Requirements Engineering

A Methodology for Writing High Quality Requirement Specifications and for Evaluating Existing Ones

Software Assurance Technology Center
NASA Goddard Space Flight Center
http://satc.gsfc.nasa.gov

Dr. Linda Rosenberg
301-286-0087
Linda.Rosenberg@gsfc.nasa.gov
Overview

Introduction
Aspects of Requirement Development
Writing Effective Requirements
  Requirement Documents
  Requirement Characteristics
  Requirement Statements
Levels of Requirements
Requirement Management & Repository
Quality, Traceability, & Linkage Metrics
Conclusion
Introduction

I SAYS WHAT I MEANS
I MEANS WHAT I SAYS
Generally accepted -

Requirements are basis for task,
verification and validation necessary

At project conclusion -

Some requirements not satisfied

==> Redo project components or accept less than what was specified
Program Development - Solution

Start at the beginning, get the requirements right

However, cannot get requirements right by magic, need tools and analysis techniques

==> Do it right the first time and start with the requirements!
Requirement Development Paths

System Requirements

- System Design
  - Hardware Requirements
  - Software Requirements

  Hardware Design
  Software Design

  ...
Requirements & Quality Management

**Quality Management** => Management of all aspects that concern quality assurance, quality assurance planning, and quality metrics, including verification and validation.

Verification - Determination whether the products fulfill the requirements established - “Are we building the system right?”

Validation - Determination of the correctness of the final product with respect to the user’s needs and requirements - “Are we building the right system?”
Requirement Concerns

- Inability to write correct software requirement specifications (SRS)
- Desire to truncate and shorten requirement phase activities
- Lack of customer specification and verification of requirement correctness
- Identification of appropriate tools and methods for requirement specification
- Lack of recognition of essentialness of requirement specification
- Lack of experience in writing requirement specifications
Error Detection Cost

Cost to Fix Errors Found in Testing Phase

∴ Find errors as soon as possible for maximum savings!
Definition Of Terms

- “Requirements Specification” and “Specification Documents” are terms that refer to the document(s) or the total set of statements that define a mission required (system level) capability and its environments.

- “Requirement(s)” or “specification statement(s)” are terms that refer to individual statements or sets of individual statements, i.e. sentences, that define individual functions or aspects of a capability or an environment.
Aspects of Requirement Development

1 - Specification

2 - Verification

3 - Management

4 - Measurement
Critical aspects in conveying information about system requirements:

**Structure**: The organization of the document to convey the appropriate level of detail

**Language**: The use of imperatives, directives, and the omission of options and ambiguity
2 - Requirement Verification

Issues critical to testing:

Stabilization:
- Is requirement volatility zero?
- Is requirement movement between builds stable?

Traceability:
- Do all requirements trace to higher and lower level documents?
- Are all requirements tested? Do they trace to a test?

Characterization to test program:
- What is the average number of requirements traced to each test?
3 - Requirements Management

Crucial management for successful delivery of a system:

– Manage requirements through the life cycle.
– Use of requirement management tools to provide important insight to the requirements.
– Extensive and comprehensive use of metrics to provide information so that management can take effective action.
4 - Requirement Measurement

What can be measured?

- Specification
  - Quality of statements for testability
- Verification
  - Stabilization
  - Trace to test
  - Trace to previous detail level
  - Expansion of requirements to test
- Management
  - Complexity of database links
Requirement Components

Requirement Management

- Requirement Specification
- Requirement Repository
- Quality, Traceability & Linkage Metrics
Writing Effective Requirements

Requirement Specification
SRS
SATC Requirement Analysis Findings

Problems Common To Most Documents
→ Documentation and style standards not used or misapplied
→ Poor organization of information content
→ Uneven emphasis and levels of detail
→ Inconsistent identification schemes
→ Verbose text
→ Poor sentence structure
→ Poor word selection
ADD_Cnode

\[ \Delta \text{HYP} \]

\[ \text{cid?} : \text{CID} \]

\[ \text{key?} : \text{KEY} \]

\[ \text{result!} : \text{RESULT} \]

\[ \text{cid?} \notin \text{cused} \quad \land \quad \text{cused'} = \text{cused} \cup \{ \text{cid?} \rightarrow \text{key?} \} \]

\[ \text{result!} = \text{ok} \]
The objective of the SRS is to DEFINE CAPABILITIES that will satisfy a mission need/problem as ....
SRS Objective

…. seen by all project participants and stakeholders.
Purpose Of SRS

● Contract Between Acquirer & Provider Of Capability
  ◆ Defines what is to be provided
  ◆ Establishes when and how things are to be provided

● Provides the Basis for:
  ◆ Assessing proposed engineering changes
  ◆ Resolution of acquirer/provider disputes
  ◆ Development of test requirements
  ◆ Preliminary user’s manual
  ◆ Maintenance & support planning
SRS Requirements

• Specify constraints on implementation
• Easy to change / maintain
• Serve as reference tool for maintenance
• Characterize acceptable responses to undesirable events
Roles in SRS

- **What should be delivered**
  - Customer
  - Design-to specification
  - Designer
  - Range of acceptable implementation
  - Coders

- **Basis for scheduling and measuring progress**
  - Manager
  - Quality Assurance

- **Basis for validation, test planning, verification**
  - Quality Assurance
Scope Of SRS

Descriptive/Prescriptive/Expectation

Current Systems
- Capabilities
- Organizations
- Personnel
- Logistics
- Maintenance

Future Systems
- Capabilities
- Organizations
- Personnel
- Logistics
- Maintenance

Change Activities
- Acquisition
- Installation
- Transition
- Retirement
### Required Capability Topics

<table>
<thead>
<tr>
<th>1. Interfaces</th>
<th>5. Safety</th>
</tr>
</thead>
<tbody>
<tr>
<td>2. Functional Capabilities</td>
<td>6. Reliability</td>
</tr>
<tr>
<td>3. Performance</td>
<td>7. Security/Privacy</td>
</tr>
<tr>
<td>4. Data Structures/Elements</td>
<td>8. Quality</td>
</tr>
<tr>
<td></td>
<td>9. Constraints &amp; Limitations</td>
</tr>
</tbody>
</table>
Basic Documentation Problems

➔ Structural
  → Organization
  → Relationships
  → Detail

➔ Natural Language (English)
  → Ambiguity
  → Inaccuracy
  → Inconsistency
The Structural Dilemma

Mission Environment

Operational Environment

Support Environment

Development Environment

Topics

DESCRIPTIVE INFORMATION

PREScriptive INFORMATION
Content topics are not isolated subjects. They have multiple and complex relationships. This compounds the structural problem.
Documentation standards establish DIDs as generic design solutions for the problem of structuring information.

