

# Case study: How well can IBM's "Requirements Quality Assistant" review automotive requirements?

Amalinda Post<sup>a</sup>, Thomas Fuhr<sup>a</sup>

<sup>a</sup>Robert Bosch GmbH, Postfach 300240, 70442 Stuttgart, Germany

## Abstract

In the last years natural language processing evolved immensely. Tools based on that technology, like e.g., the Requirements Quality Assistant (RQA) by IBM or QVscribe by QRA claim to improve the requirements process for natural language requirements. In this paper we evaluate whether RQA supports a requirements engineer at Bosch in evaluating the quality of automotive requirements. In our case study, we come to the conclusion, that pure syntactical checks, based on the INCOSE rules of writing requirements, are not sufficient to address the needs of Bosch. We think that tool vendors should integrate semantic checks to their syntactical ones to further increase the practical use of their tools.

## Keywords

case study, requirements analysis, natural language processing, Requirements Quality Assistant

## 1. Introduction

Requirements are the foundation of projects in the automotive domain. Misinterpretation of or defects in requirements lead to a substantial amount of rework or even safety critical failures if not detected in time. To minimize the risk, the development process asks for a review before starting the implementation. In the review the requirements engineer invites all affected engineers, that will implement or test the requirements in the future, to get a common understanding and find defects. The review process takes about one week, as everyone has a tight schedule and needs some time to review. For an agile team that seems like an eternity. It would be a great improvement to have a tool that gives feedback in real-time.

There are natural language tools available, that claim to assess the quality of requirements in real-time. One such tool is the Requirements Quality Assistant (RQA) by IBM [1]. In this case study, we raised the question whether RQA could support the review process of automotive requirements at Bosch. We chose to evaluate RQA, as this tool can be easily integrated in the Requirements Management Tools "Doors" and "Doors Next Gen" most commonly used in the automotive domain at Bosch.


---

*In: F.B. Aydemir, C. Gralha, S. Abualhaija, T. Breaux, M. Daneva, N. Ernst, A. Ferrari, X. Franch, S. Ghanavati, E. Groen, R. Guizzardi, J. Guo, A. Herrmann, J. Horkoff, P. Mennig, E. Paja, A. Perini, N. Seyff, A. Susi, A. Vogelsang (eds.): Joint Proceedings of REFSQ-2021 Workshops, OpenRE, Posters and Tools Track, and Doctoral Symposium, Essen, Germany, 12-04-2021*

✉ Amalinda.Post@de.bosch.com (A. Post); Thomas.Fuhr@de.bosch.com (T. Fuhr)



© 2021 Copyright for this paper by its authors. Use permitted under Creative Commons License Attribution 4.0 International (CC BY 4.0).

 CEUR Workshop Proceedings (CEUR-WS.org)

## 2. Case Study Design

### 2.1. Problem description and study goals

The Requirements Quality Assistant (RQA) [1] by IBM shall support a requirements engineer in writing good requirements. IBM claims that in using RQA the review costs can be decreased by up to 25%. In order to assess whether RQA can support a requirements engineer at Bosch in the automotive domain we raised several questions:

**Question 1** Does RQA support a requirements engineer in reviewing against Bosch's requirements quality criteria?

**Question 2** Does RQA give further valuable feedback (in addition to Bosch's quality criteria)?

**Question 3** How fast does RQA compute its analysis results?

To address Question 1, Table 1 gives an overview on Bosch's requirements quality criteria. These criteria are the minimum set of quality criteria, all requirements written at Bosch shall adhere to. In the different company parts often further quality criteria, like e.g., *feasibility* or *unambiguity*, are added in the review checklists. Bosch's quality criteria are derived from standards like, e.g., IEEE830 [2] and ISO26262 [3]. Note that some of the criteria, like, e.g., *clearness* can be checked per requirement, while others like, e.g., *completeness* or *consistency* can only be checked for the whole set of requirements. This differentiation is important, as RQA currently checks every requirement on its own.

### 2.2. Selection Criteria for Requirements

In the first step we selected requirements documents from different BOSCH projects of the automotive domain. To get a representative sampling, we applied stratified sampling over the automotive application domains power train, driving assistance and car multimedia. We then used convenience sampling to select a project out of every stratum.

