SOFTWARE REQUIREMENT & REQUIREMENT ENGINEERING

CHAPTER OUTLINE

After studying this chapter, the reader will be able to understand the

- Software Requirement
- Types of Requirements
- Requirement Engineering Process

SOFTWARE REQUIREMENT

Requirement encompasses a set of tasks which help to specify the impact of the software on the organization, customers need and how users will interact with developed software. Requirement is seen as a high level abstract statement of service that the system should provide. It set out what the system should do and define constraints on its operation and implementation. Requirements are the basis of the system design. It requirement are not correct, the end product will contain errors. Requirement is the condition or capability possessed by the software or system in order to solve a real world problem. The problem may be to automate the system, to correct shortcomings of an existing system, to control device and so on. Requirement describes how a system should act, appear or perform. For this when users request for software, they provide an approximation of what the new system should be capable of doing. Requirement differs from one user to another user and from one business to another.

Software requirements deal with establishing the needs of stakeholders that are to be solved by software. The software requirements are description of features and functionalities of the target system. Requirements convey the expectations of users from the software product. Gathering the correct and complete requirement is one of the most important thing is software development. Incorrect and incomplete requirement are main reason of why project fails.

TYPES OF REQUIREMENTS

User Requirement

User requirements are the statements in a natural, language plus diagram of what services that the system is expected to provide and the constrains under which it must operate. These often referred to as user needs, describe what the user does with the system, such as what activities that users must be able to perform. User requirement comes from the user or other type of stakeholders and express a property of domain or business process that the introduction of new system will bring about. User requirements are generally documented in user requirement document using narrative text. User requirements are generally signed off by the user and used as primary input for creating system requirements.

An important and difficult step of designing a software product is determining what the user actually wants it to do. This is because the user is often not able to communicate the entirety of their needs and wants and the information they provide may also be incomplete, inaccurate and self-conflicting. This is why user requirements and generally considered separately from system requirements.

System Requirement

System requirements set out the systems function; services and operational constraints in detail. System requirements are the building blocks that the developers use to build the system. These are the traditional "Shall" statement that describes what the system shall do.

System requirement document is the structured document setting out the detailed description of the system services written as a contractor between client and contractor.

Readers of user and system Requirement

User Requirement	System Requirement
• Client Manager	• System and user
• System end user	• Client engineer
• Contractor manager	• System Architect
• System Architects	• Software developers.

Functional Requirement

Functional requirements define the basic system behavior. Essentially, they are **what** the system does or must not do, and can be thought of in terms of how the system responds to inputs. Functional requirements usually define if/then behaviors and include calculations, data input, and business processes.

Functional requirements are the statement of the services that the system must provide or descriptions of how some computation are carried out.

These are the statement of services the system should provide, how the system should react to particular inputs, how the system should behave in a particular situation. Functional requirements are features that allow the system to function as it was intended. If the functional requirements are not met, the system will not work. Functional requirements are product features and focus on user requirements.

In some cases, the functional requirement may also explicitly state what the system should not do. The functional requirement is describing the behavior of a system as it relates to the system's functionality.

Functional requirements are the main things that the user expects form the s/w for eg: if the application is banking application, that application should be able to create a new account, update account, delete an account etc.

The functional requirement for the system should be both complete and consistent. Completeness means that all services required by the user should be defined and consistency means that requirement should not have contradictory meaning.

Example of Functional Requirements

- The software automatically validates customers against the Contact Management System
- The Sales system should allow users to record customers sales

- The background color for all windows in the application will be blue and have a hexadecimal RGB color value of 0x0000FF.
- Only Managerial level employees have the right to view revenue data.
- The software system should be integrated with banking API

Advantages of Functional Requirement

- Functional requirements document helps you to check whether the application is providing all the functionalities that were mentioned by user for that application
- A functional requirement document helps you to define the functionality of a system or one of its subsystems.
- Functional requirements along with requirement analysis help identify missing requirements. They help clearly define the expected system service and behavior.
- Errors caught in the Functional requirement gathering stage are the cheapest to fix.
- Support user goals, tasks, or activities for easy project management
- Functional requirement can be expressed in Use Case form or user story as they exhibit externally visible functional behavior.

Non-Functional Requirements

Non-functional requirement are requirements that are the not directly concerned with specific functions delivered by the system but they concerned with emergent system properties, such as reliability, response time, security, safety etc. non-functional requirements specify how the system should do it. Non-functional requirements do not affect the basic functionality of the system (hence the name, non-functional requirements). Even if the non-functional requirements are not met, the system will still perform its basic purpose.

Non-functional requirements place constraints on how the system will do so and elaborates a performance characteristic of the system. Non-functional requirements may also describe aspects of the system that don't relate to its execution, but rather to it's evaluation over time. (eg. maintainability, extensibility, reliability, etc.) For eg. for an banking application, a major non-functional requirement will be availability, the application should be available 24/7 with no down time if possible.

