Unit 3 Software Requirement Analysis

Unit Outcomes (UOs) :

3.1 Identify software requirements

3.2 Prepare software requirement specifications

3.1.1 Requirements Gathering (Collect requirement from stakeholders)

- As per IEEE the definition of requirement is → **A condition or capability needed by a user to solve a problem or achieve an objective**
- Requirements of a customer play a key role in developing any software product.
- Requirements are the description of features and functionalities of the target system. It convey the expectations of users from the software product.
- The task of gathering requirements and analysing them is performed by a *System analyst.*
- **Collecting all the information from the customer and then analyze the collected information to remove all ambiguities and inconsistencies from customer perception.**
- It is the discovery phase of the project development.
- Mainly two activities are concerned :

Requirement gathering	Requirement analysis

❖ Requirement gathering

- It is usually the first part of any software product.
- This is the base for the whole development effort.
- Requirements gathering is the process of understanding what you are trying to build and why you are building it.
- The goal of the requirements gathering activity is → to collect all related information from the customer regarding the product to be developed.
- This is done to clearly understand the customer requirements so that incompleteness and inconsistencies are removed.
- In this phase, meeting with customers, analysing market demand and features of the product are mainly focused.
- So, activity of market research (for competitive analysis) is done.
- It involves interviewing the end-users and studying the existing documents to collect all possible information.
- One of the important dimensions of requirement gathering is stack holder → who is the person associated with the project directly or indirectly.

- **Requirement gathering activities are**

  Some of the important requirement gathering activities are explained below.

- **Interview with end users or customers**
  - → This technique is most effective among all.
  - → In this method, the system analyst talks to the end user and clients directly and collecting all the project related information.
  - → It is the responsibility of system analyst to extract proper and relevant information.
- **Survey / Questionnaire**
  - → It is another effective method to collect information and requirements within a short time.
  - → The questionnaire is draft thereafter survey is taken from the stake holders.
- **Brainstorming**
  - → This technique is particularly conducted to solve complex problems.
- **Observation**
  - → Under the observation method, the responsible person observes the team in working environment and gets ideas about the software and subsequently document the observation.
- **Interface analysis**
  - → It is a special technique in which application development are determined and their interaction with other software components is measured.
- **Task analysis**
  - → Task analysis is the process of understanding the user's task thoroughly enough to help design a computer system that will effectively support users in doing the task.
- **Form analysis**
  - → Form analysis is an important and effective requirement gathering activity that is undertaken by the analyst when the project involves automating an existing manual system.
- **Group discussion**
  - → This technique is used in requirement gathering to get as many ideas as possible from group of people.
- **Prototyping**
  - → Prototyping is a relatively modern technique for gathering requirements.
  - → In this approach, you gather preliminary requirements that you use to build an initial version of the solution - a prototype. You show this to the client, who then gives you additional requirements. You change the application and cycle around with the client again.
  - → This repetitive process continues until user satisfaction.
- **Analyzing existing documents**
  - → This helps to system analyst to understand the project or system in current situation.
  - → This will help the analyst formulate questions for interviews or questionnaires to ask of stakeholders, in order to gain additional requirements.

3.1.2 Analyze the requirement (OR Requirement Analysis)

- The goal of the requirement analysis activity is → to clearly understand the exact requirements of the customers.
- **IEEE** defines requirement analysis as (1) the process of studying user needs and (2) The process of studying and refining system hardware or software requirements.
- Requirement analysis helps to understand, interpret, classify, and organize the software requirements in order to assess the feasibility, completeness, and consistency of the requirements.
- Generally it is used to identify possible solutions to problems, and clarify details of opportunities.
- **Requirement analysis involves:**

➥ *Eliciting requirements* (requirements - ને છૂટી પાડવી) : requirements are eliciting by communicating with customers and find their exact need.

➥ *Analyzing requirements* (છૂટી પાડેલી requirementsનું analysis કરવું) : requirements are then analyzed to make it complete, clear and unambiguous.

