Software Requirements

What is a requirement?

- A software requirement is a condition or capability needed by a user to solve a problem or achieve an objective and that must be met or possessed by a system or system component to satisfy a contract, standard, specification, or other formally imposed document.

What is a requirement?

- ✧It may range from a high-level abstract statement of a service or of a system constraint to a detailed mathematical functional specification.
- ✧This is inevitable as requirements may serve a dual function
    - ▪May be the basis for a bid for a contract - therefore must be open to interpretation;
    - ▪May be the basis for the contract itself - therefore must be defined in detail;

Objectives

- To introduce the concepts of user and system requirements.
- To describe functional and non-functional requirements.
- To explain how software requirements may be organized in a requirements document.

Requirements engineering

- Requirements engineering is a systematic way of developing requirements through an iterative process of analyzing a problem, documenting the resulting observations, and checking the accuracy of the understanding gained.
- Requirements engineering is comprised of two major tasks: analysis and modeling.
- The requirements themselves are the descriptions of the ***system services*** and *constraints* that are generated during the requirements engineering process.

Requirements engineering

- ✧The process of establishing the services that the customer requires from a system and the constraints under which it operates and is developed.

Software requirements for the Ola app (cab booking system)

- User Management
  - FR1: User shall be able to register/login using phone number, email, or social accounts.
  - FR2: User shall be able to manage profile (name, photo, payment options, etc.).
- Ride Booking
  - FR3: User shall be able to book a ride by entering pickup and drop location.
  - FR4: System shall display available cab options (Mini, Sedan, Auto, Bike, etc.) with estimated fare and arrival time.
  - FR5: User shall be able to schedule rides for future.
- Ride Tracking & Notifications
  - FR9: The app shall show live GPS tracking of driver location to user.
  - FR10: User and driver shall receive real-time notifications (ride confirmation, arrival, cancellation, etc.).

Software requirements for the Ola app (cab booking system) Contd..

- Payment & Wallet
  - FR11: User shall be able to pay via cash, UPI, credit/debit cards, Ola Money.
  - FR12: System shall generate digital invoices after ride completion.
- Ratings & Feedback
  - FR13: User shall be able to rate driver and ride experience.
  - FR14: Driver shall also be able to rate passengers.
- Safety Features
  - FR15: SOS button shall allow passengers to share ride details with emergency contacts.
  - FR16: Ride-sharing link shall be provided for tracking by friends/family.

User and System Requirements

- High-level needs and expectations of the end users (what the system should do, from the user’s perspective).

Example (for Ola App):

  - A user should be able to book a cab easily.
  - The system should provide fare estimation before booking.
  - A user should be able to track the driver in real-time.
  - The system should allow multiple payment options (UPI, card, cash).

System Requirements

- Detailed specifications that describe how the system will fulfill the user requirements.
- More technical and precise, often structured, sometimes using formal methods.
- They can be divided into:
  - Functional Requirements – Define what functions the system should perform.
  - Non-Functional Requirements (NFRs) – Define quality attributes like performance, security, reliability, usability, etc.
  - Focus: How the system will be designed and implemented.

Example (System requirement for Ola App)

- The system shall allow users to log in using mobile number + OTP authentication.
- The system shall store ride history in the database for at least 6 months.
- The system shall provide response time < 2 seconds for booking confirmation.
- The system shall support 50,000 concurrent users without performance degradation.
- User Requirements → What the user wants (expressed simply, business-focused).
- System Requirements →How the system will meet those needs (detailed, technical, for developers/designers).

User vs System Requirements (Example Ola)

Aspect	User requirements(what users want)	System Requirements (how system will provide it)
Booking a cab	User should be able to book a cab quickly	System shall provide a cab search and booking module that displays available drivers within 5 km and confirms booking within 2 seconds
Fare estimation	User should see the fare before confirming	System shall calculate estimated fare using distance (Google Maps API) + base fare + surge pricing rules, and display it in the app.
Ride tracking	User wants to track the driver's location in real time	System shall integrate with GPS services and update driver's location on the user's map every 5 seconds
Payment options	User wants multiple ways to pay (UPI, card, cash).	System shall integrate with payment gateways (UPI, Razorpay, PayPal, credit/debit card) and allow cash-on-delivery option.
Ride history	User wants to view past rides	System shall store ride history in the database for 6 monthsand provide an option to view/download it in the app

User vs System Requirements Contd…

- User requirements are high-level, simple, and user-focused.
- System requirements are detailed, technical, and implementation-focused

User & System Requirements Mapping

1. Booking a Cab

<u>User Requirement</u>: User should be able to book a cab quickly.

