A Closer Look on the Difficulties to Determine the Quality of Software Requirements

Patrick Kummler
Karlsruhe Service Research Institute
Karlsruhe Institute of Technology
Karlsruhe, Germany
patrick.kummler@kit.edu

Hansjörg Fromm
Karlsruhe Service Research Institute
Karlsruhe Institute of Technology
Karlsruhe, Germany
hansjoerg.fromm@kit.edu

***Abstract*—Increasing demands on quality and complexity are a major challenge for the development of industrial software products. Automotive software in particular is subject to additional safety, security and legal demands. In such software projects, the specification of requirements is the first concrete output that is mostly written in natural language. However, in practice, two problem areas exist: First, due to reasons like lack of knowledge and missing experience of engineers, requirements quality often is not at a satisfactory level. Second, a massive increase of the number of requirements for software poses a scalability issue. In our research, we want to take a closer look on the quality determination of software requirements. We present an overview of existing research approaches based on the standard ISO/IEC/IEEE 29148:2011 that offers nine essential characteristics for requirements quality. In addition, we analyze results from several sessions in which experts rate automotive software requirements.**

Keywords—Requirements Quality, Natural Language Requirements, Quality Characteristics, Requirements Rating, Software Requirements.

I. Introduction

In the automotive industry, innovative services and software functions decide the success of today's vehicles. Automotive manufacturers invest a major part of their resources in the development of customer functions and valuable services. This trend leads to a growing number of software requirements in development projects. With increasing complexity and strict safety, security and legal demands the manufacturers are faced with various challenges especially in the specification process [1-4].

In general, the success of a software project strongly depends on the quality of specified requirements [5], [6] and "requires fluid collaboration and communication between clients and software engineers" [7, p. 2]. Several established approaches, such as formal specification methods to describe and specify software requirements, already exist. But, "despite the significant advantages attributed to the use of formal specification languages, their use has not become common practice" [8, p. 1]. "Natural language is key in requirements engineering" [9, p. 1]; specifications are still organized in text documents and used as basis for the communication between relevant stakeholders [8, 10].

In our research, we explore and analyze the quality determination of textual requirements based on characteristics, attributes and desirable properties of a requirement. Different approaches largely rely on methods of natural language processing that are complemented with machine learning techniques. A machine learning algorithm is designed to automate the task that previously human experts would have done. The performance of such an algorithm should always be evaluated against a benchmark. Since quality is not objectively measurable, this benchmark can only be provided by human judgement. Thus, our research question is how reliable and consistent human experts can rate the quality of software requirements. To answer this research question, we draw upon the groundwork on requirements quality that has resulted from different standardization efforts within the software engineering community. The standard ISO/IEC/IEEE 29148:2011 [11] provides us with a set of characteristics that are believed to determine the quality of a single requirement. The main objective of our research is to take a closer look on these characteristics, especially asking the question how well human experts can rate them.

This paper contains an overview of the characteristics from [11] and presents relevant and current research about the quality measurement of textual requirements. In addition, a method for requirements rating is described and results from expert sessions are presented. The research is done in cooperation with an international automotive engineering and consulting company and enables us to have access on software requirements from the automotive industry. The rating sessions are conducted with experts from industrial practice that are currently working in automotive software development projects. The use of real requirements and a rating through experts from the automotive industry ensures reliable results for our research.

In the following chapter, we present and explain characteristics for requirements according to [11] and identify initial issues. Chapter three contains an overview of related work and reveals the research gap. In chapter four, the preparation, the execution and the results from the rating sessions are presented. The last chapter contains existing limitations.

II. Requirements Quality and Characteristics

Initially, we introduce several definitions of a requirement. Reference [12] defines a requirement as "something required, something essential to the existence or occurrence of something else". The unspecificity in this definition reveals the challenges when defining the term *requirement*. According to this, a requirement needs to be required and necessary. A more specific definition is given by [13] where a requirement is defined as: (1) a condition or capability needed by a user to solve a problem or achieve an objective; (2) a condition or capability that must be met or possessed by a system or system component to satisfy a contract, standard, specification, or other formally imposed document; (3) a documented representation of a condition or capability of the previous two arguments.

