An approach for creating sentence patterns for quality requirements
An Approach for Creating Sentence Patterns for Quality Requirements

Jonas Eckhardt
Technische Universität München
Garching b. München, Germany
eckharjo@in.tum.de

Andreas Vogelsang
DCAITI
Technische Universität Berlin
andreas.vogelsang@tu-berlin.de

Henning Femmer
Technische Universität München
Garching b. München, Germany
femmer@in.tum.de

Abstract—Requirements are usually categorized in functional requirements (FRs) and quality requirements (QR). FRs describe “things the product must do” while QRs describe “qualities the product must have”. Besides the definition, classification, and representation problems identified by Glinz, there are two further problems with current definitions of quality requirements: (i) the definitions are imprecise and thus difficult to understand and apply, and (ii) the definitions provide no guidance or support for their application in a given organizational context. To tackle these two problems, we propose an approach that—given a quality attribute (e.g., performance) as input—provides a means to specify quality requirements by sentence patterns concerning this quality attribute. In this paper, we contribute a detailed presentation and description of our approach and a discussion of our lessons learnt while instantiating it for performance requirements. Additionally, we give guidance on how to apply our approach for further quality attributes. Through this approach, we aim at encouraging researchers to help us improve the precision of definitions for quality requirements and support practitioners in eliciting and documenting better quality requirements.

Index Terms—Quality Requirements, Sentence Patterns

I. INTRODUCTION

Requirements are usually categorized in functional requirements (FRs), quality requirements (QRs) and constraints [1]. FRs are characterized as “things the product must do” contrasting QRs as “qualities the product must have” and constraints as “organizational or technological requirements”. Although the importance of QRs for software and systems development is widely accepted, up until now, there is no commonly accepted approach for the QR-specific elicitation, documentation, and analysis [2]–[4]. This lack can result in high maintenance costs in the long run [3].

Besides Glinz’s definition, classification, and representation problem [5], there are two further problems with current definitions of quality requirements: (i) the definitions are not overly precise and thus not easily understandable and applicable, and (ii) the definitions do not provide guidance or support for their application in a given organizational context.

To tackle these two problems, we propose an approach that—given a quality attribute (e.g., performance) as input—provides a means to precisely specify requirements concerning this quality attribute. Our approach is based on the identification of content elements, i.e., different types of information characterizing the quality attribute (e.g., the desired latency of a system for performance requirements). In particular, given a quality attribute, our approach provides (i) a precise and explicit definition of content elements that are needed to specify requirements concerning the quality attribute, and (ii) a set of sentence patterns for practitioners to specify requirements concerning the quality attribute for a given organizational context. We achieve the precise and explicit definition by a structured identification of relevant content elements that requirements of a specific quality attribute may consist of. Furthermore, we use the idea of activity-based quality models [6], [7] for the customization of these content elements to a given organizational context and sentence patterns for guidance and support for their application in practice.

We already instantiated our approach for one specific quality attribute (performance) and conducted an empirical evaluation with respect to its applicability [8]. The results indicated that the approach is applicable and besides the constructive nature of our approach, further supports analytic quality assessment with syntactic analyses. For example, the question how can we assess that all information necessary are documented in a given textual requirement (i.e., the completeness of the individual requirement)?

In this paper, we contribute a detailed presentation and description of our approach, a discussion of our lessons learnt while instantiating it for performance requirements, and provide guidance for how to apply our approach for further quality attributes. The objective of this paper is to encourage other researchers to create more precise definitions for quality requirements.

The remainder of this paper is structured as follows: In Section II, we present our approach and discuss its application to performance requirements in Section III. We discuss the threats to validity of our approach and lessons learnt in Section IV. Finally, in Section V, we report on related work and conclude in Section VI.