<u>System Requirements</u>:

- System shall provide a cab search and booking module.
- System shall display available drivers within 5 km of user's location.
- System shall confirm booking within 2 seconds.

User & System Requirements Mapping

2. Fare Estimation

- User Requirement: User should see the fare before confirming the ride.

System Requirements:

- System shall calculate estimated fare using distance (Google Maps API).
- System shall include base fare + surge pricing + taxes in fare calculation
- System shall display fare estimation in the app UI before ride confirmation

System requirements describe what <u>a system should do</u> (functional) and <u>how well it should perform</u> (non-functional).
They act as a bridge between user needs and the system design.

Functional and non-functional requirements

- Functional requirements
  - Statements of services the system should provide, how the system should react to particular inputs and how the system should behave in particular situations.
  - May state what the system should not do.
- Non-functional requirements
  - Constraints on the services or functions offered by the system such as timing constraints, constraints on the development process, standards, etc.
  - Often apply to the system as a whole rather than individual features or services.
- Domain requirements
  - Constraints on the system from the domain of operation

Functional Requirements:

"A requirement specifies a function that a system or component must be able to perform.“

- Specify **specific behavior or functions.**

for example: "*Display the heart rate, blood pressure and temperature of a patient connected to the patient monitor*.“

Functional Requirements:

Functional requirements are:

- Business Rules
- Transaction corrections,
- Administrative functions
- Authentication
- Authorization
- Audit Tracking
- External Interfaces

Functional Requirements should include:

Descriptions of data to be entered into the system
Descriptions of operations performed by each screen
Descriptions of work-flows performed by the system
Descriptions of system reports or other outputs
Who can enter the data into the system
How the system meets applicable regulatory requirements

Examples of Functional Requirements

Interface requirements

Field 1 accepts numeric data entry.
Field 2 only accepts dates before the current date.
Screen 1 can print on-screen data to the printer.

Business Requirements

Data must be entered before a request can be approved.
Clicking the Approve button moves the request to the Approval Work flow
All personnel using the system will be trained according to internal SOP AA-101.

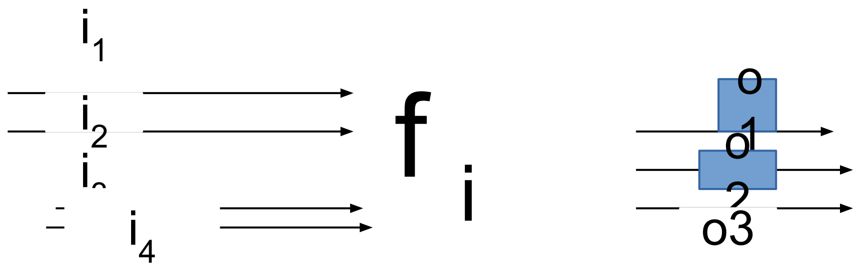
Regulatory/Compliance Requirements

The database will have a functional audit trail.
The system will limit access to authorized users.
The spreadsheet can secure data with electronic signatures.

Functional requirements:-

- The functional requirements part discusses the functionalities required from the system.
- The system is considered to perform a set of high-level functions $\{f_i\}$.
- The functional view of the system is shown in fig. 1.

- Each function f_i of the system can be considered as a transformation of a set of input data (ii) to the corresponding set of output data (o_i).
- The user can get some meaningful piece of work done using a high-level function.



Example:-

Consider the case of the library system, where -
F1: Search Book function
Input: an author's name
Output: details of the author's books and the location of these books in the library

Withdraw Cash from ATM

R1: withdraw cash

Description: The withdraw cash function first determines the <u>type of account</u> that the user has and the account number from which the user wishes to withdraw cash. It checks the <u>balance</u> to determine whether the requested amount is available in the account. If enough balance is available, it outputs the required cash, otherwise it generates an error message.

<u>R1.1 select withdraw amount option</u>

Input: “withdraw amount” option, Output: user prompted to enter the account type

<u>R1.2: select account type</u>

Input: user option, Output: prompt to enter amount

<u>R1.3: get required amount</u>

Input: amount to be withdrawn in integer values greater than 100 and less than10,000 in multiples of 100. Output: The requested cash and printed transaction statement.

