Measurement-based Quality Assessment of Requirements Specifications for Software-Intensive Systems

Jakub Mund

Technische
Universität
München

Institut für Informatik
der Technischen Universität München

Measurement-based Quality Assessment of Requirements Specifications for Software-Intensive Systems

Jakub M. Mund

Vollständiger Abdruck der von der Fakultät für Informatik der Technischen Universität München zur Erlangung des akademischen Grades eines

Doktors der Naturwissenschaften (Dr. rer. nat.)

genehmigten Dissertation.

Vorsitzender:	Univ.-Prof. Dr. Hans Michael Gerndt
Prüfer der Dissertation:	
1.	Univ.-Prof. Dr. Dr. h.c. Manfred Broy
2.	Univ.-Prof. Dr. Helmuth A. Partsch, Universität Ulm

Die Dissertation wurde am 23.02.2017 bei der Technischen Universität München eingereicht und durch die Fakultät für Informatik am 19.04.2017 angenommen.

Abstract

Requirements engineering (RE) is the process of discovering, negotiating, analyzing, communicating and managing the purpose of an envisioned software-intensive system and is widely recognized as a success factor. To prevent project failures, the suitability of its results, predominantly in terms of the requirements specification, must be ensured. One promising and frequently advocated means to this end is the use of measurement.

Despite the importance of RE and the efforts spent on developing metrics, measurement often fails in practice. According to our understanding and experiences, we see three fundamental reasons prohibiting the effective use of metrics for assessing the quality of the requirements specification, namely that we need a better understanding of (1) *if*, (2) *what* and (3) *how* to measure.

This thesis contributes to those three areas. First, we provide empirical evidence on the significance of requirements specification quality. Specifically, we studied to what extent requirements specifications are created and used for communication, and revealed that it depends on certain criteria, e.g., whether the system is safety-critical or the involved parties collaborated in the past. Furthermore, we observed that factual faults in specifications led to flawed test cases in at least 47% of all cases (depending on the ease of discovering the flaw) while negated sentences, which are commonly purported to impair understandability, did have no effect. We concluded that assessment of the requirements specification is an effective means in specific situations, but we have to carefully select which properties to assess and must not neglect the context of use.

Second, in order to understand what to measure, we studied what properties of the requirements specification are (i) relevant and (ii) measurable. Regarding the former, we contribute a novel and systematic method to define quality based on the activities which depend on the requirements specification, and, based thereon, a comprehensive quality definition model. Regarding the latter, we propose a meta-model which encompasses the necessary concepts associated with measures. Furthermore, we identified 136 measures from 2397 scientific publications in an extensive systematic literature review, which we analyzed in-depth to consistently (re-)evaluate crucial aspects, such as the actually measured attribute, threats for validity, or necessary prerequisites for application. Based on the results of the literature survey and the subsequent analysis, we instantiated the meta-model to obtain a unified and comprehensive assessment model. Moreover, by contrasting the quality definition and assessment models, we contribute an evaluation of (the limitations of) measurability which also helps us to understand the utility of the assessment approach.

Finally, we provide a process model for measurement-based quality assessment of requirements specifications complementary to the assessment model and developed specifically with requirements specification quality in mind. The process model covers planning, assessment and continuous improvement phases, and includes a tailoring procedure to customize it for a specific project.

Our results confirm that measurement-based quality assessment is no silver bullet, but rather constitutes a particular means to provide useful estimations of certain aspects of quality in specific situations.

Acknowledgments

Firstly, I want to express my sincere gratitude to Prof. Dr. Dr. h.c. Manfred Broy, not only for providing guidance on this thesis but also for giving me the opportunity to experience research in an environment which provides both individual freedom and stimulating partners from academia and practice to collaborate with. As for my next superior, the bar is set quite high.

Also, I want to thank Prof. Dr. Helmuth Partsch for agreeing to co-supervise this thesis and for providing me with valuable feedback. Those perspectives certainly improved the thesis.

Many important ideas and contributions in this thesis would not have been possible without the involvement of industrial partners and students. In particular, I want to thank Rainer Blessing, Bernhard Schumm and Rainer Wasgint, who are with the Siemens AG, and Michael Klose, Stefan Reith, Thorsten Rahf, Günther Huber, who are with Wacker Chemie AG, as well as the anonymous participants who took part in the survey. My sincere thanks also goes to the students of the requirements engineering lecture who participated in the experiment.

I am very grateful to my fellow colleagues, many of whom already were or became friends, for the inspiring discussions, the fruitful collaborations in research and industry projects, and all the fun we have had in the last six years. In no particular order, I want to thank Benedikt Hauptmann, Maximilian Junker, Andreas Vogelsang (for his equally accurate points on research and the latest gossip), Sebastian Eder (for jointly competing in the parallelization contest), Diego Marmsoler, Georg Hackenberg, Henning Femmer, Antonio "G." Vetro, Wolfgang Böhm (for introducing me to sailing), Veronika Bauer, Silke Müller, Florian Grigoleit, Thomas Kolfer (for giving me, and my neighbors, an impression how Freddie Mercury must have sounded like), Jonas Eckhardt (for being easily convincable that there exists no such thing as *one* beer) and Daniel Mendez (for convincing me there exists no such thing as *one* coffee).

Last but not the least, I would like to thank my family: my parents for supporting me throughout my thesis and my life in general, and my wife Melli and my daughters Lena and Sophie for putting everything in life in perspective and always making me remember what matters the most to me.

Contents

Chapter 1

Introduction

Requirements engineering (RE) is the iterative process of discovering, negotiating, analyzing, communicating and managing the purpose of an envisioned software-intensive system [Nuseibeh and Easterbrook, 2000]. It is widely recognized as a success factor for engineering software-intensive systems [Broy, 2006, Pohl, 2010, Hofmann and Lehner, 2001, Nuseibeh and Easterbrook, 2000] and subject to various studies. For instance, independent studies conducted by The Standish Group [1995] and Verner et al. [2005] report that factors directly related to RE, such as user involvement or incomplete requirements, are among the most significant determinants for project success or failure. An internationally replicated series of surveys which investigates the state of the practice in RE suggests that problems in RE, especially incomplete or underspecified requirements, lead to poor product quality and time/cost overrun, and in turn, to unsatisfied expectations and customers [Méndez Fernández and Wagner, 2014, Kalinowski et al., 2016]. Effective RE, in contrast, leads to improvements in the product quality and productivity of downstream development activities such as testing [Damian and Chisan, 2006]. To prevent project failures, researchers and practitioners alike recognized the need to assure the quality of the requirements engineering process and its results, most prominently in terms of the requirements specification[1]. It is of central interest for quality assurance since it commonly constitutes the agreement between contractors and suppliers,

[1] A (software) requirements specification refers to a systematic representation of the functionality and quality attributes required of the software-intensive system under consideration [Broy, 2006, Glinz, 2011], and related RE work products such as the system vision, context model (domain models), acceptance criteria, goal model or glossary (see, e.g., IEEE Standard 830 [1998], Méndez Fernández and Penzenstadler [2014], Geisberger [2005] for reference models of requirements specification contents)

and is considered the pivotal input for downstream development, e.g., system design and testing, and project-management activities, e.g., cost-estimation and risk analysis.

One particular means to assess and improve quality which gained wide-spread attention in requirements- and software engineering is the use of measurement[2] or metrics. The application of measurements in software development was advocated by influential researchers in the field and well-known approaches in academia and industry. For example, Basili et al. [1994] proposed an approach to define a measurement model based on the systematic derivation of specific measures from business or software development goals on a conceptual level. This approach, known as the Goal–Question–Metric (GQM) paradigm, received widespread attention in software engineering for several decades. The Capability Maturity Model Integration (CMMI, Team [2010]), a process assessment and improvement program which is mandatory for many U.S. Government contracts, sees the use of quantitative measures as the key characteristic of high process maturity. The common expectation is that measuring provides a precise and well-founded assessment of the specification's actual quality, thereby serving as a reference to guide improvement. Indeed, it is even advocated to be essential for improvement, as for example expressed by DeMarco [1986] in his infamous quote *"You can't control what you can't measure"*.

1.1. Problem Statement

Despite the importance of RE for project success and the efforts spent on developing measures, researchers and practitioners independently observed that many measurement programs not only failed, but often also did more harm than good [Kaner et al., 2004, Austin, 1996, Pitts, 1997]. According to our understanding and experiences, we see three fundamental problems prohibiting the effective use of measurement for assessing the quality of the requirements specification:

Problem 1: Insufficient understanding of the significance of documented requirements As mentioned before, problems in requirements engineering, such as incomplete or hidden requirements, are critical factors for project success. Yet, the question how much downstream development activities are actually influenced by the quality of the *documented* requirements, which also includes the question of how much project participants rely on the created artifacts, is not completely answered. However, the subtle distinction between elicited and documented requirements is highly relevant in practice. Since the documentation of requirements and their quality control are labor-intensive tasks, practitioners are often confronted with a trade-off between efforts spent and adherence to schedules on the one hand and the provided quantity and quality of requirements documentation on the other hand. Hence, to understand the significance of requirements specification quality, we need an empirical understanding of the extent to

[2] In general, measurement refers to assigning numbers to empirical observations according to certain rules [Fenton and Pfleeger, 1998]. In the context of quality control, those numbers are then interpreted as the magnitude to which the specification fulfills certain quality criteria, and used to support decision making.

which requirements specifications are created and used in practice, as well as the actual impact of their quality on downstream development activities.

This understanding is crucial to make profound decisions about *if and when to measure* requirements specification quality based on its significance and depending on the project setting. However, in practice, we often see measurement applied blindly based on diffuse best practices, exaggerated expectations based on anecdotal successes, or to confirm with prescriptive standards such as CMMI.

Problem 2: Lack of a precise and well-founded quality definition Various researchers [Kotonya and Sommerville, 1998, Wiegers, 1999, Robertson and Robertson, 2006, Rupp and SOPHISTen, 2007, Pohl, 2010] and standards [IEEE Standard 830, 1998, ISO/IEC 29148:2011, 2011] propose a notion of requirements specification quality. They have in common to provide a set of characteristics a good specification must possess, e.g., it needs to be precise, complete, unambiguous, verifiable, etc.

According to Femmer et al. [2015], we can classify those characteristics into two types: On the one hand, characteristics such as precise, complete or unambiguous, refer to inherent properties of the specification itself. However, those properties remain imprecise[3] regarding a detailed definition of what constitutes them (e.g., what exact properties render a requirements specification precise or unambiguous?), the part of the specification that they apply to (e.g., which information must be expressed using quantitative values? [Davis et al., 1993a]), or both. In addition, the rationale (what problems does this cause under what circumstances?) remains implicit and sometimes questionable. On the other hand, characteristics such as verifiability merely name activities to be performed with the requirements specification [ISO/IEC 29148:2011, 2011]. While for such characteristics the rationale is clear (e.g., negative consequences on verification activities and, therefore, software quality), the actual properties of the specification to be measured remain implicit and vague, also since they strongly depend on the activities' context; for instance, how suited the requirements are for testing also depends on the domain knowledge of the tester or the testing approach used.

This lack of a precise and well-founded quality definition obfuscates *what should be measured*, resulting in not only neglecting measurement of crucial characteristics but also, and arguably more harmful, in measuring irrelevant characteristics.

Problem 3: Lack of understanding about measurability and its limitations Measures are widely accepted and routinely applied in many aspects of daily life in organizations (e.g., manufacturing, finances). This can make them appear suitable to also measure any desired quality of the requirements specification. However, the ability of measures, including semi-automated and manual ones, to assess certain aspects of quality is limited, at least based on the current understanding of the matter [Pfleeger, 2008, Méndez Fernández et al., 2014]. For instance, certain measures require satisfaction of specific conditions to be applicable (e.g., presence of a tool or external documentation)

[3] The attentive reader may have noticed the irony here.

or to yield valid results, i.e., assess quality with adequate certainty (e.g., that a dictionary adequately reflects domain vocabulary).

Unfortunately, the measures proposed in literature often inadequately reflect on their prerequisites and limitations, if at all. In particular, in many cases, the reported measures actually measure an attribute more or less closely related to but nonetheless different from what they purport to measure without pointing out this discrepancy or characterizing the relationship. Furthermore, those measures are described inconsistently regarding the information provided and the terminology used. This lack of a unified body of knowledge hampers the ability of practitioners to identify and compare suitable measures for their quality criteria of interest.

Consequently, without a thorough knowledge of measures and their applicability and validity, we lack a sound understanding of *what can be measured*. This leads to situations in practice in which the attribute perceived to be measured differs from the one actually measured, leading to false conclusions and, in turn, poor decision-making. Ideally, this would be prevented by a systematic approach to quality assessment which supports a careful and sound interpretation of measurement results.

We summarize the problem statement as follows:

Problem Statement: We need a better understanding of if, when, and what to measure in order to assess the quality of the requirements specification, and a method to carefully and soundly interpret the measurement results.

1.2. Contributions

This thesis contributes novel thinking and techniques to the three areas described in the problem statement, as detailed below. Fig. 1.1 provides an overview of the contributions, their addressed problems, dependencies and role in this thesis, and the chapter in which they are presented.

Contributions regarding the Significance of the Requirements Specification We provide a better understanding of the significance of documented requirements and their quality by contributing empirical evidence obtained from two complementary studies, each emphasizing different aspects.

First, we conducted a survey with professionals from different application domains to explore to what extent requirements specifications are created and used to communicate elicited requirements to downstream development in practice. In particular, we also studied if and to what extent it depends on certain project characteristics such as the team size, geographical distribution of team members, or whether or not the contractor and client previously cooperated.

Second, we studied the impact of the quality of documented requirements on downstream development activities, provided a requirements specification is created and used for communication, by means of a controlled quasi-experiment. The experiment provides quantitative data on the extent to which certain quality defects of use-cases propagate

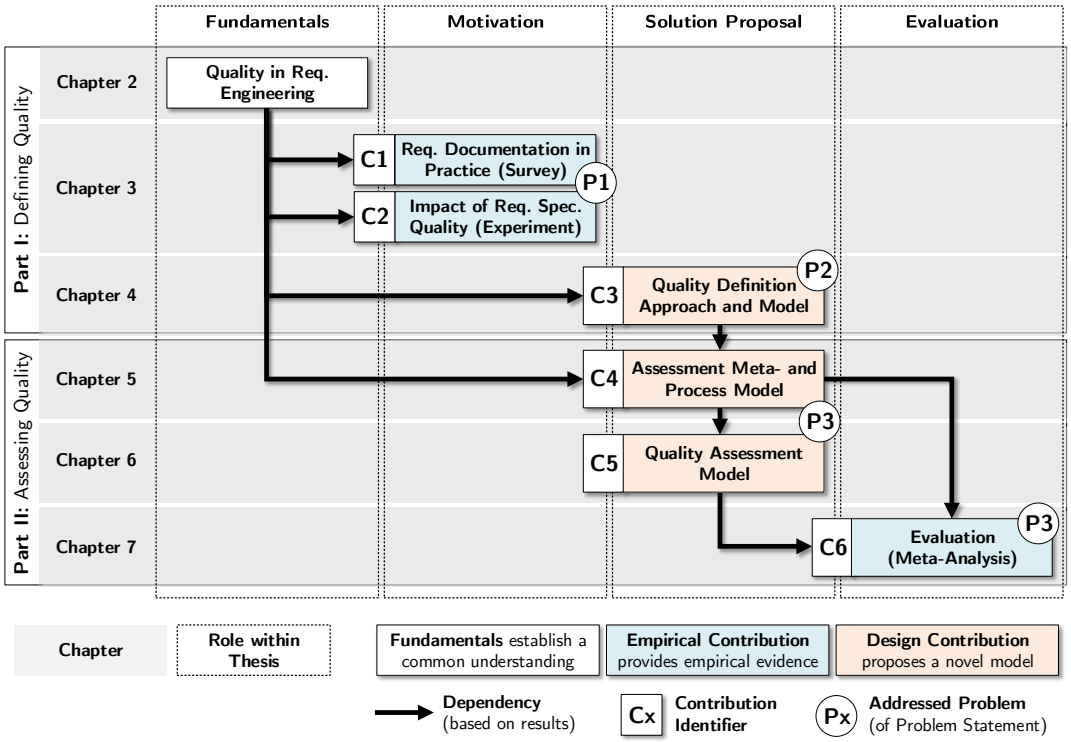


Figure 1.1.: Overview of the thesis' main contributions and their addressed problems, dependencies, informal roles, and organization according to chapters.

to system testing. Specifically, we study the effects on system testing in terms of the quality of the test-cases itself, i.e., the presence of certain flaws in the derived test-cases, as well as the perceived difficulty of deriving those test-cases from use-cases, which we interpret as an indirect indicator for required efforts.

Based on the results of both studies, we critically reflect on the significance of requirements specification quality and discuss implications for its assessment.

Contributions to Quality Definition To obtain a precise and well-founded understanding of what constitutes quality of requirements specifications, we contribute a novel, systematic method to define quality based on the activities which depend on the requirements specification, and a comprehensive quality definition model which can be further customized for a particular project.

First, we propose a method called activity-based RE quality modeling (ABRE-QM) to identify those attributes of a requirements specifications which are relevant for downstream development based on an analysis of the specification's audience and their particular activities. Essentially, this method adapts the idea of activity-based quality [Wagner et al., 2012] to requirements engineering by specifying a meta-model and an exemplary

process how to derive quality attributes from the requirements specification. As we have shown in Femmer et al. [2015], this method is applicable to define quality for a specific company as well as for (precisely specified) software process models.

Second, we contribute to a holistic, precise and well-founded understanding of requirements specification quality by means of a comprehensive quality definition model and an associated customization procedure. This model is the result of a thorough analysis of the qualities proposed in literature and the application of the ABRE-QM approach to analyze the activities of the (Rational) Unified Process as described by Jacobson et al. [1999]. This description qualified for our purpose since it (i) describes the activities in sufficient detail to enable the analysis, (ii) includes the most common activities required for software development, in particular for information systems, and (iii) explicitly specifies the inputs and outputs for each activity. The identified quality attributes are precise since they refer to detailed properties of the specification (e.g., whether input values in use-cases are specified using quantitative numbers) and sound since they specify the impact on the downstream development activities or the corresponding work results (e.g., the quality of test-cases). Last but not least, we outline a tailoring procedure to customize the quality definition model by selecting and adapting the quality attributes for a specific context of use, e.g., a project.

Contributions to Quality Assessment To understand measurability of requirements specification quality and its limitations, we propose a quality assessment approach and model for requirements specifications. This assessment model extends the quality definition model described above by integrating assessment instruments in terms of concrete measures.

First, we provide a framework for measurement-based quality assessment of requirements specifications. Essentially, this framework consists of a meta-model to specify quality assessment models and a complementary method to apply such a model to determine the quality of a requirements specification. The meta-model is holistic, in the sense that it encompasses and relates the necessary concepts associated with measures, as confirmed by practitioners, and provides a unified body of knowledge for measures. The process model covers planning, assessment and continuous improvement when implementing such an assessment model, and includes a tailoring procedure to customize the quality assessment model by considering the applicability and validity of the available measures for the specific project. Both the meta-model and process model were developed specifically with requirements specification quality and its inherent uncertainty in mind.

Second, we conducted an extensive systematic literature review and subsequent analysis to obtain a comprehensive set of measures which constitute concrete measurement instruments in a quality assessment model. The review examined 2397 scientific publications, thereof 166 in-depth, resulting in 136 measures which were systematically analyzed and embedded into the assessment model. Specifically, we provide a consistent assessment of measurability since we re-evaluated crucial aspects, in particular the threats for validity, for any identified measures. Furthermore, the model provides a refined view on

the relationship between the measured property and the associated quality attribute, the measured scope of the specification, and explicitly specifies the necessary prerequisites that must hold for the measure to be applicable.

Finally, we evaluated the adequacy of the quality assessment model by means of a meta-analysis. In particular, we contribute a qualitative characterization of (i) the context in which the assessment model is applicable, (ii) the limitations regarding the metrics' validity, and (iii) the extent the model provides actionable feedback. Furthermore, since our quality assessment model reflects the current state-of-the-art of requirements quality metrics, we also discuss limitations to measurability of requirements specification quality on a more general note.

Delimitation In general, quality of requirements specifications pervades many aspects of systems- and software engineering and is therefore too broad to be addressed exhaustively within the scope of a thesis. In this thesis, we deliberately concentrate on software development and direct assessment of the specification, since we regard those two aspects as fundamental for requirements specification quality and therefore constitute the basis for future research:

- **Quality beyond software development:** We limit our view of quality of requirements specifications by focusing on software development activities for which it serves as an essential input. Consequently, we neglect characteristics which are (only) relevant for activities beyond development, such as project management activities (e.g., cost estimation or project planning) and activities related to other phases in the software life-cycle, e.g., maintenance. While an analysis of activities specific for certain classes of software-intensive systems, e.g., hazard analysis for safety-critical systems [Ericson et al., 2015], is out of scope of this thesis, we discuss this particular direction of future research in the outlook (Chap. 8, pp. 203).
- **Measuring entities beyond the requirements specification** We only consider measures for which the requirements specification itself is the primary entity of measurement. In particular, this work does neither study the assessment of the requirements specification quality based on characteristics of the requirements engineering process, nor based on the quality of artifacts created during requirements engineering other than those contained in a requirements specification, e.g., minutes taken for meetings.

1.3. Overview of the Approach

In this section, we clarify the position of this thesis, describe how the individual contributions described in the previous section work together to form a coherent quality assessment approach, and how we envision it to support software- and systems development.

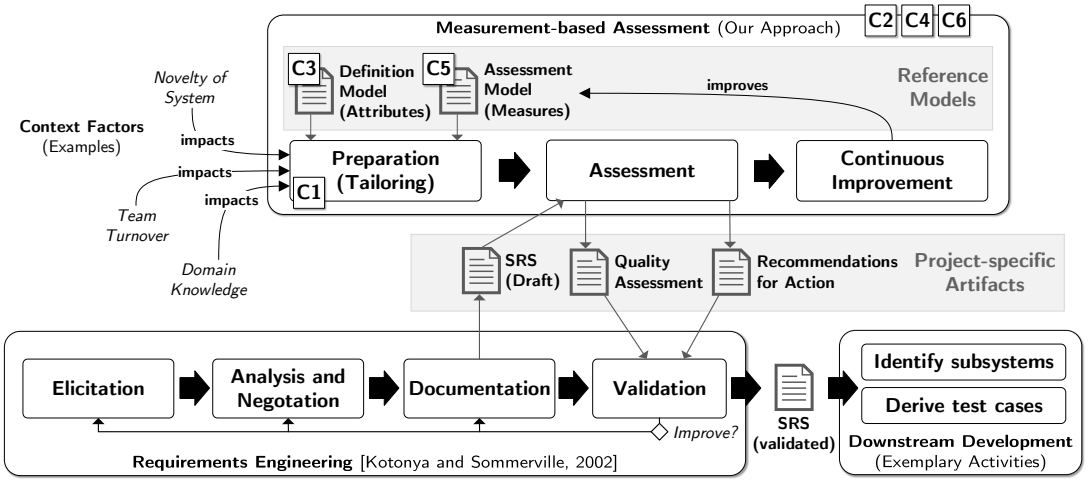


Figure 1.2.: Overview of the quality assessment approach in this thesis, and its envisioned integration in requirements- and software engineering

Position of this Thesis We strictly understand requirements engineering as a means to an end in software- and systems development, with the requirements specification being a medium to foster a valid understanding of the capabilities and characteristics to be provided by the system among its audience (e.g., architects or testers). Therefore, we understand the quality of the requirements specification as the fitness of its physical representation, syntax, and semantics for downstream development. Moreover, we consider this fitness to depend on the context of use, e.g., who performs those activities, how novel the system to be developed is, or if the team members change frequently. Finally, we attempt to approach measures purported to assess this fitness from an utterly rational stance, neither advocating nor demonizing them per se, but with the intention to provide a sound understanding of their uses and limitations for quality assessment. As confirmed by our results, measurement-based quality assessment is no silver bullet, but rather constitutes a particular means to provide adequate estimations of certain aspects of the requirements specification's fitness in specific situations.

Measurement-based Quality Assessment Approach Essentially, our assessment approach is based on the proposed process model and consists of three main phases, as illustrated in the top half of Fig. 1.2. First, in the *preparation* phase, the assessment approach is tailored for a specific context of use, and the necessary means for subsequent assessments are implemented, both technically and in the development process (e.g., establishing quality gates and assigning responsibilities). In particular, this phase seeks to determine the properties of the requirements specification which impact downstream development, and based thereon, identify, adapt and implement suitable measures. To this end, the quality definition and assessment model proposed in this thesis serve as a reference to be further customized for the concrete context of use. Furthermore, our

approach includes an evaluation of the relevance of the requirements specification (based on the empirical results of the expert survey), and identifies relevant but immeasurable properties of the specification. This phase must be performed at least once (e.g., at the start of a specific project) and whenever the context changes significantly (e.g., when outsourcing development activities in market-driven product development).

During the *assessment* phase, individual assessments for a concrete requirements specification are obtained. For each assessment, the measures identified and implemented in the preparation phase are applied to a requirements specification. The obtained assessment is not only a quantitative value, but also includes an interpretation to what extent the particular specification possesses the relevant quality attributes, and the effects those attributes have on downstream development in terms of required effort and quality of results. In addition, measures provide recommendations for actions, i.e., an advice on which tasks to perform in order to improve quality, if possible.

Finally, we acknowledge the learning curve associated with measurement in practice, and therefore recommend a retrospective evaluation to foster *continuous improvement*. This can, for instance, be accomplished by joint workshops capturing experiences made and comparing the assessment results to the actual project or product performance in order to improve future assessments, similar to touch-down meetings in project management.

Integration into Requirements- and Software Engineering Finally, we provide an overview of how we envision our approach to support requirements- and software engineering, as illustrated in the bottom half of Fig. 1.2. Kotonya and Sommerville [1998] understand requirements engineering as the iterative approach of eliciting, analyzing and negotiating requirements, and documenting the results of those activities in a (draft) requirements specification which is subject to validation. Based on the results of validation, the requirements specification is either passed to downstream development, or further improved by repeatedly conducting (some of) the aforementioned activities.

The main application of our approach is certainly as part of requirements validation. After the initial preparation, our approach can be applied to the requirements specification at hand, i.e., documented requirements typically specified at various levels of abstraction (ranging from abstract goals to detailed requirements), to obtain results specifically designed to fit requirements validation: the assessment, i.e., an estimation to what extent the requirements specification is suited for and the consequences it has on downstream development, informs the decision whether to make it available to downstream development. In case it must be revised, recommendations for actions can guide requirements engineers during subsequent iterations of elicitation, analysis, negotiation and documentation.

Moreover, our approach also determines what quality attributes are relevant but not measurable, which can be used to (a) select additional quality assurance means, both constructive and analytical, and (b) used for estimating the remaining risks.

1.4. Outline

The remainder of this thesis is structured as follows: Part I is concerned with defining quality. First, in Chap. 2, we introduce the fundamental notions of quality in requirements engineering, define the terms used in this thesis, and discuss the notion of quality with respect to requirements specifications in more detail. Subsequently, in Chap. 3, we present two complementary empirical studies on the significance of the documented requirements on downstream development activities, and discuss the results of both studies regarding the implications for quality assessment. In Chap. 4, we provide a quality definition model which is based on an approach to define quality for requirements specifications by systematically analyzing the activities which depend on it, and outline a tailoring procedure to customize it.

In Part II, we shift the focus to the assessment of quality based on the results from the first part. To this end, in Chap. 5, we introduce the necessary concepts for assessment models of requirements specifications by means of a conceptual meta-model and based on the quality definition model of Part I, and propose a process model to implement this assessment model. In Chap. 6, we provide a concrete assessment model by integrating state-of-the-art metrics as measurement instruments, and describe how to customize and apply the model for a given project. We then evaluate the resulting model in Chap. 7.

Finally, in Chap. 8, we first summarize our results and contributions before outlining possible directions for future research.

Previously Published Material

This thesis is partly based on previously published material [Mund et al., 2015, Femmer et al., 2015, Méndez Fernández et al., 2014]. The author of this thesis substantially contributed to all contents of previously published material which this thesis is based on.

Part I.

Defining Quality

Chapter 2

Quality in Requirements Engineering

This chapter introduces and relates fundamental principles, concepts, and terms associated with the different notions of quality in the context of requirements engineering. The motivation for this is twofold. First, by clarifying key terms used throughout this thesis, we aim to establish a common and clear understanding for the remainder of this thesis. Second, by relating those terms and associated notions to each other, we are able to put the quality of the requirements specification, which is the focus of this thesis, into a broader perspective, thereby providing a basic understanding of its significance and limitations within requirements and software- and systems engineering. Yet, for the sake of brevity and clarity, we refrain from providing exhaustive characterizations, and provide references to related work to be consulted by the interested reader instead.

The remainder of this chapter is organized as follows. First, we define central terms regarding quality in the context of requirements engineering, with a particular focus on requirements and their documentation in terms of requirements specifications, in Sec. 2.1. Next, we introduce the fundamental quality framework by Lindland et al. [1994] in Sec. 2.2 and subsequently apply it to requirements (specifications) from a knowledge-centric and an artifact-centric viewpoint in Sec. 2.3.1 and 2.3.2, respectively. The relationship between the resulting views is discussed in Sec. 2.3.3 in order to understand its significance. Finally, we provide a summary of central terms, notions and results in Sec. 2.4.

2.1. Fundamental Terms and Definitions

Among the many meanings associated with the term *quality*, in this thesis we understand it as:

Quality:

(1) Characteristics of an entity that bear on its ability to satisfy stated or implied needs. [ISO/IEC 9126-1, 2001]

(2) Extent to which an entity possesses the characteristics as in (1).

We refer to definition (1) as *descriptive* since it focuses on the totality of characteristics (also called *quality attributes*) of an entity under consideration. In general, an entity may refer to a system, service, product, artifact, process, person, organization, et cetera [Glinz, 2011]. To gain a better understanding of quality, we are interested in the descriptive meaning of quality as we seek to identify and precisely describe those characteristics. Hence, it will be the main focus of Part I of this thesis.

In contrast, quality also conveys an *evaluative* meaning referring to (2), i.e., the extent a specific entity actually possesses those characteristics. In other words, it denotes the extent a specific entity is able to satisfy stated or implied needs. This understanding of quality is also commonly referred to by the phrases *freedom from deficiencies* and *fitness for use*, as coined by Juran and Godfrey [1999]. The latter phrase emphasizes that both the characteristics (descriptive meaning) and the extent to which they are fulfilled (evaluative meaning) strictly depend on the actual use of the entity. The evaluative meaning becomes the main focus in Part II of this thesis, since a *quality assessment* indeed refers to an estimation of quality in the evaluative sense. Note that neither definition conveys the colloquial connotation of quality with *goodness* or *excellence* [Glinz, 2011].

Consequently, to understand the notion of quality in requirements engineering, we must clarify the needs it aims to satisfy. Unfortunately, the term *requirements engineering (RE)* is used inconsistently in software- and systems engineering. For the remainder of this thesis, we do not refer to its meaning as a science or discipline, but rather as a process within software development:

Requirements Engineering [Nuseibeh and Easterbrook, 2000]: *The process of discovering the purpose a software system is intended for, by identifying stakeholders, and their needs, and documenting these in a form that is amenable to analysis, communication, and subsequent implementation.*

While this characterization of RE is certainly selective and neglects many important aspects of RE, e.g., the iterative nature of the process [Broy, 2006] (see, e.g., Pohl [2010] or Robertson and Robertson [2006] for a comprehensive treatise of RE), it highlights the role of RE within software development: RE is no end in itself but must support downstream

development activities (e.g., architectural design or system testing) so that the likelihood of success of the development endeavor is maximized. Traditionally, success of software development is considered a function of *costs*, i.e. the efforts required to build the system[1], *time*, i.e. the adherence to deadlines and schedules, and *software quality*[2] as the degree to which the software product satisfies stated and implied needs [ISO/IEC 25010:2010, 2010]. Gorschek and Davis [2008] discuss the impact of requirements engineering on two additional dimensions, namely the impact on the company as a whole, e.g. the adherence to strategical goals or the potential for innovation, and society in terms of (both positive and negative) externalities implied by the system under consideration. Here, we refrain from these implications for both the sake of simplicity and the lack of a profound understanding of those effects. Empirical evidence obtained from independent studies support the understanding of RE as a means to an end in software development. In particular, specific characteristics of RE, such as incomplete requirements, lack of user involvement or dysfunctional communication, were identified as critical factors for project success [Kamata and Tamai, 2007, Kujala et al., 2005] or failure [Méndez Fernández and Wagner, 2014, The Standish Group, 1995].

Depending on the type of entities of interest, we distinguish between *process* and *product quality* of requirements engineering. While the former refers to characteristics of carrying out the activities, e.g., the duration of workshops held during elicitation, the latter is concerned with the results obtained, most prominently the requirements:

Requirement [IEEE Standard 610.12, 1990]: *A condition or capability (i) needed by a user to solve a problem or achieve an objective or (ii) that must be met or possessed by a system or system component to satisfy a contract, standard, specification, or other formally imposed documents.*

Essentially, requirements refer to a prescriptive characterization of the system to be developed, and may range from abstract goals that the system as a whole shall achieve (e.g., the system shall be marketed worldwide) to detailed requirements of features or functions (e.g., if within 2 seconds of continuous monitoring, the CAN message `BATT` remains below 8V, the door control unit is put into sleep mode and the CAN message `SLEEP_NOW` is transmitted)[3].

[1] Actually, costs should consider the complete lifecycle of software-intensive systems, including costs beyond development (e.g., maintenance) which are often neglected. Recently, more attention of regarding operation and maintenance during development was advocated by the *DevOps* paradigm [Roche, 2013].

[2] Agarwal and Rathod [2006] report that project stakeholders agree that attaining *scope* of development, i.e., the realization of all intended system capabilities and conditions, is the highest determinant of success.

[3] Both examples were taken from an example specification for a *car door control unit* [Houdek and Paech, 2002] which is representative for requirements specifications in the automotive industry, and translated by the author.

In order to clarify the meaning of requirements quality[4], we must first introduce the following terms:

Audience All actors who require knowledge of the systems' requirements in order to perform their assigned downstream development activities. Exemplary roles are testers, developers, architects, or UI designers. The audience may also include technical actors such as analysis tools or code generators. We refrained from the term *stakeholder* to avoid ambiguity since we need to differentiate between the stakeholders of the system (e.g., customers, users) and the stakeholders of requirements (e.g., testers, developers), i.e., the audience.

Desired system A system which meets the stakeholders' stated or implied needs, i.e., is of desired quality [ISO/IEC 25010:2010, 2010]. Also referred to as the *right* system [Robertson and Robertson, 2006, p. 5]. Note that due to the inclusion of implied needs, the desired system may differ from the system requested by the stakeholders, e.g., if they are unaware of their actual needs or the requirements are based on flawed assumptions about the domain.

Least effort With the least consumption of resources and time.

Based on the definitions of requirements and quality, the understanding of RE as a means to and end that contributes to success in software engineering, we propose the following intensional definition[5] for the quality of requirements:

Requirements Quality: *The characteristics of requirements that bear on its ability to inform the audience in order to build the desired system with least effort.*

The *requirements specification* is the documented representation of requirements, independent of the actual form of storage (e.g., on paper or stored electronically in a word-processor or specialized requirements authoring tool) [IEEE Standard 610.12, 1990]. Often, it also contains additional RE work products such as the system vision, context model (domain model), acceptance criteria, goal model or glossary (see Fig. 2.1 for an overview of the contents considered part of the requirements specification in this thesis as described in Méndez Fernández and Penzenstadler [2014], or consult IEEE Standard 830 [1998], Geisberger [2005] for additional reference models). In analogy to requirements quality, we refer to its quality as:

[4] The quality of requirements must not be confused with quality requirements; while the former refers to characteristics of requirements describing their suitability (e.g., that requirements are unambiguous), the latter refers to the subclass of requirements which describe characteristics of the system itself (e.g., that the system operates reliably or is easy to use.)

[5] For an extensional definition specifying the individual characteristics by means of a quality definition model, we refer to Chap. 4

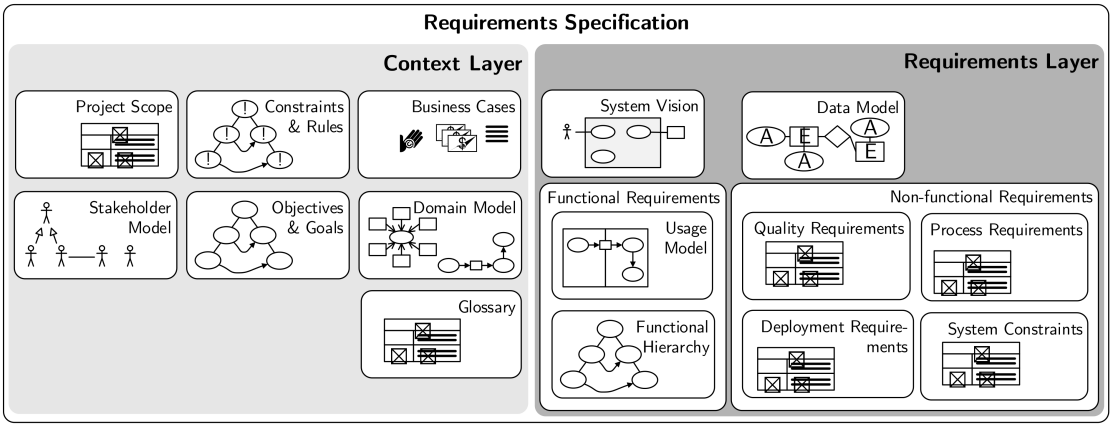


Figure 2.1.: Contents of requirements specifications, further organized in a context and requirements layer, according to Méndez Fernández and Penzenstadler [2014]

Requirements Specification Quality: *The characteristics of the requirements specification that bear on its ability to inform the audience in order to build the desired system with least effort.*

In the remainder of this chapter, we will focus on requirements and requirements specification quality. However, before proceeding to discuss those notions in further detail, we briefly discuss their relationship to RE process quality:

Side Note: RE Process Quality On the one hand, several process characteristics immediately influence the success of the development endeavor, e.g., the time spent for RE activities are part of the project total time on its own, which in turn influence secondary success factors, e.g., time-to-market. On the other hand, process and product quality form a complex interplay. For instance, certain quality characteristics of the elicitation process, e.g., the stakeholders present during elicitation workshops and therefore the efforts spent, may impact the extent the relevant requirements are identified and documented. In turn, the extent requirements are documented may impact the applicability and effectiveness of other process steps, e.g., a certain analysis based on formal methods. Consequently, to comprehensively understand quality in requirements engineering, we must not only understand quality of the requirements (documentation) and the RE process individually, but also the relationship between those. Figure 2.2 shows a schematic illustration of the discussed notions of quality in requirements engineering and their relationship with selected project success criteria in a very simplified manner (colored elements designate the main focus of this chapter). RE process (e.g., costs) and requirements (specification) quality not only mutually impact each other, but also impact project success, traditionally characterized as costs, quality, and time, which in

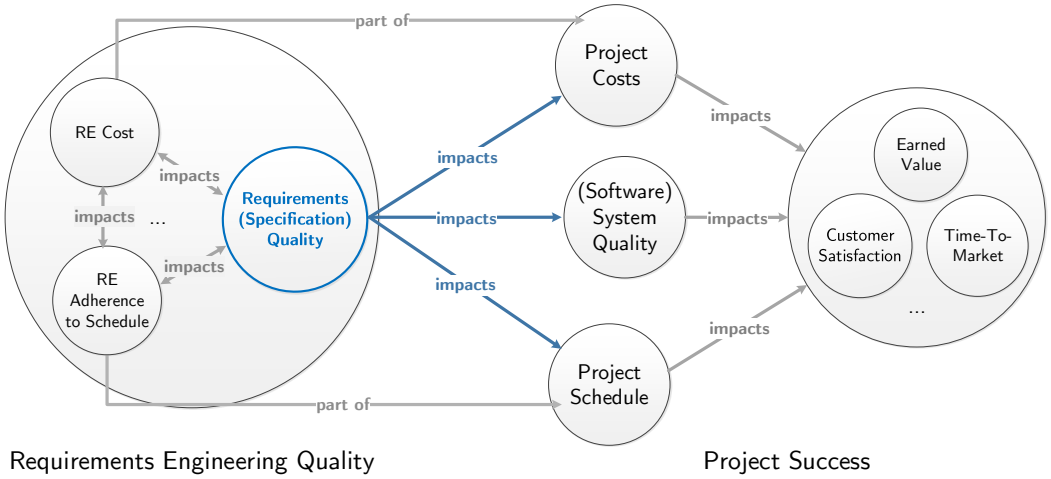


Figure 2.2.: Selected RE quality characteristics and their relationship to project success factors (vastly simplified, colored elements designate the scope of this thesis)

turn impact (secondary) characteristics such as the time-to-market or customer satisfaction. While considered well beyond the scope of this dissertation, one must remember that process quality constitutes another important aspect of RE quality. The interested reader should consult Sec. 8.2 of the conclusion (Chap. 8) for future research in this direction.

2.2. A Framework for Requirements Specification Quality

In order to further discuss and relate the notions of quality regarding requirements in general and the requirements specification in particular, a refined view on the characteristics that constitute those is required. Therefore, we shall use the quality framework originally proposed by Lindland et al. [1994] and later applied to requirements specifications by Krogstie et al. [1995a,b], Krogstie [1998]. Essentially, the framework decomposes requirements specification quality according to semiotic theory[6], leading to the notions of *syntactic*, *semantic* and *pragmatic* quality. Herein, the contribution of the semiotic dimensions is twofold. First, it constitutes a decomposition criterion for requirements specification quality, thereby providing a (coarse) classification of the requirements specifications' characteristics according to semiotic dimensions. Second, any characteristic not effecting any semiotic dimension is not considered a quality characteristic. Hence, the semiotic dimensions also provide an understanding of the role the requirements specification plays in informing the audience.

[6]Semiotics is the discipline concerned with theory of signs, pioneered by Charles Sanders Pierce (1839-1914).

The framework is based on the abstract notion of *statements* which refer to arbitrary expressions of some language [Krogstie et al., 1995a]. Hence, what actually constitutes a statement must be defined with respect to the language used. For instance, text expressed in natural language may consider sentences as statements, while for a graphical modeling language such as a UML class diagram, a statement may refer to model elements, i.e., concrete instances of meta-model elements such as a class, an attribute, or a relationship. The following sets are defined as a subset of the set of all possible statements:

- $\mathcal{L}$, the language extension, i.e., the set of all statements that are valid according to the vocabulary and syntactic rules of a language. In practice, more than one language is used and the set of valid statements is typically infinite.
- $\mathcal{D}$, the domain, i.e., the set of all statements that are correct and relevant for solving the problem. It represents the 'ideal' knowledge of a particular problem to be addressed by the system under consideration. It can be thought of as an omniscient oracle specifying the (most) desired system for the problem at hand.
- $\mathcal{M}$, the model under consideration, i.e., the set of all statements made about the system under consideration elicited from the stakeholders by the requirements engineers and documented in the requirements specification. It may also contain syntactically invalid statements, i.e., $\mathcal{M} \not\subseteq \mathcal{L}$.
- $\mathcal{A}$, the audience interpretation, i.e., the set of all statements the audience think are true about the model. Recall that audience refers to all social and technical actors who need to understand the specification in order to perform development activities, e.g., domain experts, designers, or code generators.

2.2.1. Syntactic, Semantic and Pragmatic Quality

With respect to the sets introduced in the previous section, the following high-level quality characteristics for a model $\mathcal{M}$ of the system under consideration were proposed by Lindland et al. [1994] and later extended by Krogstie et al. [1995a]:

Syntactic Quality Krogstie et al. [1995a]: *The extent to which the model is contained in the language. The set of syntactic errors is* $\mathcal{M} \setminus \mathcal{L}$.

The contrary characteristics, i.e., the extent all statements allowed according to a language's vocabulary and grammar are contained in the requirements specification, is not regarded a quality characteristic since it is expected not to influence the ability to inform the audience.

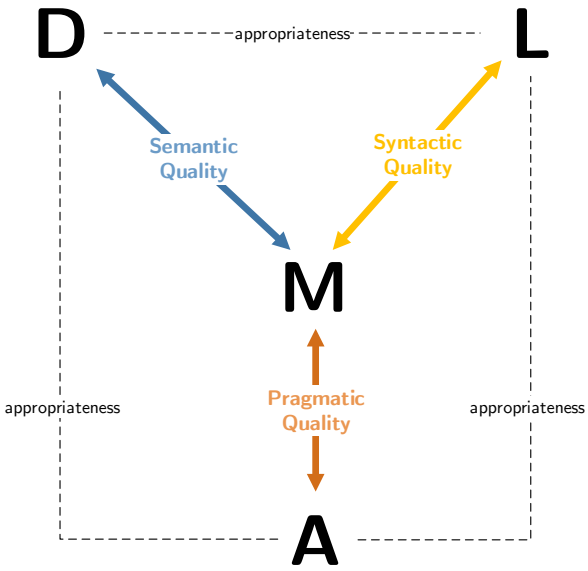


Figure 2.3.: Quality framework based on semiotic theory according to Lindland et al. [1994]

Semantic Quality Krogstie et al. [1995a]: *The extent the system specified in the requirements specification, i.e., the model $\mathcal{M}$, and the desired system, i.e., the domain $\mathcal{D}$, correspond. If $\mathcal{M} \setminus \mathcal{D} \neq \emptyset$, the system described in the model deviates from the desired system and contains incorrect statements; if $\mathcal{D} \setminus \mathcal{M} \neq \emptyset$, the model is incomplete.*

Pragmatic Quality Krogstie et al. [1995a]: *The extent of correspondence between model and audience interpretation, i.e., the extent to which the model has been understood. If $\mathcal{A} \neq \mathcal{M}$, the comprehension of the model is erroneous.*

Fig. 2.3 illustrates the quality characteristics as relations between the model on the one hand, and the domain, language and audience interpretation on the other hand. However, those characteristics are not independent in general, since, e.g., syntactic quality may in turn affect the audiences' understanding, and therefore, also influence pragmatic quality.

Besides the quality of the model itself, the quality framework also mentions indirect factors. Indirect factors are not primarily concerned with the model itself but are pairwise relationship between the language, domain and audience, and are called *appropriateness*; For example, the audience-domain appropriateness is the "extent to which the audience

is able to learn, understand, and use the language" [Lindland et al., 1994]. This is considered an indirect factor since it may impact pragmatic quality but is not directly associated with the model under consideration itself.

2.2.2. Social Quality

The quality characteristics based on semiotic dimensions introduced so far assume a positivist viewpoint on requirements engineering, in particular, a notion of perfect understanding of the problem and its domain as the point of reference for semantic quality. In contrast, several researchers [Holmström and Sawyer, 2011, Krogstie et al., 1995b, Pohl, 1994] approached requirements specification quality from a social-constructivist viewpoint, i.e, based on the premise that requirements are socially constructed by the stakeholders, and which shall, for instance in case of business information systems, form the organization's reality. According to this stance, the validity of requirements depends on the individuals' perceptions of reality and their preferences.

The present quality framework accounts to this viewpoint by introducing the notion of social quality which models the agreement of stakeholders as proposed by Pohl [1994]. To this end, we extend the original model with the set $\mathcal{S}$ which contains all statements agreed on by the stakeholders similar to Krogstie et al. [1995b]. For the sake of simplicity, we model agreement as a binary concept, i.e., we do not consider levels of disagreement, neither for each stakeholder individually nor for a group of stakeholders. Finally, we define social quality as:

Social Quality: *The extent the model is agreed upon the stakeholders, i.e., the extent the model is contained in the agreed model. If $\mathcal{M} \setminus \mathcal{S} \neq \emptyset$, the model contains statements about the system on which the stakeholders disagree.*

Conversely, if $\mathcal{S} \setminus \mathcal{M} \neq \emptyset$, stakeholders agree on statements which are not documented in the requirements specification. However, we refrain from referring to those statements under the term of social quality since it is already part of the framework as semantic completeness, independent of an positivist or constructivist viewpoint[7].

2.3. Significance of Requirements Specification Quality

Based on the fundamental understanding of requirements engineering as a means to an end in software engineering, we subsequently reflect on the significance of the requirements specification in terms of its impact on downstream development from a theoretical point of view. This treatise provides the necessary essential understanding, which also serves as a foundation for empirically studying significance of the requirements specification in practice in Cha. 2.3.

[7] From an positivist viewpoint, $\mathcal{S} \setminus \mathcal{M}$ is either already included in D (if $(\mathcal{S} \setminus \mathcal{M}) \cap \mathcal{D} = \mathcal{D}$), or correctly not included in M (if $(\mathcal{S} \setminus \mathcal{M}) \cap \mathcal{D} = \emptyset$, i.e., the stakeholders agree on factually wrong requirements). From a constructivist viewpoint, $\mathcal{S} \setminus \mathcal{M} \subseteq \mathcal{D}$, since $\mathcal{D} = \mathcal{S}$.

To this end, we consider two distinctive viewpoints on quality, namely a knowledge-centric and an artifact-centric viewpoint, based on the previously introduced quality framework. While the former provides a simplified view on the knowledge possessed, demanded and exchanged between the stakeholders and the audience, the latter extends this view with the requirements specification as the specific means for communication. By contrasting the resulting views, we identify fundamental conditions which determine whether or not the quality of the requirements specification is indeed relevant for downstream development.

2.3.1. Knowledge-Centric Viewpoint on Quality

First, we consider a knowledge-centric viewpoint on requirements quality based on the notion of *mental models* regarding the requirements of the system under consideration:

Mental model [Lim and Klein, 2006, Norman, 1983]: *A mental model is the organized understanding and mental representation of knowledge about key elements of the system under consideration.*

Here, we are not interested in its meaning as a *cognitive mechanism* [Rouse and Morris, 1986] but as *organized knowledge* [Norman, 1983, Webber et al., 2000], i.e., the facts and information acquired by an individual through experience or education. Basically, the term mental model as used in this thesis refers to the model of the system under consideration, specifically its requirements, which a stakeholder or audience member has in *mind*[8], but which is not necessarily communicated or documented. Therefore, it is inherently associated with a human being, reflects the individual's perception of reality [Brunswik, 1956], and may change over time. A mental model is indeed a model since it comprises only the characteristics of the system relevant for the individual, and is limited by their perception. For the sake of discussion, we shall assume that each project role has a single mental model of the systems' requirements.

Mental models of the system's requirements play a crucial role in developing software-intensive systems; several development tasks rely on the person carrying out the task having a valid understanding of the systems' requirements in mind. An architect, for instance, must understand the performance required from the software system in order to design the architecture, and a test developer must have an understanding of the allowed inputs and expected outputs in order to derive useful test cases. Therefore, the mental models of the audience about the system's requirements determine their development activities. Hence, it constitutes the pivotal requirements quality since it determines the requirements' impact on the projects costs and schedule, and the system's quality.

However, in general, the audience does not initially possess a sufficient understand-

[8] It remains unclear how individuals store mental models. While some researchers argue that mental models are part of short-term or working memory and constructed from background knowledge, others argue that mental models exist in long-term memory and are updated continuously (see Langan-Fox et al. [2000] for an introduction)

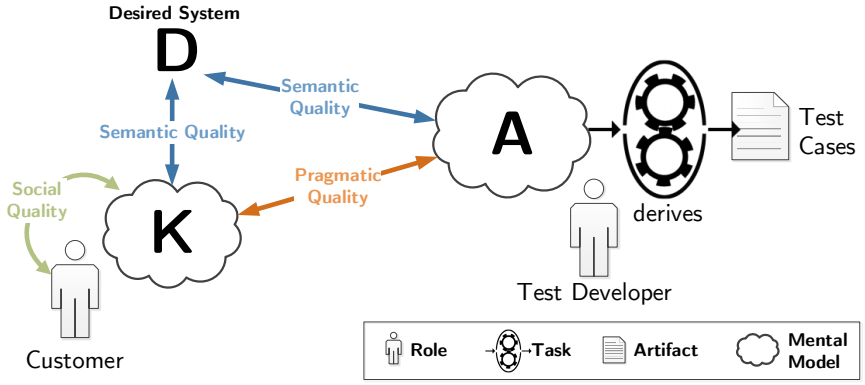


Figure 2.4.: Knowledge-centric viewpoint: Quality characteristics associated with mental models based on the quality framework (simplified)

ing of the stakeholders' needs and the system's purpose[9], but must instead first gain this understanding. Unfortunately, there simply does not exist an oracle describing the desired system, i.e., the set $\mathcal{D}$ in the framework is unknown. Instead, in practice, requirements engineering usually relies on the stakeholders' conception of the system to be build since they are supposed to be (made) aware of their needs and understand the system's domain. Based on the stakeholder's conception, RE then seeks to develop an understanding of the capabilities and characteristics of the system under consideration which is as close as possible to the (most) desired system. Finally, this understanding needs to be communicated to the audience. However, communicating the stakeholders' mental model to the audience is not flawless in general. The possible results include that the audience has an incomplete and/or inconsistent understanding of the requirements, which in turn leads to various problems with costs, schedules, and quality of the system developed, as discussed in Sec. 2.1.

The application of the quality framework to the knowledge-centric viewpoint is illustrated in Fig. 2.4. In this figure, we apply[10] the quality framework to the customer's and test developer's mental model. While the former provides the requirements expected of the system, the latter requires its knowledge to derive test cases for system testing.

[9] Remark: Often, not even the stakeholders initially have a profound understanding about their needs and the system's purpose. This is reflected in a shift of what is understood by the term requirements elicitation over time: While, in the beginnings of software and requirements engineering, elicitation was understood as the process of simply *gathering* requirements from stakeholders, this view has been recognized as deprecated and was changed later on to be understood as the process of *discovering* requirements.

[10] If we view mental models as a network of associations between domain concepts, as customary in social sciences [Langan-Fox et al., 2000], we consider statements to be atomic facts expressed as assertion about this network.

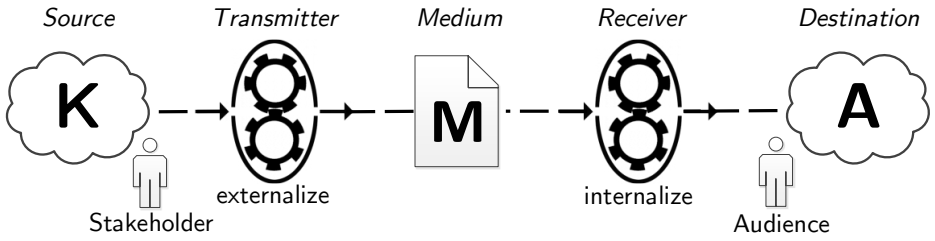


Figure 2.5.: Communication model based on Shannon [1948]

For both mental models, $\mathcal{K}$ (stakeholder *k*nowledge) and $\mathcal{A}$ (*a*udience knowledge), its semantic quality denotes the extent to which the respective mental model resembles the optimal system. In addition, the social quality of the stakeholder's mental model refers his or her sentiments towards their own understanding about the system's requirements, which may be a consensus among many stakeholders. Furthermore, since we assume the customer's understanding is communicated to the test developer, the pragmatic quality refers to the extent of equality between those two models. Note that since we abstract from the actual means of communication, no explicit statements are made in a certain language, and, hence, the concept of syntactic quality does not apply here.

2.3.2. Artifact-Centric Viewpoint on Quality

In this section, we refine the knowledge-centric viewpoint by specifically relying on the requirements specification as the means for communicating the requirements of the system-to-be to the audience.

To this end, we use the well-known communication model proposed by Shannon [1948]. According to this model, an information source wants to transmit information to a destination, which in our case corresponds to transferring the knowledge of the stakeholder to the audience as described in Sec. 2.3.1. Therefore, the stakeholder (information source)[11] first externalizes his or her mental model by formulating statements according to a certain language in the requirements specification. This translation of knowledge into messages and signs is also called codification, and includes the selection of relevant information for the audience as perceived by the stakeholder and the subsequent assignment of codes to this information. Then, secondly, the audience (destination) makes sense out of those statements by interpreting the codes, i.e., the statements in the specification. This way, the meaning can lead to alter the audiences' understanding of the system under consideration. This process is called internalization. Fig. 2.5 illustrates this basic model of

[11] or, in many situations in practice, a requirements engineer familiar with the stakeholder's mental model.

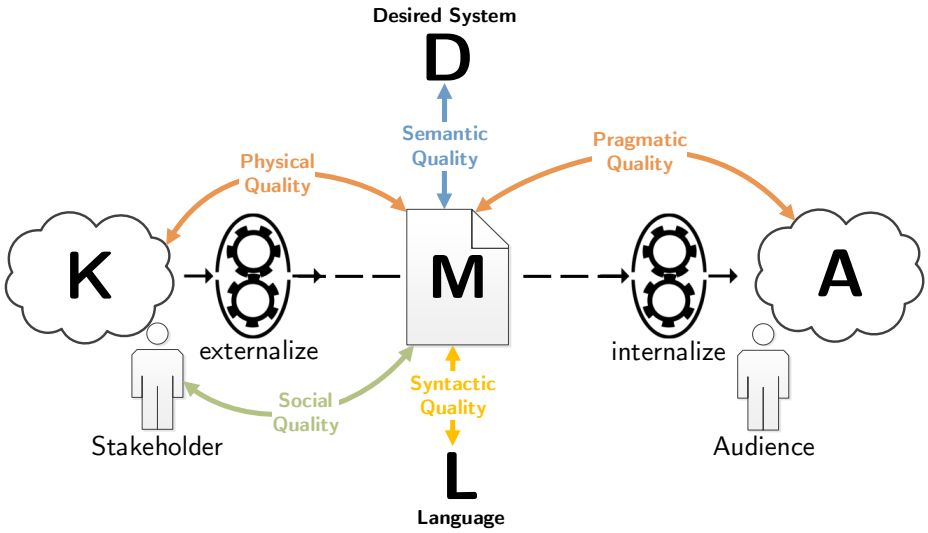


Figure 2.6.: Artifact-centric viewpoint: Quality characteristics associated with the requirements specification as obtained from applying the quality framework

communication in conjunction with the terms originally introduced by Shannon [1948].

By shifting our primary quality focus from the facts and information of mental models to the statements contained in the requirements specification, we obtain artifact-related notions of the quality characteristics compared to the knowledge-centric viewpoint. Fig. 2.6 illustrates this application of the quality framework from an artifact-centric viewpoint. Here, $\mathcal{K}$ again represents the mental model of the stakeholder, while the model of interest, $\mathcal{M}$, denotes its externalization, i.e., the requirements specification. Since this viewpoint concentrates on artifact quality, semantic quality is defined with respect to the statements contained in the requirements specification. Furthermore, compared to mental models, the explicit statements contained in the requirements specification are formulated with respect to a certain language, and, hence, the specification has a certain syntactic quality. Finally, the arguably most important difference to the knowledge-centric viewpoint concerns pragmatic quality. While from a knowledge-centric viewpoint, pragmatic quality is understood as the correspondence between the mental models of the stakeholder and the audience, the artifact-centric viewpoint features two quality characteristics of interest. First, we have the correspondence between the documented statements and the audiences' understanding of it, which we refer to as pragmatic quality as originally intended by Lindland et al. [1994]. Second, we have the correspondence between the stakeholder's mental model and the documented statements in the requirements specification, which we shall refer to as *physical quality*:

Physical Quality [Krogstie, 1998]: *The extent to which the model corresponds to its documentation, and the documented statements are made available to the audience.*

If $\mathcal{K} \setminus \mathcal{M} \neq \emptyset$, not all relevant facts and information known by the stakeholder are documented or made available. If $\mathcal{M} \setminus \mathcal{K} \neq \emptyset$, statements which exceed the stakeholders' knowledge are part of the requirements specification (e.g., due to mistakes during elicitation or documentation) and made available to the audience.

2.3.3. Equivalence of Artifact and Knowledge-Centric Quality

So far, we introduced quality characteristics associated with mental models (knowledge-centric viewpoint) and with the requirements specification (artifact-centric viewpoint) as obtained from applying the quality framework. Moreover, in Sec. 2.3.1, we argued that the quality characteristics associated with the mental models indeed impact downstream development, while neglecting those of the requirements specification. Finally, in this section, we address this shortcoming by studying the relationship between the quality characteristics from both viewpoints. More specifically, we seek to understand which quality characteristics of the requirements specification correspond to those associated with the mental models, and identify conditions for which both coincide. By checking whether those conditions hold, we know whether the quality of the requirements specification indeed reflects the impact on downstream development, and hence, is significant or not.

Fundamental Criteria First, we propose the two fundamental criteria regarding externalization and internalization, respectively:

C1: The requirements specification is of perfect physical quality. In other words, all requirements known by the stakeholder are adequately (preserving meaning) documented and made available to the audience, and the requirements specification contains no other statements.

C2: The audience internalizes all statements of the requirements specification.

Subsequently, we will briefly discuss the relationship between semantic, social respectively pragmatic quality from the artifact- and knowledge-centric viewpoint. This discussion is based on a formal treatment to ensure we identified all criteria and assumptions with necessary rigor. For the sake of brevity, in this thesis we refrain from providing the formal argument but restrict to an informal discussion.

Semantic and Social Quality Intuitively, criterion C1 ensures that the stakeholder's mental model and the requirements specification essentially describe the exact same system by demanding three important properties: all facts of the mental model are documented (completeness) in terms of statements which adequately represent the intended

facts (correctness), while statements which do not refer to a fact of the stakeholder's mental model are not part of the documentation (exclusivity). Therefore, if criterion C1 holds for a certain project, the semantic and social quality of both viewpoints coincide, i.e., the stakeholder's mental model and the requirements specification are of equal semantic and social quality.

Pragmatic Quality As already mentioned in Sec. 2.3.2, pragmatic quality of the stakeholder's mental model from the knowledge-centric viewpoint corresponds to the physical and pragmatic quality of the requirements specification since, essentially, the externalization of the mental model and internalization of the requirements specification constitute the end-to-end communication from the artifact-centric viewpoint. Obviously, the requirements specification must indeed be read and interpreted by the audience, i.e., criterion C2 must hold. However, this criterion alone is not sufficient, but the following assumptions about the relationship between both viewpoints must also hold:

A1: Every fact which is communicated to the audience (by some means) is also documented in the specification.

A2: Flaws in communication are consistent across all employed means. In other words, any form of communication used fails exactly on the same facts, and also results in the same misinterpretations.

If assumptions A1 and A2 and criterion C2 hold[12], we can be sure that the pragmatic quality of the stakeholder's mental model is equivalent to the requirements specification's physical and pragmatic quality, i.e., the mental model of the audience is equivalent to the one obtained by reading and interpreting the requirements specification, independent whether the audience also used additional means of communication or not. Basically, we can distinguish between two alternative scenarios. In the first scenario the requirements specification is the sole means of communication between the stakeholder and the audience. In this case, both assumptions are fulfilled, and the requirements specification's physical and pragmatic quality reflect the pragmatic quality of the stakeholder's mental model, provided the audience read the complete specification (criterion C2). The second scenario is that, in addition to providing a specification to the audience, requirements are also communicated via additional means, e.g., by joint meetings/workshops or bilateral talks. However, in that case, any means used must result in the audience gaining precisely the same understanding of the system as specified in the requirements specification. Otherwise, if undocumented requirements are communicated or alternative means used result in fewer or different misunderstandings, the audience mental models will be different compared to only internalizing the requirements specification, and hence its quality does not reflect pragmatic quality from the knowledge-centric viewpoint.

[12] Technically, it must also hold that the communication medium is free of noise, i.e., the requirements specification only contains statements which result from the externalization of the stakeholder's mental model.

Conclusion: Significance from a theoretical point of view In Sec. 2.3.1, we argued that quality from the knowledge-centric viewpoint, in particular the audiences' understanding of the system under consideration, constitutes the critical RE product quality for downstream development. Furthermore, in this section, we have shown that quality of the requirements specification, as defined by the quality framework, does not universally reflect quality from the knowledge-centric viewpoint, but that it rather depends on certain criteria to be fulfilled.

Interestingly, those criteria are different for semantic and social quality on the one hand, and pragmatic quality on the other hand. Consider, for instance, an in-house software project implementing an accounting module which must comply to a recently introduced law. This project may fulfill C1 but not A1, e.g., because even though the requirements were comprehensively documented due to legal reasons, they were communicated face-to-face in weekly stand-up meetings to the audience. In this case, the requirements specification's semantic quality would indeed be informative, whereas its pragmatic quality would not. Consequently, quality assurance of the former can be justified, while for the latter, not so much. As shall be seen in the next chapter, however, empirical evidence suggests that those cases are rather rare, and instead, the amount of documentation and its use for communication indeed strongly correlate.

We summarize our conclusion by proposing the following working hypothesis:

Working Hypothesis (Significance of Requirements Specification Quality):
*The extent to which criteria **C1** and **C2** and assumptions **A1** and **A2** are fulfilled determines the extent the requirements specification's quality impacts downstream development.*

Therefore, if a specific development endeavor fulfills the above criteria, the quality of the requirements specification constitutes the pivotal RE product quality since it impacts the success of the software development endeavor.

2.4. Summary

In this chapter, we introduced fundamental terms and notions related to quality in requirements engineering which are used throughout this thesis and based on the understanding of requirements engineering as a means-end for developing software-intensive systems. By the term *requirements specification quality*, we refer to the characteristics of the requirements specification that bear on its ability to inform its audience to build the optimal system with least effort. In this thesis, the term *audience* refers all project participants relying on knowing the requirements to carry out their assigned activities, in contrast to the term *stakeholders* which refers to persons or organizations which are impacted by the requirements. In addition to this descriptive meaning, the term also conveys an evaluative meaning in terms of the extent those characteristics are possessed by a particular specification. For other terms introduced in this chapter and throughout the thesis, we refer to the glossary (appendix A).

By looking at the characteristics which constitute the requirements specification's quality in further detail, we distinguish between four relevant classes according to the framework presented in Sec. 2.2. The classes are called semantic, syntactic, pragmatic and social quality, and are concerned with the specified system, the language used, the understanding obtained by its readers and the sentiments and agreement brought forward by the stakeholders, respectively.

Finally, we argued that the significance of requirements specification quality for developing software-intensive systems is a function of two criteria, namely if (i) requirements are comprehensively and thoroughly documented and (ii) the requirements specification constitutes the main information source about the system's requirements for subsequent development activities.

3 Chapter

Significance of Requirements Specification Quality

This chapter scrutinizes the conventional wisdom that quality deficiencies of RE artifacts, most prominently the software requirements specification (SRS) of a software-intensive system, causes problems in downstream development. In fact, a large number of documented project failures are attributed to inadequate requirements engineering (RE) [Kamata and Tamai, 2007, Kujala et al., 2005, Hofmann and Lehner, 2001]. Incorrect, inappropriate or missing requirements, for instance, are supposed to lead to various problems in later phases such as effort and time overrun or an increased effort in acceptance testing [Méndez Fernández and Wagner, 2014].

Yet, the question how much downstream development activities are explicitly impacted by the quality of the documented requirements, which includes also the question how much project participants eventually rely on the created artifacts, remains largely unanswered. However, the subtle distinction between elicited and documented requirements is highly relevant in practice since the explicit documentation of requirements and their quality assurance are labor-intensive tasks, and practitioners, therefore, often face a trade-off between effort and adherence to schedules on the one hand and the provided quantity and quality of requirements documentation on the other hand. While we discussed this matter from a theoretical point of view in the previous chapter, we still lack an empirical understanding of the extent to which SRS are created and used in practice, as well as the extent to which the quality of an SRS impacts downstream development activities that rely on knowing the system's requirements. Accordingly, we formulate two complementary research objectives:

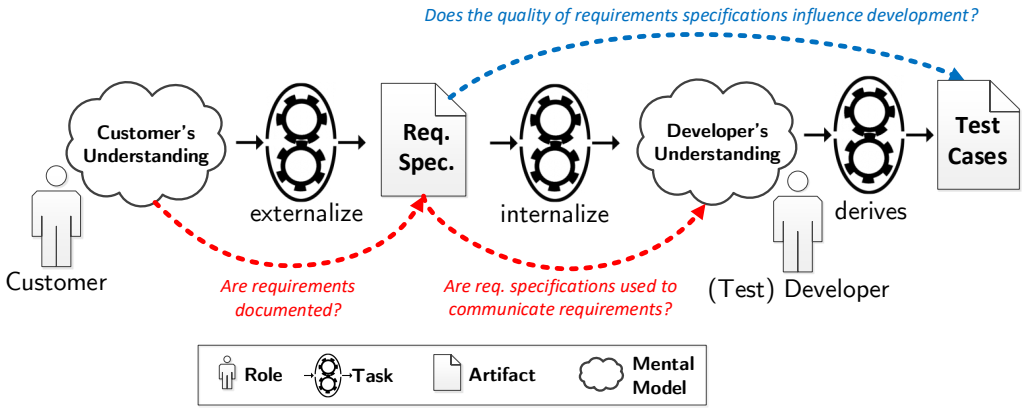


Figure 3.1.: Overview of research objectives: For RO 1 (red), we studied the extent to which requirements are documented and the SRS is used for communication, by means of an expert survey. For RO 2 (blue), we conducted a controlled quasi-experiment to quantitatively investigate the impact of its quality.

RO 1: To what extent and under which conditions are SRS created and used?

RO 2: How and to what extent does the quality of SRS impact downstream development activities?

Fig. 3.1 illustrates the two empirical studies we performed in order to address those objectives. In the first study, we conducted a survey with professionals from different application domains to explore to which extent SRS are created and used in practical environments, both general and in relation to certain project characteristics (**RO 1**). The results of our survey are presented in Sec. 3.1 In the second study, we investigated the impact of SRS quality on downstream development activities by means of a controlled quasi-experiment. Specifically, we quantitatively studied to what extent certain quality defects in use-cases influenced the quality of test-cases obtained by test engineers (**RO 2**). The results are presented in Sec. 3.2. Finally, based on both studies, we critically reflect on the relevance of the SRS and discuss implications for SRS-based quality assurance in Sec. 3.3 before relating our results to existing work and empirical evidence in Sec. 3.4. Finally, we summarize our results in Sec. 3.5.

This chapter is based on previously published material [Mund et al., 2015].

3.1. Significance of the Artifact

In this section, we address **RO 1** by presenting the study design and results of a survey conducted in collaboration with an industrial partner.

3.1.1. Research Questions

We study the questions to what extent and under which conditions (i) requirements are actually documented in the requirements specification, and (ii) used as a means to communicate the requirements to downstream activities within software development such as architecture design, implementation or testing. To this end, we propose the following research questions:

RQ 1-1: To which extent are requirements documented? We examine the degree to which requirements are documented in SRS or comparable artifacts (e.g., product backlogs). We distinguish between two dimensions when documenting requirements. First, we want to know how comprehensively requirements are documented in terms of quantity, i.e. the proportion of documented requirements compared to all requirements identified during requirements engineering. Second, as requirements can be specified with varying level of detail, we want to know in how detailed the requirements are documented.

RQ 1-2: To which extent are SRS used to communicate requirements? As described in Sec. 2.3, the requirements specification also serves the purpose to communicate requirements from stakeholders to various roles in the systems engineering process, e.g., architects, implementers, or testers. With this RQ, we investigate the extent to which the project participants' understanding of the system's requirements is based on the SRS. To this end, we want to know whether and how often a SRS is used as a means for communication within RE as well as the communication of requirements to downstream development. In case it is used, we are further interested in its relative importance compared to other means of communicating requirements in practice.

Is SRS creation and usage related to specific project circumstances? Since we do not expect the SRS to be used equally for all project circumstances (see Sec. 2.3 (p. 25) for a theoretical discussion on this matter), we want to know for both research questions if specific project criteria influence whether requirements are documented (*RQ 1-1*) and whether this documentation is indeed used for communicating requirements (*RQ 1-2*). A detailed description of the criteria considered in this survey is provided in the next section.

3.1.2. Studied Project Criteria

Since we want to study the impact of project criteria on the aforementioned research questions, we must first identify concrete and tangible project characteristics which are supposed to affect the relevance of the requirements specification (see Sec. 2.3 for a discussion on this topic from a theoretical point of view). To this end, the software process models research community provides valuable insights. Effective software process models must be able to adapt to actual circumstances by a process called *tailoring*. In a recent systematic literature review, Kalus and Kuhrmann [2013] identified and structured

project characteristics, called *project criteria*, together with implications for the software process, referred to as *actions*. Based on 127 publications, the authors identified 49 criteria, classified as either *team*, *internal environment*, *external environment* or *objectives*, and 20 actions how the process should respond to those criteria.

