1. Minimum Recommended Content

a. System overview.

b. CSCI (Computer Software Configuration Item) requirements:
   1. Functional requirements.
   2. Required states and modes.
   3. External interface requirements.
   4. Internal interface requirements.
   5. Internal data requirements.
   6. Adaptation requirements (data used to adapt a program to a given installation site or to given conditions in its operational environment).
   7. Safety requirements.
   8. Performance and timing requirements.
   9. Security and privacy requirements.
  10. Environment requirements.
  11. Computer resource requirements:
      a. Computer hardware resource requirements, including utilization requirements.
      b. Computer software requirements.
      c. Computer communications requirements.
  12. Software quality characteristics.
  13. Design and implementation constraints.
  14. Personnel-related requirements.
  15. Training-related requirements.
  16. Logistics-related requirements.
  17. Packaging requirements.
  18. Precedence and criticality of requirements.

c. Qualification provisions (e.g., demonstration, test, analysis, inspection).

d. Bidirectional requirements traceability.

e. Requirements partitioning for phased delivery.

f. Testing requirements that drive software design decisions (e.g., special system level timing requirements/checkpoint restart).

g. Supporting requirements rationale.

Software requirements and design specifications need not be textual and may include representations in rigorous specification languages, graphical representations, or specifications suitable for requirements or design analysis tools or methodologies.

2. Rationale

The Software Requirements Specification details the software performance, interface, and operational and quality assurance requirements for each computer software configuration items (CSCI).

This information specifies the product to be delivered by a provider to a customer; therefore, the software requirements specification (SRS) is a critical document/electronic record for any software project. All subsequent project products, including design, test, and implementation, are based on the SRS and any other record that captures the software requirements, e.g., data dictionary or interface requirements specification.

3. Guidance

When writing the SRS, it is important to capture specific, key information. Basic guidance for capturing that information is provided below. Other sections, not listed below, that may be included in an SRS are:

- Overview, purpose, audience, organization of the SRS.
• Definitions, acronyms, abbreviations specific to the SRS.
• Summary of history of system development, operation, maintenance. 381
• Project sponsor, acquirer, user, developer, and support agencies. 381
• Current and planned operating sites. 381
• References.
• Appendices.

Because requirements change and evolve over the life of the project, the SRS is typically revised as the project progresses. Guidance for managing these changes can be found in SWE-053 of this Handbook. Guidance for baselining and updating the SRS in preparation for life-cycle milestone reviews can be found in Topic 7.8 - Maturity of Life-Cycle Products at Milestone Reviews in this Handbook. Data on requirements volatility is tracked in the Software Metrics Report.

System overview
The SRS introduces the product whose requirements are captured in the SRS, including:

• Name or other identifier.
• General functionality, benefits, purpose, objectives, goals of the software.
• Background for the requirements, e.g., users, assumptions, constraints, functions, dependencies.
• Major components of the system and their interfaces.
• User interfaces, hardware interfaces, software interfaces, communication interfaces. (Note that software design interface descriptions will be captured in a separate Interface Design Description.

CSCI requirements
For purposes including, but not limited to, planning work, assigning resources, and understanding the size of the software, it is helpful to organize requirements by CSCIs. For each CSCI, include:

Functional requirements
Functional requirements cover the basic actions that the software should perform, "the functions that need to be done to accomplish the objectives" 273. The following questions may be helpful when identifying functional requirements:

• What functions need to be performed?
• Where do they need to be performed?
• How often? (also useful for performance and timing requirements)
• Under what operational and environmental conditions? (also useful for environmental requirements) 273

Consider the following when capturing functional requirements:

• Validity checks on the inputs. 214
• Exact sequence of operations. 214
• Responses to abnormal situations, including: 214
  • Overflow.
  • Communication facilities.
  • Error handling and recovery.
• Effect of parameters. 214
• Relationship of outputs to inputs, including: 214
  • Input/output sequences.
  • Formulas for input to output conversion.
• Relevant operational modes (nominal, critical, contingency).

Because functional requirements tend to be large in number, it may be helpful to organize them into groups or subsections appropriate for the software project. Suggested groupings include functionality, performance, or coupling.

