

Course: Software Requirements Engineering

Week 9: Specification & SRS Structure

Lecturer: Yimer Amedie (MSc.)

Addis Ababa Science and Technology University

May, 2026

Contents



- Introduction
- Requirements Specifications and SRS
- Principles of Specification
- Structure and Components of SRS
- Standards for Requirements Documentation

Figure 1. *The software requirements*

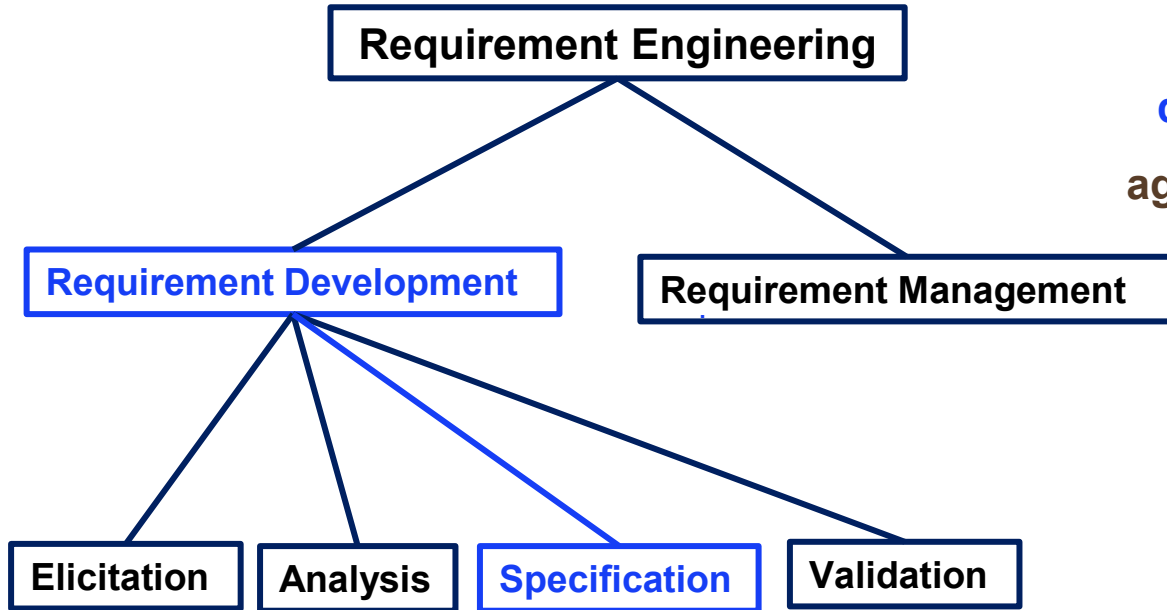
Note. Image generated using Sora by OpenAI (2026).

Learning Outcomes

After completing this lesson, you will be able to:

- Explain the purpose and importance SRS
- Identify the standard structure and components of an SRS document
- Apply IEEE/ISO standards in requirements specification & documentation
- Organize and document requirements using proper structure

Introduction



The result of **requirements development** is a documented agreement among stakeholders about the product to be built (Beatty, 2013)

Documented agreement → **SRS**

What is Specification?

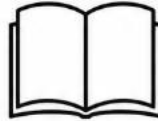
A precise, structured description of system requirements that defines what the system must achieve, without prescribing how it is implemented.



Not:

- design decisions (how to implement)
- coding details
- technology choices

Example



Design (not specification):

“The system shall use
React for the UI”




Specification:


“The system shall provide a user
interface for submitting and
tracking requests”

Requirement Specification (RS)

- ✓ **RS** is a detailed document that clearly describes what a system, software, or product should do and the constraints under which it must operate.

	SRS (Software Requirements Specification) — for software systems
---	--

	BRS (Business Requirements Specification) — focuses on business needs
---	---

	FRS (Functional Requirements Specification) — focuses on system functions
---	---

Overview of SRS



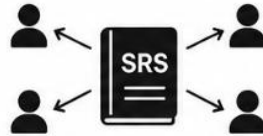
Formal document describing functional and non-functional requirements



Serve as a contract between stakeholders and developers.