Non functional requirement may constraint the process that should be used to develop the system. These arise through user needs, because of budget constraints, organizational policies, external factors such as safety regulations of privacy legislation. These are often called qualities of system, constraints, quality goals, quality of service require.

Examples of Non-functional requirements

- Users must change the initially assigned login password immediately after the first successful login.

- Employees never allowed updating their salary information. Such attempt should be reported to the security administrator.
- Every unsuccessful attempt by a user to access an item of data shall be recorded on an audit trail.
- A website should be capable enough to handle 20 million users with affecting its performance
- The software should be portable. So moving from one OS to other OS does not create any problem.
- Privacy of information, the export of restricted technologies, intellectual property rights, etc. should be audited.

Advantages of Non-Functional Requirement

- The nonfunctional requirements ensure the software system follow legal and compliance rules.
- They ensure the reliability, availability, and performance of the software system
- They ensure good user experience and ease of operating the software.
- They help in formulating security policy of the software system.

Types of Non-Functional Requirements

1. **Product Requirement**
   The requirements which specify the product behavior are called product requirements. The product's requirements include all functions, features and behaviors that the product must possess, so that it will be efficient, ease to use, safe, low cost, etc. In other words - any function, constraint or other property that is required in order to satisfy user's needs. It includes the performance requirement which specify how fast system must execute and how much memory it requires, its reliability, usability etc.

2. **Organization Requirement**
   The requirements which are derived from politics and procedures in the customer's and developer's organization are called organizational requirement. It includes the process standard used, implementation such as programming language, design methods etc. These requirement specifics the system must be developed according to company standard process.

3. **External Requirement**
   The requirement that are derived from the factors external to the system and its development process. It ensures that whether the system operates within law or not, whether it will be acceptable to its users and general public or not.

Difference between Functional and Non-functional Requirements

Functional requirements	Non functional requirements
A functional requirement defines a system or its component.	A non-functional requirement defines the quality attribute of a software system.
It specifies "What should the software system do?"	It places constraints on "How should the software system fulfill the functional requirements?"
Functional requirement is specified by User.	Non-functional requirement is specified by technical peoples e.g. Architect, Technical leaders and software developers.
It is mandatory.	It is not mandatory.
It is captured in use case.	It is captured as a quality attribute.
Defined at a component level.	Applied to a system as a whole.
Helps you verify the functionality of the software.	Helps you to verify the performance of the software.
Functional Testing like System, Integration, End to End, API testing, etc are done.	Non-Functional Testing like Performance, Stress, Usability, Security testing, etc are done.
Usually easy to define.	Usually more difficult to define.

Domain Requirement

Domain requirements are the requirements which are characteristic of a particular category or domain of projects. These requirements derived from the application domain of the system rather than from the specific needs of the system users. Domain requirement reflects the environment in which the system operates. It includes specialized domain terminology or reference to domain concepts. For instance, in academic software that maintains records of a school or college, the functionality of being able to access the list of faculty and list of students of each grade is a domain requirement. These requirements are therefore identified from that domain model and are not user specific.

Domain requirement are important because they often reflects fundaments of application domain, if these requirements are not satisfied, it may be impossible to make system work satisfactory.

Software Requirement Specification

A software requirements specification (SRS) is a detailed description of a software system to be developed with its functional and non-functional requirements. It is the process of writing the user and system requirements in a requirement document. Software requirements specification establishes the basis for an agreement between customers and contractors or suppliers on how the software product should function. Requirements specifications are a central artifact in most software development processes. They capture the goals of the software to be developed and constitute the connection between the customer/user and the developers. Many call the specifications the determining part for project success or failure. It may include the use cases of

how user is going to interact with software system. The software requirement specification document consistent of all necessary requirements required for project development. To develop the software system we should have clear understanding of Software system. To achieve this we need to continuous communication with customers to gather all requirements.

A good SRS defines the how Software System will interact with all internal modules, hardware, communication with other programs and human user interactions with wide range of real life scenarios. Using the Software requirements specification (SRS) document on QA lead, managers creates test plan. It is very important that testers must be cleared with every detail specified in this document in order to avoid faults in test cases and its expected results.

While writing a SRS following methods can be used

- **Natural Language Sentences:** In this method, requirements are expressed in natural language using numbered sentences where each sentence represents one requirement.
- **Structured Natural Language:** In this method the requirements are expressed in natural language using templates with different fields where each fiend provides the information about an aspect of the requirement.
- **Design Description Language:** This method uses a language like programming language having more abstract features to specify the requirements by defining an operational model of the system.
- **Graphical Notations:** In this method functional requirements are expressed using graphical notations instead of text notations such as: UML use case, sequence diagram etc.
- **Mathematical Specifications:** This method uses the concept of mathematical notations such as sets, finite state machines etc.

