มหาวิทยาลัยราชภัฏนครปฐม
Nakhon Pathom Rajabhat University

Software Requirements Engineering

วิศวกรรมความต้องการซอฟต์แวร์

Suphitcha Chanrueang

1

มหาวิทยาลัย
ราชภัฏนครปฐม

Chapter 7

Software Requirements Specification

การเขียนเอกสารความต้องการซอฟต์แวร์

2

Outline

- Introduction
- Overview of IEEE SRS Template
- Detailed Walkthrough of Template Sections
  1. Introduction
  2. Overall Description
  3. System Features
  4. External Interface Requirements
  5. System Attributes
  6. Other Nonfunctional Requirements
  7. Appendices

3

Objectives

1. Understand the importance and principles of Software Requirements Specification (SRS).
2. Learn techniques for clear and standardized documentation of software requirements.
3. Understand the validation and review process to ensure correctness and alignment with user needs.
4. Apply prioritization techniques to determine the importance of requirements in development.

4

Understanding the Software Requirements Specification

What is an SRS?

Think of a Software Requirements Specification (SRS) as a detailed blueprint for a software project. Just as an architect creates a comprehensive plan before constructing a building, software developers use the SRS as a definitive guide that outlines exactly what the software will do, how it will behave, and what constraints it must operate within.

Importance of SRS:

1. Clear communication between stakeholders (clients, developers, testers).
2. Ensures software meets user expectations.

5

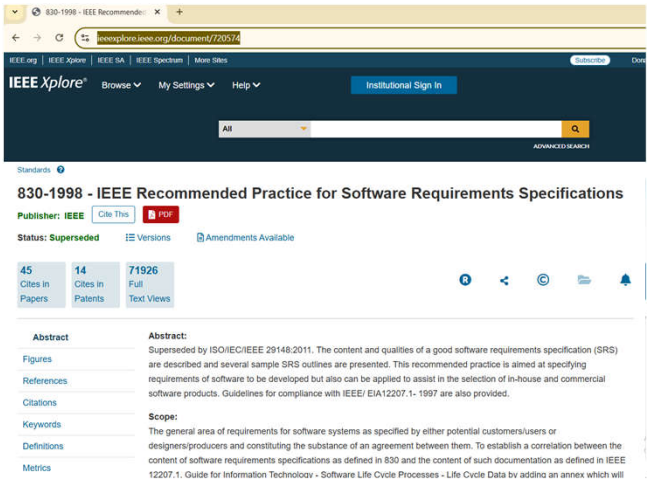
Overview of IEEE SRS Template

- IEEE Standard 830-1998: The standard provides a structured approach to creating an SRS.
- Template Structure:
  - A template with defined sections that guide the creation of an SRS.
  - Helps standardize the documentation of software requirements.

6

Overview of IEEE SRS Template

https://ieeexplore.ieee.org/document/720574



7

Section 1 - Introduction

- **Purpose:** Defines the purpose of the SRS document.

Example: "This document specifies the requirements for the XYZ Software Application.“

- **Scope:** Describes the system's boundaries and objectives.

Example: "The XYZ system is designed to manage customer relationships and streamline communication.“

- **Definitions, Acronyms, and Abbreviations:** List of terms used.

Example: "API - Application Programming Interface.“

- **References:** List of external documents, standards, or references.

Example: "IEEE Std 830-1998."

8

Section 1 (continued) - Introduction

Overview: Provides a summary of what the document will cover.

Example: "The system's functionality, requirements, interfaces, and system attributes are described in the following sections."

9

Section 2 - Overall Description

Product Perspective: Explains how the software fits into the larger system or operates independently.

Example: "This application is a standalone system that integrates with existing CRM platforms."

Product Features: A list of major features of the software.

Example: "Feature 1: User Login; Feature 2: Data Reporting; Feature 3: Email Notifications."

User Classes and Characteristics: Describes different user groups.

Example: "Admin users, regular users, guest users."

10

Section 2 (continued) - Overall Description

Operating Environment: Lists the technical environment for the software.