In [14], a requirement describes the quality that a software system must possess, as well as the prevailing conditions that are in force for its life cycle. Therefore, a requirement is responsible for general quality aspects of the implemented software. The authors in [15] describe a requirement as "something the product must do to support its owner's business, or a quality it must have to make it acceptable and attractive to the owner" [15, p. 9]. A broader definition proposed by [16] defines a requirement as "a specification of what should be implemented. They are descriptions of how the system should behave, or of a system property or attribute" [16, p. 6]. This definition considers different types of information and input that can be defined as a requirement. In most definitions a requirement serves as basis for the implementation of a target. For our research, we use the second expression of [13] as it represents the challenges of software requirements and the belonging quality determination.

The quality of software requirements can be defined by different criteria. References [11] and [17] provide a set of characteristics to define a well-written requirement as something that fulfils different criteria. Whereas in [11] *unambiguous*[1] is defined as characteristic, [17] offers a more holistic approach by using the characteristic *clear and concise* instead that also includes the understandability and the preciseness of a requirement. Thus, we use the characteristics from [11] and the modification of [17] for the quality determination of textual requirements. In the following the description of each characteristic is presented.

- ***Clear and concise.*** The requirement ensures that statements can only be interpreted unambiguous. "[T]he terms and syntax used must be simple, clear and exact" [17, p. 10]. For a clear and concise requirement, the use of weak terms, synonyms and unclear sentence structure lead to misunderstandings.
- ***Complete.*** The requirement describes adequately "the capability and characteristics to meet the stakeholder's needs" [11, p. 11]. Further explanation and enhancement of the requirement is not necessary.
- ***Consistent.*** The requirement has no conflicts. Defined terms are used consistently throughout the requirement.
- ***Feasible.*** The requirement can be implemented technically and does not need further advanced technologies. The system constraints are considered regarding legal, cost and schedule aspects.
- ***Implementation independent.*** The requirement is specified independently from the implementation: "The requirement states what is required, not how the requirement should be met" [11, p. 11].
- ***Necessary.*** The requirement contains relevant information and is not deprecated.
- ***Singular.*** A requirement cannot be divided in further requirements. It includes one single statement.
- ***Traceable.*** "The requirement is upwards [, downwards and horizontally] traceable" [11, p. 11]. Every requirement at each development stage can be traced to a requirement either to the current or to the previous and subsequent development stage. The requirement considers the dependency and possible conflicts among software.
- ***Verifiable.*** The requirement necessitates the verification of the statement by using the standard methods inspection, analysis, demonstration or test [17].

The fulfilment of the characteristics *complete*, *singular* and *clear and concise* are mentioned as main challenge during the specification [18-20]. The characteristic *clear and concise* is also described as the Achilles heel [20] of software requirements specifications. Considering [21] and [22] *clear and concise* is mainly responsible to enable the determination of the remaining characteristics. If a requirement is not fulfilling this characteristic, other characteristics can hardly be determined. Authors describe this phenomenon as "surface understanding" and "concept understanding" [21], or "clarity" and "content" [22]. Surface and clarity consider "what is stated" [21]. Concept or content focus on the question "what is meant or implied" [21].

According to [11] and [17], the characteristics are applied to individual requirements. However, with the given definition for *traceable*, a distinction between characteristics that apply to individual requirements and characteristics that strongly depend on the existence of further relevant requirements is necessary. Characteristics of the first group can be applied to individual requirements without further information needed. We identify *clear and concise*, *feasible*, *implementation independent*, *singular* and *verifiable* in this group. The characteristics *necessary* and *traceable* relate to the second group. The application of these characteristics necessitates additional relevant requirements. For *traceable*, information about linked requirements would be most helpful. Same applies for *necessary* where the necessity of a requirement can be detected when the whole requirements specification is available.

The characteristics *complete* and *consistent* have a special role and relates to both groups. *Complete*, as example, can be applied to individual requirements and considers, whether the requirement "needs no further amplification" [11, p. 11]. Also, *complete* can be applied to a set of requirements where "it contains everything pertinent to the definition of the system or system element being specified" [11, p. 11]. Same applies for the characteristic *consistent*. The categorization is similar to the distinction of requirement characteristics from the ISO standard 26262:2011 [23].

These are differences between characteristics that can be found by a purely theoretical consideration. For our research, we are interested in the detection of differences between characteristics as experts see them. We want to reveal the possibility to rate a characteristic by humans and the influence on an overall quality of a requirement that is perceived by experts.

III. Related Work and Research Gap

Many efforts have been targeted towards modeling, rating and measuring the quality of textual requirements. In Table I, we present related work that is divided in three research categories:

- **Assistance (A).** Tools that support requirements engineers in the specification of requirements. These tools mostly follow a defined and static set of metrics for the quality determination.

