UNIT I

CHAPTER 3

Requirement Analysis and System Modeling

Syllabus :

Requirements Engineering, Eliciting Requirements, SRS Validation, Components of SRS, Characteristics of SRS.

3.1 Requirements

- A *Requirement* is a condition needed by a user to solve a problem or achieve an objective. For *example*, a requirement for a car could be that the maximum speed to be at least 120 mph.

System requirements fall into two categories

1. Functional 2. Nonfunctional

Table 3.1.1 : Difference between functional and nonfunctional requirements

Sr. No.	Functional Requirements	Nonfunctional Requirements
1.	Describes Product behavior	Describes Product's quality attributes.
2.	Describes what the product should do i.e. the actions/work/services of the software.	Describes capabilities of the product i.e. how the software should behave to meet the user needs.
3.	Describe the services that a system should provide	Describes the design and implementation constraints on the services and the external interface which a product must have.
4.	Describe 'what' the system should do in a particular condition.	Describe 'how' the system should work so as to be user friendly and secured.
5	Example : A bank software has functions – open acc., close acc, withdraw, loan claim etc.	Example : Performance, reliability, portability, security – system must run on windows server 2003, system must be secured against Trojan attacks.
6	The functions are represented using use case diagram & use case specifications.	The performance, usability, reliability, and security quality attributes are documented in narrative descriptions.

Syllabus Topic : Requirements Engineering

3.2 Requirements Engineering

Requirement Engineering is a bridge to design and construction of a quality software product.

- Requirement Engineering establishes an interaction between the developers, customers and users.
- The *input* to requirement engineering is : wishes, problems, unclear requirements etc. and the *output* of requirement engineering is the *Requirement Specification* and *Analysis model* that includes complete coverage of the problem and complete-exact definition of each requirement.
- Requirement Engineering is collecting the information (requirements) about what the system should do (not how to do it).

Requirement engineering includes following tasks so as to improve Software quality :

Task 1 : Inception : Defines the scope and nature of the problem to be solved.

Task 2 : Elicitation : Helps the customer to define what is required.

Task 3 : Elaboration : Customer's basic requirements are refined and modified.

Task 4 : Negotiation : As the customer defines the problems, negotiation occurs by highlighting the priorities, what is essential and what isn't, when it is required and many such.

Task 5 : Specification (SRS): Finally, the proposed requirements are specified. Every project needs requirements specification document because it is the formal agreement between the client/end-users, the business owner/stakeholder and the project manager. It states exactly what should and should not be included in a project and what the end-user expects from the proposed project.

Task 6 : Validation : Ensures that both customer's and engineer's understanding of the problem coincide.

Task 7 : Management : Helps the software engineers to control and track the changes in requirements at any time as the project proceeds.

- The above Requirements engineering tasks are important in software development projects as it influences the development cost, time, effort and quality.
- SRS includes the demands of stakeholders (customers, managers or end users). They specify the desired functions, quality attributes and other properties of the proposed software that is to be built or assembled.
- SRS helps software engineers to better understand the problem they will work to solve. SRS involves analyzing and accurately representing the client's requirements in a manner that can be effectively implemented in a system that will confirm to the client's specifications.
- But, in many situations, enough care is not taken in establishing correct requirements. This causes problems, later, in the development life cycle, and more time and money is spent in fixing these problems. Thus, it is necessary that requirements are established in a systematic way to ensure their accuracy and completeness. Requirement Analysts require both communication and technical skills.

Example : A Library Management System is to be designed so that information on books, CDs, DVDs, Journals, etc. can be stored and retrieved.

Requirements demanded by the client i.e. 'what system should do' are:

1. *Functional requirement*: Searching by Title, Author and ISDN should be possible.
2. *Implementation requirement*: User Interface should be web based i.e. accessible via web browser.

3. *Performance requirement*: At least 20 transactions per second should be possible.
4. *Security requirement*: Users shouldn't have access to personal data of other users.

There are few requirements that are not told by the client but decided by Software Engineer :

1. System structure.
2. Implementation technology or language.
3. Development environment or methodology.

Deliverables (Output) of Requirement Gathering tasks

The requirement gathering task helps in obtaining the following results :

- Identifies Entities.
- Essential Use Cases.
- Possibly workflow diagrams, flow charts.
- ER Model – Data Objects (entities), properties of objects (attributes), operations on objects and relationships between objects.
- We are going to study 'how to generate all these outputs' in up-coming chapters.

3.3 Requirements Engineering Tasks

Requirement Engineering is a process used to discover what the customer needs, analyse those needs, assessing feasibility, negotiating a reasonable solution, specifying the solution clearly, validating the specifications and managing the requirements as the project proceeds. It is accomplished through the execution of 7 distinct tasks mentioned in the Section 3.2 and are detailed in the following sub sections.

3.3.1 Inception

- A new project is started when a new business need is identified or a new service is discovered
- The stakeholders of the system perform feasibility study to decide whether or not the proposed system is useful. The study checks :
  - Will the system add to organisational benefits ?
  - Can the system be engineered using current technology and within budget ?
  - Can the system be integrated with other systems that are used ?
- At project Inception time, software engineers ask some questions for people in the organisation based on information collected :
  - What if the system wasn't implemented ?
  - What are current process problems ?
  - How will the proposed system help ?
  - What will be the integration problems ?
  - Is new technology needed ?
  - What facilities must be supported by the proposed system ?
- The need of these questions is to find :
  - The effectiveness of collaboration between the customer and developer.
  - Basic understanding of the problem.
  - Who will use the solution ?
  - The desired nature of the solution.

Syllabus Topic : Eliciting Requirements

3.3.2 Eliciting Requirements

- Requirements Elicitation is also called as Requirements Discovery. It involves identifying the application domain and the services that the system should provide.
- Problems of requirements gathering (eliciting) :
  - Problems of Understanding :
    - End users many a times don't know what they really want
    - End users express requirements in their own terms
    - Different stakeholders may have different requirements
  - Problems of Scope :
    - The scope of the system is not defined properly as the customers specify unattainable requirements that may confuse, rather than clarify, the overall system's objectives.
    - Organisational and political factors may influence the system requirements.
  - Problems of Volatility :

    The requirements change during the analysis process. New stakeholders may emerge and the business environment changes.

To avoid these problems, requirements engineers must approach the requirement gathering activity in an organized manner.

3.3.3 Elaboration

- Elaboration is about creating an analysis model that defines the informational, functional and behavioral aspects of the problem.
- The main task is to describe the problem in a way that establishes a firm base for designing a model.
- Elaboration focuses on expanding the information (i.e. obtained from inception and elicitation) and then developing a refined technical model of software functions, features and constraints (i.e. restrictions or limitations). This process is composed of various modeling and refinement tasks.
- Different models may be produced during this activity depending on the relationships and collaboration between the various business domain entities.
- From the designed model, it would be easy to judge if the efficiency of workflow of the system is as it has been imagined.

3.3.4 Negotiation

- It happens that different stakeholders (customers, business managers or users) propose conflicting requirements. In such a situation, the requirements engineer must settle these conflicts through a process of negotiation.
- First, the key stakeholders are identified. These are the people who are will be involved in negotiation. Then, the customers, users and other stakeholders are asked to rank the requirements and then discuss about the requirements according to their priority in order to settle out the conflicts between them. Risks associated with each requirement are identified, analyzed and

accordingly modified and alternatives are developed so that each party gets satisfied. So, negotiation works towards a set of requirements that makes all the parties to win.

- Negotiation process involves - Requirements collection (from different stakeholders), Classification, Prioritisation, Conflict resolution, Requirements checking and Finalizing the requirements.
- Finally, rough estimates of development effort are made to calculate the project cost and delivery time.

3.3.5 Software Requirement Specification (SRS)

- Specification is concerned with documenting the requirements and this document is maintained over the life of the project. This is the most fundamental condition for successful implementation of the project.
- Specification is not limited to just text document, it can include graphical notations, mathematical specifications, user scenarios or prototypes.
- Specification includes a set of use cases that describe how the user interacts with the software. Use cases are also known as functional requirements. In addition to use cases, specification also includes nonfunctional requirements such as performance requirements and quality standards. (Example of requirements is given in Section 3.2).
- Specification is the complete description of behavior of the system to be developed. It is the final deliverable of requirements engineering task which serves as an artifact for further software engineering activities.

3.3.6 Requirements Validation

- Validation is concerned with checking if the specification meets the users' needs.
- Software engineer checks the specification to confirm whether he is building the right product ?
- As each element of the analysis model is created, it is validated for consistency, omissions and ambiguity. A review of the analysis model addresses the following questions :
  - Is each requirement consistent ?
  - Are all requirements specified clearly ?
  - Is the requirement really necessary and also is it attainable ?
  - Is each requirement bounded and unambiguous ?
  - Can each requirement be traced with its source ?
  - Is each requirement testable ?
  - Are the requirements very complex and if so is it broken into simple and understandable modules ?
- Have all requirement patterns been properly validated against the customer requirements ?
- Validation mechanism is the formal technical review that includes software engineers and different stakeholders who examine the specification to ensure that all the requirements have been stated clearly, errors are detected and removed, looks if there is any missing information, any inconsistencies, conflicting requirements or unrealistic requirements.
- Validation is concerned with checks for comprehensibility, validity, consistency, completeness, realism, verifiability, necessity, traceability and adaptability.

Thus, the requirement validation checklist is as follows :

- **Comprehensibility :** Are requirements stated clearly and properly understood by all stakeholders ?

- *Correct* - A set of requirements is correct if and only if every requirement stated therein represents something required of the system to be built." Davis (1993)
- *Unambiguous* - The reader of a requirement statement should be able to draw only one interpretation of it.
- **Incorrect Example** – "The system should not accept password longer than 8 characters"

  The above stated requirement can have multiple interpretation as given below :
  - The system should not allow user to enter more than 8 characters in the password field
  - The system should truncate the entered data to 8 characters
  - The system should display an error message if user enters more than 8 characters in the password field
- **Correct Example** – "The system should not accept passwords longer than 8 characters in the password field. If the user enters more than 8 characters while entering the password, an error message should prompt the user to correct the same."
- *Complete* - No requirements should be missing. A complete requirement leaves no room for guessing. Include all significant requirements, whether related to functionality, performance, design constraints, attributes, or external interfaces
- *Consistent* - A consistent requirement does not conflict with other requirements in the requirement specification.

  Incorrect Example :

  REQ1 – "The birth date should be entered in mm/dd/yyyy format"

  REQ2 - "The birth date should be entered in dd/mm/yyyy format"

  Correct Example

  REQ1 – "For the US users the birth date should be entered in mm/dd/yyyy format"

  REQ2 – "For the French users the birth date should be entered in mm/dd/yyyy format"
- *Ranked for Importance* – Requirements should be achievable. But sometimes, few requirements are not attainable.

