Software Requirements Specification (SRS) Template

Items that are intended to stay in as part of your document are in **bold**; explanatory comments are in *italic* text. Plain text is used where you might insert wording about your project.

The document in this file is an annotated outline for specifying software requirements, adapted from the IEEE Guide to Software Requirements Specifications (Std 830-1993).

Tailor this to your needs, removing explanatory comments as you go along. Where you decide to omit a section, you might keep the header, but insert a comment saying why you omit the data.

Agency Name

Project Name

Software Requirements Specification Document

Version: (n) **Date: (mm/dd/yyyy)**

Document History and Distribution

1. Revision History

Revision #	Revision Date	Description of Change	Author

2. Distribution

Recipient Name	Recipient Organization	Distribution Method

Table of Contents

1. Introduction

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

1.1 Purpose

Identify the purpose of this SRS and its intended audience. In this subsection, describe the purpose of the particular SRS and specify the intended audience for the SRS.

1.2 Scope

In this subsection:
- *(1) Identify the software product(s) to be produced by name*
- *(2) Explain what the software product(s) will, and, if necessary, will not do*
- *(3) Describe the application of the software being specified, including relevant benefits, objectives, and goals*
- *(4) Be consistent with similar statements in higher-level specifications if they exist*

1.3 Definitions, Acronyms, and Abbreviations.

Provide the definitions of all terms, acronyms, and abbreviations required to properly interpret the SRS. This information may be provided by reference to one or more appendices in the SRS or by reference to documents. This information may be provided by reference to an Appendix.

1.4 References

In this subsection:
- *(1) Provide a complete list of all documents referenced elsewhere in the SRS*
- *(2) Identify each document by title, report number (if applicable), date, and publishing organization*
- *(3) Specify the sources from which the references can be obtained.*

This information can be provided by reference to an appendix or to another document.

1.5 Overview

In this subsection:
- *(1) Describe what the rest of the SRS contains*
- *(2) Explain how the SRS is organized*

2. The Overall Description

Describe the general factors that affect the product and its requirements. This section does not state specific requirements. Instead, it provides a background for those requirements, which are defined in section 3, and makes them easier to understand.

2.1 Product Perspective

Put the product into perspective with other related products. If the product is independent and totally self-contained, it should be so stated here. If the SRS defines a product that is a component of a larger system, as frequently occurs, then this subsection relates the requirements of the larger system to functionality of the software and identifies interfaces between that system and the software.

A block diagram showing the major components of the larger system, interconnections, and external interfaces can be helpful.

The following subsections describe how the software operates inside various constraints.

2.1.1 System Interfaces

List each system interface and identify the functionality of the software to accomplish the system requirement and the interface description to match the system.

2.1.2 Interfaces

Specify:

1. *The logical characteristics of each interface between the software product and its users*
2. *All the aspects of optimizing the interface with the person who must use the system*

2.1.3 Hardware Interfaces

Specify the logical characteristics of each interface between the software product and the hardware components of the system. This includes configuration characteristics. It also covers such matters as what devices are to be supported, how they are to be supported and protocols.

2.1.4 Software Interfaces

Specify the use of other required software products and interfaces with other application systems. For each required software product, include:

(1) Name
(2) Mnemonic
(3) Specification number
(4) Version number
(5) Source

For each interface, provide:

(1) Discussion of the purpose of the interfacing software as related to this software product
(2) *Definition of the interface in terms of message content and format*

2.1.5 Communications Interfaces

Specify the various interfaces to communications such as local network protocols, etc.

2.1.6 Memory Constraints

Specify any applicable characteristics and limits on primary and secondary memory.

2.1.7 Operations

Specify the normal and special operations required by the user such as:

(1) The various modes of operations in the user organization
(2) Periods of interactive operations and periods of unattended operations
(3) Data processing support functions
(4) Backup and recovery operations

(Note: This is sometimes specified as part of the User Interfaces section.)

2.1.8 Site Adaptation Requirements

In this section:

(1) Define the requirements for any data or initialization sequences that are specific to a given site, mission, or operational mode
(2) Specify the site or mission-related features that should be modified to adapt the software to a particular installation

2.2 Product Functions

Provide a summary of the major functions that the software will perform. Sometimes the function summary that is necessary for this part can be taken directly from the section of the higher-level specification (if one exists) that allocates particular functions to the software product.