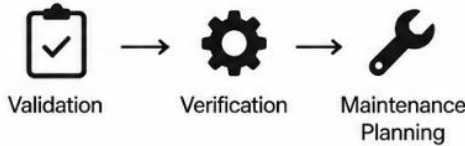
Single source of truth for all project participants



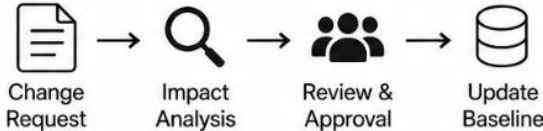
Provides a common understanding and agreement on what is to be built.



Used for validation, verification, and maintenance planning



Baseline for change control management



Distinction from business requirements and design specs



Why is it Important?



Reduces misunderstandings between client and development team



Provides basis for cost and schedule estimation



Enables early detection of conflicts or missing requirements



Supports maintenance and evolution of software



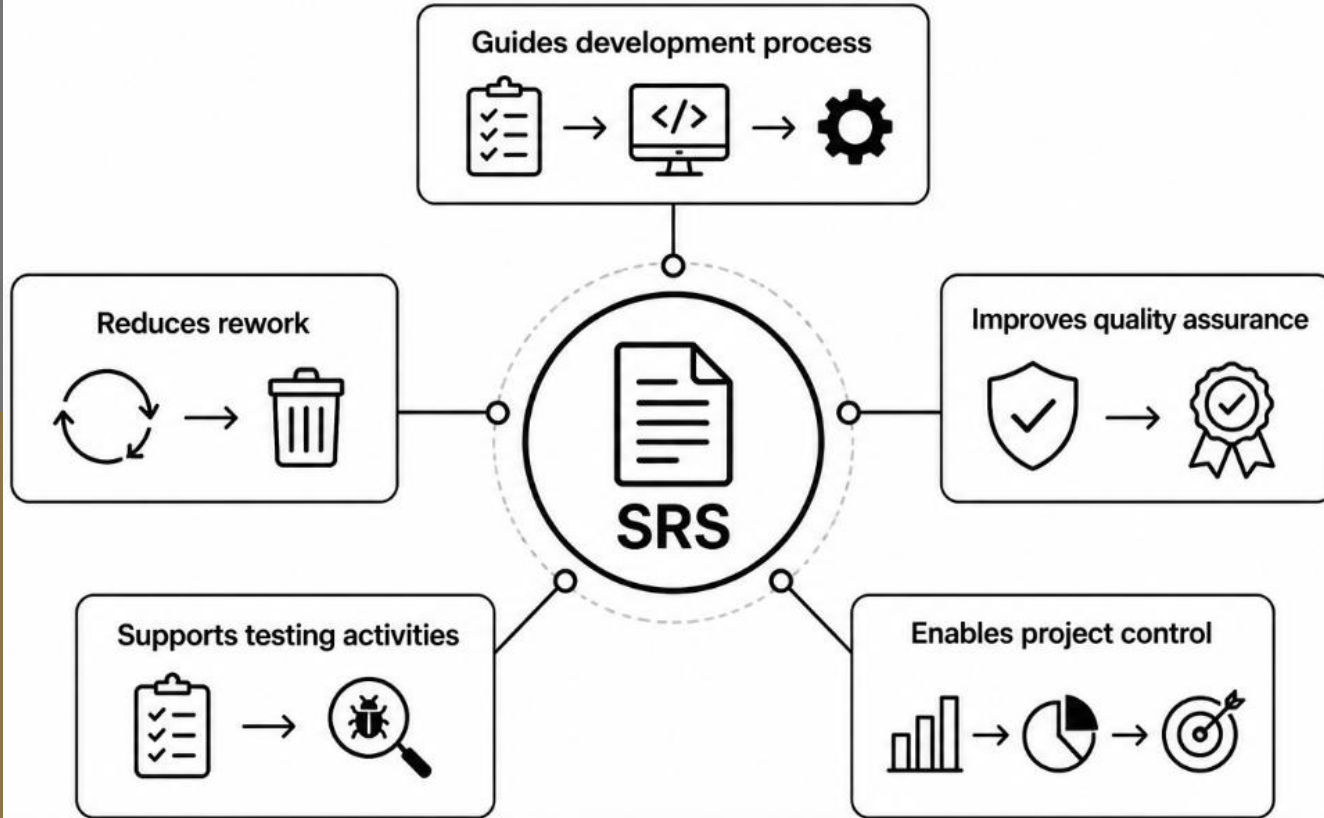
Required in safety-critical and regulated industries



Serves as a baseline for design, development, and testing

SRS focuses on **translating business needs into system requirements**

Importance of SRS in SDLC



Stakeholders of SRS

Numerous audiences rely on the SRS (Beatty, 2013).