➥ *Requirements recording or storing* (બધી જ જરૂરી requirementsને record કરવી) : all the requirements are recorded in form of use cases, process specifications, natural language documents etc.

- **System analyst should solve some of the following questions :**

➥ What is the problem ?

➥ What are the inputs and outputs ?

➥ What is important to solve ?

➥ What are the complexities ?

➥ What are the solutions ?

- Change in the environment or technical aspects may affect the requirement analysis process.
- System analyst identifies and resolves various requirements problems.
- For that, analyst has to identify and eliminate the problems of anomalies, inconsistencies and incompleteness. **Anomaly** is the ambiguity in the requirement, **Inconsistency** contradicts the requirements, and **Incompleteness** may overlook some requirements.
- Analyst detects above problems by discussing with end-users.
- Requirement analysis is necessary to develop the system that meets all the requirements of the end-user.
- Finally, make sure that requirements should be specific, measurable, timely, achievable and realistic.
- Output of this activity is →**SRS** (Software Requirements Specification).

Software requirements are the description of services which software will provide to end user.

Requirement gathering and requirement analysis & specification collectively called 'Requirement Engineering'.

3.2.1 Software Requirement Specification (SRS)

- SRS is the output of requirement gathering and analysis activity.
- SRS is a document created by system analyst after the requirements are collected from various sources.
- SRS is a detailed description of the software that is to be developed. It describes the complete behaviour the system.
- SRS describes *'what'* the proposed system should do without describing *'how'* the software will do (what part, not how).
- It is working as a reference document to the developer.
- It provides guideline for project development, so minimizes the time and efforts for software development.
- SRS is actually a contract between developer and end user. That helps to dissolve the disagreement.
- The SRS translates the ideas of the customers (input) into the formal documents (output).
- The SRS document is known as black-box specification, because:
  - In SRS, internal details of the system are not known (as SRS doesn't specify how the system will work).
  - Only its visible external (i.e. input/output) behaviour is documented.
- SRS documents serves as contract between customer and developer, so it should be carefully written. (Sometimes SRS is also written by the customers also).
- The organization of SRS is done by the system analyst.

❖ **Importance of SRS (Features of SRS)**

- SRS provides foundation for design work. Because it works as an input to the design phase.
- It enhances communication between customer and developer because user requirements are expressed in natural language.
- Developers can get the idea what exactly the customer wants.
- It enables project planning and helps in verification and validation process.
- Format of forms and rough screen prints can also be represented in SRS.
- High quality SRS reduces the development cost and time efforts.
- As it is working as an agreement between user and developer, we can get the partial satisfaction of the end user for the final product.
- SRS is also useful during the maintenance phase.

❖ **Users of SRS**

The list of users along with their role is explained below:

✓ *Customers and users*: for understanding what they are expected to get.

✓ *Project managers:* to estimate and plan the project to deliver the system.

✓ *Designers and programmers*: to know what to build.

✓ *Testers*: to prepare for testing activities.

- ✓ *Maintenance teams*: for understanding the system that they will maintain.
- ✓ *Trainers*: to prepare training material for training the end users.

❖ **Characteristics of a good SRS**

- **Concise** : SRS should contain brief and concise information regarding the project; no more detailed description of the system should be there.
- **Complete** : It should be complete regarding the project, so that can be completely understood by the analyst and developers as well as customers.
- **Consistent** : An SRS should be consistent through the project development. Requirements may not be conflict at the later stage.
- **Conceptualintegrity** : SRS should clearly provide the concepts of the system, so that can be read easily.
- **Structured** : SRS should be well structured to understand and to implement.
- **Black box view** : SRS should have black box view means; there should not be much detailing of the project in it (only describe *what* part, not *how*).
- **Verifiable** : It should be verifiable by the clients or the customers for whom the project is being made.
- **Adaptable** : It should be adaptable in both sides from the clients as well as from the developers.
- **Maintainable** : SRS should be maintainable so in future changes can be made easily.
- **Portable** : It should be portable as if we can use the contents of it for the same types of developments.
- **Unambiguous** : There should not be any alternates of SRS that creates ambiguity.
- **Traceable** : Each of the requirements should be clear and refer to the future development.