For clarity:
(1) The functions should be organized in a way that makes the list of functions understandable to the customer or to anyone else reading the document for the first time.
(2) Textual or graphic methods can be used to show the different functions and their relationships. Such a diagram is not intended to show a design of a product but simply shows the logical relationships among variables.

2.3 User Characteristics

Describe those general characteristics of the intended users of the product including educational level, experience, and technical expertise. Do not state specific requirements but rather provide the reasons why certain specific requirements are later specified in section 3.

2.4 Constraints

Provide a general description of any other items that will limit the developer's options. These can include:

(1) Regulatory policies
(2) Hardware limitations (for example, signal timing requirements)
(3) Interface to other applications
(4) Parallel operation
(5) Audit functions
(6) Control functions
(7) Higher-order language requirements
(8) *Signal handshake protocols (for example, XON-XOFF, ACK-NACK)*
(9) *Reliability requirements*
(10) Criticality of the application
(11) Safety and security considerations

2.5 Assumptions and Dependencies

List each of the factors that affect the requirements stated in the SRS. These factors are not design constraints on the software but are, rather, any changes to them that can affect the requirements in the SRS. For example, an assumption might be that a specific operating system would be available on the hardware designated for the software product. If, in fact, the operating system were not available, the SRS would then have to change accordingly.

2.6 Apportioning of Requirements.

Identify requirements that may be delayed until future versions of the system.

3. Specific Requirements

This section contains all the software requirements at a level of detail sufficient to enable designers to design a system to satisfy those requirements, and testers to test that the system satisfies those requirements. Throughout this section, every stated requirement should be externally perceivable by users, operators, or other external systems. These requirements should include at a minimum a description of every input (stimulus) into the system, every output (response) from the system and all functions performed by the system in response to an input or in support of an output. The following principles apply:

(1) Specific requirements should be stated with all the characteristics of a good SRS
- *correct*
- *unambiguous*
- *complete*
- *consistent*
- *ranked for importance and/or stability*
- *verifiable*
- *modifiable*
- *traceable*

(2) Specific requirements should be cross-referenced to earlier documents that relate
(3) All requirements should be uniquely identifiable
(4) Careful attention should be given to organizing the requirements to maximize readability

Before examining specific ways of organizing the requirements it is helpful to understand the various items that comprise requirements as described in the following subclasses.

3.1 External Interfaces

This contains a detailed description of all inputs into and outputs from the software system. It complements the interface descriptions in section 2 but does not repeat information there.

It contains both content and format as follows:

- *Name of item*
- *Description of purpose*
- *Source of input or destination of output*
- *Valid range, accuracy and/or tolerance*
- *Units of measure*
- *Timing*
- *Relationships to other inputs/outputs*
- *Screen formats/organization*
- *Window formats/organization*
- *Data formats*
- *Command formats*
- *End messages*

3.2 Functions

Functional requirements define the fundamental actions that must take place in the software in accepting and processing the inputs and in processing and generating the outputs. These are generally listed as "shall" statements starting with "The system shall…

These include:

- *Validity checks on the inputs*
- *Exact sequence of operations*
- *Responses to abnormal situation, including*
  - *Overflow*
  - *Communication facilities*
  - *Error handling and recovery*
- *Effect of parameters*
- *Relationship of outputs to inputs, including*
  - *Input/Output sequences*
  - *Formulas for input to output conversion*

It may be appropriate to partition the functional requirements into sub-functions or sub-processes. This does not imply that the software design will also be partitioned that way.

3.3 Performance Requirements

This subsection specifies both the static and the dynamic numerical requirements placed on the software or on human interaction with the software, as a whole. Static numerical requirements may include:

(a) The number of terminals to be supported
(b) The number of simultaneous users to be supported
(c) Amount and type of information to be handled

Static numerical requirements are sometimes identified under a separate section entitled capacity.

Dynamic numerical requirements may include, for example, the numbers of transactions and tasks and the amount of data to be processed within certain time periods for both normal and peak workload conditions.

All of these requirements should be stated in measurable terms.

For example,

95% of the transactions shall be processed in less than 1 second

rather than,

An operator shall not have to wait for the transaction to complete.

(Note: Numerical limits applied to one specific function are normally specified as part of the processing subparagraph description of that function.)

