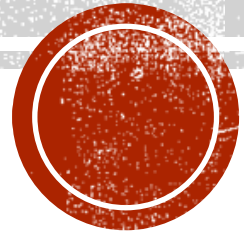
SOFTWARE REQUIREMENTS



SOFTWARE REQUIREMENTS

The software requirements are description of features and functionalities of the target system. Requirements convey the expectations of users from the software product.

The requirements can be obvious or hidden, known or unknown, expected or unexpected from client's point of view.

REQUIREMENT ENGINEERING

The process to gather the software requirements from client, analyze and document them is known as requirement engineering.

The goal of requirement engineering is to develop and maintain sophisticated and descriptive ‘System Requirements Specification’ document.

REQUIREMENT ENGINEERING PROCESS

It is a four step process, which includes –

- Feasibility Study
- Requirement Gathering
- Software Requirement Specification
- Software Requirement Validation

FEASIBILITY STUDY

- Feasibility study is focused towards goal of the organization.
- This study analyzes whether the software product can be practically materialized in terms of implementation, contribution of project to organization, cost constraints and as per values and objectives of the organization.
- It explores technical aspects of the project and product such as usability, maintainability, productivity and integration ability.
- The output of this phase should be a feasibility study report that should contain adequate comments and recommendations for management about whether or not the project should be undertaken.

REQUIREMENT GATHERING

- If the feasibility report is positive towards undertaking the project, next phase starts with gathering requirements from the user.
- Analysts and engineers communicate with the client and end-users to know their ideas on what the software should provide and which features they want the software to include.

SOFTWARE REQUIREMENT SPECIFICATION

- SRS is a document created by system analyst after the requirements are collected from various stakeholders.
- SRS defines how the intended software will interact with hardware, external interfaces, speed of operation, response time of system, portability of software across various platforms, maintainability, speed of recovery after crashing, Security, Quality, Limitations etc.
- The requirements received from client are written in natural language. It is the responsibility of system analyst to document the requirements in technical language so that they can be comprehended and useful by the software development team.

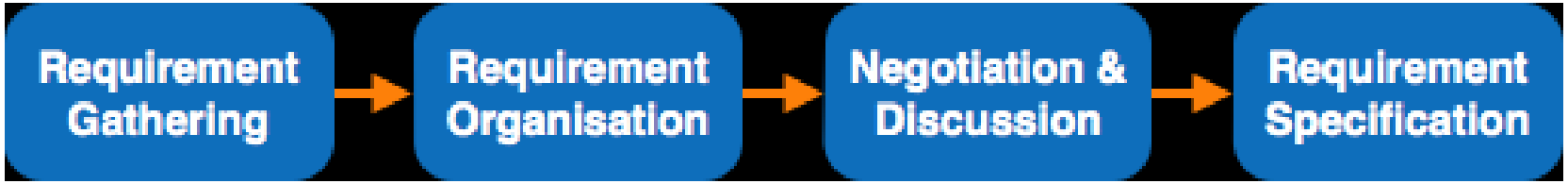
SOFTWARE REQUIREMENT VALIDATION

After requirement specifications are developed, the requirements mentioned in this document are validated. User might ask for illegal, impractical solution or experts may interpret the requirements incorrectly. This results in huge increase in cost if not nipped in the bud. Requirements can be checked against following conditions -

- If they can be practically implemented
- If they are valid and as per functionality and domain of software
- If there are any ambiguities
- If they are complete
- If they can be demonstrated

REQUIREMENT ELICITATION PROCESS

Requirement elicitation process can be depicted using the folloiwng diagram



- Requirements gathering - The developers discuss with the client and end users and know their expectations from the software.

- Organizing Requirements - The developers prioritize and arrange the requirements in order of importance, urgency and convenience.

- Negotiation & discussion - If requirements are ambiguous or there are some conflicts in requirements of various stakeholders, if they are, it is then negotiated and discussed with stakeholders. Requirements may then be prioritized and reasonably compromised.

- The requirements come from various stakeholders. To remove the ambiguity and conflicts, they are discussed for clarity and correctness. Unrealistic requirements are compromised reasonably.

- Documentation - All formal & informal, functional and non-functional requirements are documented and made available for next phase processing.

REQUIREMENT ELICITATION TECHNIQUES

Requirements Elicitation is the process to find out the requirements for an intended software system by communicating with client, end users, system users and others who have a stake in the software system development.

There are various ways to discover requirements

Interviews

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

REQUIREMENT ELICITATION TECHNIQUES

Questionnaires

A document with pre-defined set of objective questions and respective options is handed over to all stakeholders to answer, which are collected and compiled.

A shortcoming of this technique is, if an option for some issue is not mentioned in the questionnaire, the issue might be left unattended.