Each project had several requirements documents, some consisting of more than 100 pages. In order to get a representative sample we asked the corresponding requirements engineers to give us a subset of their requirements such that, first, the subset contained roughly 50 requirements, second, the subset was representative for their application domain and, third, the subset contained all requirements in one or more chapters. We asked the requirements engineers to give us chapters of requirements instead of randomly chosen, unrelated requirements, as in our experience a requirement should not be interpreted out of its context. This way we obtained the following four requirements sets: a set specifying a throttle functionality  $R_1$  (of a powertrain project), a set specifying radio functionality  $R_2$  (of a car multimedia project), one specifying cruise control functionality  $R_3$  (of the driving assistance domain), and one regarding variant coding  $R_4$  (needed in every automotive domain).  $R$ , the union of these sets, consisted of 173 artefacts: 15 headings and 158 requirements.

| <b>Bosch Quality Criteria</b>         | <b>Description</b>   | <b>criteria per req.</b> | <b>criteria per req. set</b> |
|---------------------------------------|--|--------------------------|------------------------------|
| <i>Complete</i>                       | The requirements set fully implements all requirements at the previous hierarchical level and the requirements of all stakeholders have been considered. I.e., it contains all relevant functional and non-functional requirements, boundary conditions, and implicitly expected requirements. |                          | x                            |
| <i>Correct</i>                        | The requirement is accepted as correct and valid for the stakeholder of the respective level. It is correctly derived from its superior specification.   | x                        | x                            |
| <i>Consistent and Free of overlap</i> | Each requirement is consistent with the other requirements. There is no overlap between requirements.  |                          | x                            |
| <i>Clear</i>                          | The requirements specification is comprehensible for the intended stakeholders. All requirements are quantified, with defined boundary conditions.   | x                        | x                            |
| <i>Verifiable</i>                     | The implementation of the requirement is possible and can be verified (technically assessable).  | x                        | x                            |
| <i>Traceable</i>                      | The origin of the requirement, its implementation all the way through testing, and the change history is traceable.  | x                        | x                            |
| <i>Assessable</i>                     | The requirement can be evaluated and prioritized, for example, in terms of promised performance/delighting factors, and risk/stability   | x                        | x                            |

**Table 1**  
Requirements Quality Criteria for Requirements at Bosch

### 2.3. Case Study Approach

The setting of the case study is depicted in Figure 1: As input we used  $R_1, \dots, R_4$ . In the first step, these requirements sets were analysed in a manual review. The Findings were documented per requirement in the requirements management system Doors Next Gen in the attribute "Manual Review". After that we started RQA. The tool documented its findings per requirement in the attribute "Issues Found by RQA" and a quality score in the attribute "Score". After that we compared the Findings of the manual review with the findings by the tool and documented the result in the attribute "Evaluation Result" and "Eval Comment".

We differentiated as "Evaluation Result" between: *no/bad fit*, *fits >=60%*, *fits 100%*. We chose *fits 100%*, if the Findings of the manual review and the RQA Evaluation Result were (semantically) identical. We chose *fits >=60%* if the majority of the findings of the manual review and RQA fitted to each other, otherwise we chose *no/bad fit*. Note, that this implies that we also set the "Evaluation Result" to *no/bad fit*, if the reviewers had no findings in the manual review, but the tool added findings in "Issues Found by RQA".