Additionally, it is helpful to understand the rationale behind a requirement to understand its intent; therefore, consider co-locating with the functional requirements the "supporting requirements rationale" described later in this guidance.

Required states and modes
If the software "is required to operate in more than one state or mode having requirements distinct from other states or modes ... identify and define each state and mode. Examples of states and modes include idle, ready, active, post-use analysis, training, degraded, emergency, backup, launch, testing, and deployment. ... If states and/or modes are required, each requirement or group of requirements in this specification should be correlated to the states and modes. The correlation may be indicated by a table ... an appendix ... or by annotation of the requirements ..." 381

External interface requirements
External interface requirements cover all inputs and outputs for the software system and expand on the interfaces generally described in the system overview. When capturing requirements for external interfaces, consider including interfaces to items such as test equipment or transportation systems. Note that interface specifications may be captured in a separate interface requirements document; in that case, reference to the separate document needs to be included in the SRS.

Information such as the following is included, as applicable to the system:

• Identifier (name).
• Description.
• Source (input) or destination (output).
• Valid range.
• Units of measure.
• Timing.
• Window or screen layouts and relationships.
• Data formats.
• Command formats.
• Type of interface, e.g., real-time data transfer, storage-and-retrieval of data.  
• Characteristics, e.g., data type, size, format, security, frequency, of data elements that the software must provide, store, send, access, receive.  
• Characteristics of communication methods, e.g., message format, communication bands, transfer rate.

Internal interface requirements
Internal interface requirements can cover interfaces internal to the software (i.e., interfaces between functions), if those are not left to the design phase. Note that software design interface specifications are captured in an Interface Design Description, which needs to be referenced in the SRS.

When capturing internal interface requirements, information such as that noted above for external interface requirements applies, as appropriate.

Internal data requirements
Internal data requirements define the data and data structures, e.g., files, databases, that are part of the software. Internal data requirements include information such as:

• Data types.
• Modes of access, e.g., random, sequential.
• Size and format.
• Units of measure.

If a database is used, consider capturing the following requirements:

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

Adaptation requirements
Adaptation requirements describe data used to adapt a program to a given installation site or to given conditions in its operational environment. These requirements address topics such as “installation-dependent data to be provided by the [software] (e.g., site-dependent latitude and longitude or [communication dependencies such as local access codes]) and operational parameters that the [software] is required to use that may vary according to operational needs (e.g., parameters indicating operation-dependent targeting constants or data recording).”

Safety requirements
While the SRS is not required to have a specific section that addresses the safety requirements, safety requirements are to be included in the SRS and designated (marked) as safety requirements.

“Software safety requirements are required to carry a unique identification or tag for traceability purposes. A way to mark and trace these requirements throughout the development and operational phases is needed in order to enable assessment of impacts and changes to the requirements. The unique identification or tag can be a special section in the requirements document, or a flag beside the requirement, or within a database. The method of identification is not important as long as it can be used for traceability and assessment.”

Software safety requirements “are derived from the system safety requirements, standards, program specification, vehicle or facility requirements, and interface requirements.”

“Software safety requirements are required to carry a unique identification or tag for traceability purposes. A way to mark and trace these requirements throughout the development and operational phases is needed in order to enable assessment of impacts and changes to the requirements. The unique identification or tag can be a special section in the requirements document, or a flag beside the requirement, or within a database. The method of identification is not important as long as it can be used for traceability and assessment.”

“The valid and invalid modes or states of operation [are documented] within the software safety requirements.”

“All software related safety constraints between the hardware, operator, and software [are included] in the software requirements documentation... When the software, hardware or operator performs a safety critical function, document the hardware, software, and operator roles in that function, the precedence, and the failure modes as well as any known constraints, controls, mitigations, conditions, timing constraints, limits, and wear-out factors that impact how software needs to respond.”

Performance and timing requirements
Performance and timing requirements specify measurable capacities and manageable volumes of activities for the system. Consider the following when capturing these requirements:

• Number of simultaneous users to be supported.
• Number of concurrent transactions and/or tasks to be supported.
• Amount of data to be processed within a specified time period (normal and peak loads).
• System response times.
• Failure recovery times.
• Output data availability.
• External hardware interface timing.
• Command execution timing.
• Maximum bandwidth throughput or capacity.