[1] For better identification quality characteristics are in italic.

- **Transformation (T).** Approaches that transform textual requirements into formal and logic specifications.
- **Classification (C).** Methods that enable the classification into good and bad requirements by using e.g. machine learning techniques.

We allocate related work to the research categories and analyze the occurrence of characteristics from the modified ISO standard [11, 17] (Identical or Similar). An identical characteristic in a paper is used with a corresponding definition as in [11] and [17]. Similar indicates a related use (e.g. *realizability* is similar to *feasible*). The presented overview in Table I does not claim to be conclusive. It contains an extract of relevant approaches, methods and tools from the literature and reveals the research gap.

Most of the latest papers can be assigned to the categories Transformation (T) and Classification (C). Especially the use of machine learning techniques to analyze and evaluate the quality of textual requirements is a common topic. These papers build upon the research of assisting tools (A) and enhance these approaches with methods from different disciplines, such as natural language processing and analytical methods. There is a trend towards developing algorithms and models that automatically evaluate the quality of textual requirements by using attributes and indicators to classify good or bad requirements. Most of the related work focus on the characteristics *clear and concise*, *complete* and *consistent*.

Almost every author uses at least one of these characteristics. The characteristics *feasible*, *implementation independent* and *necessary* are not investigated at all. Similar characteristics – like *correctness*, *modifiability*, *validability*, *testability* [8, 38], *understandability* [25, 28] and *abstraction* [7] – are used as well to determine the quality of requirements. The characteristics *traceable* and *verifiable* are focused mostly in research that deals with assisting tools.

Table I reveals the gap regarding the quality determination of textual requirements. Whereas the characteristics *clear and concise*, *complete* and *consistent* are adequately investigated in the research, the determination for *feasible*, *implementation independent* and *necessary* is barely available.

For our research, we do not exclude characteristics for the determination of requirements text quality, although the evaluation of some characteristics for a single requirement does not seem to be conducive at first glance. Despite the discussion in the previous chapter about the distinction of characteristics that apply to individual requirements and characteristics that apply to a set of requirements, we are convinced that the characteristics *traceable* and *necessary* are also relevant for the quality determination of an individual requirement. Although further relevant requirements would be helpful at this point, an individual requirement could also offer indications whether a requirement is *traceable* or *necessary*.

The research papers presented in Table I consist of a variety of approaches and methods. In the following, some papers are presented shortly regarding the use of characteristics from [11] and [17].

TABLE I. ANALYSIS OF RELATED WORK

Author(s)	*Research Category*	*clear and concise*	*complete*	*consistent*	*feasible*	*implementation independent*	*necessary*	*singular*	*traceable*	*verifiable*
Wilson et al. 1997 [8]	**A**	●	●	●					●	●
Mich & Garigliano 2000 [24]	**A**	●								
Fabbrini et al. 2001 [25]	**A**	●	●	●				○		
Fantechi et al. 2003 [26]	**A**	○		●						○
Ilieva et al. 2005 [27]	**T**									
Kaiya & Saeki 2006 [29]	**T**	●	●	●						
Ormandjieva et al. 2007 [21]	**C**	●								
Berry et al. 2007 [28]	**A**	●	●	●						○
Verma & Kaas 2008 [30]	**T**	●	●	●						
Holtmann et al. 2011 [31]	**T**	○	●	●						
Yang et al. 2012 [32]	**C**	○								
Genova et al. 2013 [7]	**A**	●	●	●		○		○	●	●
Huertas et al. 2013 [33]	**C**	●	●					○		
Ghosh et al. 2014 [37]	**T**		●	●	○					
Soeken et al. 2014 [34]	**C**	○			○			○		○
Arellano et al. 2015 [35]	**T**		●	●						
Parra et al. 2015 [36]	**C**	○				○		○	○	○

● *Identical* ○ *Similar*

In [8], the authors present a tool (ARM) to identify requirements that need to be improved. They use "desirable characteristics" [8, p. 2] based on IEEE Std 830-1993 [38] – a predecessor of the current ISO standard [11]: *complete*, *consistent*, *correct*, *modifiable*, *ranked*, *testable*, *traceable*, *unambiguous*, *understandable*, *valid* and *verifiable*. Identical characteristics can be found in the current ISO standard [11]. The characteristics *feasible*, *implementation independent*, *necessary* and *singular* are not used. However, this research paper serves as groundwork and is mentioned in several approaches as starting point.