For our purpose, we seek to identify only those project criteria proposed the authors which are supposed to specifically affect (i) the extent to which requirements are documented, or (ii) the extent to which the requirements specification is used as a means for communication. We proceeded as follows. First, we identified four out of the 20 actions to be relevant for this purpose, namely:

ED Expand project documentation, e.g., formalized documentation using templates; This action has a *positive* effect on criterion (i), as more knowledge is actually externalized.

RD Reduce documentation, e.g., replace paper-based documentation by stand-ups; This action has a *negative* effect on criterion (i), as less knowledge may actually be externalized.

FC Formalize project communication pattern; This action has a *positive* effect on criterion (ii), as it is more likely that communication is based on artifacts.

OC Foster open project communication; This action has a *negative* effect on criterion (ii), as it is less likely that communication is based on artifacts.

Next, we extracted and analyzed only those project criteria which advocate those actions according to the survey, and selected those which (also) apply to the SRS. Moreover, we added three criteria not included in the survey but which we identified in a preparation workshop with our industrial partner, namely the length of release cycles, the volatility of requirements, and the domain and product knowledge of the specifications' audience. Tab. 3.1 and Tab. 3.2 present project criteria which are supposed to have a positive respectively negative effect on (i) and (ii), i.e., which either promote or limit the relevance of the SRS, together with a short description and rationale. For the present survey, the identified criteria serve as candidate context criteria whose effect on the extent to which the SRS is created and used shall be empirically studied.

3.1.3. Survey Design

The survey was conducted at a large, multi-national company headquartered in Germany. Although operating in different domains and offering a wide range of products, typical products are medium to large software-intensive (embedded) systems.

Participants We targeted participants directly or indirectly involved in requirements engineering for software-intensive systems, either in the sense of taking part in eliciting and specifying requirements, or in the sense of relying on knowing the requirements for their particular activities. Exemplary roles are product owners, architects or implementers.

Condition	Description & Rationale
Large teams	Team size is an indicator for the effort of team coordination, and the requirements must be communicated among many team members. Artifact-based communication provides very efficient communication, and is hence supposed to be predominately used in large teams. (ED,FC)
Distributed teams	Highly geographically distributed teams are limited in their communication means and hence more like to rely on artifacts. (ED,FC)
High team turnover	High turnover rates require to persist knowledge in artifacts. Moreover, the long-term demand of knowing the requirements, e.g., during acceptance tests, amplifies this effect. (ED,FC)
Management unavailable	Top management is required for decision making. Missing involvement is a risk, and could be mitigated with formal, artifact-based communication. Ideally, the SRS captures many intentions of the top management about the project at hand, e.g., priorities. (ED,FC)
Financial controlling	If intensive financial controlling is required, documentation is often used to persistently relate requirements to certain financial factors. (ED,FC)
Measurement required	Artifacts are much easier to measure than intangible knowledge, and are hence the preferred entity of measurement. (ED,FC)
Many stakeholders	Basically, the same argument as for large team sizes apply, especially in case of a SRS for which stakeholder involvement is typically high. (ED,FC)
Stakeholder unavailable	Same argument as management availability for external decision making. (ED,FC)
High Complexity	A higher complexity usually requires more communication, and here, the SRS is very efficient. (ED,FC)
Safety & Security	Safety & Security usually requires a comprehensive documentation, inter alia, of the system's requirements. (ED,FC)
Long release cycles	Long release cycles promote persistent forms of documenting requirements to prevent that volatile knowledge distorts or vanishes over time (ED,FC)

Table 3.1.: Criteria supposed to promote the significance of requirements specification

Survey Instrument The questionnaire consists of two main parts. Part I includes questions on the frequency of project criteria (see previous section) occur independent of individual projects, while Part II refers to the most recently completed project the participant was involved in. For that particular project, the participants were asked to characterize (a) the project itself, again with respect to the project criteria, and specify (b) the extent to which requirements were documented in a SRS and/or (c) the use of

Condition	Description & Rationale
Small teams	Team size is an indicator for the effort of team coordination. Oral communication is often considered faster and easier in small teams compared to documentation. (ED,FC)
Local teams	Local teams have additional access to verbal communication means and face-to-face meetings. (ED,FC)
Low team turnover	Low turnover rates can render the additional overhead to persist knowledge in artifacts not worthwhile. (ED,FC)
Previous cooperation	If the stakeholders and the audience (e.g., developers) previously worked together, both parties are familiar with each other and potentially the system under consideration, which in turn, may cause a less formal communication. (ED,FC)
Good cooperation	If the stakeholders and the audience work in a good and collaborative way, the need for documentation may decrease. (ED,FC)
Project Budget	A small project budget usually implies a non-formalized process and less documentation, potentially including the SRS. (ED,FC)
Volatile Requirements	Volatile requirements result in frequent changes, and the associated costs of maintaining their documentation may deter practitioners from creating and using documentation (ED,FC)

Table 3.2.: Criteria supposed to limit the significance of requirements specification

the SRS as a means to communicate requirements. Questions of part II(b) and II(c) were shown only if the participant specified he or she was involved in the elicitation and specification of requirements or required knowledge of requirements for her tasks, in that particular project. In the questionnaire, we relied mostly on closed questions, adding open questions only to capture rationales or unforeseen options, e.g., additional means of communications. The frequency (Part I) as well as the project-individual presence of project criteria (Part II-a) were captured by 4- respectively 6-point ordinal scales ranging from *Never* resp. *I strongly disagree* to *Always* resp. *I strongly agree* in order to avoid checking the middle, while the extent of documentation (II-b) as perceived by participants was captured on 5-point ordinal scales (see Fig. 3.2 for scale levels). Details on the instrument are available online[1].

Data Collection We implemented the questionnaire as an online survey using the *Enterprise Feedback Suite 10.5* tool. Due to organizational restrictions, the survey was conducted anonymously. We made the survey available to participants working in systems engineering projects via an announcement on selected working-group mailing lists of the company. In addition, we selected participants based on former, company-internal projects conducted in collaboration with our partner. However, the list of participants was undisclosed to us.

[1] `www4.in.tum.de/~mund/metrics-companion.zip`

Data Analysis For *RQ 1-1*, we considered only those participants who stated to be involved in requirements elicitation or specification. For both, the completeness and the level of detail, we extracted the number of projects for each level of the ordinal scales. For *RQ 1-2*, we compared the number of projects in which an SRS (paper-based or tool-based) was used to those that relied on meetings/workshops, personal talks, and groupware solutions. Furthermore, for those respondents who stated to use an SRS, we evaluated the participants' ranking of the communication means according to their perceived individual relevance for informing about requirements.

In addition, for both research questions, we investigated the relationship between the project criteria introduced in Sec. 3.1.2 and the creation/usage of the SRS. Specifically, we studied the correlation between the participants' perceived general (i.e., multi-project) prevalence of those criteria (Part I) and the presence for the individual project (Part II) on the one hand, and the impact on the creation (*RQ 1-1*) and usage (*RQ 1-2*) of the SRS on the other hand (see Fig. 3.2a and 3.2b for scale levels). To this end, we relied on rank-based correlation coefficients and hypothesis test based on Kendall's τ. The null hypothesis is that the prevalence or presence of a project criterion and the creation/usage of the SRS are not correlated. We followed the common interpretation of $\tau \geq 0.3$ as moderate and $\tau \geq 0.5$ as strong correlation and used a significance level of $\alpha = .05$. Because we tested multiple hypothesis, we also calculated an adjusted p-value p_{fdr} based on Benjamini and Hochberg [1995] to mitigate the threat that correlations were only found by chance. In addition, we extracted reasons for complete, incomplete, shallow and detailed specifications from the open questions to complement the quantitative data.

Validity Procedures The survey instrument was reviewed by two additional researchers and two industrial partners who ensured no terms with unintended or ambiguous meanings within the company were used. To avoid both bias towards single projects and multiple answers, we verified all projects gathered in the survey are unique by comparing the projects' names (which was mandatory to provide by participants). Qualitative data resulting from open questions was reviewed independently to avoid misinterpretation. For identified correlations (*RQ 1-1* and *1-2*), results were visualized using bubble plots and checked for plausibility.

3.1.4. Results and Interpretation

In the following, we summarize our results structured according to the research questions. For each question, we conclude with a brief interpretation of the results.

Study Population The survey was accessed 85 times, of which 46 participants (54%) completed the survey[2]. Four participants did specify to neither being involved during requirements specification nor that knowledge on requirements was required for their tasks. The remaining 42 participants had an average experience of ≥10 years and completed

[2] 61 participants (72%) partly completed the survey, mostly until the mandatory disclosure of the project name.

Role	Specification	Required Knowledge (only[1])	Total
Product Manager	9	4 (-)	9
Project Lead	5	4 (-)	5
Req. Engineer	9	5 (1)	10
Architect	5	8 (3)	8
Implementer	-	4 (4)	4
Tester	1	1 (-)	1
Quality Manager	1	2 (1)	2
Other	1	3 (2)	3
All	31	31 (11)	42

[1] By the term *only* we refer to participants who stated that they required knowledge of the requirements specification but were not involved in requirements elicitation or specification.

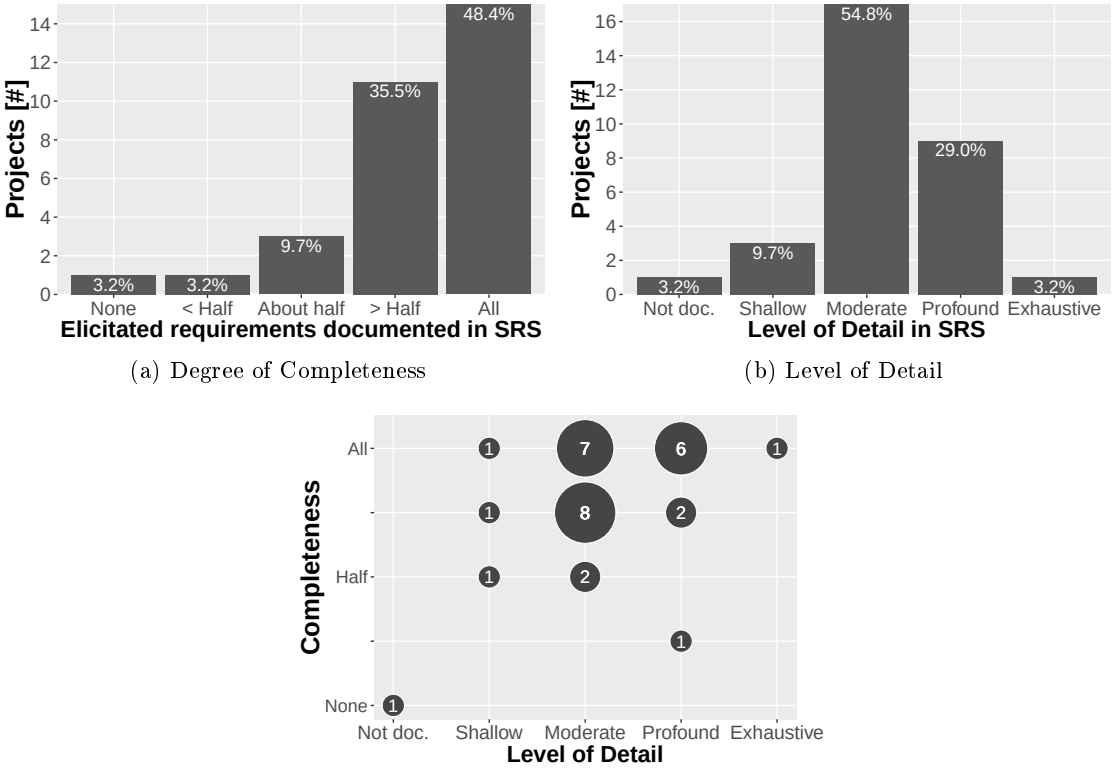
Table 3.3.: Participants by project role and RE involvement (during Specification, or in the sense of required knowledge of requirements)

$6 - 10$ projects in total on average. The participants' roles are illustrated in Tab. 3.3 distinguished by whether they were involved in elicitation and specification or required knowledge of the requirements for their project activities. Projects specified by the participants were balanced between market-driven products and customer-specific solutions (23 to 19) and between new and further development (22 to 20). The majority of projects (71%) had release cycles between six month and two years, but short ($\leq$6 months) and long ($\geq$5 years) also occurred.

RQ 1-1: Documenting requirements in SRS

In general, our results indicate a rather comprehensive documentation of requirements. Considering the quantity, the respondents stated for 15 out of 31 projects that all identified requirements were actually documented, and in only four cases (13%) about half or less than half of the identified requirements were documented (Fig. 3.2a). The level of detail of the documented requirements was mostly considered as moderate (17 cases, 55%) or profound (9 cases, 29%). In turn, it was considered shallow (3 cases, 10%) or exhaustively detailed (1 case, 3%) only rarely (Fig. 3.2b). Only for one project requirements were not documented at all, according to the participants. Furthermore, we found a moderate correlation (Kendall-$\tau = 0.33$, $p = 0.04$) between the degree of completeness and the level of detail in SRS, depicted as a bubble chart in Fig. 3.2c.

The relationship between project criteria and the documentation, i.e., the level of completeness and level of detail, is described by the rank-correlation coefficients and associated p-values in the left part of Tab. 3.4 and Tab. 3.5. The data shows positive correlations between the completeness and three criteria, namely that



(a) Degree of Completeness

(b) Level of Detail

(c) Correlation between (a) and (b)

Figure 3.2.: Survey results on requirements documentation (RQ1): portion of documented requirements (a), prevalent level of detail (b) and its correlation (c)

1. safety/security concerns are relevant for the system under consideration
2. measurements are required (both for presence in individual projects and general prevalence)
3. the team and stakeholders work in a good and collaborative way (individual projects only)

All but the last criterion also positively correlate with the level of detail in the SRS, but the project-specific correlation regarding measurements was not statistically significant. In contrast, a high complexity of the system under consideration (project-specific presence and general/multi-project prevalence) and volatile requirements (general only) correlate negatively with the completeness of the SRS. The circumstance that the stakeholders and the team worked together in previous cooperation negatively correlates with the level of details in an SRS.

Qualitative feedback supports the identified correlations. Several participants stated that the degree of completeness in the SRS was required by the development process,

with multiple participants stating that it is a consequence of the domain ("regulatory requirements (healthcare sector) enforce documentation") and/or required for ensuring traceability (which is perceived as "mandatory for healthcare products" and "required for distributed teams".

In agile processes, SRS documents were perceived as complete due to "only documented requirements entering the backlog". Moreover, participants stated that complete requirement specifications were obtained due to the "involvement of all (relevant) stakeholders" and "iterations between development and product management". In contrast, bare "[existence of] too many requirements (thousands)" or the application of "traditional RE approaches on complex projects" resulted in incomplete requirements specification according to participants. Long release cycles were also perceived as a reason either directly, since "topics [...] which are relevant late were documented very coarsely" and "long project durations [imply] many changes", or more indirectly, because of "permanent changing goals, constraints, stakeholders and project teams". Also, external documentation (e.g., "availability of legacy systems"), limited time ("restrictive time-boxing" and "not enough time [...] to elicit all requirements"), and "stakeholder lacked knowledge of requirements in detail" were mentioned.

For the level of detail, participants mentioned that "good coordination of requirements in the team allowed for less detail in documentation" and a "rough direction [is] enough [because] experts clarify details during implementation". Most predominantly, participants stated solution-orientation as a reason for a shallow level of detail including statements like "results more important than documentation", "urgency for technical results limits time for specifications" and "not enough time and resources for detailed analysis". "Development process constraints" were mentioned as an exemplary reason for detailed SRS.

Interpretation In general, the survey confirms that requirements are not exhaustively documented in an SRS for every project. Our results suggest that requirements are, however, at least to some extent documented in nearly every project, usually covering most if not all identified requirements with at least a moderate level of detail. Comparing those two dimensions, the results reveal that requirements specifications in general are perceived as more exhaustive regarding the proportion of requirements which are documented at all than regarding its level of detail, as seen by the prevalence of projects above an imaginary diagonal in Fig.3.2c, suggesting that the former is more important and/or easier to achieve for practitioners.

If we take context parameters into account, correlations obtained for the particular project are generally weaker than their general prevalence across the projects the respondents worked in. This may indicate that the extent to which requirements are documented is rather influenced by the chosen development process, which in case of our industrial partner reflects the demands and characteristics of specific domains, than of individual projects. For the positive correlations revealed, we suppose that safety/security concerns and demand for measurement cause an increased demand for documentation, while a good cooperation between stakeholders and the development team (audience) al-

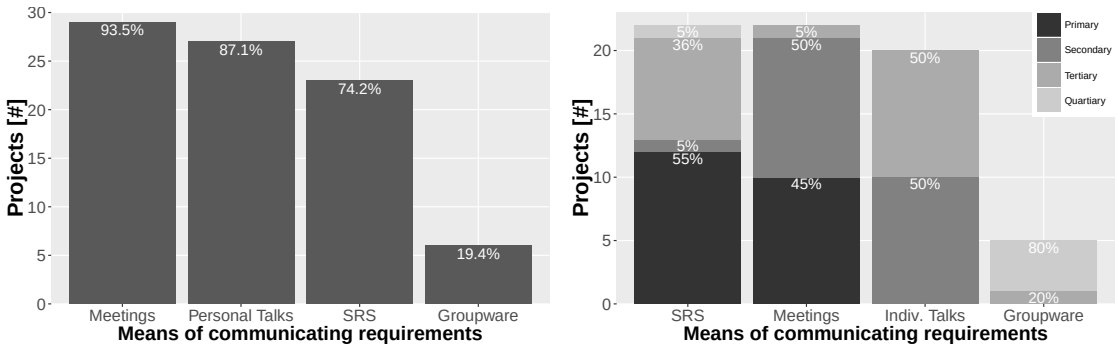
lows more exhaustive documentation. For negative correlations, we assume that volatile requirements hamper the degree of completeness of SRS. However, we are undecided if a high complexity of the system under consideration decreases the need for documentation, because of an inherent inefficiency associated with the difficulties of documenting its requirements, or whether it simply impedes the documentation of requirements ("uselessness of traditional RE approaches") without actually diminishing its need.

RQ 1-2: SRS as Communication Means

The SRS, independent of its physical representation (paper-based or tool-based), was used as a means to communicate requirements in 23 out of 31 projects (74%), ranked third behind meetings/workshops (29 projects, 94%) and personal talks (27 projects, 87%, see Fig. 3.3a). Considering only participants not involved during specification, the SRS is used slightly more often (82%, +8%), while meetings/workshops (91%, −3%) and personal talks (82%, −5%) are used marginally less often. In any case, groupware solutions are used rarely (20% and 27%, respectively.

To further investigate the importance of the SRS as a communication means, we asked the participants to rank the specified means by how informative they were perceived for their individual project tasks (see Fig. 3.3b). Our results reveal that in case an SRS is used, it is the primary source for communicating requirements (55%), but only slightly more often than meetings/workshops (45%). In fact, we observed a pronounced polarization between artifact-based and direct face-to-face communication. If meetings were the primary source, individual personal talks were specified predominantly as the secondary source of information, with the SRS being used in only one case as the secondary means. Out of five projects using groupware solutions, it was considered the least in all but one case, where it was ranked second to last, attesting groupware solutions an inferior relevance for communication. Considering only participants not involved during the requirements specification, SRS-based communication of requirements was considered as primary source significantly more often (75%, +20%), indicating its superiority for communicating requirements for development phases subsequent to requirements engineering.

Compared to the documentation of requirements (*RQ1-1*), we observed a weaker effect of project characteristics on the communication of requirements. First and foremost, we could not reject any null hypothesis for the general usage of the SRS, and identified only two statistically significant correlations for the ranking of communication means: a strong correlation ($\tau = 0.52$, $p = 0.01$) with the relevancy of safety/security for the project goal, and a moderate correlation with the length of release cycles ($\tau = 0.33$, $p = 0.08$). However, our results show several weak correlations (about $\tau = 0.2$) which are in tune with the consequences for documentation identified by Kalus and Kuhrmann [2013] (see Sec. 3.1.2). For instance, we found weak correlations for team parameters regarding size and turnover, as well as project characteristics such as the demand for measurements ($\tau = 0.27$ for ranking) or high complexity ($\tau = -0.18$ for usage and $\tau = -0.12$ for ranking). In addition, we investigated whether the background knowledge regarding the domain or product as stated by the participants impacts the communication. However,



(a) Overall Usage

(b) Ranking (if SRS is used)

Figure 3.3.: Usage of SRS: means for communicating requirements ranked according to the participants' perceived importance for information

we could not reject the null hypothesis, suggesting that background knowledge may not have that strong of an impact on the communication means used[3].

[3]Note that due to scoping, we did not investigate whether there is *less* communication, independent of the actual *means* to transfer knowledge

Parameter		Impact on Documentation in SRS: Completeness			Impact on Documentation in SRS: Level of Detail			Impact on Communication using SRS: SRS used?			Impact on Communication using SRS: SRS Ranking		
		τ	p	p_{fdr}	τ	p	p_{fdr}	τ	p	p_{fdr}	τ	p	p_{fdr}
Team Size	Proj.	-0.08	0.63	0.85	0.06	0.72	0.95	0.07	0.65	0.8	0.2	0.28	0.63
	Gen.	-0.08	0.6	0.82	0.17	0.28	0.52	0.04	0.81	1	0.05	0.79	0.89
Team Distribution	Proj.	0.06	0.7	0.85	0.02	0.89	0.95	0.09	0.59	0.8	0.02	0.92	0.92
	Gen.	-0.04	0.81	0.84	0.25	0.12	0.32	-0.14	0.41	1	0.09	0.62	0.85
Team Turnover	Proj.	-0.02	0.88	0.93	0.06	0.71	0.95	-0.01	0.94	0.94	0.22	0.26	0.63
	Gen.	-0.09	0.56	0.82	0.01	0.94	0.95	-0.04	0.83	1	0.19	0.32	0.75
Management unavailable	Proj.	0.01	0.93	0.93	0.06	0.69	0.95	0.1	0.54	0.8	0.13	0.52	0.73
	Gen.	0.03	0.84	0.84	0.28	0.08	0.29	0.08	0.61	1	0.26	0.19	0.75
Financial controlling req.	Proj.	0.09	0.59	0.85	0.12	0.44	0.95	0.15	0.37	0.77	0.12	0.53	0.73
	Gen.	-0.05	0.73	0.84	0.09	0.55	0.75	0.3	0.07	0.71	-0.18	0.34	0.75
Measurement req.	Proj.	0.34	0.03	0.18	0.27	0.1	0.53	0.14	0.4	0.77	0.27	0.17	0.62
	Gen.	0.37	0.02	0.08	0.37	0.02	0.16	0.25	0.13	0.71	-0.21	0.29	0.75
Many stakeholders	Proj.	-0.1	0.55	0.85	0.06	0.71	0.95	-0.05	0.77	0.85	0.04	0.84	0.92
	Gen.	-0.11	0.51	0.82	0.16	0.33	0.52	0.14	0.42	1	-0.15	0.46	0.8
Stakeholder unavailable	Proj.	-0.12	0.46	0.85	0.03	0.87	0.95	0.21	0.22	0.77	0.08	0.69	0.84
	Gen.	-0.17	0.29	0.64	0.17	0.28	0.52	0	1	1	-0.03	0.89	0.89
High complexity	Proj.	-0.31	0.06	0.21	0.01	0.95	0.95	-0.18	0.31	0.77	-0.12	0.53	0.73
	Gen.	-0.36	0.02	0.08	0.02	0.92	0.95	0.03	0.86	1	0.03	0.89	0.89
Safety/Security relevant	Proj.	0.38	0.02	0.18	0.36	0.02	0.25	0.14	0.42	0.77	0.52	0.01	0.08
	Gen.	0.32	0.04	0.12	0.34	0.03	0.16	0.01	0.96	1	0.38	0.05	0.51
Release Cycle Length	Proj.	-0.19	0.24	0.67	-0.01	0.95	0.95	0.26	0.11	0.77	0.33	0.08	0.46

Table 3.4.: Impact of project criteria (*Proj.*: project-specific; *Gen.*: multi-project prevalence) on the significance of the SRS (moderate corr. for $\tau \geq .3$, sig. level $\alpha = .05$)

Parameter		Impact on Documentation in SRS						Impact on Communication using SRS					
		Completeness			Level of Detail			SRS used?			SRS Ranking		
		τ	p	p_{fdr}	τ	p	p_{fdr}	τ	p	p_{fdr}	τ	p	p_{fdr}
Previous cooperation	Proj.	-0.03	0.83	0.83	-0.33	0.04	0.16	-0.11	0.52	0.81	-0.07	0.71	0.81
	Gen.	-0.01	0.95	0.95	-0.19	0.22	0.45	0.03	0.87	0.96	-0.17	0.38	0.81
Good cooperation	Proj.	0.44	0.01	0.04	0.19	0.25	0.34	0.07	0.71	0.81	-0.05	0.81	0.81
	Gen.	0.26	0.11	0.23	-0.15	0.36	0.48	-0.07	0.69	0.96	-0.29	0.15	0.81
Small Budget	Proj.	0.12	0.45	0.6	-0.19	0.23	0.34	0.04	0.81	0.81	-0.14	0.48	0.81
	Gen.	0.01	0.95	0.95	-0.27	0.08	0.32	0.18	0.28	0.96	0.02	0.92	0.81
Volatile Requirements	Proj.	-0.21	0.19	0.38	-0.04	0.81	0.81	-0.08	0.62	0.81	-0.1	0.59	0.81
	Gen.	-0.47	0	0.01	-0.01	0.95	0.95	-0.01	0.96	0.96	-0.13	0.51	0.81
Prev. Domain Knowledge	Proj.	-	-	-	-	-	-	0.28	0.09	0.18	0	1	1
Prev. Product Knowledge	Proj.	-	-	-	-	-	-	-0.03	0.85	0.85	-0.08	0.68	1

Table 3.5.: Impact of project criteria (*Proj.*: project-specific; *Gen.*: multi-project prevalence) on the significance of the SRS (moderate corr. for $\tau \geq .3$, sig. level $\alpha = .05$) (continued)

Interpretation Overall, the empirical evidence supports that requirements specification are a well-established means to communicate requirements in practice. On closer inspection, the data reveals that participants involved during the specification of requirements exhibit quite different communication preferences than participants who only required knowledge of the requirements for their individual project assignments. In particular, the prevalence and importance of the requirements specification for the communication of RE results to subsequent development activities, e.g. testing, is more pronounced compared to communication within RE activities, e.g. elicitation and negotiation, for which face-to-face communication seems prioritized. Therefore, we argue that in projects with a high degree of division of labor in terms of project activities, non-artifact-based means prevail for communication within RE and adjacent activities (e.g., high-level architecture, cf. Tab. 3.3), while for communicating requirements to subsequent development activities SRS seems to be used predominately.

In contrast to the documentation of requirements, the use of SRS as a communication means may be less determined by the development process but rather specific to the project or even individual preferences. The latter could also explain why only weak correlations were revealed, e.g., team size ($\tau = 0.20$) and distribution ($\tau = 0.22$). To explain the identified correlations, we suppose the length of release cycles impacts the use of SRS for communication (e.g., due to the persistent nature of artifacts), and that safety/security concerns demands documented traceability. Also, we propose that the extent to which requirements are documented affects the relevance of the SRS for subsequent communication as an explanation for the correlation between the extent of documentation (*RQ1-1*) and its use for communication ($\tau = 0.36$ for completeness, and $\tau = 0.46$ for level of detail).

3.1.5. Limitations

Considering the internal validity, we had to cope with limited control regarding sampling and delivery because of the particular industrial setting. Hence, we were unable to establish a random sampling. Therefore, statistical results have to be considered with a salt of grain and participation bias is likely. While we used the survey results to gain first insights into when and how an SRS is created and used, the number of participants was too low to apply statistical methods reliably when discriminating between aspects, e.g. for the subgroup of participants not involved in the specification of requirements.

Also, we cannot exclude with statistical significance that the observed correlations occurred only by chance, since most of the adjusted confidence intervals p_{fdr} were above the threshold. Finally, despite our validity procedures, we still suspect some terms to be subject to misinterpretation, partly because of the heterogeneity of requirements engineering in practice.

3.2. Significance of the Artifact's Quality

The survey presented in Sect. 3.1 concludes that, provided certain project criteria hold, the requirements specification is used to foster communication to downstream development. In this section, we investigate the second research objective of this chapter, namely the (complementary) question to what extent the quality of the requirements specification, if used, actually impacts downstream development, by means of an experiment. Since a complete analysis of all such activities is infeasible in a controlled setting, our study focuses on one activity that strongly depends on the contents of requirements specifications, namely system testing. We further focus on two specific quality defects, one concerning semantic and one pragmatic quality, of a requirements specification (in the terminology of Sec. 2.2).

3.2.1. Research Questions

In this experiment, we study the following three research questions:

RQ 2-1: Do incorrect SRS statements impact system testing? According to Sec. 2.2.1, semantic quality defects can be characterized as incomplete and/or incorrect information in the SRS with respect to the stakeholders' actual demands on the system. In *RQ 2-1*, we specifically investigate whether incorrect information in the SRS inevitably leads to flawed system test cases or makes their inference less efficient, more precisely, leads to an increase in difficulty as perceived by the participants.

RQ 2-2: Do negated SRS statements impact system testing? In contrast, *RQ 2-2* focuses on pragmatic quality, i.e. the consistent and complete comprehension of the SRS by the target audience, in this case, test engineers. Defects regarding pragmatic quality (see, e.g., ISO/IEC 29148:2011 [2011]) describe valid information but can nonetheless lead to flawed test cases if the SRS is misunderstood or not understood at all. In a recent study, Femmer et al. [2014c] revealed that practitioners were unsure about the validity of *negative statements*, one potential quality attribute for natural-language specifications, yet without empirical foundation. Therefore, we investigate the impact of this particular defect on system testing in terms of incomplete and/or incorrect test cases and its perceived difficulty.

RQ 2-3: Does a-priori knowledge compensate for quality defects in SRS? Among the main confounding factors that prevent generalization of results from controlled settings is the participants' context knowledge. Hence, we want to know if and to what extent a-priori knowledge about the application domain, also called the problem space [Jackson, 2001], effectively compensates the quality defects studied in *RQ 2-1* and *RQ 2-2*.

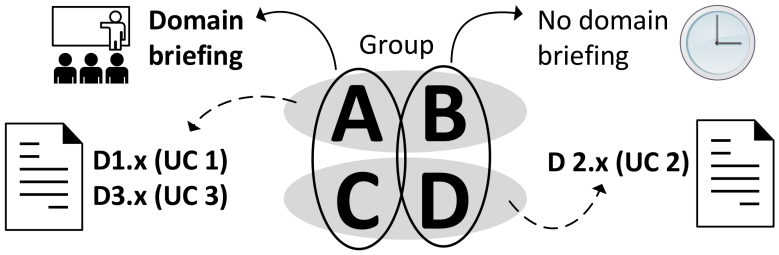


Figure 3.4.: Overview of the experiment design

UC	ID	Type	Correct statement	Flawed statement
UC1	D1.1	Incorrect	*In case the data is complete and valid, the data will be imported.*	*In case the data is complete and valid,* ***an error message occurs.***
	D1.2	Incorrect	*Data records must then be further approved by a case handler and explicitly activated.*	*Data records must then be checked by a* ***plausibility algorithm and are activated afterwards.***
UC2	D2.1	Negation	*The user must enter at least one character.*	*The user* ***is not allowed to enter zero characters.***
	D2.2	Negation	*The system must treat lowercase and uppercase letters the same.*	*The system* ***must not distinguish*** *between lowercase and uppercase letters.*
	D2.3	Negation	*The user may only select one company at a time.*	*The user* ***cannot select more than one company*** *at a time.*
UC3	D3.1	Negation	*The user may nominate up to three substitutes.*	***No user must nominate more than*** *three substitutes.*
	D3.2	Negation	*If the user selects herself as a substitute, an error message is shown.*	*A user* ***cannot select herself*** *as a substitute.*

Table 3.6.: Experiment treatment: Presence of defects (y/n) injected into use cases of SRS

3.2.2. Experiment Design

We injected pre-defined defects into real-world use cases and asked experiment participants to specify system test cases that appropriately verify the stakeholders' requirements. To this end, we provided tabular templates to be filled out within a 45 minutes time slot. No questions were allowed during the experiment. Additionally, we asked the participants about how difficult they perceived the inference of test cases for each use case, both quantitatively (8-point Lickert-scale) and qualitatively using open questions.

Metric	Description
$Total_{\texttt{Grp,Def}}$	Number of overall assessed test cases regarding defect(s) `Def`, limited to participants from `Grp`
$Correct_{\texttt{Grp,Def}}$	Number of correct test cases regarding defect(s) `Def`, limited to participants from `Grp`
$Detected_{\texttt{Grp,Def}}$ (D 1.x only)	Number of test cases which detected `Def`, limited to participants from `Grp`
$Omitted_{\texttt{Grp,Def}}$ (D 2.x & 3.x only)	Number of test cases which omitted to test for the requirement specified by `Def`, limited to participants from `Grp`
$CorrRate_{\texttt{Grp,Def}}$	$Correct_{\texttt{Grp,Def}}$ / $Total_{\texttt{Grp,Def}}$
$DetRate_{\texttt{Grp,Def}}$	$Detected_{\texttt{Grp,Def}}$ / $Total_{\texttt{Grp,Def}}$
$OmitRate_{\texttt{Grp,Def}}$	$Omitted_{\texttt{Grp,Def}}$ / $Total_{\texttt{Grp,Def}}$
$Independence_{\texttt{Def}}$	Independence between the presence of defect `Def` in the use case and the correctness respectively omission in the inferred test cases, expressed as the p-value of Pearson's χ^2 test (alt. hypotheses: $\mathbf{H}_{A,C}$ resp. $\mathbf{H}_{A,O}$)

Table 3.7.: Metrics for evaluating the quality of the obtained system tests

At the start of the experiment, we randomly assigned participants to one of four Groups A-D (see Fig.3.4), and evaluated the inserted defects (see Tbl. 3.6) as follows:

For *RQ 2-1*, we injected two defects into UC1: An obviously incorrect defect D1.1, which requires to show an error message in case of success, and a more subtle defect D1.2, which suggests that a certain, strictly required manual check is instead executed automatically by an algorithm. Groups A & B were faced with this flawed use case, whereas Groups C & D served as control group.

For *RQ 2-2*, we converted five positively stated requirements in UC2 and UC3 into their negative versions (D2.x and D3.x), carefully preserving the meaning of the each statement. Here, Groups A & B received the negated version of UC3 and serve as control group for UC2, and Groups C & D received the negated version of UC2 and served as control group for UC3.

For *RQ 2-3*, we provided the participants of Groups A & C with certain knowledge about the domain before assigning the task to them. This briefing included the purpose of the overall system, the relevant business processes, important rationales and necessary constraints from the perspective of a long-term employee of the company. In particular the briefing included, among other facts, the intended behavior for the defects D1.1 and D1.2. At the end of the briefing, questions were allowed to further foster the participants' understanding. For this RQ, Groups B & D served as control group. During the a-priori knowledge briefing, the participants of these groups were lead to a different room and involved in a discussion on a separate topic, unrelated to the tasks of the experiment.

Participants We conducted the experiment as part of our RE lecture at the Technical

University Munich (TUM). Participants were mostly advanced undergraduate or early graduate students of computer science or information systems, and the experiment was conducted late in the term so that students had a fundamental understanding of the contents and applications of SRS.

Study Objects In order to keep the setting close to reality, we reuse a real-world SRS from an industrial partner. The original requirements specification was 21 pages long and written in natural language. It contained an overview, problem statement, supported business process description, functional requirements (organized as use cases) and non-functional requirements. For the experiment, we selected three out of 18 use cases, together with the original overview description and problem statement. All company-specific terms and acronyms were either removed or renamed due to legal reasons.

Data Collection and Analysis We evaluated the obtained test cases for each defect by manual inspection: We assigned `correct` to a test case if and only if the test explicitly covered the stakeholder's intended requirements (i.e., the correct versions in Tab. 3.6), `flawed` if unintended requirements were tested, and `omit` if the test did not cover the requirement at all. Specifically for incorrect statements (RQ 2-1), we also inspected whether the participants actually `detected` the defects, i.e., were aware of them. We evaluated detection based on whether we encountered remarks on D1.1 or D1.2 in the test cases themselves, the associated use cases or as answer to the open questions. The metrics used for analysis are listed in Tab. 3.7. We applied statistical tests regarding the following alternative hypotheses:

$\mathbf{H}_{A,C}$ The presence of a defect in the use case and the correctness of the test cases (regarding this defect) are not independent.

$\mathbf{H}_{A,O}$ The presence of a defect in the use case and the omission of the corresponding requirement in the test cases are not independent.

$\mathbf{H}_{A,D}$ The perceived difficulty is different for use cases with defects present.

For the impact on the test quality, we tested $\mathbf{H}_{A,C}$ and $\mathbf{H}_{A,O}$ (D 2.x and 3.x only) using Pearson's χ^2 test for each defect. To evaluate the impact on efficiency, we applied the Mann-Whitney test to $\mathbf{H}_{A,D}$. In both cases we demanded a significance level of α=0.05. Furthermore, to validate the direction of the impact confirms to our assumptions, i.e. whether it is indeed negative for quality defects, we compared *CorrectRate* and *OmitRate* for correct and flawed use cases. Regarding the impact of a-priori knowledge (RQ 2-3), we compared the aforementioned metrics based on whether they received the domain knowledge briefing before the experiment.

Validity Procedures To ensure the reliability of the manual inspection, a second researcher independently rated 33 (25%) randomly selected test cases. The obtained inter-rater reliability measures attested a very high level of agreement: we agreed on

Defect		Independence		CorrRate		DetRate	OmitRate	
ID	Grp	Omit	Corr.	Correct	Flawed	Flawed	Correct	Flawed
	All	-	0.01	1.00	0.47	0.80	-	-
D 1.1	A&C	-	0.18	1.00	0.50	0.83	-	-
	B&D	-	0.07	1.00	0.44	0.78	-	-
	All	-	0.00	1.00	0.00	0.00	-	-
D 1.2	A&C	-	0.01	1.00	0.00	0.00	-	-
	B&D	-	0.00	1.00	0.00	0.00	-	-
D 2.1	All	0.54	0.88	0.83	1.00	-	0.68	0.53
D 2.2	All	0.20	-	1.00	1.00	-	0.26	0.53
D 2.3	All	0.28	0.71	1.00	0.80	-	0.47	0.71
D 3.1	All	0.61	0.34	0.56	0.80	-	0.25	0.42
D 3.2	All	0.22	-	1.00	1.00	-	0.08	0.35
D 2.x	All	0.15	0.91	0.90	0.93	-	0.38	0.51
&	A&C	0.19	0.83	0.84	0.90	-	0.34	0.51
D 3.x	B&D	0.59	1.00	0.96	0.95	-	0.42	0.50

Table 3.8.: Experiment results (sig. level $\alpha = .05$, 41 participants)

93% of all cases, with an inter-rater agreement (Cohen) of $\kappa = 0.90$ attesting almost perfect agreement. Furthermore, to avoid unintentional influence of groups A and C beyond explaining the business process and employee experiences, the briefing was voice-recorded and checked later on, e.g., for accidental hints about the test cases or omissions of facts.

3.2.3. Results and Interpretation

Overall, 41 students participated in the experiment, with about ten students in every group. Tab. 3.7 presents the obtained metrics and Fig. 3.5 illustrates the results of the self-evaluation of the participants.

RQ 2-1: Impact of Incorrect Statements

The correctness of the test cases obtained from flawed specifications differed substantially between the defects D1.1 and D1.2: while for D 1.2 no participant detected the defect and hence no correct test case was inferred, about half of all participants (47%) explicitly corrected the defect D 1.1 (although 80% of the participants actually recognized it).

Hence, in 20% of the cases, the defect was not detected at all. In contrast, for both D1.1 and D1.2, in the control group, a correct specification also led to correct test cases. Therefore, we were able to reject the null hypothesis in favor of $\mathbf{H}_{A,C}$ with statistical significance ($p \leq \alpha = 0.05$ for both, D 1.1 and D 1.2). Furthermore, participants perceived the task as more difficult for UC 1 with defects D 1.1 and 1.2 present, hence rejecting the null hypothesis in favor of $\mathbf{H}_{A,D}$ (p=0.02).

Interpretation The evidence suggests that the presence of incorrect statements in the SRS does indeed impact system testing. This appears to be true regarding the quality of the obtained test cases, for obvious defects (D 1.1) and even more for less obvious ones (D 1.2). In addition, the (perceived) difficulty indicates an increase in required efforts, and, hence, also impacts efficiency of testing.

RQ 2-2: Impact of Negated Statements

In contrast to incorrect use cases, the use of negative statements did not impact correctness of test cases substantially. For any negative statement but D 2.3, correct test cases were obtained at least as often as for the non-negative statement. Consequently, we were unable to reject the null hypothesis in favor of $\mathbf{H}_{A,C}$ for any negated statement. However, test cases did omit the specified requirement considerably more often (+13%) but not with statistical significance ($\mathbf{H}_{A,O}$, p=0.15). Participants perceived the task equally difficult ($\mathbf{H}_{A,E}$, p=0.46 (UC 2) and p=0.95 (UC 3)). Although not directly related to negated statements, participants expressed their dissatisfaction with the pragmatic quality of the specification as qualitative feedback, e.g., complaining about "wording of requirements", "lack of chronological structure" and the "use of passive sentences".

Interpretation Our results suggest that negated statements do not (significantly) impact testing in the sense that faults in terms of incorrect information are introduced into test cases, nor does it make the process of inferring these test cases more difficult. However, we observed a noticeable but statistically insignificant increase in omission of requirements containing negative statements, which might potentially lead to lesser quality in test cases due to untested requirements.

RQ 2-3: Relevance of A-priori Knowledge

For both incorrect as well as negative statements, results of participants introduced to the underlying business process (groups A & C) did not vary considerably compared to the control groups B & D. Notably, participants were unable to detect or correct D 1.2 (mandatory manual checks by a case handler) despite being informed about its necessity.

Interpretation We were surprised that, although we explicitly mentioned the correct behavior in D 1.2 in the briefing, participants were unable to compensate the defect. Within the experiment, the participants' knowledge of the system under consideration did neither compensate for incorrect requirements in specifications nor affected the quality of

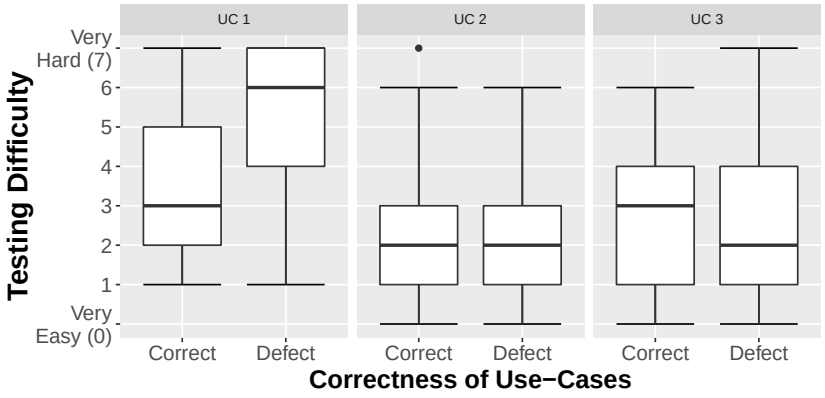


Figure 3.5.: Difficulty of deriving test cases per use case and correctness as perceived by participants

test cases inferred from specifications extensively using negative statements. Therefore, we conclude that defects can propagate through the engineering process, even when people are briefed about the correct requirements. However, due to relying on students only, we highly advocate further research in this direction.

3.2.4. Threats to Validity

We see three main threats as limitations of the experiment: First, we relied on RE students as participants. Therefore, the obtained test cases were of rather poor quality in general, and certainly not at the level of experts in the field of system testing. Second, a briefing cannot lead to the same depth of domain knowledge compared to own experiences and observations over a prolonged amount of time, which we expect to be superior, e.g., in terms of recognition and trust. We thereby need to extend the experiment with industry experts in the future. Finally, we need to emphasize we only investigated selected defects and thus have to be careful to generalize results to classes of SRS quality (see Sec. 2.2.1), especially concerning pragmatic quality factors.

3.3. Discussion

In this section, we discuss and critically reflect on the insights gained from both studies according to three distinctive questions.

Under which circumstances does SRS quality matter? As shown by the survey, the extent a SRS is created and used varies in practice. To understand the circumstances upon which this extent depends, the survey studied several characteristics which are supposed to promote or limit the relevance of the SRS, and identified moderate correlations with, e.g., the relevance of safety/security, or the length of release cycles. For several

others, including team characteristics such as size and turnover, only weak correlations were identified.

We see two explanations for this: First, we may have selected the wrong characteristics. This may be due to studying characteristics of the project which are not all that relevant while neglecting those that are. Also, we might have been erroneously focusing on characteristics closely linked to the project environment, while it rather depends on characteristics concerning either the organization such as the established communication culture, or individuals such as personal preferences and work habits. Since the characteristics were extensively based on a comprehensive literature review on software process tailoring, this would in turn imply that either there exists a substantial discrepancy about effective tailoring between practitioners and the scientific software process community, or documentation and communication within requirements engineering and to downstream development should follow different rules than the rest of the software process.

The second and, arguably, more likely explanation is that single characteristics just do not absolutely determine the extent the SRS is created and used on their own. Instead, it is determined by a complex and yet not well understood combination of several characteristics which causes the variation. In practice, the actual extent of artifact-orientation is the result of human decision-making which must balance the needs of documentation, the imposed limitations, and the associated costs, all for a specific situation. One particular relationship that struck our mind during evaluation was that certain circumstances *dominate* others, by creating such a strong need for documentation or imposing so high obstacles that other factors are neglected. Our data set already indicates, for example, that safety/security-relevant systems may be such a project characteristic in the sense that its presence dominates secondary circumstances such as team sizes. Indeed, qualitative feedback suggests that, due to legal regulations, a rigorous documentation is enforced no matter of secondary circumstances, and we observed significantly stronger correlations for several other factors when considering non-safety-critical projects only. However, the limited number of projects prohibited reliable conclusions, and hence we must refer to future work.

Last but not least, we expected domain knowledge to be a very strong factor limiting the creation and use of the specification. While recognizing several limitations of our studies, in particular relying on students and the limited comparability of a short briefing to long-term domain experience, we were still surprised results were insensitive to the participants' knowledge, both in the experiment and even more so regarding the survey. To this end, we encourage to independently study the hypothesis that domain knowledge does not significantly impact the extent SRS quality matters in practical settings, for instance, using case-study research.

Which (types of) attributes of SRS matter? In general, one may argue that the semantic and pragmatic quality of requirements specifications, as described in Sec. 2.2, become more important if requirements are documented to a larger extent, respectively used more extensively for communication. Therefore, the revealed correlations between project criteria and SRS-based documentation/communication (see Sec. 3.1) are indi-

cators for the relative importance for the semantic and the pragmatic quality as well. Moreover, our experiment yields first quantitative insights into the impact of SRS quality: The defect D 1.1, i.e., an incorrect statement we expected to be easily recognizable, leads to flawed test cases for about every second participant, and no correct test case could be derived from the less obvious defect D 1.2. Although not part of the experiment, we do not expect better results for semantic defects in terms of missing information in the SRS. Therefore, we suggest to tentatively generalize our findings by proposing that semantic quality of the SRS is generally essential for subsequent engineering activities.

However, the impact of the pragmatic quality appears to be more diffuse. While our results suggest that negated statements do not impact engineering activities, we refrain from generalizing our results to pragmatic quality in general. We may even assume that negated statements are rather easily correctable compared to other pragmatic quality issues, e.g., the use of passive voice. In fact, qualitative feedback suggested that pragmatic quality was at least perceived as an obstacle, and our participants omitted requirements expressed in negated statements considerably more often. We believe that the pragmatic quality is rich on facets we do not yet properly understand with some attributes having more severe impacts than others. Furthermore, existing empirical evidence (see Sec. 3.4) suggests that the questions whether or not a specific attribute impacts downstream development depends on the context of use. Consequently, we strongly postulate to question and rigorously investigate attributes that best practice norms on (pragmatic) quality propagate, and dependent on the context the requirements specification is used in.

How can we assure the quality of an SRS? Since quality assurance always comes at a certain cost, an immediate conclusion of our result is that SRS-based approaches applied independent of contextual circumstances are inherently inefficient. Based on the results and discussion, we advocate that quality assurance should not be applied in general, but only where actually required. This can be achieved by different means, e.g., by introducing tailoring mechanisms or by using context-specific inductive approaches. However, independent of the actual mean, an efficient SRS-based quality assurance must include a decision procedure which specifies when the quality of an SRS needs *not* to be assured. To this end, we hope the identified correlations of RQ 2-1 and RQ 2-2 will provide valuable first insights.

Finally an effective quality assurance must be able to assess[4] the quality of the SRS in a way that is meaningful for the engineering endeavor. To this end, our results provide first indicators. On the one hand, the semantic quality of the SRS strongly impacts the outcome of downstream development activities, and consequently, SRS-based quality assurance can be very effective and must be considered during quality assurance. However, certain pragmatic quality factors, proposed by best practices, were not as influential as initially thought. Since semantic quality is difficult to assess, in particular for the predominant form of natural-language specifications, approaches providing reliable indicators for semantic quality based on syntactic properties (e.g., Bernárdez et al. [2004a]) seem promising.

[4] or establish means to improve, in terms of constructive quality assurance

3.4. Related Work

Unfortunately, the number of scientific research which specifically studies requirements *documentation* in practice is quite limited. For software engineering in general, Lethbridge et al. [2003] reported on three empirical studies on documentation in practice. The authors report that documentation is extensively created in software engineering, but frequently suffers from issues such as being poorly written, and in particular, frequently out of date. Relating those observations to the survey results presented in Sec. 3.1, the latter suggests to confirm our interpretation that outdated information is indeed one reason for the negative correlation between length of release cycles and SRS completeness. Furthermore, the authors report a moderate correlation (0.43, $p \leq .05$) between the accuracy of the documentation (as perceived by its audience), and the frequency of them consulting the requirements documentation. Therefore, while one might reasonably assume that communication for project with long release cycles would rely less on the requirements specification since it is more likely to be outdated, our results actually suggests the opposite; we observed a positive correlation between the length of release cycles and the importance of SRS for communication (τ=0.33, p=0.08). One potential explanation is that, in our study, we found empirical evidence of a characteristic of requirements specifications which is more important to practitioners than perceived accuracy in long projects, namely the persistent nature of artifacts. More recently, Liskin [2015] qualitatively studied the suitability of certain types of artifacts (e.g., GUI mockups, data models, user requirements) for generic activities related with requirements specifications (e.g., for "clarifying requirements"), by means of interviews, attesting a lack of traceability in state-of-the-practice requirements specifications which severely limits their utility. Regarding the impact of quality characteristics of documented requirements, Femmer et al. [2014b] identified passive voice as a pragmatic quality factor which leads to difficulties in understanding sentences. In contrast, Krisch and Houdek [2015] provide empirical evidence that the presence of passive voice in automotive specifications rarely causes problems. The authors state that this was due to "context and discourse analysis as well as expert knowledge often [helping] to identify the actor of a requirement reliably". As mentioned in the discussion, this strengthens our confidence that we need to carefully evaluate both the quality attributes itself and the context of use to determine their impact on subsequent activities.

Furthermore, there exists empirical evidence providing insights into how requirements are *communicated*. Abelein and Paech [2014] conducted a series of semi-structured interviews concerning the state of the practice of user-developer communication in large-scale IT projects. The results of their study indicate that the direct user-developer communication is limited and that no common method for this communication in the design and implementation method exist. We extend the context of their work by adding multiple stakeholders (i.e., users of the SRS) while also focusing on the SRS as the single means for communication. Bjarnason et al. [2011] conducted an explanatory case study in order to deepen their understanding of the cause and effects of communication gaps in a large-scale industrial setup. Their results show that communication gaps cause failure to meet

the customers' expectations, quality issues, and wasted effort. In contrast to our work, their study is of explanatory nature, and furthermore has a larger scope (communication gaps in general).

However, besides the lack of studies focusing on the significance of documented requirements and their quality, success factors of and problems in requirements engineering in general have been and still are a vivid field of research, with many results also applying to documented requirements and their communication. For instance, independent studies conducted by The Standish Group [1995] and Verner et al. [2005] report that factors such as incomplete requirements are among the most significant determinants for project success or failure. In a family of surveys initially carried out in Germany and later replicated in Brazil, Méndez Fernández and Wagner [2014] and Kalinowski et al. [2016] report that practitioners perceived incomplete or underspecified requirements as being a reason for poor product quality and time/cost overrun, which in turn lead to unsatisfied expectations and customers. In contrast, Damian and Chisan [2006] report that effective RE improves the product quality and productivity of downstream development activities, such as implementation or testing, and specifically point out that validating requirements leads to improved feature coverage.

Contribution to the State-of-the-Art With respect to the current state-of-the-art, the survey and experiment presented in this chapter contribute novel empirical evidence by specifically challenging the significance of the *documented* requirements and their quality. In particular, to the best of our knowledge, the survey is the first to systematically study the impact of (project) criteria on the extent of documentation and its relevance for communication, and the experiment provides first quantitative results on the extent incorrect and negated statements in documented requirements actually impact the outcomes of system testing. By discussing both results and relating them to the existing empirical evidence, we were able to provide insights on the impact of the requirements specification's quality on downstream development in practice, and identified novel hypothesis for future research.

3.5. Summary

In this chapter, we studied two complementary research objectives, namely the extent the requirements specification is created and used, and its impact on downstream development, by means of a survey with practitioners and a controlled quasi-experiment, respectively.

Regarding the first objective, we found that the requirements specification is commonly used to document and communicate requirements in practice, especially for project participants not involved during requirements engineering. More precisely, our results suggests that the requirements specification's quality in particular matters for safety/security-relevant systems or if release cycles are rather long. Otherwise, especially in settings where requirements are volatile and the development team is rather small, has a low turnover and previously successfully collaborated with the customer, requirements are

more likely to be less extensively documented and used for communication, hence limiting the impact of the requirements specification on downstream development.

Regarding the second objective, we studied whether the quality of documented requirements impacts the efficiency and effectiveness of downstream development. To this end, we conducted a controlled quasi-experiment which quantitatively confirmed that incorrect specifications, even if rather obvious, substantially impact system testing, while negated sentences do not. Our results suggest that any form of incorrect or missing relevant information (see Sec. 2.2.1 for a high-level taxonomy) strictly constitutes a valid quality attribute, whereas we believe pragmatic quality to be rich on facets we do not yet properly understand with some factors having more severe impacts than others.

Consequently, means for quality assurance of requirements specifications must evaluate its efficiency and effectiveness depending on the specific context of use, not neglect semantic quality, and carefully select quality characteristics regarding understandability.

4 Chapter

A Quality Definition Model for Requirements Specifications

So far, we provided an intentional characterization of requirements specification quality (Chap. 2) to establish a common understanding, and motivated the need for its assurance by showing its significance on downstream development of software-intensive systems in practice (Chap. 3). In this chapter, we complete our treatise of quality definition by providing an extensional definition by means of a quality definition model for requirements specification:

Requirements Specification Quality Definition Model: *A model with the objective to describe the characteristics of the requirement specification that bear on its ability to inform the audience in order to build the desired system with least effort.*

The use of a model allows us to overcome the imperfect understanding of the overwhelmingly complex concept of requirements specification quality in reality by focusing on certain aspects and with limited details by virtue of abstraction. The model's goal is to provide a deeper understanding of the characteristics which constitute quality and thereby establish a well-founded foundation for quality assessment in Part II of this thesis. To this end, in this chapter we contribute:

1. A systematic approach to identify valid and precise properties of the requirements specification based on an analysis of the (development) activities it impacts
2. A quality definition model which consists of a comprehensive and organized set of

precise and well-founded quality attributes of the requirements specification, and a tailoring procedure to adapt this model for a specific context of use

More specifically, we propose an approach called Activity-Based RE Quality Model (ABRE-QM), which consists of a meta-model and a complementary procedure that can be applied to systematically identify properties of the requirements specification properties based on an analysis of the specification's audience and their activities and tasks (contribution 1). We apply this approach to analyze the activities of the (Rational) Unified Process (RUP) as described by Jacobson et al. [1999]. This (description of the) process model qualified for our purpose since (i) the activities are described in sufficient detail to enable an analysis, (ii) include the most commonplace activities required for software development, in particular for information systems, and (iii) explicitly defined the inputs for each activity. Furthermore, the intrinsic properties identified in the previous step extend a taxonomy of high-level attributes obtained by thoroughly studying and refining the quality attributes proposed in the current body of knowledge, resulting in an integrated quality definition model (contribution 2).

The remainder of this chapter is structured as follows: First, we provide an overview of the elements of the quality definition model and describe our contributions in more detail in Sec. 4.1. Next, in Sec. 4.2, we present the ABRE-QM approach for identifying quality attributes based on the activities the requirements specification is used in. Before presenting the quality definition model, we first describe our research method in Sec. 4.3. Subsequently, we present a taxonomy of quality attributes obtained by a systematic analysis of the current body of knowledge in Sec. 4.4. In Sec. 4.5, we extend this taxonomy with precise quality attributes obtained through applying the ABRE-QM approach to the Unified Process, thereby obtaining an integrated quality definition model. In Sec. 4.6, we critically reflect on limitations and outline applications of the quality model, which includes a proposal of a tailoring procedure to customize the quality definition model for a specific context. Finally, we discuss related work in Sec. 4.7 before summarizing our results in Sec. 4.8.

This chapter is partly based on previously published material [Femmer et al., 2015].

4.1. Overview

In this section, we give an overview on the quality definition model, and, based thereon, describe our contributions in more detail. Fundamentally, we organize quality attributes along two orthogonal dimensions, namely the (primary) entity considered and the context-specificity of the quality definition. Regarding the former, the model distinguishes but explicitly relates a quality-in-use perspective on the effects the requirements specification has on downstream development activities with intrinsic attributes inherently associated with the specification itself. Regarding the latter, we distinguish between a context-specific and generic quality definition. We provide both an approach to develop an activity-based quality definition which is valid for a particular context of use, and a generic quality definition model which serves as a reference model whenever the context

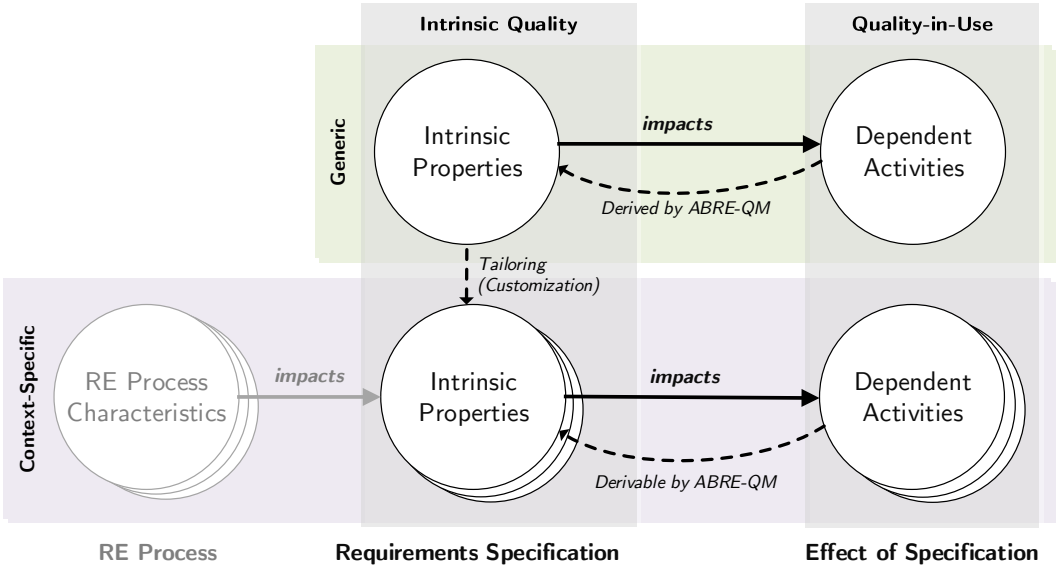


Figure 4.1.: Overview of the quality definition model and adjacent concepts

is unknown or as a starting point for customization for a specific context. Fig. 4.1 illustrates the concepts (i.e., classes of characteristics) and their relationships, which we subsequently describe in more detail.

4.1.1. Intrinsic Quality and Quality-in-Use

For defining the quality of requirements specification, we distinguish between characteristics of the RE process, the requirements specification artifact itself, and the downstream development activities it impacts, and we refer to the corresponding classes as RE process quality, intrinsic quality, and quality-in-use, respectively. Subsequently, we clarify which characteristics (or attributes) belong to those classes and discuss their relationship (also illustrated horizontally in Fig. 4.1).

Quality-in-Use The notion of quality-in-use results from the understanding of requirements engineering as a means to an end in developing software-intensive systems. It refers to the effects the requirements specification has on development activities when used by a specific actor (i.e., member of the specification's audience), in terms of efficiency, effectiveness, satisfaction, and context flexibility (see Tab. 4.1 for a description). Therefore, it is inherently associated with activities which depend on the requirements specification, such as analyzing requirements in order to design a high-level system architecture, implementing a certain feature, or developing a test strategy for system testing. In this thesis, we subsequently consider efficiency and effectiveness only due to a limited understand-

ing of the requirements specification's impact on satisfaction and context flexibility. For example, the number of defects in the test-cases derived by test engineers, as empirically studied in the previous chapter (Sec. 3.2), is a quality-in-use characteristic for the system testing activity.

Characteristic	Description
Effectiveness	The accuracy and completeness of the work results (e.g., source code, architecture, test cases) obtained as a result from development activities and tasks which depend on the SRS as an input. (ISO 9241-11)
Efficiency	The resources expended in relation to the accuracy and completeness of the obtained work results. Relevant resources can include time to complete the task (human resources), materials, or the financial cost of usage, e.g., due to utilization-based tool licensing. Formerly called 'productivity'. (ISO 9141-11)
Satisfaction	The degree to which the needs of the audience are satisfied when the SRS is used. Satisfaction is the response of the audience to interaction with the SRS, and includes attitudes towards its use. It can be further distinguished as: **Purpose accomplishment** The degree to which the audience is satisfied with the perceived achievement of pragmatic goals, including acceptable perceived results of use and its consequences. **Trust** The degree to which the audience is convinced that the SRS can be used as intended. **Pleasure** The degree to which the audience obtains pleasure from fulfilling their personal needs associated with the development activities. Personal needs include to acquire new knowledge and skills, to communicate personal identity and to provoke pleasant memories.
Context Flexibility	The degree to which a SRS can be used with effectiveness, efficiency and satisfaction in contexts beyond the one initially identified. Flexibility can be achieved by adapting a SRS for additional engineers, tasks and organizations, and enables a SRS to take account of circumstances, opportunities and individual preferences that may not have been anticipated in advance.

Table 4.1.: Quality-in-use characteristics for (development) activities depending on the requirements specification (adapted from ISO/IEC 25010:2010 [2010])

Intrinsic Quality In contrast, intrinsic quality denotes inherent properties of the requirements specification which impact the quality-in-use. Inherent properties may concern the specification's physical representation, its syntax or its semantics. For instance, the size of a requirements specification, the linguistic complexity of natural-language sentences and omission of necessary system functionality are inherent properties and commonly considered to impact quality-in-use, and hence are characteristics (or attributes) of intrinsic quality. Note that while intrinsic quality attributes do not require to actually perform a development activity, for instance in order to measure them, their validity still

depends on those activities since inherent properties only qualify as quality characteristics if they indeed impact the quality-in-use.

Contribution We provide a hierarchical model which comprises intrinsic quality attributes at different levels of abstraction. While at a high-level, we rely on the quality framework introduced in Sec. 2.2 and a thorough analysis of attributes resulting from a literature survey, we further substantiate those attributes by means of an analysis of the activities the requirements specification is required for. Furthermore, our model also explicitly specifies the impacts of those intrinsic quality attributes on the quality-in-use of those activities, thereby providing a rationale why the quality attribute is indeed valid as well as an estimation of the predicted quality-in-use.

Side Node: RE Process Quality Intrinsic quality is not determined per se but results from requirements engineering activities. Hence, the characteristics of those activities, i.e., the RE process quality, in turn impact to what extent a requirements specification entails certain intrinsic properties. While not further examined in this thesis, we will revive this thought when discussing the limitations of our model in Sec. 4.6 and future research in Chapter 8.2.

4.1.2. Generic and Context-Specific Quality Definition

In general, the validity of a quality definition depends on the specific context of use of the requirements specification since intrinsic properties do not inevitably lead to the same effects on quality-in-use under all circumstances.

Context of Use The context of use refers to (characteristics of) the environment which affects development activities that depend on the requirements specification as an input. Tbl. 4.2 structures and briefly describes exemplary characteristics which are supposed to influence the effect certain inherent properties have on the quality-in-use of downstream development activities.

Problems with Context-Specific Quality Definitions Unfortunately, since in general the precise context of use is unknown and quality-in-use is not revealed until the activities are carried out, one cannot be certain which intrinsic properties actually constitute quality attributes up until this point in time. This implies two fundamental limitations:

- *Late Determination:* Early quality assurance requires to obtain meaningful estimations during or shortly after the requirements are documented in order to take corrective actions immediately, if necessary. At this point, however, not many downstream development activities are typically started, and the context of use may still be quite uncertain.

Class	Description and Examples
Problem Domain	Characteristics which are inherently associated with the problem the system under development is supposed to solve. Examples are *novelty*, i.e., the newness of the system under development with respect to the state of the practice, the *susceptibility of changes*, and therefore frequent changes to requirements during project term, or the *domain* the system under development should operate in and which implies certain standards, norms, regulations and laws to be considered.
Infrastructure	Characteristics related to the technical infrastructure present, for instance, the *tools* used for development, e.g., IDEs such as Eclipse or requirement tools such as DOORS.
Organization of Work	Characteristics related to the devised organizational structure which defines how development tasks are divided, grouped and coordinated to achieve a common goal. Examples include the frequency members leave and or new members enter the team (*team turnover*), the *type of contract*, e.g., fixed-price vs. time & material, which leads to different strategies in handling change requests, or the extent to which tasks are delegated to *sub contractors*.
Resource	Characteristics related to scarce resources employed for the software development endeavor, in particular, the the allocated *budget* and the projected *schedule* and deadlines.
Audience	Characteristics related to the people performing the activities. Examples are the *domain knowledge* about the environment in which the system operates, the *experience* of the audience in performing the development tasks, and the extent to which the audience is *familiar with the language(s)* used in the specification.
Solution	Characteristics related specifically to the solution, for instance, the *size* of the software solution (since additional activities for managing large implementations may be required), the extent to which *commercial-of-the-shelves (COTS)* components are integrated, or the *programming language(s)* the system is implemented in.

Table 4.2.: Exemplary characteristics of the context of use influencing quality-in-use, organized in classes (based on Basili and Rombach [1987] and Kalus and Kuhrmann [2013])

- *Limited Re-Use:* Since a concrete context of use will never be exactly the same, it is not ensured that requirements specifications with equal intrinsic properties will result in the same quality-in-use. Therefore, any new project would always require to derive specific intrinsic properties for the particular context, which, in turn, would severely limit re-use of a quality definition and thus result in high costs.

Contribution To overcome these limitations, we propose a generic quality definition model, i.e., we specify intrinsic properties of the requirements specifications not limited to one particular context but which aim to adequately define quality for several contexts of use. It is based on abstract activities (e.g., deriving test cases) performed by roles (e.g., a test engineer) which results in more abstract properties (e.g., whether means

for navigation among related requirements are provided), compared to a context-specific quality definition. However, since the development activities are not performed in a vacuum, some basic assumptions about the context of use are inevitable. To this end, we considered activities to be performed according to some benchmark (e.g., a process model), and involved roles to have a reasonable level of technical qualification without further assumptions, e.g., expert knowledge about the system's domain or familiarity with the requirements specification's structure.

Before using the generic quality definition for a concrete project, we envision that the intrinsic properties are first carefully selected and refined for the particular context by a quality manager or requirements engineer, and we outline this process in Sec. 4.6.2. As shall be seen in Part II (Sec. 5.2, pp. 114), we will refer to the outcome of this customization as the (context-specific) *quality demand*, which will be of major concern for quality assessment. While the generic model provides both a point of reference for the quality assessment model presented in Chapter 6 and a starting point for customization to obtain a context-specific quality model, we also provide an alternative based on a thorough project-specific analysis using the ABRE-QM approach (see Sec. 4.2). Which approach is more appropriate for a given situation must be decided individually based on the demands concerning re-usability and validity and the constraints regarding time and costs. Moreover, this trade-off can be balanced in terms of a middle ground between a context-specific and generic definition by defining quality for a certain company or an organizational unit.

4.2. Activity-Based Quality Modeling (ABRE-QM)[1]

We strictly understand RE as a means to an end for software- and systems engineering with the goal to produce the desired software-intensive systems in a systematic and predictable way. Hence, as described in the previous section, quality of the requirements specification cannot be defined on its own but must be evaluated regarding its fitness for down-stream development. To this end, we provide an approach to define the quality of requirements specifications following an activity-based approach as proposed by Wagner et al. [2012] for software quality. Essentially, this approach, which results in what we call an activity-based requirements engineering quality model (ABRE-QM), studies the activities the requirements specification is used in, in order to identify exactly those intrinsic properties which impact its audience (e.g., architects, testers, etc.) in their ability to carry out their activities efficiently and effectively, i.e., the quality-in-use. In this section, we provide a meta-model which describes the elements to be represented in such a quality model, and describe a process to obtain it.

[1] The ABRE-QM approach presented in this section is based, in parts, on work previously published in [Femmer et al., 2015, Section III (pp. 2–3)]. The author of this thesis substantially contributed to the paper's contents which are recapitulated in this section, in particular, to the conceptualization of the approach and the meta-model. The approach presented in this section adapts and slightly extends the approach originally reported in the paper.

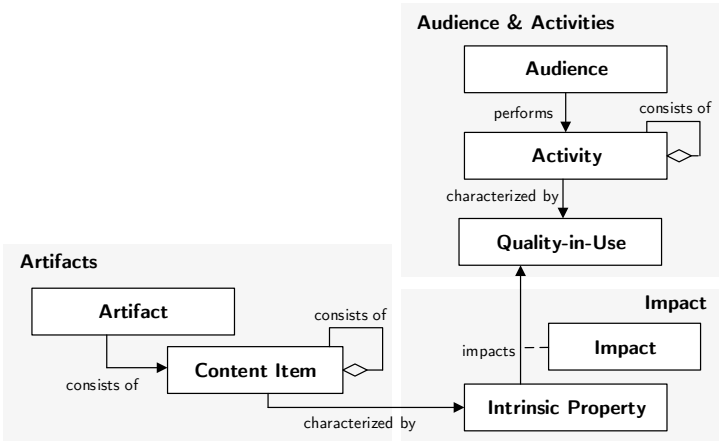


Figure 4.2.: Meta-model of the activity-based RE quality modeling (ABRE-QM) approach (adapted and extended from Femmer et al. [2015])

Meta-model The meta-model is illustrated in Fig. 4.2. Basically, it consists of three fundamental parts representing the artifacts of the requirements specification, the activities which depend on them, and the intrinsic properties of the former that impact the latter. The requirements specification is modeled in terms of *artifacts* and *content items*. While the former refers to self-contained work results (see, e.g., Méndez Fernández and Penzenstadler [2014] for a reference model of requirements specification artifacts), the latter refers to elements of it. For instance, a `usage model` is an artifact which commonly comprises several `use-case` content items. Content items may be further divided into their parts, e.g., the `precondition`, `actor`, `main steps`. The purpose of the requirements specification is modeled in terms of the dependent activities. In addition to the *activities* themselves, an ABRE-QM specifies the *audience* members (roles) performing those activities and the characteristics considered a *quality-in-use* of the particular activity (see Tbl. 4.1). Finally, *intrinsic properties* characterize content items and *impact* (the quality-in-use of) one or more activities performed by the audience, thereby providing the missing link between the requirements specification and its purpose. For each intrinsic property, the model must provide the following information: (i) The quality-in-use of one or more activities impacted by the property. Thereby, we obtain a structured quality model from those activities, e.g., if the activity is to create a system test case, we obtain a definition of testability in the model. (ii) Furthermore, for each intrinsic property and each activity/quality-in-use, a rationale must be provided. This rationale includes a reason, i.e., a justification why an artifact or content item which possesses the intrinsic property impacts the associated quality-in-use of the activity in terms of costs, schedule or quality, and a source from which this impact was derived and which can provide further information, such as a requirements quality standard or corporate

guidelines[2]. (iii) Finally, the model must provide the artifact (or content item) to which the intrinsic property applies. Since the ABRE-QM focuses on the *impact* relation, we intentionally refrained from modeling other relations (such as the relation between the audience and the artifacts).