- *Example:* "The application will run on Windows, macOS, and Linux systems."

Design and Implementation Constraints: Describes any restrictions on design or development.

- *Example:* "The system must use MySQL for data storage."

User Documentation: Describes the available user support materials.

- *Example:* "User manual and online help."

11

Section 3 - System Features

Detailed Description of Features:

- For each feature, define:
  - Description: What the feature does.
  - Stimulus/Response Sequences: How the system should behave when triggered.
  - Functional Requirements: Clear, precise requirements for each feature.

Example (Feature 1: User Login):

- Description: Allows users to log in to the system.
- Stimulus: User enters username and password.
- Response: System validates credentials and grants access.

12

Section 4 - External Interface Requirements

User Interfaces: Description of the graphical interface and user interactions.

Example: "The login screen will have fields for username and password, with a 'Submit' button."

Hardware Interfaces: Describes the physical devices the software will interface with.

Example: "The system will connect to barcode scanners via USB."

Software Interfaces: Specifies external software systems or APIs the software will interact with.

Example: "The system will use the Google Maps API for location services."

13

Section 4 (continued) - External Interface Requirements

Communication Interfaces: Specifies communication protocols or data exchange formats.

Example: "The system will communicate with external servers using HTTP/HTTPS protocols and JSON data format."

14

Section 5 - System Attributes

Reliability: Describes the expected reliability of the system.

Example: "The system should maintain 99.9% uptime.“

Availability: Describes system availability requirements.

Example: "The system must be available 24/7, with no planned downtime exceeding 2 hours per month.“

Security: Describes security requirements such as encryption and access control.

Example: "All user data should be encrypted using AES-256 encryption."

15

Section 5 (continued) - System Attributes

Maintainability: Describes how easy it should be to update or maintain the system.

Example: "The system should allow for over-the-air updates.“

Portability: Describes the system's ability to run on different platforms or environments.

Example: "The software must be compatible with both Android and iOS platforms."

16

Section 6 - Other Nonfunctional Requirements

Performance Requirements: Describes how fast the system should perform certain actions.

Example: "The login process should take no longer than 3 seconds.“

Safety Requirements: Describes how the system ensures safety during operation.

Example: "The system should alert users if a security breach is detected.“

Legal Requirements: Describes compliance with laws and regulations.

Example: "The software must comply with GDPR for users in the European Union."

17

Section 7 - Appendices

Appendix A: Glossary of terms used in the document.

Example: "API - A set of protocols for building and interacting with software applications.“

Appendix B: Supporting diagrams, tables, or additional references.

Example: "Figure 1: System architecture diagram."

18

Writing Clear and Precise Requirements

- **Key Tips for Writing Requirements:**
  - Use simple, clear language.
  - Avoid ambiguity (e.g., instead of "The system should be fast," say "The system should process requests in less than 2 seconds").
  - Be specific about the behavior of the system.

19

Common Pitfalls in Writing SRS

- **Lack of Clarity:** Requirements should be precise and unambiguous.
- **Overly Complex Language:** Use simple language to avoid misunderstandings.
- **Missing Details:** Ensure all functional and non-functional requirements are well-documented.

20

Example of a Complete Feature

Feature Name: "Order Processing"

- **Description:** The system will allow users to place orders for products.
- **Stimulus/Response Sequences:**
  - **Stimulus:** User selects a product and adds it to the cart.
  - **Response:** The system displays the cart with the selected product.
- **Functional Requirements:**
  - The system should allow users to modify the order quantity.
  - The system should calculate the total price based on quantity and discounts.

21

Conclusion

Recap of Key Points:

- The IEEE SRS template provides a clear, structured approach to documenting software requirements.
- Each section of the SRS serves a specific purpose to ensure the software is well-defined and meets user needs.

Next Steps:

- Practice writing SRS for small software projects.
- Review example SRS documents to understand real-world applications.

22

มหาวิทยาลัย
ราชภัฏนครปฐม

Thank You

23