Establishing Metrics to Encourage Broader Use of Atomic Requirements – A Call for Exchange and Experimentation

William L. Honig
Loyola University Chicago, whonig@luc.edu

Follow this and additional works at: https://ecommons.luc.edu/cs_facpubs

Part of the Computer Sciences Commons

Author Manuscript

This is a pre-publication author manuscript of the final, published article.

Recommended Citation


This work is licensed under a Creative Commons Attribution-Noncommercial-No Derivative Works 3.0 License. © The Author, 2024.
Establishing Metrics to Encourage Broader Use of Atomic Requirements – A Call for Exchange and Experimentation

William L. Honig
Department of Computer Science
Loyola University Chicago
Chicago, Illinois 60611 USA
+1.773.508.8034
whonig@luc.edu

ABSTRACT
There are seemingly many advantages to being able to identify, document, test, and trace single or “atomic” requirements during system development and maintenance. Ongoing work with Agile development has focused on “user stories” that can capture individual features for implementation. However, it is still difficult to evaluate the quality of such requirements and teaching their creation is difficult.

Based on a working definition of atomic requirement, this paper proposes a set of metrics for their evaluation. Ten metrics designed to measure atomic requirements are presented here: five used on individual requirements statements and five applied to a requirements document or set of requirement statements.

Example metrics for individual requirements include Requirement Atomic Completeness and Requirement Atomic Verifiability. Requirement Traceability is a proposed metric for a set of atomic requirements (such as a traditional requirements specification document.)

Further work is needed to make the suggested metrics more consistent and precise, including exploring the use of tools that may automate some of the measures. However, a first step is to refine the metrics themselves and ensure they adequately measure atomic requirement quality.

Hence this note shares the proposed metrics to encourage discussion and to recommend experimentation. The author is applying the metrics in software engineering courses doing team software development.

The goal is the eventual acceptance of a set of metrics for their evaluation. Ten metrics designed to measure atomic requirements (such as a traditional requirements specification document.)

The purpose of this paper is to foster discussion and comparison of metrics for evaluating atomic requirements. The author welcomes critical review of the ten metrics introduced here, alternative metrics, and proposals for experiments to further the development of atomic requirements in general.

2. DEFINING ATOMIC REQUIREMENTS
Work on requirements and requirements engineering often begins with the IEEE requirement definition as a “statement with translates or expresses a need and its associated constraints and conditions” [12, par. 4.1.17]. This requirement engineering standard also includes, in a list of quality characteristics of individual requirements “singular” which is defined as “The requirement statement includes only one requirement with no use of conjunctions” [12, par. 5.2.5].

Although this standard does not use the term atomic, others have amplified and refined the idea of singular into the concept of atomic requirements, atomic requirement statements, or atomic use cases [9,15,16,21,22]. These generally lead to defining singular to mean a complete and indivisible requirement documented as a whole. Following that work, this paper uses the following working definition:

Atomic Requirement: a natural language statement that completely describes a single system function, feature, need, or capability including all information, details, limits, and characteristics.

Individual atomic requirements are supported by a system glossary of terms, references to applicable industry standards, mockups of the user interface, etc. One way to identify such atomic requirements is from use case or system event analysis.

The Agile system development method often focuses on “user stories” [3,5,19] which can also lead to good atomic requirements for one specific system capability. As a brief example consider an application with a basic login screen and user story:

As a user, I want to be able to log in to the app with my userid and password (or reset my password when I forget it), so that I can use all the app’s features.

Several functions or activities are included or assumed in this story. A more atomic set of statements, each possibly able to be a separate story, would be as shown in Table I (next page). Requirement 4 also adds a non-functional requirement that was not explicit in the original story.

3. METRICS FOR ATOMIC REQUIREMENTS
Given the potential represented by atomic requirements, and despite the difficulties of the imprecise working definition, it is nevertheless attractive to measure and evaluate them with metrics. The metrics set described here is intended to work with any set of (supposedly) atomic
requirements describing some application, system component, or subsystem. It is designed to measure the requirements quality in general (not solely if they are proper atomic requirements or not).

The metrics are divided into two categories: those applied to an individual requirement statement and those which measure the complete set of requirements. The values of each individual metric are assigned during an inspection of the full set of requirements.

The value for each of the metrics is an integral number from 1 to 10. This range of values was picked arbitrarily to ensure a wide spread of values.

These metrics assume a proper system or application glossary. Key terms used in various requirements are defined once with all details in the glossary to ensure consistent use across the separate requirements statements. The glossary includes details on length, size, capacity, units, format when relevant to the defined terms.

Each metric has a name and an identifier for ease of reference, e.g.

**Requirement Atomic Completeness (Ra3)**

**Requirement Traceability (Rd4)**

where **R** indicates a requirements metric, **a** for metric applicable to single atomic requirement, **d** for metric measuring an entire document or set of requirements, followed by a unique sequential number.