Obtaining an ABRE-QM In this paragraph we outline a basic process which describes how a quality model can be obtained based on a through analysis of the audience and their activities. While this process is by no means prescriptive, we argue that a systematic methodology which identifies and analyzes all audience members, activities and artifacts contributes to a more complete quality model.

Essentially, the process consists of three steps corresponding to the main parts of the meta-model (Fig. 4.2). These three steps are not necessarily performed sequentially, and iterated until all members of the audience and system stakeholders are content with the result and/or no novel intrinsic properties are identified.

1. **Determine artifacts and content items.** First, we must determine the artifacts and content items for which we define quality. Therefore, documentation of past projects (e.g., in version management or issue tracking systems) provides a good overview which artifacts are used in a certain context. If unavailable, selecting relevant artifacts from a reference model (see, e.g., Méndez Fernández and Penzenstadler [2014]) is an alternative. Next, the identified artifacts are decomposed into content items. Here, the table of contents, self-contained sections or fields (e.g., in a requirements management tool or from templates) are helpful.

2. **Identify audience and activities.** In this step we first identify project participants who rely on the requirements specification for their activities, i.e., its actual audience. Since they are the persons required to understand the system's requirements[3], they must be involved in developing the ABRE-QM. Accordingly, missing relevant participants or including unnecessary participants can lead to an incomplete and/or invalid quality definition. A project lead or process engineer is usually a good starting point for finding out who interacts with the requirements specification.

   Next, we must identify what activities the audience members perform based on the requirements specification. For a concrete project, a good opportunity is to ask the identified audience members how they use the requirements specification. Coarse-grained activities (e.g., create a test case) have to be refined into smaller ones (e.g., identify the precise inputs and their order) until a thorough understanding of how a specific audience member interacts with the requirements specification is achieved.

3. **Identify intrinsic properties and their impacts.** Based on the previous steps, we have to identify intrinsic properties of the artifacts (and content items) which

[2]For simplification, we did not include these in Fig. 4.2.

[3]see Sec. 2.3 (pp. 2.3) for our understanding of the role of the audience and the requirements specification in software- and systems engineering.

impact the quality-in-use of the activities. Therefore, the identified activities must by analyzed: What helps or hinders to perform this activity and why? What helps or hinders to create better results of this activity in less time? We see several options to achieve this. For instance, in this thesis, we relied on a researcher to study the activities in detail to extract intrinsic properties. In contrast, Femmer et al. [2015] interviewed practitioners to evaluate quality attributes described in guidelines provided by the company. Finally, the identified intrinsic properties are explicitly linked to the quality-in-use of activities via impacts.

Example The following short example illustrates the approach. **1. Artifacts and Content Items:** Functional requirements of information system are commonly specified in `use-cases` (e.g., Cockburn [1998]). Use-cases represent the interactions between the system and its environment and consist of a (main) sequence of `actions` which are either performed by the system or by some actor. **2. Audience and Activities:** For the sake of example, consider an `architect` who is responsible for designing the high-level software architecture. One particular task in designing the architecture is to `identify similar functionality` in order to encapsulate it in a coherent module. To this end, the architect reads the actions described in the use-cases, and based thereon, aggregates similar actions of the system into module candidates. **3. Intrinsic Properties:** With this particular activity in mind, we see that the presence of homonyms in actions can lead to the architect not allocating similar functionality into a coherent module, while synonyms can lead to the opposite. Hence, a specification in which the actions are `named lexically consistent` enables the architect to design more coherent modules, and therefore, a more adequate software architecture. For the remainder of this chapter, we will denote a positive ('+') impact on efficiency (producitivity) and/or effectiveness (system quality) using the following shorthand:

$$\textit{Lexically consistent names @actions} \xrightarrow{+Q} \textit{Design software architecture}$$

As shown in Femmer et al. [2015], this approach can be applied with a specific context-of-use in mind, resulting in a context-specific quality model but incorporating its downsides (see Sec. 4.1.2), or to a software development process model, resulting in a generic quality definition which potentially requires further customization to be adequate for a specific context. With regards to the latter, the application of the ABRE-QM approach constitutes one central step to obtain the quality definition model presented in this chapter, as shall be seen in the next section.

4.3. Research Method

In this section we describe the research method followed in order to obtain a comprehensive, precise and well-founded quality definition model which serves as a reference for both the rest of this thesis as well as a starting point for further customization for a particular context-of-use.

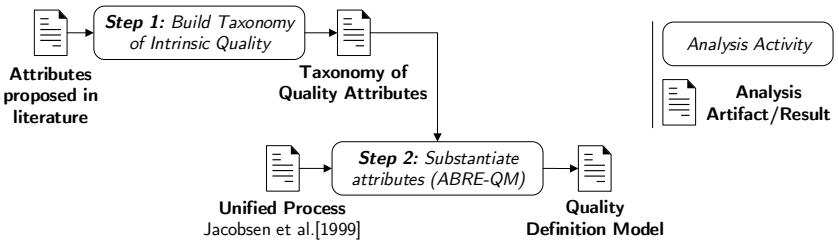


Figure 4.3.: Overview of the research methodology's main activities and results

4.3.1. Overview

Basically, our research method consists of two main steps as illustrated in Fig. 4.3. The first step is concerned with obtaining a taxonomy of high-level quality attributes based on a literature survey of quality models, thereby providing a comprehensive but rather abstract definition of quality. The results of this step are described in Sec. 4.4. Furthermore, in a second step, we substantiate those quality attributes. To this end, we apply the ABRE-QM approach described in the previous section to the Unified Process [Jacobson et al., 1999], and classify the resulting quality attributes within the taxonomy from the first step. The results of this step, including the final quality definition model, are described in Sec. 4.5. Subsequently, we explain those steps in more detail.

4.3.2. Step 1: Taxonomy of High-Level Quality Attributes

Scientific publications and industry standards propose numerous properties to be demanded of a requirements specification. However, those properties are often defined only vaguely, use terms inconsistently, lack a clear structure and do not necessary constitute intrinsic quality characteristics. In this section, we describe our methodology to obtain a hierarchical taxonomy of intrinsic quality attributes based on a thorough survey of the current body of knowledge, including precise and narrow notions and consistent terminology of quality attributes.

To this end, we proceeded as illustrated in Fig. 4.4. First, we extracted quality attributes from selected publications, including secondary literature such as Davis et al. [1993a], and well-known international standards (e.g., IEEE Standard 830 [1998], ISO/IEC 25010:2010 [2010]), by means of snowballing and to the best of the author's knowledge until saturation was achieved. In this way, 41 initial quality attributes were identified. Subsequently, the notions were revised and consolidated by pairwise comparing the definitions of all attributes in order to identify even subtle details, thereby resolving homonyms (e.g., properties subsumed by *complete*) and synonyms (e.g., *feasible* and *achievable*). This step resulted in the 27 elements presented in Sec. 4.4.1. Next, 11 attributes were discarded due to being considered inappropriate for an intrinsic quality definition model as described in Sec. 4.4.2. Finally, the remaining 15 attributes were

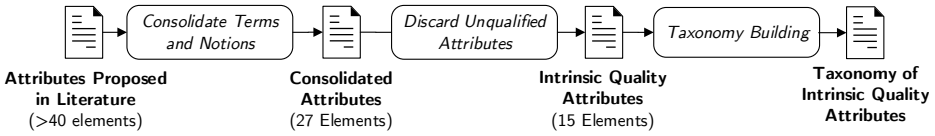


Figure 4.4.: Step 1: Literature-based method for obtaining a taxonomy of intrinsic (high-Level) quality attributes

arranged in a hierarchical quality model and organized according the quality framework presented in Sec. 2.2. Rationales are given for discarded elements and the identified specialization relationships between quality attributes as part of taxonomy construction.

4.3.3. Step 2: Substantiation of Quality Attributes

Subsequently, we present our methodology to substantiate the quality attributes identified in the taxonomy in the first step. Specifically, we seek to identify fine-grained quality attributes which specify concrete properties of the taxonomy's high-level quality attributes and/or narrow down the artifacts and content items they apply to. For instance, we want to understand which quality attributes make a requirements specification organized, or which information must be stated quantitatively precise. In addition, we also seek to make the impact of all quality attributes, the substantiated ones as well as the high-level ones, explicit by linking them to the corresponding activities and quality-in-use characteristics. Essentially, this particular step is based on applying the ABRE-QM approach and is again divided into three sub-steps. First, we need to identify the audience and activities which depend on the requirements specification. Second, we must analyze those activities to identify precise and profound quality attributes. For those two steps, we follow the process described in Sec. 4.2 and apply it to the (Rational) Unified Process (RUP). Finally, we manually classify the thereby identified (low-level) quality attributes within the intrinsic quality taxonomy of the first step. The method is illustrated in Fig. 4.5. Subsequently, we provide further details on how we applied ABRE-QM to RUP and how we integrated the resulting quality attributes into the taxonomy.

Identifying Quality Attributes based on the Unified Process

We deliberately chose to apply ABRE-QM to the (Rational) Unified Process (RUP) because it is widely known in both academia and industry, its activities are refined to the level of individual tasks which are described in detail, and so are the expected output artifacts created by the activities. Jacobson et al. [1999] therefore provides us with the information necessary to conduct steps 1 and 2 of the ABRE-QM procedure described in Sec. 4.2. Note that although the identification of quality attributes is based on RUP activities, it does not strictly depend on the actual order of how those activities are

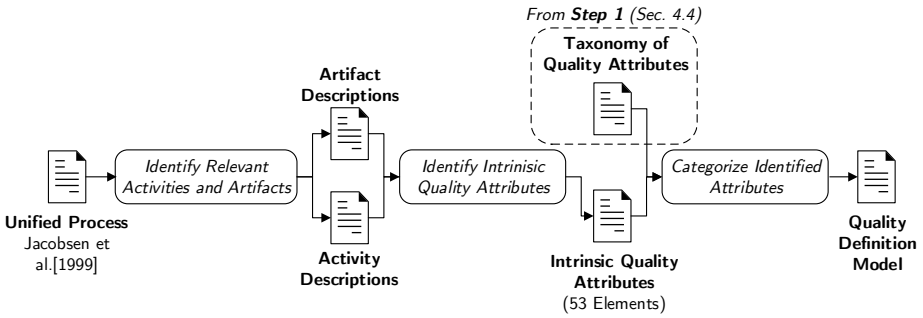


Figure 4.5.: Step 2: Obtaining low-level (precise) intrinsic quality attributes by applying ABRE-QM to the (Rational) Unified Process

performed, and thus may be applicable to software- and systems development projects based on the same or similar activities and artifacts.

Applying ABRE-QM to the Unified Process In the first step, we extract the requirements specification's artifacts, audience, and activities directly from Jacobson et al. [1999] (steps 1 and 2). In order to determine the quality attributes and their impacts (step 3), we analyze the descriptions of the activities and their output artifacts, marking all occurrences where an intrinsic property of an artifact or entity of the requirements specification can impact elementary tasks of the corresponding activity. Each such property is formulated in such a way that the impact on the associated activity is positive, and a rationale for it is given.

Example: Design System Test We demonstrate the identification of intrinsic properties on the activity *designing system test cases* [Jacobson et al., 1999, p. 308]. According to this description, one elementary task in designing a system test is to *identify and describe test cases*. This task is performed by the *test engineer* and as an output, system *test cases* (pp. 297) are obtained. By analyzing the associated activity and artifact descriptions, we identified 7 intrinsic properties for this particular task and output artifact as illustrated in Fig. 4.6, with the relevant text segments of the definitions highlighted and the impacted quality-in-use characteristic specified.

For instance, we identified the following property:

$$\textit{Inputs/outputs specified at system boundary @Use-Cases} \xrightarrow{+P,Q} \textit{Describing test cases}$$

The provided rationale states that the test engineer must otherwise deduce this information, which would result in more time spend and can lead to flaws in the test cases, hence affecting both efficiency and effectiveness. Appendix C describes the identified intrinsic properties, including those mentioned in the figure for testing, in more detail.

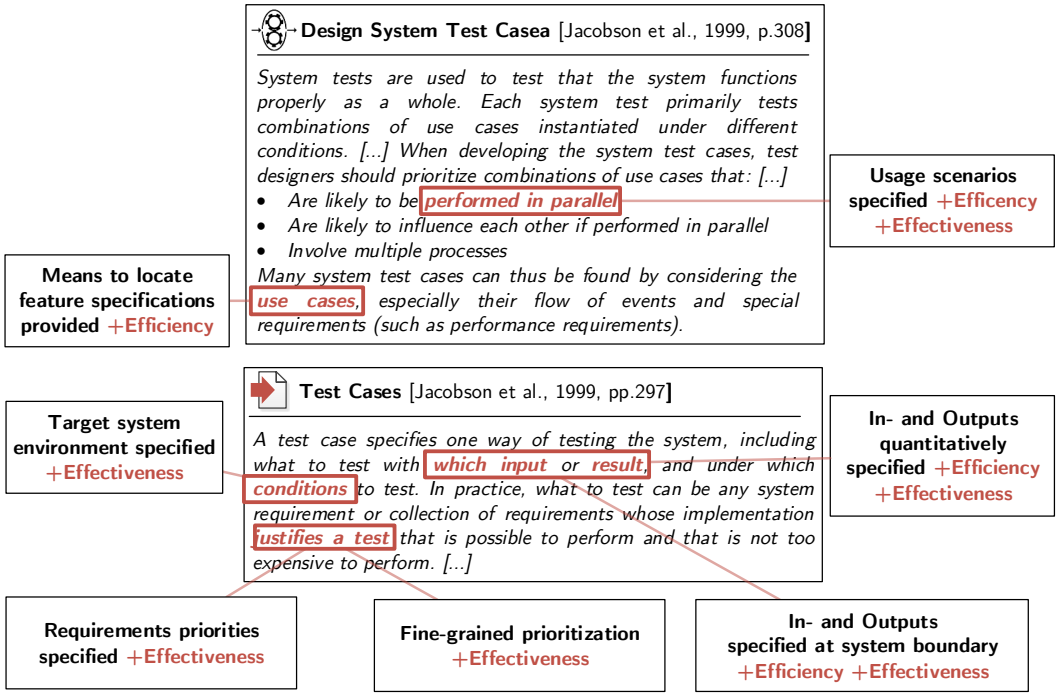


Figure 4.6.: Exemplary application of ABRE-QM to the activity *Identify and describe test cases* and the resulting *test case* artifacts according to the Unified Process [Jacobson et al., 1999].

Classifying the Attributes within the Taxonomy

Finally, we relate the taxonomy based on quality attributes suggested in literature (Sec. 4.4) to the intrinsic properties which were obtained from applying ABRE-QM to the Unified Process (Sec. 4.5), thereby substantiating the definitions of the former by the latter. Compared to the properties in the taxonomy, the intrinsic properties identified using ABRE-QM approach are more narrow in scope. Therefore, we associate each intrinsic property identified using ABRE-QM with those from the taxonomy if the former is regarded as a specialization of the latter by manual classification based on the definitions given in Sec. 4.4.1. Here, a specialization refers to the case that a property more precisely states the entity (information) that it must hold for, or refines the property itself by considering a particular aspect of it. For instance, consider the properties *Target system environment specified* (ID 47) and *No homonyms/synonyms used for actors, actions and objects* (ID 13) identified using ABRE-QM. The former states the particular entity that is required to be specified in order to effectively test the system under development, and hence contributes to a more substantial understanding of *information completeness* in the taxonomy. The latter refers to particular aspects of *lexical consistency*, namely

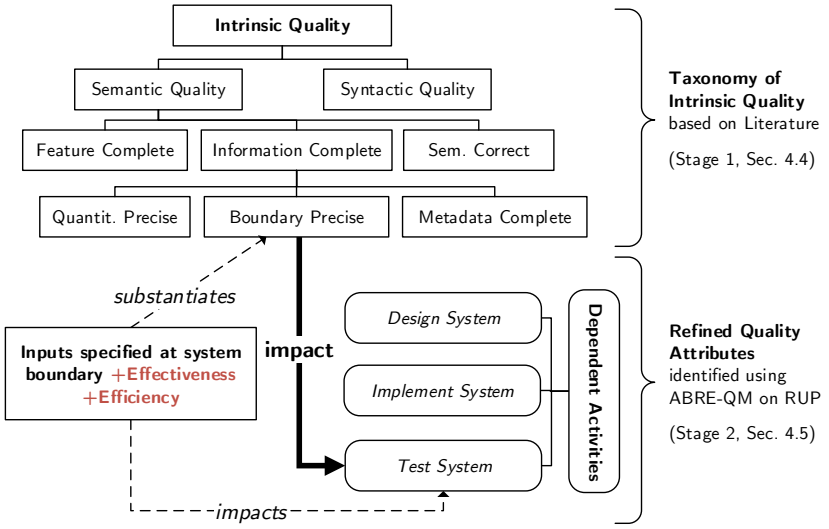


Figure 4.7.: Obtaining an integrated quality model: intrinsic attributes identified by ABRE-QM substantiate those of the taxonomy, thereby revealing their impact on the quality-in-use of downstream development activities.

homonyms and synonyms, while also specifying the content items this applies to and thereby substantiating its definition.

The relationship between the intrinsic qualities of the taxonomy and those identified by ABRE-QM is illustrated in Fig. 4.7. Here, the property *Inputs specified at system boundary* is identified to influence system testing as already seen in the example in Sec. 4.3.3, and substantiates the understanding of *boundary precise* of the taxonomy. Hence, a specification that is boundary precise is less expected to cause flawed test cases or schedule overruns during system testing. As a result, we obtain an integrated quality definition model which provides an extensive characterization of high-level quality attributes by the low-level attributes associated with it (see Sec. 4.5.1), as well as an understanding about their impact on downstream development (see Sec. 4.5.2).

4.4. A Taxonomy of Intrinsic Quality

In this section, we propose a taxonomy of quality attributes based on an analysis of the scientific literature and industrial standards, based on the first step of our research methodology as described in Sec. 4.3.2.

4.4.1. Quality Attributes of Requirements Specification

Out of the 41 attributes extracted from 11 sources [ISO/IEC 29148:2011, 2011, Wiegers, 1999, Davis et al., 1993a, Ott, 2012, Berry et al., 2006, Fabbrini et al., 1998, IEEE Standard 830, 1998, Pohl, 1994, Kotonya and Sommerville, 1998, Rupp and SOPHISTen, 2007, Meyer, 1985], we identified the following 27 unique attributes:

Feasible A set of requirements is *feasible* [ISO/IEC 29148:2011, 2011, Wiegers, 1999] (*achievable* [Davis et al., 1993a], *affordable* [ISO/IEC 29148:2011, 2011], *realizable* [Rupp and SOPHISTen, 2007]) if there exists a solution satisfying the requirements which (i) does not require major technology advances [ISO/IEC 29148:2011, 2011, Davis et al., 1993a] and (ii) is obtainable within the constraints (e.g., costs, schedule, legal, regulatory, staff) [ISO/IEC 29148:2011, 2011, Wiegers, 1999, Davis et al., 1993a, Rupp and SOPHISTen, 2007].

Therefore, feasibility is a function of both the requirements specification and external factors regarding the project (e.g., budget and schedule constraints), company (e.g., team skills) and socio-economic (e.g., legal constraints) context.

Verifiable In literature, requirements are concordantly considered *verifiable* [IEEE Standard 830, 1998, Davis et al., 1993a, Wiegers, 1999, ISO/IEC 29148:2011, 2011, Rupp and SOPHISTen, 2007] (*testable* [Berry et al., 2006, Ott, 2012]) if there exists a method to check whether the system satisfies the requirement. In other words, no feature of the product is defined in such way that a candidate solution cannot realistically be validated with respect to this feature [Meyer, 1985]. In literature, requirements are concordantly considered *verifiable* [IEEE Standard 830, 1998, Davis et al., 1993a, Wiegers, 1999, ISO/IEC 29148:2011, 2011, Rupp and SOPHISTen, 2007] (*testable* [Berry et al., 2006, Ott, 2012]) if there exists a method to check whether the system satisfies the requirement. In other words, no feature of the product is defined in such way that a candidate solution cannot realistically be validated with respect to this feature [Meyer, 1985].

The method is supposed to be finite and cost effective [Davis et al., 1993a], and the definitions remain vague in terms of the required certainty about the requirements' satisfaction, with methods based on evidence [ISO/IEC 29148:2011, 2011], e.g., testing [Wiegers, 1999, Ott, 2012, Berry et al., 2006] or measures [Rupp and SOPHISTen, 2007], and/or proof [Wiegers, 1999, Davis et al., 1993a, ISO/IEC 29148:2011, 2011] mentioned.

Singular A requirement specification is *singular* [ISO/IEC 29148:2011, 2011] (also called *atomic* [Ott, 2012]) if every requirement therein cannot be usefully separated into several requirements. This includes that no single statements define several requirements using conjunctions [ISO/IEC 29148:2011, 2011].

Design-Independent A requirement specification is *design-independent* [Davis et al., 1993a] (also called *implementation-free, bounded* [ISO/IEC 29148:2011, 2011]) if no requirement imposes limitations on potential design and implementation decisions be-

yond necessity, i.e., no unnecessary implementation details are described [Ott, 2012, ISO/IEC 29148:2011, 2011]. Meyer [1985] demands that any feature must correspond to the problem but not a possible solution.

Note that the definition of Davis et al. [1993a] which states that "there must exist more than one system design and implementation correctly implementing all requirements" is too weak to adequately express the attribute. Instead, the set of system designs and implementations satisfying the desires or needs of the customer must be equal the set of system designs and implementations described by the SRS.

Quantitatively Precise A requirement specification is *quantitatively precise* (also called *precise* [Davis et al., 1993a] or subsumed by *completeness* [ISO/IEC 29148:2011, 2011, Ott, 2012, Pohl, 1994] if numerical quantities with appropriate precision are specified whenever possible [Davis et al., 1993a, Ott, 2012].

Feature Complete A requirement specification is *feature complete* (mostly called or subsumed by the notion of *complete* in literature) if no requirements [Kotonya and Sommerville, 1998], i.e. capabilities and characteristics [ISO/IEC 29148:2011, 2011], are missing in order to meet the stakeholders' needs [ISO/IEC 29148:2011, 2011]. Meyer [1985] demands that each feature of the problem is covered by a text element.

Information Complete A requirement specification is (deveopment) *information complete* (mostly called or subsumed by the notion of *complete* in literature) if it contains all necessary information to allow the implementation and operation of the specified system over its complete life cycle.

Here, the desired system is defined with respect to the set of requirements contained in the specification, in contrast to *feature completeness* where it is defined according to the implicit or explicit customer needs. In literature, the attribute is bluntly described as the lack of missing (necessary) information [Wiegers, 1999, Berry et al., 2006, ISO/IEC 29148:2011, 2011, Kotonya and Sommerville, 1998, Rupp and SOPHISTen, 2007] or "parts of requirements" Ott [2012], if at all.

Metadata Complete Besides general information completeness, several additional attributes associated with the individual requirements are explicitly proposed in the literature. A requirement specification is *metadata complete* if every requirement is augmented with:

1. a priority reflecting its relative importance [Davis et al., 1993a, Wiegers, 1999, ISO/IEC 29148:2011, 2011, Rupp and SOPHISTen, 2007]
2. an indicator of its absolute importance in terms of legal liability [Rupp and SOPHISTen, 2007]
3. an indicator of its relative stability, i.e., which requirements are most likely to change [Davis et al., 1993a]

4. an unique identifier, which should never change nor be reused [ISO/IEC 29148:2011, 2011, Rupp and SOPHISTen, 2007]
5. an indicator of its risk which can be obtained by risk analysis techniques [ISO/IEC 29148:2011, 2011]
6. the source of the requirement, e.g., a particular stakeholder or a higher-level system specification [ISO/IEC 29148:2011, 2011, Davis et al., 1993a, IEEE Standard 830, 1998, Wiegers, 1999]
7. a rationale providing the reason why the requirement is needed [ISO/IEC 29148:2011, 2011]
8. an indicator of difficulty for implementing the requirement [ISO/IEC 29148:2011, 2011]
9. a type to describe the kind of property the requirement represents, e.g., functional or performance [ISO/IEC 29148:2011, 2011]
10. cross-references between requirements to designate redundant, more abstract or detailed, and dependent requirements [Davis et al., 1993a, ISO/IEC 29148:2011, 2011]

Total Behavior A requirement specification is *total* in terms of its behavior if all system responses to all realizable inputs in all realizable classes of situations are specified [IEEE Standard 830, 1998].

Note that this is a very strong demand and requires (i) a limited and manageable number of potential inputs and situations and (ii) an accurate understanding of the system under consideration.

Semantically Correct A requirement specification is *semantically correct* (also called *correct* [Berry et al., 2006, Wiegers, 1999, IEEE Standard 830, 1998, Ott, 2012], *necessary* [ISO/IEC 29148:2011, 2011, Wiegers, 1999]) if every requirement stated therein is one that the software shall meet [IEEE Standard 830, 1998, Wiegers, 1999, Berry et al., 2006] and is free of factual errors about the domain [Berry et al., 2006, Ott, 2012].

From a theoretical point of view, the reference for correctness is perfect domain knowledge $\mathcal{D}$ (see Sec. 2.2). In practice, requirements elude correctness, and hence the weaker but more tangible characteristic *validity* is used, which is defined with respect to the stakeholders' demands [Rupp and SOPHISTen, 2007] and higher-level requirements specification (also called external consistency).

Syntactically Correct A requirement specification is *syntactically correct* (also called *formally correct* [Ott, 2012], *syntactically valid* [Fabbrini et al., 1998]) if every statement in the requirement specification is correct regarding the (syntax of) language(s) used. For instance, no linguistic defects, e.g., spelling, punctuation or grammatical mistakes, are present in a natural-language specification [Ott, 2012].

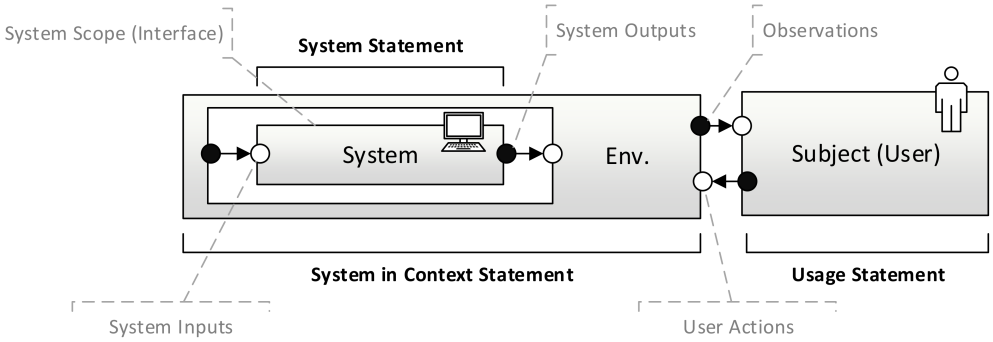


Figure 4.8.: Boundary-Precise: Different scopes of statements in requirements specifications based on Jackson [1995]

Semantically Consistent A requirement specification is *semantically consistent* (often simply referred to as (internally) *consistent* [ISO/IEC 29148:2011, 2011, Ott, 2012, IEEE Standard 830, 1998, Wiegers, 1999, Berry et al., 2006, Kotonya and Sommerville, 1998, Rupp and SOPHISTen, 2007] (or *not contradicting* [Meyer, 1985]) if no subset of statements in the requirements specification is contradictory, i.e., cannot be mutually fulfilled for any (feasible) system. Unlike [ISO/IEC 29148:2011, 2011, Ott, 2012], we do not consider redundant requirements as violating semantic consistency. Following the argument of [IEEE Standard 830, 1998], contradictions with higher-level requirements not part of the SRS are considered as part of the notion of semantic correctness rather than inconsistent.

Lexically Consistent A requirement specification is *lexically consistent* (often subsumed by *consistent* [ISO/IEC 29148:2011, 2011, Ott, 2012]) if the same term is used for the same item in all requirements [ISO/IEC 29148:2011, 2011].

Not Redundant A requirement specification is *not redundant* if the same requirement is not stated more than once [Davis et al., 1993a]) and no information is unnecessarily repeated [Kotonya and Sommerville, 1998, Meyer, 1985].

Boundary Precise A requirement specification is *boundary precise* (partly subsumed by *right level of detail* [Davis et al., 1993a], *bounded* [ISO/IEC 25010:2010, 2010]) if every prescriptive statement in the requirement specification is defined as an interface property of the software-intensice system. Here, an interface property specifies a capability or characteristic of the system observable at the boundary between the system and its context, and such observations may very well be over (extended, even infinite periods of) time. Furthermore, boundary precision is limited to prescriptive statements, i.e., actual requirements, while descriptive statements, e.g., environment/domain knowledge or assumptions on the user behavior, are explicitly excluded.

Essentially, this quality attribute is based on the crucial distinction between the system under consideration itself, called the *machine*, and its operational environment, called the *world*, with actors of the social-economical context being part of it. This distinction is illustrated in Fig. 4.8: The system (machine) is embedded in its operational context (world) with which the user interacts. We can now distinguish between three kind of statements: First, a system statement is an assertion about the system's inputs and outputs. In contrast, a system in context statement is an assertion about how the environment behaves, i.e., which observations result from which user actions. Finally, user actions are assertions about how the user behaves, i.e., the actions he triggers in the environment. Consider, for instance, the system under consideration is an instrument cluster control unit of a car responsible for computing and displaying the speed. An exemplary system statement is that, given a number of revolutions per minute as an input message, the unit outputs a corresponding speed based on a physical model. However, the statement that the actual speed of the car is shown to driver is a system in context statement. Neither the sensor measuring the revolutions per minute is part of the system itself, nor is it guaranteed that the physical model reflects reality under all circumstances (e.g., because of tire slippage). Finally, the user's interaction with the car, e.g., the use of the accelerator pedal and the perceived and displayed speed, are usage statements. Jackson [1995, 2001] advocates that RE includes the analysis of the interplay between achieving stakeholder goals in the system's environment and the role of the actual system therein, and argued that the requirements specification must specify system statements.

Concise A requirement specification is *concise* if for any subset or individual statement the concrete syntax with the highest density, e.g., the shortest formulation for natural language, is used. Hence, the specification is as short as possible without adversely affecting any other quality characteristic [Ott, 2012, Davis et al., 1993a]. This attribute clearly exceeds demanding that the requirements specification is of "reasonable size" Rupp and SOPHISTen [2007] and free of redundancy.

Traceable A requirement specification is *traceable* if (i) it is written in a manner that facilitates the referencing of each (individual) requirement [Rupp and SOPHISTen, 2007] and (ii) each a source is specified for every requirement [ISO/IEC 29148:2011, 2011, IEEE Standard 830, 1998, Davis et al., 1993a, Wiegers, 1999].

Unambiguous A requirement specification is *unambiguous* if every statement therein can be interpreted in only one way by its audience [ISO/IEC 29148:2011, 2011, IEEE Standard 830, 1998, Davis et al., 1993a, Ott, 2012, Wiegers, 1999, Fabbrini et al., 1998, Rupp and SOPHISTen, 2007]. For instance, no ambiguous words, phrases or sentences should be specified in a natural-language specification. Note that this notion of ambiguity is less restrictive than demanding the same but for any (arbitrary) reader [Meyer, 1985].

Agreed A requirement specification is *agreed* if every stakeholder is consent with every requirement [Pohl, 1994]. Agreement requires the understanding and thus communication

of the requirements among the stakeholders, and is not concerned about a common or consistent model of the system but rather about the personal views and sentiments in terms of acceptance or refusal.

Modifiable A requirement specification is *modifiable* if the requirements specification can be modified [Wiegers, 1999, Davis et al., 1993a, IEEE Standard 830, 1998, Rupp and SOPHISTen, 2007]. To this end, Wiegers [1999] argue that, among others, each requirement must be (i) uniquely labeled, (ii) expressed separately, (iii) grouped near related requirements and (iv) cross-referenced. Furthermore, Davis et al. [1993a], IEEE Standard 830 [1998] advocate to structure the specification and style such that any changes can be made easily, completely and consistently.

Understandable A requirement specification is *understandable* if any stakeholder, e.g., customers, user, project managers, developers, can easily comprehend the meaning of all requirements [Davis et al., 1993a, Berry et al., 2006, Fabbrini et al., 1998]. Kotonya and Sommerville [1998] argue that this is the most important quality attribute because it is required for validation.

Reusable A requirement specification is *reusable* if its parts, e.g., sentences, paragraphs and sections, can be easily adopted or adapted for use in a subsequent specification [Davis et al., 1993a].

Organized A requirement specification is *organized* if its contents are arranged in a sensible way so that readers can easily locate information and logical relationships among adjacent sections are apparent [Davis et al., 1993a, Kotonya and Sommerville, 1998]. This includes that no forward references, i.e., "use of features of the problem not defined until later in the text" [Meyer, 1985] are contained, and could be achieved by sorting the requirements according to sensible criteria [Rupp and SOPHISTen, 2007].

Electronically Stored A requirement specification is *electronically stored* if the entire specification is in a word processor [Davis et al., 1993a]. This is a prerequisite for allowing mutual access to and tool support for collaboratively working with the requirements [Rupp and SOPHISTen, 2007].

Executable A requirement specification is *executable* (also *interpretable*, *prototypable*) if it there exists a software tool capable of interpreting the SRS and providing a dynamic behavioral model [Davis et al., 1993a].

Formalized A requirement specification is *formalized* if every requirement therein is at least also formally represented with rigor semantics [Pohl, 1994].

Conformance to Standards A requirement specification *conforms to standards* if the requirements document and individual requirements conform to defined (company) standards [Kotonya and Sommerville, 1998].

Tab. 4.3 gives an overview of the consolidated quality attributes and their occurrence in the literature. The symbols ●, ◑ and ○ denote full, partial respectively no occurrence of the particular quality attributes in the cited publication.

4.4.2. Discarding Undesired Attributes

Out of the 27 consolidated attributes described in the previous section, twelve were rejected for the purpose of building an intrinsic quality model due to violating the following necessary conditions:

Limited to Intrinsic Quality Only attributes concerning the intrinsic quality of the requirements specification must be considered. Therefore, attributes referring to the degree certain activities are feasible are discarded. This led to the omission of seven attributes, namely *Verifiable*, *Traceable*, *Modifiable*, *Understandable*, *Reusable*, *Executable* and *Feasible*.

Universal Desirability Another prerequisite for quality attributes in our model is to be *desirable* in general. Here, desirable means that the attribute itself is considered beneficial for project success in a typical setting in practice. Four attributes are classified to lack universal desirability, and are thus discarded from the quality model:

1. *Total Behavior*: Recall that the system behavior is described totally in a SRS if and only if it defines all allowed outputs for any possible input. However, requirements may soundly constrain only a subset of potential inputs and outputs. In those cases, demanding the partial system behavior to be *complete* and *boundary-precise* instead of a total system behavior is considered more adequate in general.

2. *Not Redundant*: Redundancy, i.e. the multiple presence of the same information in the requirements specification, is a property of the specification which in itself is neither clearly desirable nor harmful in general. Even *effects* caused by redundant information according to literature can have positive, e.g., improving understandability, and negative impacts, e.g., causing inconsistencies and hampering modifications.

3. *Formalized*: Recall that formalization according to Pohl [1994] requires that every requirement is (i) formally represented and (ii) for which rigor semantics are well-defined. Although we agree on certain goals implied by (ii), e.g., *precise* and *unambigious* requirements, we disagree with (i) to consider a representation using formal notations as universally desirable.

4. *Conformance to Standards*: Standards are a means to achieve quality, but does not constitute a characteristic to be fulfilled per se. In particular, poor standards,

Attribute	Synonyms	ISO/IEC 29148:2011 [2011]	Wiegers [1999]	Davis et al. [1993a]	Ott [2012]	Berry et al. [2006]	Fabbrini et al. [1998]	IEEE Standard 830 [1998]	Pohl [1994]	Meyer [1985]	Kotonya and Sommerville [1998]	Rupp and SOPHISTen [2007]
Feasible	Achievable	●	◑	●	○	○	○	○	○	○	○	○
Verifiable	Testable	●	●	●	●	●	○	●	○	●	○	●
Singular	Atomic	●	○	○	●	○	○	○	○	○	○	○
Design-Independent	Implementation-Free, Bounded	●	○	●	●	○	○	○	○	●	○	○
Quantitatively Precise	Precise, (Complete)	◑	○	●	●	○	○	○	◑	○	○	○
Feature Complete	Complete	●	●	●	●	●	○	●	◑	●	●	●
Information Complete	(Complete)	●	●	◑	●	●	○	○	◑	○	●	○
Metadata Complete	Annotated, Prioritized, Attributed	◑	◑	◑	○	○	○	◑	○	○	○	◑
Total Behavior	(Complete)	○	○	●	○	○	○	●	○	○	○	○
Semantically Correct	Correct, Necessary, (Valid)	◑	●	●	◑	●	○	●	◑	○	○	◑
Syntactically Correct	Formally Correct	○	○	○	●	○	●	○	○	○	○	○
Semantically Consistent	Consistent	●	●	●	●	●	●	●	◑	●	●	●
Lexically Consistent	-	●	○	○	●	○	○	●	○	○	○	○
Boundary Precise	(Bounded)	◑	○	◑	○	○	○	○	○	○	○	○
Concise	-	○	○	●	●	○	●	○	○	●	○	◑
Tracable	-	●	●	●	○	○	○	●	○	○	●	●
Unambiguous	-	●	●	●	●	●	●	●	○	●	●	●
Agreed	-	○	○	○	○	○	○	○	●	○	○	○
Modifiable	-	○	●	●	○	○	○	●	○	○	○	●
Understandable	-	○	○	●	○	○	●	●	○	○	●	●
Reusable	-	○	○	●	○	○	○	○	○	○	○	○
Organized	-	○	○	●	○	○	○	○	○	◑	○	○
Executable	Interpretable	○	○	●	○	○	○	○	○	○	○	○
Electronically Stored	-	○	○	●	○	○	○	○	○	○	○	◑
Formalized	-	○	○	○	○	○	○	○	●	○	○	○
Conformance to Standards	-	○	○	○	○	○	○	○	○	○	●	○

Table 4.3.: Quality attributes in literature (●, ◑, ○: full, partial, no occurrence)

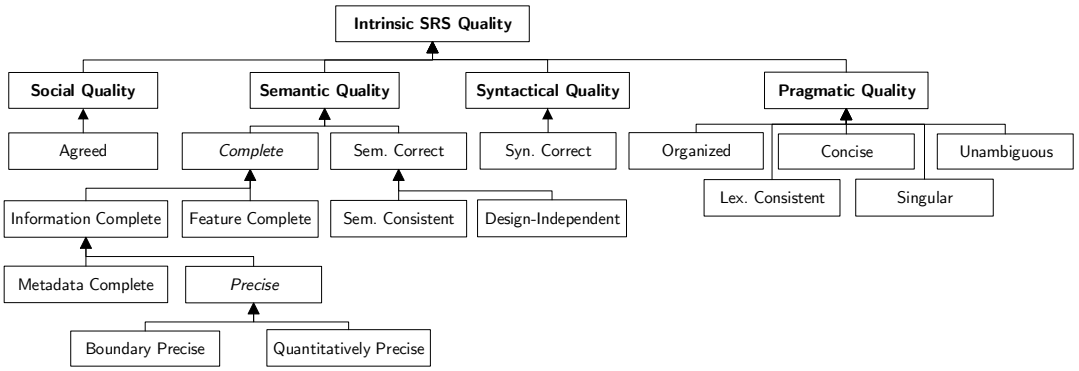


Figure 4.9.: Resulting taxonomy of intrinsic quality attributes based on the literature survey

or parts thereof, do neither necessarily imply improvement to any other intrinsic attribute nor are they beneficial with regards to quality-in-use [Femmer et al., 2015]. Therefore, we no not consider conformance as a quality attributes per se due to lack of universal desirability, but its content may still be subject to be considered for SRS quality.

No Universal Fulfillment Solely one attribute, namely *Electronically stored*, is fulfilled for any requirement specification in practice today, and can hence be omitted from the quality model.

4.4.3. Hierarchical Taxonomy of Intrinsic Quality Attributes

Last but not least, we identified specialization relations between the remaining attributes in order to obtain a hierarchically structured quality model. Therefore, we first classify the attributes into the quality notions defined according to the quality framework presented in Sec. 2.2, by means of argumentation on the set-theoretic foundations.

The majority of attributes were associated with semantic quality, i.e., the correspondence between the optimal system (denoted $\mathcal{D}$) and the system actually specified in the requirements specification (denoted $\mathcal{M}$). On the one hand, it contains attributes which refer to (certain) errors occurring in the specification, i.e., characterize the set $\mathcal{M} \setminus \mathcal{D}$, namely *Semantic Correctness*, *Semantic Consistency* and *Design-Independent*. On the other hand, it also contains attributes referring to certain mandatory statements missing in the requirements specification, i.e., characterize the set $\mathcal{D} \setminus \mathcal{M}$. This includes missing characteristics or capabilities of the system (*Feature Complete*) as well as mandatory information thereof (*Information complete*, *Meta-data complete*, *Boundary Precise* and *Quantitatively Precise*). The second most attributes were attributed to pragmatic quality, i.e., the correspondence between the specified model $\mathcal{M}$ and its interpretation by

the audience denoted $\mathcal{A}$. Since the requirements specification does not disclose results of internalization, intrinsic properties of the requirements specification considering pragmatic quality are inherently limited to means which potentially impact it. Therefore, we associate attributes with pragmatic quality if its purpose is to reduce the number of misinterpretations ($\mathcal{A} \setminus \mathcal{M}$) or omissions ($\mathcal{M} \setminus \mathcal{A}$). The attributes *Unambiguous*, *Lexically consistent*, *Concise*, *Singular* and *Organized* are all characteristics of the specification which serve this purpose. Finally, one attribute were associated with syntactic and social quality each: *Syntactic correctness* refers to the absence of syntactical errors ($\mathcal{M} \setminus \mathcal{L}$), and *Agreement* is absence of statements stakeholders disagree on ($\mathcal{M} \setminus \mathcal{S}$).

In addition to this classification, we also studied relations among the individual quality attributes. We argue that several attributes are specializations of other attributes: *Metadata Complete*, *Boundary Precise* and *Quantitatively Precise* are specializations of *Information Complete*, and *Semantically Correct* subsumes *Semantically Consistent* and *Design-Independent*. Furthermore, we introduce abstract attributes for the commonly used terms *Complete* and *Precise*. This way, we provide an interpretation in terms of their child attributes. The resulting taxonomy of intrinsic quality attributes is depicted in Fig. 4.9.

4.5. An Integrated Definition Model of Instrinsic Quality

In this section, we extend the taxonomy presented in the previous section with additional quality attributes which are obtained by applying our activity-based quality approach to the Unified process, as described in the second step of our research methodology in Sec. 4.3.3. As a result, we obtain a comprehensive definition model of intrinsic quality which can be used as a reference for further customization to a particular context of use (see Sec. 4.6.2) and provides the foundation for quality assessment in Part II.

4.5.1. Identified Quality Attributes

In total, the Unified Process specifies 33 elementary activities which directly rely on the requirements specification as an input, and which are carried our by seven unique roles (e.g., system analyst, system integrator, test engineer). Those elementary activities are grouped into five top-level activities, namely *requirements refinement*, *analysis*, *design*, *implementation* and *testing*. By analyzing the elementary activities and the resulting artifacts, we identified 53 unique intrinsic properties which impact the quality-in-use of those activities. Tbl. 4.4 shows the number of properties identified per impacted activity and quality-in-use. Note that the numbers do not add up in general since an intrinsic property may impact multiple activities and quality-in-characteristics.

Those 53 low-level intrinsic properties were associated with the taxonomy's 15 high-level attributes 122 times. On average, an attribute from the taxonomy was substantiated by 8.1 low-level properties as identified from the Unified Process. Obviously, several low-level attributes were associated with more than one high-level attribute. This was most common for attributes identified by ABRE-QM which stated that some particular

	Identified Properties (#)			
Activity	Efficiency	Effectiveness	Both	Any
Req. Refinement	9	16	7	28
Analysis	4	16	2	20
Design	2	11	5	13
Implementation	2	3	2	7
Test	2	7	3	11
All	18	33	16	53

Table 4.4.: Number of intrinsic quality attributes identified per impacted activity and quality-in-use

information about the system under development must be contained in the requirements specification. Since this information must be correct and unambiguous, it essentially substantiates both attributes and was classified accordingly. However, our analysis did not reveal low-level intrinsic properties for every attribute of the taxonomy; we found no activities or artifacts affected by the extent stakeholders agree on the requirements nor that the requirements are atomic, and we discuss this observation in the next section.

In the following, we present the identified properties organized according to our classification regarding the high-level attributes of the taxonomy (see Sec. 4.4) and in a condensed form. For the precise impact on quality-in-use, in terms of the particular activity and affected characteristic (i.e., efficiency and/or effectiveness), and the underlying rationale refer to appendix C using the IDs specified below.

Taxonomy Attribute	**ID**	**Associated attributes** (identified by ABRE-QM on UP):
	1	Shared sections in use cases explicitly specified
	5	Priorities for requirements specified
	7	Subsystem decomposition specified
	9	All business processes to be supported by the system are specified
	10	All actors using or depending on the system are specified
	15	Common Business Entities specified
Semantically	25	Detailed data model, specifying all relevant attributes and non-technical type
Correct	26	All human actors interacting with system specified
	27	All external systems interacting with system specified
	30	Source of requirements provided
	31	All actions and flows specified
	33	Quality requirements specified at function granularity
	34	Quality requirements specified quantitatively precise
	35	Quality requirements are valid and unambiguous

Taxonomy Attribute	**ID**	**Associated attributes** (identified by ABRE-QM on UP):
Semantically Correct (cont'd)	89	Value ranges and required precision specified in data model
	37	All necessary features and characteristics specified
	38	Criticality of requirements specified
	40	All system goals specified
	46	Target system environment specified
	47	Usage scenarios specified (including criticality, frequency and parallel tasks)
	48	Inputs/Outputs specified quantitatively
	49	Inputs/Outputs specified at system boundary
	50	Stability of requirements specified
	60	Individual users aggregated to roles
	65	Business objects to be modified by the system specified
	69	Inclusion and extension relationships between business activities specified
	75	All end users are specified
	80	Use-case is mapped to business process activity
Metadata Complete	1	Shared sections in use cases explicitly specified
	5	Priorities for requirements specified
	6	Fine-grained prioritization
	9	The business process associated with activities and objects is specified
	30	Source of requirements provided
	38	Criticality of requirements specified
	50	Stability of requirements specified
	69	Inclusion and extension relationships between business activities specified
	74	End users are described in detail
	80	Use-case is mapped to business process activity
Feature Complete	31	All actions and flows specified
	37	All necessary features and characteristics specified
Information Complete	7	Subsystem decomposition specified
	9	All business processes to be supported by the system are specified
	10	All actors using or depending on the system are specified
	15	Common Business Entities specified
	25	Detailed data model, specifying all relevant attributes and non-technical type
	26	All human actors interacting with system specified
	27	All external systems interacting with system specified
	89	Value ranges and required precision specified extensively
	40	All system goals specified
	46	Target system environment specified
	47	Usage scenarios specified (including criticality, frequency and parallel tasks)
	55	For each actor, at least one user who can enact it is specified
	56	Relationship between stakeholders specified
	60	Individual users aggregated to roles

Taxonomy Attribute	**ID**	**Associated attributes** (identified by ABRE-QM on UP):
	64	Detailed description of business processes supported by the system
	65	Business objects to be modified by the system specified
	75	All end users are specified
Quantitatively Precise	34	Quality requirements specified quantitatively precise
	89	Value ranges and required precision specified in data model
	48	Inputs/Outputs specified quantitatively
Boundary Precise	4	Requirements specified at system boundary scope
	33	Quality requirements specified at function granularity
	49	Inputs/Outputs specified at system boundary
Design-Independent	32	Nodes and network configurations predefined
	36	Subsystems (e.g., COTS) predefined
	79	(Parts of) UI specified
Syntactically Correct	63	Stakeholder names syntactically correct and unambiguous
	66	Activity names syntactically correct and unambiguous
	67	Activities start with verb and are named after what shall be achieved
Unambiguous	1	Shared sections in use cases explicitly specified
	7	Subsystem decomposition specified
	9	All business processes to be supported by the system are specified
	10	All actors using or depending on the system are specified
	15	Common Business Entities specified
	25	Detailed data model, specifying all relevant attributes and non-technical type
	26	All human actors interacting with system specified
	27	All external systems interacting with system specified
	30	Source of requirements provided
	31	All actions and flows specified
	33	Quality requirements specified at function granularity
	34	Quality requirements specified quantitatively precise
	35	Quality requirements are valid and unambiguous
	89	Value ranges and required precision specified extensively
	37	All necessary features and characteristics specified
	38	Criticality of requirements specified
	40	All system goals specified
	46	Target system environment specified
	47	Usage scenarios specified (including criticality, frequency and parallel tasks)
	48	Inputs/Outputs specified quantitatively
	49	Inputs/Outputs specified at system boundary
	50	Stability of requirements specified
	60	Individual users aggregated to roles
	63	Stakeholder names syntactically correct and unambiguous

Taxonomy Attribute	**ID**	**Associated attributes** (identified by ABRE-QM on UP):
Unambiguous (cont'd)	65	Business objects to be modified by the system specified
	66	Activity names syntactically correct and unambiguous
	67	Activities start with verb and are named after what shall be achieved
	69	Inclusion and extension relationships between business activities specified
	70	Terms used according to glossary or standard
	75	All end users are specified
	80	Use-case is mapped to business process activity
Organized	1	Explicit specification of relationships between (parts of) use-case flows
	2	Provides navigation means to locate features and characteristics
	11	Matching level of abstraction in business process and use-case, or explicit links are specified
	18	Structured description of use-case flows
	20	Consistent level of abstraction of use-cases
	53	Structured description of stakeholders operating with the system
	57	Provides navigation means to locate stakeholders
	58	Stakeholders are documented in one place
	67	Activities start with verb and are named after what shall be achieved
	69	Inclusion and extension relationships between business activities specified
	80	Use-case is mapped to business process activity
	81	All use-cases follow a consistent structure
Lexically Consistent	13	Use Case names, actors, actions and inputs/outputs are lexically consistent to each other
	14	Use-case names, actors, actions and inputs/outputs are lexically consistent with terms in the business model
	24	Objects referred to in use cases are lexically consistent to the data model
	62	Stakeholder names are lexically consistent
	70	Terms used according to glossary or standard
Concise	25	Detailed data model, specifying all relevant attributes and non-technical type
	52	Only prescriptive statements
	64	Detailed description of business processes supported by the system
	74	End users are described in detail

4.5.2. Impact of Quality Attributes

Since the intrinsic properties identified in the previous section (i) document the impact on specific development activities, and (ii) are classified as specializations of the intrinsic attributes of the taxonomy, the resulting integrated model provides a comprehensive view on the impact of intrinsic quality on quality-in-use, which is the subject of discussion in this section.

Impact of individual attributes Overall, we found that 62% and 34% of the properties impact the effectiveness respectively efficiency of downstream development activities, with 30% actually impacting both. On average, each property impacted 2.23 activities. A *consistent vocabulary in use-cases* (ID 13, 6 impacted activities), ensuring that *all actions and flows are specified* (ID 31, 6 impacted activities) and that *quantitative specifications of quality requirements* (ID 34, 5 impacted activities) were identified as the attributes which (directly) influence the most downstream development activities. While we aimed to express the identified properties in a positive way, 4 intrinsic properties also imply negative effects. For instance, if *nodes and network configurations are predefined* (ID 32) in the specification, less time is required to design the architecture but the resulting deployment model might be inferior (compared to if the requirements specification would not constrain those design decisions).

Aggregated view: Impact of high-level attributes An aggregated view on the impact of intrinsic quality on quality-in-use is illustrated in Fig. 4.10. Specifically, it shows the intrinsic quality attributes from the taxonomy, clustered and colored according to the semantic, syntactic and pragmatic quality, which impact the (high-level) development activities of the Unified Process. Furthermore, the relationship between intrinsic and quality-in-use is quantified by the number of (low-level) attributes which impact the respective (sub-)activities, as also visualized by the width of the arrows. As shown in the figure, our analysis confirmed an impact on efficiency and/or effectiveness on downstream development activities for all quality attributes of the taxonomy except for *Agreed* and *Singular* which are missing in the figure. Hence, we conclude that those intrinsic properties indeed are quality attributes, provided that the associated (low-level) intrinsic properties identified using the ABRE-QM approach are valid in the actual context of use. However, we refrain from dismissing agreement and singularity as unnecessary, and therefore invalid, just yet, due to the limitations of both our analysis method and the abstractions chosen in the quality model. In particular, we relied on one process model, the identification was subject to researcher bias, and the model's disregard for inter-attribute relations. Instead, we would like to encourage future work in this direction as outlined in the limitations discussed in Sec. 4.6.1 and the outlook (Sec. 8.2). Properties pertaining to semantic correctness and unambiguity collectively impacted the most activities. This is explained by the large number of attributes which demand that a certain information must be specified correctly and unambiguously which were identified. Mind that this quantification is not a reliable indicator for importance, since the quality-in-use effects are not quantified with respect to contribution to project success. For instance, even if feature completeness is substantiated by only a few number of intrinsic qualities, their consequences might still be severe. Last but not least, the figure also visualizes the diminishing direct influence of requirements specification quality for activities up to implementation, and the subsequent (although minor) increase for testing, essentially documenting the tight coupling between RE on the one hand and early

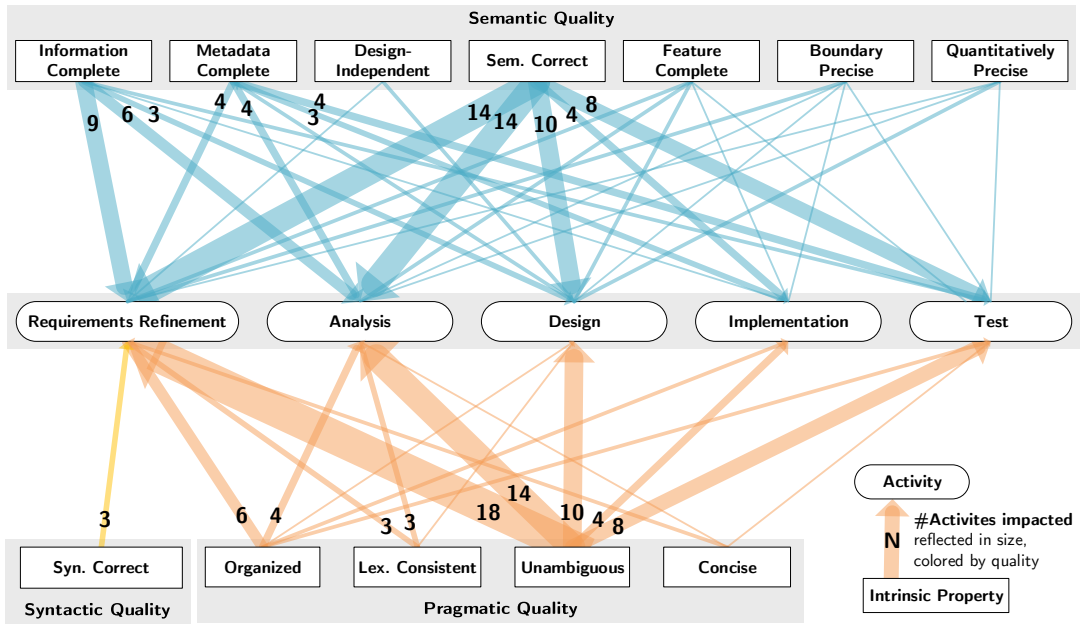


Figure 4.10.: Impact of (high-level) quality attributes on development activities as revealed by ABRE-QM, quantified in terms of the number of low-level attributes identified

and late development activities on the other hand[4].

4.6. Limitations and Applications

In this section, we discuss limitations of our quality definition model and outline potential uses of it, including its customization to obtain a context-specific quality model.

4.6.1. Model Limitations

Limitations due to Abstraction and Scoping We recognize three main limitations in the abstractions chosen for the quality model. First, we solely studied specialization/-generalization relationships among the quality attributes while neglecting other forms. For instance, the model does not reflect impacts (e.g., syntactic correctness might reduce unambiguity and lexical consistency), conflicts (e.g., between conciseness and information completeness), or correlations (e.g., requirements the stakeholders agree on might be more likely to prove correct with respect to the optimal system) between individual

[4]This resembles the V-shape of the corresponding classic software process model [Forsberg and Mooz, 1991]

qualities. Since the need to clearly understand also those types of inter-dependencies between the different criteria is well recognized [Broy, 2006], we would encourage future work to study it in more detail as outlined in the outlook (Sec. 8.2).

Secondly, our model does not consider RE process quality. On the one hand, we are thus able to provide a RE process-agnostic quality definition, but on the other hand, we lose the capability to adjust the process accordingly. While this might be acceptable for analytical quality assurance, constructive means must tackle the processes determining specification quality.

Last but not least, we studied only those intrinsic properties influencing *software development*. However, the requirements specification is also input to activities beyond the vast engineering of the system, such as project management, software usage, operation and maintenance, and detachment, as well as activities which are concerned with the software-intensive system as a whole (e.g., hazard analysis [Ericson et al., 2015]). Again, the interested reader should consult the outlook (Sec. 8.2) in which we outline future research in this direction.

Threats to Validity We recognize that the internal validity of the quality model is subject to researcher bias due to the extensive use of manual identification and classification tasks. In particular, the identification of intrinsic properties for the Unified Process relied on both the researcher's understanding of the analyzed documentation and his expertise with regards to the how downstream development activities are performed and artifacts created in detail. To mitigate this threat, we rigorously documented the rationale for manual classifications or the identified properties. Any rationale was then verified by a second researcher, and in case of disagreement, discussed and resolved, which mitigates the threat of invalid attributes in the model.

Regarding external validity, we regard the reliance on the activities described in the Unified Process as the main threat. While we consider the activities and tasks described therein to apply to many software development projects, we still suppose differences in the detailed tasks compared to, for instance, agile development (e.g., SCRUM or eXtreme programming (XP)) or process models for specific domains, such as Automotive SPICE [Automotive SIG [2015]]. To accommodate intrinsic properties influencing activities specific to a domain or process model, we envision to parametrize the quality model with certain characteristics about the context of use.

4.6.2. Model Applications

Context-Specific Customization (Tailoring) We see two alternatives for establishing a context-specific quality definition. For one thing, one can apply the ABRE-QM approach to a particular context, if known, and for another, start from the generic quality definition model proposed in this section and select and refine the intrinsic properties applicable to the context. Here, we briefly outline the steps and challenges involved in the latter from the perspective of a quality manager:

1. **Context Identification:** First, the extent of documentation and the context of use for the requirements specification must be identified. Therefore, one has to identify the activities to be performed based on the specification, the audience performing those, and the semantic content that is documented, i.e., the artifacts and entities.

2. **Selection** Then, for every activity in the generic quality model, the relevance of the detailed intrinsic properties (described in appendix C) must be decided for the specific context. For instance, the conciseness property that *only prescriptive statements* (ID 52) are specified for requirements might be suitable for an in-house project associated with low risk and high expertise, while the property would probably not apply to out-sourced development with a new contractor.

3. **Refinement** Finally, the properties classified as relevant in step 2 are subject to refinement. Here, refinement refers to adapting the properties to convey a more precise and adequate meaning for the particular context. Consider, for instance, the property that *navigation means to locate features and characteristics are provided*(ID 2). Knowing the tooling landscape the audience relies on, one could refine this property to the particular means available, e.g., by demanding that hyperlinks have to be provided. Another example would be *terms used according to glossary or standard* (ID 70). Here, one would provide the exact standards which should apply, e.g., Automotive SPICE [Automotive SIG [2015]].

We see the main challenge in the uncertainty of the concrete context, and the associated imprecision in selecting and refining the criteria. Unfortunately, both a too conservative as well as a too permissive selection are harmful; while the former would result in unnecessary overhead, the latter would result in insufficient quality assurance, with both cases therefore resulting in a reduced acceptance of the approach.

Foundation for Quality Assurance Since explicitly developed with this goal in mind, we see the primary application of the quality definition model as a foundation for quality assurance. More specifically, due to the fact that it is strictly based on the influence on quality-in-use, establishing quality assurance measures on it should contribute to the measures' effectiveness. Although developed for the particular case of measurement-based quality assessment, its applicability and the claimed benefits should span to general quality assurance. Regarding constructive quality assurance on the one hand, the model can be used to improve guidelines or templates. In Femmer et al. [2015], we have shown that guidelines could be improved by identifying missing items based on neglected intrinsic properties of the quality definition model and by validating the existing ones by challenging its impact on quality-in-use. Given that many intrinsic properties deal with content expected in the specification and its organization, we suppose similar improvements can apply to templates. Regarding analytical quality assurance, the model will be extended to a quality assessment model based on measurements in Part II of this thesis. Moreover, checklists are a pragmatic quality assurance tool popular in practice, for which basically the same arguments apply.

4.7. Related Work

As seen in the literature survey, quality of requirements specifications is a vivid field of (RE) research and various quality definitions have been proposed. Besides the widely-known standards, scientific literature and text-books defining quality in terms of a set of attributes demanded of a requirements specification, e.g., IEEE Standard 830 [1998], ISO/IEC 29148:2011 [2011] (see Fig. 4.3 (p. 85) for an overview of proposed attributes), several models question requirements specification quality in a more fundamental way. Lindland et al. [1994] and Krogstie et al. [1998, 1995b, 2002] model RE quality based on semiotic theory. Syntactic quality is concerned with the absence of errors regarding the languages used, semantic quality with the completeness and correctness (or, validity), and pragmatic quality with the degree to which a specification is understood by its audience. Pohl [1994] models RE quality along three fundamental dimensions, namely specification (degree of completeness), representation (degree of formalization), and agreement (degree to which a common view was obtained). Furthermore, depending on the representation used, specific quality models are introduced, e.g., for natural-language specifications, Berry et al. [2006] relate manifestations in terms of linguistic defects to understandability, consistency, completeness, and correctness. All these models have in common to focus on intrinsic properties of artifacts rather than on their usage for the engineering endeavor.

In contrast, the basic idea of activity-based quality models is to define quality by how well properties of a product support the activities carried out by the use of the product. Those models, as introduced by Wagner et al. [2012], were originally intended to enable the precise definition and evaluation of code quality aspects with regards to maintainability [Deissenboeck et al., 2007] and other non-functional requirements [Lochmann, 2010, Wagner et al., 2008]. In [Femmer et al., 2015], we transferred the basic concepts of the activity-based quality model to the domain of RE, more specifically, the requirements specification, to provide an approach for obtaining a profound and precise notion of requirements specification quality, as described in Sec. 4.2. This paper also contains an evaluation of the approach in terms of an industrial case study.

The notion of quality of RE in general is even more widely discussed. Several studies (e.g., Méndez Fernández et al. [2015, 2014], Verner et al. [2005], Damian and Zowghi [2003]) identify individual problems and success factors which occur in or are closely associated with RE, such as incomplete requirements or human factors. El Emam and Madhavji [1995a] define RE quality in terms of RE product quality, quality of RE service, and cost effectiveness of RE process. The authors relied on survey research (the survey instrument is described separately in El Emam and Madhavji [1996]), identified 33 quality attributes in total, and associated a weight indicating their importance. For RE product quality, the authors identified attributes such as *clarity of the business process* or *clarity of links between the (process and data model) and the system objectives* as the most important factors. Compared to our quality model, we claim to essentially provide a more comprehensive and precise definition of RE product quality. For instance, the former attribute corresponds to the attributes with IDs $9, 64, 66, 67, 69, 80$ and the latter to $11, 14, 24, 64, 80$ (e.g., that use-cases are mapped to business process activity, the levels

of abstractions of both match, explicit links are provided, and objects referred to in use-cases are lexically consistent to the data model) of the quality definition model proposed in this thesis.

Contributions to the State-of-the-Art The quality model presented in this chapter substantially extends the current understanding of the attributes which constitute requirements specification quality. First, instead of only providing an intensional definition for individual attributes such as *complete* or *organized*, our quality model also provides an extensional definition by means of more narrow and therefore tangible sub-properties, resulting in a substantially more precise quality definition. Furthermore, to the best of our knowledge, we are the first to adapt and apply the activity-based approach to define the quality of requirements specifications, in our case, based on the activities of the Unified Process. Thereby, we are able to provide explicit implications of requirements specification quality on downstream development which are implicit and sometimes questionable in current quality definitions. As a result, we claim that the quality model presented here is also more well-founded.

4.8. Summary

In this chapter, we provided two approaches for defining the quality of requirements specifications. First, we proposed the ABRE-QM approach, which applies the idea of activity-based quality to requirements engineering. More specifically, the approach consists of a meta-model and a complementary systematic procedure to identify intrinsic properties of the requirements specification which impact the quality-in-use, more specifically, the effectiveness and efficiency, of downstream development activities.

Second, we provide a quality definition model in terms of intrinsic quality attributes, i.e., properties inherently associated with the requirements specification itself. In total, our quality definition model consists of 72 unique quality attributes on various levels of abstraction. This set of attributes is comprehensive and consistent since it is based on a systematic study of quality models proposed in the current body of knowledge, and organized in terms of a hierarchical taxonomy based on identified specialization relationships among attributes. Also, high-level attributes are substantiated both in terms of the contents it applies to as well as the aspects which are considered. Moreover, the quality attributes are profound since their impact on downstream development is made explicit.

Consequently, we see two important applications of this quality definition model. On the one hand, it can be used to obtain a valid quality definition for a specific context of use by means of a customization procedure, which is essentially based on a careful selection and refinement of the model's quality attributes. On the other hand, it establishes a reference definition for quality assessment as considered in Part II of this thesis.

Part II.

Assessing Quality

Chapter 5

A Framework for Quality Assessment of Requirements Specifications

While in the first part of this thesis, we provided a definition of what constitutes the quality of a requirements specification, the second part is concerned with how to *assess* it. By quality assessment, we refer to a systematic approach which provides a justified estimation about the extent to which a requirements specification under consideration possesses the desired quality attributes. Specifically in this thesis, we study the use of measurement, i.e., the assignment of numbers to empirical observations according to certain rules [Fenton and Pfleeger, 1998], as the instrument for assessment.

However, there currently does not exist a comprehensive and systematic approach for measurement-based quality assessment of requirements specifications. In particular, we lack a structured approach to holistically specify measures targeting requirements quality. Furthermore, we need a systematic methodology to identify those measures which are relevant, applicable and valid for a given requirements specification and context, and which endorses their careful and sound application while considering the peculiarities of requirements engineering.

To overcome this limitation, we provide a framework for quality assessment of requirements specifications. Essentially, this framework consists of a meta-model to specify quality assessment models and a complementary method to apply such a model to determine the quality of a concrete requirements specification. Specifically, we contribute a meta-model which is holistic in the sense that it encompasses and relates the necessary concepts associated with measures, and a process model covering planning, assessment and continuous improvement when implementing such an assessment model, both devel-

oped specifically with requirements specification quality and its inherent uncertainty in mind. Furthermore, the meta-model and process model provide the basis for specifying respectively applying the concrete assessment model presented in Chapter 6.

This chapter is structured as follows. In Sec. 5.1, we introduce the quality assessment meta-model. Therein, we first provide an overview of the meta-model before presenting its details and an exemplary instantiation of this model in the following subsections. Furthermore, we present an evaluation of the practical necessity of the concepts represented in the meta-model by means of interviews with an industrial partner. Next, in Sec. 5.2, we propose a complementary process model which suggests a systematic methodology to assess the quality of requirements specifications. Finally, we discuss related work in Sec. 5.3 and summarize our results in Sec. 5.4, .

5.1. A Meta-Model for Measurement-based Quality Assessment

In this section, we provide a detailed quality assessment meta-model. Instances of this meta-model, i.e., quality assessment models, provide holistic specifications of measures for quality attributes in the sense that it encompasses and relates the concepts associated with measures which are necessary to support a goal-oriented and sound application of measurement for quality assessment in practice.

Essentially, the quality assessment meta-model extends the meta-model already presented for the quality definition model (see Sec. 4.5, p. 87) by introducing the concept of *measures*. The primary purpose of measures is to assess a requirements specification's intrinsic quality, i.e., provide a justified estimation of the extent to which it possesses certain quality attributes. Moreover, since intrinsic quality attributes in turn impact the quality-in-use of downstream development activities (see Sec. 4.5.2, p. 91), measures also *predict* (the impact on) the quality-in-use of those activities. This relationship is illustrated in Fig. 5.1. Subsequently, we provide a more detailed meta-model which advocates a comprehensive and structured definition of measurement-based assessment models.

Notation The meta-model is expressed in terms of a UML class diagram. For establishing a holistic understanding, it also contains coherent concepts which are not part of a quality assessment model but rather refer to the results of its application for a specific requirements specification, i.e., (individual) assessments. We designate such concepts by annotating the relevant UML classes with the $\ll result \gg$ stereotype. Furthermore, for the sake of readability, we refrain from further modeling of the introduced concepts in terms of class attributes or operations but instead describe those informally.

Overview of the Detailed Meta-Model The detailed meta-model is depicted in Fig. 5.2. The fact that an assessment model depends on two other models, namely a quality definition model and an artifact model of the requirements specifications under assessment, is reflected in the meta-model classes *Quality Attribute* and *Content Items*,

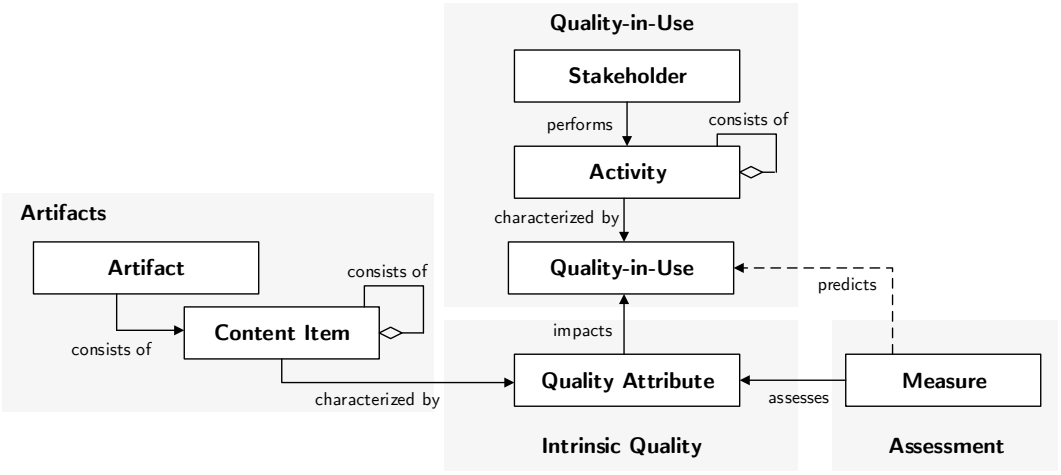


Figure 5.1.: Extended meta-model showing the measures primary and secondary purpose, namely to assess intrinsic quality and predict quality-in-use, respectively.

respectively. Regarding the former, we rely on the quality definition model presented in Chap. 4, while regarding the latter, we refer to the reference model provided by Méndez Fernández and Penzenstadler [2014] (see Fig. 2.1, p. 21). The remaining classes constitute the necessary concepts of individual measures and must be provided by a measurement-based assessment model. We further classify these concepts according to the following topics:

- *Quality Mapping:* Concepts associated with relating the assessed quality attribute to the attributes which are actually measured during data collection.
- *Data Collection:* Concepts associated with acquiring the measured value by executing a measurement procedure.
- *Data Analysis:* Concepts associated with interpreting the measured value to support decision making by means of an interpretation procedure. This interpretation procedure yields an assessment about the sufficiency of the requirements specification's quality at hand, and recommendations for action to improve it.
- *Threats to Validity:* Anticipated circumstances associated with concepts from data collection or analysis which may lead to flawed assessments.