Customers, the marketing department, and sales staff need to know what product they can expect to be delivered.



Project managers base their estimates of schedule, effort, and resources on the requirements.



Software development teams need to know what to build.



Testers use it to develop requirements-based tests, test plans, and test procedures.

Stakeholders of SRS ... cont'd



Maintenance and support staff use it to understand what each part of the product is supposed to do.



Documentation writers base user manuals and help screens on the SRS and the user interface design.



Training personnel use the SRS and user documentation to develop educational materials.



Legal staff ensures that the requirements comply with applicable laws and regulations.



Subcontractors base their work on—and can be legally held to—the specified requirements.

Principles of SRS



Separate correct requirements from incorrect ones



Ensure clarity and remove ambiguity in requirements



Develop a complete description of system behavior



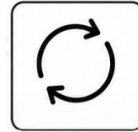
Maintain consistency throughout the specification



Make requirements verifiable



Rank requirements based on importance & stability



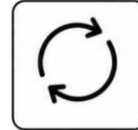
Structure the specification to be modifiable



Ensure traceability of requirements



Keep the specification independent of design decisions



Make the specification adaptable to change and evolution

Principles of SRS – Final Remark

A standard-compliant SRS should be:



Correct



Unambiguous



Complete



Consistent



Verifiable



Prioritized



Modifiable



Traceable

What is not Included in SRS