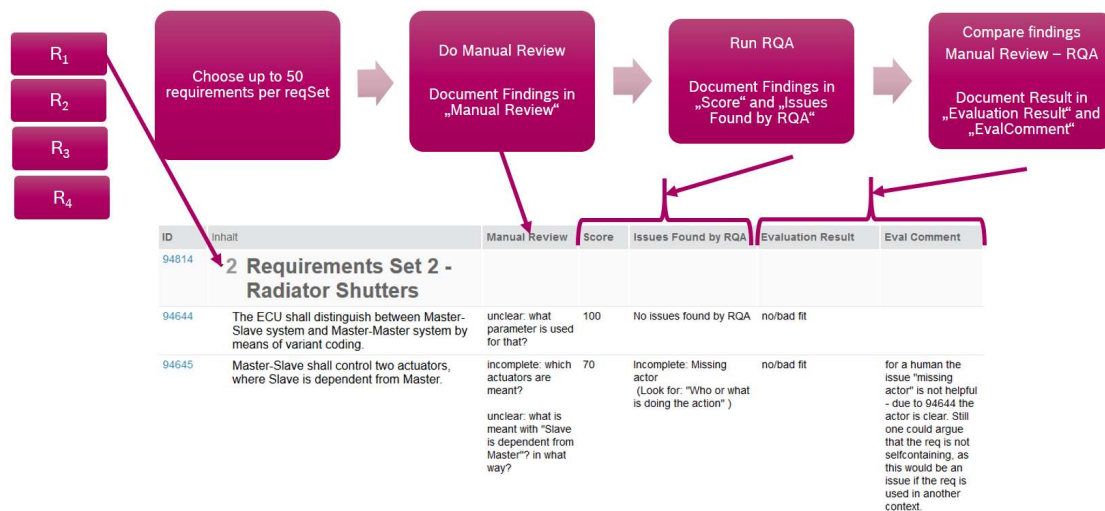


Figure 1: case study design

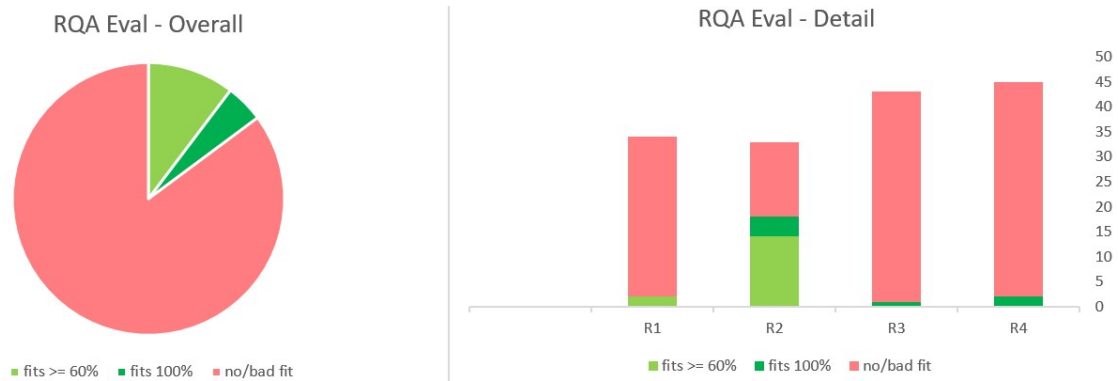
## 2.4. Case Study Evaluation

To address the first Study Question we evaluated how well the Findings of the manual review and the RQA evaluation matched. Figure 2 depicts the result for the attribute "Evaluation Result". Note, that in  $R_2$  the percentage of  $fits \geq 60\%$  findings was the highest. This requirements set was a set taken from a new project, where the requirements were a first draft that hadn't been reviewed before. Most fitting findings were addressing "compound requirements", i.e., findings, noting that the requirement was not atomic.  $R_1$ ,  $R_3$  and  $R_4$  were taken of projects in a much later development phase. In later development phases, such findings are less frequent in manual reviews, as the requirements are then already splitted into atomic parts.

The findings of 132 of 158 requirements matched with "no/bad fit". Thus, the findings of the manual review and the tool differed vastly. RQA implements checks derived of the INCOSE Guide for Writing Requirements [4]. The guide is about how to express textual requirements in the context of systems engineering. There are rules like "Avoid vague terms", "Avoid combinators", "Avoid the use of 'not'". The rules are mainly meant for junior requirements engineers to write better requirements.

However, it seems the checks do not really address many of Bosch's quality criteria. Bosch's quality criteria are mainly content related, like *completeness*, *consistency*, *correctness*, *verifiability*. Many of the manual review findings were of that sort, i.e., the reviewer stated that information was missing. The tool makes syntactical checks for natural language grammar—but these won't be sufficient to address content related properties. Semantic checks on the whole requirements set would be needed, to address the need.

To address the second Study Question, we checked whether RQA detected findings, that were not detected in the manual review, and found valuable by the requirements engineer. We saw that many of the rules resulted in false positives. For example the rule "Avoid the use of 'not'" seems to be difficult in the automotive domain. In the automotive domain, the



**Figure 2:** Case Study Result for Question 1

requirements are very detailed. Already on system level, there are > 1000 requirements for one single control unit. Often the signals on system level are already defined with a signal table. So, for a signal *SystemStatus* there is a definition of the enumeration values like, e.g., "off, init, active, standby". A formulation of a requirement like "If the *SystemStatus* is not 'active', then..." is preferred to a formulation like "If the *SystemStatus* is in 'off' OR 'init' OR 'standby', then...". The reason for that is both readability and maintainability. Just think in a later system release a further *SystemStatus* value "shutDown" is added. In the first formulation only a small subset of requirements will have to change - in the second formulation many more requirements have to be adapted, resulting in a huge effort also in the later development steps.

A check that was considered as helpful, with very few false positives, was the check for "compound requirements", which detected non-atomic requirements. Another check that would be helpful is the check for missing limits and missing units. However, currently this check is done for single every requirement. In the automotive domain, this is not a good strategy. When having 1000 requirements, writing the limits and units of a signal in every requirement will eventually lead to inconsistent requirements. Instead they are often defined centrally once per signal. So instead of doing the check per requirement, it should be done for the whole requirements set.