Functional requirements for the Mental Health Care-Patient Management System

- ✧A user shall be able to search the appointments lists for all clinics.
- ✧The system shall generate each day, for each clinic, a list of patients who are expected to attend appointments that day.
- ✧Each staff member using the system shall be uniquely identified by his or her 8-digit employee number.

Non-functional requirements (NFR)

lNon-functional requirements define the overall qualities or
lattributes of the resulting system. As per IEEE, *it describes not what the software will do, but how the software will do it*.

lFor EXAMPLE, the software performance requirement, design constraints and external Interface requirements etc.

lNon-functional requirements place restrictions on the product
lbeing developed, the development process, and specify external
lconstraints that the product must meet.

lExamples of NFR include safety, security, usability, reliability and
lperformance requirements.

lProject management issues (costs, time, schedule) are often
lconsidered as non-functional requirements

Functional vs. Non-functional requirements

- Some properties of a system may be expressed either as a
- functional or non-functional property.
- Example. The system shall ensure that data is protected from
- unauthorised access.
- Conventionally a non-functional requirement (security) because it does not specify specific system functionality Expressed as functional requirement.
- The system shall include a user authorisation procedure where users must identify themselves using a login name and password. Only users who are authorised in this way may access the system data.
- Non-functional requirements may result in new functional requirements statement

Exercise 1:

A university wants to develop an **Online Examination Management System (OEMS)** to conduct mid-semester and end-semester exams digitally. The system will allow faculty to create question papers, students to appear for exams online, and administrators to monitor and generate results. The system should support both objective and descriptive type questions.

Faculty members should be able to upload question banks, set time limits, and schedule exams. Students must be able to log in securely, view their scheduled exams, and submit answers within the given time. After submission, the system should automatically evaluate objective-type questions and store descriptive answers for manual grading by faculty.

Once evaluation is completed, students can view their results and feedback online. The system should also maintain exam records, results history, and logs of user activity for auditing purposes.

Q1. Identify the user and System Requirement for each user.

Q2. Identify the functional and non-functional requirements of the system.

Exercise 2:

A city hospital plans to implement a **Hospital Management System (HMS)** to digitalize and streamline its operations. The system will manage patient registration, doctor appointments, billing, and medical records. It should allow doctors, patients, and administrative staff to perform their specific tasks through secure login access.

When a patient visits the hospital, they can register online or at the reception. The system should allow the staff to assign doctors, manage appointment schedules, and store patient health history. Doctors can view patients' previous medical records, prescribe medicines, and generate digital prescriptions. The billing department should be able to generate and print invoices automatically after consultation or discharge.

The system must also maintain an inventory of medicines and medical equipment, alerting staff when stocks run low. Reports on patient visits, revenue, and inventory usage should be easily generated for management review.

Q1. Identify the user and System Requirement for each user.

Q2. Identify the functional and non-functional requirements of the system.

Requirements imprecision

- Problems arise when requirements are not precisely stated.
- Ambiguous requirements may be interpreted in different ways by developers and users.
- Consider the term ‘search’ in requirement 1
    - User intention – search for a patient name across all appointments in all clinics;
    - Developer interpretation – search for a patient name in an individual clinic. User chooses clinic then search.

Requirements completeness and consistency

- In principle, requirements should be both complete and consistent.
- **Complete**
    - They should include descriptions of all facilities required.
- **Consistent**
    - There should be no conflicts or contradictions in the descriptions of the system facilities.
- In practice, it is impossible to produce a complete and consistent requirements document.

Non-functional requirements implementation

- ✧Non-functional requirements may affect the overall architecture of a system rather than the individual components.
  - ▪For example, to ensure that performance requirements are met, you may have to organize the system to minimize communications between components.
- ✧A single non-functional requirement, such as a security requirement, may generate a number of related functional requirements that define system services that are required.
  - ▪It may also generate requirements that restrict existing requirements.

Non-functional requirements

- These define system properties and constraints e.g. reliability, response time and storage requirements. Constraints are I/O device capability, system representations, etc.
- **Process requirements** may also be specified mandating a particular IDE, programming language or development method.
- **Non-functional requirements may be more critical than functional requirements.** If these are not met, the system may be useless.