❖ **Examples of bad SRS (Characteristics of bad SRS)**

- **Over specification** : When in SRS, we try to address the things '*how*' the system will do this task. (As SRS focuses on 'What' not 'How')
- **Forward references** : Sometimes an SRS refer aspects that are to be referred later, due to that readability of an SRS is reduced.
- **Wishful thinking** : Sometimes in SRS, we try to concern the aspects which would be difficult to impalement.

The SRS documents that contain incompleteness, ambiguity and contradictions are considered as bad SRS documents.

3.2.2 Types of requirements in SRS

- An SRS should clearly document the following three things:
  - Functional requirements
  - Non-functional requirements
  - Constraint of the system

❖ Functional requirements of the system

- Software requirements are the → description of services which software will provide to end users. Functional requirements define the functions of the software and its components.
- It forms the core of requirement documents.
- Functional requirements are those which are related to the technical functionality of the system.
- The functional requirements for a system describe the functionalities or services that the system is expected to provide. They provide how the system should react to particular inputs and how the system should behave in a particular situation.
- Key goal of finding functional requirement is → to capture the behaviour of software in terms of functions and technology.
- It is the list of actual services which a system will provide. It is also the list of services which a user wants from the system.
- A function is described as a set of inputs, process and a set of outputs.
- Functional requirements may be calculations, data manipulations and processing, technical details or other specific functionalities that define what a system is supposed to do.
- Functional requirements of the system are captured in use cases.

➥ How to identify functional requirements ?

We can identify functional requirements from following factors:

- From informal problem description or from conceptual understanding of the system.
- From user perspective.
- Find out higher level function requirements.

➥ How to document the functional requirements ?

- Functional requirements are specified by different scenarios.
- Specify the input data domain, processing and output data domain.
- Document the functionalities supported by the system.

➥ Example: operations at ATM

Functional requirements of ATM system (likely)

- ➲ Withdraw cash
- ➲ Deposit cash
- ➲ Check balance
- ➲ Link aadhaar card
- ➲ Print mini statement
- ➲ Change pin number
- ➲ Generate PIN
- ➲ Transfer money etc.

When you list SRS, all the functional requirements must start with a 'verb'.

(ઉપર જણાવ્યા મુજબ ATM સેન્ટર પર આપણે વિવિધ કાર્ય કરી શકીએ છીએ. જેમ કે - withdraw cash, check balance, print mini statement, deposit cash વગેરે તે કામ એ આપણા SRS માટે Functional Requirement થશે. જેને આપણે **R1, R2** ગણીશું. તે દરેક Requirement માટે **step by step** એક process હશે જેને આપણે **R1.1, R1.2** ગણીશું.

આમ, આપણી system માં જેટલા જેટલા main કામ થતાં હોય અથવા તો આપણી system જેટલી main functionalities કે services પ્રોવાઈડ કરતી હોય તે તમામ functional requirements ને આપણે step by step ડીટેઈલમાં લખવી પડે.)

ખાસ નોંધ : (મિત્રો, SRS is very much important and it should be a key point for your project development in designing and analysis side, કારણ કે જેટલો તમારો **SRS** સાચો અને સ્પષ્ટ હશે તો ત્યાર બાદના બધા જ **diagrams** જેવા કે, **DFD, Use-case, Activity diagram** વગેરે બહુજ સરળતાથી બનાવી શકશો.)

In our example, here we are considering functional requirement as "withdraw cash".

R1: **withdraw cash**

Description : This function first determines the type of operation that user wants to do (i.e. withdraw), then determines the account type and amount that the user has for his transaction. It checks the balance to determine whether the requested amount is available in the account. If yes then it outputs the required cash, else it generates an error message.