 The SRS SHOULD NOT contain



Design details



Construction /
coding specifics



Testing
procedures



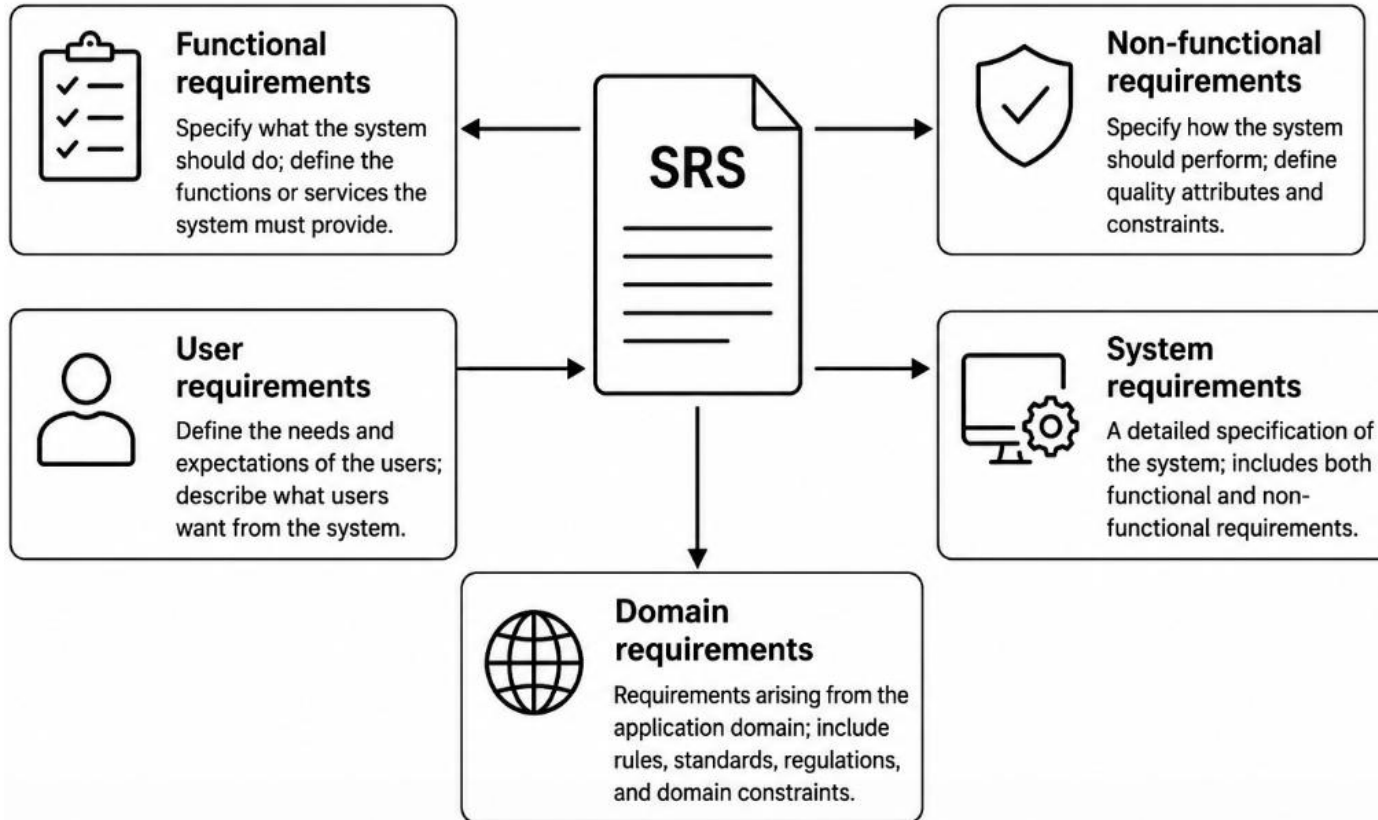
Project management
details



EXCEPTION

It should not contain design, construction, testing, or project management details other than **known design and implementation constraints**.

Types of Requirements in SRS



Role of Documentation



Provides
permanent record



Supports
communication



Enables
knowledge transfer



Assists
maintenance



Facilitates
audits and reviews

Maintaining SRS

Regular updates



The SRS is reviewed and updated at regular intervals to incorporate new information and changes.

Reflects changes



All changes in requirements, business rules, or stakeholder needs are accurately captured and reflected in the SRS.

Ensures accuracy



Updated SRS remains correct, consistent, and complete, ensuring reliable communication.

Supports evolution



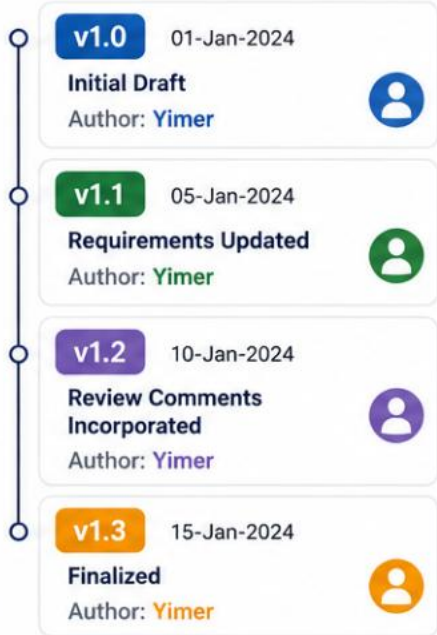
The SRS evolves along with the system and business environment to support future enhancements.

Continuous improvement



Version Control in SRS

VERSION HISTORY



Complete audit trail



VERSION CONTROL

Manages and tracks changes to the SRS document over time.



Single source of truth for all stakeholders

BENEFITS



Tracks Document Changes

Records every modification for full visibility and traceability.



Maintains History

Keeps all previous versions for reference and rollback.



Supports Collaboration

Enables multiple stakeholders to work together efficiently.



Prevents Conflicts

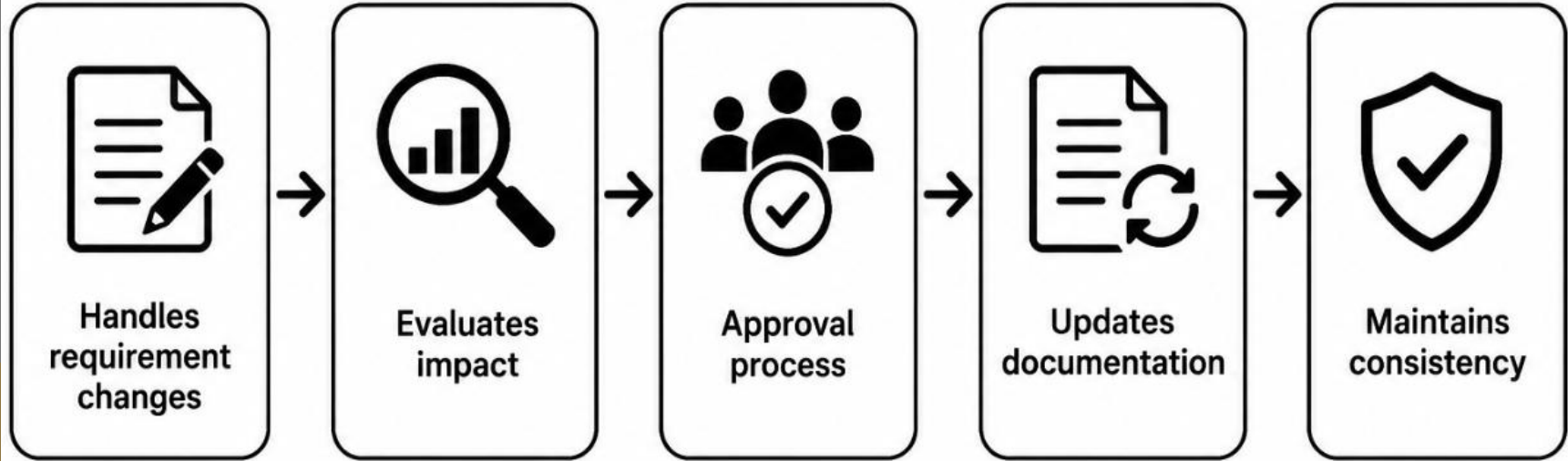
Helps identify changes early and avoid overlapping work.