3.4 Logical Database Requirements

This section specifies the logical requirements for any information that is to be placed into a database. This may include:

- *Types of information used by various functions*
- *Frequency of use*
- *Accessing capabilities*
- *Data entities and their relationships*
- *Integrity constraints*
- *Data retention requirements*

3.5 Design Constraints

Specify design constraints that can be imposed by other standards, hardware limitations, etc.

3.5.1 Standards Compliance

Specify the requirements derived from existing standards or regulations. They might include:

1. *Report format*
2. *Data naming*
3. *Accounting procedures*
4. *Audit Tracing*

For example, this could specify the requirement for software to trace processing activity. Such traces are needed for some applications to meet minimum regulatory or financial standards. An audit trace requirement may, for example, state that all changes to a payroll database must be recorded in a trace file with before and after values.

3.6 Software System Attributes

There are a number of attributes of software that can serve as requirements. It is important that required attributes by specified so that their achievement can be objectively verified. The following items provide a partial list of examples.

3.6.1 Reliability

Specify the factors required to establish the required reliability of the software system at time of delivery.

3.6.2 Availability

Specify the factors required to guarantee a defined availability level for the entire system such as checkpoint, recovery, and restart.

3.6.3 Security

Specify the factors that would protect the software from accidental or malicious access, use, modification, destruction, or disclosure. Specific requirements in this area could include the need to:

- *Utilize certain cryptographic techniques*
- *Keep specific log or history data sets*
- *Assign certain functions to different modules*
- *Restrict communications between some areas of the program*
- *Check data integrity for critical variables*

3.6.4 Maintainability

Specify attributes of software that relate to the ease of maintenance of the software itself. There may be some requirement for certain modularity, interfaces, complexity, etc. Requirements should not be placed here just because they are thought to be good design practices.

3.6.5 Portability

Specify attributes of software that relate to the ease of porting the software to other host machines and/or operating systems. This may include:

- *Percentage of components with host-dependent code*
- *Percentage of code that is host dependent*
- *Use of a proven portable language*
- *Use of a particular compiler or language subset*
- *Use of a particular operating system*

Once the relevant characteristics are selected, a subsection should be written for each, explaining the rationale for including this characteristic and how it will be tested and measured. A chart like this might be used to identify the key characteristics (rating them High or Medium), then identifying which are preferred when trading off design or implementation decisions (with the ID of the preferred one indicated in the chart to the right).

ID	Characteristic	H/M/L	1	2	3	4	5	6	7	8	9	10	11	12
1	Correctness		▨											
2	Efficiency		▨	▨										
3	Flexibility		▨	▨	▨									
4	Integrity/Security		▨	▨	▨	▨								
5	Interoperability		▨	▨	▨	▨	▨							
6	Maintainability		▨	▨	▨	▨	▨	▨						
7	Portability		▨	▨	▨	▨	▨	▨	▨					
8	Reliability		▨	▨	▨	▨	▨	▨	▨	▨				
9	Reusability		▨	▨	▨	▨	▨	▨	▨	▨	▨			
10	Testability		▨	▨	▨	▨	▨	▨	▨	▨	▨	▨		
11	Usability		▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	
12	Availability		▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨

Definitions of the quality characteristics not defined in the paragraphs above follow.

- *Correctness - extent to which program satisfies specifications, fulfills user's mission objectives*
- *Efficiency - amount of computing resources and code required to perform function*

- *Flexibility - effort needed to modify operational program*
- *Interoperability - effort needed to couple one system with another*
- *Reliability - extent to which program performs with required precision*
- *Reusability - extent to which it can be reused in another application*
- *Testability - effort needed to test to ensure performs as intended*
- *Usability - effort required to learn, operate, prepare input, and interpret output*

3.7 Organizing the Specific Requirements

For anything but trivial systems the detailed requirements tend to be extensive. For this reason, it is recommended that careful consideration be given to organizing these in a manner optimal for understanding. There is no one optimal organization for all systems. Different classes of systems lend themselves to different organizations of requirements in section 3. Some of these organizations are described in the following subclasses.

3.7.1 System Mode

Some systems behave quite differently depending on the mode of operation. When organizing by mode there are two possible outlines. The choice depends on whether interfaces and performance are dependent on mode.

3.7.2 User Class

Some systems provide different sets of functions to different classes of users.