R 1.1 : enter the card

Input : ATM card

Output : user prompted to enter PIN showing the display with various operations

R 1.2 : enter PIN

Input : valid PIN

Output : showing the display with various operations

R 1.3 : select operation type

Input : select proper option (withdraw amount option)

Output : user prompted to enter the account type

R 1.4 : select account type

Input : user option (for example : saving account or current account)

Output : user prompted to enter amount

R 1.5 : get required amount

Input : amount to be withdrawn

Output : requested cash and printed transaction

Now take one more requirement to document

R2 : **Print mini statement**

Description : this function first determines the type of operation that user wants to do (i.e. mini statement), then determines the account typethat theuser has for his transaction. Finally the mini statement (last 5 or 10 transactions) printed.

R 1.1 : enter the card

Input : ATM card

Output : user prompted to enter PINshowing the display with various operations

R 1.2 : enter the PIN

Input : valid PIN

Output : showing the display with various operations

R 1.3 : select operation type

Input : select proper option (ministatement option)

Output : user prompted to enter the account type

R 1.4 : select account type and generate statement

Input : user option (for example: saving account or current account)

Output : Mini statement in printed form

❖ Non-functional requirements of the system

- Non-functional requirements are the constraints or restrictions on the design of the system.
- Non-functional requirements are requirements that specify criteria that can be used to judge the operation of a system in particular conditions, rather than specific behaviours.
- Sometimes these requirements are also called quality attributes.
- The non-functional requirements describe the characteristics of the system that can't be expressed functionally
- Some important non-functional requirements are:
  - Portability
  - Reliability
  - Security
  - Security
  - Data integrity
  - Scalability
  - Maintainability
  - Usability
  - Performance
  - Manageability
  - Inter-operability
  - Availability

Functional requirements drive the application architecture of the system while non-functional requirements drive the technical architecture.

In one statement : Functional requirement is "what the system should do." And Non-functional requirement is "how the system should behave while performing."

❖ Difference between functional and non-functional requirements

Functional requirements	Non-functional requirements
→ These describe what the system should do.	→ These describe how the system should behave.
→ These describe features, functionality and usage of the system.	→ They describe various quality factors, attributes which affect the system's functionality.
→ Describe the actions with which the work is concerned.	→ Describe the experience of the user while doing the work.
→ Characterized by verbs.	→ Characterized by adjectives.
→ Ex: business requirements, SRS etc.	→ Ex: portability, quality, reliability, robustness, efficiency etc.

ASSIGNMENT

MCQs

1. SRS should describes .... the system should do not ....
   (a) how **(b) what** (c) where (d) why
2. Among the following which one is the full form of SRS.
   (a) Software Risk Specification **(b) System Requirement Specification**
   (c) System Risk Solution (d) Software Requirement Solution
3. When we go to the ATM to withdraw money, but ATM is out of service. Which non-functiona requirement violates here ?
   (a) scalability (b) reliability **(c) availability** (d) flexibility
4. Requirement gathering and requirement analysis & specification is collectively calle ________ .
   (a) requirement prototyping (b) customer satisfaction
   (c) life cycle model **(d) requirement engineering**
5. '24 x 7 availability of software product' is which kind of requirement ?
   (a) functional **(b) non-functional** (c) constraint (d) must need
6. Which of the following properties does not correspond to a good SRS.
   (a) ambiguous (b) complete (c) traceable (d) verifiable

True / False

1. Group discussion is one of the important techniques of requirement analysis.
   Answer : False
2. SRS is focuses on 'what' part of project development. **Answer : False**
3. SRS is also known as white box specification. **Answer : False**
4. 'wishful thinking' is one of the characteristics of a bad SRS. **Answer : True**

Short Questions

1. List various requirement gathering activities.
2. Define SRS. Explain contents of SRS.
3. List various SRS users.
4. Define functional requirements.
5. List out any four functional requirements for ATM banking system.
6. List out functional requirements for online movie ticket booking system for multiple system.

Descriptive questions

1. Explain characteristics of good SRS.
2. Explain requirement gathering activities.
3. Write a short note on requirement analysis.
4. Differentiate between functional and non-functional requirements.

***