3.3.7 Requirements Management

- Requirements management is a mechanism to control and track the changing requirements at any time as the project proceeds.
- The requirements change or say, new requirements emerge during the process and this may be because;
  - The business needs change and a better understanding of the system is developed.
  - The priority of requirements from different viewpoints changes during the development process.
- Each requirement is assigned a unique identifier i.e. useful in in developing the *traceability tables*. Each requirement traces to one or more aspects of the system.. This is shown in the Table 3.3.1;

Table 3.3.1 : A Traceability table

Requirements	Specific Aspects of the System					
	A1	A2	A3	.	.	Al
R1	✔		✔			
R2			✔	✔		✔
.		✔				
.	✔				✔	
Rn		✔	✔			

- Few traceability tables are as below :
  - **Features traceability table :** Shows how requirements relate to important product features.
  - **Source traceability table :** Identifies the requirements and the source (stakeholders) who proposed these requirements.
  - **Dependency traceability table :** Indicates how requirements are related or depended on one another.
  - **Subsystem traceability table :** Categorizes requirements by the subsystems that they govern.
  - **Interface traceability table :** Shows how requirements relate to both internal and external system interfaces.
- Traceability is useful in case of large and complex system to determine the connections between the requirements so as to understand how a change in one requirement affects the other requirement and how it will affect the different aspects of the system to be built.

3.4 Requirement Elicitation

- It is about collecting the requirements, needs and constraints.
- This task collects the information about :
  - Domain - problems and laws
  - Existing standards and systems
  - Existing specifications
- The Q &A session discussed in Section 3.3.1 (or during inception) does not lead to a very detail elicitation of requirements and so the following approaches are used for successfully eliciting the requirements.

3.4.1 Collaborative Requirements Gathering

- The Q & A session during inception only establishes the scope of the problem and overall perception of a solution. Out of these initial meetings, the stakeholders write a one or two page "product request". A meeting place, time, and date are selected and a facilitator is chosen. The software team and the other stakeholders are invited to attend this meeting. The product request is

distributed to all attendees before the meeting date. Before coming to the meeting, each attendee is asked to make a list of objects that are part of environment that surrounds the system, that are to be produced by the system and objects that are used by the system to perform its functions. Also, he is asked to list the services or functions that act on these objects. Finally, list of constraints (like business rules, cost and size) and performance criteria (i.e. speed, accuracy) are also developed.

- Then comes the actual meeting (collaborative requirement gathering) date. In this gathering, a team of stakeholders and developers work together to identify the problem, propose elements of solution, negotiate different approaches and specify a preliminary set of solution requirements.
- A collaborative requirement gathering is about :
  - Conducting meetings which are attended by both software engineers and customers.
  - Rules for preparation and participation are established.
  - An agenda is suggested to cover all important points and encourage the free flow of ideas.
  - A 'facilitator' (may be a customer or developer) controls the meeting.
  - A 'definition mechanism'(can be worksheets, flip charts, wall stickers or an electronic bulletin board, chat room or virtual forum) is used.
- The goal of such gathering is :
  - To identify the problem
  - Propose elements of the solution
  - Negotiate different approaches Specify a preliminary set of solution requirements.

Example : Consider the "Safe Home" project.

- The list of *objects* involved in this project are control panel, smoke detectors, window and door sensors, motion detectors, an alarm, an event (sensor has been activated), a display, a PC, telephone numbers, a telephone call and so on.
- The list of *services* might include configuring the system, setting the alarm, monitoring the sensors, dialing the phone, programming the control panel and reading the display. All these services act on objects.
- The list of *constraints* might be as - the system must recognize when sensors are not operating, must be user-friendly, must interface directly to a standard phone line.
- The *performance criteria* might be as - a sensor event should be recognized within a second, an event priority scheme should be implemented.

As the *gathering begins*, first topic of discussion is the need and justification for the new product - everyone should agree that the product is justified. Once the agreement has been established, each participant presents his list for discussion. The lists can be pinned to the walls of the room and they can also be posted on an electronic bulletin board or in a chat room environment. Each listed entry should be capable of being manipulated separately (deleted, added or modified). At this stage, critique and debate are strictly prohibited.

Then, a combined list is created by the group so as to eliminate the redundant entries and adds any new ideas that come up during the discussion. Next, the facilitator coordinates the discussion to shorten, lengthen or reword the combined list so as to properly reflect the system to be developed. Now, the team is divided into smaller sub teams; each works to develop mini specifications for one or more entries of the list. Each mini specification is the elaboration of the word included in the list. For example, mini specification of the control panel is - It is a wall mounted unit and is 9*5 inches in size, has wireless connectivity to sensors and a PC, user interaction occurs through a keyboard having 12 keys, a 2*2 inch CD display provides the user feedback, provides interactive prompts, echo and similar functions.

After the mini-specs are completed, each participant makes a list of validation criteria for the system. All these lists are combined and a consensus list of validation criteria is created. Finally, one or more participants are assigned the task of writing a complete draft specification using all inputs from the meeting.

3.4.2 Quality Function Deployment

- It would be very helpful to have a technique that can assist in accurate requirement gathering. Quality Function Deployment (QFD) is a technique that enables a software team to specify clearly the customer's wants and needs, and then evaluate each proposed object or service capability in terms of its impact on meeting those needs.
- QFD is a technique that translates customer requirements into technical requirements for software. QFD focuses on customer needs throughout the software engineering process.

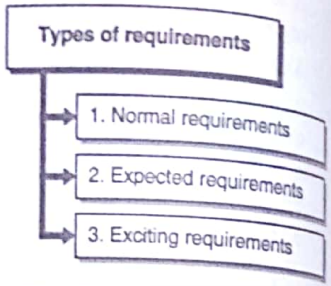


Fig. C3.1 : Types of requirements

QFD identifies 3 types of requirements :

➜ **1. Normal requirements :**

It reflects objectives and goals stated for a system during meetings with the customer. If these requirements are present, the customer is satisfied.

Example : Graphical displays, specific system functions and defined levels of performance.

➜ **2. Expected requirements :**

These are implicit to the system and may be so fundamental that the customer does not explicitly state them. Absence of such requirements causes customer dissatisfaction.

Example : Ease of human-machine interaction, overall operational correctness and reliability and ease of software installation.

➜ **3. Exciting requirements :**

These reflect features that go beyond the customer's expectations and prove to be very satisfying when present.

Example : Customer may request a Word processing software with standard features but he is pleased when he sees that the product delivered to him has number of page layout capabilities along with basic features.

- In meetings with the customer, function deployment determines the value of each function that is required for the system. Information deployment identifies data objects and events that the system must consume and produce. Finally, task deployment examines the behaviour of the system. Then value analysis is conducted to determine the relative priority of requirements determined during each of the three deployments.

QFD uses customer interviews and observation, surveys, and examination of historical data as raw data for the requirements gathering activity. These data are then translated into a table of requirements called the customer voice table. A variety of diagrams, matrices and evaluation methods are then used to extract expected requirements and to attempt to derive exciting requirements.

Benefits of QFD

- Improves user involvement.

- Improves management support and involvement.
- Shortens the development lifecycle.
- Improves project development.
- Supports team involvement .
- Structures communication processes.
- Provides a preventive tool for improving quality.
- Avoids loss of information.

3.4.3 User Scenarios

- Scenarios are descriptions of how a system is used in practice. Scenarios are real-life examples of how a system can be used.
- Scenarios are particularly useful for adding detail to an outline requirements description.
- The scenario starts with an outline of the interaction, and during elicitation, details are added to create a complete description of that interaction. In general, scenario includes :
  - Description of System state and user's expectations at the beginning of the scenario.
  - The normal flow of events.
  - What can go wrong and how this can be handled.
  - Information about other concurrent activities.
  - Description of System state on completion of the scenario.
- Collection of different user scenarios forms a use case.
- Each scenario is described from the point-of-view of an *actor* - a person or a device that interacts with the software directly or indirectly.
- Each scenario answers the following questions :
  - Who are the primary and secondary actors ?
  - What are the actor's goals ?
  - What preconditions should exist before the user story begins ?
  - What are the main functions performed by the actor ?
  - What exceptions might be considered ?
  - What variations (changes) in the actor's interactions are possible ?
  - What information does the actor desire from the system ?
  - Will the actor have to inform the system about the changes in the external environment ?
  - Does the actor wish to be informed about unexpected changes ?

Example : Safe Home Project

In this example, we consider homeowner as the primary actor; and system administrator and sensors are the secondary actors.

Homeowner interacts with the safe home security function using the control panel.

The basic use-case template for the safe home security system activation is as follows :

Use-case :	Initiate Security Monitoring system
Primary actor :	Homeowner
Goal in context :	To set the security system monitoring sensors when the homeowner leaves the house.

Preconditions :	System's password needs to be set and recognize various sensors.
Trigger :	Homeowner decodes to turn on the alarm functions.
Scenario :	Homeowner observes control panel. Homeowner enters password. Homeowner selects 'stay' or 'away'. Homeowner presses the panic button in an emergency. Homeowner activates the security system.
Exceptions :	Control panel is not ready if any of the sensors are open. Password is incorrect. Stay is selected and perimeter sensors are activated. Away is selected and all sensors are activated.
Interface :	control panel.
Secondary actors :	Support technician, sensors.
Queries / Issues:	Can we activate the system without the use of a password ? Is there any way to deactivate the system before it actually activates?
Interface :	control panel.
Secondary actors :	Support technician, sensors.

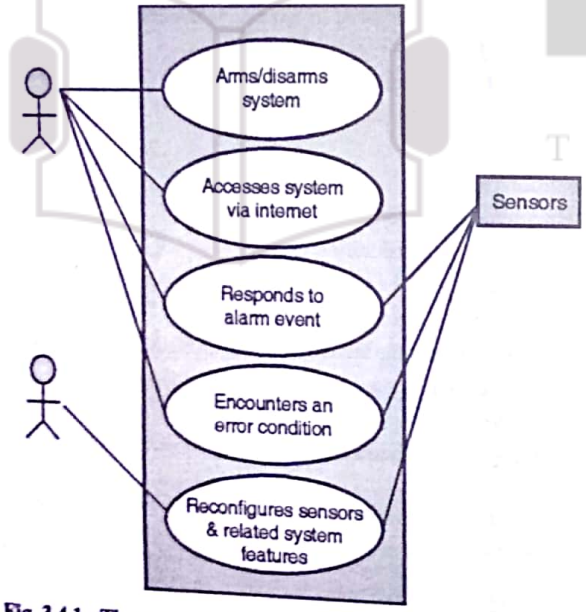


Fig. 3.4.1 : The use-case diagram for safe-home security function

3.4.4 Elicitation Work Products

This is developed as a result of requirements elicitation and it varies depending on the size of the system to be built. It describes :

- The need and feasibility of the system.
- The scope of the system.
- The list of customers, users and other stakeholders who participated in requirements elicitation.
- The technical environment i.e. required.
- List of requirements (that are organized according to their function) and the domain constraints those apply to each.
- A set of user scenarios that provide information about the usage of the system under different operating conditions.
- Any prototypes developed to better define the requirements.

Each of these work products is reviewed by all the people who have participated in requirements elicitation.

3.5 Software Requirement Specification

- Specification is concerned with documenting the requirements and this document is maintained over the life of the project. This is the most fundamental condition for successful implementation of the project.
- Specification is not limited to just text document, it can include graphical notations, mathematical specifications, user scenarios or prototypes.
- Specification includes a set of use cases that describe how the user interacts with the software. Use cases are also known as functional requirements. In addition to use cases, specification also includes nonfunctional requirements such as performance requirements and quality standards. (Example of requirements is given in Section 3.2).
- Specification is the complete description of the behavior of the system to be developed. It is the final work product produced by the requirements engineer which serves as the foundation for subsequent software engineering activities.