Subsequently, for each topic, we motivate its necessity and describe the assigned concepts and its relationships in more detail.

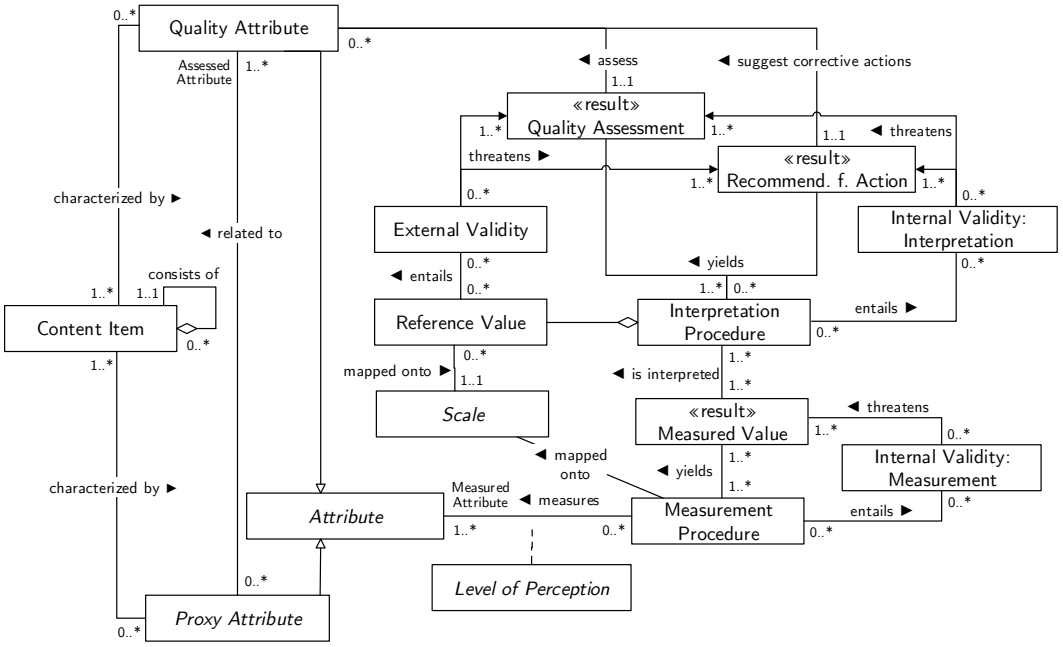


Figure 5.2.: Detailed quality assessment meta-model.

5.1.1. Relationship between Quality Definition and its Measurement

Measurement in software engineering distinguishes between direct measures (e.g., *lines-of-code (LOC)* as a size measure for programs), and indirect measures (e.g., *function points* as a measure of complexity). While the former indeed considers empirical observations of the quality attribute purport to measure, the latter is concerned with a different attribute, called proxy or surrogate attribute, which is related (e.g., by a mathematical formula) to the original attribute [Fenton and Pfleeger, 1998, Kitchenham et al., 1995].

Since many quality attributes of requirements specifications are defined with respect to external points of reference which elude direct measurement, e.g., perfect domain understanding in case of semantic quality or the audience's understanding in case of pragmatic quality, measures for requirements specification quality rely extensively on such proxy attributes. Consequently, a quality assessment model must specify the attributes actually measured in addition to the quality attribute to be assessed, and must provide a detailed specification of its relationship to allow careful and sound interpretation of measurement results. In the meta-model, this aspect is represented by the classes *Quality Attribute* and *Proxy Attribute* which are both specializations of the abstract *Attribute* class, as illustrated in the left part of Fig. 5.2.

Quality and Proxy Attributes The meta-model enforces an explicit distinction between direct measurement of a quality attribute and indirect measurement based on proxy attributes. A quality attribute is part of a quality definition model (such as the one presented in Cha. 4). Hence, it must fulfill the conditions required of a valid quality definition, in particular, impact the quality-in-use of dependent activities. Furthermore, a quality assessment model may provide proxy attributes which are related to the quality attribute and measurable, i.e., which are studied according to the measurement procedure. We may further specialize proxy attributes according to its relationship to the associated quality attribute. For instance, in this thesis, we distinguish between three kinds of proxy attributes, namely reasons, consequences or correlated attributes, based on the relationship's causality. While the first and second type suppose a causal relationship between the proxy and the assessed attribute, the third does only specify a general correlation relationship[1].

For any kind of attribute, the artifact, more precisely, the content items to which it applies, must be specified. Note that according to our model, the quality attribute under assessment and the measured attribute must not necessarily correspond to the same entity. For example, Cailliau and van Lamsweerde [2012] propose to assess the correctness of requirements by measuring certain properties of the goal model which the requirements are based on.

Relationship between Quality and Proxy Attributes For any proxy attribute contained in an assessment model, the relationship to each associated quality attribute should be described as precisely as possible to prohibit misinterpretations of the measurement results. While a thorough characterization is beyond the scope of this thesis, we outline the aspects of and views on the relationship between quality and proxy attributes we perceive as important for a quality assessment model:

- *Causality:* Are the attributes in a causal relationship? If not, then it's a pure correlation. If yes, it can be a reason or a consequence depending on the direction of causality. This question is important in order to understand the measure's actionability, i.e., whether improving the requirements specification with regards to the proxy attribute implies an improvement in quality.
- *Quantifiability:* In its current understanding, is the relationship between the proxy and quality attribute quantifiable? Or is it qualitative in the sense that no quantitative value can be reasonably associated with the relationship.
- *Certainty (Randomness):* In the current understanding, is the relationship characterized by a deterministic, stochastic (probabilistic), or non-deterministic model?
- *Evidence:* How is the validity of the relationship documented? Is an argument provided or does empirical evidence exist?

Providing this information enables the quality manager to critically and soundly interpret the measures' results.

[1] See Chap. 6, in particular, Fig. 6.1 (p. 132), for more details on their distinction.

5.1.2. Data Collection

Data collection is concerned with acquiring measurement results by executing certain measurement procedures which may be automated, semi-automated or manual. The central classes are the *measurement procedure*, *scale*, and *measured value*, which are illustrated in the bottom-right part of Fig. 5.2.

Measurement Procedure The measurement procedure is the activity of mapping empirical observations on the requirements specification, more precisely, its content items, onto an associated scale. Depending on the measured attribute, the empirical observations are perceived at the level of the requirements specification's physical representation, its syntactic structure, or its semantic content. A measurement procedure may indeed measure multiple elementary attributes which it combines into a single measurement value, a so-called aggregation or compound measure. Also, an assessment model may provide several different measurement procedures for the same attribute, e.g., by employing different techniques. However, the minimal multiplicity of 0 allows to explicitly specify (quality) attributes which are immeasurable by not associating any measurement procedure with it. A well-defined description of a measurement procedure must precisely state the steps performed on the requirements specification regarding the stimuli applied to it and the assignment of values to the observations made on it.

Scale The scale refers to the range of the measurement procedure, i.e., it specifies the values of the formal (mathematical) world that the empirical observations can be mapped to, called *levels*, as well as the relationships that hold between those levels. Instances of scales are commonly referred to as *units* [Kitchenham et al., 1995]. A concrete scale determines the mathematical operations that are *admissible* to the measured values, i.e., operations in the formal world which resemble the expected observations in the empirical world with regards to the attribute under consideration. Fenton and Pfleeger [1998] (pp. 45) distinguish between the following types of scales:

- *Nominal Scale:* A nominal scale does not assume any relationship between its levels, i.e., it defines merely categories. Consequently, two values on a nominal scale can only be compared for equality. For example, the shape of text structure (appendix B.120, p. 349) measure proposed by Wilson et al. [1997] classifies the document's shape to resemble a pyramid, hour-glass, or diamond, based on the amount of text provided for each nesting depth of a hierarchically organized requirements specification.

- *Ordinal Scale:* An ordinal scale requires an order relation, or ranking, between its levels. For instance, the complexity of the specification may be mapped on a scale with levels *low*, *medium*, and *high*, where the levels are ordered with respect to increasing complexity, i.e., $low \leq medium$, $medium \leq high$, and $low \leq high$.

- *Interval Scale:* In addition to ordinal scale, interval scales also carry information about the size of intervals that separate its levels. However, the ratio between two

intervals is not defined in a meaningful way since an interval scale lacks a non-arbitrary reference (zero) element. The celcius and fahrenheit scales are examplary interval scales for temperature, while kelvin is a ratio scale (see below).

- *Ratio Scale:* Ratio scales differ from interval scale in having a zero element. This zero element is the common reference for any two values on this scale, therefore rendering the ratio between values admissible. An exemplary measure for ambiguity mapped on a ratio scale is suggested by Davis et al. [1993a]. Suppose that 5% respectively 10% of the requirements contained in two different specifications were interpreted inconsistently by reviewers (see appendix B.4). Because we have an understanding of what value denotes no ambiguity (i.e., 0), it is admissible to say that the latter specification is twice as ambiguous as the former.

Result: Measured Value The measured value is the result obtained from the measurement procedure, and is mapped onto the associated scale. Since it is a concrete result of a measurement, it is not part of the quality assessment model itself but obtained when the model is applied to a particular requirements specification.

5.1.3. Data Analysis: Quality Assessment and Improvement

While data collection yields a formal (mathematical) value on a well-defined scale, researchers (e.g., Niessink and Van Vliet [1999], Jacquet and Abran [1997], Boegh et al. [1999], Frederiksen and Mathiassen [2005]) independently recognized that an assessment model based on measurements is inadequate for practice without providing an interpretation of the measurement results regarding the targeted quality attribute. Here, by interpretation we refer to a result of quality assessment which is suitable for decision-making (e.g., if corrective actions is required), and recommendations for action to improve its quality. In general, obtaining such an interpretation is not trivial due to the following reasons:

- *Discrepancy between measured and assessed attribute*: If the attribute measured by the measurement procedure is different from the one to be assessed, the quality manager must know the precise relationship between those attributes and how to cope with its uncertainty. For instance, the extent to which stakeholders agree with a requirements specification may be an indicator of its semantic correctness. A sound interpretation of an agreement measure regarding correctness therefore requires to understand:
  - the conditions under which the measured attribute adequately reflects the attribute to be assessed,
  - how to check if those conditions are fulfilled for a particular requirements specification and context, and
  - if the effect of improving correctness can be achieved by improving agreement among stakeholders.

- *Lack of widely accepted reference scales and values*: As already discussed in the first part of this thesis, the current understanding of requirements specification quality is still immature. In particular, we still lack reference scales and target values which are widely accepted as adequately formalizing the possible levels of magnitude of quality attributes respectively the demanded magnitude for a specific requirements specification. While the software engineering community has developed, validated and widely accepted several measures, such as various code coverage metrics for technical completeness of test-cases, no mature reference scales or target values exist for requirements specification completeness (see Sec. 6.2 (pp. 135) for an overview). Consequently, there do not exist canonical interpretations of measured value to obtain quality assessments (in the above sense).

Therefore, we argue that a quality manager must be supported in interpreting the measured values. To this end, quality assessment models must provide an *interpretation procedure*, *reference value*, *quality assessment*, and *recommendations for action*, as illustrated in the top-right part of Fig. 5.2.

Interpretation Procedure The interpretation procedure analyzes the obtained measurement value regarding two aspects. First, it provides a quality assessment of the requirements specification regarding the quality attribute under consideration. This assessment evaluates the measurement value against quality goals and provides an interpretation result which directly supporting decision making. Consider, for instance, a hypothetical measure which counts the number of spelling errors. For the sake of this example, the measurement procedure revealed that 67 spelling errors are contained in the specification. However, such a result requires further interpretation. A possible interpretation is to normalize the number of defects to the number of words of the requirements specification (provided it is a natural-language specification), compare it to certain thresholds, and provide feedback according to which thresholds it fulfills. In this example, if the normalized spelling errors are 0.09, and suppose the threshold is 0.1, i.e., $[0; 0.1)$ is interpreted as sufficient quality and $[1.0; +\infty)$ as insufficient quality, the interpretation procedure would assess the requirements specification as sufficient regarding syntactical quality.

Second, the interpretation procedure also supports quality improvement by providing recommendations for actions. Those recommendations for actions are often byproducts of the measurement procedure (e.g., concrete occurrences provided by the number of negative terms per requirement measure [Génova et al., 2013]) and provides a precise scope of the requirements specification which requires improvement together with proposing certain actions which should improve the requirements specification regarding the assessed quality attribute. However, not all measures are able to provide recommendations for actions. For instance, measures which study parts of the requirements specification considered immutable, e.g., after the fact measurements or compound metrics with extensive use of aggregations, are often inactionable (see, e.g., our work in Méndez Fernández et al. [2014] for an evaluation of actionability in RE).

Obviously, the interpretation procedure is crucial for the sound use of measurement-based assessment, and must consider the relationship between the measured attributes and the quality attribute to be assessed, as well as take into account the imperfections of measurement and interpretation itself by mitigating the associated threats to validity, or, if infeasible, appropriately reflect those in the interpretation result and recommendations for actions.

Reference Value By reference value (also called *threshold* or *target value*) we refer to a subset of a measurement scale's levels which the interpretation procedure refers to in order to obtain an interpretation result, i.e., an assessment. Hence, a reference value must not necessarily be a scalar or an interval, and it can be an absolute value or relative to other values such as prior measurement results, e.g., as used in control charts (see, e.g., Oakland [2007] for an overview). An exemplary reference values is the absolute scalar 10 for McCabe's cyclomatic complexity measure [McCabe, 1976], for which any measured value below this threshold is assessed of low or moderate complexity, and any value above this threshold as complex and subject to refactoring.

Result: Quality Assessment Assessment refers to an estimate about the extent the requirements specification possesses the quality attribute the measure purports to assess, and which is amenable to decision-making. An assessment is obtained by applying the interpretation procedure to the measurement's results, and, optionally, a set of reference values. Amenability to decision-making means that an assessment result can be used to support making justified decisions with the limited knowledge of the measure's audience, e.g., the quality or project manager. Consequently, what constitutes an adequate assessment depends on the decision to be made and the background of the persons involved. For instance, potential assessments for a method's complexity based on McCabe's measure [McCabe, 1976] is that it's complexity is either to high to be passed to downstream development and requires refactoring or not, and those assessment results are understandable to quality managers without a background in control-flow graphs.

Result: Recommendations for Action In addition to an estimation of the quality attribute under consideration, measures can also recommend actions to improve it, e.g., based on the measured value or other by-products of data collection. Actionable measures are important for feedback to requirements engineers and constructive quality assurance. However, not all measures are capable to provide such recommendations, e.g., due to lacking causality between the measured proxy attribute and the assessed quality attribute. Moreover, while many measures narrow down the specific entity, manifestation, and/or cause for impaired quality and propose counter-measures, the improvement actions can itself be costly and time-consuming. For instance, if a weak phrase such as "fast" as in the sentence "the system must react fast to user inputs" is found in a requirements specification, refining this phrase to a quantitative value may require a thorough analysis of the problem domain and require stakeholder participation.

5.1.4. Threats to Validity

Measurement-based assessment is inherently imperfect. Hence, it is crucial to understand the limitations of the employed measures to make sound decisions. To increase awareness of those limitations, assessment models confirming to our meta-model must provide an estimation of threats which are anticipated to limit validity of the measurement and interpretation. Specifically, we introduce three concepts inspired by the empirical software engineering community [Wohlin et al., 2012]:

Threats to Internal Validity: Measurement Any threats associated with the measurement procedure itself which may result in a flawed measurement values. Potential threats to validity include human factors, such as subjectivity if data collection involves human judgment, and technical factors, e.g., if an automated measurement procedure is susceptible to malformed input or false positives/negatives due to imperfect algorithms.

Threats to Internal Validity: Interpretation Any threats associated with the interpretation procedure itself which may result in an invalid assessment of the quality attribute and/or inappropriate recommendations for actions.

Threats to External Validity Any threats associated with the measure's measurement and interpretation procedure which result from its application in an usage context (see Tab. 4.2 on p. 68 for a exemplary context parameters) different to the one the measure is intended for. In particular, the reference values associated with the interpretation procedure are often subject to those threats and may require an additional adjustment.

5.1.5. Example: Number of Steps in Use-Cases

To illustrate the concepts introduced in our meta-model, we shall provide an exemplary assessment model by instantiating the meta-model. For this purpose, we rely on one single measure, namely the number of steps in a use-case as proposed by Bernárdez et al. [2004a]. A short description of the measure structured according to our meta-model can be found in appendix B.78), and its graphical representation in terms of a UML object diagram is shown in Fig. 5.3.

Note that elements annotated with the *≪result≫* stereotype are not part of the assessment model itself, but denote results of its application. In this example, a use-case consisting of two steps was measured, which was interpreted that it is likely that actions are missing and it is recommended to revise this specific use-case.

As described in the beginning of this section, since the assessment revealed that the requirements specification does not specify all actions, a quality attribute identified in Part I of this thesis. Here, the second function of the measure can be seen, namely its functions to predict the quality-in-use: The measure predicts that the particular requirements specification will likely result in diminished test quality, either due to suboptimal selection of test-cases or incompletely specified test procedures, provided the test engineer relies on the information provided in the requirements specification.

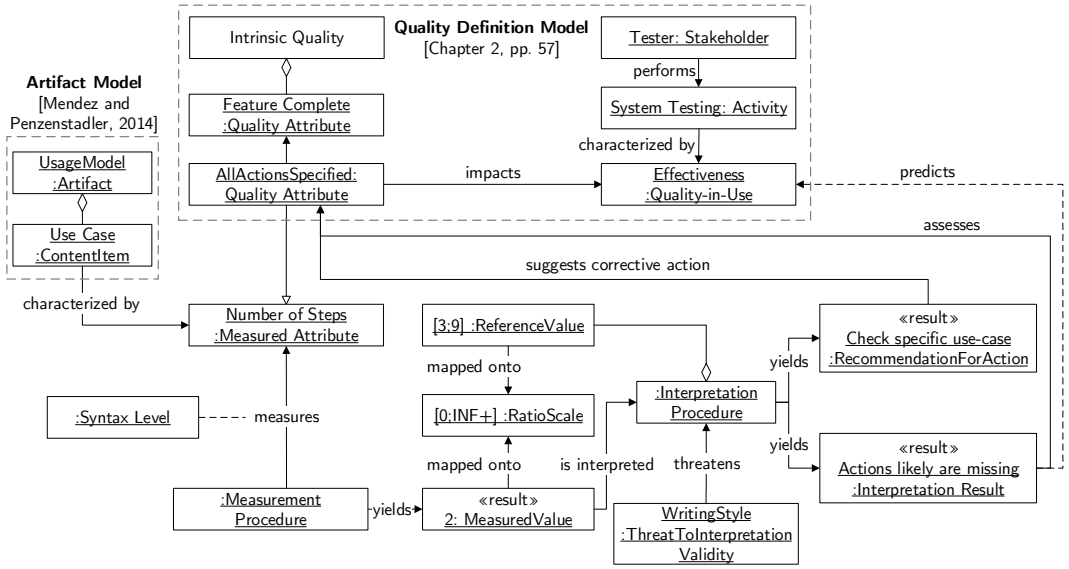


Figure 5.3.: Meta-model instantiation for the *number of steps in use-cases* measure (appendix B.78) proposed by Bernárdez et al. [2004a]

5.1.6. Relevance in Practice

To gain a first qualitative understanding of the necessity and completeness of the meta-model's concepts with regards to measurement in practice, we conducted a survey in collaboration with an industrial partner. Subsequently, we will first outline the study design before presenting and discussing the obtained results.

Study Design

The study was conducted at a worldwide operating company in the chemical business headquartered in Munich. It was situated as part of a larger collaboration between this industrial partner and three researchers from the Technical University of Munich, including the author of this thesis, and aimed to improve requirements engineering for business information systems. The enterprise application landscape consists of technologies from various partners (e.g., Microsoft, SAP), and the specified software systems were either implemented in-house or bought from external partners.

Participants Four out of seven project members from our industrial partner participated in the study. All participants were senior employees in the company. Three participants (A, B, C) classified their primary role as quality gate keeper concerned with the application of measures to obtain assessments for the projects they are responsible for (each for a different technological platform, e.g., SAP), while one participant (D)

specified to be a quality manager responsible for developing cross-project measures for software development within the company.

Survey Instrument The evaluation of practical relevance of the meta-model was part of a larger survey evaluating the RE improvement project. In this survey, we provided a section in the to the meta-model. We asked the participants to rate certain information (i.e., selected classes from the meta-model) regarding their importance for successful implementation of quality measures on a 8-point Lickert-scale ranging from *Unnecessary* to *Essential*. An even number of levels was chosen to avoid checking the middle. We selected the following classes to be rated by the participants:

- Quality Mapping
- Data Collection Procedure
- Data Analysis/Interpretation Procedure
- Reference Value(s)
- Threats to Validity
- Recommendations for Actions

In addition, we provided a text field for which we asked participants to name any information they miss for a comprehensive description of quality measures. The survey template can be found in the companion online material[2].

Data Collection and Analysis The survey was implemented as a template in a word processor and submitted to the project's contact person within the company. The contact person distributed the survey, and, after collecting all answers, submitted the survey results back to us. The survey was not conducted anonymously. For cases in which the survey result contained contradicting, unclear or otherwise interesting responses, a follow-up meeting with the participant and two researchers was arranged to gain a better understanding about the participant's intended statement. Based on this meeting, the survey's results were extended or altered.

The survey concentrated on qualitative results. For each participant individually, we looked for meta-model elements perceived as highly unimportant (rated as ≤ 2) or important (rated as ≥ 6), respectively, and extracted the reasons for this rating by interpreting the provided rationales. In addition, for each meta-model element, we calculated the average rating of all participants and considered it as an indicator for their overall importance. Finally, we extracted and discuss any information the models lacks according to the participants. Due to the small sample size, we refrained from a quantitative analysis based on statistical methods. Since the study was conducted in German, all results have been translated for publication in this thesis by its author.

Threats to Validity Since we were interested in qualitative data, we heavily relied on interpretation of the answers provided for open questions in the survey as well as the follow-up interviews. Consequently, we see interpretation bias as the main threat

[2] `http://www4.in.tum.de/~mund/metrics-companion.zip`

regarding the study's internal validity. To mitigate this threat, we applied researcher triangulation: Both the survey and interview responses were interpreted by two researchers independently, with univocal interpretations being documented. Furthermore, all statements translations were checked by a second researcher.

Since we had no control about the employees participating in the project, the generalizability of our results likely suffers from selection bias and very rather small number of participants. Remember, however, that the base population is also quite small; even in a company as big as our industrial partner, the number of quality gate keepers and managers with respects to requirements quality assurance is quite limited. In particular, the context of the study was limited to requirements engineering for business information systems to be used at one company. However, we found evidence that several elements of the meta-model are indeed crucial for applying measures to requirements specifications in practice, and this existential argument holds true even if the number of participants would be increased. On the other hand, we believe that replication, especially in conjunction with method triangulation, would largely benefit the external validity of the meta-model's completeness, since the participants asked in the present study might not have recognized the necessity of additional elements or relevant participants were not selected at all.

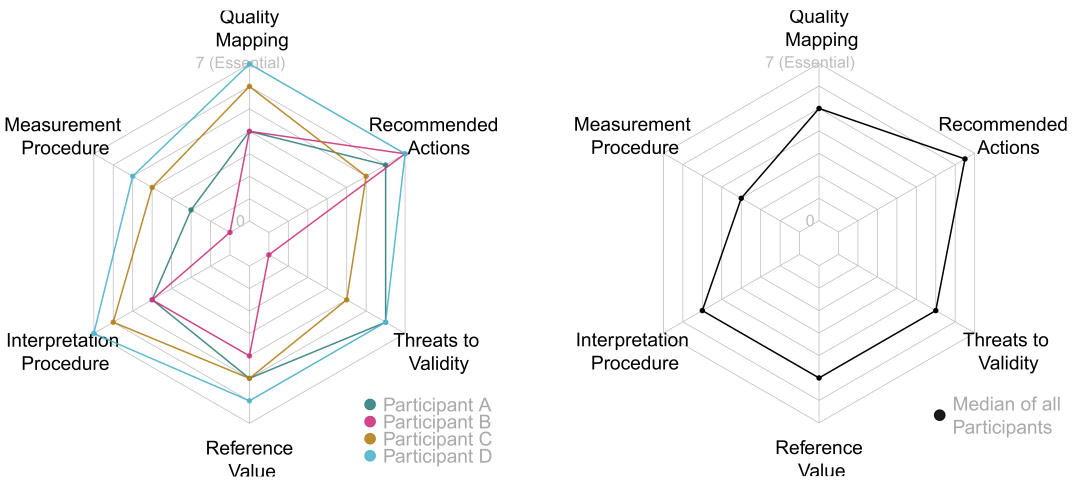
Results

The participants' ratings are visualized as a kiviat chart in Fig. 5.4, both for each participant individually (subfigure 5.4a) and the median of all participants (subfigure 5.4b).

The results revealed that the participants fairly unanimously agreed upon the importance of data analysis (interpretation) concepts, in particular the need for recommendations for action, for applying measures in practice. Practitioner C remarked that, "in all honesty, [they, i.e., the quality managers and gate keepers] all lack the experience to interpret the measures and derive implications therefrom". He further pointed out "the risk of rash conclusions not justified by the parameters but interpreted and used by the management". Practitioner A believed "it is crucial to know the measures' threats to validity in order to understand its informative power". In addition, he stated that "recommendations for action are important since they increase the measures' acceptance" since "it is not sufficient to say how *not* to do it, but also how to *improve* it".

In contrast to data analysis, participants disagreed upon the importance of the measurement procedure. While participants C and D attested above average importance, participants A and B considered data collection rather unimportant. According to a follow-up interview with Practitioner A, "the reason why data collection is considered less important as part of the measures' descriptions is, that, once defined clearly, this information is of no particular use no more".

Describing the relationship between the attribute measured and the goal of assessment in terms of a quality attribute was also considered fairly important. This quality mapping was in particular important for Practitioner D, who also provided a rationale. He stated that, according to his experience, "measures are better understood and accepted if they are associated with quality dimensions [...] such as extensibility, availability, linguistic



(a) Per Rater (b) Overall (Median)

Figure 5.4.: Importance of selected meta-model concepts as rated by practitioners (8-point Lickert-scale: Unnecessary (0)... (7) Essential)

clarity, and verifiability".

Last but not least, no participant mentioned any other concept which is missing in the model but required for applying measures in practice.

Interpretation The results suggest that the meta-model includes the most important concepts related to applying measures in practice. In particular, the participants confirmed that an assessment model for requirements specifications must specify concepts beyond measurement (in the sense of data collection) to be useful in practice, especially in terms of an interpretation procedure and recommendations for action.

Overall, since the meta-model's central concepts provide valuable information to practitioners without lacking necessary concepts, we conclude that the meta-model is adequate to describe measures holistically. Hence, an assessment model specified according to this meta-model is suitable for applying measures in an industrial context, provided that the concrete measures proposed in this model are applicable and valid themselves.

5.2. Applying the Assessment Model: A Process Proposal

Based on the quality definition presented in Part I of this thesis and the meta-model presented in the previous section, in this section, we outline a process model describing and organizing the core activities associated with assessing the quality of requirements specifications.

Guiding Principles The process model aims to support a careful and sound use of measures for quality-assessment of requirements specifications. This shall be accom-

plished by contributing to the following five guiding principles:

1. *Measure only when necessary*: As learned from Chapter 3, an efficient assessment approach must consider the significance of the requirements specification in its specific context of use. Consequently, the first activity proposed in the process model is to determine its significance, and hence, necessity.

2. *Measure only what is relevant*: A precise and valid quality definition is essential. It determines the quality attributes that are relevant and subject to quality assessment, as well as those attributes which do not impact any activities depending on the requirements specification, and hence must not be measured to avoid wasting resources and deferring attention from the crucial ones. Therefore, our approach is based on the quality definition model presented in Chapter 4.

3. *Measure only what can be measured*: Not all measures are suited for a particular context. Applying those measures may result in flawed measurements and interpretations, which, in turn, lead to poor decision-making. Therefore, the process model advocates to identify suitable measures based on the necessary prerequisites required for their application and an analysis of their threats to validity for the intended context of use.

4. *Know what cannot be measure*: Knowing for which quality attributes no suitable measures exist is important for a project or quality manager. Based on this information, one can decide on complementary means for quality assurance. In this process model, this information is obtained by comparing the quality definition model to the quality mapping (see previous section) of the set of all measures identified as suitable.

5. *Keep learning to measure*: Measurement and its use for quality assessment are inherently difficult. By incorporating activities related to a continuous improvement based on retrospective evaluation and continuous adaptation of the models, the process model supports learning curves associated with measurement-based assessment, and provides a means to constantly improve and tailor the quality assessment model to the organization and its context.

In total, the proposed process model consists of 9 activities arranged into three phases, namely preparation, assessment, and continuous improvement, as illustrated in Fig. 5.5. For each phase, the associated activities and their intended order, inputs, and outputs are shown. Subsequently, we describe those in more detail.

5.2.1. Phase 1: Preparation

During preparation, the necessary means for quality assessment are identified and implemented. This includes to determine which qualities are necessary, to identify suitable measures to assess those qualities, and their implementation in terms of infrastructure,

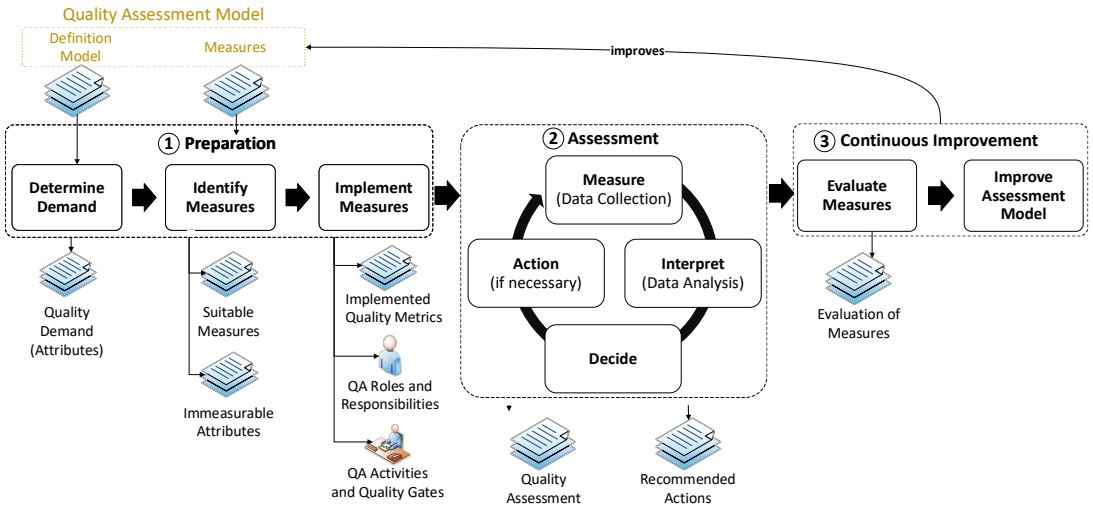


Figure 5.5.: Main activities and artifacts of the proposed assessment process model

roles and responsibilities, and activities. This phase is also referred to as the measurement design according to the life cycle model of measurement programs proposed by Jacquet and Abran [1997]. The preparation phase is envisioned to be situated between late project planning and early requirements engineering, since it requires a basic understanding of the project setting while the relevant project participants must agree on its' results before requirements validation.

Step 1–1: Determining the Quality Demand

The first step of the preparation is to determine the quality demanded from a requirements specification. Here, we distinguish between two questions: First, if the quality of the requirements specification is indeed significant, and second, what constitutes the context-specific quality.

Determining the Need for Quality Assessment As already motivated in the discussion of Chapter 3 (pp. 56), quality control of the requirements specification shall not be applied in general but only were actually required. Therefore, (project) management must decide this questions for each project individually. The significant project criteria revealed in Chapter 3 can guide this decision. In particular, for any project which develops a very complex system, is anticipated to feature volatile requirements (e.g., for highly innovative products), or which is developed in cooperation with a partner for which good collaboration is established, the effectiveness of requirements specification quality assurance should be questioned. In contrast, if the project is anticipated to take long, requires measurement of intermediate results, or develops a system which is considered safety/security-critical, the quality of the requirements specification is very likely to be

significant.

Determining the Quality Attributes Given the need for quality assessment, the attributes that constitute quality must be identified. To this end, we rely on the quality definition model presented in Chapter 4. More specifically, we see the proposed quality model as a reference model which requires further customization as described in Sec. 4.6 (pp. 93), resulting in a set of quality attributes identified for the specific project under consideration. Alternatively, the ABRE-QM approach (Sec. 4.2) could be applied.

Step 1–2: Identifying Suitable Measures

Based on the quality attributes identified in the previous step, we subsequently present how to identify measures suitable for their assessment. Here, measure is suitable if it fulfills three fundamental criteria, namely, relevancy, applicability, and validity. The goal is to restrict measurement-based assessment to those measures which are sound in the given context, and, at the same time, prohibit measuring aspects of the requirements specification which are irrelevant, and, therefore, prevent to waste efforts or misguide the requirements engineers.

Identifying Relevant Measures For each quality attribute (see Sec. 4.4, p. 77) which is considered necessary in the specific context (as determined by the previous step), the quality mapping of the assessment model (see Sec. 5.1.1) defines a set of candidate measures.

If the purpose of the quality assessment is strictly constructive, any measures which are not actionable, i.e., do not provide recommendations for actions, are excluded.

Identifying Applicable Measures For the concrete assessment model presented in this thesis, the prerequisites necessary to implement or meaningfully apply it are elicited for each individual measure contained in the assessment model. As shall be seen in Chapter 6, we distinguished prerequisites regarding the artifacts, activities, organization, skills/expertise of staff, and other project-related factors.

A measure is considered applicable if every associated prerequisite is fulfilled (or will be established before its application). For instance, several measures require a certain specification language, e.g., natural-language or KAOS goal modeling language. Applicability has to be evaluated for every measure considered relevant. For the assessment model presented in Chapter 6, this can be accomplished efficiently by consulting Tab. 6.12.

Identifying Valid Measures Finally, validity has to be evaluated for any remaining, i.e., relevant and applicable, measure. To this end, the quality manager consults the description of the threats to validity (measurement, interpretation and external). In terms the measure under consideration is a surrogate measure, i.e., it actually measures a related attribute instead of the quality attribute itself, the quality manager must also understand this relationship in detail. Finally, the quality manager judges whether or not

the measure is supposed to provide reliable and valid assessments in the specific project context.

The result obtained by identifying relevant, applicable and valid measures constitute the set of suitable measures. In addition, any quality attribute for which no suitable measures were identified is also made explicit and constitutes the immeasurable quality demand.

Step 1–3: Implementing the Identified Measures

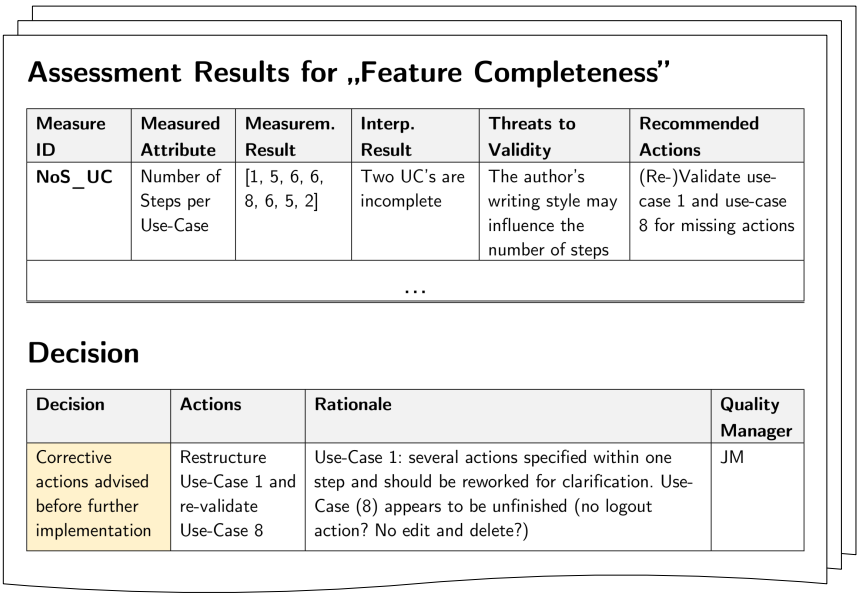
The last step of the preparation phase is concerned with the socio-technical implementation of the suitable measures. Since a thorough discussion about quality governance is beyond the scope of this thesis, we will only outline the main results from this step and provide references for further reading.

From the technical perspective, any automated or semi-automated step of the measures' measurement or interpretation procedures requires an operational infrastructure. Therefore, the necessary tooling must be implemented and maintained, including procurement, installation, configuration, bug-fixing, adaptation to changing environments, etcetera.

From an organizational perspective, the measures must be integrated into the product's or project's quality assurance program, which typically impacts, among others, planning (e.g., allocating the necessary resources) and the development process. A common approach in practice is to install quality gates at certain project milestones which ensure that only requirements (specifications) are passed on to subsequent activities which fulfill the necessary quality criteria (see, e.g., Robertson and Robertson [2006], Cha. 11, for an introduction to quality gates in requirements engineering). In particular, it is vital that the organization has a clear plan on how to act if the assessment indicates that the desired quality is not achieved [Niessink and Van Vliet, 1999]. For each measure, similar to establishing tooling in the technical perspective, necessary prerequisites for conducting semi-automated or manual steps must be established. For instance, clear responsibilities must be assigned to the staff, and training must be provided if necessary. Hall and Fenton [1997] provide further success criteria for implementation, e.g., to appoint internal metrics champions with the assessment program, invite external metrics gurus, secure commitment from management, and introduce measures incrementally.

5.2.2. Phase 2: Assessment

The assessment phase is concerned with obtaining a assessing the quality of the requirements specification, refers to a single application of a measure to one obtain assessment result, and consists of four elementary steps: measurement, interpretation, decision, and action (if applicable and necessary). In the measurement program life cycle model by Jacquet and Abran [1997], this phase covers both the application of measurement and the exploitation of the obtained measurement result. Since individual measures may be applied several times (e.g., for trend measures, to provide early feedback to requirements engineers or to check the effectiveness of corrective actions), the assessment phase



Assessment Results for „Feature Completeness"

Measure ID	Measured Attribute	Measurem. Result	Interp. Result	Threats to Validity	Recommended Actions
NoS_UC	Number of Steps per Use-Case	[1, 5, 6, 6, 8, 6, 5, 2]	Two UC's are incomplete	The author's writing style may influence the number of steps	(Re-)Validate use-case 1 and use-case 8 for missing actions
...					

Decision

Decision	Actions	Rationale	Quality Manager
Corrective actions advised before further implementation	Restructure Use-Case 1 and re-validate Use-Case 8	Use-Case 1: several actions specified within one step and should be reworked for clarification. Use-Case (8) appears to be unfinished (no logout action? No edit and delete?)	JM

Figure 5.6.: Exemplary assessment report (excerpt)

is depicted as a circle in Fig. 5.5 to indicate that subsequent measures are potentially influenced by the outcome and actions implemented according to previous assessments. Moreover, one assessment of requirements specification quality typically comprises several measures, which is illustrated by sketching multiple instances of this phase. Typically, the assessment phase is part of requirements validation.

***Step 2–1:* Measurement and *Step 2–3:* Interpretation** Measurement refers to the application of a concrete implementation of a measurement procedure as established during the preparation phase. Automated steps are executed based on the given infrastructure while manual steps are conducted by the staff members responsible. As a result, a measurement value on the associated measurement scale is obtained. In a second step, this measurement value is then interpreted based on the implemented interpretation procedure, which yields a quality assessment and, optionally, recommended actions.

The top half of the exemplary report shown in Fig. 5.6 illustrates one possibility to capture the results of the measurement and interpretation steps. This exemplary report revisits the number of steps per use-case measure, which was used to also illustrate the meta-model (see Sec. 5.1.5). Here, the application of this measure to each use-case revealed that two of them are potentially incomplete, namely use-case 1 and 8, as determined by interpreting the vector representing the individual use-cases and the interpretation procedure which states that any use-case with less than three steps is likely to be incomplete.

***Step 2–3:* Decision** According to our understanding of measurement-based assessment, the responsible quality manager (QM) has the final say if the requirements specification is of sufficient quality and which additional actions, if at all, should or must be applied. This deliberate design decision is based on the assumption that even sophisticated interpretation procedures profit from human judgment due to the extent of uncertainty and confounding factors associated with measurement. To this end, the assessment report informs the quality manager about the measures' results, threats to validity, and relationship between the measured and assessed quality attribute. If necessary, the quality manager may also review the requirements specification itself regarding the detected deficiencies. The bottom half of the exemplary report shown in Fig. 5.6 illustrates the quality manager's judgment. Here, the quality manager assessed the features are insufficiently specified, and suggests concrete actions to be taken by the requirements engineers to improve the requirements specification's quality.

***Step 2–4:* Action** Finally, if corrective actions are advised or required, those actions must be realized. In most cases, the role responsible for this task are the requirements engineers. Depending on the actual quality processes, a follow-up assessment may be invoked to ensure effectiveness of the corrective actions. This iterative assessment approach may continue until sufficient quality is established, supporting the learning curve often associated with requirements engineering for software-intensive systems [Broy, 2006, Pohl, 1994].

5.2.3. Phase 3: Continuous Improvement

Measurement-based quality assessment is seldom optimal when used for the first time, but several shortcomings with the assessment model ought to be improved over time if the assessment programs are evaluated and adjusted systematically and on a regular basis. Therefore, we envision a continuous improvement phase which evaluates the measures' performance in retrospective, i.e., after the software-intensive system is shipped to the customer or made available on the market.

Step 3–1: Evaluating Measurement-based Assessment Performance

To evaluate the measures' performance, we need a golden standard of the actual requirements specification quality. Since the continuous improvement phase is situated after project completion, we can rely on retrospective measures, e.g., customer satisfaction or change requests. Based on the retrospective measures and the teams perceived retrospective quality, discrepancies between measured quality and retrospective quality can be identified.

Step 3–2: Improving the Assessment Model

For any identified discrepancy, the team should aim to identify deficiencies and derive potential improvements associated with the measures. Potential improvements can refer

to, for instance, adapting the reference values to better fit the organization, team or project setting, extend interpretation procedures by additional validation steps to address common mistakes, remove steps which do not justify the value provided, or add or remove certain measures from the assessment model/catalogue of measures which turned out to provide valuable feedback or be unreliable, respectively. Analogously, any measure which provided precise estimations of quality should also be designed as recommended. In any case, a characterization of the organizational context of use (see Tbl. 4.2 (p. 68) for an overview) should be added to allow future projects to be able to relate measurement performance to their specific situation at hand.

5.3. Related Work

In this section, we discuss scientific work closely related to the meta-model and process model.

Structure Model

Several authors proposed conceptual structure models for measurement in software engineering similar to the meta-model presented in this chapter. Kitchenham et al. [1995] presented a structure model as part of a framework for validating software measurement. Indeed, data collection aspects of the meta-model are closely related to those of the model presented in the paper. In particular, the authors also recognized the need to explicitly model indirect measurement by mapping empirical associations between attributes to formal attribute relationship models in terms of equations. However, the model lacks several aspects we regard as necessary for quality assessment model of requirements specifications. First, the model does not distinguish between attributes of an entity in general and quality attributes of the requirements specification. Second, the model does not consider aspects of data analysis or interpretation but is limited to data collection (i.e., obtaining a measured value) instead. Third, the model does not reflect the uncertainty and limited understanding associated with measurement of requirements specification quality. Last but not least, since the model was not proposed as a meta-model, it does not designate the concepts to be described in an assessment model from concepts obtained by applying such a model.

Boegh et al. [1999] propose a method for software quality assurance called *Squid*, in which all measures are stored in a database according to an explicit data model. Unfortunately, the authors do only provide a simplified representation of this data model, resulting in certain concepts being described at a high level of abstraction. While, in general, the model is quite similar to our meta-model, we identified several differences. First, the authors do not distinguish between the measured and assessed quality attribute, and thereby either neglect the importance of specifying its relationship or outright exclude any proxy measures from the approach. Second, the model does not specify threats to validity associated with the measure. Third, a prescriptive reference value (called target value) takes the place of the more general concept of an interpretation procedure

since according to the Squid approach, interpretation is limited to comparison of measured values to reference values. Finally, the model does explicitly contain the concept of an experience base, which records measured and target values resulting from previous measurements and across projects together with the "essential characteristics of the development process such as the type of development model and the language adopted".

The structure model most closely related to our work is the Quamoco quality meta-model [Wagner et al., 2012]. Similar to our approach. it distinguishes between the meta-model's elements for specifying a quality model, called definition layer, from those of applying such a model, called application layer, which include many commonalities at both levels and can been seen as consistent to each other. However, both meta-models emphasize different aspects of assessment because of the intended application areas; while our meta-model emphasizes the limitations and uncertainty associated with measures for requirements specification quality, the Quamoco meta-model aims to specify software quality assessment models with specifications which are closer to operationalization. This manifests in subtle differences in the meta-models. Specifically, at the definition level, the meta-model presented in this thesis explicitly distinguishes between direct and proxy measures, supports a wider class and more extensive characterization of the relationships between the corresponding attributes, and incorporates threats to validity. In particular, while the Quamoco meta-model is limited to a (positive or negative) impacts between attributes, our meta-model does not necessarily require causality. Furthermore, our meta-model specifies an additional result at the application level, namely recommendations for action, to support quality improvement. In contrast, the Quamoco meta-model represents data collection and analysis in further detail and from an rather algorithmic viewpoint by suggesting a set of interpretation functions as building blocks, such as aggregation functions for the construction of composite (derived) measures, or linear functions for interpreting measurement results of base measures.

Process Model

The ISO/IEC 15939:2007 [2007] standardized a generic measurement process for software. Is is defined according to four high level steps, namely to establishing and sustaining measurement commitment, planing the measurement process, performing the measurement process, and evaluating the measurement process, which are further refined in various activities and, finally, tasks. While this process model describes crucial activities, in particular, a systematic selection of measures (steps 4.2.3.1 and 4.2.3.2) based on certain criteria (cf. annex C) and the need continuous improvement of the measures (step 4.4.1), its genericness and high level of abstraction makes it unsuitable for practitioners for the purpose of assessing requirements specification quality.

Niessink and Van Vliet [1999, 2001] propose a generic process model for measurement-based improvement which aims to ensure that measures generate value (rather than data) for an organization. The model consists of two main process areas concerned with improvement and measurement, respectively. Each process area includes an analysis as well as an implementation activity. While the authors themselves recognize the model's unsuitability as a prescriptive model for practitioners due to the high level of abstraction,

the model is used to identify success factors external to measurement, such as the necessity to explicitly specify (the assumptions on) the relationship between the measured attribute and the quality attribute, and how to react to different outcomes. The framework presented in this chapter provides a consistent but more detailed characterization of the activities of measurement and pervades aspects of improvement which are specific for assessing and improving the quality of requirements specifications. In contrast to the generic process model, improvement in out framework corresponds to identifying the quality demand for requirements specifications for a particular project or product for the analysis part, and implementing the decisions and recommendations resulting from the measures' interpretations for the implementation part.

Jacquet and Abran [1997] propose a process model structured according to four main steps: (1) Design the measure, (2) apply the measurement procedure, (3) analyze validity of measured value, and (4) exploit the measured value. The approach concentrates on developing and, to a lesser extent, applying measures based on objectives. Consequently, the focus is different compared to our process model. It describes the activities required to come up with a novel measure instead of selecting suitable measures for a given problem, and does not provide information on how to exploit the results other than naming models the measured value can be potentially used in. However, the model disagrees with our approach in one aspect: while the authors advocate a mandatory audit of any measurement result (e.g., checking "the tricky parts of mathematical calculations"), we are more liberal in the sense that deciding on the necessity of validation steps are subject to an analysis of individual measures based on factors such as its reliability, costs/benefits ratio, impact on decision-making.

Frederiksen and Mathiassen [2005] propose an approach to improve measurement-based assessment approaches. In this approach, the status-quo of measurement is contrasted with idealized views on measurement and discussed with key stakeholders in order to identify improvements to the assessment program. However, while this approach to be suited for organizations striving to improve existing assessment programs, the high efforts associated with this approach and its lack of a normative process model constitute high entry barriers for practitioners.

Boegh et al. [1999] propose a method for software quality assurance called *Squid* which includes a process model. The model's activities are classified into four process areas, namely quality specification, quality planning, data collation, and quality evaluation and monitoring. Compared to the goal-oriented process models, the model is parametric in a quality model, which must be selected during quality specification. This quality model is closer to our understanding of a quality assessment model then a definition model since it is assumed to provide measures for the quality attributes defined in it. In the Squid approach, interpretation is limited to comparing the measured value against a single prescriptive reference value, which represents either a lower or upper target. Squid also features a database which contains the definitions of all measures according to the Squid data model (see above), and an experience base which captures the target and historic values across projects, and which provides reference values for various quality activities, e.g., recommendations for target values during quality planning of upcoming projects.

Contributions to the State-of-the-Art

Essentially, while the assessment meta-model and process model proposed in this chapter are consistent with the state-of-the-art assessment frameworks in software measurement, both models are the first to specifically account for the peculiarities of measuring requirements specification quality by catering to the careful use of measures for the specific purpose of requirements validation. In particular, the framework explicitly represents and takes into account the associated threats to validity and the relationship between the measured and the assessed quality attribute, relies on an explicit interpretation procedure, makes extensive use of context-dependent tailoring, and accounts for the expected learning curves. Furthermore, it also deliberately excludes those aspects which are well beyond the current understanding in the requirements engineering community, e.g., mathematical models of relationships between quality attributes.

5.4. Summary

In this chapter we proposed a framework for the assessment of requirements specification quality which consists of a quality assessment meta-model and a complementary process model. The meta-model ensures that quality assessment models describe measures holistically. In particular, the meta-model enforces to explicitly specify the relationship between the measured and assessed attributes, to provide a procedure to interpret the results of data collection regarding decision-making and corrective actions, and to specify circumstances anticipated to threat the measures' validity. We conducted a qualitative study in which practitioners confirmed the utility of this meta-model. The complementary process model describes and organizes activities for a systematic and sound application of such a quality assessment model. It is organized according to three phases, namely preparation, assessment, and continuous improvement. The phases determine the quality demand and implement suitable measures, perform individual assessments, and provide means to constantly improve the assessment model, respectively.

6 Chapter

Assessment Model based on State-of-the-Art Measures

In this chapter, we propose a quality assessment model for requirements specifications. This assessment model extends the quality definition model presented in Chap. 4 by associating concrete measures with a specification's quality attributes based on the meta-model extension presented in Sec. 5.1. Thereby, we obtain a comprehensive quality assessment model which can be used to measure a specification's quality for concrete software projects in practice, as outlined in our process proposal in Sec. 5.2.

To this end, we conducted an extensive systematic literature review in which we examined 2397 scientific publications, thereof 166 in detail, resulting in 136 metrics identified in total. Those measures were further analyzed in-depth, using quantitative and qualitative research methods, in order to embed them into an unified assessment model, including a consistent (re-)evaluation of their crucial aspects for quality assessment. In particular, the proposed assessment model provides a refined view on the relationship between the measured property and the assessed quality attribute, the scope of measurement with regards to the requirements specification contents, and the prerequisites that must hold for the metric to be applicable.

This chapter is structured as follows: First, we describe the applied research method in Sec. 6.1, specifically the research questions, the protocol followed for the literature review, and the scientific method applied during the analysis. Next, we present the obtained assessment model, organized according to semantic, syntactic, social and pragmatic quality, in Sec. 6.2 through 6.5, with an own section for compound metrics (Sec. 6.6). Subsequently, we present the scope of measurement and the identified prerequisites of the

model's measures in Sec. 6.7 and Sec. 6.8, respectively. Finally, we outline related work in Sec. 6.9 before summarizing our results in Sec. 6.10.

6.1. Research Method

In this section, we provide details on our research method. Essentially, we conducted a systematic literature review following the guidelines of Kitchenham [2004] to identify metrics, which were further used as input to an in-depth analysis procedure which used classifications and coding techniques as known from grounded theory [Strauss and Corbin, 1998] in qualitative research.

6.1.1. Research questions

The present systematic literature review and in-depth analysis studies the publication landscape for the purpose of structuring, unifying and extending the state-of-the-art of quality measures with respect to their validity and applicability from the point of view of researchers in the context of requirements specifications for software-intensive systems. To this end, we state the following three research questions:

RQ 6.1 Which metrics assess which requirements specification quality attributes, and to what extent?

RQ 6.2 What scope of the requirements specification do those metrics measure?

RQ 6.3 What are prerequisites for applying those metrics?

RQ 6.1 investigates the metrics proposed in the scientific literature to measure quality attributes of the requirements specification. In particular, we are interested in the extent to which those metrics indeed resemble the empirical observations regarding the quality attributes to be assessed according to the publications. In RQ 6.2, we aim to understand the kind of properties and the parts of the specification actually measured during data collection. By *kind* we refer to the level of perception of an artifact in terms of its physical representation, syntax, or semantics, and by *part* we refer to the semantic contents of a specification as depicted in Fig. 2.1 (p. 21). Finally, in RQ 6.3 we are interested in what necessary prerequisites must be given for a measure to be (meaningfully) applicable. More specifically, we want to know which characteristics the artifacts, activities, project staff, organization, and project must possess during the implementation, measurement or interpretation of the identified quality measures.

All three research questions contribute to build a catalogue of quality measures for requirements specifications. This catalogue is organized according to the meta-model defined in Chap. 5.1 and, for the sake of readability, presented collectively in appendix B.

6.1.2. Databases and Search Strings

We searched for papers that are written in English and available online exclusively by querying the following scientific databases:

	Search String	Limit
S1	(requirements metrics)	100
S2	($\mathbf{OR}_R$) `and` ($\mathbf{OR}_M$)	100
S3	($\mathbf{OR}_R$) `and` ($\mathbf{OR}_M$) `and` (quality)	100
$\mathbf{S4}_{q \in Q}$	($\mathbf{OR}_R$) `and` ($\mathbf{OR}_M$) `and` q	10×50

Table 6.1.: Search strings

- IEEE Digital Library (Xplore, `http://ieeexplore.ieee.org`)
- ACM Digital Library (`http://dl.acm.org`)
- ScienceDirect (Elsevier, `http://www.sciencedirect.com`)
- SpringerLink (`http://link.springer.com`)

To formulate concrete search strings, we defined the following sets of terms for the notions of measure, requirements, and quality:

Measure M The following terms were used for the notion of measure: `Metric, Measure, Measurement, Quantification, Assessment, Indicator`

Requirements R The following terms were used for the notion of requirement: `Requirements, Requirements Engineering, Requirements Specification, Use Case`

Quality Q The following terms were used for each of the quality attributes (synonyms in parenthesis): `Correct, Complete, Precise, Consistent, Design Independent (Implementation Free), Agreed, Unambiguous (Ambiguous), Concise, Singular (Atomic), Organized`

Based on those terms, we formulated the search strings specified in Tab. 6.1. Here, an $\mathbf{OR}_X$ denotes the concatenation of all terms contained in the set X using the keyword `or`. Furthermore, for $S4$, we actually generated ten search strings, one for each quality attribute term $q \in Q$, and used an `or` for the synonyms specified in parenthesis.

6.1.3. Inclusion and exclusion criteria

We demanded the following inclusion criteria to be present:

IC1 The requirements specification under consideration is part of a systems- or software-engineering approach

IC2 Measures are a main contribution of the publication and presented in thorough detail. In particular, we must be able to identify (i) the obtained mathematical object, (ii) the associated scale, and (iii) the data collection procedure which maps empirical observations onto the mathematical object.

IC3 The associated scales and interpretation procedures must have sufficient discriminative power to support decision making in practical situations.

In contrast, we excluded papers based on the following criteria:

EC1 Papers of the following type (based on Wieringa et al. [2006]): philosophical, opinion and personal experience papers, secondary studies

EC2 Papers on measures not applied on the requirements specification (e.g., process measures)

EC3 Papers on measures not assessing intrinsic quality, i.e., the measured attribute is not equal or closely related[1] to those specified in the quality definition model (Chap. 4).

6.1.4. Quality assessment

We demand publications to be scientifically sound. Specifically, any relation between the proposed metrics and the associated quality attributes claimed must be justified. Therefore, for deductive, non-empirical papers, we demand that a comprehensible and sound argument is provided, ideally backed up by peer-reviewed scientific publications. For empirical evidence, we demand a sound empirical protocol, a rigor execution of this protocol and a conclusive reflection on the validity threats and mitigation procedures used. Empirical studies in which the validity of results is not convincing because of unmitigated threats to construct/internal validity or in which the study context severely prohibits a generalization to other software-engineering endeavors (external validity) were excluded.

6.1.5. Data collection

For data collection, we queried the aforementioned databases with the search strings of Tab. 6.1. For each search string and database, we ordered the results by relevance, and extracted the meta-data of the publications up to the limit specified in Tab. 6.1 into an Excel spreadsheet. For each publication, we extracted the data specified in Tbl. 6.2, obtained either from the digital library database or the paper directly:

6.1.6. Data analysis

Subsequently, we briefly describe our data analysis procedure consisting of three major steps: harmonization, voting, and in-depth analysis.

[1] By *closely related*, we refer to intrinsic properties which are specializations of quality attributes from the quality definition model presented in Chapter 4 (e.g., anaphoric ambiguity with regards to general ambiguity), or which provide a sound theoretical foundation for correlation (e.g., duplication and conciseness). Unfortunately, we do not know of any suitable metric to quantify this relationship for the purpose of this study.

Meta data	ID, Citation-key, Publication organ
Content	Authors, Title, Abstract, Keywords, Year
Voting	Relevance (per researcher, result of voting), Rationale

Table 6.2.: Data collection table (simplified)

Harmonization

In a first step, we harmonized the collected publications by removing duplicates. To this end, we first clustered all identified publications according to its title. For each cluster, we manually sighted and removed duplicates, preserving the reference to the digital library the paper first occurred in, if applicable. We repeated this procedure on the remaining publications by clustering according to author and year.

Voting

To select only the publications relevant for our research questions, we rated each paper's relevancy independently on a dichotomous scale (yes/no), based on the collected data (see Tbl. 6.2), and state a rationale based on the inclusion/exclusion criteria. Overall, this step took 3 full days of work.

In-Depth Analysis

The aforementioned procedure yielded the final set of papers to be investigated in further detail. To this end, the complete contribution of each paper in this set is analyzed in detail by the author of this thesis. On average, we analyzed about 3 publications in a full day of work, and the complete analysis was done in a period of about 6 month. The data collected during in-depth analysis for each identified measure is outlined in Tbl. 6.3, based on the meta-model for quality assessment described in Chap. 5. For the sake of readability, the complete catalogue of measures is presented collectively in appendix B. In the following, we summarize the in-depth analysis procedures used to answer our research questions.

RQ 6.1 – Existing SRS Quality Metrics For RQ 6.1, all measures proposed and/or evaluated in the contribution are identified. In case it measures an intrinsic quality attribute as described in Chapter4 and fulfills the quality assessment criteria (see Sec. 6.1.4), the metric was summarized and added to the model (see appendix B).

Furthermore, for any measure identified, we classified whether the measure is scientifically sound, (a) based on formal reasoning (theoretical validity) or a sound argument (underlying theory validity), (b) backed up by empirical evidence obtained in a sound, scientific method or (c), both. If the measure was classified as unsound, it was discarded from the model since it does not pass the quality assessment (Sec. 6.1.4).

Regarding the *extent*, we investigated three fundamental but related aspects based on Meneely et al. [2012]: (i) the extent to which the *representation relation directions*

ID, Name, Source	A unique identifier for the metric, a self-speaking name, and the source publication(s) it was described in
Data Collection	Description of the collection procedure, the mathematical object obtained and the associated scale
Data Interpretation	Description of the interpretation procedure, i.e., the evaluation of the associated quality based on the obtained measurements. In particular, prescriptive reference values are presented if applicable.
Recommendations for Action	Description of any actions recommended in the contribution in order to improve the associated qualities depending on the interpreted measurement.
Associated Qualities *(RQ 6.1)*	The internal quality attributes measured by the metric. Each identified relationship is manually categorized regarding (i) the *level of evidence* (backed up by empirical results and/or argumentative reasoning), and (ii) the *extent* to which the metric interpretation is able to assess the associated quality of the specification.
Measured Scope *(RQ 6.2)*	The scope of the artifact measured during data collection. Here, we distinguish between the *semiotic scope*, i.e. the kind of artifact property (physical, syntactic or semantic) measured, and the *content scope*, i.e., which contents of the requirements specification are measured.
Prerequisites *(RQ 6.3)*	Prerequisites and mandatory conditions for implementing the metric.

Table 6.3.: Elements of Measures Catalogue (simplified)

are covered by the metric, (ii) the *level of certainty* associated with those relations, and (iii) the extent to which the associated quality attribute as a whole is covered (*content validity*). According to measurement theory (e.g., Suppes and Zinnes [1962]), the representational condition demands that the empirical observations of the quality attribute are preserved in the number system of the metric, i.e., that the empirical structure and the mathematical object and scale described by the metric are isomorphic[2]. For (i), we classify whether the (a) measurements obtained indeed represent the empirical observations[3], (b) measurements imply the according empirical observations[4], or (c) both. According to Meneely et al. [2012], (a) is also called *attribute-valid* since the measurements correctly exhibit the attribute that the metric is intending to measure, while (b) promotes *non-exploitability* since the obtained measurement results (measured values) cannot be manipulated without changing the attribute being measured. For (ii), we classify the level of certainty according to two dimensions: First, we classify whether the relationship is (a) deterministic or (b) probabilistic, i.e., whether we can be certain for even a single requirements specification or if we consider effects on sets of requirements specification. Second, we classify the metric according to whether it was obtained (a) deductively (based on an a-priori theory) or (b) inductively, e.g., from data-mining without an a-priori theory. Finally, for (iii), we classify whether the metric is valid for the attribute as a whole, also called *content validity* Meneely et al. [2012]. If not, we also specify the aspects of the quality attribute the metric measures. Specifically for such an indirect relationship, i.e., every measure for which the measured attribute differs from a quality attribute of the internal quality definition model (see Sec. 4.4), we further classify the relationship between the measured attribute and the quality attribute. To this end, we proceed as illustrated by the decision tree in Fig. 6.1. First, we investigate whether the measured attribute is essentially the quality attribute but limited to a particular type or aspect of it. If not, we further investigate whether the measured attribute is in a causal relationship to the quality attribute, either in terms of the measured attribute causing or being caused by the associated quality attribute. In any case, we provide an explanation for the relationship which also serves as a rationale for the classification.

RQ 3.2 – Measured Scope Regarding the measurement procedure, we classify what scope of the requirements specification the measure actually measures. To this end, we classify for each of it what kind of artifact property is measured. Here, we distinguish between (a) physical properties concerned with the physical representation of the re-

[2] Formally, for any measurements m_A, m_B obtained for two requirements specifications A and B, the associated number system relation ($\circ_M$) must hold if and only if the corresponding empirical relation $\circ_E$ holds for two empirical observations e_A, e_B of a certain quality attribute, i.e., $m_A \circ_M m_B \Leftrightarrow e_A \circ_E e_B$. For instance, if the requirements specification of project A is (empirically) considered as more complete than the requirements specification of project B, and given a metric which computes an integer c_A and c_B for those requirements specifications on an absolute scale with the interpretation that a smaller number denotes less completeness, the representation condition would demand that $c_A \leq c_B$ holds.

[3] $e_A \circ e_B \rightarrow m_A \circ m_B$

[4] $m_A \circ m_B \rightarrow e_A \circ e_B$

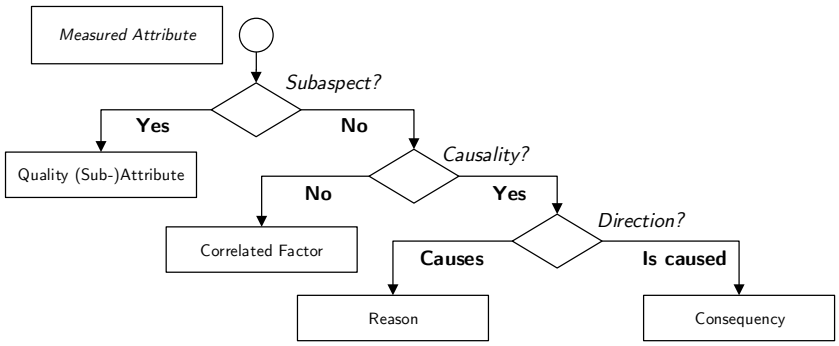


Figure 6.1.: Decision tree to classify the measured attribute with respect to the quality attribute the metric purports to measure

quirements specification (e.g., number of pages of a printed requirements specification or file-size of the document), (b) syntactic properties concerned with the language(s) used and the syntactic organization of the requirements specification (e.g., the number of grammatical mistakes or the average depth of the document's outline), and (c) semantic properties concerned with the information described in the document (e.g., the number of statements in the requirements specification which at least one stakeholder does not agree on. In addition, we describe the semantic contents which the metric under consideration is able to measure by specifying the content items it is applied on. To this end, we specify a (sub-)set of the content items defined in the AMDiRE artifact model as shown in Fig. 2.1 (p. 21).

RQ 6.3 – Necessary Prerequisites In RQ 7.3, we investigate necessary prerequisites of the measurement and interpretation procedures of the identified metrics. Specifically, we are interested in the following six aspects:

- **Artifact:** Notations, languages, templates and physical representations used in/for the requirements specification
- **Activity/Process** Software process used as well as demands regarding the presence or implementation of individual activities and tasks
- **Infrastructure** Availability of software (e.g., specific tools) and hardware (e.g., computing performance) infrastructure
- **Organizational** Demands on the organization such as the presence of certain management structures or the organizational culture
- **Skills/Expertise** Required skills, expertise and experience of the persons involved
- **Project** Characteristics of the project, such as the domain the system is developed for or the project constellation (e.g., out sourced projects)

In a first step, we extract a free-text summary of the necessary prerequisites for each identified metric and aspect. Each such summary must be backed up by a rationale,

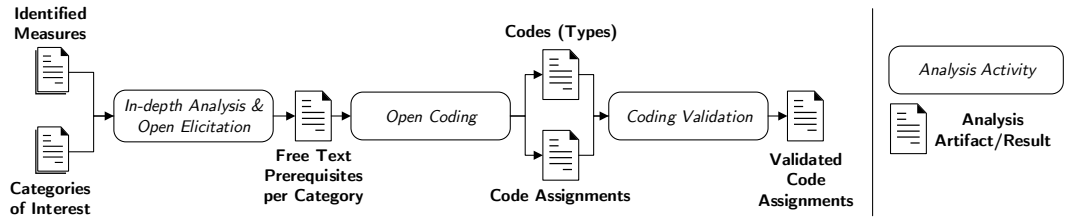


Figure 6.2.: Illustration of in-depth analysis process of necessary prerequisites (RQ 7.3)

which can be either an explicit reference to a citation in the paper or an argument provided by the researcher who conducted the analysis. Next, we perform open coding, a technique known from grounded theory [Strauss and Corbin, 1998] in a simplified manner by introducing and assigning codes to each extracted summary on-the-fly, and thereby iteratively obtaining a list of codes. Based on those initial codes, we subsequently identify relationships among codes, which allows to unify certain codes and to provide a taxonomy. As a result, we obtain obtain a finalized set of codes (types) and tentative code assignments for each measure. Finally, we must validate that the identified codes are assigned consistently. Therefore, we re-evaluate all measures against the finalized set of codes, and assign missing codes where needed to obtain the final code assignments. The procedure is illustrated in Fig. 6.2.

6.1.7. Validity Procedures

Since the classification of metrics and its coding during in-depth data analysis is performed by a single researcher, misinterpretations and subjectivity inevitably occurred and impact the study's *internal validity*. While a (complete) researcher triangulation is out of scope, we resorted to capturing rationales for major analysis results in order to increase transparency of results and avoid unjustified classifications.

To control for *external validity*, we have to ensure all relevant publications are included in the literature search. Therefore, we iteratively extended the review protocol in terms of search strings and limits to include an a-priori set of publications, obtained according to the researchers' expertise and snowballing, consisting of the following 15 publications: Bernárdez et al. [2004a], Berenbach and Borotto [2006], Costello and Liu [1995], Davis et al. [1993a], El Emam and Madhavji [1995b], Fabbrini et al. [2000], Femmer et al. [2014a], Génova et al. [2013], Hoffman [1989], Juergens et al. [2010], Kaiya et al. [2002], Li et al. [2008], Monperrus et al. [2013], och Dag et al. [2001b], Wilson et al. [1997]. Furthermore, in addition to querying the digital libraries, we also queried a manually maintained directory of about six-thousand publications from the field of requirements engineering called *Requirements Bibliography* (`www.reqbib.com`) using the search string $\mathbf{OR}_M$. We compare the publications identified there compared to the digital libraries, adding any newly identified records to the result set.

Database	Hits	Collected	(in %)	After Harm.	(in %)
ACM DL	196839	677	(0.3%)	522	(77%)
IEEE Xplore	31031	800	(2.6%)	601	(75%)
SpringerLink	81095	946	(1.2%)	688	(73%)
ScienceDirect	5500	777	(14.1%)	481	(62%)
All Databases	314465	3200	(1.0%)	2292	(72%)
Req. Bib		165		101	(61%)
Val. Set		15		4	(27%)
Total				**2397**	

Table 6.4.: Results from the search process

6.1.8. Deviations from Protocol

We were forced to deviate from the protocol regarding one aspect only. Since some of the databases exposed quite irrational results for certain operators (e.g. Boolean operators in the ACM Digitial Library) or imposed individual restrictions (e.g., regarding the length of search strings or operators), we had to fine-tune the search strings for each database. Therefore, we had to resolve some of the $\texttt{OR}_X$ by splitting into multiple strings for each $x \in X$ and adjust the associated limits respectively. Furthermore, for some databases and queries, we had to manually select the query operators that yield the results most promising according to the author's perceptions regarding the validation sets. The precise search strings, limits used, and results are available in the online companion material[5].

6.1.9. Search Results

The results of the search process are presented in Tbl. 6.4. As seen here, the search strings (see Tbl. 6.1) yielded a high number of hits in each database, ranging from 5500 (ACM) to 196839 (ScienceDirect) publications. In average, we extracted between 677(0.3%) and 777(14.1%) of the most relevant publications due to the limits specified for each search string. During harmonization, we identified 908(28%) publications to be duplicated due to overlap in the individual search strings' results. In addition to the database queries, the results also include publications identified during validation. The Requirements Bibliography (`www.reqbib.com`) identified 165 publications, of which 64(39%) were already included in the search results. In addition, the reference set of 15 highly relevant publications were contained in the search results to a large extent, with only 4 publications (27%) missing. After examining the missing publications during validation, we identified that missed publications are to a large extent older items published in heterogeneous publication organs not contained in the study, e.g., the *Cutter IT Journal* or *The American Programmer*.

[5] `http://www4.in.tum.de/~mund/thesis-companion.zip`

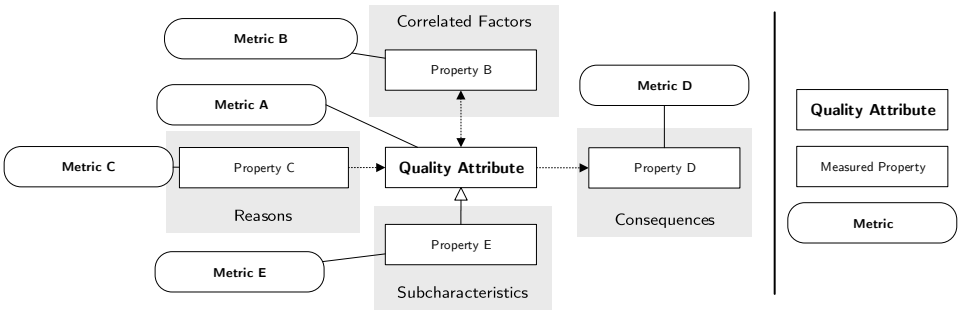


Figure 6.3.: Notation used for presenting the resulting model

6.1.10. Notation for Presenting the Resulting Model

Due to the large number of metrics identified in this study, we are unable to provide an exhaustive graphical presentation of all metrics as done exemplary for the *number of steps of use-cases* metric in Sec. 5.1.5. Instead, we have to resort to a more compact representation and refer to appendix B for details.