In [25], a tool (QuARS) displays requirements together with automatically detected indicators. Four quality properties are mentioned: *non-ambiguity*, *specification completion*, *consistency*, *understandability*. The latter property is influenced by multiplicity that is pointed out if the requirement "has more than one main verb or more than one direct or indirect complement that specifies its subject" [25, p. 4]. Reference [11] describes the characteristic *singular* in a similar way: "[t]he requirement statement includes only one requirement with no use of conjunctions" [11, p. 11]. *Clear and concise*, *complete* and *consistent* are used analogue. Other characteristics are not mentioned.

Reference [27] presents an approach for the automatic transition of natural language software requirements into formal presentation. The authors dissect each requirement sentence in its constituents subject, predicate and object and arrange word groups in tabular form. An entity relationship diagram is constructed from the tabular presentation using nouns as entities and verbs and prepositions as relationships. Characteristics are not used at all to ensure the transition into formal specification from a qualitative point of view.

Reference [30] describes a tool (RAT) to detect and flag requirements that violate presented best practices. The best practices build upon "common requirements problems" [30, p. 753] like *ambiguous* terms, *inconsistent* and *incomplete* requirements. The common problems and an analogue inverse description can be found for *clear and concise*, *consistent* and *complete* in [11] and [17].

In [7], the authors present desirable properties and indicators used to evaluate and measure quality in textual requirements. A tool (RQA) displays requirements together with automatically detected indicators and an overall perceived quality score. These indicators are based on following characteristics: *atomicity*, *precision*, *completeness*, *consistency*, *understandability*, *unambiguity*, *traceability*, *abstraction*, *validability*, *verifiability* and *modifiability*. *Abstraction* is defined as to "tell what the application must do without telling how it must do it" [7, p. 28]. The description of *implementation independent* is similar and "states what is required, not how the requirement should be met" [11, p. 11]. Same applies for *atomicity* where a requirement is "clearly determined" [7, p. 28] and *singular* where a "statement includes only one requirement" [11, p. 11].

Reference [36] describes a method to classify the quality of textual requirements by using machine-learning techniques. The authors emphasize desirable properties of a requirement: *correct*, *consistent* and *complete*. The property *correctness* is used further in the approach. However, the authors do not offer a description of *correctness*. The research is mainly based on the properties and approaches from [7] and additionally neglects *consistency* and *completeness*. Their method is composed of two tasks. In the first task, they generate classifiers. They let experts classify requirements according to their quality and use a set of metrics associated with the requirements to build their classifiers. In the second task, they estimate and evaluate the quality of new requirements based on the same set of metrics. The tool developed by [7] is used to extract the metrics for each requirement.

The characteristic *correctness* is not mentioned in the ISO standard [11]. However, previous standards [38] and several authors use it to describe properties of a requirement [8], [25], [3]. "There is no tool or procedure that ensures correctness" [38, p. 4]. In [39] *correctness* is defined as the state when "[t]he requirement [is] an accurate representation of the entity need from which it was transformed" [39, p. 18]. According to this definition, *correctness* is linked with the *traceability* of a requirement. Reference [8] defines the *correctness* of a requirement when it "accurately and precisely identif[ies] the individual conditions and limitations of all situations" [8, p. 2]. In [25] the authors describe *correctness* evaluation as "the verification that the system to be constructed is correctly described by them" [25, p. 2]. For [28] *correctness* is "the lack of factual errors […] and matching what the customer wants" [28, p. 3]. The description of *correctness* differs, and the characteristic is not clearly defined. In our research, we see *correctness* as a comprehensive criterion that influences several characteristics from the ISO standard. However, due to an inconsistent and not standardized definition, *correctness* is not used as characteristic in the rating sessions.

IV. Rating Sessions

With the overview and analysis of related work, we reveal that several characteristics for the quality determination of textual requirements are rarely investigated. Generally, we do not find approaches, which take into consideration all characteristics from [11]. Therefore, to address this research gap, we enable the rating of the whole set of characteristics.

Our work takes an additional approach from the mentioned literature: we use experts' knowledge to rate requirements but also to collect their inputs about different characteristics. Working closely with an automotive engineering company allows us to work within an industrial environment and with experts from the automotive industry. In this chapter, we present the preparation, the execution and the results of the requirements rating sessions.