Non-functional classifications

- Product requirements
  - Requirements which specify that the delivered product must behave in a particular way e.g. execution speed, reliability, etc.
- Organisational requirements
  - Requirements which are a consequence of organisational policies and procedures e.g. process standards used, implementation requirements, etc.
- External requirements
  - Requirements which arise from factors which are external to the system and its development process e.g. interoperability requirements, legislative requirements, etc.

Types of nonfunctional requirement

<u>**Product requirements**</u>

1. Usability requirements
2. Reliability requirement
3. Safety requirements
4. Efficiency requirements
5. Performance requirements
6. Capacity requirements

Organisational requirements

1. Delivery requirements
2. implementation requirements
3. standards requirements

External requirements

1. Legal constraints
2. Economic constraints
3. Interoperability requirements

Examples of nonfunctional requirements in the MHC-PMS (Mental Health Care-Patient Management System)

Product requirement
The MHC-PMS shall be available to all clinics during normal working hours (Mon–Fri, 0830–17.30). Downtime within normal working hours shall not exceed five seconds in any one day.

Organizational requirement
Users of the MHC-PMS system shall authenticate themselves using their health authority identity card.

External requirement
The system shall implement patient privacy provisions as set out in HStan-03-2006-priv.

Requirements for critical systems

There are three principal types of critical system:

1. **Business critical systems :** Failure leads to significant economic damage.
2. **Mission critical systems :** Failure leads to the abortion of a mission.
3. **Safety critical systems:** Failure endangers human life.

Requirements for critical systems

- The principal non-functional constraints which are relevant to critical systems:
  - **Reliability**
  - **Performance**
  - **Security**
  - **Safety**
  - **Usability**

Reliability

- Reliability is the ability of a system to perform its required functions under stated conditions for a specific period of time.
- Can be considered under two separate headings:
  - Availability - is the system available for service when requested by end-users.
  - Failure rate - how often does the system fail to deliver the service as expected by end-users.

Performance

- Performance requirements concern the speed of operation of a system.
- Types of performance requirements:
  - Response requirements (how quickly the system reacts to a user input)
  - Throughput requirements (how much the system can accomplish within a specified amount of time)
  - Availability requirements (is the system available for service when requested by end-users)

Product Requirements Examples

- The System service X shall have an availability of 999/1000 or 99%. This is a <u>reliability requirement</u> which means that out of every 1000 requests for this service, 999 must be satisfied.
- System Y shall process a minimum of 8 transactions per second. This is a <u>performance</u> requirement.

Security

- Security requirements are included in a system to ensure:
  - Unauthorised access to the system and its data is not allowed
  - Ensure the integrity of the system from accidental or malicious damage.

Examples of security requirements are:

- The access permissions for system data may only be changed by the system’s data administrator.
- All system data must be backed up every 24 hours and the backup copies stored in a secure location which is not in the same building as the system.
- All external communications between the system’s data server and clients must be encrypted

Usability

- Usability is the ease with which a user can learn to operate, prepare inputs for, and interpret outputs of system or component.
- Usability requirements include:
  - Well-structured user manuals
  - Informative error messages
  - Help facilities
  - Well-formed graphical user interfaces

Safety

- Safety requirements are the ‘shall not’ requirements which exclude unsafe situations from the possible solution space of the system

Examples of safety requirements

- The violated system shall not permit any further operation unless the operator guard is in place.
- The system shall not operate if the external temperature is below 4 degrees Celsius.
- The system should not longer operate in case of fire (e.g. an elevator)

Supportability

- Supportability requirements are concerned with the ease of changes to the system after deployment.

Classes:

- Adaptability: The ability to change the system to deal with additional application domain concepts (adaptivity = autonomous adaptation)
- Maintainability: The ability to change the system to deal with new technology or to fix defects.
- Portability: The ease with which a system or component can be transferred from one environment to another.

Portability

- Portability is the *degree to which software running on one platform can easily be converted to run on another*.
- Portability is hard to quantify, because it is hard to predict on what other platforms will the software be required to run.
- Portability for a given software system can be enhanced by using languages, operating systems and tools that are universally available and standardized, such as FORTRAN, COBOL or C (for languages), or such as Unix, Windows or OS/2 (operating systems).
- Portability requirements should be given priority for systems that may have to run on different platforms in the near future.