Task analysis

Team of engineers and developers may analyze the operation for which the new system is required. If the client already has some software to perform certain operation, it is studied and requirements of proposed system are collected.

Domain Analysis

Every software falls into some domain category. The expert people in the domain can be a great help to analyze general and specific requirements.

Brainstorming

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

REQUIREMENT ELICITATION TECHNIQUES

Surveys

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

Prototyping

Prototyping is building user interface without adding detail functionality for user to interpret the features of intended software product. It helps giving better idea of requirements. If there is no software installed at client's end for developer's reference and the client is not aware of its own requirements, the developer creates a prototype based on initially mentioned requirements. The prototype is shown to the client and the feedback is noted. The client feedback serves as an input for requirement gathering.

Observation

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

SOFTWARE REQUIREMENTS CHARACTERISTICS

Gathering software requirements is the foundation of the entire software development project. Hence they must be clear, correct and well-defined.

A complete Software Requirement Specifications must be:

- Clear
- Correct
- Consistent
- Coherent (logically connected)
- Comprehensible (capable of being understood)
- Modifiable
- Verifiable
- Prioritized
- Unambiguous
- Traceable
- Credible source

SOFTWARE REQUIREMENTS

Broadly software requirements should be categorized in two categories:

Functional Requirements

Requirements, which are related to functional aspect of software fall into this category.

They define functions and functionality within and from the software system.

Examples -

- Search option given to user to search from various invoices.
- User should be able to mail any report to management.
- Users can be divided into groups and groups can be given separate rights.
- Should comply business rules and administrative functions.
- Software is developed keeping downward compatibility intact.

SOFTWARE REQUIREMENTS

Non-Functional Requirements

Requirements, which are not related to functional aspect of software, fall into this category. They are implicit or expected characteristics of software, which users make assumption of.

Non-functional requirements include -

- Security
- Logging
- Storage
- Configuration
- Performance
- Cost
- Interoperability
- Flexibility
- Disaster recovery
- Accessibility

SOFTWARE REQUIREMENTS

Requirements are categorized logically as

- Must Have : Software cannot be said operational without them.
- Should have : Enhancing the functionality of software.
- Could have : Software can still properly function with these requirements.
- Wish list : These requirements do not map to any objectives of software.

While developing software, 'Must have' must be implemented, 'Should have' is a matter of debate with stakeholders and negation, whereas 'could have' and 'wish list' can be kept for software updates.

USER INTERFACE REQUIREMENTS

UI is an important part of any software or hardware or hybrid system. A software is widely accepted if it is -

- easy to operate
- quick in response
- effectively handling operational errors
- providing simple yet consistent user interface

User acceptance majorly depends upon how user can use the software. UI is the only way for users to perceive the system.

A well performing software system must also be equipped with attractive, clear, consistent and responsive user interface. Otherwise the functionalities of software system can not be used in convenient way. A system is said be good if it provides means to use it efficiently.

USER INTERFACE REQUIREMENTS

User interface requirements are briefly mentioned below -

- Content presentation
- Easy Navigation
- Simple interface
- Responsive
- Consistent UI elements
- Feedback mechanism
- Default settings
- Purposeful layout
- Strategical use of color and texture.
- Provide help information
- User centric approach
- Group based view settings.

SOFTWARE SYSTEM ANALYST

System analyst in an IT organization is a person, who analyzes the requirement of proposed system and ensures that requirements are conceived and documented properly & correctly. Role of an analyst starts during Software Analysis Phase of SDLC. It is the responsibility of analyst to make sure that the developed software meets the requirements of the client.

System Analysts have the following responsibilities:

- Analyzing and understanding requirements of intended software
- Understanding how the project will contribute in the organization objectives
- Identify sources of requirement
- Validation of requirement
- Develop and implement requirement management plan
- Documentation of business, technical, process and product requirements
- Coordination with clients to prioritize requirements and remove and ambiguity
- Finalizing acceptance criteria with client and other stakeholders

SOFTWARE METRICS AND MEASURES

- Software Measures can be understood as a process of quantifying and symbolizing various attributes and aspects of software.
- Software Metrics provide measures for various aspects of software process and software product.
- Software measures are fundamental requirement of software engineering. They not only help to control the software development process but also aid to keep quality of ultimate product excellent.
- According to Tom DeMarco, a (Software Engineer), "You cannot control what you cannot measure." By his saying, it is very clear how important software measures are.

SOFTWARE METRICS AND MEASURES

software metrics:

- Size Metrics - LOC (Lines of Code), mostly calculated in thousands of delivered source code lines, denoted as KLOC.

- Function Point Count is measure of the functionality provided by the software. Function Point count defines the size of functional aspect of software.