II. APPROACH

Fig. 1 shows an overview of our approach: the approach takes a specific quality attribute as input and creates a precise and explicit definition and customized sentence patterns for requirements concerning this quality attribute. The resulting
definitions and sentence patterns can then be integrated in the overall RE process to support the elicitation, documentation, validation, and management of requirements in the given organizational context. Thus, our approach needs to be conducted in advance for a given set of quality requirements and a given context. Then, the results can be (re)used as, for example, a company standard to specify and elicit quality requirements.

A. Goals of the Approach

Before we describe the approach in detail, we first discuss the goals of the approach. Given a quality attribute, as for example performance, we try to achieve the following four goals:

1) Identification of relevant content elements: In literature there exists a large amount of publications concerning individual quality attributes. The challenge is to collect this large amount of qualitative data and extract the relevant content elements in a structured and reproducible way that guarantees that all relevant content elements are considered. The quality of the overall results of our approach heavily depends on the content elements that are identified to be needed to specify requirements concerning the quality attribute.

2) Precise definition of relevant content elements: Given a set of relevant content elements, a further challenge is how to define these precisely such that all stakeholders have the same understanding. This is a challenging yet creative activity. For example, we may define each content element by means of a glossary entry or give a formal definition by a mapping to a system model. The challenge is to find a way to define the content elements such that they are adequately represented. This activity is highly dependent on the context (e.g., involved stakeholders).

3) Customization to a given organizational context: Another challenge is to assess whether the content elements are actually relevant for a given organizational context. The simple answer here is to provide a one-size-fits-all solution. However, we argue that such a one-size-fits-all solution does not work for requirements engineering because the organizational contexts vary heavily. These variations include not only the information and level of detail in which projects document requirements but also how projects use requirements documents in their context. Thus, the challenge is to provide an approach that can be customized for a given organizational context.

4) Provide a means to specify requirements for practitioners: Finally, based on the relevant content elements for a given context, we aim to create a means that supports the structured elicitation, documentation, and management of requirements concerning this quality attribute.

B. Overview of the Approach

To meet the goals described above, our approach consists of four steps. Fig. 2 shows an overview of our approach; it takes a specific quality attribute as input (e.g., performance). The approach is separated in two parts: The first part (Step 1 and 2) is intended to create a precise and explicit definition of the quality attribute while the second part (Step 3 and 4) is intended to customize the definition to a specific organizational context and to provide a means for practitioners to specify requirements concerning this type.

1) Context-independent Definition: The goal of this step is to create a comprehensive content model that covers all content elements and relationships that are needed to specify requirements concerning the quality attribute. Fig. 3 shows an overview of this step. To get a complete list of content elements, we propose to use qualitative literature analysis (e.g., a structured literature review or expert interviews) with the goal to identify concepts related to the specification of requirements concerning the quality attribute. Then, in a next step, we identify the models that are used for the specification of requirements; These models may be in textual, semi-formal, or formal form (indicated by different icons in the figure). Then, based on these models, we identify content elements and create a consolidated content model that contains and relates all content elements. This qualitative analysis is a highly creative and subjective approach. We suggest to use researcher triangulation to reduce this threat to the overall validity. The result of this step is a content model that ideally is a superset of all aspects concerning the quality attribute in literature.

2) Precise Definition: Given the content model from the previous step, the goal of this step is to give a precise definition for the individual content elements of the content model. For each content element of the content model, we define both, its
syntax as well as its semantics. Depending on the stakeholders, we may give a definition on different levels of detail ranging from an informal glossary entry to a formal definition. For example, if we chose to use a glossary, we may define the syntax of the content element *modality* (of a requirement), as “the modality of a requirement may be one of *exclusion*, *obligation*, *enhancement*” and its semantics as “If the modality of a requirement is *exclusion*, the property described by the requirement must not hold, if it is an *obligation*, it must hold, and if it is an *enhancement*, it may hold”. If we want to define the content element more formally, we suggest to describe its meaning in terms of a system model (e.g., [9]). For example, if we aim to define the semantics of a *requirement*, we can map it to a logical predicate, which relates input streams to output streams. As with the previous step, this step is highly creative and depending on the context. The result of this step is a definition of each content element of the content model.