A. Preparation

The preparation phase includes information about how requirements data is collected and prepared for the usage in the rating tool. We collect English and German text data from 83 software requirement specifications of different automotive development projects. These software projects aim at developing advanced driver assistance systems, such as lane assist, collision avoidance functions and other safety systems. The data initially consists of 57.801

objects, including 9.365 headings and 10.775 objects marked as information. The object types heading and information are used to structure the specification documents and do not contain relevant requirement information in general. We remove objects belonging to these types, as we only want to consider requirement objects for the rating.

In further steps, we reduce the database by English requirements, as our focus group of experts has German as mother language. We are convinced that once we prove the feasibility for German requirements the approach can easily be adapted to English requirements as well. At the end of the data cleansing steps, the dataset consists of 14.341 unique software requirements in German language. We then randomly select 766 unique requirements and import this dataset twice (1532 requirements) in a self-developed rating tool. This tool allows saving the rating results during the sessions as it is linked with a pre-defined database. We import the dataset twice as we want to have two evaluations for each requirement from different experts.

In the beginning of the survey, we ask general information about the expert's role and years of experience. The experts then rate random requirements according to the characteristics defined in [11] and [17]: *clear and concise*, *complete*, *consistent*, *feasible*, *implementation independent*, *necessary*, *singular*, *traceable* and *verifiable*. Each characteristic can be rated between 1 (very bad) and 5 (very good). If an expert cannot rate a characteristic for a requirement, the answer no rating possible (NRP) is possible to select. After the rating of each characteristic, the experts conduct the overall perceived rating of the requirement between 1 (very bad) and 5 (very good) including the option no rating possible (NRP). The experts confirm the rating and the next requirement is displayed.

B. Execution and Results

During May and October 2018, nine female and 83 male experts from the automotive industry rated 1532 software requirements (766 unique requirements). On average, the experts have five years of experience in their respective role. We identify nine different expert roles: test engineer, failure manager, systems integrator, function owner, function developer, requirements manager, requirements engineer, software developer, software architect.

We aggregate the rating results in two areas of investigation: influence and ratability. In the first area of investigation, we determine the individual coherence of each characteristic with the overall perceived rating of a requirement through the coefficient of determination.

Table II shows the coefficient of determination according to the overall perceived rating of a requirement based on univariate regression. *Singular* and *implementation independent* are less coherent with the overall perceived rating with a result less than 0.30 compared to other characteristics. The R^2-value for *clear and concise* and *complete* is greater than 0.60 and implies a strong coherence with the overall perceived rating. The R^2-value for the remaining characteristics range between 0.34 for *traceable* and 0.45 for *verifiable*.

TABLE II. COEFFICIENT OF DETERMINATION (R^2) FOR OVERALL PERCEIVED RATING

	R^2
complete	0.63
clear and concise	0.62
verifiable	0.45
feasible	0.44
necessary	0.44
consistent	0.38
traceable	0.34
singular	0.27
implementation independent	0.24

Compared with the analysis of related work in the previous chapter (cf. Table I), the research focus on *clear and concise* and *complete* is comprehensible. However, the characteristic *consistent*, also occurs often in the related work, has less coherence with the overall perceived rating of a requirement. Moreover, the characteristics *verifiable*, *feasible* and *necessary* are worth to investigate in further research about quality determination of textual requirements. Especially the two latter characteristics are almost completely neglected in the current research. Lastly, *singular* and *implementation independent* are less coherent with the overall perceived rating.

In the second area of investigation, the ratability of a requirement is analyzed. The ratability describes if experts are able to rate characteristics of a requirement and identifies possible challenges for the rating. In this analysis we follow the research question from our introduction: how reliable and consistent human experts can rate the quality of software requirements.

Every requirement is rated twice by different experts. Disagreements in the rating of characteristic are presented in Table III. The first row "Average Variance" regards the variance in the evaluation of a single requirement. If the value is low, the individual rating of the experts corresponds. A high value implies discrepancies between the evaluations.

The value for the average variance range between 0.47 for *feasible* and 0.62 for *clear and concise* and *complete*. For *clear and concise* and *complete* the average variance value above 0.60 is comparable high and indicates minor challenges regarding a consistent rating between experts. *Feasible* has the lowest variance among the ratings and can be evaluated more consistently. The remaining characteristics have medium values between 0.52 and 0.60 for the average variance.