The notation used in this section is illustrated in Fig. 6.3. Essentially, rectangles refer to intrinsic properties of the requirements specification, and the one with bold letters in the middle is the quality attribute (see Sec. 4.4) under consideration. For each such attribute, we depict related attributes by rectangles which are connected by a line. Depending on how we classified the attribute, we distinguish between reasons, consequences, correlated factors, or sub-attributes (or characteristics), as is the case for properties **B**–**E**. Finally, we denote metrics as rounded rectangles labelled with its (unique) name. The details of those metrics are found in appendix B, or, for the reader interested in a very detailed classification, in the online companion material[6]. Metrics are associated with the property they actually measure, which may also be the quality attribute itself (e.g., metric **A** in the example).

6.2. Assessment Model: Semantic Quality

In this section, we present the portion of the assessment model concerned with the extent of correspondence between the system described in the specification and an *optimal* system, i.e., the system based on perfect (domain) knowledge, as described in Sec. 2.2.1.

6.2.1. Semantic Correctness

In total, 14 measures were identified for semantic correctness. According to our quality model, semantic correctness is defined with respect to perfect (domain) knowledge as

[6] `http://www4.in.tum.de/~mund/metrics-companion.zip`

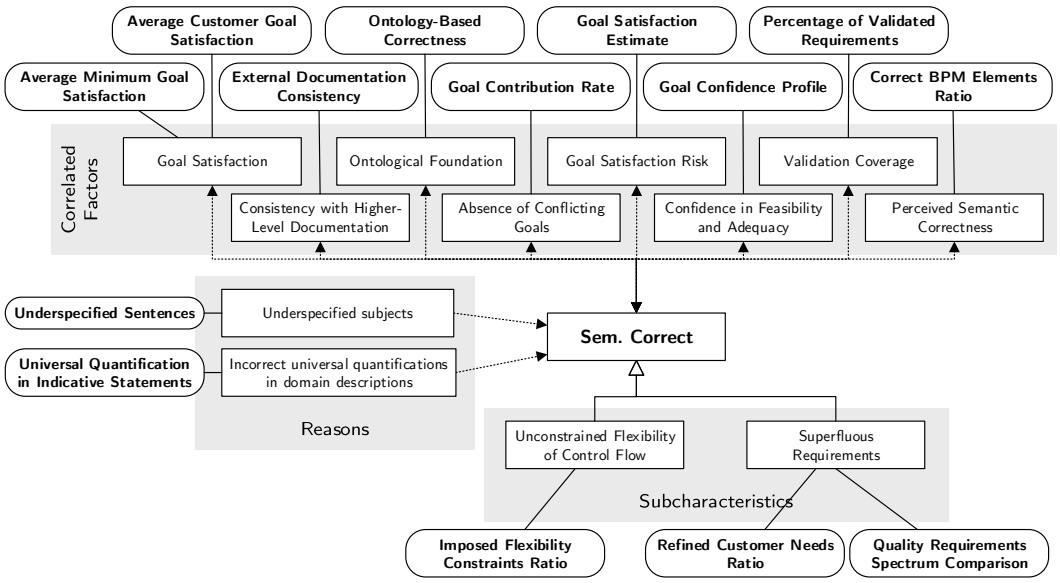


Figure 6.4.: Measures and measured attributes associated with semantic correctness

the point of reference (see Sec. 4.4.1). Consequently, its intangible nature is reflected by the types of measures proposed: No metric measures semantic correctness directly, but instead, the proposed metrics are limited to either certain factors which are argued to correlate with semantic correctness, or, to a lesser extent, potential causes for and special cases of semantic correctness. Fig. 6.4 illustrates the metrics and the associated measured attributes related to semantic correctness.

Sub-characteristics Three measures for two particular aspects of semantic correctness were identified. One such measure was suggested for a specific type of factual error of the requirements specification, namely that the control flow in the business process description is constrained despite no actual constraint in the real world (domain). Overhage et al. [2012] propose to quantify this specific flaw as the percentage of control flow elements which are constrained beyond necessity in the domain, compared to all control flow elements.

Moreover, two measures assess the extent to which unnecessary requirements are specified in the requirements specification. Kaiya and Ohnishi [2011] propose to measure the difference between the number of quality requirements present and demanded of a requirements specification. A surplus of requirements specified in the requirements specification compared to the required quality requirements implies that the requirements specification contains incorrect (contradicting or superfluous) quality requirements and are hence supposed to be incorrect; However, the reverse is not necessarily true: a re-

quirement specification with less than required quality requirements might still contain incorrect requirements. In addition, Kaiya et al. [2002] use a goal-model specified according to a specific meta-model to measure the ratio of requirements which are linked to stakeholder goals. The argument is that such a link is a justification for the requirement, and any requirement not linked to stakeholder goals is not necessary and semantically incorrect.

Reasons Underspecified subjects and incorrect use of universal quantification in domain descriptions were suggested as measurable reasons which (potentially) lead to incorrect requirements. The former is measured as the ratio of sentences for which *subjects contain a word identifying a class of objects without a modifier specifying an instance of such class* to all sentences (metric B.132), while the latter is measured as the number of universality phrases (e.g., *all, each, always, none*) occuring in indicative statements, i.e., statements in descriptions of the domain (metric B.134).

Correlated Factors However, the majority of measures measures aim to approximate semantic correctness by measuring a correlated attribute which is more tangible and explicit compared to perfect domain knowledge.

Several metrics are based on a goal-oriented RE approach [Van Lamsweerde, 2001], in which the stakeholders' high-level goals are captured in an explicit goal model and systematically refined into system requirements. Hence, perfect domain knowledge is approximated by the validity of the stakeholders' goals and its optimal refinement in terms of selecting the set of requirements that best satisfy those goals. Goal satisfaction, i.e., the extent to which the refinement of goals to requires is optimal, is measured by average customer goal satisfaction and average minimum goal satisfaction (metrics B.9 and B.7). The special case to what extent goals are in conflict with each other is measured by the goal contribution rate (metric B.95). The extent to system goals are potentially unsatisfied by the derived requirements is measured by the goal satisfaction estimate (metric B.38). The goal confidence profile measure extends goal models with confidence judgments based on the perceived fulfillment of certain criteria (see metric B.39) in order to measure the requirements' feasibility and adequacy in representing the stakeholders' needs.

In contrast to assess correctness by means of goal models, four alternatives to approximate perfect domain knowledge, and therefore semantic correctness, were identified. The correlation with consistency with respect to relevant external documentation (e.g., high-level requirements, business cases) is based on the assumption that external documentation serves as a point of reference for the requirements specification, and therefore, any deviations to this point of reference are a potential flaw in the specification. Therefore, Davis et al. [1993b] propose to measure the percentage of requirements which are consistent to external documentation (see metric B.32). Kaiya and Saeki [2005] argue that the extent to which the requirements are based on elements from an ontology of the problem domain as perceived by domain experts is related to semantic correctness. The authors regard the ontology *as a semantic basis for a specific domain* and argue

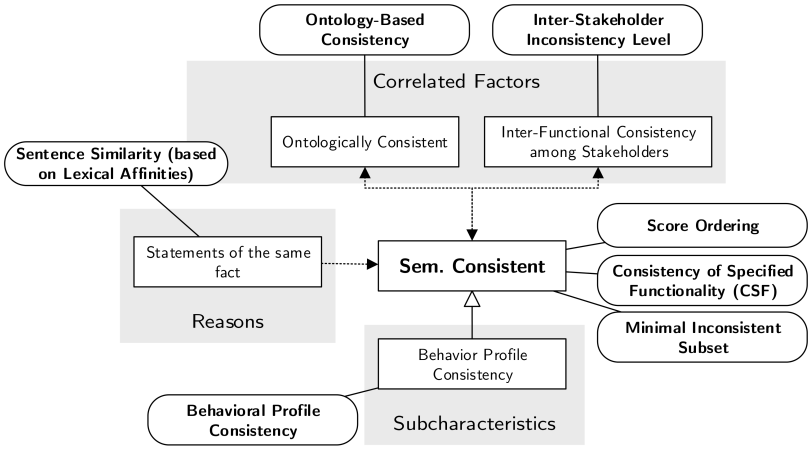


Figure 6.5.: Measures and measured attributes associated with semantic consistency

that *thus all requirements items should correspond to elements in the ontology.* Here, the experts' perception is used as an approximation of the perfect domain knowledge, and requirements which do not refer to ontological elements are classified as potentially incorrect. They propose to measure the ratio of requirements mapped into this ontology to all requirements (metric B.81). Davis et al. [1993a] proposes to measure the ratio of requirements which were validated to all requirements, without sharing details on how validation has to be done (metric B.93). Assuming that validation is done by consulting stakeholders with the requirements, the stakeholders' expectations and domain knowledge serve as the point of reference for semantic correctness. The ratio of incorrect to all elements of a business process model (metric B.27) is a measure of the *perceived* correctness due to a quality manager judging the elements' correctness.

6.2.2. Semantic Consistency

Seven measures are attributed to semantic consistency, three of which directly measure if the requirements specification contains contradicting statements. Mu et al. [2005] propose two measures, namely minimal inconsistent subset and score ordering, which are both based on specifications as logical propositions. The former identifies minimal inconsistent subsets of those propositions and measures their size in terms of cardinality. The latter extends the former by providing an ordering relation among specifications, enabling to assess a specification's semantic consistency against a reference specification. In contrast, Davis et al. [1993a] propose a measure for the semantic consistency of functions provided by the system independent of the language used. It measures the ratio of system functions which according to their specification yield non-deterministic results, i.e., different outputs for the same inputs and system states, to all system functions.

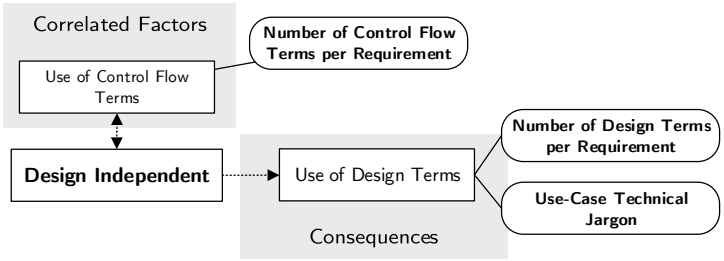


Figure 6.6.: Measures and measured attributes associated with design independence

Sub-characteristic: Behavior Profile Consistency Behavioral profile consistency denotes the extent to which a process specification is free of contradictions regarding the systems' actions. Metric B.12 measures this aspect in case the description of the systems' actions are fragmented across multiple diagrams as the percentage of transitions which are specified to occur in multiple processes but do not contradict.

Reason: Statements of the Same Fact The amount of statements which describe the same fact can be considered as an upper bound for inconsistent statements, since only those statements are candidates for potential inconsistencies. To this end, Kim et al. [1999] propose to measure the lexical similarity between sentences, with the number of pairs of statements which are similar to each other (metric B.122).

Correlated Factors Two measures suggest to measure the specification's consistency by considering external information. Ontology-based consistency (metric B.80) assumes that an ontology is provided (e.g., with help of domain experts) which includes a *contradict* relationship between its elements. Statements of the requirements specification are mapped into this ontology, and semantic consistency of the specification is than judged whether or not a contradict relationship in the ontology is present. In contrast, the inter-stakeholder inconsistency level (metric B.48) measures the consistency among the stakeholders' interpretations of the requirements. Therefore, stakeholders must express their interpretations using temporal logic, and the level of consistency among those interpretation is obtained on a ratio scale which denotes the extent of actual and potential inconsistencies among the stakeholders' interpretations.

6.2.3. Design Independence

Only three measures associated with design independence were proposed, all based on certain terms occurring in a natural-language specification. The extent to which design terms, e.g., *method, parameter, button, web-page*, are present in a requirements specification is measured using a predefined dictionary (metrics B.70 and B.140), which we

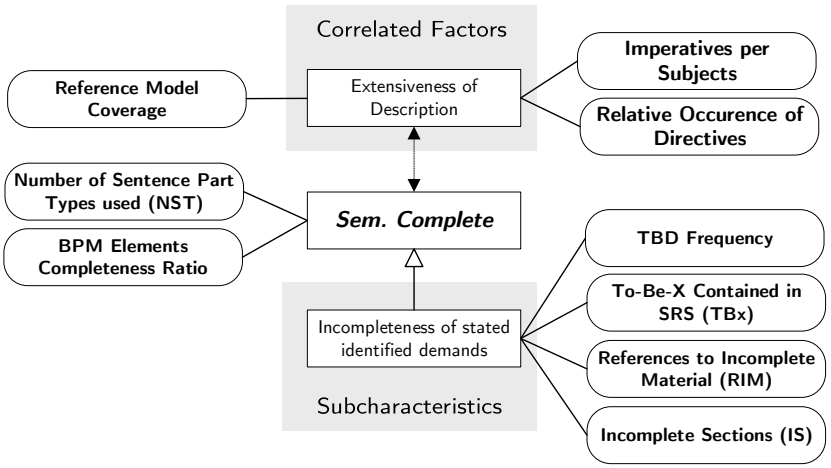


Figure 6.7.: Measures and measured attributes associated with semantic completeness

classified as a consequence of a requirements specification violating design independence. In contrast, the extent to which control flow terms such as *if ... then* and *while* occur was classified as a correlated factor due to its lack for a clear causal relationship to design independence[7]. The associated metric B.69 computes a weighted average of control flow terms compared to all terms. Fig. 6.6 illustrates the measured attributes and the associated metrics.

6.2.4. Semantic Completeness

Originally in the quality definition model (see Sec. 4.4), we intended semantic completeness to be an abstract umbrella term for information and feature completeness. However, some of the metrics identified in our study indeed targeted both notions of completeness, or were so general that they simple could not be associated with neither feature nor information completeness exclusively. Therefore, we decided to associate them with semantic completeness. Pragmatically, any measurement program which aims to measure feature and/or information completeness may also check whether to include semantic completeness metrics.

In total, the study revealed nine metrics, two of which purport to measure semantic completeness directly and holistically: The BPM elements completeness ratio (metric B.13) measures the ratio of specified to all relevant elements in the business process description, based on a manual review by the quality manager. The second measure, the number of sentence part types used (metric B.76), is based on a manual classification of

[7] Control flow terms are common for describing non-technical aspects of the system or black-box behavior, too.

requirements into sentence part types, such as an initiator of an action, an action or a constraint. Overall, Kenett [1996] provide nine such classes, and the metric measures the number of classes for which at least one part of a requirement's description is classified, suggesting that the number of classes is an indicator for the specification's semantic completeness.

Sub-characteristic: Incompleteness of identified demands Four metrics measure the only identified sub-characteristic of semantic completeness, namely the extent to which identified demands are incomplete. Here, identified needs refer to information needs about either system capabilities or characteristic or information required for downstream development to design and implement such a system. In particular, identified needs include that such a need was explicitly stated in some form within the specification.

Since the number of measurable indicators for an explicit statement of incompleteness are sparse, the four metrics identified are quite similar. TBD Frequency (metric B.128) measures the occurrence of "TBD" terms normalized to a specific size metric of the specification, while the number of reference to incomplete material (metric B.103) extends this list by several related terms. Incomplete sections (metric B.44) scans and counts the number of blank or incomplete sections according to a predefined specification template. Finally, the measure To-Be-X contained in requirements specification (metric B.130) counts both the number of empty sections and the number of occurrences of TBx according to a dictionary.

Correlated Factor: Extensiveness of Descriptions Extensiveness of descriptions refers to the extent of information provided in terms of quantity but not the quality. It is measured either using linguistic techniques, i.e., the number of imperatives per subject (metric B.41) or the number of directives (e.g., *figure, table, for example, note*) occurring relative to the specification's size (metric B.110), or with respect to a reference model. Here, the extent to which extensiveness correlates to with semantic completeness is determined by the quality of the reference model. For this reason, reference model extensiveness must be taken with care, and cannot be seen as equivalent to semantic completeness. The associated measure, reference model coverage (metric B.102), is a ratio of reference model elements which are also contained in the specification, weighted by the elements' importance.

Fig. 6.7 gives an overview of the identified measures and their relationship to semantic completeness.

6.2.5. Feature Completeness

Feature completeness, i.e., the extent to which all capabilities and characteristics to satisfy the stakeholders' needs are specified, is among the qualities for which the most measures are proposed. However, only two out of the 20 metrics proposed measure it directly. Local feature completeness (metric B.54) is based on a quality manager's estimation of the number of requirements that are *needed but not (yet) specified* as well

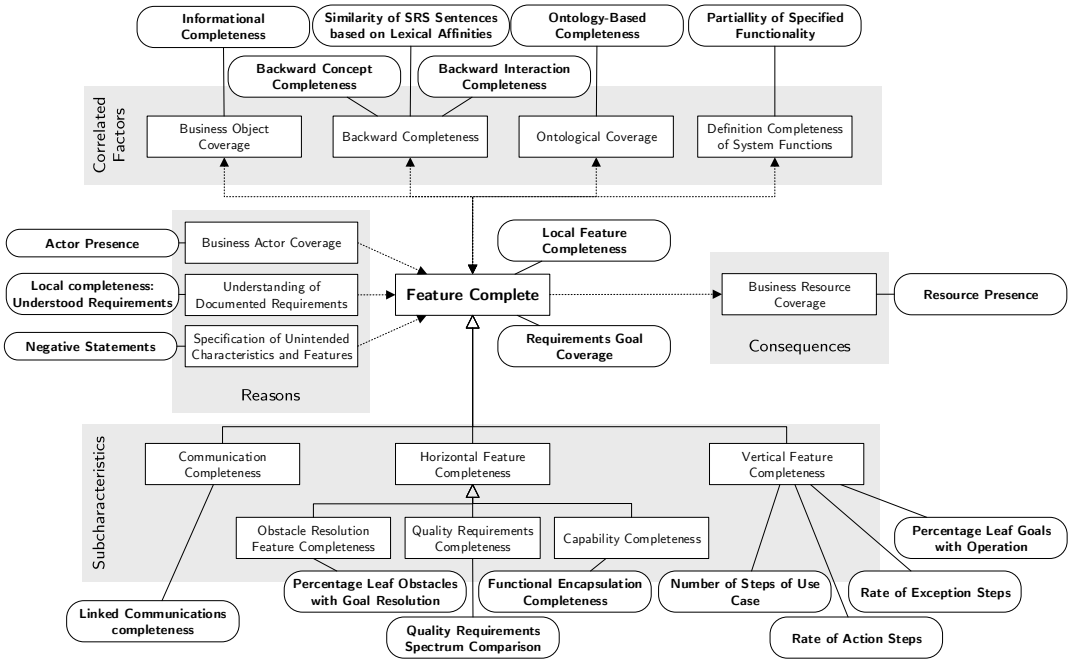


Figure 6.8.: Measures and measured attributes associated with feature completeness

as *potential requirements that are not understood well enough to be documented* in order to relate it to the number of requirements specified. In contrast, requirements goal coverage relies on an explicit goal model. It obtains the ratio of stakeholders' goals which are satisfied by requirements to all goals. Fig. 6.8 illustrates the measures and measured attributes associated with feature completeness.

Sub-characteristics About half the measures identified measure feature completeness limited to a specific aspect. On a higher-level, we identified three sub-characteristics named horizontal feature completeness, vertical feature completeness and communication completeness. Vertical feature completeness denotes the extent to which individual features are specified exhaustively. Here, exhaustively refers to the specification of an individual capability not lacking any behavioral aspects demanded by the stakeholder to be provided of the system, respectively to a specification of an individual characteristics to express the stakeholders' needs extensively. In contrast, horizontal feature completeness denotes the extent to which the set of all specified features are capable to satisfy the stakeholders' needs. In addition, communication completeness denotes the extent to which all communications between the system and external actors (persons, systems) are specified, encompassing both horizontal and vertical feature completeness.

Measures for vertical feature completeness either rely on goal models (Percentage of

requirements (leaf goals) for which an operation is specified, metric B.87) or on use-cases, for which the ratio of action respectively exception steps to all steps (metrics B.98 and B.99) or the plain number of steps (metric B.99) is measured.

Regarding horizontal feature completeness, all three measures identified are further limited to a very specific sub-aspect. The ratio of requirements in a goal model for which obstacles are identified compared to those for which a resolution was proposed

Reasons Three potential reasons are identified, with one measure provided for each. According to Etien and Rolland [2005], business actors should be present in the system in order to trigger state transitions and permit the use of their properties in the system. Otherwise, *some state transitions are not the same in the business and in the system*, resulting in an incomplete specification of the system's features. To this end, actor presence (metric B.2) measures the percentage of actors of the domain model also represented in the system's data model. Furthermore, Davis et al. [1993a] advocate that a thorough understanding of the system's features among stakeholders and requirements engineers is necessary in order to write down features completely. Otherwise, the uncertainty regarding the system's capabilities or characteristics will result in underspecified features. To this end, the understanding of requirements should be measured as the percentage of the identified requirements which are well-understood, i.e., the stakeholders and requirements engineers are certain of, based on subjective assessment. In contrast, Femmer et al. [2014a] suggest to measure a syntactic property, namely the presence of negative statements. Negative statements refer to statements of system capabilities not to be provided, which can lead to underspecification according to the authors.

Consequences Business Resource Coverage, i.e., the extent to which the various business resources associated with the system under development are mapped to system classes, constitutes the sole consequence identified. If numerous business resources are not mapped to the system classes can be interpreted that either some resources play a role in the business which does not need to be known by the system or that the correspondence of some of them in the system is missing. Due to the absence of further arguments provided by Etien and Rolland [2005], we assume that the lack of resources therefore may be either benign or that the requirements specification does not fully specify certain features. Since the metric is not able to discriminate between those cases, we classify the metric as an indicator for a specific aspect regarding the feature completeness of the requirements specification.

Correlated Factors Several measures were proposed for factors correlated with feature completeness, mostly using the same point of reference as discussed for semantic correctness (see Sec. 6.2.1). Three measures used external documentation from up-stream product design as a point of reference for semantic completeness. All three metrics have in common to be based on linguistic techniques applied to natural language specifications and input documentations (e.g., external standards, high-level requirements, transcripts or minutes of meetings), but use slightly different techniques in detail. While backward

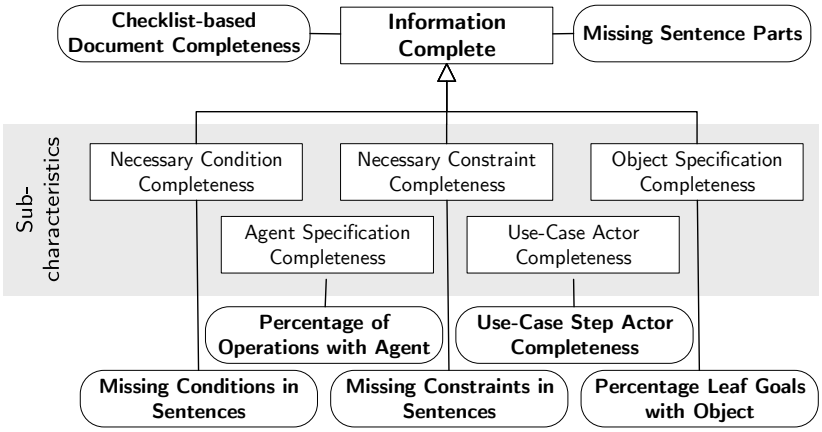


Figure 6.9.: Measures and measured attributes associated with information completeness

concept and interaction completeness provide a rational scale which denotes the extent to which central terms (regarding the mentioned concepts respectively its co-occurrence in the document) in the input documents are also included in the requirements specification, similarity of requirements specification sentences based on lexical affinities provides an absolute scale in the number of sentences for which a relationship to an external documentation was found.

Informational (metric B.45) and ontology-based completeness (metric B.79) both rely on a model-based comparison to an external reference. To this end, the former measures the percentage of business objects specified in the domain model which are also represented in the data model, while the latter measures the percentage of concepts of a domain ontology constructed by domain experts which are represented in the requirements.

One factor, namely the definition completeness of system functions, has no counterpart in semantic correctness. Here, the mathematical idea of a complete definition is measured, i.e., the extent to which a user-visible function indeed specifies an output for each possible input and (conceptual) system state. Davis et al. [1993a] propose to measure the percentage of partially defined functions to all functions. It must be noted that a partial function is neither strictly feature incomplete (since for some inputs and states, the stakeholders' satisfaction is indifferent regarding the output) nor is a fully-defined function necessary feature complete (since it still may lack inputs and/or conceptual states).

6.2.6. Information Completeness

Information completeness refers to the extent to which all information necessary for subsequent development activities are specified. Most measures focus on a very specific

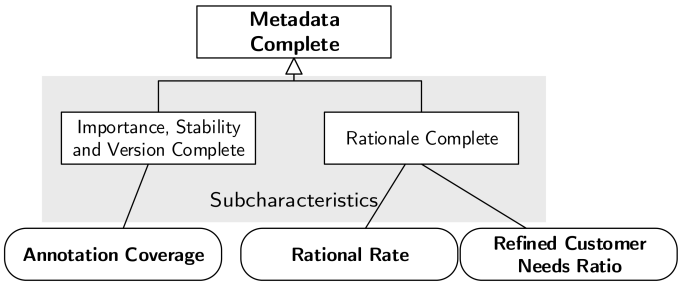


Figure 6.10.: Measures and measured attributes associated with metadata completeness

information, however, two metrics aim to assess information completeness more generally (see Fig. 6.9). Kenett [1996] classify the requirements' description according to attribute classes (e.g., a constraint or a condition). Based on this classification, a manual review for missing information according to the attribute classes is conducted, and the number of missing sentence parts relative to all specified attributes is measured. Furthermore, Matulevicius et al. [2010] suggests a checklist-based approach, defining the necessary information in a checklist against which the specification is assessed. More specifically, general presence of the necessary information and its level of detail (on a three-level ordinal scale) is aggregated to provide a measure for information completeness.

Sub-characteristics Several authors proposed a specific information necessary for downstream development together with the appropriate measure. Specifically, initiator of actions, conditions and constraints must be specified for features where necessary (metrics B.90 (metric B.138 for the special case of use-cases), B.59 and B.60, respectively), and system objects must be specified for any requirement in a goal-model (metric B.86). All measures provide a result normalized by the specification's size, except use-case actor completeness, which returns the number of use-cases for which at least one step does not provide an actor.

6.2.7. Meta-data Completeness

We identified three metrics measuring specific meta-data information, as illustrated in Fig. 6.10.

Sub-characteristics The identified metrics aim to assess the extent to which importance, stability, version, and rationale are provided for requirements. Regarding the first three attributes, Annotation Coverage (metric B.6) measures the percentage of requirement for which all those attributes are specified. Given a goal-based requirements approach (see, e.g., Van Lamsweerde [2001]), Kaiya et al. [2002] suggest to either measure rationale completeness explicitly as the percentage of all elements for which a rationale

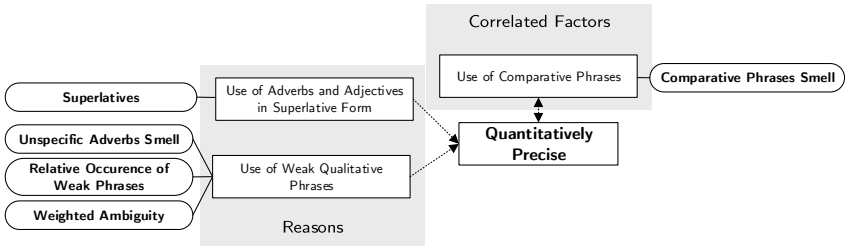


Figure 6.11.: Measures and measured attributes associated with quantitative precision

is provided (metric B.18), or implicitly as the percentage of requirements linked to stakeholder goals to all specified requirements (metric B.104).

6.2.8. Quantitative Precision

We identified five metrics related to quantitative precision as illustrated in Fig. 6.11. All measures apply to natural-language specifications and are at least partly based on the identification of certain phrases. Here, we distinguished between phrases which represent a qualitative statement and those which represent a comparison between the system under development and a reference system. Measures for the former phrases were classified as reasons due to a substantial potential in leading to an quantitative imprecise specification, while the latter was classified as a correlated factor due to a very vague relationship between the measured quality and quantitative precision.

Reasons We identified two classes of phrases which can be the source for an imprecise specification. First, three measures were identified for the class of *unspecific* or *weak* phrases (e.g., *adequate, normal, timely, easy*). To this end, such phrases are identified based on a pre-defined dictionary and counted (unspecific adverbs smell B.135 and additionally normalized by the specification's number of lines (relative occurrence of weak phrases B.111). In contrast to those measures based purely on automated linguistic techniques, Kim et al. [1999] propose to include manual reviews. Weighted ambiguity (metric B.146) measures the number of weak words which are indeed result in a quantitatively imprecise specification according to a manual review by the quality manager. The measure also feeds those judgments back into the detection algorithm to reduce spurious suggestions by associating weights to certain phrases, and therefore reducing the need for manual reviews over time. Furthermore, the number of superlatives, i.e., adverbs and adjectives in superlative form, are measured using the same dictionary-based techniques (metric B.127).

Correlated Factor: Use of Comparatives Phrases According to Femmer et al. [2014a], comparative phrases refer to a comparison of the system under development to

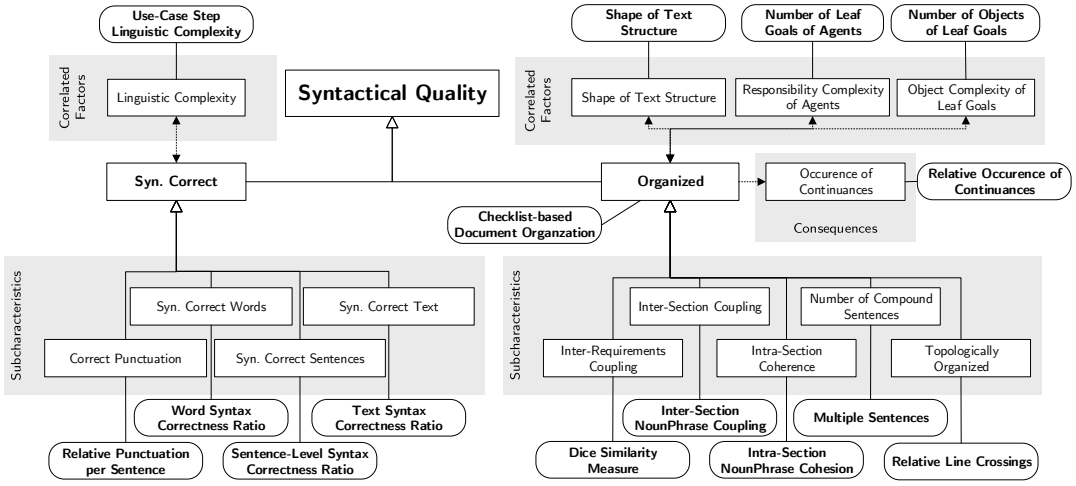


Figure 6.12.: Measures and measured attributes associated with syntactic quality

some other system, e.g., by a competitor or a legacy system. The authors argument that, depending on the information provided for the reference system respectively the extent to which quantitative information is observable, the specification may be seen as (quantitatively) imprecise. The corresponding measure (metric B.19) is the number of such comparisons identified using linguistic techniques.

6.3. Assessment Model: Syntactic Quality

In this section, we present the portion of the assessment model concerned with the extent the specification is syntactically correct and well organized.

6.3.1. Syntactic Correctness

Out of five measures identified for syntactic correctness, four measured a sub-characteristic and one a correlated factor as illustrated in the left part of Fig. 6.12.

Sub-characteristics Génova et al. [2013] propose to approximate correct usage of punctuation by measuring the punctuation signs mean per sentence, arguing that a specification with too many or too few punctuation signs indicate violations of grammatical rules. For word, sentence and text-level syntactic correctness, Overhage et al. [2012] provide three measures based on a manual review. The measurement of lexical correctness is based on a dictionary the number of words which are correct with respect to a dictionary are counted and divided by the number of all words (metric B.147). Both, sentence and text-level correctness is based on an implicit or explicit set of grammatical

rules. The extent to which the individual sentences or a text formed therefrom are syntactically correct is measured as the ratio of correct applications to all applications of the corresponding sentence-level respectively text-level grammatical rules (metrics B.118 and B.129).

Correlated Factor: Structural Complexity According to Ciemniewska et al. [2007], the authors suggest to measure the complexity of a sentence used for describing each step of a use case specifications, since *each step of a use case should be as simple as possible [because] with such a simple structure, one can be sure that the step is grammatically correct.* The corresponding measure (use-case step linguistic complexity B.139) counts the number of *complex* steps. Here, a step is considered complex if and only if more than one sentence, subject, coordinate or subordinate clause is provided.

6.3.2. Organized

The syntactic quality of organized denotes the extent to which a specification's contents are arranged in a sensible way, for which we identified ten measures as illustrated in the left part of Fig. 6.12. Checklist-based document organization (metric B.17) constitutes the sole metric which addresses multiple aspects of this quality. Essentially, its comprehensiveness stems from the fact the contents of the measure rely on the checklist covering all relevant aspects the quality. Matulevicius et al. [2010] provide a list of 13 questions as a starting point, which tackles various aspects of the specification being organized, and suggest the ratio between criteria (questions) fulfilled by the specification to all criteria applicable as the corresponding measure.

Sub-characteristics Several aspects of arranging the specification's contents in what is considered a *sensible way* were identified. Regarding the organization of the specifications in terms of sections, two measurable characteristics were proposed by Rine and Fraga [2015], Din [2008]. According to the authors, a specification is sensibly organized into sections if semantic coherence within each section is maximized and coupling (in terms of implicit or explicit cross-references) between different sections is minimized. To this end, two measures based on noun-phrase chunks, i.e., text chunks which form a noun-phrase, are proposed. Coherence (metric B.49) is measured as the extent to which adjacent sentences within sections share noun-phrase chunks on a ratio scale. Coupling (metric B.46) is measured as the extent to which the sections' noun-phrases are also contained in other sections on a ratio scale, also taking into account the distance between those references. A closely related characteristic was proposed by och Dag et al. [2001a]. Here, the authors aim to identify implicit coupling among requirements using a linguistic similarity measure (metric B.31). A specification is more sensibly organized if there are less dependencies among requirements, which, however, is limited due to the domain's complexity in practice. Fabbrini et al. [2000] take on a quite fine-grained view on the specification's organization. The authors propose to measure the number of sentences (metric B.62) which contain more than one subject or main verb in order to measure the

extent to which sentences refrain from specifying exactly one fact.

In contrast to natural-language, topological organization applies to diagrams only One aspect of it being sensibly organized regarding its topology is the extent to which line crossings occur, since crossings are commonly supposed to limit model readability. Cherfi et al. [2002, 2007] suggest to measure the topological organization as the number of line crossings occurring for a specific diagram normalized by the number of potential line crossings for a diagram.

Consequence: Use of Continuances According to Wilson et al. [1997], for a well structured natural-language specification, continuance phrases, e.g., *below:, as follows:, following:, in particular:* are expected to occur more often compared to a specification which lacks organization. Therefore, the authors suggest to measure the number of occurrences of such phrases in a specification using a dictionary-based approach, and normalize the obtained count by the specification's size (metric B.108).

Correlated Factors For goal-oriented requirements engineering, Espada et al. [2011, 2013] provide two complexity measures based on the number of associated elements. Specifically, the number of leaf goals (i.e., requirements) associated with each agent in a goal-model are counted (metric B.72). Similarly, the number of objects associated with each leaf goal is counted (metric B.74). Essentially, both metrics approximate the complexity of the agents respectively the requirements in a goal-model based on relationship size. The author's argument is that the measure provides an upper bound on the accidental complexity of the specification, which in turn indicates that the specification is not organized to its best. To this end, the extent to which the responsibility for goal objects, which are in turn allocated to individual agents, is decomposed in a goal-model. According to the authors, a lack of decomposition is interpreted as general complexity (i.e., intrinsic and accidental) and hence regarded as a lack of organization in the specification's goal model. Therefore, argent responsibility decomposition is an indicator for the extent a goal-model is organized regarding a certain aspect, namely, the structural organization of the agents, if the complexity is accidental.

Furthermore, Wilson et al. [1997] suggest that the shape of the text structure in a natural-specification organized in hierarchical sections indicates the extent to which the requirements specification is organized in a sensible way (metric B.120). To this end, the number of statement identifiers for each section are counted. By adding up all such identifiers for each section at the same depth in the specification's hierarchical organization, the shape is obtained in terms of a mapping from the specification's depth-level to the number of identifiers identified at that particular level. The authors claim that any other than a pyramidal shape, i.e., the number of statements increase monotonically with the the depth level, indicates that the specification is not well organized.

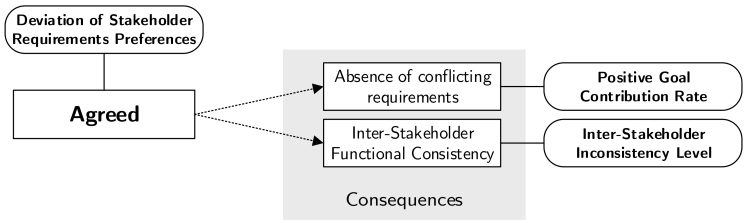


Figure 6.13.: Measures and measured attributes associated with social quality

6.4. Assessment Model: Social Quality

The extent to which the stakeholders agree on the specified requirements constitutes the sole quality attribute of the specification's social quality. The present study identified two measures extend the quality model as illustrated in Fig. 6.13.

Kaiya et al. [2002] proposed to measure agreement by directly involving all stakeholders; the authors suggest to ask them to rate their preference for each requirement on a scale of $[-10; 10]$. In addition, they were asked to also estimate the other stakeholders' preferences on the same scale. Based on those ratings, the measure provides a quantitative assessment of the extent the stakeholders agreement differs. Although originally proposed in the context of goal-oriented requirements engineering, the deviation of stakeholder-preferences measure (metric B.95)

Consequence: Freedom from Goal Conflicts One measured attribute, namely the extent to which the specification is free of requirements which conflict with stakeholder goals, was identified as a consequence of agreement. To this end, Kaiya et al. [2002] measure the percentage of goal refinements, i.e., edges along paths in goal trees, which positively contribute to the stakeholder goals. The less requirements conflict with the stakeholders' goals, the more likely the stakeholders agree on them. However, note that stakeholders may still differ in the degree of acceptance or refusal of any requirements.

6.5. Assessment Model: Pragmatic Quality

Extending the quality model regarding pragmatic quality proved the most challenging. Recall that pragmatic quality is the extent to which the audience understands the specification correctly and completely, as described in Sec. 2.2. Therefore, unlike semantic and syntactic quality, pragmatic quality depends on the actual individuals working with the requirements specification. Consequently, the quality definition model as described in Chap. 4 seeked to identify those inherent properties of the requirements specification which may impact the understanding of the audience.

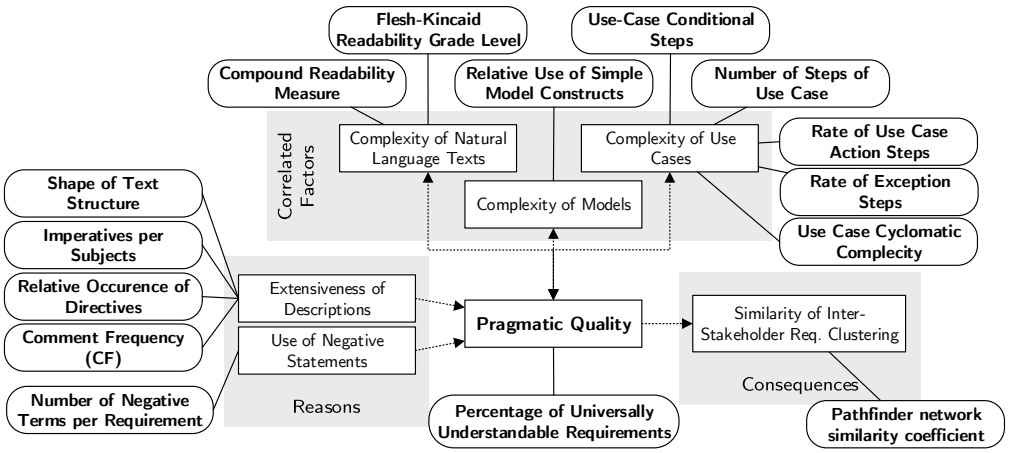


Figure 6.14.: Measures and measured attributes associated with pragmatic quality in general

Reasons Several authors measure how extensive the requirements are described in the specification in terms of the *amount* but not necessary the *quality* of provided information. Extensiveness can indeed improve pragmatic quality; On the one hand, examples and redundant but different formulations of requirements can indeed make the readers' interpretations more complete and less ambiguous. However, beyond a certain value, the effect can change (due to a trade-off between the extensiveness of description and conciseness of the requirements specification). Certain aspects of the specifications' extensiveness are measured. The shape of text structure (metric B.120) can reveal that certain parts of the specification are specified extensively, e.g., that a large amount of introductory and administrative information are contained. The ratio of imperative phrases per unique subjects is proposed by Wilson et al. [1997] to measure how extensive each functional requirement is described. For the whole specification, the same authors suggest to measure the number of directive phrases (e.g., *figure, table, for example:*) normalized by the specification's size in terms of lines of text. Finally, Fabbrini et al. [2000], Fantechi et al. [2002] suggest to measure the percentage of requirements for which at least one comment was specified, therefore depending on a designated syntactic element for comments.

Besides extensiveness, Femmer et al. [2014c] argument that negative terms make "the sentence more difficult to understand" and hence "affect specifically the desirable property of understandability". To this end, the authors' natural-language analysis yields the occurrence count of negative terms based on a dictionary.

Consequence: Requirements Clustering Similarity Kudikyala and Vaughn [2005] propose to measure the extent to which stakeholders and developers cluster requirements

according to semantic coherence similarly. According to the authors, the clustering of requirements is based on the "psychological distance" between requirementsm which in turn relies on "similarity judgments" obtained independently from the stakeholders and developers. Those clusters are expressed as Pathfinder networks, which can be used to "represent certain aspect of the human semantic memory [...] to analyze, understand and categorize software requirements in the requirements analysis phase" and represent the "mental models of software developers and users", which are used to "identify ambiguous, and misunderstood requirements".

Correlated Factors: Complexity All correlated factors identified in the study are concerned with the specification's complexity. The common argument is that a more complex specification is harder to understand correctly and holistically by the audience. To this end, measures have been proposed for three forms of representations, namely models, use-cases and natural-language in general. Again, we distinguish between intrinsic and accidental complexity, with the goal to reduce the latter to improve the quality of the specification.

Various metrics were proposed for use-case specifications. The arguably most basic measure is the number of steps a use-cases contains (metric B.78). Other measures suggested certain types of use-case steps to introduce or indicate complexity, such as conditional steps (metric B.136), exceptions from the main flow (metric B.99) or any step containing an action from the user (metric B.100). Finally, Duran et al. [2002] suggested to adopt the source-code based cyclomatic complexity metric by McCabe [1976] to use-cases by measuring the number of alternative paths of a use-case.

To determine the complexity of models in the specification, Cherfi et al. [2002] propose to measure the share of *simple* model constructs with respect to all model constructs used. Specifically, the authors aim to measure the complexity of conceptual diagrams which model entities and their relationships (including inheritances). In this particular case, any entity was considered simple in contrast to any kind of relationship, i.e., N-ary or inheritance.

Finally, several researchers [Kenett, 1996, Wilson et al., 1997, Fabbrini et al., 2000, Génova et al., 2013] suggest to apply well-known readability measures such as the Flesh-Kincaid Readability Grade Level (metric B.33) also to software specifications. Essentially, the measure determines a rational number based on the the length of sentences, words and syllables of the text. However, as pointed out by Génova et al. [2013], this measure, in particular the prescriptive reference values, must be taken with a grain of salt due to the nature of technical texts with the frequent but acceptable use of long words.

6.5.1. Conciseness

In total, we identified ten measures associated with the specification's conciseness, as illustrated in Fig. 6.15.

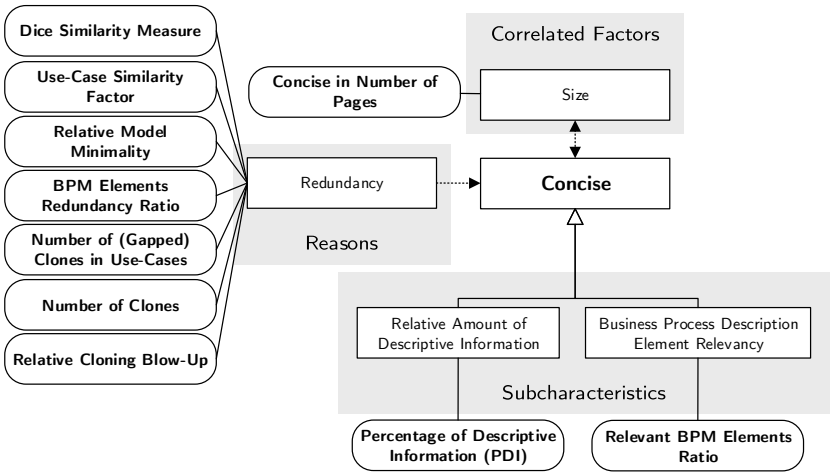


Figure 6.15.: Measures and measured attributes associated with conciseness

Sub-characteristics Kenett [1996] suggest to measure the amount of descriptive information in relation to the both descriptive and prescriptive information contained in the requirements specification. To this end, the authors propose to measure the percentage of sentences judged as descriptive based on manual reviews (metric B.89). Here, the argument is that while prescriptive information is always necessary in a requirements specification, descriptive information might not be so. Its purpose is to help the reader of the requirements specification understand the system under consideration or its context, at the expanse of a larger requirements specification. Hence, such descriptive information is superfluous in case the reader already has a precise understanding from the prescriptive information in the requirements specification alone. For business process models, Overhage et al. [2012] measure the extent to which the elements of the business process description are relevant for the requirements specification audience, by classifying each model element as relevant if and only if its removal would result in a loss of information for the specification's audience. Note that both measures have in common to measure the extent to which a particular information which is not absolutely necessary is contained in the specification, but neglect other aspects of conciseness, e.g., to what extent the highest density syntax has been used.

Reason: Redundancy We identified several measures essentially concerned with the specification's redundancy. Strictly speaking, redundancy, i.e., the multiple presence of the same information, is not the same as concise, since it can also have positive effects on certain quality characteristics and therefore contradicts the definition of concise (see Sec. 4.4). Therefore, a concise specification may include redundant information, and freedom of redundancy does not make a specification necessarily concise. However,

unnecessary redundancy results in a specification becoming less concise.

Two measures suggest to assess the extent to which requirements are duplicated. For natural-language specifications, the Dice similarity measure (metric B.31), a well-known similarity coefficient from linguistics, is used to approximate duplicated requirements if pair-wise similarity exceeds a given threshold. For use-cases, Ciemniewska et al. [2007] suggested the use-case similarity factor, which essentially measures the use-cases which are considered highly similar. To this end, the authors propose to compute a *signature* of each use-case, which basically represents the actors and the invoked actions of the use-case based on linguistic techniques (metric B.137).

Besides duplicated requirements, most of the proposed measures assess redundancy at the statements level, e.g., sentences for natural-language specifications or modeling elements for models, respectively. For natural-language specifications, clone detection was applied by Juergens et al. [2010] and Rago et al. [2014] to requirement specifications. Clone detection subsumes techniques to identify parts of an artifact which occur several times, either in identical form or with additional information added within such a part (called *gapped* clone). For general requirement specifications, Juergens et al. [2010] propose to measure the number of clones (metric B.67) and/or the extent to which an requirements specification is enlarged due to clones called relative cloning blow-up (metric B.105) based on a natural-language processing techniques combined with manual reviews to improve precision. Specifically for use-case based specifications, Rago et al. [2014] propose to measure the number of (gapped) clones in use-cases by leveraging machine-learning to classify actions within use-case according to a predefined taxonomy, which is in turn used to identify duplicated sequences of actions (metric B.64).

For conceptual models, Cherfi et al. [2002] suggest to measure the extent to which elements are duplicated, depending on its importance. Therefore, the authors to propose to measure the percentage of model elements which for which an equally-named element exists, modified by predefined weights according to the element's class (metric B.107). In contrast, Overhage et al. [2012] suggest to use manual reviews for each model element instead (metric B.14), basically obtaining a very similar measure for business process models.

Correlated Factor: Size Naturally, size in terms of amount of content is closely related to conciseness. According to our definition (Sec. 4.4), size can be understood as a kind of simplified perspective on conciseness which neglects the semantic content and its quality characteristics. Therefore, a size measure to be applicable with respect to conciseness must provide reference values which reflect the required amount of information.

To this end, Davis et al. [1993a] and Wilson et al. [1997] propose to measure the size of a specification in terms of the number of pages (metric B.25), with the results interpreted against an interval which indicates whether the size is *healthy*, i.e., the specification is neither below nor above the given thresholds and therefore contains unnecessary information (or, if below the threshold, lacks important information).

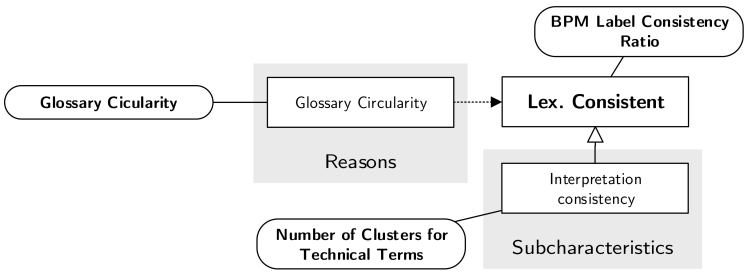


Figure 6.16.: Measures and measured attributes associated with lexical consistency

6.5.2. Lexical Consistency

Fig. 6.16 illustrates the measures associated with lexical consistency as identified in this study. The only direct measure of lexical consistency was proposed by Overhage et al. [2012]. For a given business process model, the authors suggest to measure the percentage of model elements which are judged to be labeled consistently by a manual review, compared to all model elements (metric B.15).

Sub-characteristic: Interpretation Consistency Recall that according to the quality definitions presented in Sec. 4.4, lexical consistency entails both that one concept is referred to only by one single term, and that multiple occurrences of the same term should not refer to different concepts. The latter is precisely what is referred to by the notion of interpretation consistency. Matsuoka and Lepage [2011] measure interpretation consistency by the number of different interpretations for technical terms occurring the requirements specification, based on an automatic analysis using linguistic techniques (metric B.68).

Reason: Glossary Circularity A measured attribute classified as a reason is glossary circularity, i.e., the extent to which definitions of terms are based on different terms also contained in the glossary. Duran et al. [2002] motivate the measurement of glossary circularity by referring to Leite et al. 1997. In that paper, Leite et al. argument that glossary circularity promotes a self-contained glossary which is responsible for making the requirements specification lexically consistent and, in turn, less ambiguous. Therefore, the authors propose to (automatically) measure the number of references to other terms within each glossary item and map them to a rational scale which denotes the extent to which definitions of terms are based on other terms also contained in the specification (metric B.37). The authors claim that on average, every glossary entry should reference ≥ 2 other glossary terms.

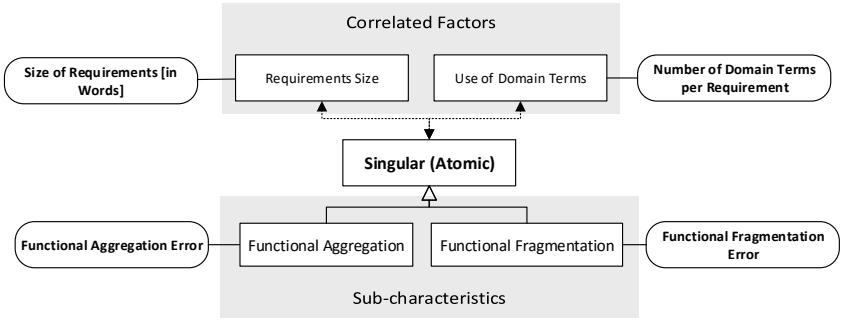


Figure 6.17.: Measures and measured attributes associated with singularity (atomicity)

6.5.3. Singularity (Atomicity)

A requirements specification is singular, also called atomic, if every requirement therein cannot be usefully separated into several requirements (see Sec. 4.4). For this quality, the present study identified four metrics, which measure attributes which are either correlated with or a consequence of the actual quality, as illustrated in Fig. 6.17.

Sub-characteristics Espana et al. [2009] suggest to measure atomicity violations by focusing on two aspects called functional fragmentation respectively aggregation errors [Wieringa, 1996]. The former denotes that the system's functionality is perceived to be wrongly split into multiple requirements, while the latter denotes that several atomic functions are wrongly specified as one requirement. Therefore, the authors assume that unity criteria are defined and propose that an expert committee determines such errors for a given specification, effectively counting the number of violations and interpreting the results with respect to the specification's atomicity (metrics B.36 and B.34).

Correlated Factors Génova et al. [2013] present two measures which correlate with singularity: the size of and the extent domain terms are used for each requirement of the specification. According to the authors, a larger requirement is more likely to be split into multiple requirements, hence violating singularity. The authors state that an "adequate size of requirement is neither too big nor too small [and] directly affects the desirable property of atomicity", defined as "not mixed with other requirements". To this end, the authors propose to measure the number of words for each requirement in the specification, and compare the size to prescriptive reference values which determine whether a requirement is singular on an ordinal scale with three levels denoting the probability (metric B.123). Furthermore, "the number of domain terms [...] should [not] be too high (which would indicate a loss of atomicity)" for individual requirements, and propose to measure its number based on linguistic techniques and a pre-defined dictionary (metric B.71). Again, the resulting number of domain terms is compared to

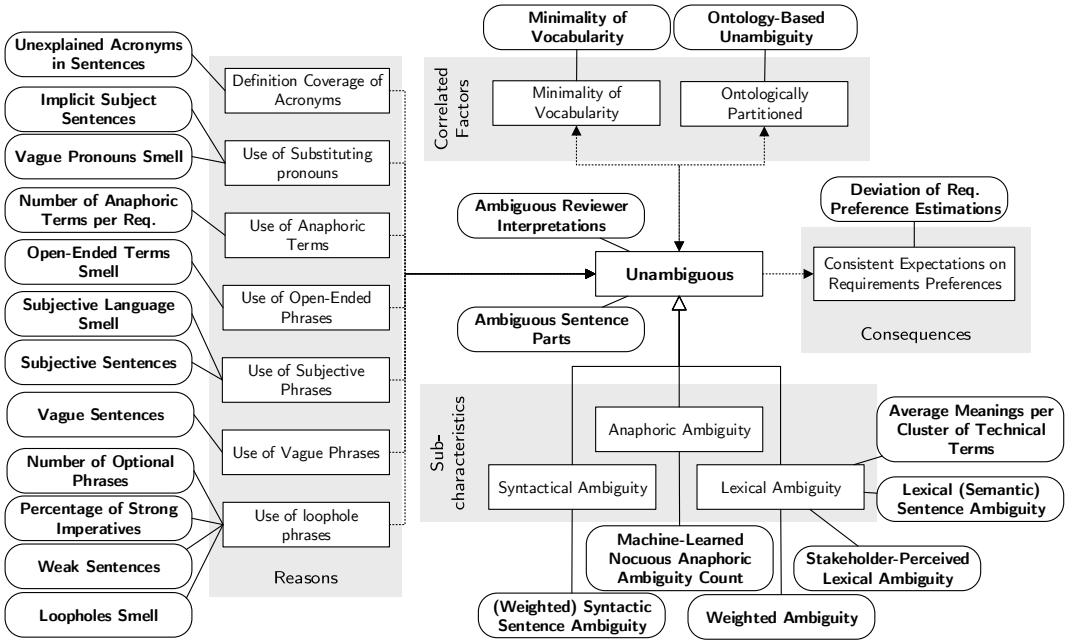


Figure 6.18.: Measures and measured attributes associated with unambiguity

prescriptive reference values suggested by the authors top obtain an interpretation on a three level ordinal scale. Finally, for both measures, the assessments of the individual requirements are aggregated for the whole specification using a scoring function based on arithmetic means of the individual interpretations.

6.5.4. Unambiguity

Unambiguity was and still is a vivid field of research in requirements engineering, resulting in the the most (23) measures identified for any quality attribute (Fig. 6.18). Thereof, two measures were based on manual judgments by the audience: Davis et al. [1993a] propose to measure the percentage of requirements which are ambiguous by comparing the interpretations of multiple readers for each requirement (metric B.4), while Kenett [1996] suggest to first decompose and extract requirement statements into so-called attributes (e.g., the actor or the action) and obtain the percentage of ambiguous attributes, again by manual reviews.

Sub-characteristics Six measures tackle certain linguistic subtypes of ambiguity. Syntactic ambiguity is the the extent to which a sentence can have multiple syntax trees without considering context. To measure this quality, Kiyavitskaya et al. [2008] proposed a tool called LOLITA which counts the number of potential syntax trees of a sentence and

applies weights to this count by considering the severity if grammar rules are violated within the sentence (metric B.1). Yang et al. [2011] assess anaphoric ambiguity, i.e., the extent to which the pronouns in a specification are interpreted with respect to the same object (antecedent) by different readers. Therefore, the authors advocate a machine-learning approach: Based on a natural-language processing tool, human judgments on the most likely pair of pronouns and its antecedents are collected, which are then input as a training set to a machine-learning tool in order to predict and count the pronouns indeed associated with multiple interpretations in the given specification (metric B.56). Last but not least, four measures were proposed for lexical ambiguity, i.e., the extent to which individual words are associated with multiple interpretations (metrics B.146, B.124, B.50 and B.8).

Reasons Several, quite different, properties of the specification were proposed to influence its ambiguity. Most of those properties investigate classes of phrases for natural-language specifications which can result in ambiguities. On the rather general side, the use of substituting pronouns (e.g., metric B.42 and B.142) or anaphoric terms (e.g., B.66) is assessed as the reader may be unable to resolve them at all or reference to a term not intended by the requirements engineer. Furthermore, specific phrases are advocated by the respective authors to potentially cause ambiguity. Those terms are classified broadly as open-ended (e.g., metric B.83), vague (e.g., metric B.143), allowing loop-holes (e.g., metrics B.75, B.91, B.144 and B.55) or subjective (e.g., metrics B.125 and B.126). All the aforementioned measures are applicable to natural-language specifications using linguistic techniques, mostly based on dictionaries, and yield either an absolute or normalized count of potentially harmful phrases. Again, the interested reader should consult appendix B for more details.

In contrast to specific phrases, Fabbrini et al. [2000] argue that unexplained acronyms in the specification are a source of ambiguity if the audience is not aware of them, and consequently propose to measure the number of such acronyms occurring in the specification seemingly based on manual reviews (metric B.133).

Consequence: Consistent Expectations on Requirements Preferences Kaiya et al. [2002] suggest that all stakeholders rate, based on the requirements specification, each others preference per requirement on a scale from -10 (complete rejection) to $+10$ (complete approval). Based on those ratings, the average deviation between those ratings is measured (metric B.28), therefore expressing the extent to which the preference estimations coincide. According to the authors, the extent to which the specification is ambiguous should be reflected in this measure since ambiguous requirements may lead to different estimated preferences by the various stakeholders, and, in turn, in a higher average deviation.

Correlated Factors According to Duran et al. [2002], ambiguity correlates with the extent to which glossary terms are used in the specification of the requirements, since those glossary terms represent "the vocabulary of the customer". Therefore, the authors

suggest to measure the average number of references to glossary items per requirement, based on an automated counting procedure due to a reference meta-model for the specification that must be used (metric B.58).

In addition, Kaiya and Saeki [2005] assume the presence of a domain ontology for which the "requirements analyst achieves [a] mapping from requirements items (statements) in a requirements specification to elements in an ontology" which in turn "has to have atomic concepts that are interpreted in the same way by any stakeholder in a specific application domain". Based on this ontology, the authors state that "when a requirements item is mapped onto several elements that are not semantically related, the item is regarded as ambiguous". Since no further information is disclosed in the paper, we can only presume that the authors intent the mapping to resembles *all* associations of stakeholders of requirement terms with ontological elements. Hence, the number of ontological partitions (weakly connected components in the ontological graph) denotes the number of different (semantically unrelated) interpretations. Therefore, a requirement with more than one ontological partition indeed associates two meanings to a term in the requirements specification if the ontology (especially its relationships) is complete. However, according to our understanding, a single ontological partition does not necessary imply that the requirement is free of contradictions. The associated metric measures the percentage of terms in the requirements specification mapped into the ontology for which all ontology mappings are (semantically) connected (metric B.82).

6.6. Assessment Model: Compound Metrics

In addition to the metrics presented so far, the study identified three compound metrics proposed by Kenett [1996] which combined multiple quality dimensions according to the quality framework (see Chap. 2 and Lindland et al. [1994]), namely pragmatic and semantic quality (see Fig. 6.19). The completeness and ambiguity of object flow metric describes the extent to which the flow of objects in a natural-language specification is described both completely and unambiguously based on manual reviews (metric B.21). This measure itself is input to the compound completeness measure (metric B.23), which is obtained as the weighted sum of measurement values of the aforementioned metric and semantic quality measures (metrics B.61, B.76 and B.128) as well as a pragmatic quality measure (metric B.5). Furthermore, the compound accuracy measure (metric B.22) is obtained as the weighted sum of measurement values regarding semantic quality (metrics B.128 and B.59) as well as pragmatic quality (metric B.5).

6.7. Measured Scope of Specification

Complementary to the quality mapping perspective in the previous sections, in this section, we provide details about the relationship of the measures represented in the quality assessment model and its primary study object, i.e., the measured scope of the requirements specification. As described in Sec. 6.1, we distinguish between the *content* scope, i.e., the precise content items which are measured (as defined by the AMDiRE artifact

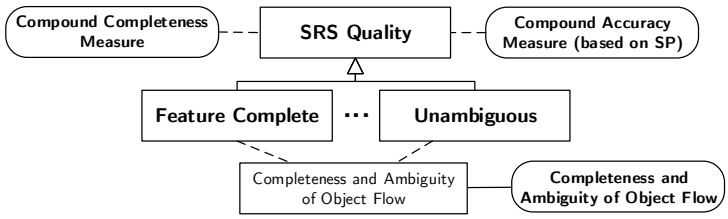


Figure 6.19.: Compound Measures

Artifact / Level of Perception	**Semantic**	**Syntactic**	**Physical**	**Total**
Independent	11	23	1	35
Context Layer	9	23	-	32
Requirements Layer	23	46	1	70
Total	43	91	2	136

Table 6.5.: Number of metrics per measured content items (independent, context-layer, requirements layer) and semiotic scope (semantic content, syntactic structure, physical representation)

model [Méndez Fernández and Penzenstadler, 2014]), and the *semiotic* scope, i.e., the level of perception on which the measurement procedure collects data (see Sec. 5.1.2 for details). Tbl. 6.5 provides an overview on the classification of the metrics' measurement procedures according to the aforementioned scopes.

6.7.1. Content-Item Independent Metrics

Many measures identified in the study are not limited to specific content items, such as the usage model or the domain description, but explicitly study the whole specification on purpose, e.g., to obtain the percentage of descriptive information about the specification (metric B.89), or are applicable to any part of the specification as long as the measure's prerequisites are fulfilled (see Sec. 6.8, such as the linguistic ambiguity measure (metric B.146).

The categorization results for content-item independent measures are illustrated in Tab. 6.7.1. Notably, all but one metric use measurement procedures which study a semantic or syntactic property: the number of pages is the only metric of the specification's physical representation. Most measures are indeed syntactic, which may be explained by the many linguistic metrics which apply to natural-language in general and investigate syntactic or lexical defects. Metrics with measurement procedures studying the specification's semantics are mostly based on human judgments and reviews, either exclusively

Semantic	Syntactic	Physical
• Average Meanings per Cluster of Technical Terms (B.8) • Checklist-based Document Completeness (B.16) • Checklist-based Document Organzation (B.17) • External Documentation Consistency (EDC) (B.32) • Number of Clusters for Technical Terms (B.68) • Percentage of Descriptive Information (PDI) (B.89) • Stakeholder-Perceived Lexical Ambiguity (B.124) • Underreferenced Sentences (UrS) (B.131) • Underspecified Sentences (US) (B.132) • Unexplained Acronyms in Sentences (UeS) (B.133) • Weighted Ambiguity (WA) (B.146)	• (Weighted) Syntactic Sentence Ambiguity (B.1) • Comparative Phrases Smell (B.19) • Flesh-Kincaid Readability Grade Level (FK-RGL) (B.33) • Implicit Subject Sentences (ISS) (B.42) • Incomplete Sections (IS) (B.44) • Inter-Section NounPhrase Coupling (B.46) • Intra-Section NounPhrase Cohesion (B.49) • Lexical (Semantic) Sentence Ambiguity (B.50) • Machine-Learned Nocuous Anaphoric Ambiguity Count (B.56) • Multiple Sentences (MS) (B.62) • Number of Clones (B.67) • Open-Ended Terms Smell (B.83) • References to Incomplete Material (RIM) (B.103) • Relative Cloning Blow-Up (B.105) • Relative Occurence of Continuances (ROC) (B.108) • Relative Occurence of Directives (ROD) (B.110) • Relative Occurence of Weak Phrases (ROWP) (B.111) • Shape of Text Structure (STS) (B.120) • Subjective Language Smell (B.125) • Superlatives (B.127) • TBD Frequency (TBDF) (B.128) • To-Be-X Contained in SRS (TBx) (B.130) • Unspecific Adverbs Smell (B.135) • Vague Pronouns Smell (B.142)	• Concise in Number of Pages (CNP) (B.25)

Table 6.6.: Measured semiotic scope for content-item independent measures

or as a false-positive elimination of otherwise purely syntactic techniques.

6.7.2. Metrics measuring the Context Layer

Recall that the context specification defines the context of the system under consideration including a specification of the overall project scope, the stakeholders, rules, goals, and constraints as well as a specification of the domain model (see Fig. 2.1, p. 21).

Table 6.7.2 shows the results of the classification. Here, all but two metrics measure either the goals/objectives or the domain model, often specifically the business process. The classification reveals that for the former most measurement procedures are limited to the specification's syntactical properties. Therefore, they rely on the information provided to be complete and correct in order to assess the semantic quality of the specification. For instance, in order to interpret average customer goal satisfaction (metric B.7 in a meaningful way the measure depends on the correct and complete specification of goal contribution rates. In contrast, measurement procedures for the domain model rely on its semantics to a larger extent. In particular, the business process-related metrics proposed by Overhage et al. [2012] rely on human judgments rather than on automated syntactic checks, which constitute the majority of semantic measures for the domain model. Besides the goals/objectives and the domain model, the context specification's glossary is measured by glossary circularity (metric B.37. Moreover, any statement about the system's context is subject to an incorrect universal quantification (metric B.134) according to Berry and Kamsties [2000].

6.7.3. Metrics measuring the Requirements Layer

The requirements layer comprehends the requirements on the system under consideration, i.e., the specification of requirements from a userâs perspective without constraining the internal realization of the system (see Fig. 2.1, p. 21).

Many of the identified metrics are applicable to requirements in general. In terms of the AMDiRE artifact model (see Méndez Fernández and Penzenstadler [2014]), the identified metrics measure the the specification's content items *usage model, service model, function hierarchy, process requirements, deployment requirements, system constraints* and *quality requirements*. The classification results for requirements are shown in Table 6.7.3. The majority of measures (26) are concerned with the syntactic level of the specification, with only a few (7) also considering the requirements' semantics and one sole metric limited to the specification's physical level of abstraction, i.e., a size measure (metric B.123).

Several measures were identified to apply to the content items *usage model, service model* and *function hierarchy*. For the sake of simplicity, we will refer to this group of content items as the *functional requirements*. Use-case based metrics measure one member of this group, namely the usage model. In contrast to the functional requirements, two measures were proposed which study the *quality requirements* for the system under development based on the specification's semantics. In requirements specifications, data is exchanged which is described in the requirements specification's *data model*. Three

	Semantic	Syntactic
Any Context Layer Item		• Universal Quantification in Indicative Statements (UQ-IS) (B.134)
Goals and Objectives	• Goal (Dis-)Satisfaction Estimate (B.38) • Goal Confidence Profile (B.39)	• Agent-Goal Ratio (B.3) • Average Customer Goal Satisfaction (CUP) (B.7) • Average Minimum Goal Satisfacton (SAT) (B.9) • Deviation of Requirements Preference Estimations (VDV) (B.28) • Deviation of Stakeholder Requirements Preferences (HDV) (B.29) • Number of Leaf Goals of Agents (ANLG) (B.72) • Number of Objects of Leaf Goals (GNO) (B.74) • Percentage Leaf Goals with Object (PLGWO) (B.86) • Percentage Leaf Goals with Operation (PLGWOp) (B.87) • Percentage Leaf Obstacles with Goal Resolution (PLOWS) (B.88) • Percentage of Operations with Agent (POpWA) (B.90) • Positive Goal Contribution Rate (POS) (B.95) • Rational Rate (B.101) • Refined Customer Needs Ratio (B.104) • Requirements Goal Coverage (COV) (B.115)
Domain Model	• Behavioral Profile Consistency (B.12) • BPM Elements Completeness Ratio (B.13) • BPM Elements Redundancy Ratio (B.14) • BPM Label Consistency Ratio (B.15) • Correct BPM Elements Ratio (B.27) • Imposed Flexibility Contraints Ratio (B.43) • Relevant BPM Elements Ratio (B.114)	• Actor Presence (B.2) • Informational Completeness (B.45) • Resource Presence (B.116) • Sentence-Level Syntax Correctness Ratio (B.118) • Text Syntax Correctness Ratio (B.129) • Word Syntax Correctness Ratio (B.147)
Glossary		• Glossary Cicularity (GLC) (B.37)

Table 6.7.: Measures per semantic scope and content item of the context layer

Semantic	Syntactic	Physical
• Ambiguous Reviewer Interpretations (ABR) (B.4) • Local completeness: Understood Requirements (B.53) • Local Feature Completeness (LFC) (B.54) • Minimal Inconsistent Subset (B.57) • Pathfinder network similarity coefficient (B.85) • Percentage of Universally Understandable Requirements (PUUR) (B.92) • Score Ordering (B.117)	• Annotation Coverage (AC): Importance, Stability, Version (B.6) • Comment Frequency (CF) (B.18) • Dice Similarity Measure (Dice-Sim) (B.31) • Imperatives per Subjects (IpS) (B.41) • Lines-of-Text per Imperative (LpI) (B.51) • Loopholes Smell (B.55) • Minimality of Vocabularity (MoV) (B.58) • Negative Statements (B.63) • Number of Acronyms per Requirement (B.65) • Number of Anaphoric Terms per Requirement (B.66) • Number of Control Flow Terms per Requirement (B.69) • Number of Design Terms per Requirement (B.70) • Number of Domain Terms per Requirement (B.71) • Number of Negative Terms per Requirement (B.73) • Number of Optional Phrases (OP) (B.75) • Ontology-Based Completeness (B.79) • Ontology-Based Consistency (B.80) • Ontology-Based Correctness (ONTCOR) (B.81) • Ontology-Based Unambiguity (B.82) • Percentage of Strong Imperatives (PSI) (B.91) • Percentage of Validated Requirements (PVR) (B.93) • Relative Punctuation per Sentence (B.112) • Similarity of SRS Sentences based on Lexical Affinities (SimLA) (B.122) • Subjective Sentences (SS) (B.126) • Vague Sentences (VS) (B.143) • Weak Sentences (WS) (B.144)	• Size of Requirements (B.123)

Table 6.8.: Measures per semantic scope for general requirements

metrics measuring syntactic properties of this data model were identified. Finally, glossary circularity (metric B.37) is applicable not only to the context specification glossary but also to the requirements specification glossary. The classification results for the aforementioned content items are shown in Table 6.7.3.

6.8. Prerequisites for Application

Finally, we are interested in the preconditions required for the identified measures to be applicable (*RQ 6.3*). Therefore, in this section, we first briefly introduce the codes representing the identified prerequisites in Sec. 6.8.1 before presenting its mapping to the individual metrics in Sec. 6.8.2.

6.8.1. Identified Codes

As described in Sec. 6.1, we elicited prerequisites per category of interest (artifact, activity, infrastructure, organization, expertise/skills, and project) as open text and applied an open coding approach. During coding, we identified 31 unique codes which were assigned 549 times to the 136 identified metrics (avg. per metric: 3.37). Several identified codes were further parametrized to capture certain sub-aspects. For instance, any metric which requires the specification to be expressed in a specific language, the code *language* was assigned. However, since we are also interested in both the actual language used (e.g., a formal logic or natural language) and the content items to which this prerequisite applies (e.g., the usage model), we introduced two parameters. This way, we avoid taxonomies with multiple inheritance.