3) Context-dependent Customization: The goal of this step is to achieve a customization of the context-independent content model for a given organizational context. To achieve this, we propose to use the idea of activity-based quality models [6], [7] and use the context-independent content model as input for the creation of the activity-based quality model for the given context. In particular, we use a model of stakeholders and their development activities that take requirements of the quality attribute as input (e.g., design a test based on a performance requirement). Based on this model of activities, we successively analyze the content elements that a stakeholder needs in a requirement to complete the activity efficiently and effectively. For example, to perform the activity designing a test, it is necessary to know the scope of the requirement. Therefore, we classify content elements as mandatory or optional for an activity. The result of this step is a content model for the quality attribute that is adapted to a specific set of activities and where each content element is justified by at least one of these activities. By this, we achieve a customization of the content model to a given organizational context.

4) Concretization: In the final step, we must provide a means for practitioners to specify requirements concerning the quality attribute. To achieve this, we propose to derive a set of sentence patterns from the context-dependent content model. Sentence patterns have the advantage that they are easy to use for the documentation of requirements and support the structured elicitation and management of requirements. Fig. 4 shows an overview of the step. In particular, for each of the content elements in the content model, we derive a *sentence fragment*. The sentence fragment is intended to represent the meaning of the content element as close as possible. For example, let’s assume that the content element *modality* of a requirement may be an *enhancement*, an *obligation*, or an *exclusion*. In this case, we can create the sentence...
Sentence Fragments describes a specific start event, we can represent it by the if we aim to represent an optional content element which replaced by values when the pattern is instantiate. For example, if we target an operational context that encompasses all quality attributes and relationships, we add the content element in several refinement rounds. Table I shows a part of the glossary for performance requirements.

requirements specify relative time or resource constraints like “The system must have a processing speed of x requests/second”, and capacity requirements specify limits of the system like “The system must support at least x concurrent users”. Furthermore, literature defines further aspects related to performance requirements that apply for all three types of performance requirements. We call these aspects auxiliary conditions (e.g., the location of a measurement).

We then coded the results of the literature review as suggested by Grounded Theory to assemble a conceptual model of the quality attribute in form of a content model. The resulting content model contains content elements of the quality attribute and relations between them. Furthermore, we added content elements that apply to requirements in general (e.g., the scope of a requirement). The result of this step is a content model for performance that ideally is a superset of all performance aspects mentioned in literature. A detailed description of the resulting content model for performance can be found in Eckhardt et al. [8].

B. Step 2: Precise Definition

Given the content model from the previous step, the goal of this step is to provide a precise definition for the individual content elements of the content model (see step 2 in Fig. 2).

We started with an informal definition of the content elements by a simple glossary. To create the glossary, we iterated through the content model. We discussed the meaning of each content element in several refinement rounds. Table I shows a part of the glossary for performance requirements.

For a more formal definition of the content elements, we mapped them to FOCUS, a formal modeling theory and its probabilistic extension as introduced by Neubeck because we found in a previous study that performance requirements describe probabilistic and timed behavior of a system.

C. Step 3: Context-dependent Customization

In the third step, our goal is to achieve a customization of the content model for a given operational context (see step 3 in Fig. 2). To achieve this, we follow the idea of activity-based quality models and use the context-independent content model as input for the creation of the activity-based quality model for the given context. Here, we first consider all stakeholders and analyze their development activities that take
requirements of the quality attribute as input, such as design test of the test designer in case of performance requirement. We identify necessary and important content elements that these requirements must contain to complete the development activities efficiently and effectively. We accordingly classify content elements, marking crucial content elements as mandatory and the contributing content elements as optional. The result of this step is a content model that is customized for a given operational context and each content element is justified by at least one development activity.