☞ **The basic issues that the SRS shall address are the following :**

(i) **Functionality :** behaviour or services provided by the software

(ii) **External interfaces :** external interactions of the software such as with end users, system's hardware or external software

(iii) **Performance :** the throughput, the availability, the response time, the recovery time of the software.

(iv) **Quality attributes :** portability, correctness, maintainability, security, etc.

(v) **Constraints :** implementation language, policies for database integrity, resource limits, operating environments etc.

Example : The ATM system is dependent on the network in the village areas. Due to flood and consequent network jams there will be a delay in the transactions

3.5.1 Benefits (Need) of SRS

- **SRS forms the basis for agreement between customers and development organization on what the software product is expected to do.** Complete descriptions of the functions that are

expected from the software are specified in the SRS. This will help the end users to verify whether the software meets the specified needs or not and if not, then how the software must be manipulated so as to meet their requirements.

- **SRS reduces the development effort.** The work of preparing SRS forces various stakeholders various concerned groups in the customer's organization to think thoroughly of all the requirements before the project design begins, thus reducing the efforts needed for redesigning recoding, and retesting. Careful inspection of the requirements specified in SRS can reveal the outs, misinterpretations and inconsistencies early in the development cycle; hence it becomes easy to correct these problems.
- **SRS forms a base for cost estimation and project scheduling.** As complete description of the product to be developed is specified in the SRS, it helps in estimating the project costs and can be used to obtain approval from the customer for the decided price estimates.
- **SRS provides a baseline for verification and validation.** Software development teams develop their verification and validation plans or say, test plans much more effectively from a well prepared SRS document. As SRS provides a baseline against which requirement confirmation can be measured.
- **SRS facilitates transfer of new software to new environments.** SRS makes it easier to transfer the software product to new clients or new hardware machines. Therefore, developers feel easier to transfer the software to new clients and customers feel easier to transfer the software on other systems of their organization.
- **SRS serves as a basis for software improvement.** SRS describes the product features and not the process of project development; therefore, it serves as a basis for enhancements by allowing the developers to do any later modification if required on the finished product. But then, SRS needs to be altered in that case i.e. SRS forms a base for continuous product evaluation.

Syllabus Topic : Characteristics of SRS

3.5.2 Characteristics of SRS

Great SRS leads to Great Product: We have to keep in mind that the goal is to create great product and great software. A great product can be created only from a great specification. Systems and software these days are so complex that to get on with the design before knowing what you are going to build is foolish and risky.

A great SRS has following quality factors :

- **Correct :** A set of requirements is correct if and only if every requirement stated therein represents something required of the system to be built." Davis (1993)
- **Unambiguous :** The reader of a requirement statement should be able to draw only one interpretation of it.
- **Incorrect Example** – "The system should not accept password longer than 8 characters"

  The above stated requirement can have multiple interpretation as given below :
- The system should not allow user to enter more than 8 characters in the password field
- The system should truncate the entered data to 8 characters
- The system should display an error message if user enters more than 8 characters in the password field

Correct Example – "The system should not accept passwords longer than 8 characters in the password field. If the user enters more than 8 characters while entering the password, an error message should prompt the user to correct the same."

Complete - No requirements should be missing. A complete requirement leaves no room for guessing. Include all significant requirements, whether related to functionality, performance, design constraints, attributes, or external interfaces

Consistent - A consistent requirement does not conflict with other requirements in the requirement specification.

Incorrect Example :

REQ1 – "The birth date should be entered in mm/dd/yyyy format"

REQ2 - "The birth date should be entered in dd/mm/yyyy format"

Correct Example

REQ1 – "For the US users the birth date should be entered in mm/dd/yyyy format"

REQ2 – "For the French users the birth date should be entered in mm/dd/yyyy format"

- *Ranked for Importance* – Requirements should be achievable. But sometimes, few requirements are not attainable.
- *Verifiable* - Don't put in requirements like – "It should provide the user a fast response" or "The system should never crash". Instead, provide a quantitative requirement like: "Every key stroke should provide a user response within 100 milliseconds."
- *Traceable* - Each requirement should be expressed only once and should not overlap with another requirement. Every requirement has a unique identification number so that requirements can be easily traced through out the specification.
- **Adaptability :** Can the requirement be changed without a large impact on other requirements ?

Syllabus Topic : Components of SRS

3.5.3 Components of SRS

Following is the IEEE standards SRS format which depicts the components of a SRS document.

1. INTRODUCTION
   - 1.1 PRODUCT OVERVIEW
     (Product description).
   - 1.2 PURPOSE
     (Product description)
   - 1.3 SCOPE
     Write the benefits, objectives, and goals.
   - 1.4 AUDIENCE
     (lists the users of SRS such as developers, project managers, marketing staff, users, testers, and documentation writers.).
   - 1.5 DEFINITIONS, ACRONYMS AND ABBREVIATIONS
     (defines all the terms necessary to properly interpret the SRS).

1.6 CONVENTIONS

(describes the standard conventions followed while writing this SRS such as fonts or special significant highlights) .

1.7 REFERENCES

(lists the reference documents or web page address used as reference)

2. OVERALL DESCRIPTION

2.1 PRODUCT PERSPECTIVE

(lists the reference documents or web page address used as reference)

2.2 PRODUCT FUNCTIONALITY

(designs such as DFDs that describe major functions of the system).

2.3 USER CHARACTERISTICS

Identify various users who will be using this product. Differentiate them based on frequency of use, subset of product functions used, technical expertise, security or privilege levels, educational level, or experience.

2.4 OPERATING ENVIRONMENT

(Designs that describe that shows the major components of the overall system, subsystem interconnections, and external interface.).

2.5 DESIGN AND IMPLEMENTATION CONSTRAINTS

Describe the issues of the developers such as - hardware limitations, interfaces to other applications, specific technologies, tools, and databases to be used, language requirements, communications protocols, security considerations, design conventions.

2.6 USER DOCUMENTATION

List out the user manuals, on-line help, and tutorials that will be delivered along with the software.

2.7 ASSUMPTIONS AND DEPENDENCIES

List any assumed factors that could affect the requirements stated in the SRS. These include third-party components or issues around the development or operating environment. The project could be affected if these assumptions are incorrect, are not shared, or change.

Also list out the dependencies such as software components that you plan to reuse from another project.

3. SPECIFIC REQUIREMENTS

3.1 EXTERNAL INTERFACE REQUIREMENTS

3.1.1 USER INTERFACES

List out the different screens that will be available to the user.

3.1.2 HARDWARE INTERFACES

List out any special libraries that are used to communicate with your software mention them.

3.1.3 SOFTWARE INTERFACES

(Describes databases, operating systems and tools, services and the data that will be shared across software components)

3.1.4 COMMUNICATIONS REQUIREMENTS

(Describes the communication related requirements such as e-mail, web browser, network server communications protocols, electronic forms and the communication standards such as FTP or HTTP.)

3.2 FUNCTIONAL REQUIREMENTS

(describes the functional requirements in detail with proper explanations regarding each and every function.)

3.3 BEHAVIOUR REQUIREMENTS

3.3.1 USE CASE VIEW

Draw a use case diagram that will illustrate the entire system and all possible actors.

4. OTHER NON FUNCTIONAL REQUIREMENTS

4.1 RELIABILITY

4.2 AVAILABILITY

4.3 SAFETY and SECURITY

(describes the functional requirements in detail with proper explanations regarding each and every function.)

4.4 MAINTAINABILITY

4.5 PORTABILITY

4.6 PERFORMANCE

Provide at least 5 different performance requirement such as -"Any transaction will not take more than 10 seconds".

5. SUPPORTING INFORMATION

This section is Optional. Define any other information/requirements not stated elsewhere in the SRS. This might include internationalization requirements, legal requirements, reuse objectives for the project, and so on.

3.6 Identifying the Stakeholders

- Stakeholder is the personnel who benefits in a direct or indirect way from the system that is being developed.
- Different stakeholders included in the Requirement Engineering/Analysis process are:

1. Client /Customer

- He is the contract partner who orders the software.
- Decides on budget and system functionalities

2. Users

Uses the system

3. Advocate

- Speaks two languages – one of the customer and one of the engineers.
- Should be technically expert and domain expert.

4. **Project Manager**
   - Is the contract partner.
   - Controls the budget.
5. **Requirement Engineer/ Programmer/ Software engineer**

Has knowledge of software engineering and is actually responsible for software development.

☞ **Other stakeholders will include**

- Maintenance operators : anyone who operates the system
- Financial, Safety and Other regulators : organizations which regulate the aspects of the system
- Those organizations who integrate horizontally with the organization for whom the developer is designing the system.

Example : ATM Stakeholders

- **Bank customers :** receive services from the system.
- **Representatives of other banks :** have reciprocal agreements that allow each other's ATMs to be used.
- **Bank managers :** obtain management information from the system.
- **Counter staff :** are involved in the day-to-day running of the system.
- **Database administrators :** are responsible for integrating the system with the bank's customer database.
- **Security managers :** ensure that the system will not pose a security hazard.
- **Marketing department :** uses the system as a means of marketing the bank.
- **Hardware and software maintenance engineers :** are responsible for maintaining and upgrading the hardware and software.

3.7 Issues of Requirement Gathering

Requirement Gathering for the proposed new software needs to identify :

- The technical staff working with customers (stakeholders) so as to find out about the application domain.
- The services that the system should provide.
- The system's operational constraints i.e. how the system is to be used on a day-to-day basis.
- But this all is very difficult to find out as number of problems are faced while requirement gathering.

☞ **Problems of requirements gathering**

There are also, possible problems caused by engineers and developers during requirement analysis :

☞ **Problems of Understanding**

- Stakeholders don't know what they really require. They do not have a clear idea of their requirements.
- Stakeholders express requirements in their own terms. Thus, there is a conflict of opinions between the customer and the developer about the system to be built.

- Different stakeholders may have different requirements.

- Nearly, 40% to 60% of software failures and defects are the result of poor software management and requirements definition. This means, about half of the problems encountered could have been avoided by making it clear, from the very beginning, what the customer expected from the respective project. This means, the programming was fine and the developers did their job well - only they did a different job from what they were supposed to.

☞ **Problems of Scope**

- The scope of the system is not defined properly as the customers specify unnecessary technical detail that may confuse, rather than clarify, the overall system's objectives.
- Organisational and political factors may influence the system requirements.

☞ **Problems of Volatility**

- The requirements change during the analysis process. New stakeholders may emerge and the business environment changes.
- Users or Customers do not be committed to a set of written requirements. Users insist on new requirements in later SDLC phases even after the cost and schedule has been fixed.

- This leads to the situation where user requirements keep changing even when system or product development has been started.

☞ **Problems of Engineer and Developers**

- Technical person and end users may have different understanding of the proposed project and therefore, they may wrongly believe they are in perfect agreement until the finished product is delivered.
- Engineers and developers may try to fit the requirements in an existing system, rather than developing a system i.e. specific to the needs of the client.
- System Analysis may be often carried out by engineers who don't have personal skills and the domain knowledge of the proposed project.