Tbl. 6.8.1 (Artifact, Activitiy, Infrastructure) and 6.8.1 (Organization, Expertise/Skills, Project) summarize the codes identified for the categories given in parentheses.

6.8.2. Prerequisites for Metrics

In this section, we present the prerequisites associated with the individual metrics. For the sake of brevity, in this thesis we limit ourselves to the prerequisites occurring most often respectively considered most important, as presented in Tbl. 6.12. The reader interested in the comprehensive association of all prerequisites described in Tbl. 6.8.1 and 6.8.1 is advised to consult the companion material available online[8].

[8] `http://www4.in.tum.de/~mund/metrics-companion.zip`

	Semantic	Syntactic
Functional Requirements	• Ambiguous Sentence Parts (B.5) • Completeness and Ambiguity of Object Flow (B.21) • Compound Accuracy Measure based on Sentence Parts (B.22) • Compound Completeness Measure based on Sentence Parts (B.23) • Compound Readability Measure based on Sentence Parts (B.24) • Consistency of Specified Functionality (B.26) • Functional Aggregation Error (B.34) • Functional Fragmentation Error (B.36) • Missing Conditions in Sentences (B.59) • Missing Constraints in Sentences (B.60) • Missing Sentence Parts (B.61) • Number of Sentence Part Types used (B.76) • Partiallity of Specified Functionality (B.84)	• Backward Concept Completeness (B.10) • Backward Interaction Completeness (B.11) • Functional Encapsulation Completeness (B.35) • Inter-Stakeholder Inconsistency Level (B.48) • Linked Communications completeness (B.52) • Number of (Gapped) Clones in Use-Cases (B.64)
Usage Model		• Number of Steps of Use Case (B.78) • Rate of Action Steps (B.98) • Rate of Exception Steps (B.99) • Rate of Use Case Action Steps (B.100) • Use-Case Conditional Steps (B.136) • Use-Case Similarity Factor (B.137) • Use-Case Actor Completeness (B.138) • Use-Case Linguistic Complexity (B.139) • Use-Case Technical Jargon (B.140) • Use Case Cyclomatic Complexity (B.141)
Quality Requirements	• Quality Requirements Spectrum Comparison (B.97)	
Data Model		• Actor Presence (B.2) • Informational Completeness (B.45) • Resource Presence (B.116)
Glossary		• Glossary Cicularity (B.37)

Table 6.9.: Measures per semantic scope and content item of the requirements layer

Category	Code	Description	Parameters
Artifact	*DesignatedElem(e)*	A specific (designated) syntactic element (symbol, word) must be used for marking the semantic element e.	e: *Element*
	Reference(E_1, E_2)	References between E_1 and E_2 must be explicitly specified if they are in a certain relationship (see measures' descriptions for details)	$E_1 \subseteq$ *Element*, $E_2 \subseteq$ *Element*
	Language(l, a)	The content entity a must be specified according to the symbols and grammar of language l	l: *Language*, a: *ContentEntity*
	Theory(t, a)	The artifact's content entity a must be specified/modeled according to the theory t	t: *Theory*, a: *ContentEntity*
	SelfContained	The requirements specification is self-contained in the sense that it is not based on external documentation	
	FragmentedModel(m)	The model m is fragmented into multiple diagrams	m: *Model*
	Specified(a)	The content item a is (completely) specified as part of the requirements specification	a: *ContentEntity*
	Template(a)	A template must be provided and/or used for the artifact content item a	a: *ContentEntity*
	Quality(q, a)	The artifact/entity a must posses the characteristic g	q: *QualityAttribute*, a: *ContentEntity*
	Provided(a_E)	The artifact/content item a_E is provided. In contrast to *specfied*, a_E is external to the requirements specification, i.e., not considered to be part of a typical specification.	ae: *ExternalContentEntity*
Activity	*ManualQA*(t)	A manual quality assessment activity, more specifically the task t, if provided, must be performed per measurement.	t: *Task* [opt]
	ManualQAOnce(t: *Task*[opt])	A manual quality assessment activity (specifically t, if provided) must be performed at least *once* per measure (and can thereafter be reused for subsequent measurements/projects)	t: *Task* [opt]
	Quality(q, t)	The quality assessment activity (specifically t, if provided) must possess the characteristic q	q: *QualityAttribute*, t: *Task* [opt]
	IncrementalSpec	The requirements specification is developed continuously and incrementally instead of being written at once as the final step	
	PerStakeholderReq	Requirements are elicited from individual stakeholders in isolation prior to developing a consolidated specification	
	Additional QA	The measure should be used in conjunction with other measures for the same quality according to the authors	
Infrastructure	*Tool*(t)	A specific tool is required, either commercial off-the-shelves (COTS), or to be implemented and/or customized (see individual metrics)	t: *Tool*

Table 6.10.: Codes identified for artifact, activity and infrastructure prerequisites

Category	Code	Description	Parameters
Organization	*AllowExtReviewers*	The organization must allow and provided access to the specification for reviewers which are external to the project and/or organization.	
	MultiProject	The project the requirements specification is developed for is part of a multi-project environment within the company and allows interchange of information and results of its projects.	
	ArtifactOrientation	The organization widely and thorougly employs artifacts in requirements engineering, in the sense that decisions are documented and documents are the predominant form of information interchange/communication	
	MaintainArtifact(a)	The artifact content item a is continuously maintained by the organization; Therefore, a is kept up-to-date and does not deprecate.	a: *ContentEntity*
	HighRessources	The organization is willing to allocate substantial resources for quality assessment of specifications	
	Policy(p)	The organization should enforce the policy p	p: *Policy*
Skills & Expertise	*Approach-Knowledge*(r, t)	The project participants acting in the role r must have a substantial knowledge about the software- or systems development approach t.	r: *Role*, t: *Theory*
	Domain-Knowledge(r)	The project participants acting in the role r must have a substantial knowledge about the domain of the system under development.	r: *Role*
	Technical-Understanding(r)	The project participants acting in the role r must have a substantial knowledge about the technical means of the system under development.	r: *Role*
	Resemble(r_1, r_2)	The concrete project participants acting in the role r_1 must closely resemble those acting in the role r_2. For the specific characteristics that both groups must possess the reader should consolidate the individual measures. However, as an example, reviewers participating in quality assessment should have an equal domain understanding and possess equal language skills.	r_1, r_2: *Role*
Project	*Stakeholder-Participation*	The stakeholders must be willing and able to participate during requirements engineering.	
	Developer-Participation	The developers must be willing and able to participate during requirements engineering. This implies that the developers are already known early.	
	BusinessIS	The system under development is classified as a business information systems, i.e., an information system which purpose is to support specific business processes.	
	UnderstoodDomain	The system under development is situated in a domain which is well-understood by the stakeholders and requirements engineers.	

Table 6.11.: Codes identified for organization, expertise, and project prerequisites

Metrics	Artifact: Theory	Artifact: Language	Artifact: Provided	Artifact: Template	Act.: ManualQA	Infr.: Tooling	Organization: Maintain	Organization: HighEfforts	Organization: Multiproject	Expertise: Approach Knowledge	Expertise: Domain Knowledge	Expertise: Technical Understanding	Project: Stakeholder Participation	Project: Developer Participation	Project: Business IS	Project: Understood Domain
(Weighted) Syntactic Sentence Ambiguity		NL				✓										
Actor Presence						✓					✓	✓	✓		✓	
Agent-Goal Ratio		KAOS				✓				✓						
Ambiguous Reviewer Interpretations (ABR)					✓											
Ambiguous Sentence Parts (ASP)		NL			✓											
Annotation Coverage (AC): Importance, Stability, Version																
Average Customer Goal Satisfaction (CUP)	AGORA				✓							✓	✓			
Average Meanings per Cluster of Technical Terms		NL				✓										
Average Minimum Goal Satisfacton (SAT)	AGORA				✓						✓	✓	✓			
Backward Concept Completeness		NL				✓										✓
Backward Interaction Completeness		NL				✓										✓
Behavioral Profile Consistency		BPM			✓	✓				✓	✓				✓	
BPM Elements Completeness Ratio					✓					✓	✓				✓	
BPM Elements Redundancy Ratio					✓					✓	✓				✓	

Metrics	Artifact				Act.	Infr.	Organization			Expertise					Project				
	Theory	Language	Provided	Template	ManualQA	Tooling	Maintain	HighEfforts	Multiproject	Approach Knowledge	Domain Knowledge	Technical Understanding			Stakeholder Participation	Developer Participation	Business IS	Understood Domain	
BPM Label Consistency Ratio					✓					✓	✓						✓		
Checklist-based Document Completeness				✓	✓		✓											✓	
Checklist-based Document Organzation					✓		✓												
Comment Frequency (CF)		NL			✓	✓													
Comparative Phrases Smell		NL				✓													
Completeness and Ambiguity of Object Flow (CAOF)		NL			✓						✓								
Completeness and Ambiguity of Object Flow (CAOF)		NL			✓						✓								
Compound Accuracy Measure based on Sentence Parts (ComAM-SP)		NL			✓						✓								
Compound Completeness Measure based on Sentence Parts (ComCM-SP)		NL			✓						✓								
Compound Readability Measure based on Sentence Parts (ComRM-SP)		NL			✓						✓								
Concise in Number of Pages (CNP)																			
Consistency of Specified Functionality (CSF)	State-based				✓						✓	✓						✓	
Correct BPM Elements Ratio					✓					✓	✓						✓		
Deviation of Requirements Preference Estimations (VDV)	AGORA				✓							✓			✓				

Metrics	Artifact				Act.	Infr.	Organization			Expertise					Project				
	Theory	Language	Provided	Template	ManualQA	Tooling	Maintain	HighEfforts	Multiproject	Approach Knowledge	Domain Knowledge	Technical Understanding			Stakeholder Participation	Developer Participation	Business IS	Understood Domain	
Deviation of Stakeholder Requirements Preferences (HDV)	AGORA				✓							✓			✓				
Dice Similarity Measure (Dice-Sim)		NL				✓													
Dice Similarity Measure (Dice-Sim)		NL				✓													
External Documentation Consistency (EDC)					✓						✓	✓							
Flesh-Kincaid Readability Grade Level (FK-RGL)		NL				✓													
Functional Aggregation Error					✓														
Functional Encapsulation Completeness																			
Functional Fragmentation Error					✓														
Glossary Cicularity (GLC)	REM	XML				✓				✓									
Goal (Dis-)Satisfaction Estimate		KAOS			✓				✓	✓	✓				✓			✓	
Goal Confidence Profile		KAOS			✓					✓	✓	✓			✓				
Imperatives per Subjects (IpS)		NL				✓													
Imperatives per Subjects (IpS)		NL				✓													
Implicit Subject Sentences (ISS)		NL			✓	✓													
Imposed Flexibility Constraints Ratio					✓						✓						✓		

Metrics	Artifact				Act.	Infr.	Organization			Expertise				Project				
	Theory	Language	Provided	Template	ManualQA	Tooling	Maintain	HighEfforts	Multiproject	Approach Knowledge	Domain Knowledge	Technical Understanding		Stakeholder Participation	Developer Participation	Business IS	Understood Domain	
Incomplete Sections (IS)				✓		✓												
Informational Completeness	Ontology	✓				✓					✓	✓		✓		✓		
Inter-Section NounPhrase Coupling		NL				✓											✓	
Inter-Stakeholder Inconsistency Level		Logic			✓			✓						✓				
Inter-Stakeholder Inconsistency Level		Logic			✓			✓						✓				
Intra-Section NounPhrase Cohesion		NL				✓											✓	
Lexical (Semantic) Sentence Ambiguity		NL			✓	✓												
Lines-of-Text per Imperative (LpI)		NL				✓												
Linked Communications completeness																		
Local completeness: Understood Requirements					✓						✓							
Local Feature Completeness (LFC)					✓						✓						✓	
Loopholes Smell		NL				✓												
Machine-Learned Nocuous Anaphoric Ambiguity Count		NL			✓	✓			✓									
Minimal Inconsistent Subset		Logic			✓	✓												
Minimality of Vocabularity (MoV)	REM	XML				✓								✓				
Missing Conditions in Sentences (MCdS)		NL			✓						✓							

Metrics	Artifact				Act.	Infr.	Organization			Expertise				Project				
	Theory	Language	Provided	Template	ManualQA	Tooling	Maintain	HighEfforts	Multiproject	Approach Knowledge	Domain Knowledge	Technical Understanding		Stakeholder Participation	Developer Participation	Business IS	Understood Domain	
Missing Constraints in Sentences (MCsS)		NL			✓						✓							
Missing Sentence Parts (MSP)		NL			✓						✓							
Multiple Sentences (MS)		NL			✓	✓												
Negative Statements		NL				✓												
Number of (Gapped) Clones in Use-Cases	UC	NL				✓												
Number of Acronyms per Requirement		NL				✓												
Number of Anaphoric Terms per Requirement		NL				✓												
Number of Clones		NL			✓	✓												
Number of Clusters for Technical Terms		NL				✓												
Number of Control Flow Terms per Requirement		NL				✓												
Number of Design Terms per Requirement		NL				✓	✓											
Number of Domain Terms per Requirement		NL				✓	✓										✓	
Number of Leaf Goals of Agents (ANLG)		KAOS				✓				✓								
Number of Negative Terms per Requirement		NL				✓												
Number of Objects of Leaf Goals (GNO)		KAOS				✓				✓								

Metrics	Artifact				Act.	Infr.	Organization			Expertise				Project				
	Theory	Language	Provided	Template	ManualQA	Tooling	Maintain	HighEfforts	Multiproject	Approach Knowledge	Domain Knowledge	Technical Understanding	Stakeholder Participation	Developer Participation	Business IS	Understood Domain	
Number of Optional Phrases (OP)		NL			✓	✓											
Number of Sentence Part Types used (NST)		NL			✓												
Number of Steps of Use Case (NOS)	UC																
Number of Steps of Use Case (NOS)	UC																
Ontology-Based Completeness					✓			✓			✓	✓	✓				
Ontology-Based Consistency					✓			✓			✓	✓	✓				
Ontology-Based Correctness (ONTCOR)					✓			✓			✓	✓	✓				
Ontology-Based Unambiguity					✓			✓			✓	✓	✓				
Open-Ended Terms Smell		NL				✓	✓										
Partiallity of Specified Functionality (PSF)	State-based				✓						✓	✓				✓	
Pathfinder network similarity coefficient					✓					✓			✓	✓			
Percentage Leaf Goals with Object (PLGWO)		KAOS				✓				✓							
Percentage Leaf Goals with Operation (PLGWOp)		KAOS				✓				✓							
Percentage Leaf Obstacles with Goal Resolution (PLOWS)		KAOS				✓				✓							
Percentage of Descriptive Information (PDI)		NL			✓											✓	

Metrics	Artifact				Act.	Infr.	Organization			Expertise				Project				
	Theory	Language	Provided	Template	ManualQA	Tooling	Maintain	HighEfforts	Multiproject	Approach Knowledge	Domain Knowledge	Technical Understanding		Stakeholder Participation	Developer Participation	Business IS	Understood Domain	
Percentage of Operations with Agent (POpWA)		KAOS				✓				✓								
Percentage of Strong Imperatives (PSI)		NL				✓												
Percentage of Universally Understandable Requirements (PUUR)					✓													
Percentage of Validated Requirements (PVR)																		
Positive Goal Contribution Rate (POS)	AGORA				✓						✓			✓				
Positive Goal Contribution Rate (POS)	AGORA				✓						✓			✓				
Quality Requirements Spectrum Comparison					✓			✓		✓	✓							
Quality Requirements Spectrum Comparison					✓			✓		✓	✓							
Rate of Action Steps (NOASRATE)	UC																	
Rate of Exception Steps (NOERATE)	UC																	
Rate of Exception Steps (NOERATE)	UC																	
Rate of Use Case Action Steps (NOUSRATE)	UC																	
Rational Rate	AGORA				✓									✓				
Reference Model Coverage		✓	Ref.-Model			✓												
References to Incomplete Material (RIM)						✓												

Metrics	Artifact				Act.	Infr.	Organization			Expertise					Project				
	Theory	Language	Provided	Template	ManualQA	Tooling	Maintain	HighEfforts	Multiproject	Approach Knowledge	Domain Knowledge	Technical Understanding			Stakeholder Participation	Developer Participation	Business IS	Understood Domain	
Refined Customer Needs Ratio	AGORA				✓						✓	✓			✓				
Refined Customer Needs Ratio	AGORA				✓						✓	✓			✓				
Relative Cloning Blow-Up		NL			✓	✓													
Relative Line Crossings		✓				✓													
Relative Model Minimality (Absence of Redundancy)		✓				✓													
Relative Occurence of Continuances (ROC)		NL				✓													
Relative Occurence of Directives (ROD)		NL				✓													
Relative Occurence of Directives (ROD)		NL				✓													
Relative Occurence of Weak Phrases (ROWP)		NL				✓													
Relative Punctuation per Sentence		NL				✓													
Relative Use of Simple Model Constructs		✓				✓													
Relevant BPM Elements Ratio					✓					✓							✓		
Requirements Goal Coverage (COV)	AGORA				✓						✓				✓				
Resource Presence						✓					✓	✓			✓		✓		
Score Ordering		Logic			✓	✓													
Sentence-Level Syntax Correctness Ratio					✓												✓		

Metrics	Artifact				Act.	Infr.	Organization			Expertise				Project				
	Theory	Language	Provided	Template	ManualQA	Tooling	Maintain	HighEfforts	Multiproject	Approach Knowledge	Domain Knowledge	Technical Understanding		Stakeholder Participation	Developer Participation	Business IS	Understood Domain	
Shape of Text Structure (STS)		NL				✓												
Shape of Text Structure (STS)		NL				✓												
Similarity of SRS Sentences based on Lexical Affinities (SimLA)		NL	Higher-Level			✓												
Similarity of SRS Sentences based on Lexical Affinities (SimLA)		NL	Higher-Level			✓												
Size of Requirements [in Words]		NL																
Stakeholder-Perceived Lexical Ambiguity		NL			✓									✓				
Subjective Language Smell		NL				✓	✓											
Subjective Sentences (SS)		NL			✓	✓					✓							
Superlatives		NL				✓	✓											
TBD Frequency (TBDF)		NL			✓												✓	
Text Syntax Correctness Ratio					✓											✓		
To-Be-X Contained in SRS (TBx)						✓												
Underreferenced Sentences (UrS)		NL			✓													
Underspecified Sentences (US)		NL			✓													
Unexplained Acronyms in Sentences (UeS)		NL			✓													

Metrics	Artifact				Act.	Infr.	Organization			Expertise				Project				
	Theory	Language	Provided	Template	ManualQA	Tooling	Maintain	HighEfforts	Multiproject	Approach Knowledge	Domain Knowledge	Technical Understanding	Stakeholder Participation	Developer Participation	Business IS	Understood Domain		
Universal Quantification in Indicative Statements (UQ-IS)		NL				✓					✓							
Unspecific Adverbs Smell		NL				✓	✓											
Use-Case Conditional Steps	UC	NL				✓												
Use-Case Similarity Factor	UC	NL				✓												
Use-Case Step Actor Completeness	UC	NL			✓	✓												
Use-Case Step Linguistic Complexity	UC	NL				✓												
Use-Case Technical Jargon	UC	NL	Dict			✓												
Use Case Cyclomatic Complecity (UCCC)	UC																	
Vague Pronouns Smell		NL				✓												
Vague Sentences (VS)		NL			✓	✓												
Weak Sentences (WS)		NL			✓	✓												
Weighted Ambiguity (WA)		NL			✓	✓			✓									
Weighted Ambiguity (WA)		NL			✓	✓			✓									
Word Syntax Correctness Ratio					✓										✓			

6.9. Related Work

As seen in this survey, research in measurement-based quality assessment mostly concentrates on studying individual measures for a particular attribute of the requirements specification (e.g., Kiyavitskaya et al. [2008], Matsuoka and Lepage [2011], Weidlich et al. [2011], Matulevicius et al. [2010], och Dag et al. [2001b], Boness et al. [2011], Rine and Fraga [2015], Yang et al. [2011], Mu et al. [2005]) or individual techniques measuring specific aspects with several quality attributes (e.g., Kaiya et al. [2002], Fabbrini et al. [2000], Femmer et al. [2014c], Wilson et al. [1997], Génova et al. [2013], Espana et al. [2009]).

Yet, several assessment models provide a more comprehensive assessment of the requirements specification's quality. Davis et al. [1993a] propose a set of 13 quality attributes (see Tbl. 4.3, pp. 85 for the included attributes), and associate measures with each. However, the proposed measures actually remain unmeasurable in practice due to being vague and/or relying on data not collectable in practice. For instance, the authors propose to measure ambiguity as the ratio of unambiguous to all requirements without specifying further details. Consequently, the proposed measures rather qualify to guide discussion about reference metrics for certain quality attributes than to provide concrete measurement results in practice. Kenett [1996] provided an assessment model based on measurement of natural-language specifications. Essentially, the model relies on identifying certain parts of the sentences, called attributes by the authors, such as initiators of an action, actions, objects, conditions, contraints, et cetera. The model includes base measures, e.g., the ratio of missing conditions to all sentences requiring a condition, as well as compound measures, which are obtained as a weighted sum of base measures. Unfortunately, the authors do not specify the precise data collection procedure, so we conservatively assume that it is based on manual reviews. The 3QM-framework proposed by Overhage et al. [2012] assesses the quality of business process models. In addition to providing individual measures, it also associates weights with each of it to obtain an overall quality assessment. As a by-product of studying what information is necessary to apply measures of requirements quality, Monperrus et al. [2013] performed a secondary study on requirements metrics. While similar to our approach, the scope of the study was limited due to the different objectives pursued. Specifically, the study excluded measures which collect data on a syntactic level or which are "natural-language based". The authors state that this is due to the former not being appropriate to obtain semantic information, while the later "are often imprecise". The study investigated 11 papers which resulted in 78 measures identified, most of which only differed slightly. Unfortunately, the only information disclosed was a short description of the measured property, while, e.g., data interpretation was considered out of scope by the authors.

Compared to the assessment model presented in this chapter, state-of-the-art assessment models rely on a severely limited number of measures. While the 3QM-framework is limited to business process models only, the measures suggested by Davis et al. [1993a] and Kenett [1996] are not applicable in practice due to unclear data collection procedures. If we would assume manual data collection, the expected costs of measurement

would be so extremely high that measurement would be uneconomical.

Contributions to the State-of-the-Art The presented assessment model contributes a consistent (re-)evaluation of the state-of-the-art quality metrics regarding aspects which are considered mandatory for a comprehensive and sound quality assessment (as described in the meta-model in Sec. 5.1, pp. 102) but commonly neglected or completely ignored in literature. In particular, we clarify their relationship to the specification's quality attributes and interpretation procedures, elicit the associated threats to validity and prerequisites for application, and evaluate their actionability. Finally, based on those results, we compose a unified and comprehensive measurement-based assessment (reference) model, which is first of its kind regarding the extent and depth, and propose a systematic method how to tailor this model for a specific context of use.

6.10. Summary

In this chapter, we proposed a quality assessment model based on state-of-the-art measures. To obtain this model, we conducted an extensive systematic literature review and in-depth analysis which resulted in 136 metrics being (re-)evaluated and embedded into an unified assessment model. In particular, this assessment model provides a detailed view on:

- the relationship between the assessed quality attributes defined and the properties actually measured by the individual metrics
- the measured scope of the specification artifact in terms of artifacts and their content items and the level of perception (semantic content, syntactic structure, physical representation)
- the prerequisites required to apply those metrics

The assessment model is specified according to the meta-model presented in Sec. 5.1 and can hence be used as envisioned in the process model in Sec. 5.2. Based on this assessment model and process proposal, we envision quality managers and requirements engineers to be able to use measures as an instrument to carefully and soundly assess certain quality attributes of requirements specifications.

7 Chapter

Evaluation and Limitations of Measurability

In this chapter we evaluate the quality assessment model proposed in Cha. 6 which reflects the current state-of-the-art of the requirements engineering community and is at the heart of our assessment approach (see Cha. 5). Unfortunately, all models are wrong, in the sense that no model is an *exact* representation of any system in the real world, but some are useful [Box, 1979]. Consequently, models must not be evaluated in terms of "truthfulness", but with regards to their utility (or adequacy) for their specific purpose, in this case, the assessment of a requirements specification at hand regarding its fitness for downstream development. We investigate three factors which are crucial for its utility in practice, according to our experiences and understanding, namely: (1) its applicability in the specific context of use, (2) the validity of the assessments provided, and (3) the actionability of the results, i.e., whether recommendations for action are provided.

To this end, we study the measures contained in the quality assessment model by means of an exploratory (meta-)analysis. Specifically, we extent and summarize the data collected (see Sec. 6.1) from the original publications on the individual measures to evaluate the adequacy of the model and to find commonalities among the measures and peculiarities of the quality attributes under assessment. As a result, we provide a qualitative characterization of (i) the context in which the assessment model is applicable, (ii) the limitations to validity associated with the types of measures and the quality attributes under assessment, and (iii) the extent the model provides actionable feedback. Based on those results, we discuss (the limitations of) measurability of our model, which also reflects the current state-of-the-art, on a more general note.

Object of study:	(Concrete) quality assessment model (i.e., the result of tailoring the quality assessment model as described in Sec. 5.2.1, pp. 115)
Purpose:	Evaluation
Focus:	Adequacy (in terms of applicability, validity, and actionability)
Viewpoint:	Quality managers
Context:	Development of software or software-intensive systems in practice

Table 7.1.: Study objective described according to the GQM template [Basili et al., 1994]

The chapter is structured as follows: First, we clarify our research questions in Sec. 7.1. Next, we describe our research method in Sec. 7.2 and discuss its threats to validity in Sec. 7.3. In Sec. 7.4, we present our results and their interpretations with respect to the research questions. Finally, we discuss the implications of our meta-analysis regarding the limitations of measurability in Sec. 7.5 before summarizing our evaluation in Sec. 7.6.

7.1. Research Questions

The purpose of the study presented in this section is to evaluate the adequacy of the quality assessment model from the point of view of quality managers in the context of software-development in practice, as summarized in Tbl. 7.1. To this end, we study the applicability, validity, and actionability of its measures as formulated in the following research questions:

RQ 1 (Applicability): In what context can the model be applied to what extent? As observed in the previous chapter, most measures require specific prerequisites to be fulfilled in order to be applicable. For instance, they may require a certain language (e.g., first-order logic), tool (e.g., a part-of-speech (POS) tagger), or expertise (e.g., the requirements engineer must be familiar with the domain). Hence, not any (project) context may allow to apply all measures. RQ 1 seeks to identify which contexts allow the assessment model to be applied to measure which quality attributes, based on the prerequisites identified regarding artifacts, activities, infrastructure, organization and work structure, staff and expertise, and the project in general (see Sec. 6.1.6 for a description).

RQ 2 (Validity): To what extent does the model provide valid assessments? Despite applicability, an assessment model must in the first place provide assessments which reflect the specification's actual quality to an appropriate extent. While what is considered appropriate depends on the use-case (e.g., the influenced decisions and the associated risks), RQ 2 seeks to characterize the measures validity by studying the evidence provided by the original authors and systematic as well as individual threats which potentially limit it.

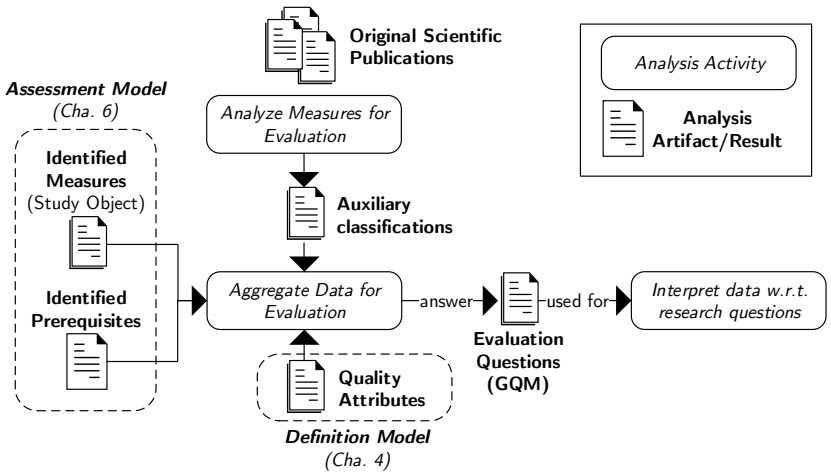


Figure 7.1.: Meta-analysis for evaluating the adequacy of the assessment model.

RQ 3 (Actionability): To what extent can the model be used for constructive quality assurance? Meneely et al. [2012] defines a measure to be *actionable* if it reflects some property of the specification in a way that enables managers to make decisions for improvement during requirements engineering. According to practitioners (see Sec. 5.1.6), actionable measures are a means for constructive quality assurance which greatly increases the value of measurement-based assessment. Therefore, in RQ 3, we study to what extent the quality assessment model provides recommendations for actions in addition to assessments.

7.2. Research Method

We provide qualitative answers to the above research questions by means of an exploratory (meta-)analysis. More specifically, we extent and summarize the data collected (see Sec. 6.1) from the original publications about the individual measures. Based on this data, we seek to identify commonalities among the measures and peculiarities of the quality attributes under assessment. The research method is illustrated in Fig. 7.1, and we subsequently describe our data collection and analysis procedure in more detail.

Data Collection Essentially, for data collection, we relied on two kinds of sources, namely the results of the systematic literature review and additional classifications specifically obtained for the purpose of this evaluation. Regarding the former, we used the information obtained for individual measures as described in Tbl. 6.3 (pp. 130), their associated prerequisites for their application resulting from an open coding process (see Sec. 6.8, pp. 165), and their mapping to the attributes from the quality definition model

(Cha. 4). Moreover, we require additional information specifically for this evaluation, e.g., what kind of evidence the authors provided that support validity of the proposed metrics. This auxiliary information was obtained by directly analyzing the original publications by the author of this thesis.

Data Analysis To prepare analysis, we followed the Goal–Question–Metric (GQM) paradigm [Basili et al., 1994] and derived specific questions related to our research questions which serve as an initial guidance (see Tbl. 7.2). During analysis, in a first step, we summarized the collected data to answer those questions and visualized the results. Based on those initial findings, we proceeded driven by curiosity, i.e., we explored the available data regarding questions which we considered to be of most value to answer the studied research questions or provide the most valuable insights about the adequacy of the assessment model. In addition, we also provide examples from individual metrics to provide a richer picture of our findings. All data was available in the *R Project for Statistical Computing* and we used *RStudio 1.0.136*[1] for our analysis.

7.3. Threats to Validity

In this section we reflect on the threats to validity of this study.

Threats to Internal Validity It is possible that our R scripts are flawed. To mitigate this threat, we visualized all data and occasionally analyzed individual metrics to check correctness of the scripts. Furthermore, when eliciting the evidence provided for the validity of the proposed measures by the original authors, we only studied the originally paper. In case evidence beyond those found in that particular paper is published later, it is not considered in this study. Hence, the provided evidence discussed in Sec.7.4 is an lower bound about the evidence indeed provided by the scientific community.

Threats to Construct Validity We heavily relied on the *amount* of measures contained in the assessment model as an empirical construct for the extent some quality attribute is measurable. This is, however, an oversimplification since it does not account for the *quality* of the measures. One sophisticated measure for syntactic quality might provide better assessment results than several for, say, semantic correctness. Hence, we cannot conclude that semantic correctness is more measurable than syntactic quality. Instead, we can only conclude that semantic correctness is more likely to be measurable than syntactic correctness if we do not know anything about the measures itself (or if we know that they are of equal quality regarding their fitness for quality assessment). However, since how suitable a measure is depends on the precise context, we have to accept we have no better empirical method to discuss measurability in general.

[1] `www.rstudio.com`

Goal	Question (Metric)	Rationale/Use for Interpretation
RQ 1	*Which prerequisites apply to how many metrics?*	Common prerequisites limit the applicability to a greater extent; used to identify critical prerequisites, i.e., those which are frequently required by the model but seldom given in practice.
	Which quality attributes require which prerequisites?	We expect prerequisites to not be equally distributed; used to answer which prerequisites allow or must be established for a particular quality attribute.
RQ 2	*Which quality attributes are measured at which level of perception by how many metrics?*	We distinguish between measures which perceive the SRS at the level of its physical representation, syntax, and semantics; we argument that the level of perception must correspond to the quality attribute, or otherwise, validity may be compromised.
	What is the relationship between the measured and the assessed attribute?	We distinguish between direct and indirect measures (relying on proxy attributes). Depending on the kind of proxy attribute, validity may be impaired and/or threatened.
	For how many metrics have threats to validity been identified?	We use this to characterize to what extent the measures' validity is threatened on an individual basis.
	What kind of evidence is provided by the measures' authors regarding its validity?	The evidence provided justifies the measures' validity; In case of empirical evidence, it often also provides a quantification.
RQ 3	*How many metrics provide recommendations for actions?*	Some measures provide recommendations for action while others don't. We interpret the number of metrics as the extent to which the model is actionable or not.

Table 7.2.: Questions used to analyze the research questions based on data collected for individual measures

Threats to External Validity The main threat to validity is the use of expert classifications. While we strived to maintain a rational and objective perspective on measures, misinterpretations and -evaluation are still very likely to occur. Consequently, the results are subject to researcher bias, and we would advocate future research to study measurability, e.g., by means of case study research.

7.4. Results and Interpretation

In this section, we present the results of our study, and interpret them with respect to our research questions.

7.4.1. RQ 1: Applicability

Results Fig. 7.2 shows the frequency of the identified prerequisites associated with the 136 measures contained in the assessment model. The codes used for prerequisites are described in Tbl. 6.8.1 (pp. 167). For the sake of readability, prerequisites associated with only one measure were omitted from the figure but rare overall (13.7% of all assignments).

As expected for assessing the requirements specification document, artifact-related prerequisites were most common. 87 (64.0%) respectively 24 (17.6%) measures relied on a specific language or theory to be used. If we break-down those prerequisites, as illustrated in Fig. 7.3a and 7.3b, we see that natural-language is the predominant language constraint, with measures relying on requirements being specified in other languages (e.g., logic or graphical modeling languages) only rarely. In addition, 9 (6.6%) measures require goal models to be specified according to the KAOS modeling language. Regarding the use of certain theories for specification, 13 measures required the requirements to be specified in terms of use-cases, while 10 measures required goals to be specified according to the AGORA meta-model.

The second most common prerequisites (assigned to 78 measures, 57.4%) concerned the use of specific tools (e.g., specialized NLP tools) during measurement. In contrast, 68 (50%) measures relied on manual activities. Consequently, the vast majority of measures were either fully manual or fully automated ones, with only 10 (7.4%) semi-automated measures (e.g., guiding measurement using a built-in interactive false-positive elimination). Out of those manual measures, 10 measures require very high efforts, for instance, the functional aggregation measure (app. B.34) which relies on manually evaluating each system function according to a set of criteria. In addition, four measures required a manual quality-assurance to take place once for a specific project/context, but not during each measurement (denoted by the code `ManualQA-Once`), e.g., the noxious anaphoric ambiguity count (metric B.56) which relies on human judgments as training for machine learning to automate measurement.

Also, several measures rely on expertise of the requirements engineers (writing the specification) and quality managers (analyzing the specification), as illustrated in Fig. 7.3c. Domain knowledge is assumed by 19 measures (14.0%), and applies more often to the requirements engineer writing the specification than the quality manager assessing its

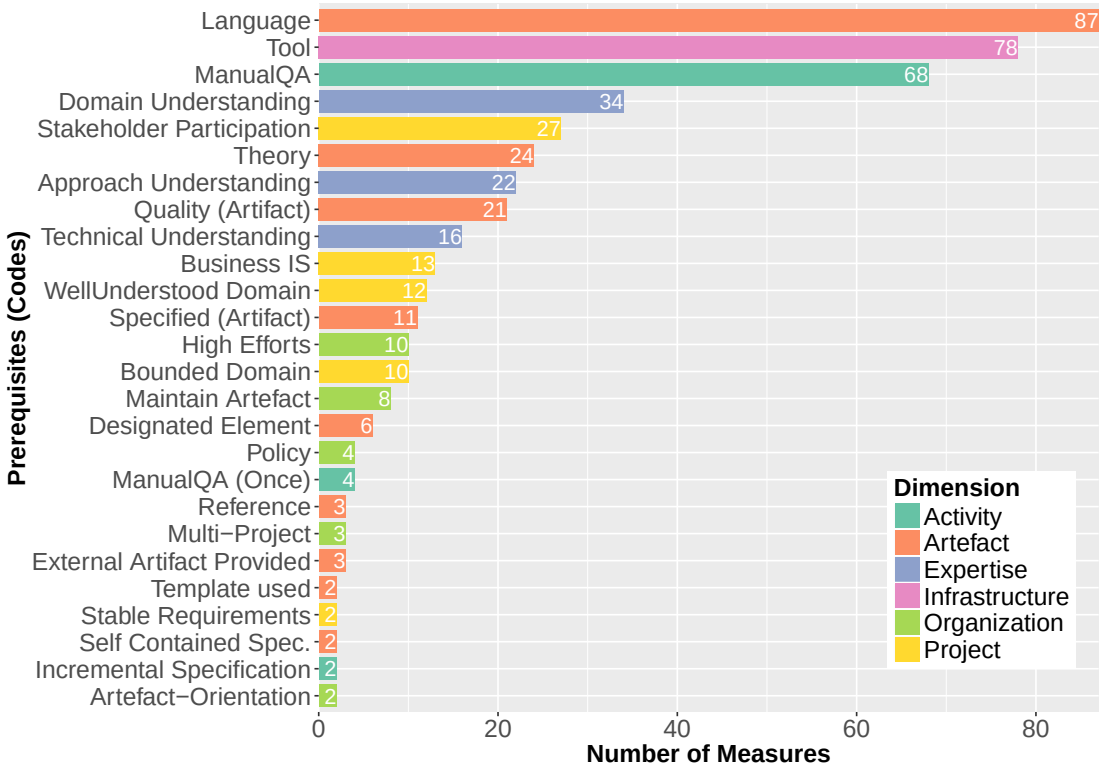
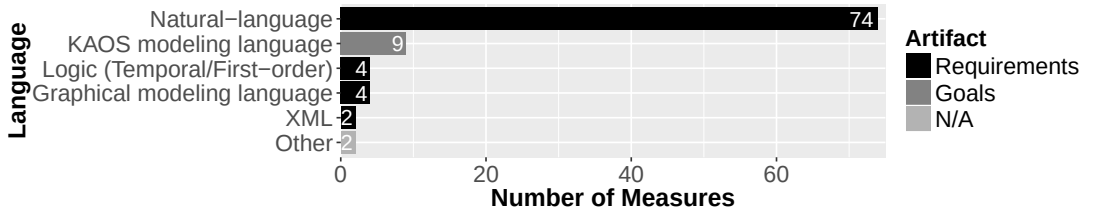


Figure 7.2.: Frequency of prerequisites for applying the assessment model's measures (Codes are described in Tbl. 6.8.1, pp. 167)

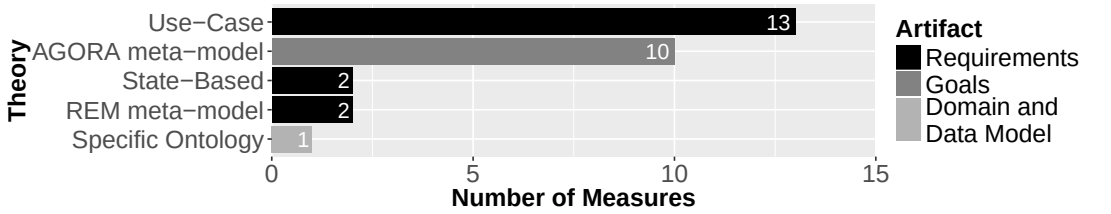
quality (12 to 7 times). In contrast, approach knowledge, i.e., knowledge about a specific RE approach, is more frequently required of the quality manager compared to the requirements engineer. Indeed, some measures use a specific approach for evaluation (for instance, pathfinder networks to capture a readers understanding of the specification, as in metric B.85) without demanding any specific expertise from the requirements engineer. Last but not least, 8 measures require the requirements engineers to have a technical understanding of the system to be developed (e.g., because he or she must know who the system's functionality contributes to its goals as in metric B.104), while solely one measure requires this from the quality manager.

Other more frequently identified prerequisites are that stakeholders are participating in quality assessment (27 measures, 19.6%), the requirements specification or its artifacts and content items possess certain qualities (e.g., metric B.46 assumes lexical consistency; 21 measures, 15.4%), or the system is a business information system (13 measures, 9.6%).

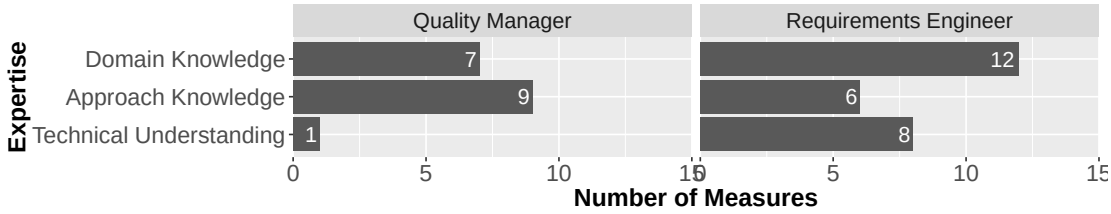
Finally, Fig. 7.4 illustrates selected prerequisites associated with measures which with respect to the quality attribute they assess.



(a) Language



(b) Theory



(c) Staff Expertise

Figure 7.3.: Breakdown of language, theory and staff expertise prerequisites

Interpretation Applicability is, as seen by the wide spectrum of codes and parameters, a very individual characteristics of each metric. However, generally speaking, our results suggest that applicability out-of-the-box is rather poor because of the large number of identified prerequisites. In particular, according to the frequency of prerequisites and our experience in practice, we expect the main barriers for application to be the tools that must be developed or acquired, the availability and training of staff, and the availability and participation of stakeholders during RE quality assurance. Consequently, any measurement program targeting a comprehensive assessment must overcome those barriers, if necessary. In contrast, we consider the constraint of about half of all measures to natural-language less problematic because of the prominent role it plays in requirements engineering in practice. In fact, we even suspect the high number of measures proposed for natural-language to be a consequence of the authors' intentions to be applicable in practice.

Provided those barriers are overcome, applicability becomes much less of an issue, especially if manual and semi-manual measures are acceptable. In this case, applicability becomes more of a question of individual measures. The assessment models comprises several quite specialized measures, e.g., for business IS, multi-project situations, if high-

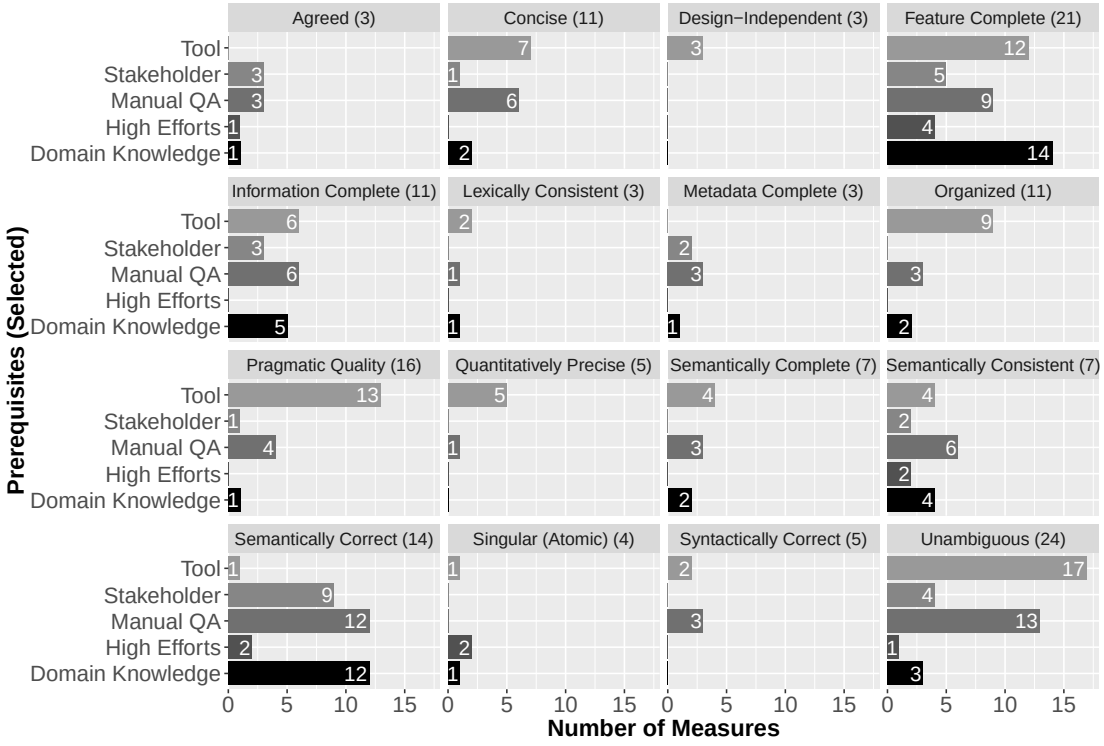


Figure 7.4.: Frequency of selected prerequisites per quality attribute (number in parentheses denotes the measures for the attribute in total; codes are described in Tbl. 6.8.1, pp. 167)

level documentation is available, or high efforts are acceptable. In contrast to the above prerequisites, we expect applicability as given or at least achievable for a substantial number of measures, and therefore not inhibiting a measurement program per se, for most projects in practice.

7.4.2. RQ 2: Validity

Subsequently, we present the results obtained regarding the extent the assessment model provides valid assessments of the specification's actual quality. To this end, we study both systematic threats to validity, i.e., potential causes which limit the validity of a group of measures due to the way it is assessed, and individual threats specific to each measure, as well as the evidence provided by the original authors.

Results: Systematic Threats to Validity We studied two reasons which limit the validity of certain measures, namely an insufficient level of perception of the measurement procedure and the use of indirect measurement.

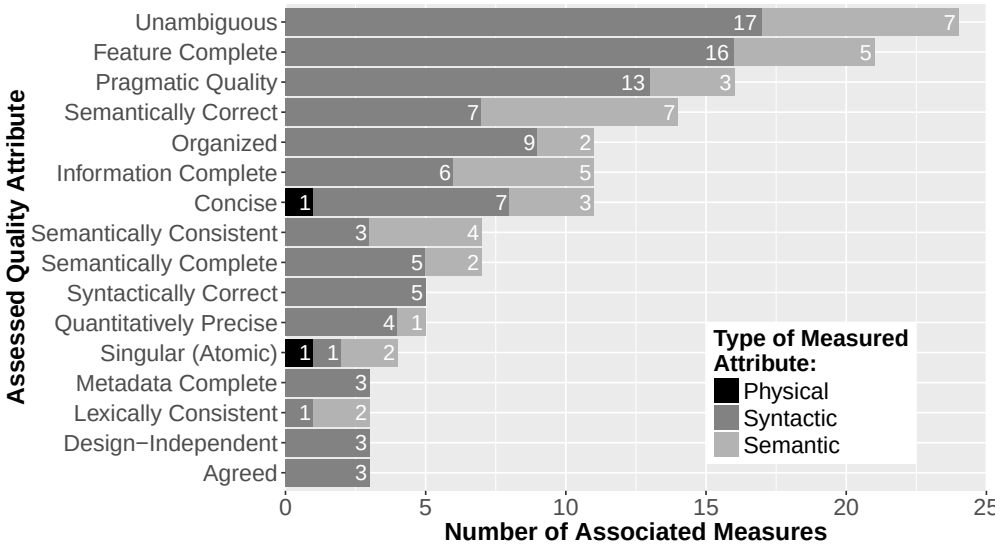


Figure 7.5.: Level of perception of metrics (per assessed quality attributes)

By level of perception we refer the depth of a measurement procedure in understanding the content of the requirements specification. Specifically, we distinguish between measuring the physical representation, syntactic structure, or semantic content (see Sec. 6.1.6). We consider the level of perception of a measure insufficient if it measures the quality attribute under assessment by relying on a simplified view on the requirements specification than actually necessary. An instance of an insufficient perception are the number of steps of use-cases (metric B.98) which assess feature completeness by measuring a syntactic property using a tool-based analysis of use-cases and comparing it to predefined thresholds. Another example is the assessment of the requirements specification's freedom of unnecessary design decisions by scanning for certain keywords from a dictionary (metric B.70). Fig. 7.5 shows the level of perception of the assessment model's measures organized according to the associated quality attribute under assessment. All but syntactic correctness would require measures to perceive the requirements specification's semantics, or, in the case of attributes ascribed to pragmatic quality, its understanding by the audience. However, the majority of measures perceives only the requirements specification's syntax. While this has pragmatic reasons, the drawback is that all syntactic measures are inherently imperfect (as are the measures perceiving the physical representation, even though those are very rare).

In addition to insufficient perception, the use of indirect measurement is a systematic threat to validity. While direct measurement actually measures the attribute under assessment, indirect measurement studies so-called proxy (or surrogate) attributes but interprets the measurement results with respect to a (different) quality attribute under assessment. In particular, for indirect measurement, we distinguish between sub-attributes

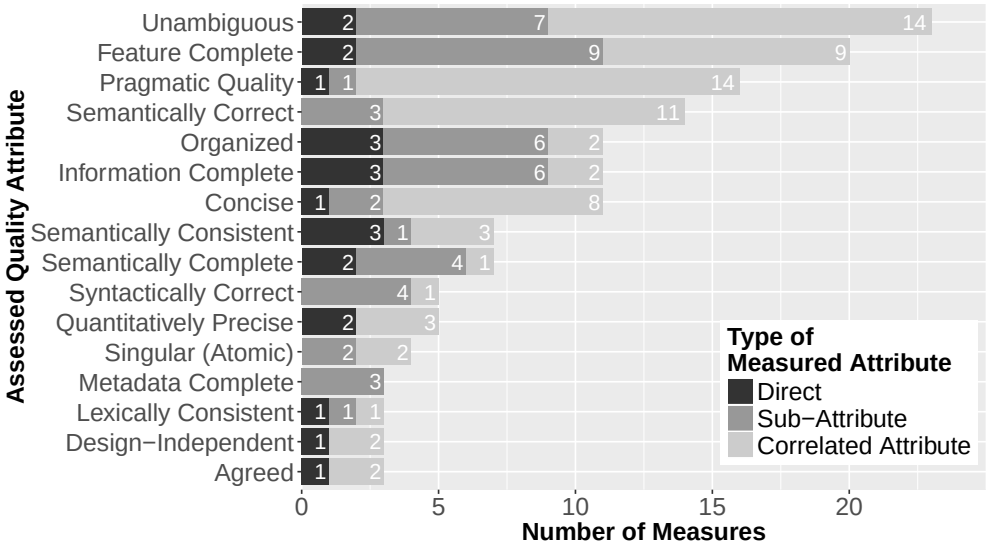


Figure 7.6.: Direct and indirect measures per quality attribute. The latter further distinguishes between sub-attributes and correlated attributes.

which are specializations of a quality attribute regarding a specific aspect, and attributes correlated to some extent with the quality attributes. Fig. 7.6 shows the result of classifying the individual measures according to those categories (see Fig. 6.1 (p. 132) for a decision-tree and methodology of how we classified the measures), and organized according to quality attributes they purport to assess. Our results revealed that only a small portion of measures are actually direct measurements of a quality attribute. Instead, in general, most metrics actually only measure a sub-characteristic of the attribute (e.g., anaphoric ambiguity instead of ambiguity in general) or related attributes (e.g., the coverage of higher-level documentation by the requirements specification as an indicator for feature completeness).

Results: Individual Validity and Provided Evidence Regarding individual validity, we studied whether threats to validity were identified for individual metrics during our in-depth analysis (see Sec. 6.1.6). The results are illustrated in Fig. 7.7. Overall, for the vast majority of measures (85.1%), the validity of measurement, interpretation or generalization was threatened. Conversely, for only 22 measures, no individual threats were identified. Those measures had in common to be based on formal models, thereby removing human judgment from the measurement and interpretation procedures, e.g., but required (parts of) the specification to be formalized (e.g., functional requirements expressed as propositions in formal logic, as in the minimal-inconsistent-subset metric B.57). In addition, measures concerned with semantic completeness and correctness relied on being provided valid external artifacts, such as an ontology of the system's

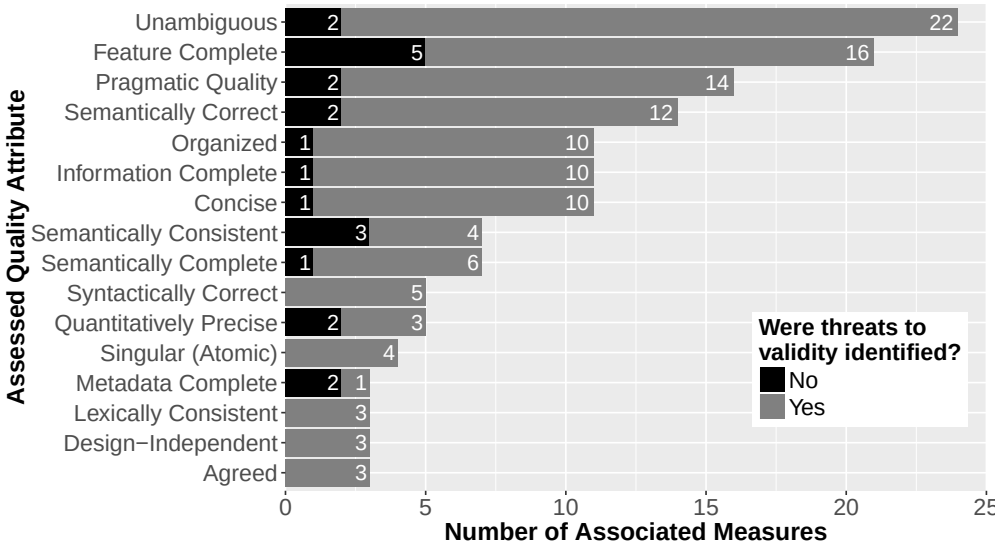


Figure 7.7.: Validity: Classification of measures according to whether or not threats to validity have been associated, per quality attribute.

domain (metric B.2 or B.79), thereby effectively replacing threats to validity by assuming validity. For certain attributes of pragmatic quality, the measures indeed provide valid assessments, however only for very specific sub-attributes. For instance, while the relative number of line crossings in diagrams (metric B.106) can be validly assessments, topological organization is only one aspect for a specification to be considered organized.

Furthermore, we studied what kind of evidence was provided in the original publication itself. Specifically, we looked for two types of evidence, namely whether an a-priori rationale explaining why the interpreted measurement result constitutes a valid assessment of the attribute under assessment (e.g., why the use of universal quantification indicates limited semantic correctness) is explicitly given, and whether in addition empirical evidence is provided which underpins and ideally quantifies the measures validity. The results are shown in Fig. 7.8. Overall, we see that most measures lack an empirical evaluation but are instead only based on an argument. This is particularly true for measures of semantic quality, while pragmatic quality measures are more common to also provide empirical evidence.

Interpretation In general, our results confirm that measures are imperfect means to assess the quality of requirements specifications since we identified at least one potential threat to their validity (or they impose unrealistic assumptions on external material to be provided, e.g., a correct and complete domain ontology). This applies likewise to manual measures based on judgments as well as automated ones relying on tools. While the latter suffer from not being able to fully understand the semantics of the specification,

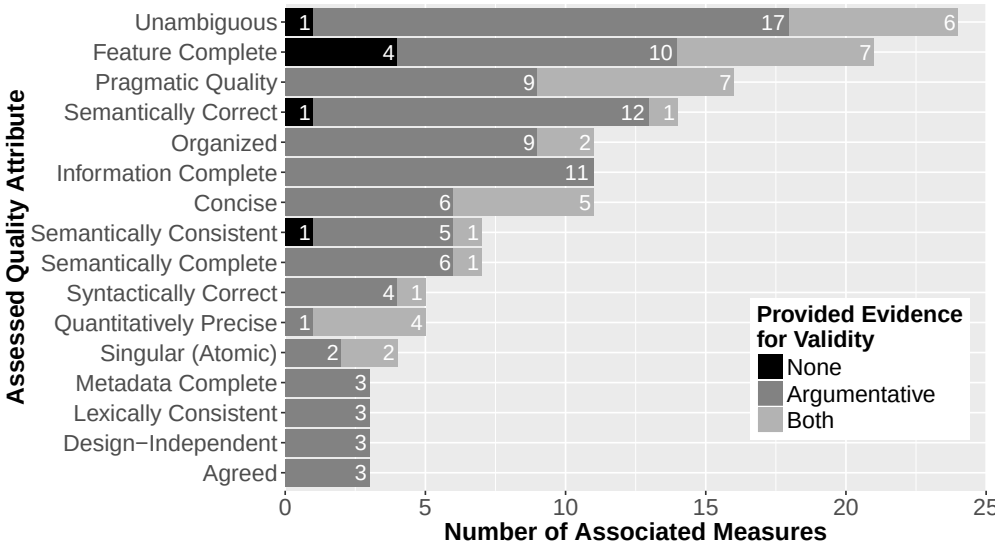


Figure 7.8.: Validity: Evidence provided by the measures' authors

thereby inherently limiting validity of assessments concerning pragmatic and semantic quality attributes, the former lack internal validity since they are subject to human error and misjudgment.

That being said, the proposed measures can still be useful, e.g., if the decisions and improvements resulting from the assessments justify the costs of measurement, and provided the potential and extent for invalid results is adequately handled (see the process model described in Sec. 5.2). To this end, lightweight semantic analysis techniques and machine-learning approaches (e.g., Bernárdez et al. [2004b], Yang et al. [2011]) blur the borders between syntactic and semantic analysis as well as manual and automated ones, lowering the costs of measurement while increasing the validity of results. Finally, while we agree with Meneely et al. [2012] that theoretical validity, i.e., a sound argument, is important, empirical evidence often makes the extent to which probabilistic proxy measures are valid explicit and therefore easier to evaluate for quality managers seeking to identify suitable measures.

7.4.3. RQ 3: Actionability

Results Fig. 7.9 shows the number of measures which provide recommendations for action compared to those which do not. Overall, 78.4% of measures provide some recommendations for actions. However, those recommendations differ in terms of how accurately they can pinpoint to the problem and how much effort is required to perform them. They range from only stating the contents of the requirements specification which should be reconsidered and a vague action to be performed (e.g., which use-case is likely to miss actions and is subject to discussion with stakeholders, metric B.78) to precise

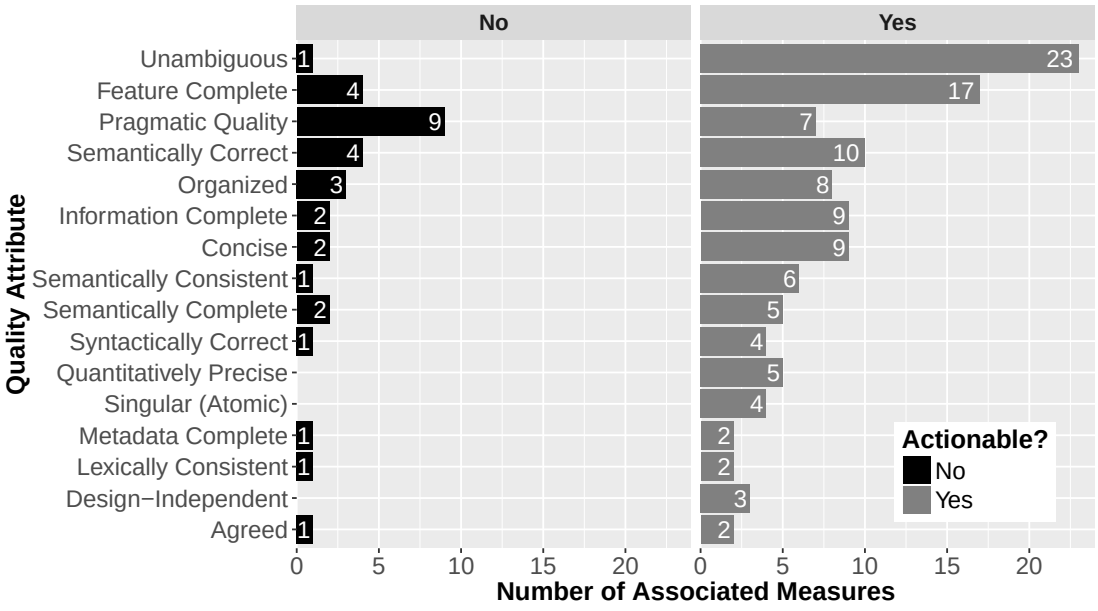


Figure 7.9.: Number of measures contained in the assessment model per quality attribute, classified according to their actionability

suggestions for improvements (e.g., a sentence which should be split into multiple requirements, one for each subject, metric B.41). The only quality attribute for which more measures lack recommendations for actions is pragmatic quality in general, i.e., not including its sub-attributes (e.g., unambiguous, concise). Interestingly, all nine metrics which lack actionability measure some kind of complexity, such as the Flesh-Kincaid Readability (metric B.33) which measures the complexity of natural-language texts based on sentences and word lengths.

Interpretation Since the clear majority of measures provide recommendations for actions, we can interpret that the assessment model provides adequate support for actionable results. This is in line with our results in [Méndez Fernández et al., 2014], in which we provided empirical evidence based on independent expert judgments that suggests that certain RE phenomena, including several of the attributes measured by the proposed assessment model, are actionable.

7.5. Discussion: Limitations of Measurability

Finally, in this section, we discuss our findings regarding the (limitations of) measurability of our assessment model on a more general note. Also, since it is based on a comprehensive literature review of the measures proposed up-to-date in the scientific literature,

we claim that it also reflects the current state-of-the-art in requirements engineering. Subsequently, we first discuss the measurability of each quality attribute individually before providing a short summary at the end of this section.

Semantic Quality

Semantic quality refers to the correspondence between the system desired by the stakeholders and the system described in the requirements specification, and includes the following attributes:

Semantic Correctness Measuring semantic correctness, while arguably one of the most important quality attributes of the requirements specification, requires significant costs while only able to provide limited results. Almost exclusively all measures require manual quality assessments, stakeholders to be involved in quality assessment, and quality managers to have extensive domain knowledge. Also, a substantial number of measures are based on a goal-oriented requirements engineering approach. Yet, no direct measures have been proposed; measuring correctness of requirements, i.e., that they define a system which is (will be) desired by the stakeholders is impossible. Therefore, most measures refer to approval of the stakeholders instead. Also, all measures are limited to requirements at a high-level of abstraction; no measures were proposed to assess the correctness/validity of goals, and the ones proposed are not well suited to assess details in the specifications (e.g., whether value ranges are correctly specified). Furthermore, while several measures assess whether important meta-data (e.g., the priority or source of a requirement) is specified, none check whether those information are indeed valid. On the upside, most of the measures provide assessments suited for case-by-case decision-making since while the results only specific aspects related to semantic correctness, they are quite reliable in assessing those. Moreover, some measures assessing the feature completeness of the specification leverage their results to also identify superfluous requirements, and thereby, are more cost-effective.

Feature Completeness The approaches for measuring feature completeness are quite similar to those regarding semantic correctness. However, in addition, several metrics based on the the number of (certain) steps of use-cases were proposed which purport to assess feature completeness. Those measures are, however, inherently flawed, and therefore better suited as indicators for manual follow-up tasks than to provide reliable decision-support on a case-by-case basis on their own.

Semantic Consistency In general, semantic consistency appears to be significantly easier to measure compared to semantic correctness and feature completeness. Studying the measures proposed, we observed quite diverse approaches; consistency is assessed based on logical approaches for formalized requirements, stakeholder feedback, mapping to and analysis of a domain ontology, and similarity of sentences based on linguistic techniques, therefore enabling a rather widespread applicability and catering to different uses.

Information and Meta-data Completeness In general, we argue that information and meta-data completeness is rather easy to measure given the right circumstances. For instance, if specific meta-models or templates are used, empty fields can be detected. Specifically for information completeness, Kenett [1996] proposed to measure several important information for downstream development (e.g., that conditions and constraints are specified), but relies on a manual classification. Furthermore, several metrics measure *stated* omissions, e.g., terms like *to-be-determined* or *TBD*.

Design-Independent and Quantitatively Precise While actually semantic properties, only measures based on dictionaries targeting specific terms (e.g., design or control flow terms, weak phrases) were proposed. Those measures seem to not be well suited as a sole means for decision-making on a case-by-case basis, but rather qualify as indicators for guiding subsequent inspections or if stochastic quality assurance is sufficient.

Boundary Precise We identified no measure which assess whether the requirements are indeed boundary precise, i.e., whether they are specified as stimuli and observations at the system's boundary or in its environment. While we expected this property to be hardly measurable, it is also often neglected in the literature and might therefore have no associated measures. In any case, future research should investigate this issue (see also Chap. 8.2).

Social Quality (Agreement)

In principle, agreement is easier to measure than semantic correctness since it refers to a system *stated* to be desired. Other than that, it has basically the same constraints, first and foremost, that it relies on manual quality assessments and stakeholder participation. To this end, according to our judgment, the measures propose sophisticated ways to assess that the stakeholders actually agree.

Syntactic Quality (Correctness)

Syntactic correctness is assessed by measuring to what extent words, punctuation, sentences and text passages confirm to dictionaries and grammatical rules. Besides requiring appropriate tooling (for the specific language), the proposed measures are both effective and efficient. However, we also expect syntactic correctness to have the least effect on downstream development.

Pragmatic Quality

Pragmatic quality refers to factors which influence the extent the requirements specification is understood by the audience, and includes the following sub-attributes:

Unambiguity Unambiguity seems to be one of the most studied and best measurable quality attributes of the requirements specification. Not only is it the quality

attribute for which the most measures were proposed, but they also cover many different aspects of it (including lexical, syntactic and anaphoric ambiguity), and rely on the most sophisticated techniques (e.g., the use of semantic dictionaries or machine-learning). The vast majority of measures is based on syntactic properties, applies to a single aspect or type of ambiguity only, requires natural-language specifications, and uses tools to automate measurement. However, several measures also assess ambiguity by means of human judgments; in those cases, the participants must resemble the actual readers of the specification as close as possible to provide valid results.

Organized While no (set of) measures are able to assess whether a requirements specification is organized in any aspect possible, several reasonable aspects appear to be indeed measurable, for instance, the coherence within and coupling between requirements (sections). The measures are to a large extent automated and only occasionally require input required a quality manager, with the exception of a checklist-based metric.

Concise According to the state-of-the-art, it is not possible to comprehensively assess whether a specification is concise, i.e., the syntax with the highest density is used. Instead, measurement is limited to the size, which however is an extreme oversimplification since it completely neglects the semantics, and one factor which reduces conciseness, i.e., redundancy. To measure those properties, various metrics have been proposed, which all have in common to be largely or exclusively based on tools.

Singular (Atomic) To assess whether a requirement is indeed singular (or atomic), two extreme approaches have been proposed. On the one hand, measuring the size of requirements in words is a very lightweight but grossly imprecise means. On the other hand, a manual evaluation of every single requirement based on (so-called unity) criteria requires high efforts.

Summary on Measurability

In general, we consider measurement of the requirements specification's semantic and social quality severely limited, at least according to the current understanding, since it causes significant costs while only being able to provide partial results. In particular, assessing whether the requirements are valid, agreed and cover all necessary features requires manual quality assessments, involvement of stakeholders, and extensive knowledge about the domain of the system.

In contrast, pragmatic quality strongly emphasizes the use of tools for automating measurement. To this end, many different measures assessing certain narrow properties of the specification were proposed, most of them based on linguistic techniques. Unfortunately, their capability to provide valid assessments differs among the attributes. While we can measure ambiguity, and to a lesser extent, organization of a requirements specification, the measures do not provide conclusive assessments whether it is concise

	Quality Attribute												
Factor	Semantic. Correct	Feature Complete	Information Complete	Metadata Complete	Quantitatively Precise	Design-Independent	Semantic. Consistent	Agreed	Syntac. Correct	Unambiguous	Organized	Singular (Atomic)	Concise
Tool Support		◐	◐	●	●	●	◐		●	●	●	◐	●
Manual QA	●	●	◐				●	●	◐	◐		◐	
Stakeholder Participation	●	◐						●					
Domain Knowledge	●	●	◐				◐	◐					
High Efforts	◐	◐						◐					
Goal-Oriented RE	◐	◐											

Table 7.3.: Factors impacting the measurability of quality attributes (● (virtually) mandatory, ◐ substantially enhances)

or singular. In this case, we see the sound use of measurement limited to provide further guidance (e.g., for reviews) or as part of stochastic quality control.

Finally, syntactic quality appears to be comprehensively and efficiently measurable. However, we expect it to have the least effect on downstream development.

We summarize the results of our discussion regarding (limitations) of measurability in Tbl. 7.3. The table reflects our judgment about the critical factors which impact the measurability of the requirements specification's quality attributes. Here, a ● denotes that a factor is (virtually) mandatory for measurement, while a ◐ denotes a substantial enhancement of measurability when present.

7.6. Summary

In this section we qualitatively studied to what extent the proposed assessment model is adequate in practice, in terms of applicability, validity, and actionability, by means of an exploratory (meta-)analysis of its individual measures.

Our results suggest that the main barriers for application are the tools that must be developed or acquired, the availability and training of staff, and the availability and participation of stakeholders during requirements engineering (quality assurance). Furthermore, we confirmed that measures are highly susceptible to threats to validity, both

manual ones based on human judgments as well as automated ones, can often only provide valid results for specific attributes of the specification, and may hence be better used as guidance than an actual assessment. Finally, the vast majority of measures is able to provide recommendations for actions in addition to an assessment.

Consequently, we believe that whether measurement-based quality assessment in its current form is a valuable tool or not depends on the specific context of use, a profound understanding of the project participants about the measures' validity, their careful use for decision-making, and whether the requirements engineers can make use of the provided recommendations for action.

Epilog

8 Chapter

Conclusions and Outlook

Finally, in this chapter, we first summarize the contributions of this thesis and discuss their conclusions in Sec. 8.1 before discussing possible directions for future research in Sec. 8.2.

8.1. Conclusions

In the initial problem statement (Sec. 1.1), we argued that we need a better understanding of if, when, and what to measure in order to assess the quality of the requirements specification, and a method to carefully and soundly interpret the measurement results. Subsequently, we summarize our results and contributions and outline our conclusions regarding this problem statement according to the three topics identified.

8.1.1. Problem 1: Significance of Documented Requirements

We argued that the decision about *if and when to measure* the quality of requirements specifications requires a profound understanding of the significance of documented requirements. Yet, the question how much downstream development activities are actually determined by the quality of the documented requirements, which also includes the question how much project participants eventually rely on the created artifacts, is not completely answered.

Contributions In chapter 3, we studied two complementary research objectives, namely the extent requirements specifications are created and used, and the impact of quality

defects on downstream development, by means of a survey with practitioners and a controlled quasi-experiment, respectively. Regarding the significance of the artifact itself, we found that the requirements specification is commonly used to document and communicate requirements in practice, especially for project participants not involved during requirements engineering. Furthermore, we observed that the extent of documentation and its use for communication correlates with certain project criteria. Specifically, our results suggest that requirements are documented in particular for safety/security-critical systems or if release cycles are rather long. In contrast, if requirements are volatile, the development team is rather small, has a low turnover and already successfully collaborated in the past, requirements are more likely to be less extensively documented and used for communication.

In addition, we studied the complementary question if and how much certain quality defects in requirements specifications actually impact the efficiency and effectiveness of downstream development activities. Results obtained from a quasi-experiment suggest that incorrect statements in specifications, even if rather obvious, substantially impact both the difficulty of testing and the quality of the test-cases obtained. Specifically, incorrect statements in specifications resulted in flawed test-cases in 47% (apparent defect, i.e., evident from the specification alone) respectively 100% (hidden defect/not evident from the specification alone) of the times, with no measurable influence of prior briefing of participants about the system's domain. In contrast, we were unable to show that negated sentences influence system testing contrary to popular proposition (e.g., Femmer et al. [2014c], Génova et al. [2013]).