### 3.1 Metrics For a Single Atomic Requirement

Table II presents a working set of metrics to measure each atomic requirement individually, and gives a high-level description of each. This section described the motivation for these metrics and further details on the current methods for measuring them.

The first two metrics, Requirement Correctness (Ra1) and Requirement Unambiguity (Ra2), are like oft-used terms for describing requirement quality [2,6]. They have their usual definitions and are evaluated in the usual way. For purposes of this paper, which chiefly looks at the issues raised with atomic requirements they will be briefly discussed.

The Requirement Correctness (Ra1) metric measures the legitimacy and genuine need for the capability or features described. For a high-quality measure this metric would use a formal requirements verification process (in a simpler, educational environment, the inspection team supplies a subjective rating). Requirements Correctness must consider whatever information the requirement provides on priority, importance, optionality, etc.

The Requirement Unambiguity (Ra2) metric evaluates the understandability of the requirement for intended audience(s) including stakeholders, developers, testers, etc. The goal is to prevent differing interpretations of the requirement by different readers; the system glossary supports this evaluation.

These two metrics are included in the individual metrics set to ensure that the overall value of the requirement is analyzed. The remaining three metrics address particularly the atomicity of the individual requirement.

### 3.1.1 Requirement Atomic Completeness (Ra3)

The Requirement Atomic Completeness (Ra3) metric determines if everything that belongs in the requirement is present. A simple chemistry analogy with an element shows the intent of this metric – to completely describe an element includes listing things such as its symbol, atomic number, common isotopes, atomic weight, etc. It’s unlikely that an element would be completely described without each of these.

Of course, potential atomic requirements are nowhere near as indivisible as chemical elements. The goal for the requirement inspection process is to measure how well a single requirement statement meets the completeness goal; to do so the following checklist is used:

- Are all likely types and values of inputs covered including clear indications of which are legitimate and which are invalid?
- Does the requirement cover all possible variations and sub cases for the feature or function, and clearly specify which ones are to be handled?
- Is it clear what outputs, changes in system state, and other results must or may be expected?
- There is nothing missing to make the requirement completely describe a single function or feature; are there any “What about this…” questions that can still be asked?
- Does the glossary include every term necessary to understand this requirement; are each of these terms defined fully in the glossary?

Currently, no individual values or weights are provided with this checklist; reviewers determine a single metric value for the requirement statement after considering all the points in the checklist.

### 3.1.2 Requirement Atomic Verifiability (Ra4)

The Requirement Atomic Verifiability (Ra4) metric evaluates the atomicity of a single requirement by measuring the likelihood that completing suitable test cases would indicate that the requirement was either fully met or not met. In other words, a truly atomic requirement can be tested as a single unit and give a single pass or fail result. Returning to the chemistry analogy, most elements can be tested to determine what they are – it is either carbon or it’s not.
In other words, the metric measures the degree to which the tests are bound together and mutually interdependent. This checklist supports the inspection (using the same one to ten scale where ten mean “most likely”, “most difficult”, etc.):

• How obvious is it what to do first in the test case (where to begin)? Are the inputs or stimulus to begin the test easy to determine?
• How well defined are the outputs or results of the feature or function? Are the values or state changes clearly determined (at least within each alternative)?
• Would it be difficult or impossible to skip one or more test cases (or steps in a test case) and still determine if the test passes (for at least some alternatives in the requirement)?
• How obvious is/are the test case(s) needed to test this requirement (if no test cases now exist)?

This checklist is, of course, easier to evaluate if the test(s) are already defined and documented and the tests are traceable back to the requirements they evaluate. Otherwise, the inspection team must use subjective evaluation to determine this metric.

3.1.3 Requirement Atomic Undecomposability (Ra5)

The Requirement Atomic Undecomposability (Ra5) metric determines if anything can be removed from the requirement (and likely put elsewhere in another atomic requirement). In other words, would removing some part of the requirement statement leave an inconsistent or ill-defined description of the function.

A general term for this characteristic might be stickiness or cohesion; a checklist to prompt thinking includes these questions:

• Is everything in the requirement necessary and important for it to be understandable? In other words, if something were to be removed, would the requirement no longer make sense or become incomplete?
• Is it likely that separating the requirement into two or more parts would be difficult, less understandable, or be likely to cause redundancy?
• Would a user or customer be able to express agreement or disagreement with the requirement with a clear “yes” or “no”?
• Would it be difficult to partially implement the requirement and achieve a working application or system?

When requirements have been defined with use cases or user stories, an atomic requirement will be tied to a complete system event or interaction between the environment and the system. Automated processing of requirements statements to test if they are decomposable may also be possible [7].