To avoid these problems, requirements engineers must approach the requirement gathering activity in an organized manner. Solution to avoid these issues is to use techniques like prototyping, Unified Modeling Language (UML), use cases, and Agile software development where customer is an active participant from the beginning to throughout the software development cycle.

3.8 Techniques of Requirement Gathering

There are few well-recognised techniques for gathering the information needed in a SRS.

3.8.1 Questionnaires

Different types of questions are asked to different stakeholders at the time of *questionnaire*. Some of these are discussed below :

- Questions asked by the requirement engineer enables the software team to better understand the problem and allows the customer to put his views about the solution (s/w) :
  - How would you characterize "good" output that would be generated by the solution ?
  - What business operations you do ?
  - What steps do you follow to do it i.e. how do you do it ?

- What problems will the solution deal with ?
- In which business environment, will the solution be used ?
- Will special performance issues or constraints affect the way the solution is approached ?

Example : Consider the "Safe Home" project.

- The list of *objects* described for safe home might include the control panel, smoke detectors, window and door sensors, motion detectors, an alarm, an event (sensor has been activated), a display, a PC, telephone numbers, a telephone call and so on.
- The list of *services* might include configuring the system, setting the alarm, monitoring the sensors, dialing the phone, programming the control panel and reading the display. All these services act on objects.
- The list of *constraints* might be as – the system must recognize when sensors are not operating, must be user-friendly, must interface directly to a standard phone line.
- The *performance criteria* might be as - a sensor event should be recognized within a second, an event priority scheme should be implemented.
- Another set of questions focus on the effectiveness of communication activity. This set of questions is also called as "*meta questions*":
  - Are you the right person to answer these questions? Are your answers 'official'?
  - Am I asking you the relevant questions to the problem that you have ?
  - Am I asking too many questions ?
  - Can anyone else provide additional information ?
  - Should I ask you anything else ?

The need of these questions is to '*break the ice*' and initiate the communication between the stakeholders and the requirement engineer that is essential for successful elicitation.

3.8.2 Reviews

Reviewing the existing documents and learning about the existing systems that match the proposed system is necessary to get extra detailed information. This will help in understanding the proposed project better.

Reviews can be obtained by following task:

- Analysts should dialog with users of existing systems that are similar to proposed system.
- Analysts must read documentation on existing system
- Review the reports, forms, procedures, descriptions of other companies.
- Review the Industry journals and magazines reporting "best practices"
- Analysts should validate the discovered information with system end-users.

3.8.3 Interviews

- Interviewing the stakeholders involved in the project is a very effective way of gathering the requirements but it is important to interview the right people and also make sure that the interview questions stay focused on o the project relevancy.
- Two kinds of Interviews can be conducted to investigate the requirements - Structured and Unstructured.

1. Unstructured Interview

It is of a question and answer format which is suitable only to obtain general information about the system. This format allows the users to share their needs and ideas.

Advantages	Drawbacks
– Interviewer can gather more and more information as the user is free to answer in his own words. – Interviewer comes to known about new problems and needs during the interview. – Interviewer may come to know about the importance of few areas that were otherwise over looked by them.	– Time consuming as the user just goes on putting his unclear ideas and needs. – Extra information is gathered even which is not needed. – Therefore, analysis of requirements becomes lengthy.

Example : *Unstructured Interview* conducted by an Analyst to a Librarian

Analyst : Hi, I have come to know regarding the functioning of your library.

Librarian : Hello, please come in. I was waiting for you. Will you elaborate about your work?

Librarian : A big problem is managing the identity cards of members. There are so many numbers of cards and many times the cards get misplaced or lost. Then we have to issue a duplicate card instead of it. But it is difficult to find out whether the member is lying or is he genuine.

Analyst : Ok. Then according to you what may be an ideal solution to this?

Librarian : There should be no use of cards at all and all the information should be put into computer which will ease the issue and check how many books are already with a particular member.

Analyst : How often the new members are registered to the library membership?

Librarian : Very often. About 50 to 100 new members are registered in a month. But since two months we have stopped adding new membership because it is already very difficult to manage the existing 500 members. But if the computerized system replaces our manual work then we'll again start registering the new members. From this system, management hopes to earn huge profit.

Analyst : Can you explain me how?

Librarian : See, we get about 50-100 membership requests every month but we are not able to manage it right now. When the new system is built, we will re-open the membership and there is a membership fees to be paid. Management is planning to increase the fee from 400 to 500 for half yearly and 1000 for the whole year. So in this way, we can get huge revenues after automating the system.

Analyst : Do you have different categories in the membership?

Librarian : No, we don't have any categorization for members.

Analyst : How many books do you have in your library?

Librarian : About 10000 books

Analyst : Do you keep records for them?

Librarian : Yes, we do.

Analyst : Do you categorize your books?

Librarian : Yes, by subject.
Analyst : Would you prefer online registration rather than filling up the printed form?
Librarian : Yes, because sometimes we lose these forms and then we don't have any information about that particular member. So, it would be great to computerize the information.
Analyst : Do you have any other requirements from the new system or any other suggestions.
Librarian : It should produce various types of reports and that too faster.
Analyst : What different types of reports do you produce presently?
Librarian : We produce reports for books in the library, reports of the library members, reports about the current suppliers of the books and reports for finance.
Analyst : Can you show me some format of them?
Librarian : Yes, and we want that the same format be used by the new system.
Analyst : Yes, we'll take care of that and any more suggestions?
Librarian : No. we have covered almost all the fields.
Analyst : Thanks for your co-operation and it was nice talking to you.
Librarian : My pleasure. Bye.

Analyst also conducts unstructured interviews of few members of the library in order to know the member's viewpoint and their expectations from the new system :

Analyst : Hello. If you are free, can I ask you few questions.
Member : Sure. About what?
Analyst : Do you know that the library people are planning to automate the library management system?
Member : Yes, I know that.
Analyst : Are you ready to pay little more if there is a computerized system?
Member : Yes, I will if the overall functioning is going to improve and if it helps us in finding the books easily. But by what amount, it should matter.
Analyst : Well as far as I know they are planning to increase the membership fee from 400 to 500 for half year and 1000 for full year.
Member : Uff! That's too much. Then in that case, they should increase the number of books being issued and also the number of days a book can be kept by the member.
Analyst : Ok, then how many books to be allowed for issue and for how many days.
Member : Well should increase number of books from 3 to at least 4 and the number of days for which a book can be kept should be increased from 10 to 15 days. Only then the fee hike will be acceptable.
Analyst : Oh! Yes, they have such plans.
Member : Then it might not bother the members.
Analyst : Ok. And on what basis a search for a particular book can be done?
Member : It must be searched by subject or title.
Analyst : How often you visit this library?
Member : Daily
Analyst : Have you ever recommended this library to your friends, relatives, or to your acquaintances?
Member : Yes, I like this library and I often recommend it to my near ones.
Analyst : Till now, to how many people you have recommended?

Member : May be about 30 people.
Analyst : And how many of them have actually become its members?
Member : 22 of them.
Analyst : That's really nice. Thank You. It was nice talking to you.
Member : Thank You.

After interviewing different people and getting different perceptions about the proposed system, analyst got to know about various types of requirements about the new system.

2. Structured Interview

It is also a question and answer format but it has prescribed answers and the user has to choose the answer from these available choices. It is of either :

- *open response format* where the user is free to answer in his own words such as the question "Why are you dissatisfied with the current system?" or
- *close response format* which limits the users to opt their answer from a set of already prescribed choices such as the question "Are you satisfied with the current system?" or "Do you think that the manual processing be changed with some automated procedure?". The answers to such questions are either 'yes' or 'no'.

Advantages	Drawbacks
– Ensures uniformity in the question types as similar questions with similar choices are asked to the users. Users are not allowed to answer in their own words. Thus, there is uniformity in answering the questions. – Interviewer can easily analyze and interpret the answers as they are chosen from the already prescribed answers. – As similar set of questions are asked, therefore, there is no need of extensive interviewer training. – Less time is needed as the interviews are short because the answers are from the prescribed ones.	– Questionnaire preparation cost is high, – Users most probably do not like to choose the answers; rather they want to put up their needs in more detail and this possible by answering in their own words. – Interviewer is not able to gather requirements in detail.

☞ **Structured v/s Unstructured**

Structured	Unstructured
Less time consuming and the interviewer need not be a trained person	Very much time consuming as the interviewer has no standard set of questions. He interviews the user on the basis of his previous answers and it goes lengthy.
Easier to evaluate objectively since answers obtained are uniform.	High level of structure and mechanical questions are asked.
Users are not allowed to answer in their own words.	Users are free to answer and present their views.

Structured	Unstructured
There is uniformity in answering the questions. Users have to choose the answers from the prescribed choices.	Views are not restricted. So the interviewer gets a bigger area to further explore the issues regarding the system.
Interviewer is not able to gather requirements in detail.	Some issues might come up suddenly while answering some other question.
Interviewer can easily analyze and interpret the answers as they are chosen from the already prescribed answers.	It may happen that the interviewer goes in some undesired direction and the basic facts for which the interview was organized does not get relieved.

☞ **Break up the interview into three steps**

1. **Preparation**
   - Establish the objective for the interview
   - Identify the correct users of the proposed system
   - Identify the project development team members who will participate in the interview.
   - List out the questions and issues to be discussed in the interview
   - Review the related existing documents and systems
   - Set the time and venue to conduct the meeting
   - Inform all the stakeholders about the interview objectives, time and venue.
2. **Enactment i.e. conducting the Interview**
   - List out the issues and error conditions
   - Explore the details by asking more and more questions but relevant to the topic.
   - Take thorough notes
   - Identify and list out the unanswered questions.
3. **Follow-up i.e. making the required changes**
   - Review the interview notes thoroughly for correctness and proper understanding.
   - Write down the information under appropriate models i.e. in proper documents.
   - Identify the areas that need further clarification

3.8.4 Workflows

- Draw the Diagrammatical representation of all the information that is gathered.
- To draw a Sample diagram which can be the representation of the Workflow, follow the below points:
  - Identify agents to create the appropriate swim lanes.
  - Represent steps of workflow with appropriate ovals.
  - Connect activity ovals with arrows to show direction.
  - Use decision symbol to represent either/or situation.
  - Use synchronization bars for parallel paths.

Example : An Activity Diagram to Demonstrate Workflow : Activity diagram to withdraw money from a bank account through ATM.