- A SRS DID is a high level, generic structure for organizing requirements specifications by predefined subjects.

- Generic design structures must be adapted/tailored to satisfy the needs of a particular project.

- Variations: IEEE, DoD, NASA
SRS DID - IEEE

REQUIREMENTS - IEEE DID-830-1993

1.0 Introduction
  1.1 Purpose
  1.2 Scope
  1.3 Definitions, acronyms, and abbreviations
  1.4 References
  1.5 Overview

2.0 Overall description
  2.1 Product perspective
  2.2 Product functions
  2.3 User characteristics
  2.4 Constraints
  2.5 Assumptions and dependencies

3.0 Specific requirements

Appendices
Index
SOFTWARE REQUIREMENT SPECIFICATION - DI-IPSC-81433

1. Scope
2. Reference Documents
3. Requirements
   3.1 Required states and modes
   3.2 CSCI capability requirements
   3.3 CSCI external interface requirements
   3.4 CSCI internal interface requirements
   3.5 CSCI internal data requirements
   3.6 Adaptation requirements
   3.7 Safety requirements
   3.8 Security & privacy requirements
   3.9 CSCI environment requirements
   3.10 Computer resource requirements
   3.11 Software quality factors
   3.12 Design and Implementation constraints
   3.13 Personnel-related requirements
   3.14 Training-related requirements
   3.15 Logistics-related requirements
   3.16 Other requirements
   3.17 Packaging requirements
   3.18 Precedence and criticality of requirements

5. Requirements Traceability
6. Notes
   A. Appendixes

CSCI = Computer software configuration item
### REQUIREMENTS - NASA DID-P200

1.0  Introduction
2.0  Related documentation
3.0  Requirements approach and tradeoffs
4.0  External interface requirements
5.0  Requirements specification
    5.1  Process and data requirements
    5.2  Performance and quality engineering requirements
    5.3  Safety requirements
    5.4  Security and privacy requirements
    5.5  Implementation constraints
    5.6  Site adaptation
    5.7  Design goals
6.0  Traceability to parent’s design
7.0  Partitioning for phased delivery
8.0  Abbreviations and acronyms
9.0  Glossary
10.0 Notes
11.0 Appendices
Tailor SRS Design To Satisfy Project Needs

Arbitrary grouping of information makes the document difficult to understand and difficult to maintain.

- Group requirements that are part of a single function.
- Address functions with common inputs and outputs within the same area of the document.
- Address functions connected by output-to-input relationships in that order.
- Tie processes that must be accomplished in the same timeframe together.
- Emphasize functional similarities, but ensure that the functional requirements are distinct.
Document Tailoring

Tailoring is adapting the design of the general purpose solution (the Data Item Description) to fit the unique needs of the current documentation problem.

♦ “Stub” sections that don’t apply with “N/A” at highest node and provide or cite reason.

♦ Add new sections at end of appropriate level branch of the document’s structure.

♦ Don’t change the document’s basic identification scheme established by the DID.
Tailoring Example

3.8 Security & Privacy - *N/A*, *An open system with no classified or privacy data.*

3.9 Environmental Requirements ……

3.10 Computer Resources Required
   3.10.1 Hardware……..type, quantity, etc.
   3.10.2 Hardware Utilization ……CPU time, disk capacity, etc.
   3.10.3 Software ………..OS, DBMS, etc.
   3.10.4 Communications ……….Networks, Links, Nodes, etc.
   3.10.5 *Uninterruptable Power Sources …….. Motor/Generators*

3.11 Software quality factors. *TBD by 09/30/97, WM Wilson C300.1*
SRS Quality Attributes

A SRS should be:

1. Complete
2. Consistent
3. Correct
4. Modifiable
5. Concise
6. Testable
7. Traceable
8. Unambiguous
9. Understandable
10. Validatable
11. Verifiable
12. Independent
13. Annotated
14. Appropriate Abstraction Level
A “COMPLETE” requirements specification must precisely define:

- **All** the known real world situations that will be encountered by the prescribed capability
- The capability’s responses to those situations
- Full labels and references to all figures, tables and diagrams

A “COMPLETE” requirements specification must **NOT** include:

- Situations that **will not** be encountered
- Unnecessary capability features.
3.8 Security & Privacy - TBD
3.9 Environmental Requirements - TBD
3.10 Computer Resources Required
   3.10.1 Hardware - Not yet selected
   3.10.2 Hardware Utilization - TBD
   3.10.3 Software ..........OS, DBMS, etc.
   3.10.4 Communications ...Networks, Links, Nodes, etc.
3.11 Software quality factors. - TBD
2 - Consistent

A “CONSISTENT” requirements specification is one where:

- There is no conflict between:
  - Individual statements of required capabilities
  - Individually specified capabilities’ behavioral properties

- Constraints do not adversely impact essential behavioral properties.

Functions and performance levels must be compatible and required quality features (reliability, safety, security, etc..) must not negate the capability’s utility.
An Inconsistent Specification

“3.7 Safety Requirements”

“3.7.4. 1 In the event of a liquid nitrogen (LN) spill, a 30 dB audible alarm shall be activated and continued until launch tower LN sensors return to a null reading.”