Natural Language Specification

In requirement engineering, the difficulties in understanding and eliciting the desired functionalities from the customer and then delivering the correct implementation of a software system lead to delays mistakes, and high costs. Natural language is often used to write system requirements specifications as well as user requirements. However, because system requirements are more detailed than user requirements, natural language specifications can be confusing and hard to understand. Various problems can arise when requirements are written in a natural language sentences in a text document.

- Natural language understanding relies on the specification readers and writers using the same words for the same concept. This leads to misunderstandings because of the ambiguity of natural language.
- A natural language requirements specification is over flexible. You can say the same thing in completely different ways. It is up to the reader to find out when requirements are the same and when they are distinct.

- There is no easy way to modularize natural language requirements. It may be difficult to find all related requirements. To discover the consequence of a change, you may have to look at every requirement rather than at just a group of related requirements.

Problem with Natural Language Specification

Following are the main problems that often arise when requirements are written in natural language sentences:

- It is sometimes difficult to use language in a precise and unambiguous way without making the document wordy and difficult to read.
- Requirements confusion Functional requirements, non-functional requirements, system goals and design information may not be clearly distinguished.
- Requirements amalgamation several different requirements may be expressed together as a single requirement.
- It is sometimes difficult to use language in precise and unambiguous way without making the document wordy and difficult to use.
- Functional requirement, non-functional requirement system goals and design information may not be clearly distinguished.
- Several different requirements may be expressed together as a single requirement.

REQUIREMENT ENGINEERING PROCESS

Requirements engineering is the process of eliciting stakeholder needs and desires and developing them into an agreed-upon set of detailed requirements that can serve as a basis for all subsequent development activities. It is a process that involves all of the activities required to create and maintain a system requirement document. The purpose of requirements engineering is to make the problem that is being stated clear and complete, and to ensure that the solution is correct, reasonable, and effective. As its name suggests the engineering discipline of establishing user requirements and specifying software systems. It is the area of systems engineering that deals with the process of developing and verifying the system requirements. Following good requirements engineering practices helps achieve the primary objective of making sure that the delivered system meets the customer's needs. The requirements engineering process starts with requirements formulation, which is followed by validation, and then by verification. The process used for requirement varies widely depending on the application domain, the people involved and the organization developing the requirement. However, there are number of generic activities, common to all processes:

- Feasibility study
- Requirement elicitation and analysis
- Requirement specification
- Requirement validation

