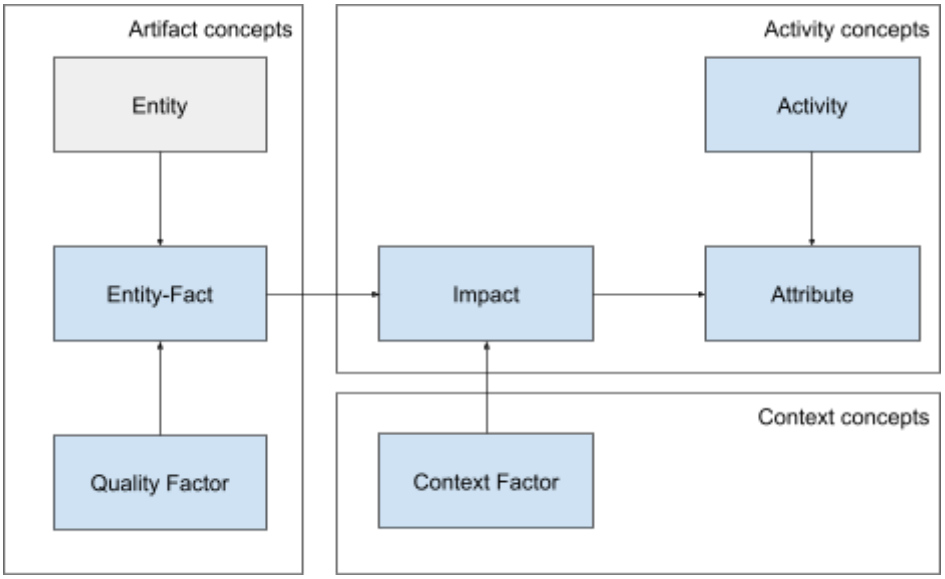
Coding Guideline

Perceived relevant Factors of Requirements Quality

This document guides the coding process for the requirements quality interview transcripts. During the interview, participants were prompted to report quality factors in the requirements specification that impact subsequent activities.

Concepts

The following concepts are relevant for the extraction.



The “entity” concept will not be coded because all interview statements revolve around the same type of requirements specification.

Concept	Description
Quality Factor	Property of the requirements specification
Entity-Fact	Value of the Quality Factor
Frequency	Frequency of occurrence of the factor
Context Factor	Given context about the impact of the quality factor
Activity	Impacted activity
Attribute	Affected property of the activity
Impact	Strength of the impact

Guideline

The following sections guide the coding process of the interview extractions. All examples provided in this guideline (written in “...”) are made up due to the confidentiality of the interview transcripts.

Entity Codes

The entity codes describe the requirements entity (i.e., the requirements specification). A *Quality Factor Mention* can be coded by up to two pairs of *quality factors* and *entity facts*.

Quality Factor & Entity Fact

Definition: A *quality factor* is a normative metric that maps a textual requirement of a specific granularity to a scale that informs about the quality of this input [2]. It describes a property of the entity and should be decidable based on the entity alone - though it does not have to be decidable automatically (e.g., sentence length can be automatically decided, conciseness cannot be automatically decided). An *entity fact* is the value that the variable *quality factor* has.

Coding Rule: Assign 1-2 codes to the *Quality Factor Mention* that best represents the property of the requirement described by the interview participant.

Quality Factor	Entity Fact	Definition
orientation	problem; solution	Specifies the solution rather than the problem
semantically redundant	true; false	Same meaning conveyed in different variants of text
lexically redundant	true; false	Same text repeated in multiple positions
horizontal traces	missing, existing	A requirement is linked to other, relevant requirements
vertical trace to input	missing, existing	A requirement is linked to input artifacts
vertical trace to output	missing, existing	A requirement is linked to output artifacts
quantified	true; false	Every numeric property of a requirement is specified in quantitative terms (e.g., requiring a response time to be "below 10ms" instead of "fast")
missing requirement	true; false	The requirements artifact misses a full requirement.
missing information	true; false	The requirements specification misses a specific piece of information (e.g., the semantic agent of a sentence in passive voice)
level of detail	too little; sufficient; too much	The specificity with which the requirements are specified
up-to-date	true; false	All information in one artifact is currently correct