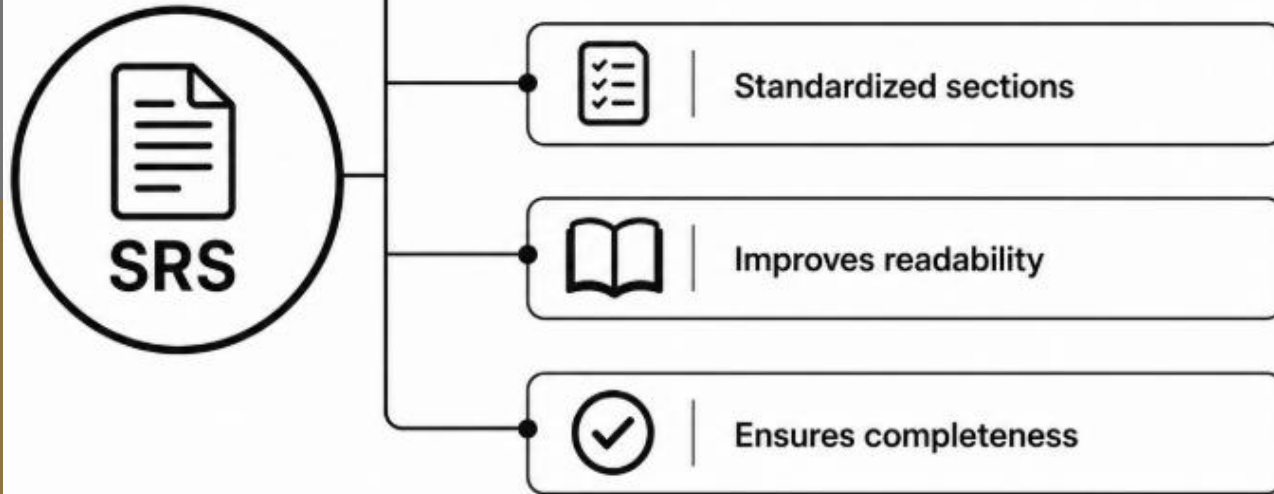
Ensures Consistency

Maintains uniformity and accuracy across versions.

Change Management in SRS



SRS Structure



Requirements Document Standard



Common Standards

1 IEEE 830 (Classic Standard)



- ✓ Widely used traditional SRS format
- ✓ Defines structure and qualities of good SRS

2 ISO/IEC/IEEE 29148 (Modern Standard)














- ✓ Updated and more comprehensive
- ✓ **Covers:**
 - Requirement engineering processes
 - Documentation standards
 - Lifecycle integration



Note: ISO/IEC/IEEE 29148 is the current industry-recommended standard as it supports the full requirements lifecycle.

IEEE Std 830 Vs. ISO/IEC/IEEE 29148

IEEE 830-1998 Historical Standard for SRS Structure	vs.	ISO/IEC/IEEE 29148:2011 (Revised 2018) Supersedes 830
 <p>IEEE 830-1998: historical standard for SRS structure</p>		 <p>ISO/IEC/IEEE 29148:2011 (revised 2018) supersedes 830</p>
 <p>Focused on software requirements specification (SRS) structure</p>		 <p>29148 integrates system and software requirements engineering</p>
 <p>Provides a structured outline for Software Requirements Specification</p>		 <p>29148 adds annexes for writing techniques and quality indicators</p>
 <p>Both standards emphasize</p>  <ul style="list-style-type: none">✓ Completeness✓ Consistency✓ Unambiguity✓ Verifiability		