Further, we analyze the percentage of "NRP" (no rating possible) for each characteristic. If an expert cannot rate a characteristic for a requirement, this option is possible to select. With the interpretation of this value, we can derive additional information about the ratability and reveal difficulties in the rating. The characteristic *consistent*, for example, could not be rated for 17% of all requirements.

TABLE III. EVALUATION DISCREPANCIES

	Average Variance	*% NRP*
perceived overall rating	0.52	2%
complete	0.62	5%
clear and concise	0.62	<1%
verifiable	0.56	4%
feasible	0.47	13%
necessary	0.57	16%
consistent	0.56	17%
traceable	0.55	35%
singular	0.54	2%
implement. independent	0.60	4%

The experts also have difficulties in rating *traceable*, *feasible* and *necessary*. An explanation for the rating difficulties is that further information is not available for several individual requirements. In addition, the individual requirement does not offer sufficient indications to rate the characteristic. Another reason could be the expert's background and level of experience that leads to different forms of implicit knowledge and the ability to rate requirements despite less information value. A deeper analysis is part of the future research. On the other hand, for *clear and concise*, *singular* and the perceived overall rating the experts are able to rate almost every requirement. For *complete*, *verifiable* and *implementation independent*, the rating is less difficult as for about 5% of the requirements a rating through the experts was not possible.

Clear and concise is a characteristic well considered in the literature about the quality determination of textual requirements. Thus, the coherence with the overall perceived rating of a requirement is not surprising at all as well as the ratability of this characteristic. However, the experts rating for *clear and concise* is not corresponding as compared to other characteristics. Same applies for *complete*. Other characteristics, such as *feasible* and *necessary*, reveal coherence with the overall perceived rating of a requirement as well. The variance for *feasible* is comparable low. However, experts take the option "no rating possible" more often. Approaches in the literature especially for *feasible* and *necessary* are barely available. Regarding these facts it is crucial to consider these characteristics in the future research about the quality determination of textual requirements.

V. CONCLUSIONS AND LIMITATIONS

This paper sets up to explore and analyze quality approaches of textual requirements. We reveal that literature on requirements quality largely falls into three categories: Assistance, Transformation and Classification. Several attempts to define requirements quality are available. We identify the modified ISO/IEC/IEEE 29148:2011 standard [11, 17] as the relevant one. The standard describes how requirements quality is defined and offers relevant characteristics to determine quality as well. Several relevant approaches use parts of these quality criteria and enable a qualitative analysis of textual requirements. However, we do not find approaches that use the whole set of proposed characteristics.

With the identification of the research gap, we develop a rating tool that enables experts to rate the quality of textual requirements. The rating is based on the characteristics from the modified ISO standard [11, 17]. The tool let experts carry out ratings on textual requirements from industrial projects of the automotive industry. The results from the rating sessions help us to reveal relations between individual characteristics and the overall perceived rating of a requirement. We derive following insights:

- **Influence.** We reveal coherences from individual characteristics with the overall perceived rating based on univariate regression. We identify *clear and concise* and *complete* as coherent characteristics with the overall perceived rating of a requirement. Research about the quality of textual requirements based on these characteristics is represented well in the related work. We also recognize *singular* and *implementation independent* to be less coherent with the overall perceived rating. Further, the characteristics *feasible* and *necessary*, barely investigated in the current research about quality determination of textual requirements, have a higher coherence with the overall perceived rating compared to the characteristic *consistent*. We identify a lack of research regarding the coherence of characteristics with the overall perceived rating and existing literature. Moreover, we do not find a comprehensive approach considering all presented characteristics.
- **Ratability.** The ratability enables us to identify challenges and difficulties in the rating of requirements. We find characteristics with similar variance values that indicates a common understanding and rating between experts. Based on a comprehensive analysis we derive several results regarding the ratability of characteristics. For experts the characteristics *singular*, *clear and concise* and the perceived overall rating is possible to rate. For some characteristics the experts have difficulties in rating and providing consistent answers. This implies especially for *traceable*, where more than a third of the requirements could not be rated.

This research has also some limitations. In particular, our approach is based on the assumption, that requirements can be handled as natural language despite their high technical and specific vocabulary. As only 1532 requirements (766 unique requirements) are rated until today, the results comprise only a first step towards a comprehensive approach.