3.2 Metrics For a Set of Requirements

Table III lists metrics which measure a set of requirements, typically the complete requirements document for a system, subsystem, or application. These metrics are based on commonly used terms for requirements quality [4] and are not new or specific to atomic requirements; however, the methods for evaluating them are adapted and specialized to ensure atomicity is part of the metric. Each metric is valued with a single number.

Requirement Completeness (Rd1) evaluates the set of requirements statements for completeness. When used with a set of atomic requirements this evaluation offers a way to determine if any other singular requirement needs to be added. The granularity of atomic requirements may make this test easier – it becomes a simple question: shall another requirement be added to the set?

Requirement Consistency (Rd2) is also more straightforward when based on atomic requirements. Although it may be conceptually challenging, the metric can be calculated by looking at every possible pair of two requirements and asking if those two are consistent. The metric value is then reduced for any pair which raises concerns on compatibility or consistency.

Requirement Importance Ranking (Rd3) tests the existence of an importance ranking (e.g. essential, desirable, optional [4]) and subjectively evaluates the appropriateness of the rankings stated. Atomic requirements lend themselves to ordering and counting of requirements by their individual rankings; this comparison can support selecting a value for this metric (e.g. when the majority of requirements are all “essential” does that genuinely represent what’s needed).

Requirement Traceability (Rd4) evaluates the ability of the requirements to support typical approaches to traceability (linking to design, tests, and system changes). Each atomic requirement should have a unique and unchanging identification (such as a sequential number). Document section or paragraph numbers do not meet this need as they change often. When clearly delineated atomic requirements are used with a suitable naming scheme, this metric requires little subjective thought. An expanded definition, and more consistent evaluation of this metric would be possible using models of the traceability [8].

Requirement Purity (Rd5) measures the appropriateness of the requirement set as pure statements of system need without inappropriate assumptions about design, implementation, schedule, etc. While atomic requirements would support clear linkage with design choices, inclusion of design within the requirement set makes it difficult to clearly delineate separate atomic requirements. For example, many requirements may imply that information will be stored or updated in a typical relational database to support information queries; however, the actual structure and keys for that data table are best determined during system design.

While not a metric shown in Table III, the count of individual atomic requirements is a valuable piece of information as well. Well-structured atomic requirements, like atomic Agile user stories, are likely to each require a significant and separable amount of design, implementation, and testing during development. While very approximate, the total requirement count provides an indication of system size and changes in the number offer a insight into requirements churn.

4. OBSERVATIONS AND GOING FORWARD

Refinement to the metric set introduced here and further work on the definition and measurement techniques for atomic requirements seems promising. Some areas for further work include:

<table>
<thead>
<tr>
<th>Metric</th>
<th>Brief Definition</th>
<th>Range</th>
</tr>
</thead>
<tbody>
<tr>
<td>Rd1</td>
<td>Requirement Completeness</td>
<td>Is this set of atomic requirements complete – does it provide a full definition of functionality for the system or subsystem?</td>
</tr>
<tr>
<td>Rd2</td>
<td>Requirement Consistency</td>
<td>Is this set of atomic requirements internally consistent, with no contradictions, no duplication between individual requirements?</td>
</tr>
<tr>
<td>Rd3</td>
<td>Requirement Importance Ranking</td>
<td>Are each of the atomic requirements individually assigned suitable importance categories and are the assignments appropriate?</td>
</tr>
<tr>
<td>Rd4</td>
<td>Requirement Traceability</td>
<td>Are each of the atomic requirements uniquely identified with unchanging identifiers?</td>
</tr>
<tr>
<td>Rd5</td>
<td>Requirement Purity</td>
<td>Is this set of atomic requirements free from system design, project schedule, staffing, and other non-requirements material?</td>
</tr>
</tbody>
</table>
Some of these metrics, e.g., Requirement Consistency (Rd2), seem amenable to mechanization using machine learning or automated tools. Work on extracting dependencies from requirements [18] can provide a first step and is directly relevant to Requirement Consistency. A more prescriptive structure of natural language to facilitate an automated first review of proposed atomic requirements statements [10] may be helpful. Similarly, more automated analysis and improvement of Agile user stories as in [5] may be based on similar metrics.

A more controlled experiment, as in [13,23], could contrast student efforts on requirements with and without metric use. This approach would be interesting to indicate the separate value of atomic requirements with and without the use of metrics.

The author is currently conducting an experiment as suggested above. It uses the atomic requirements metrics presented in this paper with advanced software engineering students who are creating complete requirements specifications for a complex real-world application. The process for iterative review of the requirements calculates and uses the metric values as part of a requirement quality checklist. Defects discovered in the requirements (both during reviews and in later stages of development) are tracked and used to estimate the impact of the requirements metrics.

More discussion, exchange, and critical testing of atomic requirements metrics are clearly needed. The author welcomes feedback, suggestions, and opportunities to continue experiments that will lead to expanded use of metrics for atomic requirements.

5. REFERENCES