Generic SRS Structure (Clause-by-Clause)



Section 1:
Introduction

Purpose, Scope
Definitions, References



Section 2:
Overall description

User characteristics, Constraints
Assumptions & dependencies, Product perspectives
Summary of product features



Section 3:
Specific requirements

FRs, NFRs
Interface requirements



Section 4:
Appendices and index

Appendices, Index

Generic SRS Structure ... cont'd



Section 3: Specific requirements

Functional requirements: what the system shall do

Non-functional requirements: quality attributes and constraints

Interface requirements: user, hardware, software, communication interfaces

Section 3 Subdivision – Organized by Features or Use Case Categories

3.1 Feature / Use Case Category 1

- Functional requirements
- Non-functional requirements
- Interface requirements
- Business rules (if any)

3.2 Feature / Use Case Category 2

- Functional requirements
- Non-functional requirements
- Interface requirements
- Business rules (if any)

3.3 Feature / Use Case Category 3

- Functional requirements
- Non-functional requirements
- Interface requirements
- Business rules (if any)

...

3.n Feature / Use Case Category N

- Functional requirements
- Non-functional requirements
- Interface requirements
- Business rules (if any)

Standard SRS Structure

A standard SRS document is typically structured as follows:

1.



Introduction

- Purpose of the document
- Scope of the system
- Definitions, acronyms, abbreviations
- References

2.



Overall Description

- Product perspective
- Product functions
- User classes and characteristics
- Operating environment
- Constraints
- Assumptions and dependencies

3.



System Features (Functional Requirements)

- Detailed description of system functionalities
- Organized by features/modules

4.



External Interface Requirements

- User interfaces (UI)
- Hardware interfaces
- Software interfaces
- Communication interfaces

5.



Non-Functional Requirements

- Performance
- Security
- Usability
- Reliability
- Scalability

6.



Other Requirements

- Legal requirements
- Regulatory compliance
- Business rules

Why Use Standards?