Relationships between user needs, concerns and NFRs

Function	1.Ease of use 2. Unauthorised access 3.Likelihood of failure	1.Usability 2. Security 3.Reliability
Performance	1. Resource utilization 2.Performance verification 3.Ease of interfacing	1.Efficiency 2.Verifiability 3.Interoperability
Change	1.Ease of repair 2.Ease of change 3.Ease of transport 4.Ease of expanding or upgrading capacity or performance	1.Maintainability 2. Flexibility 3. Portability 4.Expandability

Metrics for specifying non-functional requirements

Property	Measure
Speed	Processed transactions/second User/event response time Screen refresh time
Size	Mbytes Number of ROM chips
Ease of use	Training time Number of help frames
Reliability	Mean time to failure Probability of unavailability Rate of failure occurrence Availability
Robustness	Time to restart after failure Percentage of events causing failure Probability of data corruption on failure
Portability	Percentage of target dependent statements Number of target systems

Examples of measurable metrics for Non-functional requirements

Performance	1.Processed transactions per second 2.Response time to user input
Reliability	1.MTTF, MTTR, MTBF 2.Rate of occurrence of failure
Availability	1.Probability of failure on demand
Size	1.Kbytes, Mbytes
Usability	1.Time taken to learn the software 2.Number of errors made by user
Robustness	1.Time to restart the system after failure

Nonfunctional Requirements:

Trigger Questions

- Performance characteristics
  - Are there any speed, throughput, or response time constraints on the system?
  - Are there size or capacity constraints on the data to be processed by the system?

Nonfunctional Requirements: Trigger Questions

- Quality issues:
  - What are the requirements for reliability?
  - What is the maximum time for restarting the system after a failure?
  - What is the acceptable system downtime per 24-hour period?

Nonfunctional Requirements: Trigger Questions

- Resources and Management Issues:
  - How often will the system be backed up?
  - Who will be responsible for the back up?
  - Who is responsible for system installation?
  - Who will be responsible for system maintenance?

QUESTION ANSWER

Identify the requirement type:

1. The system shall be developed for PC and Macintosh platforms.
2. The system must encrypt all external communications using the RSA algorithm.

Ans

- Both are NFRs (There are product requirements which relate to the source code of the system)
- Examples: The system shall be developed for PC and Macintosh platforms. This is a <u>portability requirement</u> which affects the way in which the system may be designed.
- The system must encrypt all external communications using the RSA algorithm. This is a <u>security requirement</u> which specifies that a specific algorithm must be used in the product.

Software efficiency

- It refers to the level of use of scarce computational resources, such as CPU cycles, memory, disk space, buffers and communications channels.
- Efficiency can be characterized as follows:
  - **Capacity** - maximum number of users/terminals/ transactions/... the system can handle without performance degradation
  - **Degradation of service** - what happens when a system with capacity X widgets per time-unit receives X+1 widgets? We don't want the system to simply crash! Rather, we may want to stipulate that the system should handle the load, perhaps with degraded performance.

QUESTION ANSWER

Identify the requirement type:

- "...The system shall handle up to and including 20 simultaneous terminals and users performing any activities without degradation of service below that defined in section X.Y.Z;
- other systems may make short requests, at a maximum rate of 50/hr and long requests at the rate of 1/hr...”
- "…For more than 20 simultaneous terminals, the system will continue to operate with degraded services below what is defined in section X.Y.Z..."

Process Requirements

- Process requirements are constraints placed upon the development process of the system. (**development time limitations, resource availability, methodological standards etc**.)
- Process requirements include:
  - Requirements on development standards and methods which must be followed.
  - CASE tools which should be used.
  - The management reports which must be provided

Examples of process requirements

- The development process to be used must be explicitly defined and must be conformant with ISO 9000 standards.
- The system must be developed using the XYZ suite of CASE tools.
- Management reports setting out the effort expended on each identified system component must be produced every two weeks.
- A disaster recovery plan for the system development must be specified.

External requirements

- May be placed on both the product and the process.
- Derived from the environment in which the system is developed.