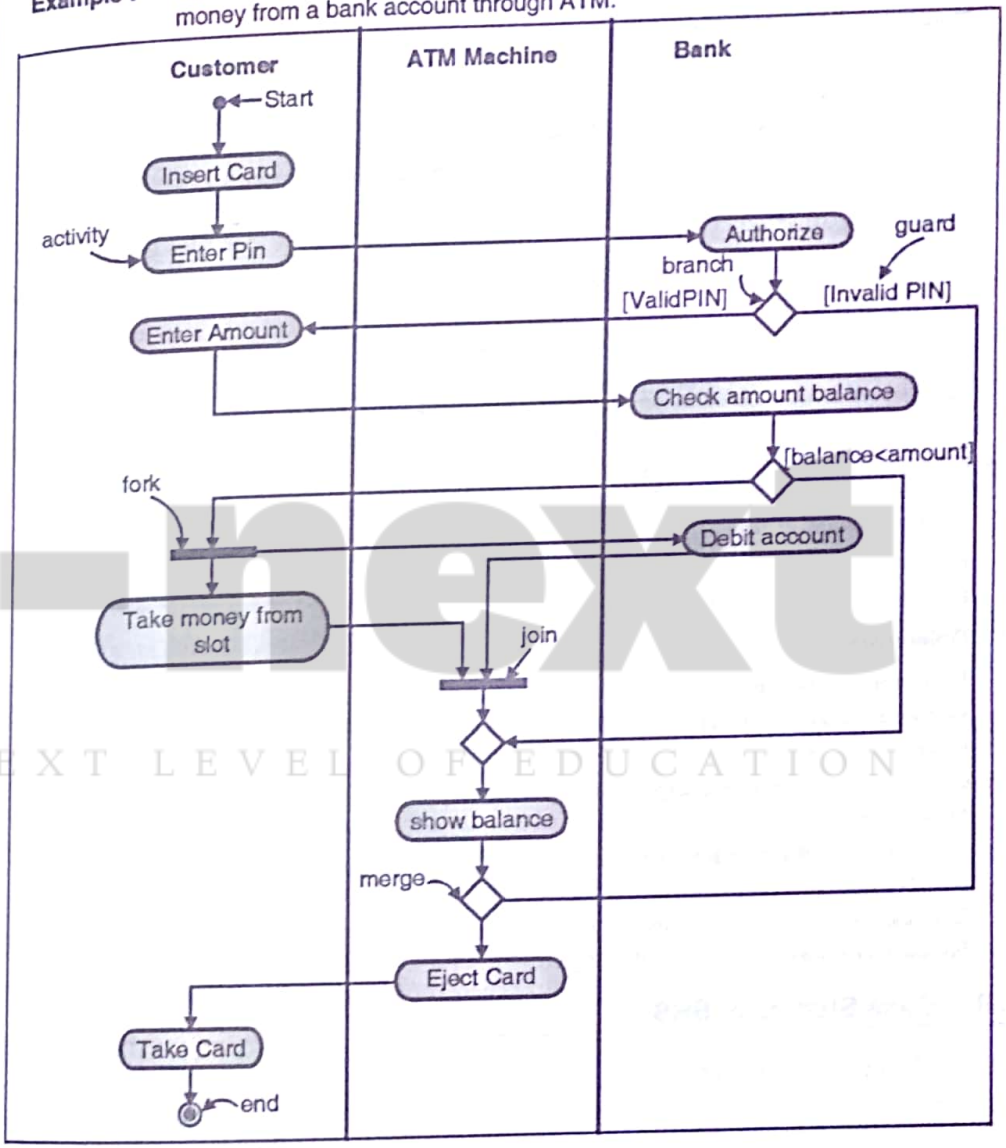


Fig. 3.8.1 : Activity diagram to withdraw money from a bank account through ATM

3.8.5 Building Prototypes

- It is often difficult for the developers as well as for the users to assume or visualize the proposed product when it is still completely new and is yet to be developed. So for the projects which are not

an extension of improvements of an existing project, it is useful to build a model of the system that the end-users and clients can visualize what the final product may look like. And these models are called as Prototypes.

- Prototype is a technique to build a quick and rough version of a desired system.
- Prototypes help to identify the inconsistencies - possible problems and usability issues.
- Prototypes allow users to make design decisions without waiting for the system to be built.
- Prototypes improve the communication between users and developers which lead to fewer changes later and hence reduce overall costs.

Issues with Prototypes

- Designers often try to use patches of code in the real system rather than wasting time in writing the code from the starting.
- Prototypes help in user interface design decisions but they cannot tell you what the original requirements were.
- Designers and end-users focus too much on user interface design and too less on services of the business process which may involve complex database updates and calculations.

3.8.6 Structured Walkthroughs

Tasks of walkthrough

- Reviews findings
- Reviews models based on findings

Objectives

- Find errors and problems
- Ensure that model is correct
- Possible alternatives to custom software development.

Break up the interview into three stages:

1. Preparation:
   - Determine documents to be reviewed
   - Select analyst i.e. the reviewers
2. Enactment i.e. conducting the Interview
3. Follow-up i.e. making the required changes.

3.9 Case Studies of SRS

3.9.1 Online Library System

Q. A library receives 1300 journals of varying periodicals. The journals received have to be recorded and displayed. Action has to be taken journals are not received in time or lost in mail. Unless request for replacement Periodical is sent quickly, it may not be possible to get replacement. Periodicals have to be ordered at different times during the year and subscription renewed in time. Late payment of subscription may lead to non-availability of earlier issues or paying higher amounts for those issues. Current manual systems is not able to meet these requirements. Prepare SRS and systems specification for an information systems for ordering Periodicals.

1. Introduction

Borrowing, returning and viewing the available books at the Library of ITM College is currently done manually where the student has to go to the Library to do the book transactions. Students check the list of available books and borrow it if it is not borrowed by any other else it is wastage of time for the student to come to the library. Then the librarian checks the student id, allows him to check out the book and then updates the member and the books database. This takes at least one to two hours for the member to go to and fro to the library and get his work done.

(a) **Product overview** : The proposed system would be Online Library Management System which would be used by members i.e. either students or professors to check the availability and borrow the books and by the librarian to update the corresponding databases.

The purpose of this SRS is to analyze the needs and features of this proposed *Online Library System* on an high-level.

(b) **Purpose** : The purpose of SRS document is to describe the external behaviour of the Online Library System - operations, interfaces, performance, quality assurance requirements and the design constraints. The SRS includes the complete software requirements for the proposed system.

(c) **Scope :** The *Online Library System* provides the members and employees of the Library with all the books information, online blocking of books and many other facilities. The Online Library System will have the following features.

- The system will be running all day.
- Users can sign-up and login to the system.
- Members can check their account and change their password whenever needed.
- The system allows the members to block the books 24 x 7 hours a day and all through the semester.
- Library staff can check which members have blocked the books and whether they can borrow any more books or not.
- Librarian can create and maintain the books catalog - add/delete books.
- The system updates the billing system whenever a member borrows or returns a book.
- We also have an order department which manages to add or remove a book from the Library.

(d) **Definitions and Abbreviations :**

- ITM – Information Technology and Management
- PIN – Personal Identification Number
- LAN – Local Area Network
- ASP – Active Server Pages
- HTML – Hyper Text Markup Language

(e) **References :**

The *SRS document* uses the following documents as references :

- ITM Information Security Requirements to provide security to the proposed system based on the security system currently used by ITM.
- The Billing System to provide the interface between the proposed system and the billing system currently in use by ITM to update the member account due whenever they borrow and return the books.

2. Overall Description

(a) Product perspective :

The Online Library System is a software package that is useful to improve the efficiency of Libraries, Librarians and Users. The complete overview of the system is as shown in the diagram below. The proposed product has interactions with various kinds of users - Librarian, Members i.e. students and professors of ITM.

The software has to interact with other systems also like: Internet, Billing System and the ITM Information Security System.

(b) Product functions

The Product functions of the system describe the different types of services provided by the system based on the type of users [Member/Librarian].

- The member is provided with the updated information about the books catalog.
- The members can borrow the books they want, if all the other required rules hold correct.
- The member can check his account information and change it any time in the given valid period.
- The members are provided with the books catalog to choose the books which they need.
- The librarian can get the information about the members who have borrowed or returned the books.
- The librarian can add/delete the books available in the book catalog.
- The due to be paid by the member is calculated if in case he is late in submitting the books or his fees is pending and the information along with the due amount about the member is sent to the university billing system.
- The system uses the ITM information security requirements to provide the login facility to the users.

(c) User characteristics

The users of the system are members and librarians of the ITM College and the administrators who maintain the system. The members and the librarian are assumed to have basic knowledge of the computers and Internet browsing. The administrators of the system have more knowledge about the internals of the system because he is responsible to rectify the system in cases of small problems that may arise due to disk crashes, power failures or any other catastrophes.

(d) Constraints

- Information of all the users (members and librarians) must be stored in a database and that must be accessible by the proposed System.
- ITM information security system must be compatible with the Internet applications.
- The Online Library System should be running all 24 x 7 hours a day.
- The users must be able to access the Online Library System from any computer that has Internet connection.
- The billing system is connected to the Online Library System and the database used by the billing system must be compatible the Internet applications.

- The users must be provided with correct usernames and passwords to login to the Online Library System.

(e) Assumptions and dependencies :

- Users have basic knowledge of computers.
- ITM College computer should have Internet connection and Internet server capabilities.
- Users knew English language as the user interface is in English
- The proposed application software can access the college student database.

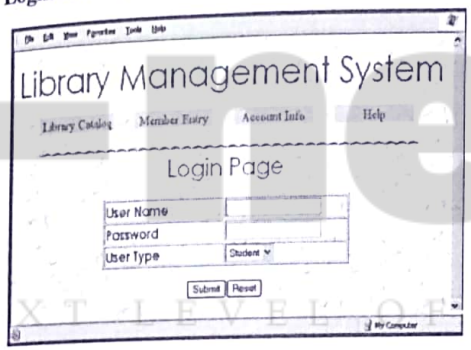
3. Specific requirements

(a) External interfaces :

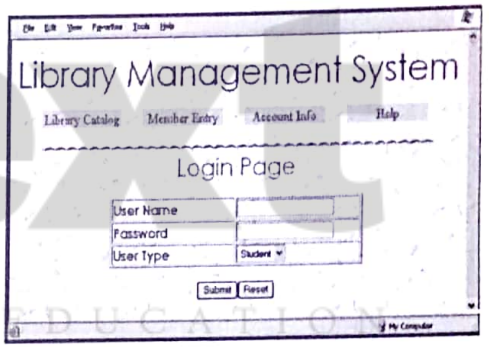
(i) User interfaces : Web Browsers : Microsoft Internet Explorer or Netscape.

The user-interface of the system shall be designed as shown in the user-interface prototypes.

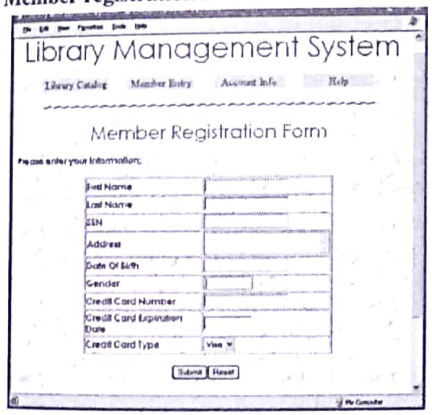
Login screen :



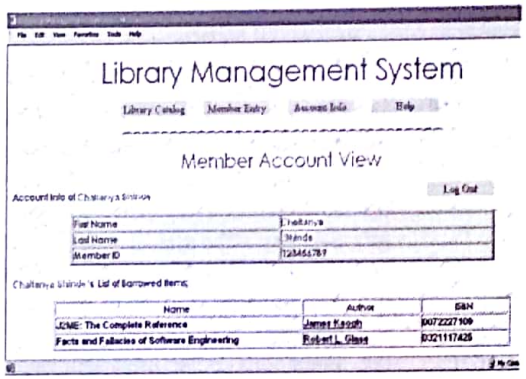
Home page of ITM library:



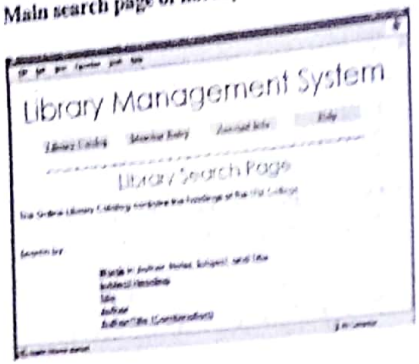
Member registration screen :



Member information once logged in :



Main search page of library catalog :



(ii) **Hardware interfaces :** LAN will be used for collecting data from the users and also for updating the Library Catalogue.