“3.13 Personnel Related Requirements”

“3.13. 7 Personnel in the area of the launch tower during tanking operations shall wear hearing protective devices that provide a minimum of 35 dB audio attenuation.”
A “CORRECT” requirements specification must:

- Accurately and precisely identify the individual conditions and limitations of situations that the desired capability will encounter
- Define the capability’s proper response to those situations that will be encountered
3.5.1 The building’s entrance shall be a standard 6.8’ by 2’ 6” doorway.

Should be 3’ 8”
4 - Modifiable

A “MODIFIABLE” requirements specifications is such that changes can be made easily, completely, and consistently.

- Groups related concerns together
- Separates unrelated concerns
- No redundancy
- Express each requirement separately

This attribute is exhibited by a logical organization of specifications based on their relationships.
3.1 The CCS shall ingest and store spacecraft engineering data, both recorded and real-time ….
3.1.1 The CCS shall be able to ingest and store engineering data from new ORU/ORIs.
3.1.2 The CCS shall be able to store converted engineering data received through the common test device interface.
3.1.3 The CCS shall be able to ingest and store onboard computer memory dump data for the life of the mission.
3.1.4 The CCS shall be able to store converted engineering data received from a simulation facility.

3.2 The CCS shall be able to ingest and store onboard computer memory dump data for the life of the mission.
A “CONCISE” requirement specification is a short as possible without adversely affecting any other quality of the SRS.

! Major reductions in SRS size are rarely possible without adversely effecting other qualities.
Baroque Writing

3.1 The check printing function of the payroll system shall provide the capability to validate check amounts.

$$\Rightarrow$$ 3.1 The payroll system shall validate check amounts.

*Concise and Understandable*
A “TESTABLE” requirements specification must state each requirement in such a manner that pass/fail or quantitative assessment criteria can be derived from the specification itself and/or referenced information.

- There must exist a finite cost effective technique to verify each requirement is satisfied by the system.

! If you can’t test it, why request it?
Be Specific

3.1 The system shall be user friendly and fast.

Specification is nonspecific due to the use of vague words. Its implementation cannot be objectively assessed based on the specification.

3.1.1 The system’s functions shall be activated and terminated by menu selections.
3.1.1.1 Functions shall be initiated within 500 \( \mu \) sec. after their selection.

The requirements are specific. The implementation can be directly tested against the specification.
8.1.6 In the case of a reactor melt-down, the system shall reduce the deaths of personnel within a 20 mile radius by at least 80%.

*Cannot test, not worth the cost.*
A “TRACEABLE” requirement specification uniquely identifies each stated requirement.

- Backward traceability to previous stages of development by explicit reference source in earlier documents
- Forward traceability to all documents spawned by SRS with each requirement having a unique name or reference number.

! Number each requirement hierarchically
! Include only one requirement per paragraph
! Use a convention for individual requirements such as “shall”
**Uniquely Identify Each Requirement For Traceability**

**A Gaggle Of Requirements**

“The XYZ system shall provide variance/comparative information that is timely, itemized in sufficient detail so that important individual variances are not hidden because they cancel each other, pinpoints the source of each variance, and indicates the area of investigation that will maximize overall benefits.”

**Better But Not Good (vague words)**

5.1 The XYZ system shall provide variance/comparative information.

5.1.1 Variance/comparative information shall be *timely*.

5.1.2 Variance/comparative information shall be itemized in *sufficient* detail to:

5.1.2.1 Prevent *important* individual variances from being hidden.

5.1.2.2 Pinpoints the source of each variance.

5.1.2.3 Indicate the area of investigation that will *maximize overall* benefits.
8 - Unambiguous

An “UNAMBIGUOUS” requirement statement can only be interpreted one way.

! Natural language is inherently ambiguous. Review by an independent party to identify ambiguities.

! Alternative - requirement specification languages but have long learning curves and few understand.
WHAT WAS WRITTEN:
“The system shall ignore anomalies 20 seconds prior to engine shut down.”

WHAT WAS IMPLEMENTED:
“The system shall clear all anomaly indicators 20 seconds prior to engine shut down.”

WHAT WAS MEANT:
“The system shall ignore any anomaly occurring during the 20 second period immediately prior to engine shut down.”
An “UNDERSTANDABLE” specification’s meaning is easily grasped by all of its intended readers with minimum explanation.

! English is the ‘common denominator’ that is understood by all project participants. Words must be selected with care and the language must be used properly in order for a specification’s intent to be correctly comprehended.

! The larger and more complex the problem addressed by the requirements specification, the more difficult is the task to design a document that aids rather than inhibits understanding.
“Users attempting to access the ABC database shall be reminded by a system message that must be acknowledged and page headings on all reports that the data is sensitive and access is limited by their system privileges.”

3.1 Users attempting to access the ABC database shall be reminded by a system message that data is sensitive and access is limited by their system privileges.

3.1.1 The system data classification message must be acknowledged by the user before access to the ABC database is permitted.”

3.2 Page headings on all reports shall remind users that the data in the report is sensitive and cannot be distributed to unauthorized individuals.
10 - Validatable

A "VALID" requirements specification is substantiated as being true as stated by each individual and organization having a vested interest in the system solution.

To validate a requirements specification all the project participants, managers, engineers and customer representatives, must be able to understand, analyze and accept or approve it.
Valid & Invalid Requirements

Operational requirements for the Boeing 747-200B/VC-25A, USAF tail number 29000.

3.3 In the event of an inflight emergency, the aircraft shall land at the nearest US military, NATO or commercial airfield.

3.4 In the event of a national emergency, the aircraft shall effect inflight transfer of NCA personnel to the Airborne Command Post.
11 - Verifiable

A “VERIFIABLE” requirement specification is consistent with specifications at higher and lower levels of abstraction.

“Verify: to prove to be true or correct by comparison to a standard or reference to ascertainable facts.”

! Ambiguous specifications are not verifiable.
Can You Verify?

— The system shall have a good human interface.
— The program shall never enter an infinite loop.

Output of the program shall be produced within 20s of the event x 60% of the time and shall be produced within 30s of the event x 100% of the time.
An “INDEPENDENT” specification specifies what is to be accomplished, not how it is done.