External requirements are based on:

- –application domain information
- – organisational considerations
- –the need for the system to work with other systems □
- –health and safety or data protection regulations

- Legal requirements: Concerned with licensing, regulation, and certification issue

Examples of external requirements

- Medical data system: The organization's data protection officer must certify that all data is maintained according to data protection legislation before the system is put into operation.
- A software is regulated under the GNU General Public License make sure the software is free for all its users, that is, everybody can share and change the software □ No warranty, no patents

Goals and requirements

- ✧Non-functional requirements may be very difficult to state precisely and imprecise requirements may be difficult to verify.
- ✧Goal
  - ▪A general intention of the user such as ease of use.
- ✧Verifiable non-functional requirement
  - ▪A statement using some measure that can be objectively tested.
- ✧Goals are helpful to developers as they convey the intentions of the system users.

Usability requirements

- ✧The system should be easy to use by medical staff and should be organized in such a way that user errors are minimized. (Goal)
- ✧Medical staff shall be able to use all the system functions after four hours of training. After this training, the average number of errors made by experienced users shall not exceed two per hour of system use. (Testable non-functional requirement)

Domain requirements

- The system's operational domain imposes requirements on the system.
  - For example, a train control system has to take into account the braking characteristics in different weather conditions.
- Domain requirements be new functional requirements, constraints on existing requirements or define specific computations.
- If domain requirements are not satisfied, the system may be unworkable.

Example - Train protection system

- ✧This is a domain requirement for a train protection system:
- ✧The deceleration of the train shall be computed as:
  - ▪Dtrain = Dcontrol + Dgradient

  - ▪where Dgradient is 9.81ms2 * compensated gradient/alpha and where the values of 9.81ms2 /alpha are known for different types of train.
- ✧It is difficult for a non-specialist to understand the implications of this and how it interacts with other requirements.

Software Requirements Specification For <Project> with IEEE Standard

1. Introduction

1.1 Purpose: describe the purpose of this document, not the purpose of the software being developed.

1.2 Scope: describe the scope of this document, not the scope of the software being developed.

1.3 Definitions, Acronyms, and Abbreviations.:

1.4 Overview

1.5 References

Hotel Management System Software Requirements Specifications

1 1. Introduction

The following subsections of the Software Requirements Specifications (SRS) document provide an overview of the entire SRS.

1.1 **Purpose**

The Software Requirements Specification (SRS) will provide a detailed description of the requirements for the Hotel Management System (HMS). This SRS will allow for a complete understanding of what is to be expected of the HMS to be constructed. The clear understanding of the HMS and its' functionality will allow for the correct software to be developed for the end user and will be used for the development of the future stages of the project. This SRS will provide the foundation for the project. From this SRS, the HMS can be designed, constructed, and finally tested.

1.2 Scope

The software product to be produced is a Hotel Management System which will automate the major hotel operations.

The first subsystem is a Reservation and Booking System to keep track of reservations and room availability.

The second subsystem is the Tracking and Selling Food System that charges the current room.

The third subsystem is a General Management Services and Automated Tasks System which generates reports to audit all hotel operations and allows modification of subsystem information.

These three subsystems’ functionality will be described in detail in section 2-Overall Description.

There are two end users for the HMS. The end users are the hotel staff (customer service representative) and hotel managers. Both user types can access the Reservation and Booking System and the Food Tracking and Selling System. The General Management System will be restricted to

1.3 Definitions, Acronyms, and Abbreviations.

1 <u>Definitions, Acronyms, and Abbreviations:</u>

SRS – Software Requirements Specification

HMS – Hotel Management System

Subjective satisfaction – The overall satisfaction of the system

End users – The people who will be actually using the system

1.4 Overview

The SRS is organized into two main sections. The first is The Overall Description and the second is the Specific Requirements.

The Overall Description will describe the requirements of the HMS from a general high level perspective.

The Specific Requirements section will describe in detail the requirements of the system.

Software Requirements Specification For <Project> with IEEE Standard

- **2. The Overall Description**
    - **2.1 Product Perspective**
        - **2.1.1 Hardware Interfaces**
        - **2.1.2 Software Interfaces**
    - **2.2 Product Functions**
    - **2.3 User Characteristics**
    - **2.4 Appropriation of requirements**

Software Requirements Specification
For <Project> with IEEE Standard

- **3. Specific Requirements**
  - **3.1. External Interfaces**
  - **3.2. Functional Requirements**
  - **3.2 Non-Functional Requirements**
- **4. Change Management Process**
- **5. Document Approval**

Properties of a good SRS document

- Concise.
- Structured.
- Black-box view.
- Conceptual integrity.
- Response to undesired events.
- Verifiable.