(iii) **Software interfaces :** A firewall will be used with the server to prevent unauthorized access to the system.

(iv) **Communications interfaces :** The Online Library System will be connected to the World Wide Web.

(b) **Design constraints**

- **Programming Language :** The languages that will be used for coding the Online Library System are ASP, HTML, JavaScript, and VBScript. To run ASP pages, the Internet Information Services (IIS) Server needs to be installed.
- **Development Tools :** We will make use of online references for developing the application in ASP, HTML using the two scripting languages - JavaScript and VBScript.

(c) **Functionality**

- **Login Capabilities :** The system shall provide the users with login capabilities.
- **Mobile Devices :** The Online Library System will also be supported on mobile devices such as cell phones.
- **Alerts :** The system will alert the Librarian or the administrator in case of any problems.

(d) **Software system attributes**

(i) **Reliability :** The system has to be very much reliable to avoid the damages to data and prevent from entering incorrect or incomplete data.

(ii) **Availability :** The system is available to the user all 24 × 7 hrs and 365 days a year.

(iii) **Security :** The system shall support the ITM information security requirements and use the same standard as the ITM information security requirements.

(iv) **Maintainability :** The maintenance of the system shall be done as per the maintenance contract.

(v) **Portability :** The users will be able to access the Online Library System from any computer that has internet connection.

(vi) **Performance :**

- **Response Time :** The Information page should be able to be downloaded within seconds using a 56K modem. The information is refreshed every two minutes. The response time for a mobile device must be less than a minute.
- **Throughput :** The number of transactions is dependent on the number of users.
- **Capacity :** The system is capable of handling 200 users at a time.

(e) **Other requirements**

(i) **Licensing requirements :** The usage is restricted to only ITM Library who is purchasing the Online Library System from Library InfoSys and signs the maintenance contract.

(ii) **Applicable standards :** The ISO/IEC 6592 guidelines for the documentation of computer based application systems will be followed.

3.9.2 Purchase Order System

Q. The purchase order system functions as follows : After receiving the purchase requisition from store department, enquiries are made to various suppliers. The suppliers send quotation to the company. All quotation are analysed and final selection of supplier is done and accordingly purchase order to respective supplier are send. The supplier sends invoice along with raw material. Prepare SRS for the above system.

1. Introduction

Store department of a company offers purchase requisition. Upon receiving this requisition, it makes enquiries to various suppliers. Store department then gives specification for stock, name of products and their quantity. Depending on this specification form, suppliers send their quotations of stock quantity and rate of the required products. Depending on the number of suppliers submitting the Quotation, comparative charts are prepared to analyze and then final selection of supplier is done. All these functions are currently done manually which takes a number of days to send the requisition and then receive quotations and then analyze them manually one by one.

(a) **Product Overview**

The proposed system would be Online Purchase Order (OPO) system which would be used by company staff and various suppliers to offer the purchase requisition and send the corresponding quotations, also computationally analyze the quotations so as to select the appropriate suppliers.

(b) **Purpose**

The purpose of this document is to describe the external behaviour of the Store Departments, description of products, including handling persons, product perspective, overview of requirements, general constraints. It will also provide the specific requirements and functionality needed for this system.

(c) **Scope**

The purchase order system assists the company to purchase product according to their needs and specification at minimum rate and in minimum time so that it fulfills storekeeper's needs and specifications. The availability of the product will be fulfilled before the shortage and requirements rose. The OPO System will have the following features :

- Stock Maintenance
- Regular product master
- Payment and receipt maintenance according to date and time specification.
- Report Generation (weekly, Monthly, Yearly) of various details such as :

(i) Total stock of products
(ii) Total Form collection (total number of applied suppliers)
(iii) List of products expired.

(d) Definition, Acronyms and Abbreviation

- OPOS - Online Purchase Order System
- ITM – Information Technology Management

(e) References

The *SRS document* uses the following documents as references :

- Store requisition form and supplier's quotation form as a sample format so as to de the form in the same format when the current system is automated.
- Information Security Requirements to provide security to the proposed system base the security system currently used by the company.
- The Billing System to provide the interface between the proposed system and the bill system currently in use by store to update the product information whenever a p product is purchased or ordered (sold).

2. Overall Description

(a) Product Functions

- OPOS generates Quotation form for different supplier for same product requirements.
- OPOS will be automated and centralized hence it will create a comparative chart for sam product specification but different rating.
- Also that chart have to show sorting of rates through which we can easily analyze th suppliers.
- OPOS keeps the record of the product code, product name, quantity and the oth specification.

(b) User Characteristics

System user is storekeeper/ Store Manager/suppliers. The users have elementary compute knowledge.

(c) Design Constraints

- OPOS requires a computer equipped with 133 MHZ Intel Pentium Processor, with minimum 64/a24 MB RAM, a CPU speed of 200 MHZ or above for good performance.
- At least WIN OS should be there.
- Minimum 64 MB RAM recommended.
- ORACLE server should be installed.
- Minimum DOT MATRIX PRINTER should be installed.

(d) Assumptions and dependencies

- Users have basic knowledge of computers.
- Company's computers should have Internet connection and Internet server capabilities.
- Users knew English language as the user interface is in English
- The proposed application software can access the company product database.

3. Specific Requirements

(a) User interface

All the forms are GUI based and are used by user and manager interactively except the Quotation form.

(b) Database Names

- Product Master
- Stock Mater
- Concession Rule Details
- Tax Details
- Suppliers' Norms
- Departure details

(c) Functional Requirements

Determine the last date and time for the filling of Quotation form and constraints are satisfied.

- No more than one Quotation form should be filled at the same time for same product specification.
- The supplier should follow the whatever stock requirement specification.
- Preferences is given to discount offering and quality maintained suppliers.
- The supplier should follow rules an constraint as defined, in any manner they does not violate these constraints.
- Allow the user to maintain separate form for each supplier.
- Allow the user to maintain a database of products, quantity.
- To maintain a list of all product and stock specification available in store department.
- To generate report for number of suppliers, supplying number of products, quality details, rates, discount and tax calculation, date of supplying after receiving order etc.

(d) Software System Attributes

- The generated problems should be solvable.
- The database should be properly synchronized with record generated.

(i) **Reliability :** The system has to be very much reliable to avoid the damages to data and prevent from entering incorrect or incomplete data.
(ii) **Availability :** The system is available to the user all 24 × 7 hrs and 365 days a year.
(iii) **Security :** The system shall support the ITM information security requirements and use the same standard as the ITM information security requirements.

(iv) **Maintainability :** The maintenance of the system shall be done as per the maintenance contract.

(v) **Portability :** The users will be able to access the OPOS from any computer that has Internet connection.

(vi) Performance :

- **Response Time :** The Information page should be able to be downloaded within seconds using a 56K modem. The information is refreshed every two minutes. The response time for a mobile device must be less than a minute.
- **Throughput :** The number of transactions is dependent on the number of users.
- **Capacity :** The system is capable of handling 200 users at a time.

4. Acceptance Criteria

Before accepting the system, the developer must demonstrate that the system works on the number of stock data, product, quantity specification. The developer will have to show through test cases that all conditions are satisfied.

3.9.3 Hospital Management System

1. INTRODUCTION

➢ 1.1 PRODUCT OVERVIEW

- This Software Requirements Specification formally specifies Hospital Patient Management System (HPMS). It includes the resulting decisions of both - business analysis and systems analysis efforts. The objective of this document is to describe the system's high level requirements such as functional requirements, non-functional requirements and business rules and constraints.
- The detail structure of this document *is organized* as follows:

  Section 2 of this document provides overview of the business domain functions that the proposed Hospital Patient Management System (HPMS) will support.

  Section 3 presents detail requirements and overview of the Hospital Patient Management System's functions.

➢ 1.2 PURPOSE

- The purpose of this SRS document is to describe the external behaviour of the Hospital Patient Management System (HPMS) - operations, interfaces, performance, quality assurance requirements and the design constraints.
- Developers should refer to this document as the only source of requirements for the project. They should not consider any requirements written or verbal as valid until they appear in this SRS document or in its revised version.

➢ 1.3 SCOPE

- The system will allocate beds to patients on priority basis, and assigns doctors to patients in the designated wards as needed.
- Doctors will use the system to keep track of the patients assigned to them.
- Nurses will use the system to keep track of available beds, patients, and the type of medication given and to be required for each patient.

- The current system in use is a paper-based system which cannot provide updated lists of patients within a reasonable timeframe. Therefore, the aim of the system is to reduce over-time pay and increase the number of patients that can be treated accurately.

➢ 1.4 AUDIENCE

The intended audience include all stakeholders such as administrative staff, doctors, nurses, surgeons and developers.

i. *Front-desk staff:* They are responsible for patient's check-in or notification of appropriate people (e.g. notify administrator or nurse when an event occurs).

ii. *Administrators*: Every administrator has basic computer knowledge. They are responsible for all the scheduling and updating the day/night employee shifts; and assigning doctors and nurses to patients.

iii. *Nurses*: They are responsible for assigning patients to appropriate wards if the beds are available, otherwise putting patients on the waiting list.

iv. *Doctors and Surgeons*: They will use the HPMS to check their patient's list and their duty schedule.

➢ 1.5 DEFINITIONS, ACRONYMS AND ABBREVIATIONS

HPMS	Hospital Patient Management System
PHN	Personal Health Number on health card
Report	an account of patients
Database	collection of information in a structured form
Front-desk staff	administrative staff at reception desk
Logon ID	a user identification number to enter the system
Password	a word that authenticates a user to the system
Web-based application	an application that runs on Internet
ID	Patient Identification number
GUI	Graphical User Interface
SRS	Software Requirements Specification

2. OVERALL DESCRIPTION

➢ 2.1 PRODUCT PERSPECTIVE

This HPMS is a self-contained system that manages at most activities of the hospital such as bed assignment, operations scheduling, personnel management and administrative issues.

➢ 2.2 PRODUCT FUNCTIONALITY

The system functions are as follows:

i. *Registration :* When a patient is admitted, the front-desk staff checks whether the patient is already registered with the hospital. If so, his/her PHN is entered into the computer else a new PHN is given to this patient and his information including his date of birth, address and contact number is entered into the system.

ii. *Consultation :* The patient then goes to consultation-desk to explain his/her condition so that the consulting nurse can decide what kind of ward should be assigned to him/her. There are two possible situations :

If a bed is available in the ward, then the patient is allotted that bed and asked to wait for the doctor to come.

