Implementing Structured Requirements to Improve Requirements Quality

Dr. Ron Carson, Fellow, International Council on Systems Engineering
ronald.s.carson@boeing.com

We have implemented a closed-loop feedback control process for requirements development that couples measures of requirements quality to the process of writing requirements.
Using structured requirements

- The problem: ambiguous requirements
- The solutions:
  - Agile SE – user stories; avoid requirements
  - Formal Language – use mathematics
    - Structured, natural language
- This approach helps our requirements engineers write more unambiguous and verifiable requirements as required by MIL-STD-961 and related commercial standards*.

From MIL-STD-961E, 5.8:
- a. Each requirement shall be stated in such a way that an objective verification can be defined for it.
- b. …
- c. Only requirements that are necessary, measurable, achievable, and verifiable shall be included.
- d. Requirements shall be worded to provide a definitive basis for acceptance or rejection.
- e. …
- f. Requirements shall be worded such that each paragraph only addresses one requirement or topic.

*ARP4754A, ISO/IEC 29148:2011
Graphics from US Patent #8,732,109
Structured Requirements Syntax – Decomposition for Engineered Requirements

The Basic Structure:
- The agent shall what, how well, under what conditions.

Agent is the product or service entity which has the required characteristic or performs the intended function, e.g., a system or element thereof.

Shall identifies the statement as a mandatory characteristic – a requirement.

What is the function that describes what the agent does that is observable at its boundary, or another mandatory characteristic or attribute of the agent (e.g., size, color).

How well is the measurable characteristic of the function or a design attribute. This is the performance attribute, and includes timing of the function.

Under what conditions addresses two specific considerations
- Conditions are the modes, states or environmental conditions that are present when the agent performs its function or has the stated property
- Inputs are the triggering or initiating events, observable at the boundary, that cause the agent to perform the function
Limiting the Types

- Boeing has identified four types of specification requirements (plus a verification requirement type – not addressed here).
  - Functional/Performance
  - Design
  - Environmental
  - Suitability

---

SELECT THE STATEMENT BELOW THAT REFLECTS THE INTENT OF THE NEW REQUIREMENT

- Functional/Performance Requirement
  - To define the functional behavior for an agent, i.e. something the agent does
  - To define the performance of an agent, i.e. how well the agent does something
  - To identify the interface with another agent, i.e. how the agent interacts with another agent
Elements vs. Types of Requirements

- Each type of requirement has a standard grammar: a set of mandatory and optional elements that ensure verifiability related to the type.

- Functional/Performance - The AGENT shall FUNCTION in accordance with INTERFACE-OUTPUT with PERFORMANCE [and TIMING upon EVENT TRIGGER in accordance with INTERFACE-INPUT] while in CONDITION.

- Design - The AGENT shall exhibit DESIGN CONSTRAINTS [in accordance with PERFORMANCE while in CONDITION].

- Environmental - The AGENT shall exhibit CHARACTERISTIC during/after exposure to ENVIRONMENT [for EXPOSURE DURATION].

- Suitability - The AGENT shall exhibit CHARACTERISTIC with PERFORMANCE while CONDITION [for CONDITION DURATION].
Other Approaches to Patterns


INCOSE “Guide for Writing Requirements”, 5.4.1, 2012

The <subject clause> shall <action verb clause> <object clause> <optional qualifying clause>, when <condition clause>.

EXAMPLE: When signal x is received [Condition], the system [Subject] shall set [Action] the signal x-received bit [Object] within 2 seconds [Constraint].
Quality measures for requirements address each element of the individual type, and the average. Risk is identified based on how well the an instance conforms to the template.

A “graded” (0 to 4) vs. “binary” score is used to clarify required improvements and residual risk, “based on whether or not content is missing in elements and whether or not the content that is present in elements is correct for the identified type of requirement.” (US Patent #8,732,109)
Implementing Functional/Performance Requirement – Closed-loop Improvement

This combinational approach has been implemented in requirements management tools to improve productivity and quality of the requirements.

1. User selects a requirement type: “Functional/Performance” – current requirement and required pattern are displayed
2. Allow user to add/replace text
3. “Show Requirement” displays the concatenated result
4. Preliminary “RQ” scoring and rationale displayed
5. User updates as necessary
6. Select “Save” when done
Benefits: Return on Investment

- We are seeing SE cost avoidance as the process is used to improve the quality of requirements.
  - Weighted averages over nine specifications: $\Delta RQM=1.0$; Requirements rework cost avoided: 44%
Summary

We have implemented a closed-loop feedback control process and tools for requirements development that couples measures of requirements quality to the process of writing requirements (US Patent #8,732,109)

- Identified four broad types of requirements with individual patterns
- Augments structured, natural language requirements with a multi-level quality measurement for the elements of the structured requirements.
- Implemented in DOORS®, Excel® and Teamcenter for Systems Engineering®

Benefit is realized immediately in identifying requirements deficiencies while writing each requirement

Earlier identification of ambiguous and unverifiable requirements reduces program risk and yields cost avoidance compared with later discovery