coherent	true; false	All information within one artifact aligns and does not contradict
consistent	true; false	All information in one artifact aligns with all other artifacts
jargonic	true; false	Requirements specification contains domain-specific terminology
overloaded term	true; false	Using a term that has more than one meaning
feasible	true; false	A requirements specification can be implemented at all, i.e., is realistic
atomic	false; true	A specification implies exactly one requirement
concise	true; false	A requirements specification contains only necessary content, nothing additional or superfluous
scope	too small; sufficient; too large	The general scope of the requirement, i.e., the size of the implied feature
size	too small; sufficient; too large	The general size of the requirement, i.e., the size of the specification
density	too low; sufficient; too high	The amount of information per size
maturity	immature; mature	clear statement in which stage of the life-cycle the requirement is (i.e., intended for prototype or production purposes)
end-to-end view	missing; existing	Whether or not a requirement contains an end-to-end view of the feature. Not containing an end-to-end view is a symptom of solution-orientation
committed	true; false	Whether or not the feature which the requirement describes has already been committed to, i.e., sold to the customer.
type	functional; non-functional; mixed	Type of the requirement

Examples:

- "If the requirement is written from the perspective of a solution …" This statement describes the *quality factor* **orientation** with the *entity-fact* **solution** because the participant talks about a solution-oriented requirement.
- "When the requirement contains a lot of jargon and much information in only few sentences …" This statement describes two quality factors:
    - the *quality factor* **jargonic** has the *entity-fact* **true**, and
    - the *quality factor* **density** has the *entity-fact* **too high**.
- "The requirement lacks references to previously completed but related requirements." Here, the statement describes the *quality factor* **horizontal traces** (i.e., trace links to the same artifact type) with the *entity fact* **missing** as the interview participant talks about the lack of traceability between a requirement and other related requirements (i.e., artifacts of the same type).

Frequency

Definition: The *frequency* describes how often the interview participant encounters the quality factor.

Coding Rule: If the Frequency Mention contains a numeric assessment, map it to a code according to the range description. If not, use the keywords.

Code	Range	Keywords
Almost always	80-100%	
Often	60-79%	many times
Sometimes	40-59%	from time to time
Rarely	20-39%	rare
Almost never	0-19%	

Context Code

Definition: The *context* describes variables that influence the impact that the entity fact has on the activity fact but are not caused by the entity itself. Context covers all influences of the product, process, practices & tools, and people.

Coding Rule: The *Context Mention* can contain up to 3 individual codes describing the impact of other variables.

Context Factor	Definition
Novelty	Previously unknown feature, not an update to an existing feature
Agile	Using an agile development process
Involvement	Whether or not a person was involved in the inception of the requirement
Supplementary Communication	Communication supplementing the information conveyed in the requirement (e.g., via standup meetings or clarification syncs)
Dependency Management Tool	Tool to visualize and maintain dependency relations between relevant objects
Peer Review	Involving peers to make a pass on the requirement
Experience	General experience of an involved stakeholder/engineer
Domain Knowledge	General knowledge about the domain in question of an involved stakeholder/engineer

Examples:

- "*experienced* person" refers to the context factor **experience**
- "This occurs when the domain of a requirements specification is *new*." refers to the context factor **novelty**
- "If the domain of the specification is new but the engineer is very knowledgeable about it …" refers to two context factors: **novelty**, and **domain knowledge**

Activity Codes

The activity codes describe the activity that is impacted by the requirements entity. An *Activity Mention* can be coded by up to two pairs of *activity*, *attribute*, and *impact*. The activity can be the same in both activity code sets.
The activity codes mirror the entity codes (entity ≙ activity, quality factor ≙ attribute, entity-fact ≙ impact). A notable difference is that while the object on the entity side is fixed (the interview only discussed one specific entity), the object on the activity side can vary.

Activity

Definition: A requirements-affected *activity* is impacted by an entity fact and context factors.

Coding Rule: The available activities are organized in a tree, where a parent activity represents a more abstract version (i.e., a "superclass") of the child activity. The notation within the tree is "**Activity** (applicable attributes): description."