Performance requirements need to be defined in terms of a “minimum acceptable value needed for the system to carry out its mission” and “the baseline level of performance desired.” This gives the design team a set of parameters in which to design the software solution.

Security and privacy requirements
Security and privacy requirements “specify the factors that protect the software from accidental or malicious access, use, modification, destruction, or disclosure. Specific requirements in this area could include the need to:

1. Utilize certain cryptographical techniques;
2. Keep specific log or history data sets;
3. Assign certain functions to different modules;
4. Restrict communications between some areas of the program;
Environmental requirements
Environmental requirements address environmental conditions that the system may encounter and/or in which the software must operate including “ground test, storage, transportation, launch, deployment, [target processors,] and normal operations from beginning of life to end of life.”

Computer resource requirements
Computer resource requirements include hardware resource requirements (including utilization requirements), computer software requirements, and computer communications requirements.

Computer hardware resource utilization includes items such as "maximum allowable use of processor capacity, memory capacity, input/output device capacity, auxiliary storage device capacity, and communications or network equipment capacity. The requirements (e.g., stated as percentages of the capacity of each computer hardware resource) include the conditions, if any, under which the resource utilization is to be measured." 381

Computer software requirements include items such as "operating systems, database management systems, communications and network software, utility software, input and equipment simulators, test software, ... The correct nomenclature, version, and documentation references of each such software item should be provided." 381

Computer communications requirements include items such as "geographic locations to be linked, configuration and network topology, transmission technique, data transfer rates, gateways, required system use times, type and volume of data to be transmitted or received, time boundaries for transmission, reception, and response, peak volumes of data, and diagnostic features.” 381

Software quality characteristics
Software quality characteristic requirements include requirements that specify software system attributes such as 381:

- Reliability (the ability to perform with correct, consistent results).
- Maintainability (the ability to be easily corrected).
- Availability (the ability to be accessed and operated when needed).
- Flexibility (the ability to be easily adapted to changing requirements).
- Portability (the ability to be easily modified for a new environment).
- Reusability (the ability to be used in multiple applications).
- Testability (the ability to be easily and thoroughly tested).
- Usability (the ability to be easily learned and used).
- Other attributes.

Design and implementation constraints
Design and implementation constraint requirements address constraints that "can be imposed by other standards, hardware limitations, etc." 214

"Examples include requirements concerning

- a) Use of a particular [Computer Software Configuration Item] CSCI architecture or requirements on the architecture, such as required databases or other software units, use of standard, military, or existing components, or use of Government- or acquirer-furnished property (equipment, information, or software)
- b) Use of particular design or implementation standards, use of particular data standards, and use of a particular programming language
- c) Flexibility and expandability that must be provided to support anticipated areas of growth or changes in technology, threat, or mission." 381

Personnel-related requirements
Personnel requirements include requirements for the way users interact with the system (e.g., responsibilities, activities, modes of operation) as well as qualifications (e.g., skill level, education, background) of each user role. Interactions among user roles and permissible activities for those roles are also considered, as well as "requirements for [the] number of simultaneous users and for built-in help or training features. Also included should be the human factors engineering requirements, if any, imposed on the CSCI. These requirements should include, as applicable, considerations for the capabilities and limitations of humans, foreseeable human errors under both normal and extreme conditions, and specific areas where the effects of human error would be particularly serious. Examples include requirements for color and duration of error messages, physical placement of critical indicators or keys, and use of auditory signals." 381

Training-related requirements
Training requirements may be part of personnel-related requirements if they describe the training required before users can properly and safely interact and use the system. Training requirements may also describe training software to be included in the software system.