As depicted in Table I of the paper, several characteristics are neglected in the current research about the quality determination of textual requirements. Although the evaluation of some characteristics does not seem to be conducive at the first sight, we do not exclude characteristics in our research and in the rating sessions with experts. Thus, the observed difficulties for rating e.g. *traceable* and *consistent* stem from the evaluation approach: As each participant only rates a single, random requirement, the expert is not able to have a full view on

all requirements even not at the very end of the rating session process. In order to depict the resulting end system or to check the dependencies to other requirements – in terms of *consistency* or horizontal *traceability* – full knowledge of the whole set of requirements would be helpful. Though, we are convinced that experts can evaluate a single requirement as well. When rating e.g. the characteristic *consistent*, the expert only concentrates on the individual requirement and focus the consistency within its statement(s).

Further limitations include implicit knowledge as an important fact that strongly correlates with the expert's level of experience. It stresses the point that the creation of requirements should follow closely defined quality criteria suggested by standards like ISO/IEC/IEEE 29148:2011 [11] to allow a better understanding for persons with less implicit knowledge.

Based on these results, further research questions arise for future research. The quality of textual requirements can now be determined by using characteristics from the modified ISO standard. The next step is to investigate, how to measure the quality of textual requirements by using quality characteristics. Is it possible to derive proven relations between quality characteristics and indicators? As quality characteristics are qualitative, they can only be judged and not be measured. Thus, we need to identify indicators that can be measured quantitatively and represent defined quality characteristics as well.

REFERENCES

[1] D. Mohr et al., "The road to 2020 and beyond: What's driving the global automotive industry?" McKinsey & Company, 2013.

[2] F. Sagstetter et al., "Security challenges in automotive hardware/software architecture design," in Proceedings of the Conference on Design, Automation and Test in Europe, pp. 458–463, 2013.

[3] H. Lan, C. Zhang, and H. Li, "An open design methodology for automotive electrical/electronic system based on quantum platform," in Advances in Engineering Software, vol. 39, pp. 526–534, 2008.

[4] M. Wolf, Security Engineering for Vehicular IT Systems. Vieweg+Teubner, 2009.

[5] M.I. Kamata and T. Tamai, "How does requirements quality relate to project success or failure?" in 15th IEEE International Requirements Engineering Conference, pp. 69–78, 2007.

[6] E. Knauss, C. El Boustani, and T. Flohr, "Investigating the impact of software requirements specification quality on project success," in International Conference on Product-Focused Software Process Improvement, pp. 28–42, 2009.

[7] G. Génova, J.M. Fuentes, J. Llorens, O. Hurtado, and V. Moreno, "A framework to measure and improve the quality of textual requirements," in Requirements Engineering, vol. 18, pp. 25–41, 2013.

[8] W.M. Wilson, L.H. Rosenberg, and L.E. Hyatt, "Automated analysis of requirement specifications," in International Conference on Software Engineering, pp. 161–171, 1997.

[9] D.M. Berry, "Ambiguity in natural language requirements documents," in Monterey Workshop, Springer, Berlin, Heidelberg, pp. 1–7, 2007.

[10] J. Kasser, "The First Requirements Elucidator Demonstration (FRED) tool," in Systems Engineering, vol. 7, pp 243–256, 2004.

[11] ISO/IEC/IEEE 29148:2011, Systems and software engineering - Life cycle processes - Requirements engineering, Geneva, Switzerland, 2011.

[12] Merriam-Webster, Definition of requirement, https://bit.ly/2st8p1c (Accessed: 11.01.2019).

[13] IEEE Standard Glossary of Software Engineering Terminology, The Institute of Electrical and Electronics Engineers, IEEE Std. 610.12-1990, New York, 1990.

[14] C. Rupp, Requirements-Engineering und -Management: Aus der Praxis von klassisch bis agil, 6th ed., Carl Hanser Verlag, München, 2014.

[15] S. Robertson and J. Robertson, Mastering the requirements process: Getting requirements right, 3rd ed., Addison-Wesley, 2013.

[16] G. Kotonya and I. Sommerville, Requirements engineering: processes and techniques, 1st ed., Wiley Publishing, 1998.

[17] G. Fanmuy, J. Marvin, and H. Ronald, Deployment Package - Needs and Requirements Engineering - Systems Engineering Basic Profile. Technical report, International Council on Systems Engineering (INCOSE), 2014.

[18] F.P.J. Brooks, "No silver bullet-essence and accidents of software engineering," in Proceedings of IFIP Tenth World Computer Conference, pp. 1069–1076, 1986.