3.7.3 Objects

Objects are real-world entities that have a counterpart within the system. Associated with each object is a set of attributes and functions. These functions are also called services, methods, or processes. Note that sets of objects may share attributes and services. These are grouped together as classes.

3.7.4 Feature

A feature is an externally desired service by the system that may require a sequence of inputs to effect the desired result. Each feature is generally described in as sequence eof stimulus-response pairs.

3.7.5 Stimulus

Some systems can be best organized by describing their functions in terms of stimuli.

3. 7.6 Response

Some systems can be best organized by describing their functions in support of the generation of a response.

3.7.7 Functional Hierarchy

When none of he above organizational schemes prove helpful, the overall functionality can be organized into a hierarchy of functions organized by either common inputs, common outputs, or common internal data access. Data flow diagrams and data dictionaries can be use dot show the relationships between and among the functions and data.

3.8 Additional Comments

Whenever a new SRS is contemplated, more than one of the organizational techniques given in 3.7 may be appropriate. In such cases, organize the specific requirements for multiple hierarchies tailored to the specific needs of the system under specification.

Three are many notations, methods, and automated support tools available to aid in the documentation of requirements. For the most part, their usefulness is a function of organization. For example, when organizing by mode, finite state machines or state charts may prove helpful; when organizing by object, object-oriented analysis may prove helpful; when organizing by feature, stimulus-response sequences may prove helpful; when organizing by functional hierarchy, data flow diagrams and data dictionaries may prove helpful.

In any of the outlines below, those sections called "Functional Requirement i" may be described in native language, in pseudocode, in a system definition language, or in four subsections titled: Introduction, Inputs, Processing, Outputs.

4. Change Management Process

Identify the change management process to be used to identify, log, evaluate, and update the SRS to reflect changes in project scope and requirements.

5. Document Approvals

Identify the approvers of the SRS document. Approver name, signature, and date should be used.

6. Supporting Information

The supporting information makes the SRS easier to use. It includes:

- *Table of Contents*
- *Index*
- *Appendices*

The Appendices are not always considered part of the actual requirements specification and are not always necessary. They may include:

(a) Sample I/O formats, descriptions of cost analysis studies, results of user surveys
(b) Supporting or background information that can help the readers of the SRS
(c) A description of the problems to be solved by the software
(d) Special packaging instructions for the code and the media to meet security, export, initial loading, or other requirements

When Appendices are included, the SRS should explicitly state whether or not the Appendices are to be considered part of the requirements.

Tables on the following pages provide alternate ways to structure section 3 on the specific requirements.

Outline for SRS Section 3
Organized by Mode: Version 1

3. Specific Requirements
  - 3.1 External interface requirements
    - 3.1.1 User interfaces
    - 3.1.2 Hardware interfaces
    - 3.1.3 Software interfaces
    - 3.1.4 Communications interfaces
  - 3.2 Functional requirements
    - 3.2.1 Mode 1
      - 3.2.1.1 Functional requirement 1.1
      - .....
      - 3.2.1.*n* Functional requirement 1.*n*
    - 3.2.2 Mode 2
      - .....
    - 3.2.*m* Mode *m*
      - 3.2.*m*.1 Functional requirement *m*.1
      - .....
      - 3.2.*m.n* Functional requirement *m.n*
  - 3.3 Performance Requirements
  - 3.4 Design Constraints
  - 3.5 Software system attributes
  - 3.6 Other requirements

Outline for SRS Section 3
Organized by Mode: Version 2

3. Specific Requirements
  3.1 Functional Requirements
    3.1.1 Mode 1
      3.1.1.1 External interfaces
        3.1.1.1 User interfaces
        3.1.1.2 Hardware interfaces
        3.1.1.3 Software interfaces
        3.1.1.4 Communications interfaces
      3.1.1.2 Functional Requirement
        3.1.1.2.1 Functional requirement 1
        .....
        3.1.1.2.*n* Functional requirement *n*
      3.1.1.3 Performance
    3.1.2 Mode 2
      .....
    3.1.*m* Mode *m*
  3.2 Design constraints
  3.3 Software system attributes
  3.4 Other requirements

