Requirements Writing Cheat Sheet

A requirement is a need specified using exacting language to avoid any ambiguity or confusion. Requirements can be abstract (i.e. "fuzzy") or exact. If the requirement is abstract, additional derived requirements are added until there is enough information to define one or more test cases. Since it is impossible to totally bound an abstract requirement, the complete set of derived requirements needed to successfully define the set of all possible test cases are said to "satisfice" the abstract requirement. Again, the acid test as to whether a requirement is concrete is the ability to specify all necessary test cases to ensure that the requirement has been satisfied (in the eyes of the stakeholders).
When the requirement levels get too low for display in a requirements diagram, they are shown in a requirements

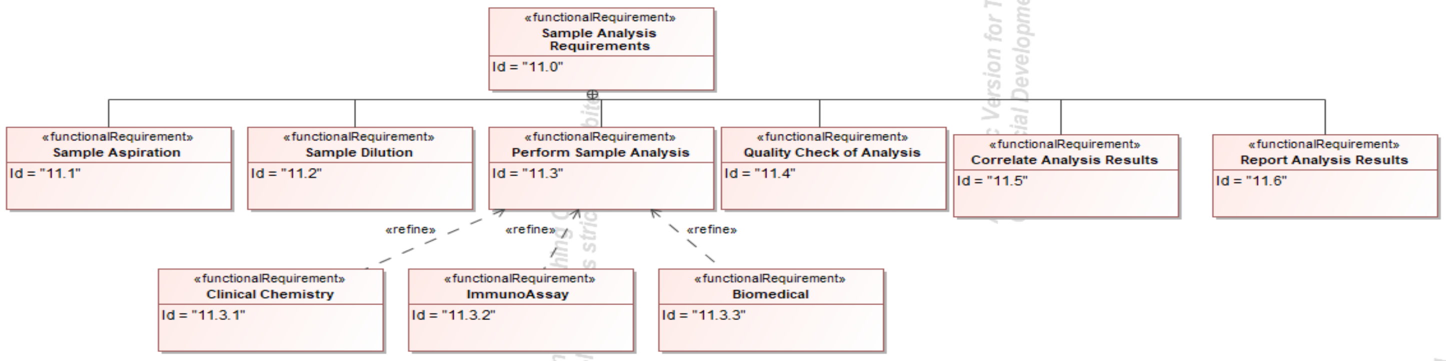
«functionalRequirement» Sample Analysis Requirements Id = "11.0"
«functionalRequirement» Sample Aspiration Id = "11.1"
«functionalRequirement» Sample Dilution Id = "11.2"
«functionalRequirement» Perform Sample Analysis Id = "11.3"
«functionalRequirement» Quality Check of Analysis Id = "11.4"
«functionalRequirement» Correlate Analysis Results Id = "11.5"
«functionalRequirement» Report Analysis Results Id = "11.6"
«refine»
«refine»
«refine»
«functionalRequirement» Clinical Chemistry Id = "11.3.1"
«functionalRequirement» ImmunoAssay Id = "11.3.2"
«functionalRequirement» Biomedical Id = "11.3.3"

Figure 1 Abstract and derived concrete requirements

database, sometimes displayed in a spreadsheet. The most common fields are:

- Requirement Id
- Requirement title – short title of the requirement
- Requirement body text – full description of the requirement, sufficient for creating test cases (if not abstract)
- Priority – used to determine cutoff and schedule information
- Stability – likelihood of the requirement changing
- Author(s) – which stakeholder or analyst wrote the requirement
- Trace(s) – the authority document from which the requirement derives
- Other – Project specific fields as required including keywords, risks and mitigations, feasibility, approval(s), etc.

Id	Status	Title	Requirement Body	Stability	Feasibility
10.1.3	Reviewed	Whole Blood Processing	The System shall support whole blood sample processing from the uncapped primary collection container without requiring the operator to manually pre-treat the whole blood sample.	Stable	Yes
10.1.2	Reviewed	Pre-Dilution	The CC Module shall support predilution of the sample using System diluent.	Change pending	Yes
10.1.3	Reviewed	System HIL Detection	The System shall support user configurable detection of hemolysis, icterus and lipemia in samples. Constraints: The HIL detection shall take place on the CC module.	Change pending	Yes
11.3.1.1	Reviewed	Assay Technology	The CC module shall support photometry and potentiometry.	Likelihood of change 40%	
11.3.1.1	Reviewed	Assay Definition	The CC Module shall utilize existing chemistry assays (from either ADVIA chemistry, Dimension chemistry or business defined third party supplier).		Yes
11.3.1.2	Reviewed	Assay Technology	The System shall support heterogeneous methodologies utilizing the Acridinium Ester Chemiluminescence technology.		Pending analysis

Rules for Writing Good Requirements

- ✓ Write concise requirements
- ✓ Avoid words such as “should”, “could”, “might”, and “may”
- ✓ Use verb conventions:
    - Shall = what the system is required to do.
    - Will = about the domain after the system is deployed in it.
- ✓ Keep the requirement short; rationale should be in the description/comment field
- ✓ Avoid designing in the requirements
- ✓ Do not speculate
- ✓ Do not express possibilities
- ✓ Do not think wishfully
- ✓ Avoid word-sense ambiguity (riverbank vs. money bank)
- ✓ Avoid generalities – all and plural, e.g., "all", "always", "never", "none", “everyone”, “each person”, “everything”.
- ✓ Avoid synonyms
- ✓ Avoid ambiguous constructs, e.g., "only" and "also"
- ✓ Avoid using "not"
- ✓ Avoid ambiguous conjunctions – "and" and "or" combinations
- ✓ Eliminate vagueness
- ✓ Avoid unclear pronouns
- ✓ Avoid stand-alone "this"
- ✓ Write requirements in an active voice
- ✓ Avoid dangerous parentheses
- ✓ Do not use "/"
- ✓ Do not use "etc."
- ✓ Do not use "TBD"
- ✓ Define all technical terms and acronyms used in a glossary
- ✓ Use a well-defined set of keywords for later searching
- ✓ Review a requirement set to ensure there are no conflicts
- ✓ System centric, not user centric, e.g. “the system shall, NOT, “the user shall”