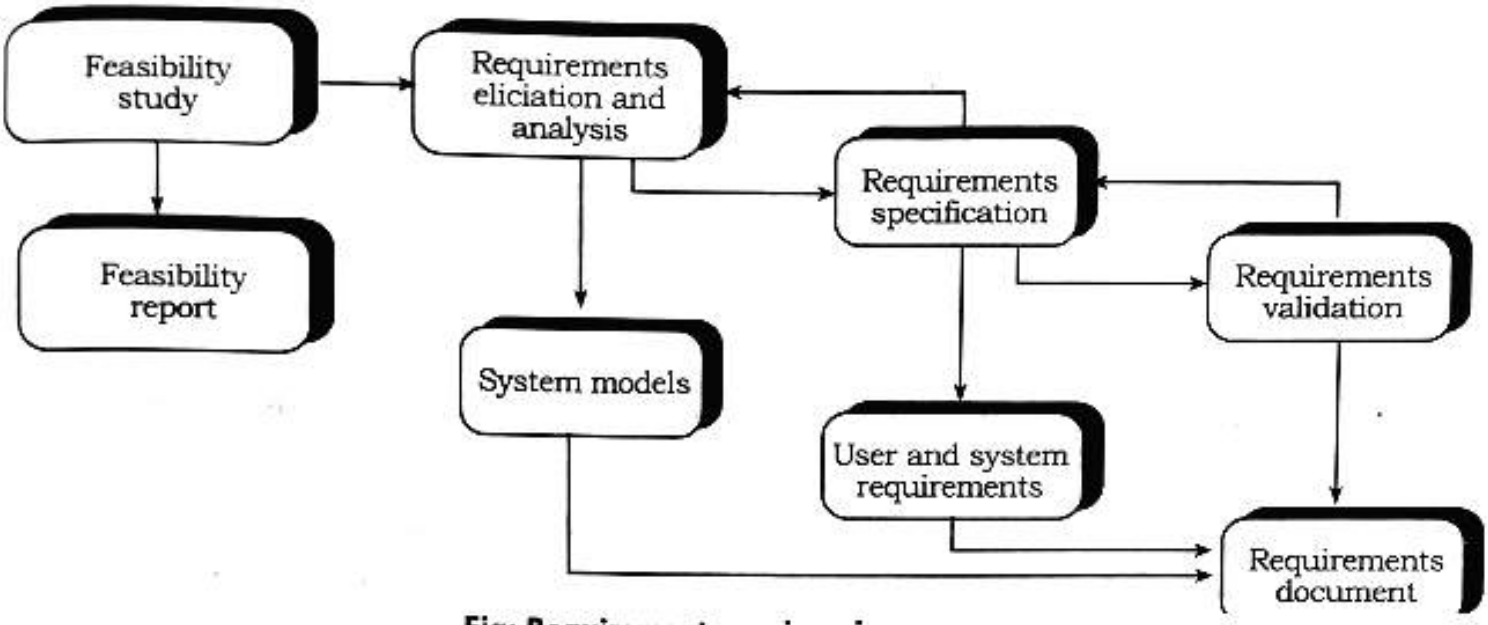

Fig: Requirements engineering process

Feasibility Study

Feasibility study is one of the essential activities that need to be carried out in system engineering process. It is a measure of now beneficial development of system is to an organization. To evaluate feasibility, a feasibility study is performed, which determines whether the solution considered to accomplish the requirements is practical and workable in the software. Information such as resource availability, cost estimation for software development, benefits of the software to the organization after it is developed and cost to be incurred on its maintenance are considered during the feasibility study. The objective of the feasibility study is to establish the reasons for developing the software that is acceptable to users, adaptable to change and conformable to established standards. A feasibility study takes into account various constraints within which system should be implement and operate such as computing equipment, manpower, cost etc.

Importance of Feasibility Study

- Does the system contribute to overall objective of an organization?
- To provide management with enough information to know whether the project can be done, whether the final product will be benefiting its intended users.
- To determine limitation and constraints before starting to develop the system.
- To suggest whether the system can be implemented using current technology within given cost and schedule constraints.
- Provide quality information for decision making.
- Provides documentation of the investigated system.

The result of feasibility study should be a report which recommends whether or not it is worth carrying on with requirement engineering and system development process called feasibility report. There are basically six types of feasibility study

1. **Technical Feasibility**

Technical feasibility assesses the current resources (such as hardware and software) and technology, which are required to accomplish user requirements in the software within the allocated time and budget. For this, the software development team ascertains whether the current resources and technology can be upgraded or added in the software to accomplish specified user requirements. Technical feasibility also performs the following tasks.

- Analyzes the technical skills and capabilities of the software development team members.
- Determines whether the relevant technology is stable and established.
- Ascertains that the technology chosen for software development has a large number of users so that they can be consulted when problems arise or improvements are required.

2. **Economic Feasibility**

Economic feasibility determines whether the required software is capable of generating financial gains for an organization. It involves the cost incurred on the software development team, estimated cost of hardware and software, cost of performing feasibility study, and so on. For this, it is essential to consider expenses made on purchases (such as hardware purchase) and activities required to carry out software development. In addition, it is necessary to consider the benefits that can be achieved by developing the software. Software is said to be economically feasible if it focuses on the issues listed below.

- Cost incurred on software development to produce long-term gains for an organization.
- Cost required conducting full software investigation (such as requirements elicitation and requirements analysis).
- Cost of hardware, software, development team, and training.

3. **Operational Feasibility**

Operational feasibility assesses the extent to which the required software performs a series of steps to solve business problems and user requirements. This feasibility is dependent on human resources (software development team) and involves visualizing whether the software will operate after it is developed and be operative once it is installed. Operational feasibility also performs the following tasks.

- Determines whether the problems anticipated in user requirements are of high priority.
- Determines whether the solution suggested by the software development team is acceptable.

- Analyzes whether users will adapt to new software.
- Determines whether the organization is satisfied by the alternative solutions proposed by the software development team.

Requirement Elicitation and Analysis

This is the second stage of requirement engineering process also called requirement discovery. Elicitation is gathering of all the system requirements from the stakeholders and it encompasses all activities involved in discovering the requirements of a system. The system developers and engineers work in a close relationship with the customers and end-users to determine more about the problem to be solved and to bridge the gap between the stakeholders and the developers. Requirements elicitation is the practice of researching and discovering the requirements of a system from users, customers, and other stakeholders. Elicitation techniques facilitate this process by, Finding out more about the problem to be solved.

- Describing the functionalities of the system and non functional attributes.
- Enhances the performance of the system.
- Overcomes hardware constraints.
- Bridges the gap between the stakeholders and the developers.

It may involves end users, managers, engineers involved in maintenance, domain experts, trade union etc. these are called stakeholders.

Here are the four main activities of requirements elicitation and analysis.