If there is no bed, the patient is put on a waiting list until a bed becomes available.

iii. *Check out* : When a patient checks out, the front-desk staff shall delete that patient's PHN from the system and the just vacate bed is included in available-beds list.

iv. *Report Generation* : The system generates reports of the following information: patients, bed availability and staff schedules after every six hours.

2.3 USER CHARACTERISTICS

The system will be used by the hospital staff who include administrators, doctors, nurses and front-desk staff who will be trained on using the system. The system is also designed to be user-friendly by making use of a Graphical User Interface (GUI).

2.4 OPERATING ENVIRONMENT

i. The system will use MySQL Database which is open source and free.

ii. The Development environment will be Windows XP SP2.

iii. The system will be a Web-based application.

2.5 DESIGN AND IMPLEMENTATION CONSTRAINTS

- The system must be delivered by August 25th 2012.
- The existing Telecommunication infrastructure of HPMS is based on IEEE100802 standards and the system must confirm to this standard.

2.6 ASSUMPTIONS AND DEPENDENCIES

i. It is assumed that 100 IBM compatible computers will be available before the system is installed and tested.

ii. It is assumed that the hospital will have enough trained staff to take care of the system.

3. SPECIFIC REQUIREMENTS

3.1 FUNCTIONAL REQUIREMENTS

i. Registration

- Add patients : The front-desk staff will use HPMS to add new patients to the system.
- Assign ID : The front-desk staff will use HPMS to give each patient an ID and add it to the patient's record.

ii. Consultation

- Assign Ward : The consulting nurse will use HPMS to assign the patient to an appropriate ward.
- Assign to Waiting List : The consulting nurse will use HPMS to assign Patient to a waiting list if no bed is available.

iii. Medical Management

- Assign Doctor : The administrative staff will use HPMS to assign a doctor to a given patient.
- Assign Nurse : The administration staff will use HPMS to assign a nurse to a given patient.
- Inform Doctors : The HPMS will inform doctors about the new patients.
- Inform Nurses : The HPMS will inform the nurses about new patients.

- Emergency Case : In case of emergency, the administrative staff will use HPMS to assign an emergency room, doctors and nurses immediately to the patient.
- Surgery case : In case of surgery, the administrative staff will use HPMS to assign a surgery room, surgeon and nurses to the patient.
- Generate Report : The HPMS will generate the patient's situation record every two hours for normal patients, and for every half hour for severe patients.
- Record Procedure : The complete statement procedure for the patient will be recorded by the system.
- Inform Patient : The HPMS will automatically inform about the patients who are on waiting list and about available beds whenever they become available.

iv. Check Out

- Delete Patient ID : The administrative staff will use HPMS to delete the ID of the patient from the system when the patient checks out.
- Add to beds-available list : The administrative staff allowed to put the beds just evacuated in beds-available list.

v. Report Generation

- Patient information : Every six hours the HPMS will generate reports about the following information - patient's PHN, patient's name, ward name, bed number and the doctor's name.
- Bed Availability : Every six hours the HPMS will generate report on bed availability that includes ward name, bed number, occupied/unoccupied.
- Staff Schedule : Every six hours the HPMS will generate report on staff schedule that includes staff ID, staff name, staff type, duty shift.

vi. Database

- Patient Mandatory Information : first name, last name, phone number, personal health number, address, postal code, city, country, patient identification number.
- Update Patient Information : The HPMS shall allow the user to update any of the patient's above information.
- Search for Patient : The HPMS will allow user to search for patient's information by last name or PHN or patient ID.
- Staff Mandatory Information : identification number, first name, last name, phone number, address, postal code, city, country, employee type, duty schedule.
- Update Staff Information : The HPMS will allow user to update any of the staff's information described above.
- Search Employee Information : The HPMS will allow user to search for employee information by last name, or ID number.
- Ward Types : Maternity, Surgical, Cancer and Cardiac.
- Ward Information : ward name, ward number, list of rooms in ward.
- Room Information : room number, list of beds in room, full/not full.
- Bed Information : bed number, occupied/unoccupied, patient PHN.
- Ward Search : The HPMS will allow users to search the ward, room, and bed directly by ward number, room number and bed number respectively.

3.2 BEHAVIOUR REQUIREMENTS

➢ 3.2.1 USE CASE VIEW

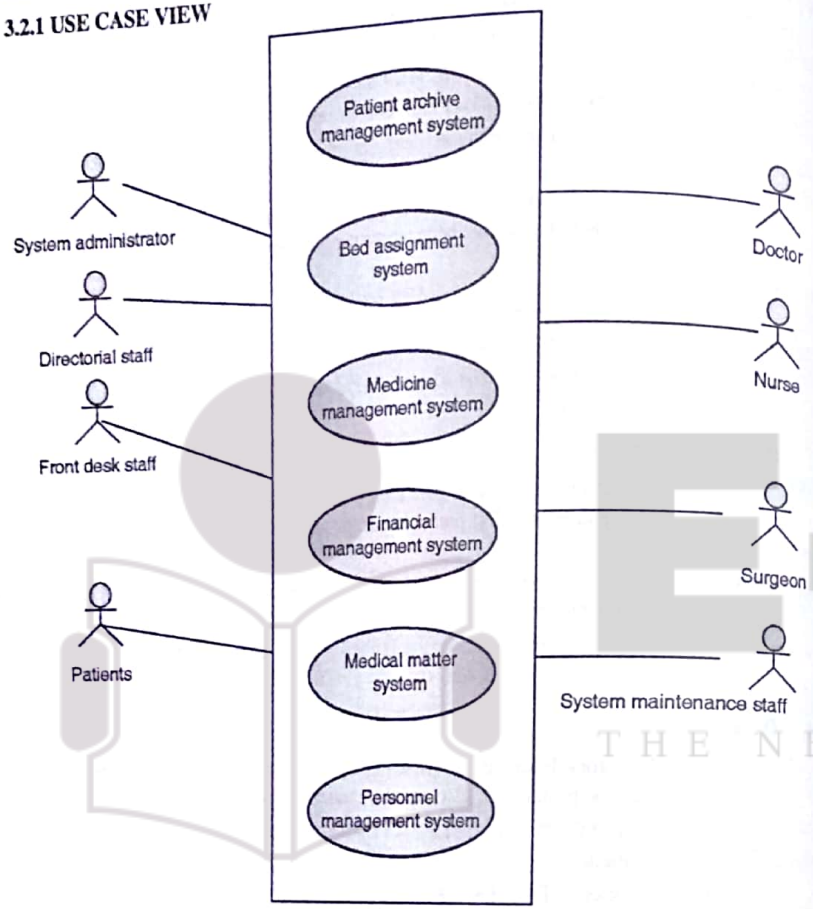


Fig. 3.9.1

4. OTHER NON FUNCTIONAL REQUIREMENTS

➢ 4.1 RELIABILITY

The system has to be very much reliable to avoid the damages to data and prevent from entering incorrect or incomplete data.

➢ 4.2 AVAILABILITY

The system is available to the user all 24 × 7 hrs and 365 days a year.

➢ 4.3 SAFETY and SECURITY

- Patient Identification : The system identifies the patient using PHN.
- Login ID : Any user who uses the system shall have a Login ID and Password.

- Modification : Any modification such as insert, delete, update for the Database shall be allowed only to the administration staff.
- Compliance : The system must comply with the Regional Health Authority Regulations.
- Front Desk staff Rights : Front Desk staff can view and add all information in HPMS, but shall not be able to modify any information in it.
- Administrators' Rights : Administrators can view and modify all information in HPMS.
- Nurses' Rights : Nurses can only view the information in HPMS.
- Doctors Rights : Doctors can view all information in HPMS.

➢ 4.4 MAINTAINABILITY

- Back Up : The system will provide back-up capability to the Data.
- Errors : The system shall keep a log of all types of errors.

➢ 4.5 PERFORMANCE

- Response Time: The system will give responses in a second after checking the patient's information.
- Capacity : The System will support 1000 people at a time.
- User-interface : The GUI will respond within 5 seconds.
- Conformity : The systems must conform to the Microsoft Accessibility guidelines.

3.9.4 Catering System

Example 3.9.1 :

Joshi Caterers Pvt. Ltd. wants to develop the order processing and billing software which presently works as under:

Company collects order from different corporate customers or individuals. The customer fills up the order form describing various details like order details, customer details, menu item details and number of thalies. 50% advance is collected only from individual customers. Then after receiving the orders, Kitchen Order Ticket (KOT), is issued to the kitchen and then kitchen issues the list of raw material (excluded from available stock) to be purchased from the suppliers. Purchase order is given and ordered material is received from the fixed suppliers and forwarded further to the kitchen. After completion of the delivery of the order, bill is issued to the customer on the basis of actual number of thalies or ordered number of thalies whichever is more. Payment is accepted and receipt is given to the customer. Prepare SRS and system specification for the above system.

Soln. :

1. INTRODUCTION

➢ 1.1 PRODUCT OVERVIEW

The objective of the new system is to provide scalability, reliability and efficiency to enable productivity increases over the ordering and invoicing process resulting in reduced administration and processing costs.

➢ 1.2 PURPOSE

This Software Requirements Specification formally specifies Joshi Caterers Pvt. Ltd. (JCPL). It includes the resulting decisions of both - business analysis and systems analysis efforts. The

purpose of this document is to specify the software requirements for JCPL and provide an ongoing reference for project staff.

1.3 SCOPE

The JCPL system will allow the following functionalities:

i. *Collect order:* customer fills up the order form describing various details like order details, customer details, menu item details and number of thalies
ii. *Advance Payment* : 50% advance is collected only from individual customers.
iii. *Issue KOT* : After receiving the orders, Kitchen Order Ticket (KOT), is issued to the kitchen.
iv. *Check Stock Level* : kitchen issues the list of Utensils and Food Items below minimum stock level
v. *Purchase Order* : Purchase order is given and ordered material is received from the fixed suppliers and forwarded further to the kitchen.
vi. *Issue bill* : After completion of the delivery of the order, bill is issued to the customer on the basis of actual number of thalies or ordered number of thalies whichever is more.
vii. *Payment* : customer makes payment and gets receipt.
viii. *Total Income* : income generated from services annually

1.4 AUDIENCE

The intended audiences include administrative staff, customers and developers.

i. *administrative staff:* Every administrator has basic computer knowledge. They are responsible for collecting orders, informing the suppliers for raw material and collecting payment from the customers.

1.5 DEFINITIONS and ABBREVATIONS

- *Acceptance* – the date on which the contract is made.
- *Change Database* - manages all change proposals and this is linked to a version management system.
- *Critical Failure* - failure that results in System's data not being available to more than 3 users.
- *Data Recovery* - restoring data that has been physically damaged.
- *Fault Tolerance* - a technique that ensures that system errors do not result in system failures.
- *Form Editor* – allows change proposal forms to be filled in once again.
- *Interface Generator* - graphical screen design system where interface components such as menus, field, icons and buttons are selected from a tool box and positioned on the interface.
- *System Availability* –total number of hours the system was Operational divided by the total number of hours the system was under a critical failure.
- *JCPL* - Joshi Caterers Private Limited.
- *KOT* - Kitchen Order Ticket
- *CM* - Configuration Management a standard process involving the application of predetermined procedures. They require careful management of very large amounts of data and attention to detail is essential.