Conclusions The empirical evidence revealed that measuring the quality of documented requirements is neither mandatory nor necessary in absolute terms but instead depends on the context of use. Hence, we must not ask *if* but *when* to measure it. We see this result as important in itself for two reasons. First, it questions approaches which unconditionally advocate or reject quality assurance of requirements specifications. Second, it motivates future research in requirements engineering to pay more attention to the subtle difference between (elicited) requirements in general and its documentation in particular.

We also addressed the question *when* to measure by providing an initial set of criteria which correlate with the extent requirements are documented and used for communication. While we consider those criteria as helpful for practitioners seeking to design future or question present quality assurance approaches, our results also raise novel research questions. We identified weak to moderate correlations for several others of the criteria proposed by Kalus and Kuhrmann [2013], which, as discussed in Sec. 3.3 (pp. 56) in more detail, potentially stand in complex interplay with each other. Therefore, we would encourage to study relationships among the context criteria and investigate their joint impact as the next step to obtain a comprehensive model of requirements specification significance.

8.1.2. Problem 2: Lack of Precise and Well-founded Quality Definition

We argued that we lack a precise and well-founded quality definition which obfuscates *what should be measured* and results in not only neglecting measurement of crucial characteristics but also, and arguably more harmful, in measuring irrelevant characteristics.

Contributions In chapter 4, we presented two approaches for defining the quality of requirements specifications. First, we proposed the ABRE-QM approach, which applies the idea of activity-based quality notions to requirements engineering. More specifically, the approach consists of a meta-model and a complementary process proposal to identify intrinsic properties of the requirements specification which impact the quality-in-use, most prominently, the effectiveness and efficiency, by analyzing downstream development activities.

Second, we provided a quality definition model in terms of intrinsic quality attributes, i.e., properties inherently associated with the requirements specification itself. In total, our quality definition model consists of 72 unique quality attributes on various levels of abstraction. This set of attributes is comprehensive and consistent since it is based on a systematic study of quality models proposed in the current body of knowledge, and organized in terms of a hierarchical taxonomy based on identified specialization relationships among attributes. Moreover, high-level attributes are decomposed into low-level attributes which specify the precise content of the requirements specification they apply to as well as the concrete properties which are considered. Also, the quality attributes are well-founded since their impact on downstream development is made explicit. To obtain a context-specific quality definition, we provided a customization procedure, essentially based on a careful selection and further refinement of the model's quality attributes.

Conclusions We have shown that the quality of requirements specifications (i) is no end in itself but must be defined with the activities it is used as an input in mind, and (ii) depends on the actual context of use. To this end, we have provided two alternative means to achieve this. First, by applying an inductive approach in analyzing the activities and specific context of use in a systematic way (ABRE-QM), and second, by starting from a quality (reference) model and adapting it by carefully selecting and refining its quality attributes. While both approaches eventually result in a context-specific model, we associate different strengths and weaknesses with each. We expect an inductive analysis following the ABRE-QM approach to be more expensive but yield precisely those quality attributes considered most important by project participants, and therefore achieving a high commitment. In contrast, we expect the tailoring of a quality reference model to be less expensive, more comprehensive, and better suited if the context is unknown or highly uncertain. In both cases, the resulting quality definition models are context-specific and oriented towards quality-in-use. Consequently, such models provide a more precise and well-founded (valid) basis for quality assurance of requirements specification.

8.1.3. Problem 3: Lack of Understanding about Measurability and its Limitations

Without a thorough knowledge of measures, in particular their applicability and validity, we lack a sound understanding of *what can be measured*. This leads to situations in practice in which the attribute perceived to be measured differs from the one actually measured, leading to false conclusions and, in turn, poor decision-making. Ideally, this would be prevented by a systematic approach to quality assessment which supports a careful and sound interpretation of measurement results.

Contributions In chapter 5.1 we proposed a framework for the assessment of requirements specification quality which consists of a quality assessment meta-model and a complementary process model. The meta-model ensures that quality assessment models describe measures holistically. In particular, the meta-model enforces to explicitly specify the relationship between the measured and assessed attributes, to provide a procedure to interpret the results of data collection regarding decision-making and corrective actions, and to specify circumstances anticipated to threat the measures' validity. We conducted a qualitative study in which practitioners confirmed the utility of this meta-model. The complementary process model describes and organizes activities for a systematic and sound application of such a quality assessment model. It is organized according to three phases, namely preparation, assessment, and continuous improvement. The phases determine the quality demand and implement suitable measures, perform individual assessments, and provide means to constantly improve the assessment model, respectively.

Based on this framework, we proposed a quality assessment model based on state-of-the-art measures in chapter 6. To obtain this model, we conducted a systematic literature review and in-depth analysis which resulted in 136 metrics being embedded in a unified assessment model. In particular, the obtained assessment model provides a refined view (i) on the relationship between the qualities defined and the attributes actually measured by the individual metrics, (ii) the measured scope of the specification artifact in terms of content items and levels of abstraction, and (iii) the prerequisites required to apply those metrics. Finally, in chapter 7, we evaluated the adequacy of the quality assessment model, and thereby the limitations of measurability, by means of a meta-analysis. Therein, we revealed that certain quality attributes remain immeasurable in most or almost all circumstances in practice. For instance, we lack measures for assessing the extent that requirements are boundary-precise, i.e., specified as stimuli and observations at the system's boundary rather than in its environment, must spend high efforts to assess semantic correctness or feature completeness, or lack reliable procedures to interpret measured values for metrics attributed to the specification's understandability.

Conclusions Overall, we claim that our contributions provide a more precise and holistic understanding of what is measurable, and that the systematic approach to measurement of requirements specification quality provides the necessary guidance to apply measures more appropriately. In particular, we are optimistic that by carefully following

our approach the measurement of unnecessary (irrelevant) attributes of the specification can be avoided and the application of suitable, i.e., applicable and sound, measures can be achieved. However, while the total number of 136 measures appear comprehensive at first, we also revealed that all measures require some prerequisites (most commonly, that requirements are specified in natural-language, a certain tool is used, or that at least some steps are carried our manually), may lack validity, or do not holistically measure the quality attribute purported to be measured. Hence, while we provide an explicit understanding of quality attributes which are immeasurable, at least in terms of the current understanding in the requirements engineering research community, our work raises two novel questions: namely (1) whether the subset of quality attributes assessable by measurement is indeed sufficient to justify the efforts of measurement, and (2), if and which complementary means can be used in addition to measurement of requirements specifications to obtain a (more) complete assessment. Regarding the latter, we discuss one potential direction in terms of process assessment in the following section. Finally, we recognize the limitations of relying on a single research method to evaluate our model. Consequently, we would encourage future research in this direction, in particular, in terms of in-vivo case studies. We see the potential of our quality assessment model and process to be applied both in academia, e.g., to assess and compare existing and novel RE approaches based on a common benchmark, as well as in industry, e.g., to assess concrete specifications and thereby hopefully provide empirical evidence in terms of case studies.

8.2. Outlook

In this section, we now outline directions for future research and potential improvements to the contributions in this thesis.

8.2.1. Quality Definition

Peculiarities of Software-Intensive Systems Strictly speaking, the quality definition model presented in Chap. 4 tackles activities of software development only. However, software-intensive systems are not composed of software alone but comprise other parts, e.g., sensors, actuators, computation and communication hardware. Therefore, the development of software-intensive systems typically relies on augmented or additional activities to ensure all components jointly provide the intended features and characteristics. For example, a hazard analysis ensures that the system as a whole operates safely within its environment (see, e.g., [Ericson et al., 2015] for an overview on techniques and methods). We expect that those activities and their resulting artifacts require properties of the requirements specification not yet represented in the quality definition model. We envision future research to overcome this limitations by studying the activities of systems engineering and its sub-fields (e.g., safety, reliability, or performance engineering) to identify new quality attributes as well as provide explicit impacts of already represented attributes. Similar to our analysis of the Unified Process, systems engineering method-

ologies could be used as a starting point (see, e.g., Estefan et al. [2007] for an overview). Promising candidates are the NASA [2007] approach, the extension of the Rational Unified Process to systems engineering [Cantor and Plug, 2003], and the approach resulting from the Software Platform Embedded Systems (SPES) project [Pohl et al., 2012].

Agile Development Methodologies In contrast to requirements specifications written in traditional approaches, the artifacts created and used in methodologies referred to as 'agile', such as eXtreme Programming (XP) [Beck, 2000] or Scrum [Schwaber and Beedle, 2001], are quite different. In Scrum, for instance, requirements are specified in terms of user stories and collectively stored in the product backlog. Furthermore, comprehensive internal documentation such as the requirements specifications is advocated to be replaced with face-to-face communication where possible, as stated in the agile manifesto and its fundamental principles[1]. Since those methodologies have gained wide-spread attention and acceptance over the last decade [West et al., 2010], extending the quality definition model to activities and artifacts specific to those methodologies would benefit researchers and practitioners alike. Recently, a first step in this direction was done by Heck and Zaidman [2016], who performed a systematic literature review on quality criteria for agile requirements specifications. The results indicate to confirm our expectations; most quality attributes are already contained in the quality definition model presented in Chap. 4, attesting the universal nature of attributes such as *concise*, *rationale provided* or *consistent vocabulary*, while some are indeed specific to artifacts in agile development (e.g., that *functionality is specified in story name*, *structured story cards* are used). However, due to the study design, the quality attributes remain at a high level of abstraction and the impact on development activities is not given, which we see as subject to future research.

Inter-Attribute Relations We agree with Broy [2006] that a comprehensive understanding of quality includes a theory of relationships between quality attributes. While Knee [2006, pp. 80] discusses causality and mutual exclusivity among many of the attributes also identified in this thesis, we recognize the need to further extend this research. It remains an open question which types and associated characteristics of relationships among quality attributes are relevant to define quality. Furthermore, we must ideally also understand the conditions under which those relationships apply to concrete attributes, or if infeasible, their probabilities based on empirical evidence. Finally, from the point of view of measurement-based assessment, those relationships need to be quantified. This would allow to express the magnitude of one attribute by the magnitude of a different attribute, which would be particularly useful for attributes related to semantic quality given its importance on the one hand and its limitations in measurement on the other hand.

[1] `agilemanifesto.org`

8.2.2. Measurement-Based Quality Assessment

Novel Measures Throughout this thesis, we identified several opportunities for novel measures. First, by studying the body of knowledge of current quality measures, we observed that several authors independently worked on closely related attributes or employed similar techniques, as documented in the quality assessment model presented in Chap. 6. Future work may study if suitable combinations of different measures and techniques for the same attribute can provide additional benefits compared to applying the measures individually.

Second, by comparing the quality definition with the measures proposed (see Chap. 7), we identified attributes which are hardly or not at all covered by measures. For instance, no measure was specified for the quality attribute *boundary precise*, i.e., that the system's capabilities and characteristics are described at the boundary between the system and its context (see Sec. 4.4.1, pp. 79). Indeed, we found this particular observation surprising and disappointing given the attention paid by the scientific community to the work by Jackson [1995, 2001].

Last but not least, we envision to also integrate process quality measures into the assessment model, since the requirements specification's quality is not given per se but determined by various activities which ultimately led to its creation. Exemplary attributes of the process are, e.g., the number of changes to requirements over time (sometimes referred to as stability or volatility), the number of issues raised in each requirements activity [Costello and Liu, 1995], or if requirements were negotiated or not [Ahmad and Asmai, 2016]. While an integrated model requires to relate process qualities to requirements specification qualities, which we see as very ambitious due to the associated uncertainty and high number of confounding factors, we see the potential to obtain measures for quality aspects which are hard to measure based on the artifact alone, e.g., semantic correctness.

Opportunities for Tool Support One particular aspect of implementing a measurement program in practice which is recognized as a success factor is the use of tools [Hall and Fenton, 1997]. Regarding the approach presented in this thesis, we see two distinct areas which can benefit from tool support. First, the metrics' individual data collection procedures can greatly benefit from tool support for several reasons. By automating data collection, the manual overhead of measurement can be significantly reduced, while at the same time, objectivity of the results can be increased. The saved effort and increased trust in the results can in turn contribute to a higher acceptance of the measurement program, which by itself is a success factor. Regarding this matter, our evaluation suggests room for improvement concerning the current measures. While exactly every second metric requires manual steps to be performed during assessment (see Fig. 7.2, p. 187), only 31.7% actually measure a semantic property of the specification (see Fig. 7.5, p. 190). Studying the measures in question more closely, we observed that many of those were proposed over a decade ago and relied on manual annotations and classifications of syntactic (e.g., conditions or attributes [Kenett, 1996]) or semantic elements (e.g., distinguish between prescriptive and descriptive statements [Kim et al.,

1999]), or used manual false positive elimination (e.g., Fabbrini et al. [2000]). Yet, advances in natural-language processing and the continuous improvement of the tools[2] over the last decades have not only rendered many of those steps unnecessary, but also lead to more sophisticated measures based on lightweight semantic NLP techniques (e.g., Yang et al. [2011], Femmer et al. [2014a]). In particular, the combination of machine-learning with advanced natural-language processing techniques provided remarkably accurate results in detecting *noxious* ambiguity, while at the same time reducing manual efforts to judgments within a single training phase (per context, e.g., an organizational unit or business sector) instead of being carried for each assessment. We would encourage future research to apply the latest technological advances in natural-language processing, machine learning and artificial intelligence to requirements measures and to explore ways of increasing efficiency and accuracy.

Despite individual measures, we also see benefits of tool support for the proposed process as a whole. We envision an assistance software tool which supports the process engineer and quality manager in tailoring the quality assessment model for a particular context. In particular, we see a strong use-case for a browsable metrics database and a step-by-step assistant similar to those proposed for software process tailoring, e.g., by Kalus [2013]. To this end, we could also imagine the use of recommender systems (see Adomavicius and Tuzhilin [2005] for an overview of the state-of-the-art) in assisting to the user in this task to avoid problematic configurations as well as recommend complementary measures based on an experience database. Last but not least, according to our experiences, a dashboard to visualize the assessment results, both the current state and over time, is the preferred form of reporting among practitioners.

Suitability Criteria for Measurement Research in measuring the quality of requirements specification was until now limited to studying individual measures. However, by providing our assessment model and approach, we obtained the machinery to evaluate multiple measures in a unified way. Based on this approach, we envision to study the application of a set of measures within programs. By evaluating those measures, e.g., during retrospective workshops as proposed for the continuous improvement phase in the assessment process (Sec. 5.2.3, pp.120), we suggest to study which measures worked and which did not in order to identify "suitability criteria", i.e., commonalities among the successful respectively unsuccessful ones. Those suitability criteria would enable us to further understand the practical limitations of measurability for requirements specification quality, develop better measures and select more suitable ones.

8.2.3. Learning from Measurement

So far, we have considered measurement only from an assessment perspective, i.e., we applied measures in order to obtain an estimation of its quality. However, measurement in its widest sense, i.e., proposing, applying and discussion measures as well as

[2]Most notably during our study was the widespread use of the Stanford NLP tools available at `nlp.stanford.edu/software`, in particular the parser, NLP tagger and CoreNLP suite, and the WordNet lexical (semantic) dictionary available at `wordnet.princeton.edu`

comparing the outcomes of measures with the empirical observation of the requirements specification, can also benefit learning what actually constitutes quality.

In this regards, we believe the quality assessment model proposed in this thesis could provide a valuable contribution. We see two interesting directions. First, since we studied the attributes purported to be measured as well as the ones actually measured, we identified several attributes which are related but not necessarily equal to the model's quality attributes. For instance, we identified additional (measurable) sub-aspects of unambiguity, such as syntactic, lexical, or anaphoric ambiguity, as described in Sec. 6.5.4). By studying those relationships in more detail, we could gain a better understanding of what actually constitutes a quality attribute under which circumstances.

Second, since the quality attributes identified in the assessment model are quantitatively specified, the associated scales may be seen as candidate scales for the quality attributes described in the quality definition model (Chap. 4). Based on those candidate scales, we would encourage future work to study if they qualify as representations of those quality attributes for the field of requirements engineering, or if they can guide discussion in obtaining reference scales for quality attributes. While we admit that this is no short-term goal, we yet stress the importance of such reference scales for the field; only if we have widely-accepted reference scales for core qualities of our most important artifact, we can consistently evaluate the performance of our techniques and methods and align our future work accordingly.

Bibliography

Ulrike Abelein and Barbara Paech. State of practice of user-developer communication in large-scale it projects. In *Requirements Engineering: Foundation for Software Quality*, pages 95–111. Springer, 2014.

Steve Adolph, Alistair Cockburn, and Paul Bramble. *Patterns for effective use cases*. Addison-Wesley Longman Publishing Co., Inc., 2002.

Gediminas Adomavicius and Alexander Tuzhilin. Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. *IEEE transactions on knowledge and data engineering*, 17(6):734–749, 2005.

Nitin Agarwal and Urvashi Rathod. Defining success for software projects: An exploratory revelation. *International journal of project management*, 24(4):358–370, 2006.

Sabrina Ahmad and Siti Azirah Asmai. Measuring software requirements quality following negotiation through empirical study. *International Journal of Applied Engineering Research*, 11(6):4190–4196, 2016.

Robert D. Austin. *Measuring and Managing Performance in Organizations*. Dorset House Publishing, New York, 1996. ISBN 978-0-932633-36-1.

SIG Automotive. Automotive spice process assessment model. *Final Release, Version 3.0*, 4:46, 2015.

V. R. Basili and H. D. Rombach. Tailoring the software process to project goals and environments. In *Proceedings of the 9th International Conference on Software Engineering*, ICSE '87, pages 345–357, Los Alamitos, CA, USA, 1987. IEEE Computer Society Press. ISBN 0-89791-216-0. URL `http://dl.acm.org/citation.cfm?id=41765.41804`.

Victor R Basili, Gianluigi Caldiera, and H Dieter Rombach. The goal question metric approach. *Encyclopedia of software engineering*, 2:528–532, 1994.

Kent Beck. *Extreme Programming Explained: Embrace Change*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 2000. ISBN 0-201-61641-6.

Yoav Benjamini and Yosef Hochberg. Controlling the false discovery rate: a practical and powerful approach to multiple testing. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 289–300, 1995.

Brian Berenbach and Gail Borotto. Metrics for model driven requirements development. In *Proceedings of the 28th international conference on Software engineering*, pages 445–451. ACM, 2006.

Beatriz Bernárdez, Amador Durán, and Marcela Genero. Empirical evaluation and review of a metrics-based approach for use case verification. *Journal of Research and Practice in Information Technology*, 36(4):247–258, 2004a.

Beatriz Bernárdez, Marcela Genero, M Toro, et al. A controlled experiment for evaluating a metric-based reading technique for requirements inspection. In *Software Metrics, 2004. Proceedings. 10th International Symposium on*, pages 257–268. IEEE, 2004b.

Daniel M Berry, Antonio Bucchiarone, Stefania Gnesi, Giuseppe Lami, and Gianluca Trentanni. A new quality model for natural language requirements specifications. In *Proceedings of the international workshop on requirements engineering: foundation of software quality (REFSQ)*, 2006.

D.M. Berry and E. Kamsties. The dangerous ldquo;all rdquo; in specifications. In *Software Specification and Design, 2000. Tenth International Workshop on*, pages 191–193, 2000. doi: 10.1109/IWSSD.2000.891140.

Elizabeth Bjarnason, Krzysztof Wnuk, and Björn Regnell. Requirements are slipping through the gaps?a case study on causes & effects of communication gaps in large-scale software development. In *Requirements Engineering Conference (RE), 2011 19th IEEE International*, pages 37–46. IEEE, 2011.

Jorgen Boegh, Stefano Depanfilis, Barbara Kitchenham, and Alberto Pasquini. A method for software quality planning, control, and evaluation. *IEEE software*, 16(2):69, 1999.

K. Boness, A. Finkelstein, and R. Harrison. A lightweight technique for assessing risks in requirements analysis. *Software, IET*, 2(1):46–57, February 2008. ISSN 1751-8806.

Kenneth Boness, Anthony Finkelstein, and Rachel Harrison. A method for assessing confidence in requirements analysis. *Information and Software Technology*, 53(10):1084 – 1096, 2011. ISSN 0950-5849. doi: http://dx.doi.org/10.1016/j.infsof.2011.05.003. URL `http://www.sciencedirect.com/science/article/pii/S0950584911001054`. Special Section on Mutation Testing.

George EP Box. Robustness in the strategy of scientific model building. *Robustness in statistics*, 1:201–236, 1979.

Manfred Broy. Requirements engineering as a key to holistic software quality. In Albert Levi, Erkay Savaş, Hüsnü Yenigün, Selim Balcısoy, and Yücel Saygın, editors, *Computer and Information Sciences – ISCIS 2006: 21th International Symposium, Istanbul, Turkey, November 1-3, 2006. Proceedings*, pages 24–34, Berlin, Heidelberg, 2006. Springer Berlin Heidelberg. ISBN 978-3-540-47243-8. doi: 10.1007/11902140_3. URL `http://dx.doi.org/10.1007/11902140_3`.

Egon Brunswik. Perception and the representative design of experiments, 1956.

A. Cailliau and A. van Lamsweerde. A probabilistic framework for goal-oriented risk analysis. In *Requirements Engineering Conference (RE), 2012 20th IEEE International*, pages 201–210, Sept 2012. doi: 10.1109/RE.2012.6345805.

Murray Cantor and RUP Plug. Rational unified process for systems engineering: Part 1 – introducing rup se version 2.0. *The Rational Edge (August 2003)*, 2003.

Samira Si-Said Cherfi, Jacky Akoka, and Isabelle Comyn-Wattiau. Conceptual modeling quality-from eer to uml schemas evaluation. In *Conceptual Modeling – ER 2002*, pages 414–428. Springer, 2002.

Samira Si-said Cherfi, Jacky Akoka, and Isabelle Comyn-Wattiau. Perceived vs. measured quality of conceptual schemas: An experimental comparison. In *Tutorials, Posters, Panels and Industrial Contributions at the 26th International Conference on Conceptual Modeling - Volume 83*, ER '07, pages 185–190, Darlinghurst, Australia, Australia, 2007. Australian Computer Society, Inc. ISBN 978-1-920682-64-4. URL `http://dl.acm.org/citation.cfm?id=1386957.1386987`.

Alicja Ciemniewska, Jakub Jurkiewicz, Lukasz Olek, and Jerzy Nawrocki. Supporting use-case reviews. In Witold Abramowicz, editor, *Business Information Systems*, volume 4439 of *Lecture Notes in Computer Science*, pages 424–437. Springer Berlin Heidelberg, 2007. ISBN 978-3-540-72034-8. doi: 10.1007/978-3-540-72035-5_33. URL `http://dx.doi.org/10.1007/978-3-540-72035-5_33`.

Alistair Cockburn. Basic use case template. *Humans and Technology, Technical Report*, 96, 1998.

Rita J. Costello and Dar-Biau Liu. Metrics for requirements engineering. *Journal of Systems and Software*, 29(1):39 – 63, 1995. ISSN 0164-1212. doi: http://dx.doi.org/10.1016/0164-1212(94)00127-9. URL `http://www.sciencedirect.com/science/article/pii/0164121294001279`. Oregon Metric Workshop.

Daniela Damian and James Chisan. An empirical study of the complex relationships between requirements engineering processes and other processes that lead to payoffs in productivity, quality, and risk management. *IEEE Transactions on Software Engineering*, 32(7):433–453, 2006.

Daniela E Damian and Didar Zowghi. Re challenges in multi-site software development organisations. *Requirements engineering*, 8(3):149–160, 2003.

A. Davis, S. Overmyer, K. Jordan, J. Caruso, F. Dandashi, A. Dinh, G. Kincaid, G. Ledeboer, P. Reynolds, P. Sitaram, A. Ta, and M. Theofanos. Identifying and measuring quality in a software requirements specification. In *Software Metrics Symposium, 1993. Proceedings., First International*, pages 141–152, May 1993a. doi: 10.1109/METRIC.1993.263792.

Alan Davis, Scott Overmyer, Kathleen Jordan, Joseph Caruso, Fatma Dandashi, Anhtuan Dinh, Gary Kincaid, Glen Ledeboer, Patricia Reynolds, Pradhip Sitaram, et al. Identifying and measuring quality in a software requirements specification. In *Software Metrics Symposium, 1993. Proceedings., First International*, pages 141–152. IEEE, 1993b.

Florian Deissenboeck, Stefan Wagner, Markus Pizka, Stefan Teuchert, and Jean-François Girard. An activity-based quality model for maintainability. In *Software Maintenance, 2007. ICSM 2007. IEEE International Conference on*, pages 184–193. IEEE, 2007.

Tom DeMarco. *Controlling software projects: Management, measurement, and estimates.* Prentice Hall PTR, 1986.

Chao Y. Din. *Requirements Content Goodness and Complexity Measurement Based On NP Chunks.* VDM Verlag, Saarbruecken, Germany, 2008. ISBN 383649888X, 9783836498883.

Julio Cesar Sampaio do Prado Leite, Gustavo Rossi, Federico Balaguer, Vanesa Maiorana, Gladys Kaplan, Graciela Hadad, and Alejandro Oliveros. Enhancing a requirements baseline with scenarios. *Requirements Engineering*, 2(4):184–198, 1997.

A. Duran, A. Ruiz-Cortes, R. Corchuelo, and M. Toro. Supporting requirements verification using xslt. In *Requirements Engineering, 2002. Proceedings. IEEE Joint International Conference on*, pages 165–172, 2002. doi: 10.1109/ICRE.2002.1048519.

K. El Emam and N.H. Madhavji. Measuring the success of requirements engineering processes. In *Requirements Engineering, 1995., Proceedings of the Second IEEE International Symposium on*, pages 204–211, Mar 1995a. doi: 10.1109/ISRE.1995.512562.

Khaled El Emam and Nazim H Madhavji. Measuring the success of requirements engineering processes. In *Requirements Engineering, 1995., Proceedings of the Second IEEE International Symposium on*, pages 204–211. IEEE, 1995b.

Khaled El Emam and NazimH. Madhavji. An instrument for measuring the success of the requirements engineering process in information systems development. *Empirical Software Engineering*, 1(3):201–240, 1996. ISSN 1382-3256. doi: 10.1007/BF00127446. URL `http://dx.doi.org/10.1007/BF00127446`.

Clifton A Ericson et al. *Hazard analysis techniques for system safety.* John Wiley & Sons, 2015.

P. Espada, M. Goulao, and J. Araujo. Measuring complexity and completeness of kaos goal models. In *Empirical Requirements Engineering (EmpiRE), 2011 First International Workshop on*, pages 29–32, Aug 2011. doi: 10.1109/EmpiRE.2011.6046252.

Patrícia Espada, Miguel Goulão, and João Araújo. A framework to evaluate complexity and completeness of kaos goal models. In Camille Salinesi, MoiraC. Norrie, and Oscar Pastor, editors, *Advanced Information Systems Engineering*, volume 7908 of *Lecture Notes in Computer Science*, pages 562–577. Springer Berlin Heidelberg, 2013. ISBN 978-3-642-38708-1. doi: 10.1007/978-3-642-38709-8_36. URL `http://dx.doi.org/10.1007/978-3-642-38709-8_36`.

S. Espana, N. Condori-Fernandez, A. Gonzalez, and O. Pastor. Evaluating the completeness and granularity of functional requirements specifications: A controlled experiment. In *Requirements Engineering Conference, 2009. RE '09. 17th IEEE International*, pages 161–170, Aug 2009. doi: 10.1109/RE.2009.33.

Jeff A Estefan et al. Survey of model-based systems engineering (mbse) methodologies. *Incose MBSE Focus Group*, 25(8), 2007.

Anne Etien and Colette Rolland. Measuring the fitness relationship. *Requirements Engineering*, 10(3):184–197, 2005.

F. Fabbrini, M. Fusani, S. Gnesi, and G. Lami. Quality evaluation of software requirement specifications. In *Proceedings of the Software and Internet Quality Week 2000 Conference*, pages 1–18, 2000. URL `http://fmt.isti.cnr.it/WEBPAPER/paper8A2.pdf`.

Fabrizio Fabbrini, Mario Fusani, Vincenzo Gervasi, Stefania Gnesi, and Salvatore Ruggieri. On linguistic quality of natural language requirements. In *REFSQ*, pages 57–62, 1998.

A. Fantechi, S. Gnesi, G. Lami, and A. Maccari. Application of linguistic techniques for use case analysis. In *Requirements Engineering, 2002. Proceedings. IEEE Joint International Conference on*, pages 157–164, 2002. doi: 10.1109/ICRE.2002.1048518.

Henning Femmer, Daniel Méndez Fernández, Elmar Juergens, Michael Klose, Ilona Zimmer, and Jörg Zimmer. Rapid requirements checks with requirements smells: Two case studies. In *Proceedings of the 1st International Workshop on Rapid Continuous Software Engineering*, RCoSE 2014, pages 10–19, New York, NY, USA, 2014a. ACM. ISBN 978-1-4503-2856-2. doi: 10.1145/2593812.2593817. URL `http://doi.acm.org/10.1145/2593812.2593817`.

Henning Femmer, Jan Kucera, and Antonio Vetrò. On The Impact of Passive Voice Requirements on Domain Modelling. In *International Symposium on Empirical Software Engineering and Measurement (ESEM)*, 2014b. ISBN 9781450327749.

Henning Femmer, Daniel Méndez Fernández, Elmar Juergens, Michael Klose, Ilona Zimmer, and Jörg Zimmer. Rapid Requirements Checks with Requirements Smells: Two

Case Studies. In *Proc. of the 36th International Conference on Software Engineering (ICSE'14)*, 2014c.

Henning Femmer, Jakob Mund, and Daniel Méndez Fernández. It's the Activities, Stupid! A New Perspective on RE Quality. In *Proc. of the 37th International Conference on Software Engineering (ICSE'15)*, 2015.

Norman E Fenton and Shari Lawrence Pfleeger. *Software metrics: a rigorous and practical approach*. PWS Publishing Co., 1998.

Alessio Ferrari, Felice dell Orletta, Giorgio Oronzo Spagnolo, and Stefania Gnesi. Measuring and improving the completeness of natural language requirements. In *International Working Conference on Requirements Engineering: Foundation for Software Quality*, pages 23–38. Springer, 2014.

American Society for Quality. Quality Glossary - Q. `http://asq.org/glossary/q.html`, accessed 02/23/16. URL `http://asq.org/glossary/q.html`.

Kevin Forsberg and Harold Mooz. The relationship of system engineering to the project cycle. In *INCOSE International Symposium*, volume 1, pages 57–65. Wiley Online Library, 1991.

Helle Damborg Frederiksen and Lars Mathiassen. Information-centric assessment of software metrics practices. *IEEE Transactions on engineering management*, 52(3):350–362, 2005.

Eva Geisberger. *Requirements Engineering eingebetteter Systeme: Ein interdisziplinärer Modellierungsansatz*. Shaker, 2005.

Gonzalo Génova, José M Fuentes, Juan Llorens, Omar Hurtado, and Valentín Moreno. A framework to measure and improve the quality of textual requirements. *Requirements Engineering*, 18(1):25–41, 2013.

Mark P. Ginsberg and Lauren H. Quinn. Process tailoring and the the software capability maturity model. Technical Report CMU/SEI-94-TR-024, Carnegie Mellon University, 11 1995.

Martin Glinz. A glossary of requirements engineering terminology. *Standard Glossary of the Certified Professional for Requirements Engineering (CPRE) Studies and Exam*, 1, 2011.

Martin Glinz and Roel J Wieringa. Guest editors' introduction: stakeholders in requirements engineering. *IEEE Software*, 24(2):18–20, 2007.

Tony Gorschek and Alan M Davis. Requirements engineering: In search of the dependent variables. *Information and Software Technology*, 50(1):67–75, 2008.

Tracy Hall and Norman Fenton. Implementing effective software metrics programs. *IEEE software*, 14(2):55, 1997.

Petra Heck and Andy Zaidman. A systematic literature review on quality criteria for agile requirements specifications. *Software Quality Journal*, pages 1–34, 2016. ISSN 1573-1367. doi: 10.1007/s11219-016-9336-4. URL `http://dx.doi.org/10.1007/s11219-016-9336-4`.

Gabriel D Hoffman. Early introduction of software metrics. In *Aerospace and Electronics Conference, 1989. NAECON 1989., Proceedings of the IEEE 1989 National*, pages 559–563. IEEE, 1989.

Hubert F Hofmann and Franz Lehner. Requirements engineering as a success factor in software projects. *IEEE software*, 18(4):58–66, 2001.

Jonny Holmström and Steven Sawyer. Requirements engineering blinders: exploring information systems developers' black-boxing of the emergent character of requirements. *European Journal of Information Systems*, 20(1):34–47, 2011.

Frank Houdek and Barbara Paech. Das türsteuergerät-eine beispielspezifikation. *IESE-Report Nr*, 2, 2002.

C. Huertas, M. Gomez-Ruelas, R. Juarez-Ramirez, and H. Plata. A formal approach for measuring the lexical ambiguity degree in natural language requirement specification: Polysemes and homonyms focused. In *Uncertainty Reasoning and Knowledge Engineering (URKE), 2011 International Conference on*, volume 1, pages 115–118, Aug 2011. doi: 10.1109/URKE.2011.6007860.

IEEE Standard 1471. *IEEE Recommended Practice for Architectural Description for Software-Intensive Systems*. Institute of Electrical and Electronics Engineers, 2000.

IEEE Standard 610.12. *Standard Glossary of Software Engineering Terminology*. Institute of Electrical and Electronics Engineers, 1990.

IEEE Standard 830. Ieee guide to software requirements specifications. *IEEE Std 830-1998*, 1998.

ISO/IEC 15939:2007. *Systems and Software engineering – Measurement Process*. International Organization for Standardization, Geneva, Switzerland, 2007.

ISO/IEC 25010:2010. *Systems and Software Engineering – Systems and Software Product Quality Requirements and Evaluation (SQuaRE) – System and Software Quality Models*. International Organization for Standardization, Geneva, Switzerland, 2010.

ISO/IEC 29148:2011. *Systems and Software Engineering – Life Cycle Processes – Requirements Engineering*. International Organization for Standardization, Geneva, Switzerland, 2011.

ISO/IEC 9126-1. *Software engineering – Product quality – Part 1: Quality model*. International Organization for Standardization, Geneva, Switzerland, 2001.

Michael Jackson. The world and the machine. In *Software Engineering, 1995. ICSE 1995. 17th International Conference on*, pages 283–283. IEEE, 1995.

Michael Jackson. *Problem frames: analysing and structuring software development problems*. Addison-Wesley, 2001.

Ivar Jacobson, Grady Booch, James Rumbaugh, James Rumbaugh, and Grady Booch. *The unified software development process*, volume 1. Addison-Wesley Reading, 1999.

J-P Jacquet and Alain Abran. From software metrics to software measurement methods: A process model. In *Software Engineering Standards Symposium and Forum, 1997. Emerging International Standards. ISESS 97., Third IEEE International*, pages 128–135. IEEE, 1997.

Elmar Juergens, Florian Deissenboeck, Martin Feilkas, Benjamin Hummel, Bernhard Schaetz, Stefan Wagner, Christoph Domann, and Jonathan Streit. Can clone detection support quality assessments of requirements specifications? In *Proceedings of the 32Nd ACM/IEEE International Conference on Software Engineering - Volume 2*, ICSE '10, pages 79–88, New York, NY, USA, 2010. ACM. ISBN 978-1-60558-719-6. doi: 10.1145/1810295.1810308. URL `http://doi.acm.org/10.1145/1810295.1810308`.

Joseph Juran and A Blanton Godfrey. *Quality handbook*. McGraw Hill New York, NY, 5th edition edition, 1999.

H. Kaiya and A. Ohnishi. Quality requirements analysis using requirements frames. In *Quality Software (QSIC), 2011 11th International Conference on*, pages 198–207, July 2011. doi: 10.1109/QSIC.2011.21.

H. Kaiya and M. Saeki. Ontology based requirements analysis: lightweight semantic processing approach. In *Quality Software, 2005. (QSIC 2005). Fifth International Conference on*, pages 223–230, Sept 2005. doi: 10.1109/QSIC.2005.46.

H. Kaiya and M. Saeki. Using domain ontology as domain knowledge for requirements elicitation. In *Requirements Engineering, 14th IEEE International Conference*, pages 189–198, Sept 2006. doi: 10.1109/RE.2006.72.

Haruhiko Kaiya, Hisayuki Horai, and Motoshi Saeki. Agora: Attributed goal-oriented requirements analysis method. In *Requirements Engineering, 2002. Proceedings. IEEE Joint International Conference on*, pages 13–22. IEEE, 2002.

Marcos Kalinowski, Michael Felderer, Tayana Conte, Rodrigo Spinola, Rafael Prikladnicki, Dietmar Winkler, Daniel Méndez Fernández, and Stefan Wagner. Preventing incomplete/hidden requirements: reflections on survey data from austria and brazil. In *International Conference on Software Quality*, pages 63–78. Springer, 2016.

Georg Kalus. *Projektspezifische Anpassung von Vorgehensmodellen*. Dissertation, Technische UniversitÃ€t MÃŒnchen, MÃŒnchen, 2013.

Georg Kalus and Marco Kuhrmann. Criteria for software process tailoring: a systematic review. In *Proceedings of the 2013 International Conference on Software and System Process*, pages 171–180. ACM, 2013.

Mayumi Itakura Kamata and Tetsuo Tamai. How does requirements quality relate to project success or failure? In *Requirements Engineering Conference, 2007. RE'07. 15th IEEE International*, pages 69–78. IEEE, 2007.

Cem Kaner et al. Software engineering metrics: What do they measure and how do we know. In *In 10th International Software Metrics Symposium*. Citeseer, 2004.

R.S. Kenett. Software specifications metrics: a quantitative approach to assess the quality of documents. In *Electrical and Electronics Engineers in Israel, 1996., Nineteenth Convention of*, pages 166–169, Nov 1996. doi: 10.1109/EEIS.1996.566920.

Harksoo Kim, Youngjoong Ko, Sooyong Park, and Jungyun Seo. Informal requirements analysis supporting system for human engineer. In *Systems, Man, and Cybernetics, 1999. IEEE SMC '99 Conference Proceedings. 1999 IEEE International Conference on*, volume 3, pages 1013–1018 vol.3, 1999. doi: 10.1109/ICSMC.1999.823367.

B. Kitchenham, S. L. Pfleeger, and N. Fenton. Towards a framework for software measurement validation. *IEEE Transactions on Software Engineering*, 21(12):929–944, Dec 1995. ISSN 0098-5589. doi: 10.1109/32.489070.

Barbara Kitchenham. Procedures for performing systematic reviews. *Keele, UK, Keele University*, 33(2004):1–26, 2004.

Nadzeya Kiyavitskaya, Nicola Zeni, Luisa Mich, and Daniel M Berry. Requirements for tools for ambiguity identification and measurement in natural language requirements specifications. *Requirements Engineering*, 13(3):207–239, 2008.

Cornelia Knee. Zielorientierte überprüfung von natürlichsprachigen anforderungsdokumenten. Master's thesis, Universität Ulm, April 2006.

Gerald Kotonya and Ian Sommerville. *Requirements Engineering: Processes and Techniques*. Wiley Publishing, 1st edition, 1998. ISBN 0471972088, 9780471972082.

Jennifer Krisch and Frank Houdek. The myth of bad passive voice and weak words an empirical investigation in the automotive industry. In *Requirements Engineering Conference (RE), 2015 IEEE 23rd International*, pages 344–351. IEEE, 2015.

John Krogstie. Integrating the understanding of quality in requirements specification and conceptual modeling. *ACM SIGSOFT Software Engineering Notes*, 23(1):86–91, 1998.

John Krogstie. A semiotic approach to quality in requirements specifications. In Kecheng Liu, RodneyJ. Clarke, Peter Bøgh Andersen, RonaldK. Stamper, and El-Sayed Abou-Zeid, editors, *Organizational Semiotics*, volume 94 of *IFIP The International Federation for Information Processing*, pages 231–249. Springer US, 2002. ISBN 978-1-4757-6111-5. doi: 10.1007/978-0-387-35611-2_14. URL `http://dx.doi.org/10.1007/978-0-387-35611-2_14`.

John Krogstie, Odd Ivar Lindland, and Guttorm Sindre. Defining quality aspects for conceptual models. *ISCO*, 1995:216–231, 1995a.

John Krogstie, Odd Ivar Lindland, and Guttorm Sindre. Towards a deeper understanding of quality in requirements engineering. In *Advanced Information Systems Engineering*, pages 82–95. Springer, 1995b.

Udai Kumar Kudikyala and Rayford B. Vaughn. Software requirement understanding using pathfinder networks: discovering and evaluating mental models. *Journal of Systems and Software*, 74(1):101 – 108, 2005. ISSN 0164-1212. doi: http://dx.doi.org/10.1016/j.jss.2003.09.028. URL `http://www.sciencedirect.com/science/article/pii/S0164121203002954`. Automated Component-Based Software Engineering.

Sari Kujala, Marjo Kauppinen, Laura Lehtola, and Tero Kojo. The role of user involvement in requirements quality and project success. In *Requirements Engineering, 2005. Proceedings. 13th IEEE International Conference on*, pages 75–84. IEEE, 2005.

Janice Langan-Fox, Sharon Code, and Kim Langfield-Smith. Team mental models: Techniques, methods, and analytic approaches. *Human Factors: The Journal of the Human Factors and Ergonomics Society*, 42(2):242–271, 2000.

Timothy C Lethbridge, Janice Singer, and Andrew Forward. How software engineers use documentation: The state of the practice. *Software, IEEE*, 20(6):35–39, 2003.

Li Li, Shuguang He, and E-S Qi. On software requirement metrics based on six-sigma. In *Advanced Management of Information for Globalized Enterprises, 2008. AMIGE 2008. IEEE Symposium on*, pages 1–3. IEEE, 2008.

Beng-Chong Lim and Katherine J Klein. Team mental models and team performance: A field study of the effects of team mental model similarity and accuracy. *Journal of Organizational Behavior*, 27(4):403–418, 2006.

Odd Ivar Lindland, Guttorm Sindre, and Arne Solvberg. Understanding quality in conceptual modeling. *Software, IEEE*, 11(2):42–49, 1994.

Olga Liskin. How artifacts support and impede requirements communication. In *Requirements Engineering: Foundation for Software Quality*, pages 132–147. Springer, 2015.

Klaus Lochmann. Engineering quality requirements using quality models. In *Engineering of Complex Computer Systems (ICECCS), 2010 15th IEEE International Conference on*, pages 245–246. IEEE, 2010.

Ana Belén Barragáns Martínez, José J Pazos Arias, Ana Fernández Vilas, Jorge García Duque, Martín López Nores, Rebeca P Díaz Redondo, and Yolanda Blanco Fernández. Composing requirements specifications from multiple prioritized sources. *Requirements Engineering*, 13(3):187–206, 2008a.

Ana Belén Barragáns Martínez, José J Pazos Arias, Ana Fernández Vilas, Jorge García Duque, Martín López Nores, Rebeca P Díaz Redondo, and Yolanda Blanco Fernández. On the interplay between inconsistency and incompleteness in multi-perspective requirements specifications. *Information and Software Technology*, 50(4):296 – 321, 2008b. ISSN 0950-5849. doi: http://dx.doi.org/10.1016/j.infsof.2007.02.001. URL `http://www.sciencedirect.com/science/article/pii/S0950584907000055`.

J. Matsuoka and Y. Lepage. Ambiguity spotting using wordnet semantic similarity in support to recommended practice for software requirements specifications. In *Natural Language Processing andKnowledge Engineering (NLP-KE), 2011 7th International Conference on*, pages 479–484, Nov 2011. doi: 10.1109/NLPKE.2011.6138247.

Raimundas Matulevicius, Naji Habra, and Flora Kamseu. Validity of the documentation availability model: Experimental definition of quality interpretation. In Barbara Pernici, editor, *Advanced Information Systems Engineering*, volume 6051 of *Lecture Notes in Computer Science*, pages 236–250. Springer Berlin Heidelberg, 2010. ISBN 978-3-642-13093-9. doi: 10.1007/978-3-642-13094-6_20. URL `http://dx.doi.org/10.1007/978-3-642-13094-6_20`.

Thomas J McCabe. A complexity measure. *IEEE Transactions on software Engineering*, (4):308–320, 1976.

D. Méndez Fernández and B. Penzenstadler. Artefact-based requirements engineering: The amdire approach. *Requirements Engineering (To appear)*, 2014.

Daniel Méndez Fernández and Stefan Wagner. Naming the pain in requirments engineering: A design for a global family of surveys and first results from germany. *IST*, 2014.

Daniel Méndez Fernández, Jakob Mund, Henning Femmer, and Antonio Vetrò. In quest for requirements engineering oracles: Dependent variables and measurements for (good) re. In *Proceedings of the 18th International Conference on Evaluation and Assessment in Software Engineering*, EASE '14, pages 3:1–3:10, New York, NY, USA, 2014. ACM. ISBN 978-1-4503-2476-2. doi: 10.1145/2601248.2601258. URL `http://doi.acm.org/10.1145/2601248.2601258`.

Daniel Méndez Fernández, Stefan Wagner, Marcos Kalinowski, Andre Schekelmann, Ahmet Tuzcu, Tayana Conte, Rodrigo Spinola, and Rafael Prikladnicki. Naming the

pain in requirements engineering: Comparing practices in brazil and germany. *IEEE Software*, 32(5), 2015.

Andrew Meneely, Ben Smith, and Laurie Williams. Validating software metrics: A spectrum of philosophies. *ACM Transactions on Software Engineering and Methodology (TOSEM)*, 21(4):24, 2012.

Bertrand Meyer. On formalism in specifications. *IEEE software*, 2(1):6, 1985.

Martin Monperrus, Benoit Baudry, Joël Champeau, Brigitte Hoeltzener, and Jean-Marc Jézéquel. Automated measurement of models of requirements. *Software Quality Journal*, 21(1):3–22, 2013.

Kedian Mu, Zhi Jin, Ruqian Lu, and Weiru Liu. Measuring inconsistency in requirements specifications. In Lluís Godo, editor, *Symbolic and Quantitative Approaches to Reasoning with Uncertainty*, volume 3571 of *Lecture Notes in Computer Science*, pages 440–451. Springer Berlin Heidelberg, 2005. ISBN 978-3-540-27326-4. doi: 10.1007/11518655_38. URL `http://dx.doi.org/10.1007/11518655_38`.

Jakob Mund, Daniel Mendez Fernandez, Henning Femmer, and Jonas Eckhardt. Does quality of requirements specifications matter? combined results of two empirical studies. In *2015 ACM/IEEE International Symposium on Empirical Software Engineering and Measurement (ESEM)*, pages 1–10. IEEE, 2015.

NASA. Systems engineering handbook, revision 1 (nasa/sp-2007-6105). Technical Report NASA/SP-2007-6105, National Aeronautics and Space Administration (NASA), Washington, DC, USA, 2007.

Frank Niessink and Hans Van Vliet. Measurements should generate value, rather than data [software metrics]. In *Software Metrics Symposium, 1999. Proceedings. Sixth International*, pages 31–38. IEEE, 1999.

Frank Niessink and Hans Van Vliet. Measurement program success factors revisited. *Information and Software Technology*, 43(10):617–628, 2001.

Donald A Norman. Some observations on mental models. *Mental models*, 7(112):7–14, 1983.

Bashar Nuseibeh and Steve Easterbrook. Requirements engineering: a roadmap. In *Proceedings of the Conference on the Future of Software Engineering*, pages 35–46. ACM, 2000.

John S Oakland. *Statistical process control*. Routledge, 2007.

J Natt och Dag, Björn Regnell, Pär Carlshamre, Michael Andersson, and Joachim Karlsson. Evaluating automated support for requirements similarity analysis in market-driven development. In *Proc. 7th Int. Workshop on Requirements Engineering: Foundation for Software Quality ({REFSQ'01)}*. Citeseer, 2001a. URL `http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.24.9730&rep=rep1&type=pdf`.

J Natt och Dag, Björn Regnell, Pär Carlshamre, Michael Andersson, and Joachim Karlsson. Evaluating automated support for requirements similarity analysis in market-driven development. In *Proc. 7th Int. Workshop on Requirements Engineering: Foundation for Software Quality (REFSQ'01)*. Citeseer, 2001b.

Daniel Ott. Defects in natural language requirement specifications at mercedes-benz: An investigation using a combination of legacy data and expert opinion. In *Requirements Engineering Conference (RE), 2012 20th IEEE International*, pages 291–296. IEEE, 2012.

Sven Overhage, DominikQ. Birkmeier, and Sebastian Schlauderer. Quality marks, metrics, and measurement procedures for business process models. *Business & Information Systems Engineering*, 4(5):229–246, 2012. doi: 10.1007/s12599-012-0230-8. URL `http://dx.doi.org/10.1007/s12599-012-0230-8`.

S Park, H Kim, Y Ko, and J Seo. Implementation of an efficient requirements-analysis supporting system using similarity measure techniques. *Information and Software Technology*, 42(6):429 – 438, 2000. ISSN 0950-5849. doi: http://dx.doi.org/10.1016/S0950-5849(99)00102-0. URL `http://www.sciencedirect.com/science/article/pii/S0950584999001020`.

Shari Lawrence Pfleeger. Software metrics: progress after 25 years? *IEEE Software*, 25 (6):32, 2008.

David R Pitts. Metrics: problem solved. *Crosstalk: The Journal of Defense Software Engineering*, 1997.

Klaus Pohl. The three dimensions of requirements engineering: a framework and its applications. *Information systems*, 19(3):243–258, 1994.

Klaus Pohl. *Requirements Engineering: Fundamentals, Principles, and Techniques*. Springer Publishing Company, Incorporated, 1st edition, 2010. ISBN 3642125778, 9783642125775.

Klaus Pohl, Harald Hönninger, Reinhold Achatz, and Manfred Broy. *Model-Based Engineering of Embedded Systems: The SPES 2020 Methodology*. Springer Science & Business Media, 2012.

Alejandro Rago, Claudia Marcos, and J.Andres Diaz-Pace. Identifying duplicate functionality in textual use cases by aligning semantic actions. *Software & Systems Modeling*, pages 1–25, 2014. ISSN 1619-1366. doi: 10.1007/s10270-014-0431-3. URL `http://dx.doi.org/10.1007/s10270-014-0431-3`.

DavidC. Rine and Anabel Fraga. Chunking complexity measurement for requirements quality knowledge representation. In Ana Fred, Jan L.G. Dietz, Kecheng Liu, and Joaquim Filipe, editors, *Knowledge Discovery, Knowledge Engineering and*

Knowledge Management, volume 454 of *Communications in Computer and Information Science*, pages 245–259. Springer Berlin Heidelberg, 2015. ISBN 978-3-662-46548-6. doi: 10.1007/978-3-662-46549-3_16. URL `http://dx.doi.org/10.1007/978-3-662-46549-3_16`.

Suzanne Robertson and James Robertson. *Mastering the Requirements Process (2Nd Edition)*. Addison-Wesley Professional, 2006. ISBN 0321419499.

James Roche. Adopting devops practices in quality assurance. *Communications of the ACM*, 56(11):38–43, 2013.

William B Rouse and Nancy M Morris. On looking into the black box: Prospects and limits in the search for mental models. *Psychological bulletin*, 100(3):349, 1986.

Chris Rupp and SOPHISTen. *Requirements engineering und management*. Carl Hanser Verlag, 2007.

Ken Schwaber and Mike Beedle. *Agile Software Development with Scrum*. Prentice Hall PTR, Upper Saddle River, NJ, USA, 1st edition, 2001. ISBN 0130676349.

Claude E. Shannon. A Mathematical Theory of Communication. *The Bell System Technical Journal*, 27(3):379–423, 1948.

Anselm Strauss and Juliet Corbin. *Basics of qualitative research: Techniques and procedures for developing grounded theory* . Sage Publications, Inc, 1998.

Patrick Suppes and Joseph L Zinnes. *Basic measurement theory*. Technical Report No. 45, Stanford University, 1962.

CMMI Product Team. Cmmi for development, version 1.3. Technical Report CMU/SEI-2010-TR-033, Software Engineering Institute, Carnegie Mellon University, Pittsburgh, PA, 2010. URL `http://resources.sei.cmu.edu/library/asset-view.cfm?AssetID=9661`.

The Standish Group. The chaos report, 1995. URL `www.standishgroup.com`.

Axel Van Lamsweerde. Goal-oriented requirements engineering: A guided tour. In *Requirements Engineering, 2001. Proceedings. Fifth IEEE International Symposium on*, pages 249–262. IEEE, 2001.

June Verner, Karl Cox, Steven Bleistein, and Narciso Cerpa. Requirements engineering and software project success: an industrial survey in australia and the us. *Australasian Journal of information systems*, 13(1), 2005.

Stefan Wagner, Florian Deissenboeck, and Sebastian Winter. Managing quality requirements using activity-based quality models. In *Proceedings of the 6th international workshop on Software quality*, pages 29–34. ACM, 2008.

Stefan Wagner, Klaus Lochmann, Lars Heinemann, Michael Kläs, Adam Trendowicz, Reinhold Plösch, Andreas Seidl, Andreas Goeb, and Jonathan Streit. The quamoco product quality modelling and assessment approach. In *Proceedings of the 2012 International Conference on Software Engineering*, pages 1133–1142. IEEE Press, 2012.

Sheila Simsarian Webber, Gilad Chen, Stephanie C Payne, Sean M Marsh, and Stephen J Zaccaro. Enhancing team mental model measurement with performance appraisal practices. *Organizational Research Methods*, 3(4):307–322, 2000.

M. Weidlich, J. Mendling, and M. Weske. Efficient consistency measurement based on behavioral profiles of process models. *Software Engineering, IEEE Transactions on*, 37(3):410–429, May 2011. ISSN 0098-5589. doi: 10.1109/TSE.2010.96.

Dave West, Tom Grant, M Gerush, and D Dsilva. Agile development: Mainstream adoption has changed agility. *Forrester Research*, 2(1):41, 2010.

Karl Wiegers. Writing quality requirements. *Software Development*, 7(5):44–48, 1999.

R. J. Wieringa. *Requirements Engineering: Frameworks for Understanding*. John Wiley & Sons, Inc., New York, NY, USA, 1996. ISBN 0-471-95884-0.

Roel Wieringa, Neil Maiden, Nancy Mead, and Colette Rolland. Requirements engineering paper classification and evaluation criteria: a proposal and a discussion. *Requirements Engineering*, 11(1):102–107, 2006.

William M Wilson, Linda H Rosenberg, and Lawrence E Hyatt. Automated analysis of requirement specifications. In *Proceedings of the 19th international conference on Software engineering*, pages 161–171. ACM, 1997.

Claes Wohlin, Per Runeson, Martin Höst, Magnus C Ohlsson, Björn Regnell, and Anders Wesslén. *Experimentation in software engineering*. Springer, 2012.

Hui Yang, Anne de Roeck, Vincenzo Gervasi, Alistair Willis, and Bashar Nuseibeh. Analysing anaphoric ambiguity in natural language requirements. *Requirements Engineering*, 16(3):163–189, 2011. ISSN 0947-3602. doi: 10.1007/s00766-011-0119-y. URL `http://dx.doi.org/10.1007/s00766-011-0119-y`.

A Appendix

Glossary

In this section, we provide brief explanations for fundamental terms used in this thesis in alphabetical order (terms contained in this glossary are designated by → *Term*).

Artifact A self-contained work result which constitutes a physical representation, syntax, and semantics.

Example: A system vision is an artifact which describes the purpose of the system (semantics) by means of natural-language (syntax) and is stored as a file in a certain format (physical representation), e.g., portable document format (PDF).

Audience Social and technical actors who require knowledge of the systems' → *requirements* in order to carry out downstream development activities. Exemplary roles are testers, developers, architects, user interface designers or spell checkers.

Note: We refrained from the term *stakeholder* to avoid ambiguity since we need to differentiate between the stakeholders of the system (e.g., customers, users) and the stakeholders of requirements (e.g., testers, developers), i.e., the audience.

Context of use Characteristics of the environment which affect development activities that depend on the → *requirements specification* as an input.

Note: Tbl. 4.2 (p. 68) structures and briefly describes selected characteristics which are supposed to influence the effect certain inherent properties have on the quality-in-use of downstream development activities.

Downstream Development The part of the development process which directly or indirectly (transitively) depends on the requirements.

Intrinsic Quality A → *quality attribute* which is inherent to the entity under consideration. Specifically for → *requirements specifications*, it refers to essential properties of the → *artifact* itself concerning its physical representation (e.g., the document's size), syntax (e.g., the structure of the table of contents), or semantics (e.g., whether the system's reaction is described for any possible stimuli/input).

Note: The quality definition model provides an extensional definition in Sec. 4.1 (pp. 64).

Note: 'Inherent' as opposed to 'Assigned'

Measure A quantitative scale and an associated → *measurement procedure*.

Note: This meaning refers to the most basic understanding of what constitutes a measure. In Sec. 5.1 (pp. 102), we provide a more comprehensive model of the concepts attributed to a measure.

Mental model The organized understanding and mental representation of knowledge about key elements of the system under consideration [Lim and Klein, 2006].

Note: Mental models in the context of requirements engineering are described in further detail in Sec. 2.3.1 (pp. 26).

Meta-Model A → *model* of a → *model*, i.e., an abstraction of models designed to answer specific questions about those models.

Metric synonym for → *measure*.

Measurement Procedure The process of assigning quantitative values to empirical observations regarding a certain attribute of a particular entity, and according to well-defined rules [Fenton and Pfleeger, 1998].

Measurement Value The quantitative value on a measurement scale obtained as a result of applying a → *measurement procedure*.

Model An abstraction of an object designed to answer specific questions about the object.

Quality refers to:

(1) Characteristics of an entity that bear on its ability to satisfy stated or implied needs. [ISO/IEC 9126-1, 2001],[ASQ]

(2) Degree to which an entity possesses the characteristics as in (1).

Note: Quality in the meaning of (2) can therefore be understood as the degree an entity satisfies stated or implied needs.

Quality-in-Use The effects the requirements specification has on the performance of development activities when used in terms of efficiency, effectiveness, satisfaction, and context flexibility (see Tab. 4.1 (p. 66) for a description).

The degree to which a product (requirements specification) used by specific users (audience) meets their needs to achieve specific goals (conduct the downstream development activity) with effectiveness, efficiency, safety and satisfaction in specific contexts of use [ISO/IEC 25010:2010, 2010].

The extent to which a requirements specification used by specific members of the audience informs them in order to conduct their activities with effectiveness, efficiency, satisfaction and safety.

Quality Assessment refers to:

(1) A justified estimation of the entity under assessment (in this thesis, usually the → *requirements specification*) which reflects the extent to which it is of a certain → *quality*, i.e., possesses certain quality characteristics.

(2) The process of obtaining a rating as in (1).

Quality Attribute Characteristic considered a → *quality* of the entity under consideration.

Quality Control The process that consists of analyzing the actual quality of a product (in this thesis, usually the → *requirements specification*), comparing it to quality requirements, and taking necessary actions to correct the difference.

Quality Criterion synonym for → *quality attribute*.

Quality Factor synonym for → *quality attribute*.

Quality Measure A → *measure* measuring a → *quality attribute*.

Reference Model A → *model* blueprint that defines concepts and relations to be used as orientation for a particular application domain

Requirements A condition or capability (i) needed by a user to solve a problem or achieve an objective or (ii) that must be met or possessed by a system or system component to satisfy a contract, standard, specification, or other formally imposed documents. [IEEE Standard 610.12, 1990]

Requirements Documentation synonym for → *requirements specification*.

Requirements Engineer A participant of a project responsible for eliciting and specifying the system requirements [Kotonya and Sommerville, 1998].

Requirements Engineering The iterative process of discovering, negotiating and managing the purpose of a software-intensive system is intended for, by identifying

→ *stakeholders*, and their needs, and documenting these in a form that is amenable to analysis, communication, and subsequent implementation.

Note: Based on Nuseibeh and Easterbrook [2000] but modified to emphasize its iterative nature and to include also requirements management activities.

Requirements Quality The characteristics of → *requirements* that bear on its ability to inform the → *audience* in order to build the optimal system with least effort.

(Software) Requirements Specification A collection of documents representing the → *requirements*. [IEEE Standard 610.12, 1990]

Note: Refer to Méndez Fernández and Penzenstadler [2014], Geisberger [2005], or IEEE Standard 830 [1998] for contents of the requirements specification.

Requirements Specification Quality The characteristics of → *requirements specifications* that bear on its ability to inform the → *audience* in order to build the desired system with least effort.

Note: A desired system is a system which meets the stakeholders' stated and implied needs (see Sec. 2.1, pp. 18).

Software-Intensive System A software-intensive system is any system where software contributes essential influences to the design, construction, deployment, and evolution of the system as a whole. IEEE Standard 1471 [2000]

Stakeholder A stakeholder is a person or organization who influences a system's requirements or who is impacted by that system. Glinz and Wieringa [2007]

Individual or organization having a right, share, claim or interest in a system or its possession of characteristics that meet their needs and expectations [ISO/IEC 25010:2010, 2010].

Note: This definition refers to a stakeholder of the system under consideration. The stakeholders of the requirements specification are referred to as → *audience* to avoid misunderstandings.

Tailoring The act of adjusting the definition and/or particularizing the terms of a general description to derive a description applicable to an alternate (less general) environment [Ginsberg and Quinn, 1995].

B Appendix

Catalogue of Quality Measures

In this appendix, we present the measures identified by the systematic literature review which are part of the assessment model described in Chap. 6. Here, each measure is presented according to the data collection elements described in Tbl. 6.2 (p. 129), which are a simplified version of the meta-model presented in Chap. 5.1.

B.1. (Weighted) Syntactic Sentence Ambiguity [Kiyavitskaya et al., 2008]

Assessed (Measured) Attribute	Unambiguous (Syntactical ambiguity of Individual Sentences: Extent to which a sentence can have multiple syntax trees without considering context.)
Data Collection	Authors presume the usage of a COTS NLP tool called LOLITA. The measure is applied on individual natural language sentences. For each sentence S, the number of possible syntactic parse trees based on any (ambiguous) interpretation of their lexicals (words) is calculated (denoted delta of S) by LOLITA. This number of possible parse trees is weighted by an addition information obtained by LOLITA called the "penalty" of a syntax tree t for a sentence S. This penalty is described as "LOLITA's statement of how much effort it spent building t [and] an attempt to model the likelihood for t to be be the parse tree intended by the person who said or wrote S". The penalty values are informally defined as "major structural problems" (value: 4), "major feature clash such as [...] real dative or infinitive use of inappropriate verbs" (3), "minor feature clash" (2), "at most some less common but nevertheless correct constructs" (1), or "mo problems whatsoever" (0). The paper includes example sentences for the penalty levels.
Scale	Natural number (Ratio scale, The scale denotes the extent to which a sentence syntactic structure is ambiguous, i.e., allows multiple interpretations with respect to the grammar, taking into account the presence and severity of syntactical violations as an amplifier.)
Interpretation	The authors provide reference thresholds against which the measurement value should be compared.
Reference Value	According to "[the authors'] experience, it seems right to classify a value of less than or equal to 5 as signifying little or no ambiguity, a value of greater than or equal to 20 as signifying highly ambiguous, and a value of greater than 5 but less than 20 as signifying somewhat ambiguous."
Recommendations for Action	Any sentence interpreted "highly ambiguous" must and every sentence of critical requirements interpreted "somewhat ambiguous" should be checked by a reviewer.
Threats to Validity	• Writing style and languages might require an adjustment of the measurement procedure (limits) and/or the prescriptive reference values

B.2. Actor Presence [Etien and Rolland, 2005]

Assessed (Measured) Attribute	Feature Complete (Business Actor Coverage: The extent to which all relevant business actors are specified in the system's description of the SRS)
Data Collection	The authors assume the presence of a description/model of the business domain and the systems implementation according to specific meta-models (ontologies) as well as a mapping between elements of those models in terms of mapping between elements (equivalent concepts) and a representation mapping (a system element may represent certain aspects of a business element). For this particular metric, the business actors (actors which are involved in the business process supported by the system under consideration), both all and only those for which a system class (i.e., an object in the data model) exists, are counted. Depending on the actual tool, this process can be automated.
Scale	Rational number (Ratio scale, The scale denotes the extent to which business actors are also represented in the system to be build. A value of 0 and 1 denotes all respectively no actors are also represented in the system.)
Interpretation	The authors do no provide an explicit interpretation procedure. In the paper, a single value is computed, however, no preference values/thresholds are specified.
Recommendations for Action	"In order to increase [the metric's] value, it is possible (1) to remove the business actor who is not present in the system if the check demonstrates that he does [not] play a significant role in the besuiness process or, (2) to introduce the actor in the system".

B.3. Agent-Goal Ratio [Espada et al., 2011, 2013]

Assessed (Measured) Attribute	Information Complete (Agent Specification Completeness: Extent to which agents have been completely specified for all leaf goals)
Data Collection	Authors assume that goals are specified according to the KAOS methodology. During data collection, the number of goals which have (at least) one actor specified as well as all goals are counted, and the ratio is computed according to the pseudo formula. For [espada2011measuring] this task is done manually, while in [espada2013framework], a prototypical implementation (modularKAOS) was used.
Scale	Rational number (Ratio scale, The extent to which the "assignment of all goal responsibilities to agents [is completed]", on a scale of [0;1].)
Interpretation	The authors do not specify an absolute interpretation procedure; Instead, the progress is compared. A value of 1 is desirable, any value below 1 is against the KAOS modeling guidelines and should be regarded as incomplete.
Recommendations for Action	During data collection, the goals without actors are identified, and should be reported to be completed (with help of the stakeholders if necessary).

B.4. Ambiguous Reviewer Interpretations (ABR) [Davis et al., 1993a]

Assessed (Measured) Attribute	Unambiguous
Data Collection	Measures all specified requirements in a SRS via review; That is, each requirement is interpreted by a reviewer, and the percentage of requirements that have been interpreted in a unique manner by all (at least two) reviewers are counted and divided by the total number of requirements. Hence, an quotient $Q1 = n_{ui}/n_r$ is obtained, in which n_ui is the number of requirements for which all reviewers presented identical requirements, and n_r is the number of requirements present in a SRS.
Scale	Rational number (Ratio scale, Percentage of requirements which are ambiguous compared to all requirements. Therefore, granularity depends on the number of requirements present in a SRS, and is limited to the "requirements as a unit" level.)
Interpretation	The metric is interpreted with regards to the quality attribute "ambiguity" exclusively. A ratio close to/of 0 denotes that every single requirement is ambiguous, and of 1 means the SRS is free of ambiguity. Since we are on a ratio scale, the intervals in between are meaningful. Hence, we can consolidate distances in ambiguity for interpretation
Recommendations for Action	As a byproduct of comparing the reviewers' interpretations, differences are collected. To resolve ambiguity, those differences should be addressed in the SRS
Threats to Validity	• The comparison of the individual interpretations provided by the reviewers must be powerful enough to capture even subtle ambiguities. • Reviewers selected do not adequately represent the consignee of the SRS. Furthermore, a prescriptive reference value other than 1 is hard to find and justify since no empirical evidence exists

B.5. Ambiguous Sentence Parts (ASP) [Kenett, 1996]

Assessed (Measured) Attribute	Unambiguous
Data Collection	For a NL-SRS, each sentence is (presumably manually, but not specified in the publication) decomposed into attributes, e.g., the initiator of an action, the action, the object, etc. (9 types of attributes are given). Furthermore, each sentence is (again, presumably manually) reviewed to determine whether the attribute is ambiguous or not.
Scale	Rational number (Ratio scale, Ratio between the cardinality of two sets: the count of all ambiguous attributes, and the count of all attributes of the SRS)
Interpretation	Values between 0 and 1 are obtainable in theory, with 0 being the interpreted as the best result (no ambiguous attributes), and 0.5 meaning: for every attribute specified, there is also one ambiguous attribute, and 1 meaning that every attribute is ambiguous. The authors argue that practical experience yielded thresholds, but no not disclose those in the publication.
Recommendations for Action	Ambiguous attributes are identified, and therefore, can be targeted by corrective action.
Threats to Validity	• Ambiguity not only depends on the syntax and semantics of the SRS, but also depends on the reader of the SRS. Therefore, the reviewer might consider an attribute as ambiguous while it has a unique meaning to the receipee of the SRS, and vice versa. • Rating an attribute as ambiguous is also influenced by the domain knowledge of the reader. Therefore, different project contexts might require different interpretation values

B.6. Annotation Coverage (AC): Importance, Stability, Version [Davis et al., 1993a]

Assessed (Measured) Attribute	Meta-data Complete (Annotation Completeness regarding Importance, Stability and Version: The extent of annotation with three specific aspects)
Data Collection	For each requirement, decide if (i) it is annotated by relative importance, (ii) by relative stability and (iii) by version, or not. Therefore, three different flags must be added to each requirement, which is syntactically inspected by the metric.
Scale	Rational number (Ratio scale, Percentage of annotation completeness regarding three (equivalently important) dimensions: importance, stability and version.)
Interpretation	The values are interpreted in a range from 0 (no requirement is annotated) to 1 (every requirement is annotated with regards to every aspect (importance, stability, version))
Recommendations for Action	For any requirement not associated with a (relative) importance, stability or version identified should be completed.
Threats to Validity	• A reference value other than 1 is not provided and might be hard to justify. Furthermore, the flags must be present for every requirement, therefore, in case of changes, the SRS must be maintained.

B.7. Average Customer Goal Satisfaction (CUP) [Kaiya et al., 2002]

Assessed (Measured) Attribute	Semantically Correct
Data Collection	The authors assume the presence of an attributed goal-graph according to their specific meta-model, with root elements (no predecessors) representing stakeholder demands and leaf elements representing finalized goals, i.e., refined requirements requiring certain functionality or characteristics of the (software) system. For any requirement (=finalized goal), a customer preference value must be associated which describes the extend the customer likes resp. dislikes that the system includes the requirement, on a scale from -10 (dislikes) to +10 (likes). The metric can therefore be automatically computed by aggregating the annotated values of the goal model.
Scale	Rational number (Ratio scale, The scale is a rational number between -10 and +10 which expresses the average customer satisfaction of the requirements derived in the goal model. Values of +10 and -10 denote maximal approval resp. rejection of the obtained requirements from the customer's perspective.)
Interpretation	The authors do not specify an interpretation procedure. Since a single value is obtained, prescriptive thresholds might work. However, because of the metric's complexity and the inherent trade-off associated with goals in practice (see also threats to external validity), the applicability of prescriptive reference values should be investigated empirically.
Threats to Validity	• Goal satisfaction is at least project-specific, i.e., the degree of achievable satisfaction depends for every single project (and can even change during projects in practice). Therefore, external validity seems hardly achievable. Therefore, we could imagine the interpretation procedure to include a derivation of project-specific reference values.

B.8. Average Meanings per Cluster of Technical Terms [Matsuoka and Lepage, 2011]

Assessed (Measured) Attribute	Unambiguous (Lexical ambiguity of words within sentences: Extent to which a sentence's technical terms have multiple interpretations without considering context.)
Data Collection	In a first step, "those words that are relevant for the requirements at hand, i.e., technical terms" have to be identified. Therefore, for a set of candidate terms (not specified in the paper, we assume all terms except stop words) a c-value measure (higher number= term is composed of more words and occurs frequently) is computed, and the terms with the highest value are extracted. For any such term, all N meanings are retrieved using the WordNet semantic database. In case N>1 (multiple meanings), the words similarity to other words in the sentence is investigated (using a similarity measure based on WordNets hierarchical structure) and each occurrence of the word is classified in a cluster according to the most significant meaning. In case the similarity measure is equal for more than one occurrence of the term, the occurrences are classified in a cluster associated with multiple meanings. Afterwards, Inter-sentence similarity is used to move sentences/occurrences of multiple-meaning clusters into cluster with only one meaning, according to a sentence similarity measure with a prescriptive threshold. Finally, the number of clusters are counted and the number of meanings of all those clusters (>= 1 per cluster) are added together to obtain its ratio.
Scale	Rational number (Absolute scale, The extent to which technical terms are used ambiguously in the SRS, on a [1.0,INF+]. A value of 1 denotes that all occurrences of technical terms can be interpreted in only one (but not necessarily the same) way, and higher values in general denote the existence of more interpretations.)
Interpretation	No interpretations procedure is explicitly stated; In the paper, authors distinguish a value of 1 to not cause any action, while any value >1 is ambiguous and should be inspected.
Reference Value	1.0
Recommendations for Action	Candidate terms are identified; Because of the limitations to the most important technical terms only, this approach seems to be applicable in practical situations
Threats to Validity	• The WordNet database, as an socio-technical system, might fail or provide wrong results; Furthermore, typos might flaw results such that look-ups do not work. • Due to the complexity of language, the WordNet database is incomplete, and hence some additional interpretations might be missing. Furthermore, the hierarchical structuring of words as the only measure of similarity reduces relationships between objects to only one aspect, which might lead to incorrectly computed word similarities. • A source for disambiguation not considered in the paper is that of the background/domain knowledge of the person, which might allow to rule out certain ambiguities. Furthermore, additional interpretation assistance might be provided in terms of process or artifact knowledge, e.g., the use of templates and the interpretation of content because of certain artifact types.