Ensures consistency



Improves communication

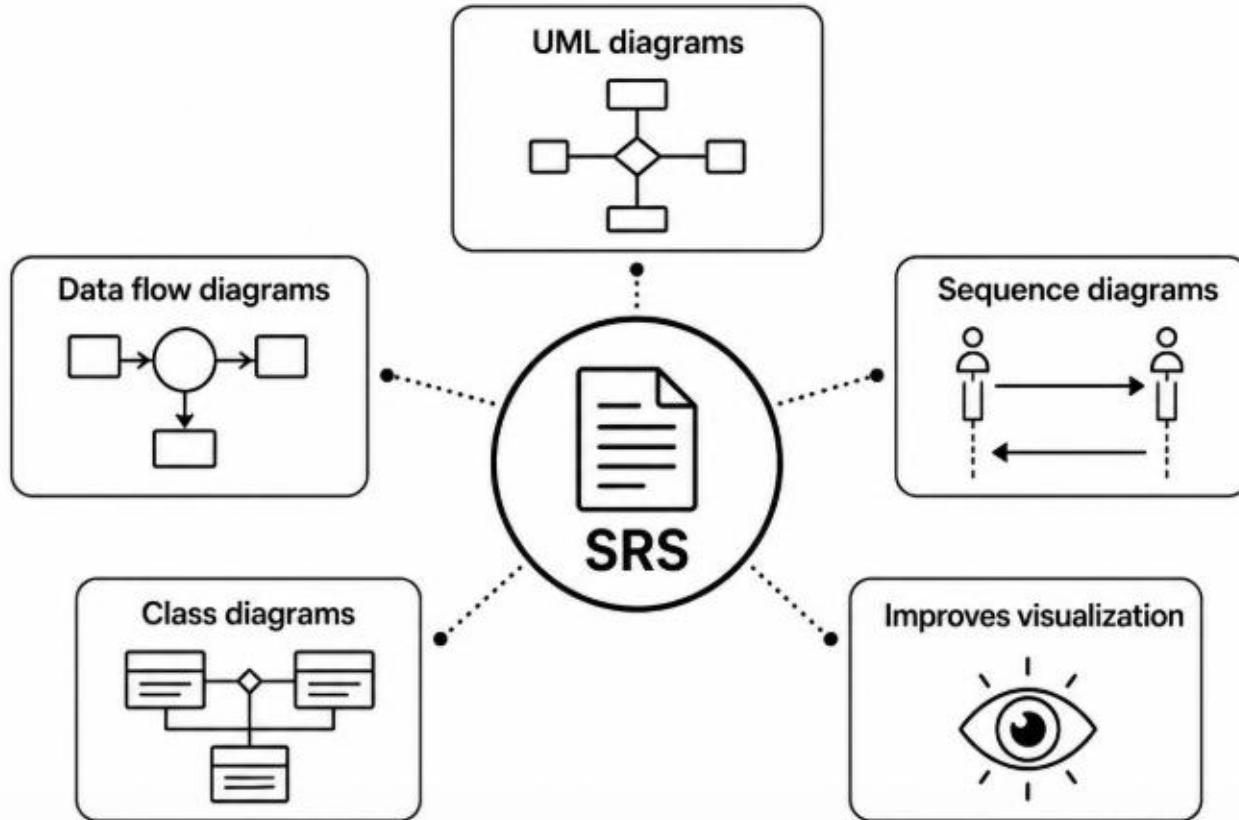


Facilitates review and validation

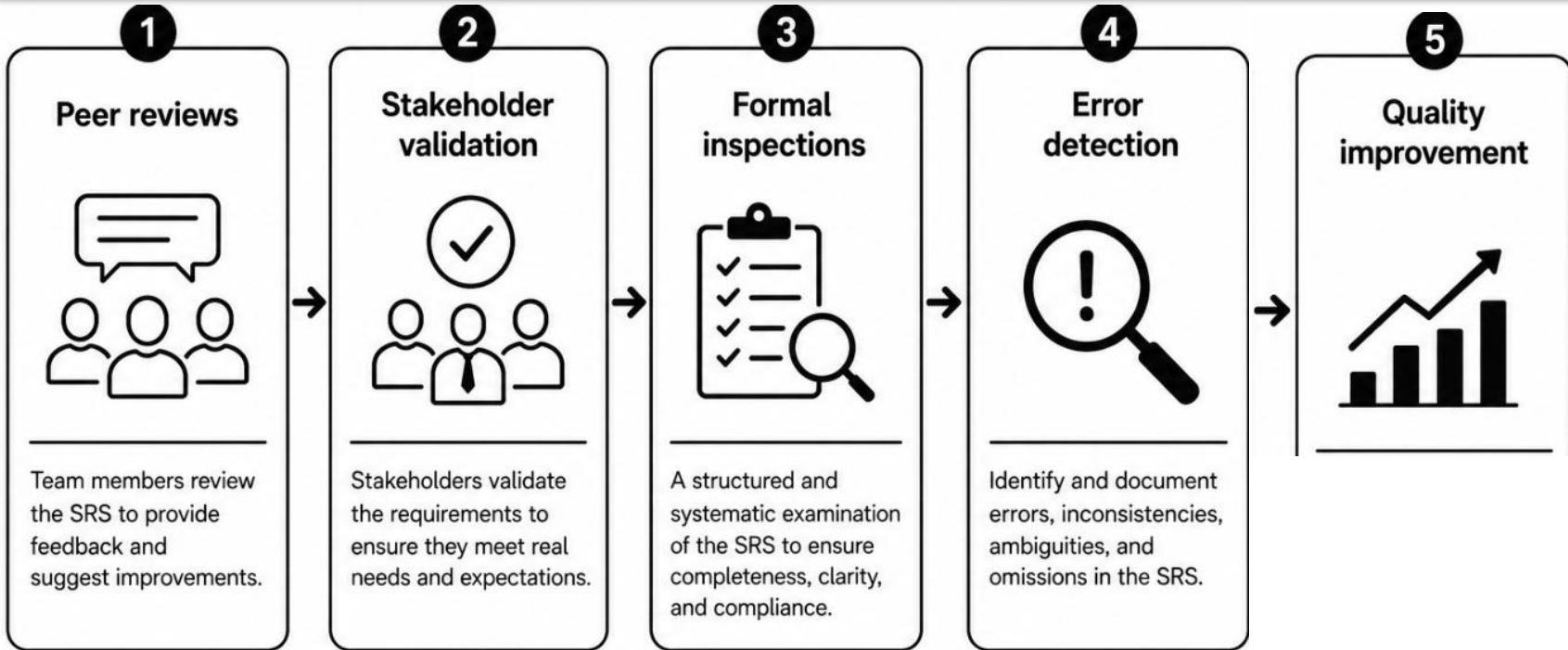


Supports compliance and auditing

Use of Diagrams in SRS



Review Process of SRS



Review typically done using **checklists**

Sample SRS Checklists

Category	Checklist Questions
Correctness	Do requirements reflect actual stakeholder needs?
	Are all requirements necessary and valid?
Clarity	Are requirements clear and unambiguous?
	Are vague terms avoided or properly defined?
Completeness	Are all functional requirements included?
	Are non-functional requirements specified?
	Are inputs, outputs, and error conditions defined?
Consistency	Are there any conflicting requirements?
	Is terminology used consistently?
Verifiability	Can each requirement be tested or measured?
	Are acceptance criteria defined?
Feasibility	Are requirements realistic within constraints (time, budget, technology)?

Sample SRS Checklists ... cont'd

Category	Checklist Questions
Prioritization	Are requirements ranked by importance and stability?
Modifiability	Is the document well-structured and easy to modify? Is redundancy minimized?
Traceability	Does each requirement have a unique ID? Can requirements be traced backward and forward?
Design Independence	Does the SRS avoid implementation details?
Interfaces	Are external interfaces (user, hardware, software) defined?
Environment	Is the operating environment clearly specified?
Overall Quality	Is the document easy to read and well-organized? Has the SRS been reviewed and approved?

Approval of SRS



Stakeholder agreement

All stakeholders review the SRS and agree that the requirements meet their needs and expectations.



Formal sign-off

Authorized stakeholders provide formal approval by signing off on the SRS.



Baseline establishment

The approved SRS is established as the official baseline for future reference and change control.



Ready for development

With approval, the project team can proceed with system design, development, and related activities.