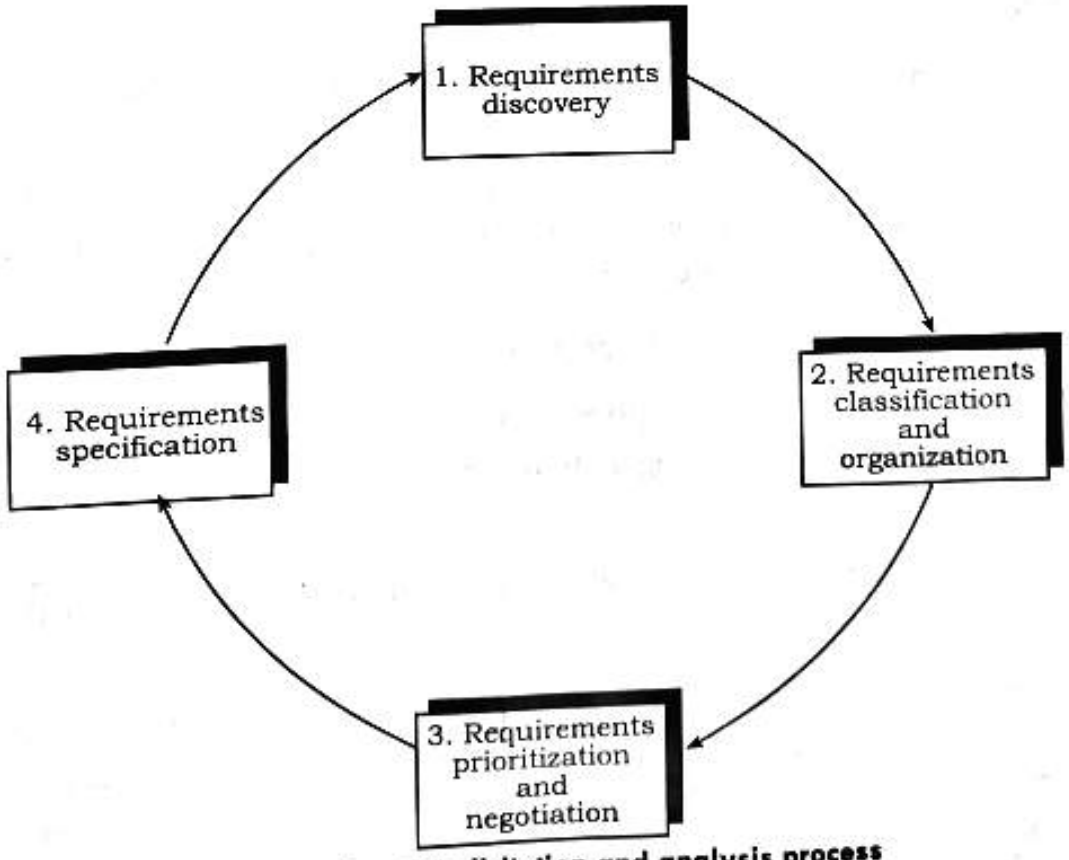

Fig: Requirement elicitation and analysis process

It shows that it's an iterative process with a feedback from each activity to another. The process cycle starts with requirements discovery and ends with the requirements document. The cycle ends when the requirements document is complete.

1. **Requirement Discovery: (Domain Understanding)**

Requirement discovery is the process of interacting with, and gathering the requirements from, the stakeholders about the required system and the existing system (if exist). During this activity domain requirements and documentations are also discovered. To discover the requirements different requirement gathering tools are used such as: interview, questionnaire, scenario, ethnography, use case etc. Discovering the requirements from the stakeholders of the system is a difficult process for following reasons:

- Stakeholders don't know what they really want from system. They might find difficult to articulate requirements.
- Stakeholders express equipment in their own terms with implicit knowledge of their work.
- Different stakeholders may have conflicting requirements.
- The organizational and political factors may influence the system requirements.
- The economic, technical and business environment in which analysis takes place is dynamic.

2. **Requirement Classification and Organization**

It's very important to organize the overall structure of the system. Putting related requirements together, and decomposing the system into sub components of related requirements. Then, we define the relationship between these components. This activity takes the unstructured collection of requirements, groups the related requirements and organizes them into coherent clusters.

3. **Requirement Prioritization and Negotiation**

This activity is concerned with prioritizing requirements and finding and resolving requirements conflicts through negotiations until you reach a situation where some of the stakeholders can compromise.

We shouldn't reach a situation where a stakeholder is not satisfied because his requirement is not taken into consideration.

Prioritizing your requirements will help you later to focus on the essentials and core features of the system, so you can meet the user expectations. It can be achieved by giving every piece of function a priority level. So, functions with higher priorities need higher attention and focus.

4. **Requirement Specification**

In this step, requirements are checked to discover whether they are complete, consistent or not and they are documented and input in to the next round of spiral.