There should be more than one system design and implementation that correctly implements the requirements.
Freedom of Choice!

The requirement database shall store the requirements using a numerical hierarchical numbering schema such as 1, 1.1, 1.1.1 ...

The requirement database shall store the requirements in such a manner that an exact count of the number of requirements can be electronically extracted.
“ANNOTATED” specifications are easily understood as to the importance (ranked), relative stability, and/or version.

- Accomplished by adding appropriate suffix
- Should be done on all or none of the requirements, not partially completed
Sample Annotations

**Importance** - For budgeting or inclusion

<table>
<thead>
<tr>
<th>Mandatory</th>
<th>(M)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Desirable</td>
<td>(D)</td>
</tr>
<tr>
<td>Optional</td>
<td>(O)</td>
</tr>
<tr>
<td>Essential</td>
<td>(E)</td>
</tr>
<tr>
<td>Conditional</td>
<td>(C)</td>
</tr>
<tr>
<td>Optional</td>
<td>(O)</td>
</tr>
</tbody>
</table>

**Stability** - Where to build in flexibility - Probability to change

<table>
<thead>
<tr>
<th>High</th>
<th>(H)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Medium</td>
<td>(M)</td>
</tr>
<tr>
<td>Low</td>
<td>(L)</td>
</tr>
</tbody>
</table>

**Version** - In database, column for each version, X in column if in version
3.1.4 The XYZ system shall generate reports showing detailed and summary information about the maintenance schedule for:

a. Routine maintenance schedules
b. Non-routine maintenance schedules
c. Upgrade maintenance schedule

Implication may be incorrect, add:

3.1.4.1 Implementation, and operational priority for the schedule reports is in order stated above.
“ABSTRACTION LEVEL” is dependent on the function of the SRS.

It should be specific enough that any system built that satisfies the requirements satisfies all user needs, and abstract enough that all systems that satisfy all user needs also satisfy all requirements.
Levels of Abstraction

1 System shall provide communications.

1.1 System shall provide voice communication.

1.1.1 Telephone system shall provide voice communication

1.1.1.1 Telephone system shall provide local calls, long distance calls, call forward ...

1.1.1.1.1 Telephone shall provide local calls where user hears dial tone within 3 seconds of lifting receiver ...
Not in SRS

- Cost
- Delivery Schedule
- Report Procedures
- Software Development Methods
- Quality Assurance
- Validation and Verification Criteria
- Acceptance Procedures

SATC NASA
Requirement vs. Design Implementation

Needs vs. Implementation

Dangers:
Forcing a design when not intended
Believing all requirements are covered when not
Needs vs. Implementation

Requirement: “The contractor shall provide a database for requirement management.”

What was meant:

*Provide the capability for tracing between requirements.*

*Provide the ability to add requirement attributes.*

*Provide the ability to sort requirements.*

Problem - May need a requirement management tool, not a database as specified.
Solution -

State what is needed not how it is to provided.

Ask **WHY** the requirement is needed.
If this does not lead to “real” requirement, then probably appropriate as stated.
Requirement Characteristics

1 - **Type** - Identifies the source and contractual applicability

2 - **Application** - Identifies the object of a requirements

3 - **Categorization** - Identifies purpose of requirement

4 - **Compliance Level** - Identifies the depth of compliance mandated for a requirement

5 - **Priority** - Identifies the relative importance of a requirement in terms of implementation or sequence of testing
1 - Type

Primary - Usually a requirement levied on a contractor / producer under force of contract

“The payload shall be transported into orbit in the payload bay.”

Derived - Requirements that are generated apart from the primary requirements; if not primary than is derived.

“The payload shall have a diameter of less than 14 feet.”
2 - Application

**Product parameter** - Requirement that applies the product or service to be developed

“The external surfaces of all equipment shall be white.”

• Qualitative - Contains no measurable requirement
  “The mixer shall produce a mixture of homogeneous appearance.”

• Quantitative - Measurable requirement
  “The mixer shall produce a mixture of \( x \) granularity in five minutes.”
Program parameter - Requirement that applies to the activities associated with enabling the creation of a product or service.

“The contractor shall develop a concept of operations.”

• Task - Identifies an analysis or other effort to be performed

  “Prepare a systems management plan.”

• Compliance evaluation - Identifies the methodology for measuring compliance

  “NASA DID P200 shall be used for requirement specification.”

• Regulatory - Identifies administrative elements

  “Deliverable data shall be furnished with unlimited rights to the government.”
3 - Categorization

Functional - describes system service or function

Non-function - constraint placed on system or development process (response time or language standard)
<table>
<thead>
<tr>
<th>Category</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Input</td>
<td>receive EDI data</td>
</tr>
<tr>
<td>Output</td>
<td>export a particular format</td>
</tr>
<tr>
<td>Reliability</td>
<td>mean time to failure</td>
</tr>
<tr>
<td>Availability</td>
<td>expected hours of operation</td>
</tr>
<tr>
<td>Maintainability</td>
<td>ease with which components can be replaced</td>
</tr>
<tr>
<td>Performance</td>
<td>response time</td>
</tr>
<tr>
<td>Environmental</td>
<td>dust levels that must be maintained</td>
</tr>
<tr>
<td>Ergonomic</td>
<td>colors to minimize eye strain</td>
</tr>
<tr>
<td>Safety</td>
<td>limits for radiation</td>
</tr>
<tr>
<td>Standards</td>
<td>conform to ASME codes</td>
</tr>
</tbody>
</table>
# 3 - Categorization (cont.)