2. OVERALL DESCRIPTION

2.1 PRODUCT PERSPECTIVE

The proposed system would be Online Catering Management System which would be used for order processing and billing.

As indicated in below figure, interaction between the subsystems will be via the underlying data stores. Each of the subsystems in self-contained, hence, there is no direct interaction required between any of the subsystems.

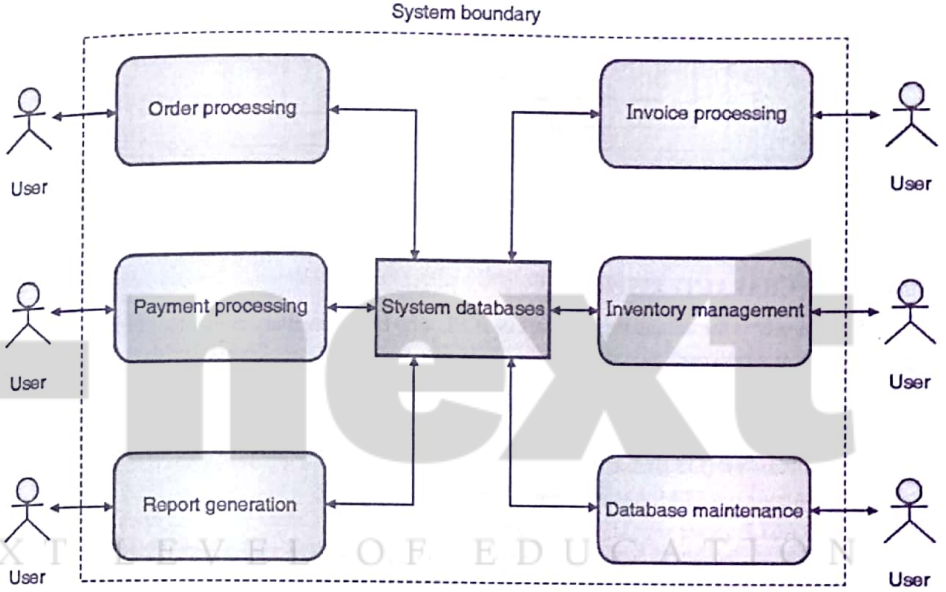


Fig. P. 3.9.1

2.2 PRODUCT FUNCTIONALITY

- JCPL collects order from different corporate customers or individuals.
- JCPL stores various details like order details, customer details, menu item details and number of thalies.
- JCPL generates KOT which is then issued to the kitchen.
- JCPL finds the list of Utensils and Food Items below minimum stock level.
- JCPL places purchase order to its suppliers giving the list of requirement.
- JCPL generates catering bill and issues acknowledgement after delivering the order to the customer.

All the above functions can be summarized through the following context level diagram;

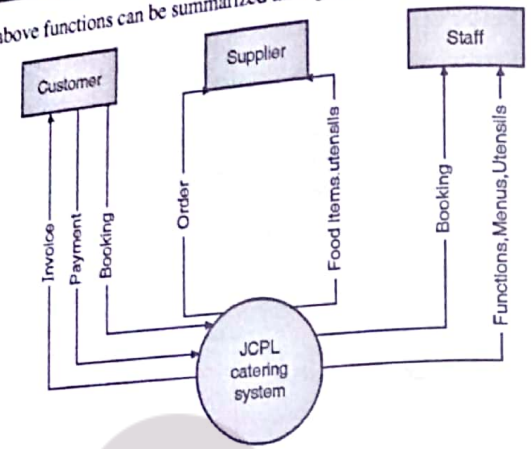


Fig. P. 3.9.1(a)

2.3 USER CHARACTERISTICS

System user is the administrative staff of JCPL who has elementary computer knowledge.

2.4 DESIGN CONSTRAINTS

- JCPL requires a computer equipped with 133 MHZ Intel Pentium Processor, with a minimum 64/a24 MB RAM, a CPU speed of 200 MHZ or above for good performance.
- At least WIN OS should be there.
- Minimum 64 MB RAM recommended.
- ORACLE server should be installed.
- Minimum DOT MATRIX PRINTER should be installed.

2.5 ASSUMPTIONS AND DEPENDENCIES

- Users have basic knowledge of computers.
- Users knew English language as the user interface is in English
- The proposed application software can access the company product database.

2.6 USER DOCUMENTATION

The System shall be delivered to JCPL attached with all of the following documents:

- A complete online help facility that will assist system users to easily operate the system.
- Electronic user manual on CD-ROM in Adobe Acrobat format which will be useful for initial training of System users.
- Instruction manual for System Administrator on CD-ROM in Adobe Acrobat format which will be useful for initial training of System Administrators.

3. SPECIFIC REQUIREMENTS

3.1 USER INTERFACE

All the forms are GUI based.

3.2 DATABASE NAMES

- Menu Master
- Stock Mater
- Order Master
- Concession/Discount Rule Details
- Suppliers Details
- Payment Details
- Customer Details

3.3 FUNCTIONAL REQUIREMENTS

i) Customer Details

a) JCPL enables addition, modification, deletion, display and storage of information about customers such as a unique identifier, name, address, contact number and E-mail address (optional).

b) JCPL avoids redundancy – while the user is adding custoemr details, if the customer's name and address match an existing customer entry, the System will reject the new information and tells the user that the customer already exists.

ii) Supplier Details

a) JCPL enables addition, modification, deletion, display and storage of information about suppliers such as a unique identifier, name, address, contact number and E-mail address (optional).

iii) Food Item and Utensil Details

a) JCPL enables addition, modification, deletion, display and storage of information about food items and utensils.

b) JCPL stores following information about food item and utensils - a unique identifier, Food Item and Utensil Name, suppliers who provide this food items and utensils, and per-unit price that each supplier charges for the food item and utensil.

c) JCPL provides following details about below minimum level of food items and utensils in stock :

- The number of units of the food items and utensils currently in stock
- The number of units of the food items and utensils currently on order
- The price per unit of the food item and utensil
- The stock reorder level or minimum stock level

iv) Service Details

a) JCPL enables addition, modification, deletion, display and storage of information about services.

b) JCPL provides following details about each service:

- A unique identifier
- Service Type
- The number of food dishes on ordered menu
- The number of utensils required for the service

- The price per service

v) **Booking Details**

a) JCPL enables addition, modification, deletion, display and storage of information about bookings.

b) JCPL will provide following information about each booking:
- A unique identifier
- booking date
- the customer that made the booking
- The quantity and type of each service booked
- The invoice associated with the booking

c) JCPL stores 50% advance payment details for a booking if the customer is an individual person.

d) JCPL generates KOT after accepting the menu order from the customer. This KOT is then issued to the kitchen.

vi) **Invoice Details**

a) JCPL generates catering bill/invoice on the basis of actual number of thalies or ordered number of thalies whichever is more.

b) JCPL stores following information for each invoice:
- A unique identifier
- Invoice date – the date that the invoice is entered
- The customer to whom the invoice applies
- The services that make up the invoice

vii) **Payment Details**

a) JCPL enables addition, modification, deletion, display and storage of information about payments.

b) JCPL provides following information for each payment:
- A unique identifier
- Payment date
- The customer making the payment
- The invoice against which the payment is being done.
- The amount paid

viii) **Unique Identifier Selection**

JCPL shall automatically assign and display a unique numerical identifier when a new instance of any of the following objects is created :
- A customer,
- A supplier,
- A menu,
- A company,
- A food item,
- A menu1,

- An order,
- A payment, or
- An invoice.

ix) **Report Generation**

a) JCPL shall display a requested report at the user's data display device by sorting the information according to the preference specified by the user.

b) JCPL shall include the following information at the start of each report:
- The date and time at which the report was produced, and
- The title of the report.

A) **Minimum Stock Level Report**

a) When requested by a user, JCPL system shall automatically generates a minimum stock level report for all food items and utensils where the quantity in stock is below the stipulated minimum level for that food item or utensil.

b) JCPL system shall display at least the following information when a minimum stock level report has been generated:
- The nominated minimum storage level for the food item or utensil
- The actual storage level for the food item or utensil.

B) **Customer Report**

a) JCPL System shall automatically generate customer monthly service report that lists all details of customer services during the last month.

C) **Nil Payment Report**

a) JCPL System shall automatically generate nil payment report that lists all details of all invoices raised in the previous month against which no payments have been made.

D) **Total Income Report**

a) JCPL System shall automatically generate a total income report that calculates the total income generated from functions and services during the whole annual year.

4. OTHER NON FUNCTIONAL REQUIREMENTS

4.1 PERFORMANCE

- The System shall display the latest information stored in the System at the time of any user request.
- The System shall respond to all user commands within an average time of 5 seconds of a request being made.
- The System shall complete all users commands and display the result within an average time of 15 seconds of a request being made.

4.2 PORTABILITY

The System shall be able to run under any combination of each of the following operating systems:

a) Windows XP b) Windows 2000,
c) Solaris d) Unix.

4.3 SECURITY

The System shall provide appropriate facilities to ensure that only authorised users have access to the information stored in the System.

Review Questions

Q. 1 What is SRS ? Explain need & benefits of SRS.

Q. 2 Explain the issues of requirement gathering ?

Q. 3 What are the techniques of requirement gathering ?

Q. 4 What is Requirements engineering?

Q. 5 Describe the requirements engineering tasks?

Q. 6 What is the need of Requirements' Validation ?

Q. 8 Explain Requirement Management.

Q. 9 List the various stakeholders involved in requirement analysis ?

Q. 10 Write an SRS for Maharashtra State Electricity Board. Assume the functional and non-functional requirements.

□□□

UNIT I

CHAPTER 4

Object-oriented Design using UML

Syllabus

Class diagram, Object diagram, Use case diagram, Sequence diagram, Collaboration diagram, State chart diagram, Activity diagram, Component diagram, Deployment diagram

4.1 Introduction

- Unified Modeling Language (UML) begins by constructing a model which is a simplification of reality.
- Model is an abstraction of some underlying problem.
- Model is a complete description of a system from any particular perspective constructed to understand the system before building the new or before modifying the existing system.

Static and Dynamic Models

- A *static model* is the snapshot of system's parameters at rest or at a specific point of time. For example, a customer could have more than one account at a time. Static models have stability and there is no change of data in future. The UML *class diagram* is a type of a static model.
- A *dynamic model* is a collection of behaviours of the system. It represents how objects interact with each other in order to perform any task. The UML *interactive* (*sequence and collaborative*) and *activity diagrams* are types of a dynamic model.

Benefits of a Model

1. Clarity

Models make it easier to. We can very easily express complex ideas using models as it allows visual examination of the whole system possible.

2. Complexity

Models reduce complexity by separating unimportant aspects from those that are important.

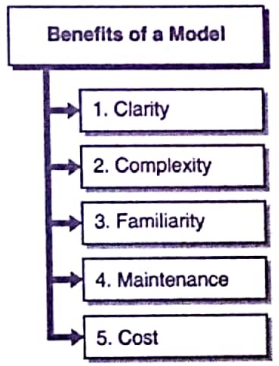


Fig. C4.1 : Benefits of a Model