Outline for SRS Section 3
Organized by User Class

3. Specific Requirements
   - 3.1 External interface requirements
     - 3.1.1 User interfaces
     - 3.1.2 Hardware interfaces
     - 3.1.3 Software interfaces
     - 3.1.4 Communications interfaces
   - 3.2 Functional requirements
     - 3.2.1 User class 1
       - 3.2.1.1 Functional requirement 1.1
       - .....
       - 3.2.1.*n* Functional requirement 1.*n*
     - 3.2.2 User class 2
       - .....
     - 3.2.*m* User class *m*
       - 3.2.*m*.1 Functional requirement *m*.1
       - .....
       - 3.2.*m.n* Functional requirement *m.n*
   - 3.3 Performance Requirements
   - 3.4 Design Constraints
   - 3.5 Software system attributes
   - 3.6 Other requirements

Outline for SRS Section 3
Organized by Object

3 Specific Requirements
- 3.1 External interface requirements
  - 3.1.1 User interfaces
  - 3.1.2 Hardware interfaces
  - 3.1.3 Software interfaces
  - 3.1.4 Communications interfaces
- 3.2 Classes/Objects
  - 3.2.1 Class/Object 1
    - 3.2.1.1 Attributes (direct or inherited)
      - 3.2.1.1.1 Attribute 1
      - .....
      - 3.2.1.1.*n* Attribute *n*
    - 3.2.1.2 Functions (services, methods, direct or inherited)
      - 3.2.1.2.1 Functional requirement 1.1
      - .....
      - 3.2.1.2.*m* Functional requirement 1.*m*
    - 3.2.1.3 Messages (communications received or sent)
  - 3.2.2 Class/Object 2
  - .....
  - 3.2.*p* Class/Object *p*
- 3.3 Performance Requirements
- 3.4 Design Constraints
- 3.5 Software system attributes
- 3.6 Other requirements

Outline for SRS Section 3
Organized by Feature

3 Specific Requirements
- 3.1 External interface requirements
  - 3.1.1 User interfaces
  - 3.1.2 Hardware interfaces
  - 3.1.3 Software interfaces
  - 3.1.4 Communications interfaces
- 3.2 System features
  - 3.2.1 System Feature 1
    - 3.2.1.1 Introduction/Purpose of feature
    - 3.2.1.2 Stimulus/Response sequence
      - 3.2.1.3 Associated functional requirements
        - 3.2.1.3.1 Functional requirement 1
        - .....
        - 3.2.1.3.*n* Functional requirement *n*
  - 3.2.2 System Feature 2
    - .....
  - 3.2.*m* System Feature *m*
    - .....
- 3.3 Performance Requirements
- 3.4 Design Constraints
- 3.5 Software system attributes
- 3.6 Other requirements

Outline for SRS Section 3
Organized by Stimulus

3 Specific Requirements
- 3.1 External interface requirements
  - 3.1.1 User interfaces
  - 3.1.2 Hardware interfaces
  - 3.1.3 Software interfaces
  - 3.1.4 Communications interfaces
- 3.2 Functional requirements
  - 3.2.1 Stimulus 1
    - 3.2.1.1 Functional requirement 1.1
    - .....
    - 3.2.1.*n* Functional requirement 1.*n*
  - 3.2.2 Stimulus 2
  - .....
  - 3.2.*m* Stimulus *m*
    - 3.2.*m*.1 Functional requirement *m*.1
    - .....
    - 3.2.*m*.*n* Functional requirement *m*.*n*
- 3.3 Performance Requirements
- 3.4 Design Constraints
- 3.5 Software system attributes
- 3.6 Other requirements

Outline for SRS Section 3
Organized by Response

3 Specific Requirements
- 3.1 External interface requirements
  - 3.1.1 User interfaces
  - 3.1.2 Hardware interfaces
  - 3.1.3 Software interfaces
  - 3.1.4 Communications interfaces
- 3.2 Functional requirements
  - 3.2.1 Response 1
    - 3.2.1.1 Functional requirement 1.1
    - .....
    - 3.2.1.*n* Functional requirement 1.*n*
  - 3.2.2 Response 2
  - .....
  - 3.2.*m* Response *m*
    - 3.2.*m*.1 Functional requirement *m*.1
    - .....
    - 3.2.*m.n* Functional requirement *m.n*
- 3.3 Performance Requirements
- 3.4 Design Constraints
- 3.5 Software system attributes
- 3.6 Other requirements