<table>
<thead>
<tr>
<th>Category</th>
<th>Requirement</th>
</tr>
</thead>
<tbody>
<tr>
<td>Security</td>
<td>authorization of users</td>
</tr>
<tr>
<td>Facility</td>
<td>temperature requirements</td>
</tr>
<tr>
<td>Transportability</td>
<td>weight limits</td>
</tr>
<tr>
<td>Training</td>
<td>number and length of tutorials</td>
</tr>
<tr>
<td>Documentation</td>
<td>on-line help</td>
</tr>
<tr>
<td>External interfaces</td>
<td>industry standard communication</td>
</tr>
<tr>
<td>Testing</td>
<td>remote diagnostics</td>
</tr>
<tr>
<td>Quality provisions</td>
<td>calibration intervals</td>
</tr>
<tr>
<td>Conversion</td>
<td>accept data from older system versions</td>
</tr>
<tr>
<td>Growth</td>
<td>will support an additional number of users</td>
</tr>
<tr>
<td>Installation</td>
<td>ability to put new system into use</td>
</tr>
</tbody>
</table>
4 - Compliance Level

**Mandatory** - Typically contains “shall” - mandates conformance

**Guidance** - Typically a “will” statement - accomplishment is desired/preferred

**Information** - Supporting or giving insight into a measurable requirement; non-binding
5 - Priority

Values of priority will be dependent on program and company needs

High / Medium / Low

Catastrophic / Critical / Marginal / Negligible
Requirement Specification Statements
Formulating Requirements Specification Statements

1. Perspective And Selection of Imperatives
2. Sentence Structure
3. Words and Phrases To Avoid
4. Use Of Examples And References
5. Use Of Tables And Charts
ELEMENTS of a specification are:

1. Entity
2. Action
3. Event
4. Condition
5. Target
6. Object
7. Constraint
8. Localization
Specification Statement Structure

Specification Structure:

[Localization] [Actor] [Action] [Target] [Constraint]

Example:
7.1 When three or more star trackers lose reference stars, the spacecraft shall immediately align its main axis on the Earth-Sun line unless the optical instrument’s cover is open.

Localization: “When three or more star trackers lose reference stars”
Actor: “spacecraft”
Action: “align”
Target: “main axis”
Constraint: “unless the optical instrument’s cover is open”
Perspective And Selection of Imperatives

**Imperatives** are those words and phrases that command that something must be provided.

*Shall* is usually used to dictate the provision of a functional capability.

*Must/must not* is used to establish performance requirements or constraints.

*Are applicable* is used to include, by reference, standards or other documentation as an addition to the requirements being specified.

*Responsible for* is used in requirements documents that are written for systems whose architectures are predefined.

*Will* is used to cite things that the operational or development environment are to provide to the capability being specified.

*Is required to* passive voice, *Should* is advisory. Neither should be used in requirement specification statements.
Words And Phrases To Avoid

1. Weak Phrases
2. Options
3. Generalities
Weak Phrases are clauses that are apt to cause uncertainty and leave room for multiple interpretations.

Phrases such as “adequate”, “as appropriate” and “timely” indicate that what is required is either defined elsewhere or, worse, that the requirement is open to subjective interpretation.

Phrases such as “but not limited to”, “as a minimum”, and “TBD” provide a basis for expanding a requirement or adding future requirements.
Options are words such as "may" and "optionally", that give the developer latitude in satisfying the specification statements that contain them.

! Options loosen the specification,
! Reduces the acquirer’s control over the final product, and
! Establishes a basis for possible cost and schedule risks.
Generalities provide gross quantitative or qualitative descriptors that indicate direction of intent but no useful information.

“About”
“Almost”
“Bad”
“Close”
“Good”
“Many”
“Most”
“Timely”
Use the most simple word appropriate to the intent.

**Hide** “is to put out of sight”

**Obscure** “is lacking light or dim”

**Circumference** “is the line bounding a circular area”

**Perimeter** “is a line bounding an area of any shape”

**Periphery** “is the boundary of a solid object”
Using Examples

- Immediately follow what is to be illustrated with the example.
  
  \textit{Attention spans are short and shrinking!}

- Repeat an example if it is not located on the same page as its second or later use.
  
  \textit{It’s better to be repetitive than to divert the reader’s attention!}

- Ensure that the example is not mistaken for part of the specification through the use of italics, quotes, or being explicit.

  \textit{For example: “This is an example.”}
References

- Identify all external documents in the SRS section designated for that purpose. For DI-IPSC-81433 and NASA DID-P200, this is Section 2.

- Identify each reference cited with a unique number or identifier, such as “2.14.”

- Cite references by short or common title, full title, version or release designator, date, publisher or source, and document number or other unique document identifier. For example:

- Use the unique citation identifier when referencing information in that document. For example: “. . . as defined by Section 3.1 of reference document 2.14, NMI 2410.10.”
Tables And Charts

- Title and identify each table and chart with a unique identifier.

- List each table and chart in the SRS’s table of contents by title, unique identifier, and page number.
  
  *Help the reader find it!*

- Identify the purpose of the table or chart in the text immediately preceding it.
  
  *No surprises!*

- Explain each aspect or element of the table or chart (columns, rows, symbols, blanks, etc.) from right to left then top to bottom.
  
  *No puzzles!*

---

SATC NASA
3.4.5 Each software module shall be assigned to one of the criticality categories identified by the first column of Table 3-2. The second column identifies the criticality criteria for each category. The third column establishes the level of failure tolerance that shall be provided by each module assigned to each category. Column four establishes the minimum redundancy level requirement for each module assigned to each category.

<table>
<thead>
<tr>
<th>CRITICALITY CATEGORY</th>
<th>FUNCTIONAL CRITICALITY</th>
<th>FAILURE TOLERANCE</th>
<th>FUNCTIONAL REDUNDANCY</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Crew safety</td>
<td>2 failures</td>
<td>3 minimum</td>
</tr>
<tr>
<td>2</td>
<td>Critical mission support</td>
<td>1 failure</td>
<td>2 minimum</td>
</tr>
<tr>
<td>3</td>
<td>Noncritical functions</td>
<td>0 failures</td>
<td>1 minimum</td>
</tr>
</tbody>
</table>

Table 3-2. Functional Redundancy Levels
The SRS is an item of software. Ensure that:

- Projects requirements for SRS document are defined.
- SRS structure is tailored to satisfy its requirements.
- The document and individual specifications exhibit all desirable documentation quality characteristics.
- All topics are addressed in the SRS.
- Individual specifications:
  - Are logically structured and simply stated.
  - Stated with the imperative, words and phrases that are appropriate to the intended meaning.
Levels of Requirement Specifications

Requirement Specification

Requirement Specification

Requirement Specification
2 - Level Requirements

Level 1 - Requirement Definition
High level abstract description in natural language of what services system is expected to provide, constraints under which it must operate. Use customer provided information.