Reference document

The approved SRS serves as the official reference for all future project activities, including testing and maintenance.

SRS as a Contract

A Formal Agreement Between Two Parties



Figure 2. *SRS as a contract*

Note. Image generated using ChatGPT by OpenAI (2026).

SRS as a Contract ... cont'd



The SRS serves as a contract that aligns both parties and ensures a common understanding throughout the project lifecycle.



LEGAL REFERENCE

Acts as a legal document in case of disagreements.



ACCOUNTABILITY

Defines responsibilities and ensures ownership.



DEFINES EXPECTATIONS

Sets clear requirements and success criteria.



REDUCES DISPUTES

Minimizes misunderstandings and conflicts.



EFFICIENCY & TRUST

Promotes transparency, trust, and project success.



SRS is not just a document—it's a commitment to deliver value together.

Common Mistakes in SRS

- ≠ Ambiguous and incomplete requirements
- ≠ Inconsistencies between requirements
- ≠ Design solutions included instead of actual requirements
- ≠ Lack of traceability between requirements, design, testing & implementation
- ≠ Poor organization and overly long paragraphs.
- ≠ Missing glossary
- ≠ FRs and NFRs mixed without clear labeling
- ≠ Lack of unique identifiers

Summary

- ✓ If a desired capability or quality doesn't appear somewhere in the requirements agreement, no one should expect it to appear in the product.



Requirements specification is a critical foundation document



A well-structured SRS improves project success

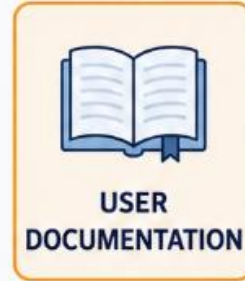


Standards like IEEE and ISO guide documentation

Summary



The SRS is the basis for subsequent project planning, design, and coding, as well as the foundation for system testing and user documentation.

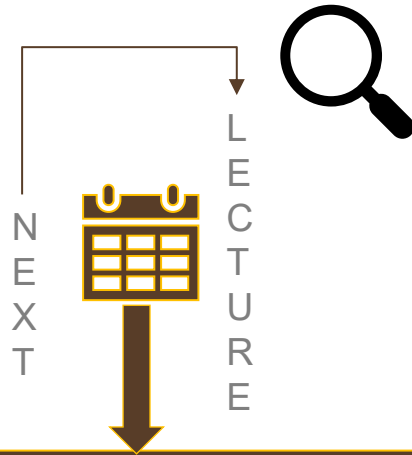


The SRS provides a common understanding and serves as a contract between stakeholders and the development team.

References

1. Beatty, K. W. (2013). Software Requirements (3rd ed.). Washington: Microsoft Press.

Thank You!



Writing Requirements