For our running example, we used testing activities as described in the (rational) unified process (RUP) [14]. For each of the stakeholders’ activities, we identified the corresponding necessary content elements from the content model to complete the activity efficiently and effectively. As the description of the activities in the RUP is rather high-level and does not provide detailed insights about the required artifacts for an activity, we performed an in-depth analysis of the description of the respective activities. Then, in a pair of researchers, we discussed the activities and identified the necessary content elements of a requirement for that activity: We marked a content element as necessary when we agreed that its absence would require a stakeholder to invest additional effort for completing the activity or would even make the activity impossible. Table II shows the resulting mapping between the necessary content elements and the activities. For example, for the activity design test by the test engineer, the time/throughput/capacity property of the requirement is necessary as its absence would make it impossible to set up an adequate test environment. Furthermore, the scope of the requirement is necessary for the activity plan test, as the test engineer needs this information for assigning the test to a person/team responsible. The final context-dependent content model can be found in Eckhardt et al [8].

D. Step 4: Concretization

In the final step, we aim to provide a means for practitioners to specify requirement concerning the quality attribute. To achieve this, we propose to derive a set of sentence patterns from the context-dependent content model (see step 4 in Fig. 2). In particular, for each of the content elements in the content model, we derive a sentence fragment. The sentence fragment is intended to represent the meaning of the content element as closely as possible. Finally, we merge these fragments into sentences. The result of this step is a set of sentence patterns for the specification of requirements concerning the quality attribute.

In our running example, we iterated through the set of content items in a pair of researchers and discussed how to adequately represent this content element in terms of a sentence fragment. The complete set of patterns can be found in Eckhardt et al [8]. An exemplary instance of a sentence is

The system must have a processing time of \( \leq 10 \text{ ms} \) between event "receiving a request" and event "answering a request", when under a maximal load. Measurement takes place on production hardware. Included is browser render time.

IV. DISCUSSION

In this section, we discuss limitations and threats and direct implications of our approach.

A. Limitations and Threats

The quality of the results of our approach heavily depends on how the individual steps are performed. Furthermore, all steps require a high amount of creative and qualitative work and thus may be error-prone. To mitigate this threat, we provided guidance in this paper that shows how to perform the individual steps on the example of performance requirements. We described how we performed the individual steps and the respective results in detail and provided hints how to ensure quality.

1) Context-dependent Definition: In the first step of our approach, there are some threats that affect the generalizability and applicability of the results. The initial collection of literature may miss some important work, the extraction of models may miss models or include unimportant models, and finally the coding and consolidation of the models may lead to inconsistent or inadequate models. We try to mitigate these threats by using a structured and reproducible approach (e.g., a structured literature review) and by performing the extraction and coding steps in a pair of researchers (researcher triangulation). Furthermore, we suggest to validate the resulting models with quality requirements from practice or perform validating interviews with practitioners.

2) Precise Definition: The goal of the second step is to create a precise definition such that we reduce misunderstandings. We propose to use either a glossary or a formal definition by means of a system modeling theory. However, in both cases, it is a highly challenging and creative activity and the individual content elements can be contradictory or inadequate. To mitigate this, we propose to perform a validation in form of interviews with researchers as well as with practitioners.
3) Context-dependent Customization: The result of this step is highly dependent on how the customization is performed. We propose to use activity-based quality models that try to make the relation between activities, artifacts, and quality attributes explicit. However, the quality of the results still depends on the level of detail and adequacy of the activity-based quality model. In our running example, we build our customization based on the activities for testing as described in the RUP. However, the description of these activities was on a very high level of detail, and thus, we discussed each activity in a pair of researchers. In summary, to mitigate this threat, we propose to either use a detailed activity-based quality model or perform a cross validation or researcher triangulation.

4) Concretization: In the final step, the creation of sentence patterns is straight-forward. However, the quality of the overall approach depends on how well practitioners can apply the sentence patterns to requirements and are how much they are willing to use the patterns. To mitigate this, we propose to validate the resulting patterns with quality requirements in practice and furthermore conduct interviews with practitioners concerning their willingness to use the patterns.