Level 2 - Requirement Specification
Detailed description of what system must do. Sets out system services in detail; precise. May serve as contract between system buyer and software developer.
Example - 2 Level

Requirement Definition:
1. The software must provide a means of representing and accessing external files created by other tools.

Requirement Specification:
1.1 The user must be provided with facilities to define the type of external files
1.2 Each external file type must have an associated tool which may be applied to the file
1.3 Each external file type must be represented as a specific icon on the user’s display.

...
NASA’s Levels of Requirement

**Level 1: Mission-level**
Very high level
Rarely, if ever, change.

**Level 2: Allocated**
Change should be minimal.
Project’s development started

**Level 3: Derived**
Contracts are bid
Acceptance Tests

**Level 4: Detailed**
Used to design and code the system.
System Tests

Levels of Requirement Detail

- **Level 1**: Mission
  - Rarely if ever changed

- **Level 2**: Allocated
  - Minimal changes
  - 1 to Many Bi-directional Linkage

- **Level 3**: Derived
  - Contract bids

- **Level 4**: Detailed
  - Design & Code

- **Acceptance Test**

- **System Test**
Verification Of Specifications Across Documents

CONCEPT DOCUMENT

3.1.1 Solar power shall be the main source of all subsystems electricity.
3.1.2 When the sun is eclipsed, electricity shall be provided by batteries.

SYSTEM REQUIREMENTS DOCUMENT

3.1.2.3 Combined weight of primary and secondary batteries shall not exceed 500 pounds.
3.1.2.4 Secondary S/C batteries shall be charged in parallel with powered subsystems during S/C daylight.

DESIGN DOCUMENT

3.1.2.3.1 Sodium sulfur (NaS) batteries shall be used as the S/C’s secondary batteries.
Structure Level at Which Requirement Occurs

Expected

Level 3 - Derived

Level 4 - Detailed

High Level Detail

Lower Levels of Detail

Level ==>  
1  
1.1  
1.1.1 ...
Requirement Management
Requirement Management Process

- The process is goal directed and aimed at the production of a set of requirements
- The system boundaries are defined
- All requirements are solicited, fairly evaluated, and documented
- Requirements are specified as capabilities and that qualifying conditions and bounding constraints are identified distinctly from capabilities
- Requirements are validated, or purged if invalid, from the requirement set
- Consideration is given to consistency when many authors
- The developing set is understood, at the appropriate level of detail, by all individuals
Development Process

Analysis
Identify Requirements

Requirement Collection

Definition
Build Well-formed Requirements

Specification
Organize Requirements

Present Requirements

Customer

Environment

Technical

Requirement Collection

Requirement Repository

Customer

Technical community
Development Activities

**Analysis** - Derive system requirement prototypes if necessary

**Definition** - Translate requirement information into well-formed statements to accurately reflect what customer wants. Written so end user and customer can understand.

**Specification** - Detailed and precise description of system requirements to act as basis for contract between customer and developer
Important Concepts

- Use a requirement management tool or database as a repository to store requirements
- Design the structure of the repository carefully to fit the data not the management structure
- Consider the metrics to be collected in the design of the repository
- Start the metrics program with the initial requirement specification
Requirement Repository
Requirement Repository

- Chosen prior to start of project
- Purpose - how requirements will be prepared, tracked, traced, measured and tested
- Choice - what capabilities are needed; flexibility, etc.

<table>
<thead>
<tr>
<th>Word Processor</th>
<th>Spreadsheet</th>
<th>Relational Database</th>
<th>Requirement Tool</th>
</tr>
</thead>
<tbody>
<tr>
<td>Word</td>
<td>Excel</td>
<td>Access</td>
<td>RTM</td>
</tr>
<tr>
<td>Word Perfect</td>
<td>Lotus</td>
<td>Dbase</td>
<td>DOORS</td>
</tr>
<tr>
<td></td>
<td>Quattro Pro</td>
<td>Oracle</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Sybase</td>
<td></td>
</tr>
</tbody>
</table>
Requirement Allocation & Traceability

Specification
- 1a
- 1b

Design
- 15x
- 15y
- 15z

Release 1
- 1a

Release 2
- 1a, 1b (added functionality)
  (Regression testing)

Testing - System
- Test 23

Testing - Acceptance
- Test 114
Test Coverage

- **Test A**
  - Requirement 1
  - Requirement 2
  - Requirement 3
  - Requirement 4
  - Requirement 5
  - Requirement 6

- **Test B**
  - Requirement 1
  - Requirement 2
  - Requirement 3
  - Requirement 4
  - Requirement 5

**Tested by 1 Test**

**Tested by >1 Test**

**Untested - PROBLEM**

*Sample Linkage*
Requirement Simplification

The box shall provide protection to its contents and form a barrier to unwanted traffic.

