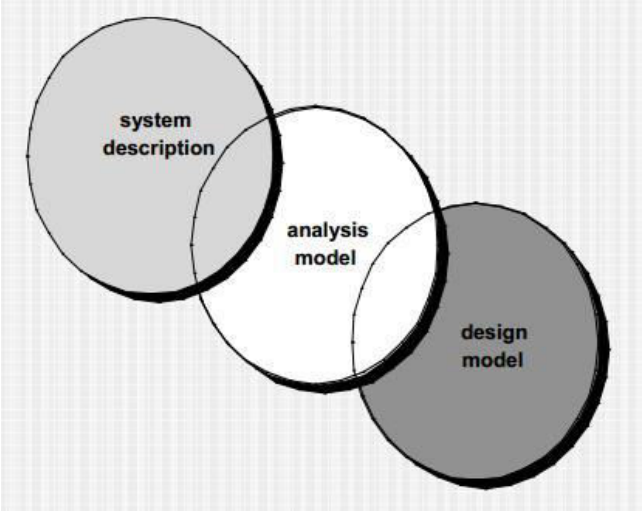
Each of these work products are reviewed by all people who have participated in requirement elicitation.

6) Explain requirement Analysis.

Requirement analysis can be defined it as follows.

“Requirements analysis is a software engineering task that bridges the gap between system level requirements engineering and software design”.

- Requirements engineering activities result in the specification of software's operational characteristics indicate software's interface with other system elements, and establish constraints that software must meet.
- Requirements analysis allows the software engineer to refine the software allocation and build models of the data, functional, and behavioural domains that will be treated by software.
- Software requirements analysis may be divided into five areas of effort:



(1) problem recognition
(2) evaluation and synthesis
(3) modeling
(4) Specification
(5) Review

- Each of these tasks serves to describe the problem so that an overall approach or solution may be synthesized.
- Once problems have been identified, the analyst determines what information is to be produced by the new system and what data will be provided to the system.
- For instance, the customer desires a daily report that indicates what parts have been taken from inventory and how many similar parts remain. The customer indicates that inventory clerks will log the identification number of each part as it leaves the inventory area.
- Upon evaluating current problems and desired information (input and output), the analyst begins to synthesize one or more solutions. To begin, the data objects, processing functions, and behaviour of the system are defined in detail. Once this information has been established, basic architectures for implementation are considered.
- A database management system would seem to be required, but is the user/customer's need for associatively justified? The process of evaluation and synthesis continues until both analyst and customer feel confident that software can be adequately specified for subsequent development steps.

7) A Library lends books and magazines to member, who is registered in the system. Also it handles the purchase of new titles for the Library. Popular titles are bought into multiples copies. Old books and magazines are removed when they are out or date or in poor condition. A member can reserve a book or magazine that is not currently available in the library, so that when it is returned or purchased by the library, that person is notified. The library can easily create, replace and delete information about the tiles, members, loans and reservation in the system. Prepare Software Requirement Specification and Use Case Diagram.

SRS for Library Management System:

Table of contents

1. Introduction.

1.1. Purpose

This document gives detailed functional and non functional requirements for the Library management

system. This product will support lot of flexibility.

1.2. Scope

This product will automate the Library system.

1.3. Overview

This system provides an easy solution

1.4. Additional information

This system will work together with Library computer. It will not be operated independently. Various computers in library might be networked together.

2. General description:

Using Library System one can issue and inquire about the various statuses of the books.

3. Functionality Requirement:

This section provides the requirement of product. The project will require the PHP as a front end and at back end the database MYSQL will be running. Various functions are:

1. Login
2. Validation
3. Get book information
4. Issue the book.
5. Return the book.
6. Manage membership details.
7. Report Generation

4. Interface Requirements.

The user interface must be highly intuitive or interactive because there will not be limited assistance for the user who is operating the Library system.

4.1. Hardware Interface:

4.1.1. The barcode readers which can read the barcodes.

4.1.2. Student Identity cards for verification.

4.1.3. Student library cards for issue copy.

4.2. software interface:

4.2.1. Any windows operating system.

4.2.2. The PHP must be installed.

4.2.3. For database handling MYSQL must be installed.

4.2.4. The final application must be packaged in a set up program, so that the product can be easily installed on machine.

5. Performance Requirements:

None.

6. Design constraints:

None.

7. Other Non-functional Attributes:

7.1 Security
This application will be password protected.

7.2 Reliability
This application should be highly reliable.

7.3 Availability
Any information about the students or books should be quickly available from any computer to authorised user.

7.4 Maintainability:
The application should be maintainable in such a manner that if any new requirement occurs then it should be easily incorporated in an individual module.