B. Syntactic Analyses: Challenging Incompleteness

Besides the constructive nature of our approach, we can further support analytic quality assessment with syntactic analyses. For example, through such patterns, we can syntactically detect that a textual individual requirement does not document a specific information, such as the location for a performance requirement. For example, in case of a performance requirement stating “The delay between [event A] and [event B] shall be short”, the content regarding the quantifier is not contained.

- The requirement does not contain the content. For example, in case of a performance requirement stating “The delay between [event A] and [event B] shall be short”, the content regarding the quantifier is not contained.
- The requirement implicitly contains the content. With implicit, we mean that the content is contained in the requirement, but we need to interpret the requirement to derive the content. For example, in case of a performance requirement stating “The delay between [event A] and [event B] shall be typically be 10ms”. In this case, regarding the quantifier, we can interpret “typically” as “median”.
- The requirement explicitly contains the content. With explicit, we mean that the content is contained without interpretation. For example, in case of a performance requirement stating “The delay between [event A] and [event B] shall have a median value of 10ms”. In this case, regarding the quantifier, the content is explicitly contained.

We can now derive the following definitions for strong and weak completeness and for incompleteness of requirements of a given quality attribute:

**Definition (Strong Completeness).** A requirement of a given quality attribute is strongly complete, if all mandatory content elements (w.r.t the context-dependent content model of the attribute) are explicitly contained in its textual representation.

**Definition (Weak Completeness).** A requirement of a given quality attribute is weakly complete, if all mandatory content elements (w.r.t the context-dependent content model of the attribute) are explicitly or implicitly contained in its textual representation.

**Definition (Incompleteness).** A requirement of a given quality attribute is incomplete, if at least one mandatory content element in the context-dependent content model.
elements (w.r.t the context-dependent content model of the attribute) is missing in its textual representation.

We argue that this definition of completeness for requirements of a given quality attribute can be used to detect incompleteness and thus to pinpoint to requirements that are hard to comprehend, implement, and test. For example, requirements of class incomplete are not testable at all, requirements in class weakly complete need to be interpreted by the developer and tester and therefore bear the risk of misinterpretations, and requirements in class strongly complete contain all content necessary to be implemented and tested. Thus, we argue that our approach further provides a helpful and actionable definition of completeness for quality requirements. This definition of completeness can then be used to support analytic as well as constructive quality control.

C. Analyses of the Content of Quality Requirements

Besides the assessment of completeness, one can further leverage our approach to analyze the content of quality requirements in practice. Our approach results in a context-independent content model for a given quality attribute and in a context-dependent content model for that attribute. The context-independent content model provides a general definition of the content elements of the quality attribute and the context-dependent model provides a justification for each content model.

We can now analyze textual quality requirements and map the content elements found in the requirements to the content model. If we have a sufficiently large data set, we can now analyze observations and draw conclusions about the content elements of quality requirements in general. For example, a common point of view of quality requirements is that they are cross-functional and consider the system as a whole. When analyzing performance requirements, we also included the scope of a requirement in the content model. This allows us to quantitatively analyze the distribution of the scope of performance requirements found in practice.

D. Implications for Industry

Our approach is a step towards increasing the completeness of quality requirements. Not only the concretization via sentence patterns could be easily implemented in a requirements authoring or management tool. Such a tool may provide instant feedback to the requirements engineer about missing or optional content elements, similar to requirements smells [15], [16]. Furthermore, the tool might check the terms used in a requirement with respect to an underlying domain model. The tool could then uncover terms that are neither part of the consolidated terminology nor defined through the pattern semantics.

An additional benefit of our approach is that it makes content in natural language requirements explicit and traceable through content elements. This allows connecting specific content elements of requirements with specific content elements in related artifacts such as test cases or components within the implementation. Updates within requirements may then be propagated directly to corresponding test cases for example, making maintenance activities more efficient and effective.