- The box shall provide protection to its contents.
- The box shall form a barrier to unwanted traffic.
Requirement
Decomposition & Simplification

Req10
  Req10a
  Req10b

Req20
  New
  Delete
  Test 23
  (attribute)

Req30
  Delete

Req20a

Req20b

Duplicate 1b - Delete
Requirement Expansion & Decomposition

Expansion

ReqX

Bell

ReqY

Button

Decibel

Length

Size

Color

Bell
## Requirement Repository - Project Requirements

<table>
<thead>
<tr>
<th></th>
<th>Word Processor</th>
<th>Spreadsheet Processor</th>
<th>Relational Database</th>
<th>Requirement Tool</th>
</tr>
</thead>
<tbody>
<tr>
<td>Document config. mgt</td>
<td>X</td>
<td></td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>Document preparation</td>
<td>X</td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>Function decomposition</td>
<td></td>
<td>X</td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>Metrics</td>
<td></td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>Report preparation</td>
<td></td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>Requirement allocation</td>
<td></td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>Requirement config. mgt</td>
<td></td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>Requirement expansion</td>
<td></td>
<td></td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>Requirement importation</td>
<td></td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>Requirement simplification</td>
<td></td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>Requirement storage</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>Requirement traceability</td>
<td></td>
<td></td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>Test coverage/adequacy</td>
<td></td>
<td></td>
<td>X</td>
<td>X</td>
</tr>
</tbody>
</table>
## Requirement Repository-Database Capabilities

<table>
<thead>
<tr>
<th>Word Processor</th>
<th>Spreadsheet</th>
<th>Relational Database</th>
<th>Requirement Tool</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ad Hoc querying</td>
<td>X</td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>Classification by attribute</td>
<td>X</td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>Classification by linkage</td>
<td>X</td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>Decomposition Inheritance</td>
<td></td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>Historical comparisons</td>
<td></td>
<td></td>
<td>X</td>
</tr>
</tbody>
</table>
## Requirement Repository-Metric Capabilities

<table>
<thead>
<tr>
<th></th>
<th>Word Processor</th>
<th>Spreadsheet</th>
<th>Relational Database</th>
<th>Requirement Tool</th>
</tr>
</thead>
<tbody>
<tr>
<td>Document size</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Dynamic changes over time</td>
<td></td>
<td>X</td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>Release size</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>Requirement expansion profile</td>
<td></td>
<td></td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>Requirement types</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>Requirement verification</td>
<td></td>
<td></td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>Requirement volatility</td>
<td></td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>Test coverage</td>
<td></td>
<td></td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>Test span</td>
<td></td>
<td></td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>Test types</td>
<td></td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
</tbody>
</table>
Class Design $\iff$ Link Design

A requirement in Class Level 3 with appropriate decomposition

Minimal Linkage
A requirement in Class Level 3 with pseudo-decomposition

Level 3: 1,500 requirements with 6-30 subdivisions

Level 4: 6,000 requirements

==> 19,000 links between Level 3 and Level 4

Complex Linkage
Requirements Repository Summary

Requirement repository tool needs to be chosen with care based on project requirements

Need effective dialogue between requirement engineers and database engineers

Tools with more capabilities require more resources

Resources needed include training and learning curve, design and set up, and maintenance
Requirement Components

Quality, Traceability & Linkage Metrics
Requirement Quality Attributes

- **Ambiguity** - Requirements with potential multiple meanings.
- **Completeness** - Items left to be specified
- **Traceability** - The traceability of the requirements upward to higher level documents and downward to code and tests.
- **Understandability** - the readability of the document.
- **Requirement Volatility** - The rate and time within the life cycle changes are made to the requirements.
Requirement Metrics

- **Ambiguity** = Weak Phrases (adequate, as appropriate, as applicable, but not limited to, normal, if practical, timely, as a minimum) + Options (can, may, optionally)
- **Completeness** = TBD + TBA + TBS + TBR
- **Understandability** = Numbering Scheme
- **Traceability** = Number of Items traced to tests, between builds, between levels of detail
- **Volutility** = Number of Changes / Number of Requirements

**Number of Requirements**: = Imperatives (shall, must, will, required, responsible for, should, are to, are applicable) + Continuances (below:, as follows:, following:, listed:, in particular, support:, : )
Metric Sources

- Requirement Metrics
  - Requirement Specification
  - Requirement Repository
Automated Requirement Measurement Tool (ARM)

Available free: http://satc.gsfc.nasa.gov
Basic Underlying Documentation Problems

- Structural
  - Organization
  - Relationships
  - Detail

- Language
  - Ambiguity
  - Inaccuracy
  - Inconsistency

- Careless Prose
Automated Requirement Measurement Tool (ARM)

Objective - Provide measures that can be used to evaluate the quality of a requirements specification document.*

- Available early in the life cycle
- Simple to use
- Easy to understand output
- Identify specific requirement weaknesses (structure and language)
- Indicator of specification areas that can be strengthened
- Basis for estimating required resources

*Not “Did we write the right requirement?” But “Did we write the requirements right?”
Automated Requirements Analysis (ARM)*

- Established database containing 46 requirements documents
- Developed ARM Tool
  - Full text scan
  - Counts:
    - Lines of text
    - Specification statements
    - Unique specification subjects
    - Quality Indicators
    - Levels of document structure
    - Specifications at each level
  - By finding:
    - Sentences & Section titles
    - Sentences with imperatives
    - Words preceding imperatives
    - Special words & phrases
    - Section identifiers
    - Identifiers & imperatives
- Produces summary & detail reports
- Compare Hand counts to automated counts

*Available at no cost: http://satc.gsfc.nasa.gov
## ARM Analysis of Project X

<table>
<thead>
<tr>
<th>56 DOCUMENT</th>
<th>Lines of Text - Count of the physical lines of text</th>
<th>Imperatives - shall, must, will, should, is required to, are applicable, responsible for</th>
<th>Continuances - as follows, following, listed, in particular, support</th>
<th>Directives - figure, table, for example, note:</th>
<th>Weak Phrases - adequate, as applicable, as appropriate, as a minimum, be able to, be capable, easy, effective, not limited to, if practical</th>
<th>Incomplete (TBD, TBS)</th>
<th>Options - can, may, optionally</th>
</tr>
</thead>
<tbody>
<tr>
<td>Minimum</td>
<td>143</td>
<td>25</td>
<td>15</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Median</td>
<td>2,265</td>
<td>382</td>
<td>183</td>
<td>21</td>
<td>37</td>
<td>7</td>
<td>27</td>
</tr>
<tr>
<td>Average</td>
<td>4,772</td>
<td>682</td>
<td>423</td>
<td>49</td>
<td>70</td>
<td>25</td>
<td>63</td>
</tr>
<tr>
<td>Maximum</td>
<td>28,459</td>
<td>3,896</td>
<td>118</td>
<td>224</td>
<td>4</td>
<td>32</td>
<td>130</td>
</tr>
<tr>
<td>Stdev</td>
<td>759</td>
<td>156</td>
<td>99</td>
<td>12</td>
<td>21</td>
<td>20</td>
<td>39</td>
</tr>
<tr>
<td>Project X</td>
<td>34,664</td>
<td>1,176</td>
<td>714</td>
<td>873</td>
<td>13</td>
<td>480</td>
<td>187</td>
</tr>
</tbody>
</table>
Project X Evaluation