[19] S. Boyd, D. Zowghi, and A. Farroukh, "Measuring the expressiveness of a constrained natural language: an empirical study," 13th IEEE Internationl Conference on Requirements Engineering, pp. 339–349, 2005.

[20] D.M. Berry, E. Kamsties, and M.M. Krieger, From contract drafting to software specification: linguistic sources of ambiguity, Handbook, 2003.

[21] O. Ormandjieva, I. Hussain, and L. Kosseim, "Toward a text classification system for the quality assessment of software requirements written in natural language," in Fourth international workshop on Software quality assurance: in conjunction with the 6th ESEC/FSE joint meeting, pp. 39–45, 2007.

[22] A.J. Tyznik, S. Verma, Q. Wang, M. Kronenberg, and C.A. Benedict, "Distinct requirements for activation of NKT and NK cells during viral infection," in The Journal of Immunology, vol. 192, no. 8, pp. 3676–3685, 2014.

[23] International Organization for Standardization, International Standard 26262: Road vehicles - Functional safety, International Standard, 1st ed., 2011.

[24] L. Mich and R. Garigliano, "Ambiguity measures in requirement engineering," in Proceedings of the International Conference on Software Theory and Practice, pp. 39–48, 2000.

[25] F. Fabbrini, M. Fusani, S. Gnesi, and G. Lami, "The linguistic approach to the natural language requirements quality: benefit of the use of an automatic tool," in Proceedings of the 26th Annual NASA Goddard Software Engineering Work, 2001.

[26] A. Fantechi, S. Gnesi, G. Lami, and A. Maccari, "Application of linguistic techniques for use case analysis," in Requirements Engineering, vol.8, pp. 161–170, 2002.

[27] M. Ilieva and O. Ormandjieva, "Automatic transition of natural language software requirements specification into formal presentation," in Natural Language Processing and Information Systems, vol. 3513, pp. 427–434, 2005.

[28] D.M. Berry, A. Bucchiarone, S. Gnesi, G. Lami, and G. Trentanni, "A new quality model for natural language requirements specifications," in Proceedings of the International Workshop on Requirements Engineering: Foundation of Software Quality, REFSQ, pp. 1–12, 2006.

[29] H. Kaiya and M. Saeki, "Using domain ontology as domain knowledge for requirements elicitation," in Requirements Engineering, 14th IEEE International Conference, pp. 189–198, 2006.

[30] K. Verma and A. Kass, "Requirements analysis tool: A tool for automatically analyzing software requirements documents," in International Semantic Web Conference, pp. 751–763, 2008.

[31] J. Holtmann, J. Meyer, and M. von Detten, "Automatic validation and correction of formalized, textual requirements," in Proceedings of the 4th IEEE International Conference on Software Testing, Verification and Validation Workshops, pp. 486–495, 2011.

[32] H. Yang, A. De Roeck, V. Gervasi, A. Willis, and B. Nuseibeh, "Speculative requirements: automatic detection of uncertainty in natural language requirements," in 20th IEEE International Requirements Engineering Conference (RE), pp. 11–20, 2012.

[33] C. Huertas and R. Juárez-Ramírez, “Towards assessing the quality of functional requirements using English/Spanish controlled languages and context free grammar,” in The Third International Conference on Digital Information and Communication Technology and its Applications (DICTAP2013), pp. 234–241, 2013.

[34] M. Soeken et al., “Quality assessment for requirements based on natural language,” Special Session at the Forum on specification & Design Languages (FDL), Munich, 2014.

[35] A. Arellano, E. Carney, L. Martin, C. Park, and M.A. Austin, “Frameworks for natural language processing of textual requirements,” in International Journal On Advances in Systems and Measurements, vol. 8, pp. 230–240, 2015.

[36] E. Parra, C. Dimou, J. Llorens, V. Moreno, and A. Fraga, “A methodology for the classification of quality of requirements using machine learning techniques,” in Information and Software Technology, vol. 67, pp. 180–195, 2015.

[37] S. Ghosh, N. Shankar, P. Lincoln, D. Elenius, and W. Li, “Automatic requirements specification extraction from natural language (ARSENAL),” SRI International Menlo Park CA, pp. 1–14, 2016.

[38] IEEE Std 830-1993 - Recommended Practice for Software Requirements Specifications, 1993.

[39] INCOSE: Guide for writing requirements. Technical report, Requirements Working Group, 2015.