In the case study, the requirements engineers came to the conclusion that in the current state, RQA didn't support them in the review. It didn't address semantic checks, and many rules lead to false positives.

Regarding the third study Question, RQA is fast. It took some seconds to analyse the requirements set and write the analysis result into the requirements management system. Compared to a manual review (which in case of an inspection takes about 1 week), this is a huge improvement. Thus, if RQA would address the quality criteria of Table 1, the tool could help to vastly shorten the review process.

## 2.5. Threats to validity

In this section, we analyze threats to validity defined in Neuendorf [5], Krippendorff [6], and Wohlin [7].

### 2.5.1. External Validity

**Sampling Validity** [5] This threat arises if the sample is not representative for the requirements. In order to minimize this threat we used the selection procedure described in Section 2.3, i.e. requirements from four different projects and three automotive application domains. A limitation of the case study is that the samples are small and we only used requirements of BOSCH projects. Thus we cannot extend our results to the whole automotive domain but only for BOSCH's automotive domain.

**Interaction of Selection and Treatment** [7] This threat arises if the requirements engineer in this study (see Section 2.3) are not representative for BOSCH requirements engineers. The requirements engineers in this case study were experienced requirements engineers, with more than 10 years of experience in requirements engineering in the automotive domain. Thus, it may be that the results differ for less experienced requirements engineers.

### 2.5.2. Internal Validity

**Selection** [7] This threat arises due to natural variation in human performance. The requirements engineer in this study (see Section 2.3) could have been especially good or bad in reviewing requirements. For two sets the same evaluator did the manual review on the requirements. For two other sets the manual review findings were taken from the reviews by the project team. To minimize the threat, the requirements engineers double checked the evaluation results of the other samples. If they found differing review findings in the manual review, they discussed the findings.

### 2.5.3. Construct Validity

**Experimenter expectancies** [7] Expectations of an outcome may inadvertently cause the evaluators to view data in a different way. The evaluators have no benefit or disadvantage from a good or bad outcome for the applicability of the RQA tool. Thus, such psychological effects probably did not affect the evaluators.

**Semantic Validity** [6] This threat arises if the analytical categories of texts do not correspond to the meaning these texts have for particular readers. In the case study the same evaluator mapped the categories *no/bad fit*, *fits >=60%*, *fits 100%* to the requirements for all four sets of requirements.

Note, that in the study we didn't address, whether the tool might help when formulating requirements. Another setting would be needed to investigate that aspect.

## 3. Conclusion

This case study investigates the question whether in practice RQA supports a requirements engineer in the automotive domain at Bosch in reviewing requirements. Based on the results of Section 2.4 we come to the conclusion, that in the current state RQA does not sufficiently address

the need. The main reason is that the tool mainly checks for syntactical rules, derived from INCOSE rules. In contrast the BOSCH requirements engineers are interested in semantic checks. If RQA would also integrate semantic checks, then the tool could lead to big improvements, as a direct feedback on, e.g., *consistency* or *completeness* when writing the requirements would be very valuable. We think this information is very important for tool vendors. Other NLP-based tools like QVscribe [8] by QRA also base their tool solely on the INCOSE rules. Based on our results, we claim that a tool only addressing these rules will not meet the needs of the automotive domain at Bosch.

## References

- [1] IBM, Engineering requirements quality assistant, <https://www.ibm.com/products/requirements-quality-assistant>, 8.2.2021. URL: <https://www.ibm.com/products/requirements-quality-assistant>.
- [2] IEEE830, Recommended Practice for Software Requirements Specifications, 1998. URL: [http://ieeexplore.ieee.org/xpls/abs\\_all.jsp?arnumber=720574](http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=720574).
- [3] ISO26262, Road vehicles - Functional safety, Part 8, Baseline 17, 2010.
- [4] INCOSE, Guide for writing requirements, <https://connect.incose.org/Pages/Product-Details.aspx?ProductCode=TechGuideWR2019Soft>, 2019. URL: <https://connect.incose.org/Pages/Product-Details.aspx?ProductCode=TechGuideWR2019Soft>.
- [5] K. A. Neuendorf, Content Analysis Guidebook, Sage Publications, 2002.
- [6] K. H. Krippendorff, Content Analysis: An Introduction to Its Methodology, 2nd ed., Sage Publications, Inc, 2003.
- [7] C. Wohlin, P. Runeson, M. Höst, M. C. Ohlsson, B. Regnell, A. Wesslén, Experimentation in software engineering: an introduction, Kluwer Academic Publishers, Norwell, USA, 2000.
- [8] QRA, Qvscribe by qra corp, <https://qracorp.com/qvscribe/>, 8.2.2021. URL: <https://qracorp.com/qvscribe/>.