Document Attributes by Standard Deviation

Document Normalized Attribute Comparison

Normalized by Lines of Text
Structure Level at Which Imperative Occurs

<table>
<thead>
<tr>
<th>Structure Level</th>
<th>Expected</th>
<th>Actual</th>
</tr>
</thead>
<tbody>
<tr>
<td>Derived</td>
<td><img src="image1" alt="Derived Graph" /></td>
<td><img src="image2" alt="Derived Graph" /></td>
</tr>
<tr>
<td>Detailed</td>
<td><img src="image3" alt="Detailed Graph" /></td>
<td><img src="image4" alt="Detailed Graph" /></td>
</tr>
</tbody>
</table>
## Project Z - 2 Specification Levels

<table>
<thead>
<tr>
<th>DOCUMENT</th>
<th>LINES OF TEXT - Count of the physical lines of text</th>
<th>Imperatives - shall, must, will, should, is required to, are applicable, responsible for</th>
<th>Continuances - as follows, listed, in particular, support</th>
<th>Directives - figure, table, for example, note:</th>
<th>Weak Phrases - adequate, as applicable, as a minimum, be able to, be capable, easy, effective, not</th>
<th>Incomplete - TBD, TBS, TBR</th>
<th>Options - can, may, optionally</th>
</tr>
</thead>
<tbody>
<tr>
<td>Average</td>
<td>4772</td>
<td>682</td>
<td>423</td>
<td>49</td>
<td>70</td>
<td>25</td>
<td>63</td>
</tr>
<tr>
<td>Level 3</td>
<td>1011</td>
<td>588</td>
<td>577</td>
<td>10</td>
<td>242</td>
<td>1</td>
<td>5</td>
</tr>
<tr>
<td>Level 4</td>
<td>1432</td>
<td>917</td>
<td>289</td>
<td>9</td>
<td>393</td>
<td>2</td>
<td>2</td>
</tr>
</tbody>
</table>

### Problem
- Excessive weak phrases - not testable

![Graph showing distribution of imperatives, continuances, directives, weak phrases, and options across levels 3 and 4]
ARM INCOMPLETE REPORT FOR FILE .txt
TBD # 1: In Line No. 86, ParNo. a., @ Depth 1
a. CERES: 2 in building 1250, LaRC; 2 in building TBD, LaRC; 1 at SAIC; 1 at building 1300; 1 each at 2 other buildings TBD, LaRC

TBD # 2: In Line No. 86, ParNo. a., @ Depth 1
a. CERES: 2 in building 1250, LaRC; 2 in building TBD, LaRC; 1 at SAIC; 1 at building 1300; 1 each at 2 other buildings TBD, LaRC

ARM WEAK PHRASE REPORT FOR FILE ProjectA.txt
provide for # 1: In Line No. 65, ParNo. d., @ Depth 1
F-FOS-00490 The ProjectA shall PROVIDE FOR security safeguards to cover unscheduled system shutdown (aborts) and subsequent restarts, as well as for scheduled system shutdown and operational startup.
Requirement Metrics
Requirement Volatility

Combination of BOTH views indicate risk area - requirements are NOT YET stable
Requirement Volatility

![Bar chart showing requirement volatility over different releases and months.](image)
Requirement Stabilization By Build

(Build 2)

(Build 3)

(Number of Requirements)

(Detailed L4)
Requirement Traceability

Derived (L3) to Detailed (L4)

- **Build 1:**
  - Total: 155
  - Linked: 78%
  - Unlinked: 22%

- **Build 2:**
  - Total: 1049
  - Linked: 84%
  - Unlinked: 16%

- **Build 3:**
  - Total: 1638
  - Linked: 84%
  - Unlinked: 16%
Requirement Expansion

![Requirement Expansion Graph]

- **Specification**
  - Mar-96
  - Sep-96
  - Nov-96

- **Design**
  - Mar-96
  - Sep-96
  - Nov-96

- **Releases**
  - Release 1
  - Release 2
  - Release 3

- **No. Req**
  - 0
  - 500
  - 1000
  - 1500
  - 2000
  - 2500
  - 3000
  - 3500

**Timeline**
- Mar-96
- Sep-96
- Nov-96

**Comparison**
- Specification vs. Design
- Releases 1, 2, and 3
Requirement Decomposition

Derived (L3) to Detailed (L4) Requirements

# Design Requirements

# Specification Requirements

- 1 Design req. supports 640 Specification req
- .84 Design req supports 2 Specification req

Expected

Slide. 140
Requirement Verification - Trace to Test

Derived (L3) Requirements

Detailed (L4) Requirements
Test Span
System test Profile (CDR)

445 requirements are each tested by only 1 test
Test Comprehensiveness

Too many tests verifying only one requirement?

52 tests each verifying only two requirements

43 requirements tested by one test

Sufficient Testing?
Metrics Summary

Requirement metrics assist in identifying potential project risks

Multiple metrics are needed for comprehensive evaluation

Evaluation of requirement text can yield risk information very early in the life cycle

Metric collection is cheaper, faster and more reliable with requirement management tools

Detailed design of database schema necessary for effective evaluation
Use of natural language for requirements may result in problems later - need care, attention and review to language usage and structure.

Repository for requirements can provide benefits in management of project if you use correct tool.

Metrics can be used to track requirements process and give valuable insight into project status and early warning of problems.

IEEE Recommended Practice for Software Requirement Specifications, 830-1993