- **Processing** (duration, feasibility, completeness): an abstract activity that considers a requirements specification as input and produces some output.
    - **Understanding** (uniqueness): comprehending a sentence on a general, lexical level
    - **Interpreting** (uniqueness): comprehending a requirements specification on a semantic level and relating it to its domain
    - **Translating** (stability, coherence): transforming a requirements specification into a different artifact
        - **Implementing**: developing code that exhibits the features described in the requirements specification
        - **Verifying**: deriving test cases that assess whether a piece of code exhibits the features described in the requirements specification
    - **Assessing** (precision): evaluating a requirement
        - **Assessing feasibility**: determining whether a requirement is realistic to be implemented
        - **Estimating effort**: predicting the effort to implement a requirement
    - **Planning** (stability): determining the life-cycle of a requirement
        - **Coordinating**: orchestrating the subsequent work involving a requirement with the owner of related requirements
    - **Reusing**: using an existing artifact (not the requirements specification) for a new activity, e.g., reusing existing code to fulfill a new requirement

Examples:
- "the person using the requirement could misunderstand the text" refers to **understanding**
- "you might miss some test cases implied by the requirement" refers to **verifying** the requirement
- "A requirement must be connected to related requirements such that I can interact with the other authors." To "interact with other authors" refers to **coordinating**.

Attribute

Definition: An *attribute* is the (measurable) property of the activity that is impacted.

Coding Rule: Once an activity has been determined, select the appropriate attribute that describes the impacted property of the activity. All attributes of a parent activity also apply to all available child nodes.

Attribute	Definition
Unspecific	No specific attribute of an activity is mentioned, just general "ease" or "success"
Uniqueness	Whether the output of an activity is always the same or can differ depending on other factors.
Duration	The amount of time that the completion of the activity takes
Completeness	The degree to which the output of the activity covers the implied content of the input (e.g., whether the derived test cases cover all functionality implied by a requirement)
Precision	Accuracy of a prediction
Stability	How stable the results of an activity are (e.g., how reliable the subsequent plan of a requirement is)
Feasibility	Whether it is realistic that an activity can be completed at all
Coherence	Whether the output of an activity remains coherent with the existing artifacts
Traceability	Whether the output of an activity can be traced back to the causing requirement, e.g., to understand a decision

The code “unspecific” is the default, fallback code. If the *Activity Mention* describes an attribute more specifically, then use this more specific attribute. Otherwise, fall back to “unspecific.”

Examples:

- “The person using the requirement could misunderstand the text.” Here, the impacted attribute of the **understanding** activity is **uniqueness** because it describes that the requirement has more than one semantic resolution.
- “You might miss some test cases implied by the requirement.” Here, the attribute **completeness** of the **verifying** activity was impacted.
- “A requirement must be connected to related requirements such that I can interact with the other authors.” This is an example of the **unspecific** code: the statement just makes clear that horizontal trace links allow or improve the **coordinating** activity, but it does not specify exactly in what regard.

Impact

Definition: The impact represents the strength and direction with which the entity-fact and context factors influence the activity's attribute.

Coding Rule: The *Activity Mention* contains the direction and the *Impact Mention* contains the strength of the impact. The following codes are available

Code	Meaning	Usage
+3	Strong positive impact	Code for emphasized strong positive influence
+2	Positive impact	Default code for a positive influence
+1	Slight positive impact	Code for explicitly moderate positive influence
+0	No impact	Default code for no influence
-1	Slight negative impact	Code for explicitly moderate negative influence
-2	Negative impact	Default code for a negative influence
-3	Strong negative impact	Code for emphasized strong negative influence

Sometimes, the *Quality Factor Mention* has to be consulted in addition to determine the direction of the impact.

Example:

- "A requirement must be connected to related requirements such that I can interact with the other authors." There is no information on whether the impact is particularly strong or weak. Hence, fall back to the default code (-2 or +2). Since the described relation connects the missingness of relations to an impediment of the correlation activity, the entity-fact and impact need to be chosen accordingly (e.g., "missing" and -2)
- "The person using the requirement is bound to misunderstand the text." The "**is bound to** misunderstand" emphasizes the negative impact (-3)
- If an *Activity Mention* like "This might impact the understandability" is complemented by the *Impact Mention* "minimal," the impact is only small (-1)
- "It does not cause any problems." does not imply any impact at all (+0)

References

[1] Frattini, J., Montgomery, L., Fischbach, J., Mendez, D., Fucci, D., & Unterkalmsteiner, M. (2023). Requirements quality research: a harmonized theory, evaluation, and roadmap. *Requirements Engineering*, 1-14.

[2] Frattini, J., Montgomery, L., Fischbach, J., Unterkalmsteiner, M., Mendez, D., & Fucci, D. (2022, August). A live extensible ontology of quality factors for textual requirements. In *2022 IEEE 30th International Requirements Engineering Conference (RE)* (pp. 274-280). IEEE.