Tools used in Requirement Elicitation

1. **Interviewing**

In Interviews, requirements engineering teams put the questions to the stakeholder about the system that's currently used, and the system to be developed, and hence they can gather the requirements from the answers. Interviews may be of two types:

- Closed interviews: In this interview, a predefined set of questions are asked to the stakeholders and the stakeholder answers these set of questions.
- Open interviews: In this interview, there is no predefined agenda. There is no a pre-defined expected answer; they are more of generic questions. It's used to explore issues that are not clear in a less structured way. The requirement engineer explores a range of issues with system stakeholders and hence develops a better understanding of their needs.

2. **Scenarios**

Scenarios can be particularly useful for adding detail to an outline requirements description. Scenarios are stories which explain how a system might be used. They should include. Scenarios are examples of interaction sessions which describe how a user interacts with a system. Discovering scenarios exposes possible systemu interactions and reveals system facilities which may be required

They are descriptions of example interaction session. During requirement elicitation several forms of scenarios have been developed, each of which provides different levels of details about the system. The scenarios start with an outline the interaction, and during elicitation, details are related to create a complete description of the interrelation:

i. A description of what the system and users expect when scenario starts.

ii. A description of the normal flow of events in the scenario.

iii. A description of what can go wrong and how this is handled.

iv. A description of the system state when the scenario finishes.

Use Cases

Use cases are a scenario based technique for requirements elicitation which were first introduced by Jacobsen in 1993 and used as fundamental feature of UML notation for describing object oriented system models. Use case identifies the type of interaction between the system and its users and the actors involved. In use case, actor in the process is represented as stick figure, each of interaction is represented as a normal ellipse and system boundary is represented by rectangle box.

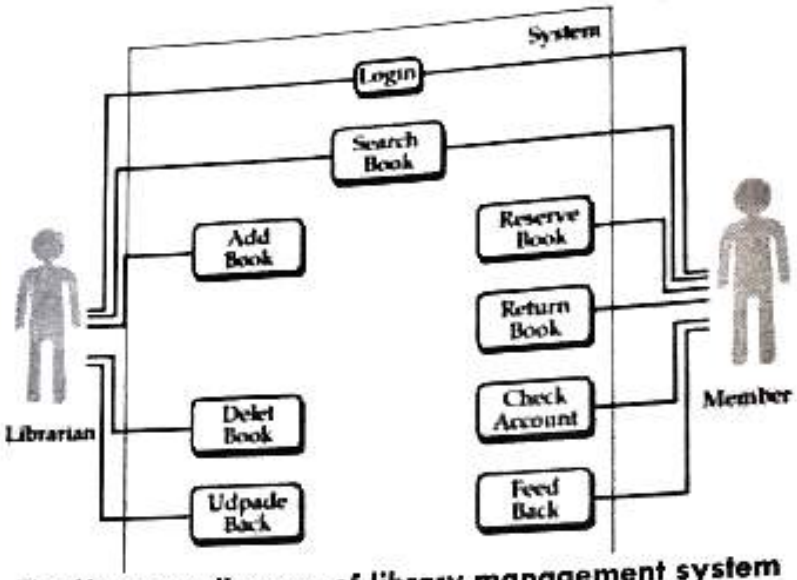

Fig: Use case diagram of library management system

Ethnography

Software systems do not exist in isolation, they are used in a social and organizational context and s/w system requirements may be derived or constrained by that context.

"Ethnography is an observational technique that can be used to understand social and organizational requirements. It is a technique from the social sciences which has proved to be valuable in understanding actual work processes. Actual work processes often differ from formal, prescribed processes. In this, an analyst immerses him or herself in the working environment where the system will be used and observes the day to day work and notes made of the actual tasks in which participants are involved and building up a picture of how work is done. The value of ethnography is that it helps the analysts to discover implicit system requirements that reflect the actual rather than the formal process in which people are involved.

Ethnography is particularly effective at discovering two types of requirements:

- Requirements that are derived from the way in which people actually work rather than the way in which process definitions say that ought to work.
- Requirements that are derived from co-operation and awareness of other people's activities.

Viewpoints

Viewpoint is a technique used in requirement elicitation that recognizes multiple perspectives and provides a framework for discovering conflicts in the requirements proposed by different stakeholders. Viewpoints can be used as a way of classifying stakeholders and other sources of requirements. There are three generic types of viewpoints:

- **Interactor Viewpoints:** It represents people or other systems that interact directly with the system. In the bank ATM system, bank's customer and bank's account database act as this type.