Logistics-related requirements
Logistics-related requirements "may include system maintenance, software support, system transportation modes, supply system requirements, impact on existing facilities, and impact on existing equipment." 381

Packaging requirements
Packaging requirements address packaging, labeling, preparing, and handling the software for delivery, e.g., delivery on DVDs that are labeled and packaged a certain way. 381

Precedence and criticality of requirements
Precedence of requirements, linking them to safety, criticality, performance, or prioritizing them based on cost, dependency, or importance facilitates selection of existing software to fulfill the requirements, allows selection of the most important requirements to implement, and can be used to define the capabilities of each software release or build. 061 "Having prioritized customer requirements ... ensures that functional and quality attribute requirements critical to the customer and other stakeholders are addressed quickly." 157 Setting precedence and criticality of requirements also facilitates planning for potential schedule or budget shortfalls. Interviewing stakeholders can help facilitate prioritization of requirements, particularly for development life-cycle models that focus on addressing high-priority requirements as defined by the customer.

5. Check data integrity for critical variables." 214
Qualification provisions, e.g., demonstration, test, analysis, inspection
In addition to the requirements themselves, their verification methods "should be included in the software requirements document, either when the requirement is stated or in a separate verification matrix at the end of the document." Verification methods include test, inspection, analysis, demonstration.

Bidirectional requirements traceability
Guidance for bidirectional requirements traceability is found in SWE-052 of this Handbook.

Requirements partitioning for phased delivery
"If the software will be developed and released in increments (phased delivery), define which requirements ... will be met for each release." 381

Testing requirements that drive software design decisions
Systems may have special testing requirements, such as special system-level timing requirements, checkpoint restart requirements, or built-in self tests that must be considered when making software design decisions or when the testing may not be feasible or possible. When those situations occur, requirements reflecting those conditions need to be captured and included in the SRS.

Supporting requirements rationale
Supporting rationale "Provides additional information to help clarify the intent of the requirements at the time they were written.” 273 The supporting rationale may be located with the requirements to which it applies. Consider including the following in the requirements rationale:

- Reason for the requirement.
- Assumptions, e.g., requirement assumes availability of new technology development.
- Relationships, e.g., requirement is based on expectation for how users will interact with the software.
- Design constraints, e.g., requirements written based on decisions made during design phase.

Guidance for SWE-050 also provides information on requirements rationale.

Additional guidance related to documenting software requirements may be found in the following requirements in this Handbook:

<table>
<thead>
<tr>
<th>Requirement</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SWE-050</td>
<td>Software Requirements</td>
</tr>
<tr>
<td>SWE-052</td>
<td>Bidirectional Traceability Between Higher Level Requirements and Software Requirements</td>
</tr>
<tr>
<td>SWE-053</td>
<td>Manage Requirements Change</td>
</tr>
</tbody>
</table>

4. Small Projects

Software requirements documents are typically created from a template with the information filled in as the document grows throughout the requirements engineering activities. This is the simplest method but can be time-consuming and error-prone. Automating all or part of the requirements document creation can help with both issues. Some requirements management tools (from a previous project or owned by the Center or Agency if costs are an issue) may be able to export requirements in a format that can be included in the requirements document. Another option is to use an appropriate scripting language to write a script to pull the requirements from the tool and import them into the requirements document in an automated fashion.

For projects with a relatively small number of requirements to document, it may be helpful to use a simple spreadsheet to document and maintain the requirements. Software requirements need not be textual and may include representations in rigorous specification languages, graphical representations, or specifications suitable for requirements analysis tools or methodologies.

5. Resources

- (SWEREF-273)
5.1 Tools

Tools relative to this Topic may be found in the table below. You may wish to reference the Tools Table (click here to visit) in this handbook for an evolving list of these and other tools in use at NASA. Note that this table should not be considered all-inclusive, nor is it an endorsement of any particular tool. Check with your Center to see what tools are available to facilitate compliance with this requirement.

6. Lessons Learned

- Probable Scenario for Mars Polar Lander Mission Loss (1998) (Affects of an incomplete software requirements specification). Lesson Number 0938: "All known hardware operational characteristics, including transients and spurious signals, must be reflected in the software requirements documents and verified by test." 529
- Consider Language Differences When Conveying Requirements to Foreign Partners (1997) (Diagrams may be useful in requirements specifications). Lesson Number 0608: "It is especially important when working with foreign partners to document requirements in terms that describe the intent very clearly; include graphics where possible." 511