- Complexity Metrics - McCabe's Cyclomatic complexity quantifies the upper bound of the number of independent paths in a program, which is perceived as complexity of the program or its modules. It is represented in terms of graph theory concepts by using control flow graph.
- Quality Metrics - Defects, their types and causes, consequence, intensity of severity and their implications define the quality of product.

- The number of defects found in development process and number of defects reported by the client after the product is installed or delivered at client-end, define quality of product.

- Process Metrics - In various phases of SDLC, the methods and tools used, the company standards and the performance of development are software process metrics.
- Resource Metrics - Effort, time and various resources used, represents metrics for resource measurement.

SOFTWARE REQUIREMENT SPECIFICATION (SRS) FORMAT

1. Introduction
    - (i) Purpose of this document
    - (ii) Scope of this document
    - (iii) Overview
2. General description
3. Functional Requirements
4. Interface Requirements
5. Performance Requirements
6. Design Constraints
7. Non-Functional Attributes
8. Preliminary Schedule and Budget
9. Appendices

SOFTWARE REQUIREMENT SPECIFICATION (SRS) FORMAT

- Software Requirement Specification (SRS) Format as name suggests, is complete specification and description of requirements of software that needs to be fulfilled for successful development of software system.
- These requirements can be functional as well as non-requirements depending upon type of requirement.
- The interaction between different customers and contractor is done because its necessary to fully understand needs of customers.
- Depending upon information gathered after interaction, SRS is developed which describes requirements of software that may include changes and modifications that is needed to be done to increase quality of product and to satisfy customer's demand.

SOFTWARE REQUIREMENT SPECIFICATION (SRS) FORMAT

Introduction :

(i) Purpose of this Document –

At first, main aim of why this document is necessary and what's purpose of document is explained and described.

(ii) Scope of this document –

In this, overall working and main objective of document and what value it will provide to customer is described and explained. It also includes a description of development cost and time required.

(iii) Overview –

In this, description of product is explained. It's simply summary or overall review of product.

SOFTWARE REQUIREMENT SPECIFICATION (SRS) FORMAT

General description :

In this, general functions of product which includes objective of user, a user characteristic, features, benefits, about why its importance is mentioned. It also describes features of user community.

Functional Requirements :

In this, possible outcome of software system which includes effects due to operation of program is fully explained. All functional requirements which may include calculations, data processing, etc. are placed in a ranked order.

Interface Requirements :

In this, software interfaces which mean how software program communicates with each other or users either in form of any language, code, or message are fully described and explained. Examples can be shared memory, data streams, etc

SOFTWARE REQUIREMENT SPECIFICATION (SRS) FORMAT

Performance Requirements :

In this, how a software system performs desired functions under specific condition is explained. It also explains required time, required memory, maximum error rate, etc.

Design Constraints :

In this, constraints which simply means limitation or restriction are specified and explained for design team. Examples may include use of a particular algorithm, hardware and software limitations, etc.

Non-Functional Attributes :

In this, non-functional attributes are explained that are required by software system for better performance. An example may include Security, Portability, Reliability, Reusability, Application compatibility, Data integrity, Scalability capacity, etc.

SOFTWARE REQUIREMENT SPECIFICATION (SRS) FORMAT

Preliminary Schedule and Budget :

In this, initial version and budget of project plan are explained which include overall time duration required and overall cost required for development of project.

Appendices :

In this, additional information like references from where information is gathered, definitions of some specific terms, acronyms, abbreviations, etc. are given and explained.

PRACTICES TO FOLLOW WHILE WRITING THE SRS FOR A PROJECT

A software requirements specification (SRS) is a description of a software system to be developed. In order to write a good SRS, some common practices have to be followed which are as follows :

Communication Practices or Principles :

- Listen to the speaker and concentrate on what is being said.
- Prepare before you meet, by searching, researching and understanding the problem.
- Someone should facilitate the meeting and should have an agenda.
- Face to Face communication is best but also have a document or presentation to focus on discussion.
- Strive for collaborations and decisions.
- Document all the decisions.

PRACTICES TO FOLLOW WHILE WRITING THE SRS FOR A PROJECT

Planning Practices or Principles :

- Understand the scope of the project.
- Involve the customer in the planning activity.
- Recognize that planning is iterative and things will change.
- Estimation should be done based on only what you know.
- Consider the risk and it should be defined in the plan.
- Be realistic on how much can be done on each day.
- Define how you ensure quality can be achieved.

PRACTICES TO FOLLOW WHILE WRITING THE SRS FOR A PROJECT

Construction Deployment Practices or Principles :

- Understand the problem you are trying to solve.
- Understand basic design principles and concepts.
- Pick a programming language that meets the need of the software to be built.
- Create a set of unit test cases that will be applied once the component you code is completed.
- Constraint your algorithm by structured programming practices.
- Understand the software architecture and create interfaces that are compatible with it.
- Keep conditional logic as simple as possible.
- Write a code that is self documenting.