- **Indirect Viewpoints:** It represents stakeholders who do not use the system themselves but who influence the requirement in some way. Example: In ATM system, management of bank and bank security staff falls under this category.
- **Domain viewpoints:** Represents domain characteristics and constraints that influence the system requirements. **Example:** Standard that has been developed for interbank communication

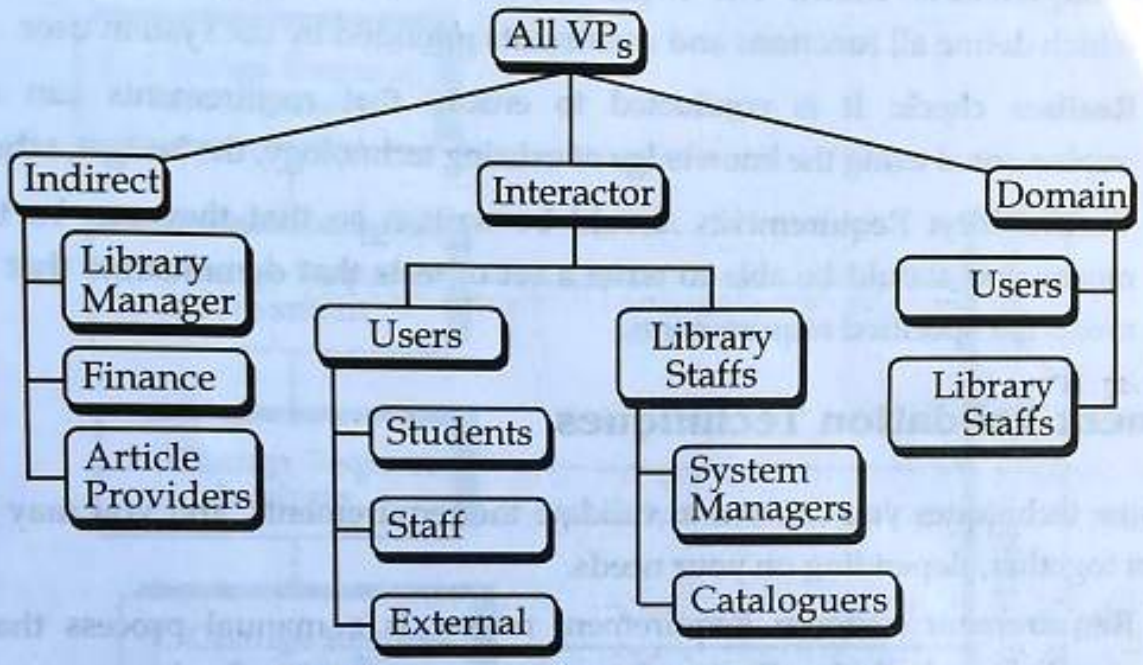

Fig: Library management system viewpoints

Requirement Specification

The management of the organization studies feasibility report and suggests the modifications in the requirement if any. Knowing the constraints on available resources and modified requirements specified by organization, final specification of the system to be developed is drawn up by the system analyst.

Requirement Validation

Requirement validation is the process of checking the requirements for validity, consistency, completeness, realism and verifiability. Requirement validation concerned with showing that the requirement actually define the system which the customer wants. It's a process of ensuring the specified requirements meet the customer needs. It's concerned with finding problems with the requirements. Requirement validation is important because errors in the requirement document can lead to extensive rework cost of design and development when they are discovered during development.

During the requirement validation process different types of check should be carried out on the requirements.

- **Validity check:** The functions proposed by stakeholders should be aligned with what the system needs to perform. You may find later that there are additional or different functions are required instead. A user may think that a system is needed

to perform certain functions, however analysis may identify additional or different function that are required, at that time validity check identify that, the requirement is valid or not.

- **Consistency check:** Requirements in the document shouldn't conflict or different description of the same function.
- **Completeness check:** The requirement document should include requirement which define all functions and constraints intended by the system user.
- **Realism check:** It is conducted to ensure that requirements can actually be implemented using the knowledge of existing technology, the budget, schedule, etc.
- **Verifiability:** Requirements should be written so that they can be tested. This means you should be able to write a set of tests that demonstrate that the system meets the specified requirements.

Requirement Validation Techniques

There are some techniques you can use to validate the requirements, and you may use one or more of them together, depending on your needs.

a. **Requirement reviews:** Requirement review is a manual process that involves people from both the client and contractor organization. In this approach, the SRS is carefully reviewed by a group of people including people from both the contractor organizations and the client side, the reviewer systematically analyses the document to check error and ambiguity.

b. **Prototyping:** In this approach to validation, an executable model of the system is demonstrated to the customer and end users to validate, and ensure if it meets their needs. Prototyping is usually used when the requirements aren't clear. So, we make a quick design of the system to validate the requirements. If it fails, we then refine it, and check again, until it meets the customer needs. This definitely will decrease the cost as a result of having clear, understandable, consistent requirements.

c. **Test case generation:** Requirement should be testable if the tests for the requirement are devised as part of the validation process, this often results requirement problem. If a test is impossible or impossible to design, this usually means that requirement will be difficult to implement and should be re-considered.