7.5 Portability:
The application should be portable on any windows based system.

8. Preliminary schedule:
This product must be completed within 7 months.

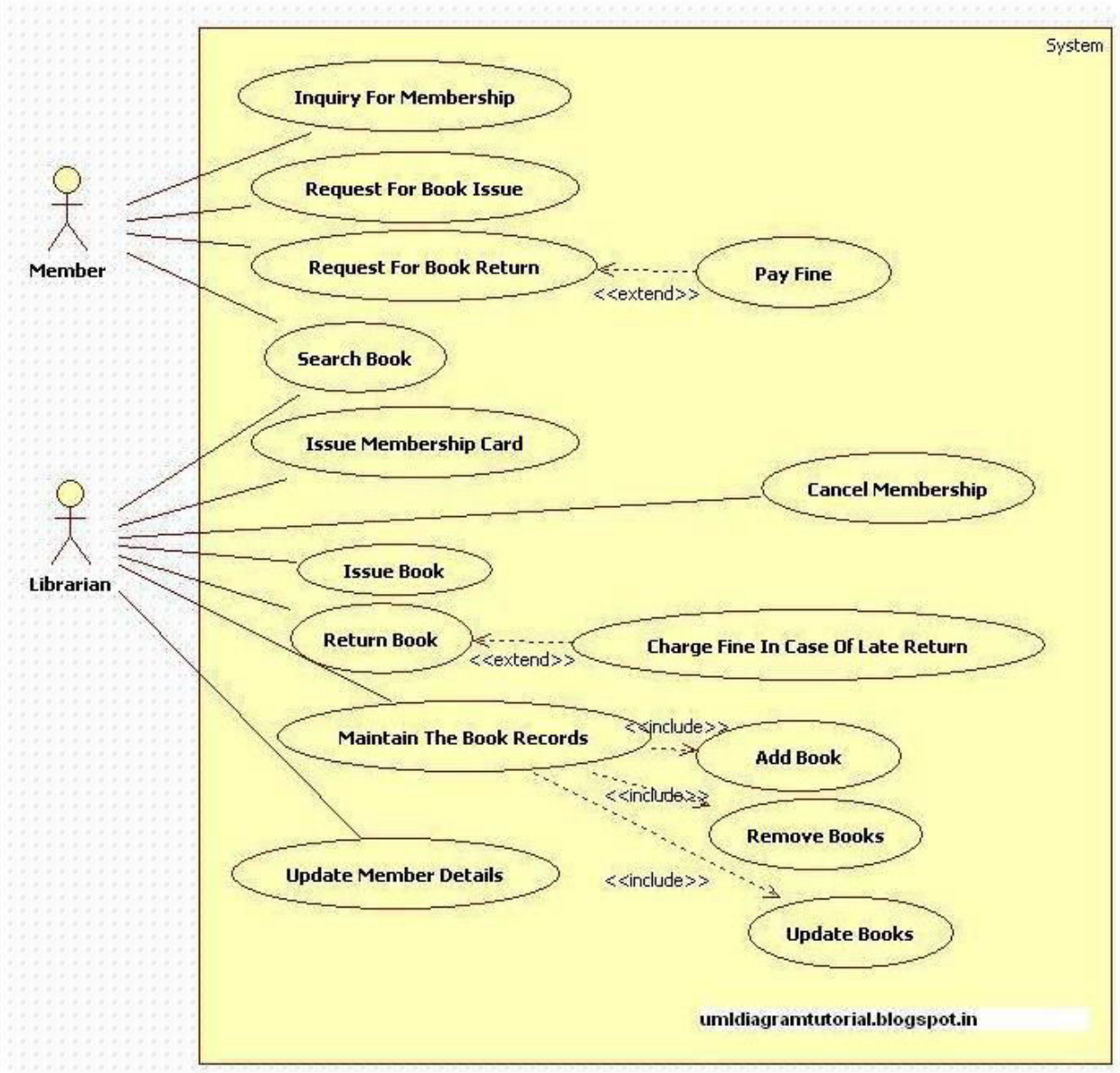
Library Management System – Use Case Diagram

- UML Use Case diagram for Library Management System is shown below. The various participants of the same are detailed below:-
- Actors:- Member, Librarian

The corresponding use cases for these actors are:-

- Member: Inquiry For Membership, Search Book, Book Issue, Book Return, Pay Fine
- Librarian: Search Book, Issue Membership Card, Cancel Membership, Issue Book, Return Book, Charge Fine In Case of Late Return, Maintain the Book Records, Add Books, Remove Books, Add Members, Remove Members, Update Member Details
- Here we have some dependencies also like Request For Book Return <<extends>> Pay Fine. Also Maintain Book Records <<includes>> Add Book, Remove Book and Update Book. Again Return Book <<extends>> Charge Fine in case of late return.

- The Use Case UML diagram for Library Management System is shown here:-



Use case diagram for library management system