V. RELATED WORK

There is a variety of work on requirement patterns in RE. Franch et al. [17] present a metamodel for software requirement patterns. Their approach focuses on requirement patterns as a means for reuse in different application domains and is based on the original idea of patterns by Alexander et al. [18], i.e., each pattern describes the core of a solution of a problem that occurs over and over again. In particular, the PABRE framework contains a catalogue of 29 QR patterns [19], 37 non-technical patterns [20], and a method for guiding the use of the catalogue in RE [21]. Their approach for creating the patterns catalogue is similar to ours, as it is also based on requirements literature and a content analysis. However, they provide solutions for recurring problems while our sentence patterns provide a means for the specific of customized requirements.

Supakkul et al. [22] present four kinds of NFR patterns for capturing and reusing knowledge of NFRs and apply these patterns in a case study. Their patterns and, in particular, the objective pattern can be used to identify important NFRs for a context or capture a specific definition of an NFR from the viewpoint of a stakeholder. Thus, their patterns define important content elements of a quality attribute in terms of soft goals, which is similar to our context-dependent content model [23], [24]. Our approach provides a structured way to define and customize these content elements and also provides sentence patterns to specify requirements. However, their patterns can be used to define the specific quality attribute but furthermore provide solutions and alternatives and thus go one step further into the architecture or design of a system. Our approach focuses on definition, customization, and concretization of requirements concerning a specific quality attribute.

Withhall [25] presents a comprehensive pattern catalogue for natural language requirements in his book. The pattern catalogue contains a large number of patterns for different types of requirements. In contrast to their work, in our approach, we derive patterns from literature and customize them to a specific application context. Almeida Ferreira and Rodrigues da Silva [26] introduce RSL-PL, a language for the definition of requirements sentence patterns. Their pattern definition language can be used to represent our sentence patterns.

Kopczyńska and Nawrocki [27] present a method for eliciting non-functional requirements, which is composed of a series of brainstorming sessions driven by the ISO 25010 quality sub-characteristics. Elicitation is supported by Non-functional Requirements Templates (NoRTs), which are statements that require some completion to become a well-formulated NFR. Similar to our sentence patterns, the authors differentiate between core parts, parameters, and optional parts within the templates. The sentence patterns derived by our approach are additionally adapted to specific classes of quality requirements.

Mylopoulos et al. [23] propose a comprehensive framework for representing and using QRs in the development process.
Similar to our approach, they propose a means to integrate QRs in the development process, however, they do not provide a structured approach for explicitly stating the content elements for specifying requirements concerning quality attributes and do not provide a means for specifying QRs.

VI. Conclusion

We provided an approach that—given a quality attribute as input—provides a means to precisely and explicitly defines the content elements that are needed to specify requirements concerning this quality attribute, and provides a means for practitioners to specify these requirements for a given organizational context based on sentence patterns. The approach consists of four steps:

1) **Context-independent Definition**: Relevant content elements are identified by means of qualitative literature analysis and coding.

2) **Precise Definition**: The resulting content elements are precisely defined by e.g., a glossary or formalization by means of a mapping to a system model.

3) **Context-dependent Customization**: The content elements are customized to a given organizational context by using the idea of activity-based quality models.

4) **Concretezation**: Sentence patterns are used as a means for practitioners to specify requirements concerning the quality attribute.

As our main goal was to provide guidance for the application of our approach, we furthermore discussed threats to validity and lessons learnt while instantiating it for performance requirements. Finally, we argue that our approach is applicable for performance requirements and besides its constructive nature, provides a means for various statical analyses, as for example completeness analyses.

We are planning to apply our approach for further quality attributes, in particular, for availability as a direct next step. As a broader vision, we are planning to unify the resulting content models in one content model for quality requirements.

**Acknowledgements**

We would like to thank M. Broy, S. Eder, and M. Junker for their helpful comments on earlier versions of this work. This work was performed within the project Q-Effekt; it was partially funded by the German Federal Ministry of Education and Research (BMBF) under grant no. 01IS15003 A-B. The authors assume responsibility for the content.

**References**