Outline for SRS Section 3
Organized by Functional Hierarchy

3 Specific Requirements
- 3.1 External interface requirements
  - 3.1.1 User interfaces
  - 3.1.2 Hardware interfaces
  - 3.1.3 Software interfaces
  - 3.1.4 Communications interfaces
- 3.2 Functional requirements
  - 3.2.1 Information flows
    - 3.2.1.1 Data flow diagram 1
      - 3.2.1.1.1 Data entities
      - 3.2.1.1.2 Pertinent processes
      - 3.2.1.1.3 Topology
    - 3.2.1.2 Data flow diagram 2
      - 3.2.1.2.1 Data entities
      - 3.2.1.2.2 Pertinent processes
      - 3.2.1.2.3 Topology
      - .....
    - 3.2.1.*n* Data flow diagram *n*
      - 3.2.1.*n*.1 Data entities
      - 3.2.1.*n*.2 Pertinent processes
      - 3.2.1.*n*.3 Topology
  - 3.2.2 Process descriptions
    - 3.2.2.1 Process 1
      - 3.2.2.1.1 Input data entities
      - 3.2.2.1.2 Algorithm or formula of process
      - 3.2.2.1.3 Affected data entities
    - 3.2.2.2 Process 2
      - 3.2.2.2.1 Input data entities
      - 3.2.2.2.2 Algorithm or formula of process
      - 3.2.2.2.3 Affected data entities
      - …..
    - 3.2.2.*m* Process *m*
      - 3.2.2.*m*.1 Input data entities
      - 3.2.2.*m*.2 Algorithm or formula of process
      - 3.2.2.*m*.3 Affected data entities
  - 3.2.3 Data construct specifications
    - 3.2.3.1 Construct 1
      - 3.2.3.1.1 Record type
      - 3.2.3.1.2 Constituent fields
    - 3.2.3.2 Construct 2
      - 3.2.3.2.1 Record type
      - 3.2.3.2.2 Constituent fields
      - …..

3.2.3.*p* Construct *p*
3.2.3.*p*.1 Record type
3.2.3.*p*.2 Constituent fields
3.2.4 Data dictionary
3.2.4.1 Data element 1
3.2.4.1.1 Name
3.2.4.1.2 Representation
3.2.4.1.3 Units/Format
3.2.4.1.4 Precision/Accuracy
3.2.4.1.5 Range
3.2.4.2 Data element 2
3.2.4.2.1 Name
3.2.4.2.2 Representation
3.2.4.2.3 Units/Format
3.2.4.2.4 Precision/Accuracy
3.2.4.2.5 Range
…..
3.2.4.*q* Data element *q*
3.2.4.*q*.1 Name
3.2.4.*q*.2 Representation
3.2.4.*q*.3 Units/Format
3.2.4.*q*.4 Precision/Accuracy
3.2.4.*q*.5 Range
3.3 Performance Requirements
3.4 Design Constraints
3.5 Software system attributes
3.6 Other requirements

Outline for SRS Section 3
Showing Multiple Organizations

- 3 Specific Requirements
  - 3.1 External interface requirements
    - 3.1.1 User interfaces
    - 3.1.2 Hardware interfaces
    - 3.1.3 Software interfaces
    - 3.1.4 Communications interfaces
  - 3.2 Functional requirements
    - 3.2.1 User class 1
      - 3.2.1.1 Feature 1.1
        - 3.2.1.1.1 Introduction/Purpose of feature
        - 3.2.1.1.2 Stimulus/Response sequence
        - 3.2.1.1.3 Associated functional requirements
      - 3.2.1.2 Feature 1.2
        - 3.2.1.2.1 Introduction/Purpose of feature
        - 3.2.1.2.2 Stimulus/Response sequence
        - 3.2.1.2.3 Associated functional requirements
        - …..
      - 3.2.1.*m* Feature 1.*m*
        - 3.2.1.*m*.1 Introduction/Purpose of feature
        - 3.2.1.*m*.2 Stimulus/Response sequence
        - 3.2.1.*m*.3 Associated functional requirements
    - 3.2.2 User class 2
    - .....
    - 3.2.*n* User class *n*
    - .....
  - 3.3 Performance Requirements
  - 3.4 Design Constraints
  - 3.5 Software system attributes
  - 3.6 Other requirements