Requirement Change Management

Modern software-intensive systems are developed in a world where only constant thing is change. This results in continuously changing requirements. Failure to manage these changing requirements properly may result in failure of system. To manage these requirements, many requirements change management process model has been proposed. Requirement management is the process of managing changes to a system requirement. It is often the case that more than 50%

of a systems requirement will be modified before it is put into service. New requirement emerge and existing change due to errors, increased understanding and change in external circumstances. It is the process of understanding and controlling changes to the system requirement. Change management should be applied to all proposed changes to a system's requirements after the requirement document has been approved. It is essential because you need to decide if the benefits of implementing new requirements are justified by the costs of implementation.

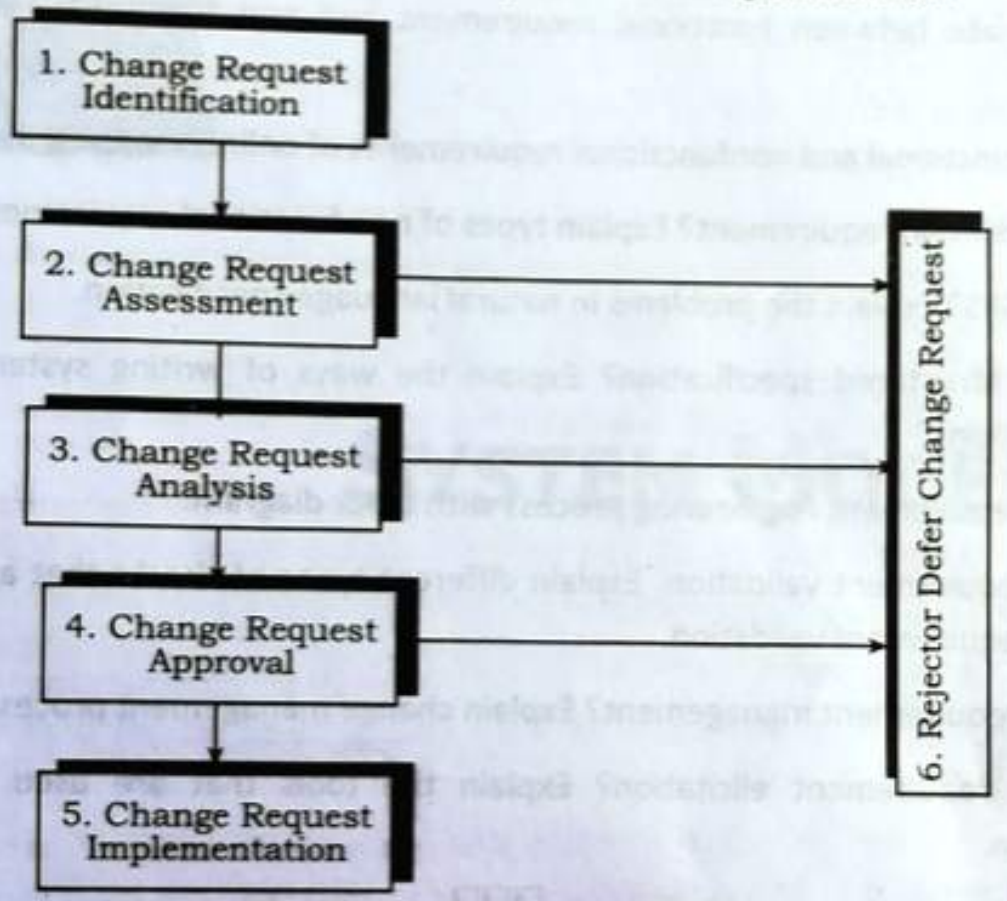

Fig: Change Management Process

There are three principal stages to a change management process:

a. **Problem analysis and change specification:** In this stage the problem or change proposal is analyzed to check that it is valid. This analysis is feedback to the change requestor who may respond with a more specific requirements change proposal. Or decide to withdraw the request.

b. **Change Analysis and Costing:** the effect of the proposed change is assessed using traceability information and general knowledge of the system requirements. The cost of making change is estimated both in terms of modification to the requirements documents and if appropriate to the system design and implementation. Once this analysis is completed, a decision is made whether or not to proceed with the requirements change.

c. **Change Implementation:** In this the requirement document, system design and implementation are modified if necessary. Requirement document should be organized in such a way that we can make change to it without extensive rewriting the document.

EXERCISE

1. What do you mean by software requirement? Explain user requirement and system requirement.
2. Differentiate between functional requirement and non functional requirement with example.
3. List the functional and nonfunctional requirements of online shopping system.
4. What is domain requirement? Explain types of non functional requirement.
5. What is SRS? Explain the problems in natural language specification.
6. What is structured specification? Explain the ways of writing system requirements specification.
7. Explain requirement engineering process with block diagram.
8. Define requirement validation. Explain different types of checks that are conducted as part of requirement validation.
9. What is requirement management? Explain change management process.
10. What is requirement elicitation? Explain the tools that are used in requirement elicitation.

❑❑❑