B.9. Average Minimum Goal Satisfacton (SAT) [Kaiya et al., 2002]

Assessed (Measured) Attribute	Semantically Correct
Data Collection	The authors assume the presence of an attributed goal-graph according to their specific meta-model, with root elements (no predecessors) representing stakeholder demands and leaf elements representing finalized goals, i.e., refined requirements requiring certain functionality or characteristics of the (software) system. For any edge, an assigned contribution value denotes the degree to which a goal contributes to the achievement of the parent goal, expressed as an integer from -10 (maximal negative) to +10 (maximal positive contribution). Therefore, model conforming to this meta-model can be analyzed automatically, according to the formula, in which contribution values on path from finalized goals (leaf nodes) to initial goals (root nodes) are aggregated using min and average (denoted as mu) operations.
Scale	Rational number (Ratio scale, The scale is a rational number between -1 and +1, which expresses the degree the stakeholder needs are satisfied by the finalized goals (requirements). A value of +1 and -1 denote maximal satisfaction and dissatisfaction of the stakeholders, respectively.)
Interpretation	The authors do not specify an interpretation procedure. Since a single value is obtained, prescriptive thresholds might work. However, because of the metric's complexity and the inherent trade-off associated with goals in practice (see also threats to external validity), the applicability of prescriptive reference values should be investigated empirically.
Threats to Validity	• Goal satisfaction is at least project-specific, i.e., the degree of achievable satisfaction depends for every single project (and can even change during projects in practice). Therefore, external validity seems hardly achievable. Therefore, we could imagine the interpretation procedure to include a derivation of project-specific reference values.

B.10. Backward Concept Completeness [Ferrari et al., 2014]

Assessed (Measured) Attribute	Feature Complete (Concept Completeness regarding Input Documentation: The extent to which the SRS mentions all important (domain-specific and frequently occurring) concepts contained in the input documents)
Data Collection	The author assume that functional requirements are specified in natural language (NL) and a set of NL input documents such as "international standards", "preliminary specifications, transcripts of meetings with customer, etc". The first step is to identify a ranked list of terms. Therefore, part-of-speech (POS) tagging extracts all words and groups of words (multi-words) which follow certain POS patterns (e.g., "<adjective, noun, noun>"). The terms obtained this way are ranked according to a "termhood" metric called C-NC, which essentially orders the words/multi-words according to its "conceptual independence" by measuring how frequently it co-occurs with the same words in comparison to different words. In the next step, the terms' domain specificity, i.e., to what extent the term is specific to the domain in contrast to domain-independent terms, is measured. Therefore, a contrastive analysis is conducted, which essentially compares the terms occurrence in the input documents to a domain-generic corpus, for which the authors used a database of articles published in the Wall Street Journal (more detail in the paper). Finally, the SRS is analyzed if the obtained terms above a given threshold (0.99 was proposed by the authors), i.e., the most domain-specific, conceptually independent terms most frequently occurring in the input documents, are also present in the SRS, and the ratio is computed. All steps have been automatized in a prototypical tool implementation.
Scale	Rational number (Ratio scale, Extent to which the SRS includes all domain-specific terms which are central (conceptually independent and frequently occurring) to a set of input (reference) documents.)
Interpretation	The authors use the measurement as a constructive means during requirements specification, but do not specify an interpretation procedure. Implicitly, the metric is used to compare it against different versions of the same document to assess the improvement regarding completeness.
Recommendations for Action	Domain-specific terms not included in the SRS are identified are reported to the quality manager. The recommended action is to investigate, potentially including the stakeholders, whether or not a requirement is indeed missing or the concept is already described elsewhere or not relevant for the SRS (anymore).
Threats to Validity	• Missing input documents, lexically inconsistent terms between various input document and/or the SRS • Using the metric to assess semantic completeness of the SRS with respect to the actual stakeholder needs, completeness of the input documents is assumed. This is a very volatile assumption in general since preliminary input documents created during elicitation can be highly incomplete, due to high uncertainty early on or only partial documentation of results. Furthermore, correctness of the input documents also impacts the metrics completeness interpretation: If input documents contain requirements later identified as incorrect, and hence, the respective terms are not included in the SRS any longer, the measurement might suggest a lower semantic completeness of the SRS. • Contexts with volatile terminology might result in lower completeness values (see internal validity). Furthermore, the project's organization regarding the extent and quality of documenting results impacts the metrics interpretation reliability.

B.11. Backward Interaction Completeness [Ferrari et al., 2014]

Assessed (Measured) Attribute	Feature Complete (Concept Interaction Completeness regarding Input Documentation (Backward completeness): The extent to which the SRS mentions all important (frequent and exclusive) concept interactions of domain-specific terms contained in the input documents)
Data Collection	The author assume that functional requirements are specified in natural language (NL) and a set of NL input documents such as "international standards", "preliminary specifications, transcripts of meetings with customer, etc". The metric is based on Backward Concept Completeness by relying on the same identification of central, domain-specific terms from the input documents (denoted InputTerms). Then, the metric identifies candidate relatations between terms in the input documents by extracting any terms occurring in the previous, same or following sentence for every term in InputTerms. Afterwards, for each of those candidate relations, the log-likelihood metric for binomial distributions (described in a publication referred to in the paper) is computed. Essentially, a higher value is assigned to a candidate pair if there is a sort of exclusive and frequent relation about those terms, i.e., if the pair appear frequently together but it does not often occur with other terms. In case the likelihood-value is above a threshold (authors recommend a threshold of 10.83 "to select only relevant relations"), the pair is included in the set InputRels. Finally, the number of relations between terms is counted and compared to the number of those term co-occurrences in the SRS.
Scale	Rational number (Ratio scale, Extent to which domain-specific, central terms of input documents also co-occur in the SRS, given that a co-occurrence between those terms is also present, frequent and exclusive in the input documents.)
Interpretation	The authors use the measurement as a constructive means during requirements specification, but do not specify an interpretation procedure. Implicitly, the metric is used to compare it against different versions of the same document to assess the improvement regarding completeness.
Recommendations for Action	Relations of domain-specific terms which are not reflected in the SRS are identified are reported to the quality manager. The recommended action is to investigate, potentially including the stakeholders, whether or not a requirement is indeed missing or the concept interaction is already described elsewhere or not relevant for the SRS (anymore).
Threats to Validity	• Missing input documents, lexically inconsistent terms between various input document and/or the SRS • Using the metric to assess semantic completeness of the SRS with respect to the actual stakeholder needs, completeness of the input documents is assumed. This is a very volatile assumption in general since preliminary input documents created during elicitation can be highly incomplete, due to high uncertainty early on or only partial documentation of results. Furthermore, correctness of the input documents also impacts the metrics completeness interpretation: If input documents contain requirements later identified as incorrect, and hence, the respective terms are not included in the SRS any longer, the measurement might suggest a lower semantic completeness of the SRS. • Contexts with volatile terminology might result in lower completeness values (see internal validity). Furthermore, the project's organization regarding the extent and quality of documenting results impacts the metrics interpretation reliability.

B.12. Behavioral Profile Consistency [Weidlich et al., 2011]

Assessed (Measured) Attribute	Semantically Consistent (Behavior Profile Consistency: Extent to which a process specification is free of contradictions regarding the actions for which the specification is fragmented accross multiple diagrams)
Data Collection	
Scale	Rational number (Ratio scale, The extent to which the transitions shared between two processes are arranged equally, in the sense that the relations between individual transitions do not contradict, on the scale [0;1]. A value of 0 denotes complete inconsistency, while a scale of 1 denotes perfect (behavioral profile) consistency. "A degree of 0.9, in turn, indicates that the constraints on the order of potential activity occurrences are equal solely for 90% of the relations between aligned activities".)
Interpretation	Authors propose to us "consistency thresholds", however do not provide concrete numbers since "[they] assume these thresholds to be highly dependent on a specific project setting".
Threats to Validity	• Depending on the way the correspondence mapping between the two processes is achieved; For instance, if it is based on consistent naming, typos or lexical inconsistencies can flaw the measurement. Furthermore, if not tool-based, human mistakes in computing the consistent transition pairs might arise due to the task's complexity, both regarding size and demanded formal rigor. • Authors emphasize the fact that this "does not imply that both models are (projected) trace equivalent".

B.13. BPM Elements Completeness Ratio [Overhage et al., 2012]

Assessed (Measured) Attribute	Semantically Complete
Data Collection	Authors do not explicitly specify a measurement process; Therefore, we will assume a manual measurement process by means of quality managers (e.g., requirements engineers) on the business process description (both, using natural and graphical languages) as suggested in the paper's evaluation. Therefore, the real world excerpt is analyzed (unspecified how) for completeness, i.e., whether or not all relevant aspects are captured in the model according to the quality manager. Finally, the ratio of missing real world objects to all elements in the model is established by counting both sets.
Scale	Rational number (Ratio scale, The extent to the business process describes all aspects of the real world excerpt relevant for the SRS audience, on a scale of $[0; \infty)$. A value of 0 denotes that the model is complete, while values of 1 and 2 denote that there are as many respectively twice as many relevant real world objects as there are elements in the model.)
Interpretation	Authors do not explicitly specify an interpretation procedure. Results during evaluation yielded rates between 0.00 and 13.9 (mean: 3.52, median: 2.14), also depending on the model element. Therefore, assuming that the models used in the evaluation resemble the models for the system under consideration, one may interpret the results particular to this "training set". For both kinds of interpretation, absolute and relative to a training set, further empirical evidence is required.
Recommendations for Action	Missing elements are identified and should be added to the description.
Threats to Validity	• The metric is based on the quality manager's judgment about the completeness of the model w.r.t. the real world excerpt. Therefore, the measurement is subjective and hence potentially lack robustness and consistency, due to the quality manager's perception or knowledge of the real-world domain. Furthermore, misinterpretations may also lead to flawed results. This is amplified in case no precise guidelines for judgment are given, which are rare for semantic model issues. • The metric measures perceived completeness from the viewpoint of the quality manager, therefore a potential gap between actual and perceived quality has to be considered especially since in many practical circumstances the picture of the future system operating in the real world is often quite vague. Furthermore, the metric does not account for the extent to which a model element is incomplete i.e., it does not distinguish between a minor and major omission. Therefore, the metric can only provide a limited view on whether a SRS is sufficiently complete or not. Furthermore, an interpretation relative to a training set bears the problem that it may lead to false conclusion, e.g., because of the limited expressiveness of the training set on the general quality. • If relative prescriptive reference values are used, it must be made sure that the training set adequately resembles the system under consideration and project context.

B.14. BPM Elements Redundancy Ratio [Overhage et al., 2012]

Assessed (Measured) Attribute	Concise (Business Process Description Element Redundancy: The extent to which the business process description contains redundant elements)
Data Collection	Authors do not explicitly specify a measurement process; Therefore, we will assume a manual measurement process by means of quality managers (e.g., requirements engineers) on the business process description (both, using natural and graphical languages) as suggested in the paper's evaluation. Therefore, "all carriers of meaning [here: model elements] which are unnecessarily depicted repeatedly in a business process model" by the quality manger. "This, for instance, applied to any activities that are recurrently depicted in individual threads of parallel flows". Finally, the ratio between elements judged redundantly to all elements is established based on the count of those elements and the analysis results.
Scale	Rational number (Ratio scale, The extent to which a business process description contains redundant elements, on a scale of $[0; 1]$. A value of 1 denotes that every element is (at least once) replicated, while a value of 0 denotes that the description is free of redundant elements.)
Interpretation	Authors do not explicitly specify an interpretation procedure. Results during evaluation yielded rates between 0.53 and 2.3 (mean: 1.45, median: 1.53). Therefore, assuming that the models used in the evaluation resemble the models for the system under consideration, one may interpret the results particular to this "training set". For both kinds of interpretation, absolute and relative to a training set, further empirical evidence is required.
Recommendations for Action	Redundant elements are identified. One potential step would be to use references instead of duplication.
Threats to Validity	• The metric is based on the quality manager's judgment. In terms of redundancy, replicated items on the semantic level (i.e., not by looking at the symbols and words only) might be both identified as redundant or actual redundancies may be overlooked, due to a misinterpretation or lack of understanding of the real world excerpt. • The measure highly depends on the quality manager, in particular her knowledge of the domain and the terminology used in the model. Any interpretation, positive or negative, must therefore take this risk into account. Furthermore, an interpretation relative to a training set bears the problem that it may lead to false conclusion, e.g., because of the limited expressiveness of the training set on the general quality. • If relative prescriptive reference values are used, it must be made sure that the training set adequately resembles the system under consideration and project context.

B.15. BPM Label Consistency Ratio [Overhage et al., 2012]

Assessed (Measured) Attribute	Lexically Consistent
Data Collection	Authors do not explicitly specify a measurement process; Therefore, we will assume a manual measurement process by means of quality managers (e.g., requirements engineers) on the business process description (both, using natural and graphical languages) as suggested in the paper's evaluation. Therefore, the quality manager analyzes all pairs of labels for lexical inconsistency, i.e., whether different synonymous labels of model elements are used, and potentially mark those labels as inconsistent. Finally, the ratio of all labels marked as inconsistent to all labels occurring in the model is computed.
Scale	Rational number (Ratio scale, The extent to which the labels of business process description elements are lexically consistent, i.e., the same terms are used throughout the description. The scale is [0;1], where 0 denotes that all labels are lexically consistent.)
Interpretation	Authors do not explicitly specify an interpretation procedure. Results during evaluation yielded rates between 0.97 and 13.36 (mean: 5.89, median: 3.35). Therefore, assuming that the models used in the evaluation resemble the models for the system under consideration, one may interpret the results particular to this "training set". For both kinds of interpretation, absolute and relative to a training set, further empirical evidence is required.
Recommendations for Action	Lexically inconsistent labels are identified.
Threats to Validity	• The metric is based on the quality manager's judgment, which is not a trivial task since it involves the correct identification of all synonyms for a real world excerpt. Therefore, the metric is subjective and highly depends on the quality manager's knowledge of the real world excerpt. • The measure highly depends on the quality manager, in particular her knowledge of the domain and the terminology used in the model. Any interpretation, positive or negative, must therefore take this risk into account. Furthermore, an interpretation relative to a training set bears the problem that it may lead to false conclusion, e.g., because of the limited expressiveness of the training set on the general quality. • If relative prescriptive reference values are used, it must be made sure that the training set adequately resembles the system under consideration and project context.

B.16. Checklist-based Document Completeness [Matulevicius et al., 2010]

Assessed (Measured) Attribute	Information Complete
Data Collection	Authors presume the existence of content templates for the SRS, and provide such as a starting point in (Matulevicius, Kamseu, and Habra: "Measuring Open Source Documentation Availability", CONQUEST 2009 Proceedings). The SRS is then (a) checked whether the content items has been filled in and (b) the level of detail is rated on a scale of [0<1<2<3] denoting the level of detail from "lowest" to "highest", both manually by a quality manager.
Scale	Rational number (Ratio scale, The extent to which a SRS contains "complete information, presented at the complete level of detail", on the scale [0.0;1.0] where 0 and 1 denote minimal respectively perfect completeness.)
Interpretation	In the present paper, authors propose the use of four discrete levels for interpretation defined using prescriptive reference values. The levels describe the relative level of organization of the SRS with respect to 28 open-source projects, which were empirically investigated in the paper.
Reference Value	A value of [0.0,21.5) denotes that the information is "not available" due to lack of content, detail or a combination of both (i.e. at the level of the poorest 15% of investigated open-source projects). A value of [21.5;37.0) is interpreted that the documentation availability is "limited" due to a relatively poor level of completeness (i.e., that it is worse than average). A value of [37.0,44.5) is interpreted that the document completeness is average, while a value of [44.5,1.0] denote high degree of completeness in the SRS document.
Recommendations for Action	Part of the result should be the template information that is not specified at all, or only with low detail.
Threats to Validity	• The quality manager's judgment is highly subjective, in particular because the "level of detail" is not defined despite the catchwords "low, medium, high". A quality manager has no particular robust and repeatable procedure to follow. • The metric is an aggregation of quantification of two distinct issues, namely that some content is not specified at all, or that it is not detailed enough. Therefore, the interpretability of a single value is limited. Furthermore, since the metric uses summation and normalization, negative outliers might be masked by the metric. Hence, the quality manager must make sure not to conclude that a high degree of "completeness" means that the important information is specified in detail. • For external validity, the applied templates and reference set might need to be changed and in case of the latter, reference values reevaluated.

B.17. Checklist-based Document Organization [Matulevicius et al., 2010]

Assessed (Measured) Attribute	Organized
Data Collection	Authors presume the existence of a checklist regarding the organization of the document, and provide one as a starting point in (Matulevicius, Kamseu, Habra, "Measuring Open Source Documentation Availability", CONQUEST 2009 Proceedings) consisting of 13 questions, of which 8 are optional (e.g., whether tables are annotated with an identifier is only applicable to SRS containing tables). The checklist is then applied by a quality manager manually, and each question is answered with "yes" or "no". The measurement value is obtained by dividing the number of questions answered with "yes" by the number of all questions applicable to the SRS.
Scale	Rational number (Ratio scale, The extent to which a SRS is organized according to a raters decision regarding pre-defined questions of a checklist.)
Interpretation	In the present paper, authors propose the use of four discrete levels for interpretation defined using prescriptive reference values. The levels describe the relative level of organization of the SRS with respect to 28 open-source projects, which were empirically investigated in the paper.
Reference Value	A value of [0.0,21.5) denotes that the information is "not available" due to very poor level of organization (i.e. at the level of the poorest 15pct of investigated open-source projects). A value of [21.5;37.0) is interpreted that the documentation availability is "limited" due to a relatively poor level of organization (i.e., that it is worse than average). A value of [37.0,44.5) is interpreted that the document organization is average, while a value of [44.5,1.0] denote high degree of organization in the SRS document.
Recommendations for Action	The actionability of the metric boils down to the actionability of the individual questions of the checklist. Here, the checklist provided does only contain items that are actionable, most of them in a very obvious way (e.g., insert chapter summaries, insert missing identifiers fir tables and figures)
Threats to Validity	• The evaluation of the checklist is subjective due to the ambiguity and vagueness of the checklist. • A checklist is inherently incomplete; Hence, some items are missing (e.g., the question "is the document organized in sections and subsections?" is included, but semantic coherence is not part of it) or lack discriminative power due to yes/no questions (e.g., "do cross-references exist between chapters in the document?"). Moreover, since the prescriptive reference values are defined according to an empirical evaluation of a reference set, this set might not expose the full spectrum or distribution of SRS documentation the quality manager (interpreter) has in mind, resulting in wrong conclusions. • For external validity, the predefined checklist and the reference set might need to be changed and in case of the latter, reference values reevaluated.

B.18. Comment Frequency (CF) [Fabbrini et al., 2000, Fantechi et al., 2002]

Assessed (Measured) Attribute	Pragmatic Quality
Data Collection	No description of the data collection method is given. To the best of our experience, we would imagine that there exists a technical or syntactical construct denoting both a requirement as one unit (e.g., subsections or templates per requirement) and comments added to those requirements (e.g., using designated template placeholders or word processor features)
Scale	Rational number (Ratio scale, Ratio of commented requirements to all requirements)
Interpretation	The comment frequency should be compared to prescriptive reference values which are project-specific. Furthermore, it is not specified whether a higher comment frequency is an indicator for high or low quality.
Reference Value	Specified in [fantechi2002application]: The reference value (interval) of 0.1 - 03 is "derived from the good practices of NL requirements".
Threats to Validity	• Violations of the syntactic or technical constructs for documenting requirements or comments lead to flawed results. For instance, adding comments as plain text as part of the requirements description although a designated template placeholder is provided) • The kind of comments associated with requirements are not distinguished, and may therefore provide a flawed assessment of the SRS quality. Therefore, comments should only count if they provide a significant help to understand the requirements by providing domain explanations, external documentation, etc.

B.19. Comparative Phrases Smell [Femmer et al., 2014a]

Assessed (Measured) Attribute	Quantitatively Precise (Use of Comparative Phrases: The extent to which words "which express a relation of the system to other systems" are contained within the requirements specification.)
Data Collection	In a first step, a natural language specification is parsed into single requirements. Afterwards, the part-of-speech (POS) for each word of every sentence is determined using a NLP tool. Subsequently, every word which is either (i) tagged as an adverb and adjective and in comparative form (determined by a morphological analysis), or (ii) is tagged as a conjunction of comparison, is reported and added to an overall count. According to the tool presented in the paper, this process is automated using the Stanford POS tagger and a morphological analysis.
Scale	Natural number (Absolute scale, Extent to which comparative phrases occur in the SRS.)
Interpretation	According to the authors, any occurrence of a comparative phrase is reported to the user, who is responsible to ultimately judge whether the finding is indeed an error or not. Since the authors do not explicitly specify a scale, neither a prescriptive reference value nor an interpretation procedure depending on the number of occurrences is given.
Recommendations for Action	Resolve identified terms, potentially with stakeholders

B.20. Completeness and Ambiguity of Object Flow (CAOF) [Kenett, 1996]

Assessed (Measured) Attribute	Feature Complete (Completeness and Ambiguity of Object Flow: The extent to which the flow of objects in NL-SRS sentences are described completely and unambiguously.)
Data Collection	For a NL-SRS, each sentence is (presumably manually, but not specified in the publication) decomposed into attributes, e.g. the initiator of an action, the action, the object, etc. For any sentence with an object, its flow (in terms of source and destination) can be specified and is rated as a source and destination attribute, respectively. Furthermore, for any sentence, missing source or destination attributes are identified (presumably using manual reviews). In addition, every source or destination attribute contained in the SRS is rated as ambiguous or not.
Scale	Rational number (Ratio scale, Ratio between missing and ambiguous source attributes, and all attributes specified in the SRS)
Interpretation	Values between 0 and +INF are obtainable (+INF due to the number of missing attributes can exceed the number of source attributes), with 0 being interpreted as the best value, meaning no object flow information is missing or ambiguous. The author mentions that practical experiences yielded thresholds (reference values), but did not disclose those in the publication.
Recommendations for Action	Ambiguous and/or missing attributes are identified, and therefore, can be targeted by corrective action.
Threats to Validity	• Ambiguity not only depends on the syntax and semantics of the SRS, but also depends on the reader of the SRS. Therefore, the reviewer might consider an attribute as ambiguous while it has a unique meaning to the receipee of the SRS, and vice versa. In addition, the number of missing arguments might be wrong since it is a quite difficult task. In particular, given the bounded knowledge of stakeholders on the actual requirements, the number of missing attributes might be indeed higher compared to the result of the review process. • The reference values seem to depend on the project, since the knowledge and boundedness of the problem space can significantly influence the number of missing attributes. Furthermore, ambiguity is reader-dependent and indluenced by domain knowledge.

B.21. Completeness and Ambiguity of Object Flow (CAOF) [Kenett, 1996]

Assessed (Measured) Attribute	Unambiguous (Completeness and Ambiguity of Object Flow: The extent to which the flow of objects in NL-SRS sentences are described completely and unambiguously.)
Data Collection	For a NL-SRS, each sentence is (presumably manually, but not specified in the publication) decomposed into attributes, e.g. the initiator of an action, the action, the object, etc. For any sentence with an object, its flow (in terms of source and destination) can be specified and is rated as a source and destination attribute, respectively. Furthermore, for any sentence, missing source or destination attributes are identified (presumably using manual reviews). In addition, every source or destination attribute contained in the SRS is rated as ambiguous or not.
Scale	Rational number (Ratio scale, Ratio between missing and ambiguous source attributes, and all attributes specified in the SRS)
Interpretation	Values between 0 and +INF are obtainable (+INF due to the number of missing attributes can exceed the number of source attributes), with 0 being interpreted as the best value, meaning no object flow information is missing or ambiguous. The author mentions that practical experiences yielded thresholds (reference values), but did not disclose those in the publication.
Recommendations for Action	Ambiguous and/or missing attributes are identified, and therefore, can be targeted by corrective action.
Threats to Validity	• Ambiguity not only depends on the syntax and semantics of the SRS, but also depends on the reader of the SRS. Therefore, the reviewer might consider an attribute as ambiguous while it has a unique meaning to the recipe of the SRS, and vice versa. In addition, the number of missing arguments might be wrong since it is a quite difficult task. In particular, given the bounded knowledge of stakeholders on the actual requirements, the number of missing attributes might be indeed higher compared to the result of the review process. • The reference values seem to depend on the project, since the knowledge and boundedness of the problem space can significantly influence the number of missing attributes. Furthermore, ambiguity is reader-dependent and influenced by domain knowledge.

B.22. Compound Accuracy Measure based on Sentence Parts (ComAM-SP) [Kenett, 1996]

Assessed (Measured) Attribute	*Multiple attributes (compound measure)*
Data Collection	Compound, weighted metric obtained from the following metrics: Ambiguity of Sentence Parts, TBD Frequency and Missing Conditions/Constraints in Sentences
Scale	Rational number (Ratio scale, The extent to which the SRS is "accurate", on the scale [0; 65.5]. A value of 0 denotes maximal "accuracy", while a value of 65 denotes maximal "vagueness" or lack of "accuracy." Note that "accuracy" here is defined according to the elementary metrics used in this compound metric, while might not reflect the common understanding of "accuracy" of a SRS.)
Interpretation	Values between 0 and +INF are obtainable, where 0 denotes perfect readability The values are to be interpreted against prescriptive reference values (thresholds), based on practical experiences but undisclosed in the publication.
Threats to Validity	• See individual metrics • The selection of elements to be contained in the compound metric seems questionable. Therefore, in any case, the individual metrics subsumed here must be ensured to measure the desired quality adequately. Furthermore, adding up the individual metrics against each other can obfuscate significant quality concerns of individual aspects. • The weights used for the compound metric might differ across contexts.

B.23. Compound Completeness Measure based on Sentence Parts (ComCM-SP) [Kenett, 1996]

Assessed (Measured) Attribute	Information Complete
Data Collection	Compound, weighted metric obtained from the following metrics: Missing Sentence Parts, Ambiguous Sentence Parts, Completeness and Ambiguity of Object Flow, Number of Sentence Part Types used and TBD Frequency.
Scale	Rational number (Ratio scale, The extent to which the SRS is complete, measured as a compound metric on a scale of [0;52], where 0 denotes the maximal completeness and 52 the maximal incompleteness.)
Interpretation	Values between 0 and +INF are obtainable, where 0 denotes perfect completeness. The values are to be interpreted against prescriptive reference values (thresholds), based on practical experiences but undisclosed in the publication.
Threats to Validity	• See individual metrics • The selection of elements to be contained in the compound metric seems questionable. Therefore, in any case, the individual metrics subsumed here must be ensured to measure the desired quality adequately. Furthermore, adding up the individual metrics against each other can obfuscate significant quality concerns of individual aspects. • The weights used for the compound metric might differ across contexts.

B.24. Compound Readability Measure based on Sentence Parts (ComRM-SP) [Kenett, 1996]

Assessed (Measured) Attribute	Pragmatic Quality (Readability of Natural Language Texts: The ease of understanding natural language texts, i.e., the ease of deriving a correct meaning of text by reading where the individual concepts are understood in principle by the reader.)
Data Collection	Compound, weighted metric obtained from the following metrics: Completeness and Ambiguity of Object Flow and Percentage of Descriptive Information
Scale	Rational number (Ratio scale, The extent to which the SRS is readability, measured as a compound metric on a scale of [-2.68; 42], where -2.68 denotes maximal readability and 42 the least readability. A value of 0 is considered as the optimal readability, which results from the object object flow being both complete and unambiguous, and the descriptive statements making up 1/3 of all statements.)
Interpretation	Values between 0 and +INF are obtainable, where 0 denotes perfect readability The values are to be interpreted against prescriptive reference values (thresholds), based on practical experiences but undisclosed in the publication.
Threats to Validity	• See individual metrics • The selection of elements to be contained in the compound metric seems questionable. Therefore, in any case, the individual metrics subsumed here must be ensured to measure the desired quality adequately. Furthermore, adding up the individual metrics against each other can obfuscate significant quality concerns of individual aspects. • The weights used for the compound metric might differ across contexts.

B.25. Concise in Number of Pages (CNP) [Davis et al., 1993a, Wilson et al., 1997]

Assessed (Measured) Attribute	Concise (Size: The extent to which a SRS is considered "small" in size.)
Data Collection	The number of pages (electronically stored or printed) are counted.
Scale	Rational number (Ordinal scale, Hyperbole of size, normalized on a scale from 0 to 1)
Interpretation	The values are interpreted in a range from 0 to
Threats to Validity	• First and foremost, the size in terms of pages is hardly comparable between different projects, because it is not specified as the difference between the minimum size and actual size per project-specific SRS. Hence, a different system under consideration may require more or less amount in terms of pages to describe it. Furthermore, different templates and word processors/tools lead to different sizes in terms of pages. Therefore, the generalizability is limited

B.26. Consistency of Specified Functionality (CSF) [Davis et al., 1993a]

Assessed (Measured) Attribute	Semantically Consistent
Data Collection	For all functional requirements, specified in terms of functions mapping system states and inputs to outputs, count the number of unique (not counting redundant) functions. Furthermore, count the number of those (remaining) functions which are non-deterministic, i.e., which specify different outputs for the same inputs and system states.
Scale	Rational number (Ratio scale, Percentage of unique functions which are also consistent)
Interpretation	The values are interpreted in a range from 0 (100% internally inconsistent) to 1 (100pct internally consistent). However, no reference value in-between is given.
Recommendations for Action	Each non-deterministic transition is a consistency violation and is subject to further resolution.
Threats to Validity	• Identification of equivalent inputs or system states not trivial. E.g., synonyms could be used for inputs, or different data formats ("1" vs. "1.0") • A reference value other than 1 is not provided and might be hard to justify.

B.27. Correct BPM Elements Ratio [Overhage et al., 2012]

Assessed (Measured) Attribute	Semantically Correct
Data Collection	Authors do not explicitly specify a measurement process; Therefore, we will assume a manual measurement process by means of quality managers (e.g., requirements engineers) on the business process description (both, using natural and graphical languages) as suggested in the paper's evaluation. Therefore, any element is analyzed for correctness, i.e., whether or not it is inconsistent with the real world excerpt according to the quality manager. Finally, the ratio of incorrect elements to all elements is established by counting both sets.
Scale	Rational number (Ratio scale, The extent to which the business process description correctly represents the real world excerpt, on a scale of $[0; 1$. A value of 0 denotes that every element of the model is correct, while a value of 1 denotes that every element of the model contradicts a fact of the real world excerpt.)
Interpretation	Authors do not explicitly specify an interpretation procedure. Results during evaluation yielded rates between 0.00 and 2.31 (mean: 0.85, median: 0.54), also depending on the model element. Therefore, assuming that the models used in the evaluation resemble the models for the system under consideration, one may interpret the results particular to this "training set". For both kinds of interpretation, absolute and relative to a training set, further empirical evidence is required.
Recommendations for Action	Incorrect elements are identified and should be correct, potentially by asking the stakeholder and/or domain experts.
Threats to Validity	• The metric is based on the quality manager's judgment about the correctness of the model w.r.t. the real world excerpt. Therefore, the measurement is subjective and hence potentially lack robustness and consistency, due to the quality manager's perception or knowledge of the real-world domain. Furthermore, misinterpretations may also lead to flawed results. This is amplified in case no precise guidelines for judgment are given, which are rare for semantic model issues. • The metric measures perceived correctness from the viewpoint of the quality manager, therefore a potential gap between actual and perceived quality has to be considered. Furthermore, the metric does not account for the extent to which a model element is incorrect, i.e., it does not distinguish between a minor flaw and a major incorrectness. Therefore, the metric can only provide a limited view on whether a SRS is sufficiently correct or not. Furthermore, an interpretation relative to a training set bears the problem that it may lead to false conclusion, e.g., because of the limited expressiveness of the training set on the general quality. • If relative prescriptive reference values are used, it must be made sure that the training set adequately resembles the system under consideration and project context.

B.28. Deviation of Requirements Preference Estimations (VDV) [Kaiya et al., 2002]

Assessed (Measured) Attribute	Unambiguous
Data Collection	The authors assume the presence of an attributed goal-graph according to their specific meta-model, with root elements (no predecessors) representing stakeholder demands and leaf elements representing finalized goals, i.e., refined requirements requiring certain functionality or characteristics of the (software) system. For any requirement (=finalized goal), all stakeholders must provide their personal preference as well as estimate the preference provided by all other stakeholders, rated on a scale from -10 (complete rejection) to +10 (complete approval). This annotation data is directly used to compute the aggregated metric described by the pseudo-formula, where mu denotes the arithmetic mean and AvgDev the average of absolute deviation regarding the mean value.
Scale	Rational number (Ratio scale, The scale is a rational number between 0 and 1 which expresses the average deviations of the requirement preference estimations provided by the stakeholders. A value of 1 denotes perfect estimation of others' requirement preferences, while a value of 0 denotes maximal disagreement betwenn all stakeholders.)
Interpretation	The authors do not specify an interpretation procedure. Since a single value is obtained, prescriptive reference values might work. However, because of the metric's complexity and limited current understanding of causality, empirical investigations are needed.
Recommendations for Action	Not provided by the author; Requirements with alarming deviations should be discussed by the involved stakeholders to confirm and resolve potential ambiguities.
Threats to Validity	• The estimation agreement depends also on the fact that stakeholders know each others preferences and goals. Hence, it depends on the team and project constellation as well as the domain.

B.29. Deviation of Stakeholder Requirements Preferences (HDV) [Kaiya et al., 2002]

Assessed (Measured) Attribute	Agreed
Data Collection	The authors assume the presence of an attributed goal-graph according to their specific meta-model, with root elements (no predecessors) representing stakeholder demands and leaf elements representing finalized goals, i.e., refined requirements requiring certain functionality or characteristics of the (software) system. For any requirement (=finalized goal), all stakeholders must provide their personal preference as well as estimate the preference provided by all other stakeholders, rated on a scale from -10 (complete rejection) to +10 (complete approval). This annotation data is directly used to compute the aggregated metric described by the pseudo-formula, where mu denotes the arithmetic mean and AvgDev the average of absolute deviation regarding the mean value, which essentially measures the deviations of preferences per requirement as specified by the stakeholders.
Scale	Rational number (Ratio scale, The scale is a rational number between 0 and 1 which expresses the average deviations of the requirement preferences of the different stakeholders. A value of 1 denotes perfect agreement on the requirements importance/preference, while a value of 0 denotes maximal disagreement between all stakeholders.)
Interpretation	The authors do not specify an interpretation procedure. Since a single value is obtained, prescriptive reference values might work. However, because of the metric's complexity and limited current understanding of causality, empirical investigations are needed.
Recommendations for Action	Conflicting requirements can be identified, which are subject to further inspection and must be resolved.
Threats to Validity	• The metric also includes the stakeholders' estimations, which depends on the fact that stakeholders know each others preferences and goals. Hence, it depends on the team and project constellation as well as the domain.

B.30. Dice Similarity Measure (Dice-Sim) [och Dag et al., 2001a]

Assessed (Measured) Attribute	Organized (Dependencies among requirements: The number of dependencies among requirements. Dependencies investigated comprise: (mutual and unidirectional) requirement, impact on costing, impact on customer value, mutual exclusion)
Data Collection	In a first step, stemming and stop-word-removal (frequently occurring words, e.g., 'a', 'the') is performed for all requirements by a NL-SRS tool. In a second step, requirements are compared pairwise for similarity by computing the Dice coefficient, i.e., by counting the share of words occurring in both requirements to the sum of words of each requirement.
Scale	Rational number (Ratio scale, A rational number representing the extent to which two requirements are similar, where 1 denotes perfect similarity and 0 no similarity at all.)
Interpretation	The obtained similarity coefficient is compared against a prescriptive reference value (threshold), and all values above the threshold are regarded as (potential) duplicates. The authors do not provide a metric for the extent of duplicates in a complete SRS.
Reference Value	Multiple reference values are investigated. The authors do not recommend a distinct reference value but advocate the selection based on the importance of finding duplicates. According to our understanding, a reference value of 0.75 provides an accuracy rate of 99.9pct, with 102 false positives (spurious warnings,) and 69 false negatives (unidentified duplicates), out of 615000 pairs of requirements)
Recommendations for Action	Duplicates are identified, and hence, can be removed or referenced
Threats to Validity	• Dictonary-based stop-word removal is incomplete. Also, typos etc. are unable to detect and can lead to flaws. • Authors claim that the reference value must be chosen based on the context and appropriate false positive/negative rates. However, no systematic approach to select a threshold is presented.

B.31. Dice Similarity Measure (Dice-Sim) [och Dag et al., 2001a]

Assessed (Measured) Attribute	Concise (Requirement Duplication: The number of clones or duplicated requirements occurring in a requirement specification.)
Data Collection	In a first step, stemming and stop-word-removal (frequently occurring words, e.g., 'a', 'the') is performed for all requirements by a NL-SRS tool. In a second step, requirements are compared pairwise for similarity by computing the Dice coefficient, i.e., by counting the share of words occurring in both requirements to the sum of words of each requirement.
Scale	Rational number (Ratio scale, A rational number representing the extent to which two requirements are similar, where 1 denotes perfect similarity and 0 no similarity at all.)
Interpretation	The obtained similarity coefficient is compared against a prescriptive reference value (threshold), and all values above the threshold are regarded as (potential) duplicates. The authors do not provide a metric for the extent of duplicates in a complete SRS.
Reference Value	Multiple reference values are investigated. The authors do not recommend a distinct reference value but advocate the selection based on the importance of finding duplicates. According to our understanding, a reference value of 0.75 provides an accuracy rate of 99.9pct, with 102 false positives (spurious warnings,) and 69 false negatives (unidentified duplicates), out of 615000 pairs of requirements)
Recommendations for Action	Duplicates are identified, and hence, can be removed or referenced
Threats to Validity	• Dictionary-based stop-word removal is incomplete. Also, typos etc. are unable to detect and can lead to flaws. • Authors claim that the reference value must be chosen based on the context and appropriate false positive/negative rates. However, no systematic approach to select a threshold is presented.

B.32. External Documentation Consistency (EDC) [Davis et al., 1993a]

Assessed (Measured) Attribute	Semantically Correct (Consistency with external documentation: The extent to which the SRS is free of contradictions (inconsistencies) compared to all relevant, external documentation.)
Data Collection	For measurement, all related external documentation must be accessible. Every requirement in the SRS is compared to every statement in any of the related external documentation and analyzed for inconsistencies. Finally, the number of requirements which are consistent to external documentation are counted, and compared to the number of all requirements in the SRS. Details of the comparison procedure are, however, not specified.
Scale	Rational number (Ratio scale, Percentage of requirements in the SRS which are consistent with related external documentation.)
Interpretation	The values are interpreted in a range from 0 (every requirement is inconsistent with external documentation) and 1 (every requirement is consistent with external documentation). A reference value other than 1 is not given.,
Recommendations for Action	If an inconsistency is identified, it must be checked whether it indeed is a problem (may not be so because of deprecated external documentation). In case it is, the SRS must be fixed accordingly.
Threats to Validity	• The internal validity depends on two potential threats: (i) the identification of and access to the external documentation, and (ii) the validity of the comparison procedure itself. • A reference value other than 1 is not provided and might be hard to justify.

B.33. Flesh-Kincaid Readability Grade Level (FK-RGL) [Kenett, 1996, Wilson et al., 1997, Fabbrini et al., 2000, Génova et al., 2013]

Assessed (Measured) Attribute	Pragmatic Quality (Complexity of Natural Language Texts: The complexity of natural language texts w.r.t. the ease of deriving a correct meaning of text by reading where the individual concepts are understood in principle by the reader.)
Data Collection	The number of sentences, words, and syllables are counted, e.g., using a function in the electronic word processor or a simple tool.
Scale	Rational number (Interval scale, The obtained mathematical object, a rational number, is called the flesh reading grade level. It does not have a fixed origin.)
Interpretation	Values are interpreted against prescriptive reference values (e.g., 3 levels as proposed by Génova et al. [2013]), for which the smaller the measurement value the more readable the text. Depending on the reader, the reference values might have to be adjusted.
Reference Value	"A Reading Grade Level of 7 to 8 is considered standard. Values of 4 to 5 are considered easy, and 15 to 16 very difficult.". In addition, Wilson et al. [1997] referred to the same reference values, however, empirical results suggest that this is skewed when specifying "scientific subjects which tend to contain words of considerable length". For the analyzed SRSs, the mean was 10.76 (min: 7.80, max: 13.80, std-dev: 1.59)
Threats to Validity	• According to [genova2013framework], that the metric received numerous criticism when applied to SRS, because "being technical texts addressed to professionals, it is completely acceptable that they contain numerous long words". • Measurement results depend on the vocabulary of the domain, e.g., the frequent use of certain technical phrases, and hence reference values might need to be adapted.

B.34. Functional Aggregation Error [Espana et al., 2009]

Assessed (Measured) Attribute	Singular (Functional Aggregation: Extent to which functionality is wrongly aggregated into single functions.)
Data Collection	The authors assume the existence of unity criteria which describe whether a set of functions (including single functions) should be modeled together or not. During data collection, an expert modeling committee "analyze a given domain and agree a model that strictly follows best practices in modeling". Then, an independent reviewer uses the aforementioned unity criteria together with this model as an aid to decide for each individual function whether it violates one (or more) unity criteria, i.e., whether the functions should be decomposed, and counts those violations.
Scale	Natural number (Absolute scale, The scale denotes the amount of system functions which are wrongly aggregated according to the unity criteria, on a scale of $[0; +\infty]$. A value of 0 denotes that no functions are wrongly aggregated.)
Interpretation	In the paper, the authors use the metrics to compare two approaches, and therefore, two SRS. Therefore, prescriptive reference values might be used for single SRS obtained by building a knowledge base of previous SRS and their completeness.
Recommendations for Action	Functions to be decomposed are identified; Hence, the object and the action are available.
Threats to Validity	• Manual review might miss violations or misinterpret unity criteria

B.35. Functional Encapsulation Completeness [Espana et al., 2009]

Assessed (Measured) Attribute	Feature Complete (Functional Encapsulation Completeness: Extent to which all user-visible functions are specified.)
Data Collection	The authors refer to (Wieringa, 1996, "Requirements Engineering: frameworks for understanding"), which defines a function as "a service provided by the IS to its environment" but use the notion of "functional encapsulations" to "highlight the importance of determining the boundaries of the encapsulation". During data collection, an expert modeling committee "analyze a given domain and agree a model that strictly follows best practices in modeling". This modeled is compared against the one specified in the SRS by counting the number of system functions in each model.
Scale	Rational number (Ratio scale, The scale denotes the amount of system functions specified in the SRS normalized w.r.t. a reference model, on a scale of $[0; \infty+)$ where 0 denotes that no functions are specified, and 1 that the SRS specifies as many functions as the reference model. A value above denotes that the SRS actually specifies more functions than the reference model.)
Interpretation	In the paper, the authors use the metrics to compare two approaches, and therefore, two SRS. Therefore, prescriptive reference values might be used for single SRS obtained by building a knowledge base of previous SRS and their completeness.
Threats to Validity	• No matching between the functions of the reference model and the SRS is performed; Therefore both, the SRS containing incorrect requirements or strongly fragmented SRS (see below) might suggest higher functional encapsulation completeness. This hinders the use of strict thresholds.

B.36. Functional Fragmentation Error [Espana et al., 2009]

Assessed (Measured) Attribute	Singular (Atomic) (Functional Fragmentation: Extent to which functionality is wrongly split into multiple functions.)
Data Collection	The authors assume the existence of unity criteria which describe whether a set of functions (including single functions) should be modeled together or not. During data collection, an expert modeling committee "analyze a given domain and agree a model that strictly follows best practices in modelling". Then, an independent reviewer uses the aforementioned unity criteria together with this model as an aid to decide for each subset (cardinality >= 2) of functions whether they collectively violate one (or more) unity criteria, i.e., whether the functions should be jointly specified) and counts those violations.
Scale	Natural number (Absolute scale, The scale denotes the amount of system functions which are wrongly fragmented according to the unity criteria, on a scale of [0;INF+]. A value of 0 denotes that no functions are wrongly fragmented.)
Interpretation	In the paper, the authors use the metrics to compare two approaches, and therefore, two SRS. Therefore, prescriptive reference values might be used for single SRS obtained by building a knowledge base of previous SRS and their completeness.
Recommendations for Action	Functions to be aggregated (joined) are identified; Hence, the object and the action are available.
Threats to Validity	• Manual review might miss violations or misinterpret unity criteria. It remains unclear how to proceed with "non-minimal" subsets, since only "minimal" subsets should probably be counted.

B.37. Glossary Cicularity (GLC) [Duran et al., 2002]

Assessed (Measured) Attribute	Lexically Consistent (Glossary Circularity: The extent to which definitions of terms are based on different terms also contained in the glossary)
Data Collection	For every glossary item (definition), the number of references (not including self-references) to other glossary items are counted using an automated tool (in the case of the paper, XSLT was used). In addition, the total number of glossary items is counted.
Scale	Rational number (Ratio scale, A rational number representing the extent to which the glossary is cross-referenced. A value of 0 denotes minimal, i.e., no, cross-references among glossary items, and a value of X means that on average, every glossary item references X other glossary items.)
Interpretation	The obtained GLC measurement is compared against prescriptive reference values, and this comparison yields an interpretation for the quality of the glossary.
Reference Value	GLC < 1 indicates "a low quality glossary". "It seems clear that glossary items not referencing other glossary items, or referencing just a few ones (less than 2, for example), should be verified for potential problems". Therefore, a GLC < 2 is also considered check-worthy, while a GLC $>= 2$ is implicitly assumed by the authors to indicate "a good quality glossary".
Threats to Validity	• The mean value is suspect to outliers, and the interpretation can be flawed (positively and negatively) • The reference values are not empirically obtained but more or less a guess from the authors ("for example"). Different contexts would probably require different reference values.

B.38. Goal (Dis-)Satisfaction Estimate [Cailliau and van Lamsweerde, 2012]

Assessed (Measured) Attribute	Semantically Correct (System Goal Satisfaction Risk: The extent to which loss of satisfaction regarding system goals may occur)
Data Collection	Authors assume the presence of a goal-model which includes formal specifications of the goals (expressed as temporal logic formulas), identified obstacles to goal satisfaction, and the use of specific patterns for goal refinement where possible (e.g., milestone-driven decomposition). Furthermore, root goals are associated with a required probability of satisfaction (RPS: Goal -> [0;1]). During data collection, the probability of satisfaction of the obstacle condition is estimated for each leaf obstacle. However, the concrete procedure is not specified in any detail in the paper but the authors state to "rely on domain knowledge [...] - typically, through statistical data about past system behaviors". Next, the occurrence probabilities of all obstacles are calculated by up-propagation on the goal/obstacle tree, yielding the estimated goal satisfaction probability (denoted EPS: Goal -> [0;1], see paper for full description of technique). For any root goal with an associated required probability (RDS), the extent of (dis)satisfaction of the goal is obtained as the difference of RDS and EPS.
Scale	Rational number (Absolute scale, The extent the stakeholders' goals are satisfied respectively dissatisfied, on a scale of [-1;1]. Any negative value denotes goal satisfaction, while any positive denotes goal dissatisfaction (violation), while the (absolute) number denotes the extent as the difference in percent.)
Interpretation	The set of all requirements (leaf goals) for which a root goal is associated with a positive value (i.e., is unsatisfied) are regarded as (potentially) incorrect due to not being achievable in its current form and requires further actions. In turn, any negative value or zero is regarded as satisfiable under the assumptions in the goal model and hence are interpreted as being correct with respect to achievability. The authors interpret measurements for single goals only – we applied the interpretation procedure to root goals.
Reference Value	Any requirements for which a root goal's measurement is within the interval (0;1] is considered (potentially) incorrect, while requirements for which all root goal's measurements are within [-1;0] are considered correct.
Recommendations for Action	In case of potentially incorrect requirements, the assumptions and requirements (both, the behavioral aspect as well as the required probability of it) must be reconsidered. This may include to correct or re-negoiate the demanded probabilities of system behavior or add requirements specific to avoid or limit the occurrence of obstacles.
Threats to Validity	• Obviously, the measurement results depend on the correct- and completeness of the goal model it is based on. In particular, the estimates of obstacle occurrence probabilities are left unspecified, and one must rely on domain knowledge and experience of experts rather than empirical data because such data is often not available or influenced by too many confluent factors. Therefore, the estimation step of measurement is highly subjective and lacks repeatability and falsifiability. • The interpretation procedure relies on the correctness and completeness of the goal model. This is particularly challenging regarding the assumptions and obstacles of the environment and the associated probabilities, due to the multitude of involved parameters and lack of reliable empirical data in practical settings. Therefore, any interpretation should include a validation of those model elements.

B.39. Goal Confidence Profile [Boness et al., 2011, 2008]

Assessed Attribute	Semantically Correct (Confidence in Feasibility and Adequacy: The extent to which the development team can be confident to have derived feasible and adequate goals)
Data Collection	Authors presume that a goal model exists, which essentially is a goal graph with explicit statement of assumptions. Furthermore, the goal model has annotated (relative or absolute costs) to each leaf element. During data collection, each node is associated with a confidence judgment on a four-point ordinal scale (None,Low,Mid,High) regarding the following aspects: Assumption Soundness (for any leaf node that is an assumption), Achievability (for any leaf node that is a goal), Refinement Soundness (for any element, HIGH per definition for the root) and stakeholder engagement (for any element). For each aspect, criteria are defined in the paper which must be present in order to justify the judgment in order to reduce subjectivity. However, important details are unspecified in the publication (e.g., the negotiation process when contradicting judgments exist from multiple judges). Next, Feasibility and Adequacy of goals is assessed according to a well-defined algorithmic procedure based on the combination of achievability and assumption soundness respectively refinement soundness and stakeholder engagement, and each goal is classified on a three-point scale (Proceed, (Proceed with) Caution, and Do not Proceed) by logically evaluating feasibility and adequacy results. For instance, a goal for either feasibility or adequacy is assessed as "Low" is rated as "Caution" only if the other is assessed as "High", otherwise if is rated as "Do not Proceed". Finally, a vector of relative cos accumulated in each of the three levels is obtained according to a provided formula.
Scale	Vector of rational numbers(Ratio scale, The extent to which a SRS contains goals which are "predicted to cause pain" respectively the confidence about the goals correctness regarding feasibility and adequacy. The measurement object obtained is a vector of three rational numbers each on rational scales [0;1], which define the cost-normalized share of goals which give confidence to proceed (1st element), to proceed with caution (2nd element), or lack the confidence to proceed (3rd element).)
Interpretation	Authors do not propose any concrete interpretation procedure to assess the correctness other than to inspect cases of "DontProceed" goals. In particular, no implications of the "Caution" area is given, and hence, it is up to interpretation what the authors want to suggest with the notion of "proceeding with caution".
Recommend. for Action	Individual goals for rework are identified, together with a hint of how to proceed based on the basic judgments, warrants and backings provided during data collection.
Threats to Validity	• Although the authors explicitly state that the judgments have to be supported up by warrants and backings, and constrain what kind of warrants and backings are admitted for each of the four aspects, they are still quite subjective. Furthermore, since no advice is given by the authors in terms of contradicting warrants from one or multiple judges, this is also a threat to internal validity. • Due to subjective judgments, only perceived but not actual correctness is measured. Furthermore, the measured aspects are only potential impact factors. For instance, stakeholder engagement can improve the adequacy of the system to satisfy stakeholder needs, but a SRS can be adequate without extensive stakeholder engagement (e.g., because of domain knowledge and/or prior long-term business relationships). Last but not least, the metric only takes into account four aspects, ignoring many others such as contradictions introduced by multiple stakeholders. • Stakeholder engagement depends on the principal willingness of stakeholders to participate during goal modeling; Depending on the project context (domain, customer relationship) this might vary and result in flawed results. Furthermore, since the judgments are a vital part, the domain expertise of the judges has a large impact on the judgments: different project contexts might lead to different results of the metric due to the judges' knowledge of the particular domain.

B.40. Imperatives per Subjects (IpS) [Wilson et al., 1997]

Assessed (Measured) Attribute	Information Complete (Extensiveness of Descriptions: The extensiveness of information provided in terms of the "amount" but not the "precision" of provided information.)
Data Collection	Imperatives ("shall", must (not)", "is required to", "are applicable", "responsible for", "will" and "should", ordered by their "strength as a forceful statement of a requirement" in descending order) are automatically counted by phrase-detection. Furthermore, "the number of subjects used in the specification document is a count of unique combinations and permutations of words immediately preceding imperatives in the source file.
Scale	Rational number (Ratio scale, $[0; \infty+)$, ratio of imperative phrases to unique subjects identified)
Interpretation	The authors suggest to compare this number against SRS of previous projects. However, no specific reference value was given.
Recommendations for Action	Identified potentially underspecified (incomplete) requirements
Threats to Validity	• Typos, synonyms and unspecified or implicit imperatives are not found by the automated procedure. Furthermore, the uniqueness of subjects is hard to ensure, because of typos, spelling, synonyms, permutations. • Different project contexts can have significant impact on the reference values, because the authors writing style or the use of automated tools to generate SRS documents can influence the imperatives used.

B.41. Imperatives per Subjects (IpS) [Wilson et al., 1997]

Assessed (Measured) Attribute	Pragmatic Quality (Extensiveness of Descriptions: The extensiveness of information provided in terms of the "amount" but not the "precision" of provided information.)
Data Collection	Imperatives ("shall", must (not)", "is required to", "are applicable", "responsible for", "will" and "should", ordered by their "strength as a forceful statement of a requirement" in descending order) are automatically counted by phrase-detection. Furthermore, "the number of subjects used in the specification document is a count of unique combinations and permutations of words immediately preceding imperatives in the source file.
Scale	Rational number (Ratio scale, $[0; \infty+)$, ratio of imperative phrases to unique subjects identified)
Interpretation	The authors suggest to compare this number against SRS of previous projects. However, no specific reference value was given.
Recommendations for Action	Identified potentially underspecified (incomplete) requirements
Threats to Validity	• Typos, synonyms and unspecified or implicit imperatives are not found by the automated procedure. Furthermore, the uniqueness of subjects is hard to ensure, because of typos, spelling, synonyms, permutations. • Different project contexts can have significant impact on the reference values, because the authors writing style or the use of automated tools to generate SRS documents can influence the imperatives used.

B.42. Implicit Subject Sentences (ISS) [Fabbrini et al., 2000]

Assessed (Measured) Attribute	Unambiguous (Lexically Unambiguous: The extent to which a SRS contains lexicals (words) which are ambiguous)
Data Collection	Subjects in all sentences in a NL-SRS are analyzed to contain a demonstrative adjective ("this, these, that, those"), pronouns ("it, they"), propositions ("above, below") and adjectives such as "previous, next, following, last, first"). Although not specified in detail, it appears to be automatable. In a second step, a manual verification is done by a reviewer to exclude "false positives".
Scale	Natural number (Absolute scale, Number of sentences with implicit subjects)
Interpretation	The obtained number of sentences containing options is compared to restrictive prescriptive reference values (called "qualit y requirements" for the indicators presented in the paper).
Reference Value	The reference value is 0, i.e., no implicit subjects are allowed
Recommendations for Action	Resolve vague phrases (with stakeholders)
Threats to Validity	• List of phrases is incomplete, algorithm may not detect certain occurrences (e.g., typos)

B.43. Imposed Flexibility Contraints Ratio [Overhage et al., 2012]

Assessed (Measured) Attribute	Semantically Correct (Unconstrained Model Flexibility of Control Flow in Business Process Descriptions: The extent to which the control flow is constrained in a business process description despite no actual constraint in the real-world)
Data Collection	Authors do not explicitly specify a measurement process; Therefore, we will assume a manual measurement process by means of quality managers (e.g., requirements engineers) on the business process description (both, using natural and graphical languages) as suggested in the paper's evaluation. Therefore, the "number of control flow constraints [is determined] during the comparison of the business process model with the underlying real world excerpt" by and according to the quality manager. According to the authors, "a constraint emerges each time when actually independent control flows are connected to each other in sequence". Finally, the ratio of the limited control flows to all control flows is calculated.
Scale	Rational number (Ratio scale, The extent to which a business process imposes constraints on the control flow which are not justified w.r.t. the real world excerpt, on a scale of [0;1]. A value of 1 denotes that every flow in the model is constrained.)
Interpretation	Authors do not explicitly specify an interpretation procedure. Results during evaluation yielded rates between 0.97 and 7.780 (mean: 4.48, median: 4.68). Therefore, assuming that the models used in the evaluation resemble the models for the system under consideration, one may interpret the results particular to this "training set". For both kinds of interpretation, absolute and relative to a training set, further empirical evidence is requied.
Recommendations for Action	The constrained control flow descriptions are identified,
Threats to Validity	• The metric is based on the quality manager's judgment about the flexibility of the model w.r.t. the real world excerpt. Therefore, the measurement is subjective and hence potentially lack robustness and consistency, due to the quality manager's perception or knowledge of the real-world domain. Futhermore, misinterpretations may also lead to flawed results. This is amplified in case no precise guidelines for judgment are given, which are rare for semantic model issues. • The metric measures perceived flexibility from the viewpoint of the quality manager, therefore a potential gap between actual and perceived quality has to be considered. Furthermore, an interpretation relative to a training set bears the problem that it may lead to false conclusion, e.g., because of the limited expressiveness of the training set on the general quality. • If relative prescriptive reference values are used, it must be made sure that the training set adequately resembles the system under consideration and project context.

B.44. Incomplete Sections (IS) [Costello and Liu, 1995]

Assessed (Measured) Attribute	Semantically Complete (Incompleteness of stated identified demands: The amount of (SRS) content identified to be required but (yet) not specified)
Data Collection	The SRS is scanned for "blank or omitted sections", and the metric is obtained by counting those occurrences.
Scale	Natural number (Absolute scale, Number of (sub-sections) left blank or omitted)
Interpretation	No interpretation procedure provided. However, it is stated that the metric "cannot provide a complete picture of the work remaining [...] [because] typically varying amounts of effort to address [the identified areas are required]. Thus, TBC metrics reports should be presented with an analysis of the difficulty (technical and organizational) of resolving each item".
Recommendations for Action	Contribute to the identified missing parts of the SRS
Threats to Validity	• An oracle for what (sub)sections are required must be available, complete and valid. What is considered blank? No other characters than white-spaces? A clear definition is required. • A reference value is not provided. 0 is the optimum, but may not be feasible in practice • The oracle of what sections are mandatory can vary from context to context, and hence, a valid oracle must be provided for every instance.

B.45. Informational Completeness [Etien and Rolland, 2005]

Assessed (Measured) Attribute	Feature Complete (Business Object Coverage: The extent to which all business (domain) objects are specified in the system's description of the SRS)
Data Collection	The authors assume the presence of a description/model of the business domain and the systems implementation according to specific meta-models (ontologies) as well as a mapping between elements of those models in terms of mapping between elements (equivalent concepts) and a representation mapping (a system element may represent certain aspects of a business element). For this particular metric, the business entities (here, we consider any object associated with the business process to be supported by the system under consideration according to the domain model), both all and only those for which a system class (i.e., an object in the data model) exists, are counted. Depending on the actual tool, this process can be automated.
Scale	Rational number (Ratio scale, The scale denotes the extent to which domain elements relevant for the business process to be supported by the system under consideration are also represented in the description of the system to be build. A value of 0 and 1 denotes all respectively no such domain elements are also represented in the system.)
Interpretation	The authors do no provide an explicit interpretation procedure. In the paper, a single value is computed, however, no preference values/thresholds are specified.
Recommendations for Action	"Corrective action should be taken to increase the information completeness ratio either by modifying the system to manipulate the new classes or by removing business class if they demonstrate to be of low value". Therefore, a stakeholder/domain export must be involved.

B.46. Inter-Section NounPhrase Coupling [Rine and Fraga, 2015, Din, 2008]

Assessed (Measured) Attribute	Organized (Inter-Section Linguistic Coupling: Extent to which a section is linguistically (lexically) related to other, distant sections of the SRS.)
Data Collection	The authors assume the presence of a natural-language SRS composed of requirements which are further organized into sections, and this structure is identifiable to the data collection procedure. During data collection, the input NL-SRS is tokenized (using regular expressions), the identified tokens are tagged using a part-of-speech tagger, and noun phrase (NP) chunks are identified, using regular expressions on the POS-tagged tokens. Noun-phrase chunks are non-overlapping (no portions of the text are shared among chunks) and non-exhaustive (not all portions of the text are allocated to chunks) terms (words or multi-words) which serve as noun phrases in sentences. Afterwards, stop-word chunks such as "the system", "the information", "the authorized users" are filtered out (concrete dictionary is not specified), and stemming and text normalization is applied to the remaining chunks. Next, "collections of adjacent sentences in a requirements section that share one or more NP chunks [called 'chunks'] are identified", and for each section and identified cluster, the centroid is computed. For each noun-phrase occurring in a section, the distance in terms of number of sentences to any equivalent noun-phrase occurring in different sections are computed (in case the noun-phrase within the section is part of a cluster, the cluster's centroid is used as the point of origin for the distance calculation). Finally, the cumulative sum is of all obtained distances per section is calculated.
Scale	Rational number (Ratio scale, The scale combines the number of noun-phrases contained within a section which are also contained in other sections of the SRS, with the distance in terms of number of sentences to those. In case a section does not have any NP chunks in common with other sections, the value is 0. The other extreme case is that a section contains no clusters and shares NP chunks with all the other sentences in the SRS, in which case coupling is N*N, where N denotes the number of sentences in the SRS.)
Interpretation	Authors do not specify an interpretation procedure.
Recommendations for Action	Section with coupling could be resolved by moving sentences from one section to another, or by rearranging the SRS for those section to be closer to each other.
Threats to Validity	• Typos can result in different noun-phrase chunks. • Does only consider noun-phrases while not considering e.g. actions expressed as verbs. This can lead to both false positives and negatives. Furthermore, linguistic (and lexical, in particular) coherence can only be an indicator for semantic coherence, e.g., because of synonyms or ambiguous terms. Hence, lexical consistency and unambiguity can have a positive influence on the metric's interpretation validity. • The linguistic analysis depends on the writing style of the SRS author.

B.47. Inter-Stakeholder Inconsistency Level [Martínez et al., 2008a,b]

Assessed (Measured) Attribute	Agreed (Potential and Actual Inter-Stakeholder Functional Consistency: Extent to which the stakeholder's functional requirements on the system are free of actual and potential (because of intentional or unintentional partiality in stakeholder perspectives) contradictions.)
Data Collection	Authors presume that N groups of stakeholders express their individual perspective on the system's requirements as temporal-logic formulae using a specific logic proposed by the authors called S(imple)CTL and based on a consistent vocabulary (e.g., by using a common dictionary) . Those formulas are then translated into multi-valued LTS, in which each action can either be allowed (value=1), disallowed (value=0) or unspecified (value=1/2). During data collection, the N models are combined into a common model, and for each state and action of the common model, the level of confrontation (CL) and level of uncertainty (UL) are computed as follows: For CL, the number of models in which the action in the given state is annotated as allowed (1) and disallowed (0) is counted and the minimum of both is considered the level of confrontation. Similarly, the number of models in which the action is not specified (1/2) for the given state is counted and constitutes the level of uncertainty (UL). Finally, the inconsistency level is computed based on UL and CL according to the pseudo-formula, in which odd(X) returns 1 if X%2=1, and 0 otherwise.
Scale	Rational number (Ratio scale, The scale denotes the extent of actual and potential inconsistency for a given state and action (see data collection method description) w.r.t. to the maximal inconsistency on the [0;1] interval, where 0 denotes no inconsistency and 1 maximal inconsistency.)
Interpretation	The authors do not provide reference values; However, examples for stakeholder values are given in conjunction with the measurement values to give an initial assessment of the measurements.
Recommendations for Action	The metric does pinpoint the state and action of the LTS, which can according to the authors be backtraced to the original SCTL formula. Therefore, the action must be processed further, and authors suggest several techniques to do so, all having in common some additional work though.
Threats to Validity	• The measurement depends on a common dictionary, therefore, this dictionary must be complete and unambiguous. Furthermore, it must be ensured that the requirements formulation as temporal formulae is correct. Last but not least, it is assumed that all individual views are consistent, but only the integrated view among stakeholders might lead to inconsistencies. • The metric actually is a composite measure of two inconsistency definitions: potential and actual inconsistency; Because the results are combined they are not distinguished in their severity; interpretations in the range of [0.1;1.0] are hard to make. This is further amplified by the fact that only one single value for unspecified (1/2) is given, which can constitute both a stakeholder's "i don't know" and a stakeholder's "i don't care", for which only the first case bears the problem of a potential inconsistency.

B.48. Inter-Stakeholder Inconsistency Level [Martínez et al., 2008a,b]

Assessed (Measured) Attribute	Semantically Consistent (Potential and Actual Inter-Functional Consistency among Stakeholders: Extent to which the stakeholder's functional requirements on the system are free of actual and potential (because of intentional or unintentional partiality in stakeholder perspectives) contradictions.)
Data Collection	Authors presume that N groups of stakeholders express their individual perspective on the system's requirements as temporal-logic formula using a specific logic proposed by the authors called S(imple)CTL and based on a consistent vocabulary (e.g., by using a common dictionary) . Those formulas are then translated into multi-valued LTS, in which each action can either be allowed (value=1), disallowed (value=0) or unspecified (value=1/2). During data collection, the N models are combined into a common model, and for each state and action of the common model, the level of confrontation (CL) and level of uncertainty (UL) are computed as follows: For CL, the number of models in which the action in the given state is annotated as allowed (1) and disallowed (0) is counted and the minimum of both is considered the level of confrontation. Similarly, the number of models in which the action is not specified (1/2) for the given state is counted and constitutes the level of uncertainty (UL). Finally, the inconsistency level is computed based on UL and CL according to the pseudo-formula, in which odd(X) returns 1 if X%2=1, and 0 otherwise.
Scale	Rational number (Ratio scale, The scale denotes the extent of actual and potential inconsistency for a given state and action (see data collection method description) w.r.t. to the maximal inconsistency on the [0;1] interval, where 0 denotes no inconsistency and 1 maximal inconsistency.)
Interpretation	The authors do not provide reference values; However, examples for stakeholder values are given in conjunction with the measurement values to give an initial assessment of the measurements.
Recommendations for Action	The metric does pinpoint the state and action of the LTS, which can according to the authors be backtracked to the original SCTL formula. Therefore, the action must be processed further, and authors suggest several techniques to do so, all having in common some additional work though.
Threats to Validity	• The measurement depends on a common dictionary, therefore, this dictionary must be complete and unambiguous. Furthermore, it must be ensured that the requirements formulation as temporal formulae is correct. Last but not least, it is assumed that all individual views are consistent, but only the integrated view among stakeholders might lead to inconsistencies. • The metric actually is a composite measure of two inconsistency definitions: potential and actual inconsistency; Because the results are combined they are not distinguished in their severity; interpretations in the range of [0.1;1.0] are hard to make. This is further amplified by the fact that only one single value for unspecified (1/2) is given, which can constitute both a stakeholder's "i don't know" and a stakeholder's "i don't care", for which only the first case bears the problem of a potential inconsistency.

B.49. Intra-Section NounPhrase Cohesion [Rine and Fraga, 2015, Din, 2008]

Assessed (Measured) Attribute	Organized (Intra-Section Linguistic Coherence: Extent to which all adjacent pairs of sentences of a section are linguistically (lexically) related to each other.)
Data Collection	The authors assume the presence of a natural-language SRS composed of requirements which are further organized into sections, and this structure is identifiable to the data collection procedure. During data collection, the input NL-SRS is tokenized (using regular expressions), the identified tokens are tagged using a part-of-speech tagger, and noun phrase (NP) chunks are identified, using regular expressions on the POS-tagged tokens. Noun-phrase chunks are non-overlapping (no portions of the text are shared among chunks) and non-exhaustive (not all portions of the text are allocated to chunks) terms (words or multi-words) which serve as noun phrases in sentences. Afterwards, stop-word chunks such as "the system", "the information", "the authorized users" are filtered out (concrete dictionary is not specified), and stemming and text normalization is applied to the remaining chunks. Next, "collections of adjacent sentences in a requirements section that share one or more NP chunks [called 'chunks'] are identified", and for each section and identified cluster, the number sentences contained in the cluster are counted, and the result minus one is called the "cluster size". The procedure is undisclosed, but appears to be implementable using sequential scanning on the sentences. Finally, the cohesion metric is computed by summing the size of all clusters in a section, divided by the number of sentences occurring in the section minus 1. If a section contains only one requirement, authors define the cohesion measure to be 1.
Scale	Rational number (Ratio scale, The extent to which adjacent sentences within a section share noun-phrase chunks, on a scale of [0;1]. Here, a value of 1 means that each pair of adjacent sentences share at least one noun-phrase chunk, while a value of 0 means that no adjacent sentences share any noun-phrase chunks.)
Interpretation	Authors do not specify an interpretation procedure.
Recommendations for Action	Clusters can be regarded as semantic gaps within a section, and are subject to consider for reorganization of the SRS.
Threats to Validity	• Typos can result in different noun-phrase chunks. • Does only consider noun-phrases while not considering e.g. actions expressed as verbs. This can lead to both false positives and negatives. Furthermore, linguistic (and lexical, in particular) coherence can only be an indicator for semantic coherence, e.g., because of synonyms or ambiguous terms. Hence, lexical consistency and unambiguity can have a positive influence on the metric's interpretation validity. • The linguistic analysis depends on the writing style of the SRS author.

B.50. Lexical (Semantic) Sentence Ambiguity [Kiyavitskaya et al., 2008]

Assessed (Measured) Attribute	Unambiguous (Lexical ambiguity of Individual Sentences: Extent to which a sentence's words have multiple interpretations without considering context.)
Data Collection	Authors presume the existence of a dictionary from which the number of different meanings for a word can be obtained (function meanings(w) = Number of meanings of w according to dictionary). The authors used the free dictionary WordNet for this purpose. Based on this dictionary, the number of meanings for all words occurring in a sentence S are summed. This task is automated using a tool written by the authors.
Scale	Natural number (Ratio scale, The scale denotes the total number of interpretations of all words in a sentence, on an interval [n;+INF] where n denotes the number of words of a sentence. A value of n denotes that every word has only one interpretation (i.e., every word is unambiguous), while a value of n+m denotes that there are in total m ambiguous interpretations in total for all words of the sentence.)
Interpretation	The authors provide reference thresholds against which the measurement value should be compared.
Reference Value	The authors provide the following reference values: "Ambiguity values less than or equal to 5 are assigned the green color, and the ambiguity values greater than or equal to 10 are assigned the red color; implicitly, ambiguity values greater than 5 but less than 10 are assigned the yellow color." However, according to our understanding, the values appear to be flawed, since it does not depend on the number of words in a sentence and seems to be unintentionally low; Therefore, we assume the reference value X to be assumed as a reference value of X+n, where n is the number of words of a sentence.
Recommendations for Action	Review highly ambiguous sentences.
Threats to Validity	• The measurement depends on the dictionary used; According to authors' experiments, the WordNet dictionary is suitable. • Lexical (semantic) sentence ambiguity does not include context information, which potentially removes a plethora of ambiguities. Moreover, the reference values do not take sentence length into account in the sense that a longer sentence will probably be more ambiguous simply because it contains more words. • Depending on the vocabulary of a certain domain, lexical ambiguity might be more or less predominant.

B.51. Lines-of-Text per Imperative (LpI) [Wilson et al., 1997]

Assessed (Measured) Attribute	Concise
Data Collection	Imperatives ("shall", must (not)", "is required to", "are applicable", "responsible for", "will" and "should", ordered by their "strength as a forceful statement of a requirement" in descending order) are automatically counted. Furthermore, the lines of text is counted by a tool.
Scale	Rational number (Ratio scale, $[0; \infty+)$, phrases count (imperatives) and size measure (lines of text))
Interpretation	The authors suggest to compare this number against SRS of previous projects. However, no specific reference value was given.
Recommendations for Action	Identified potentially bloated requirements
Threats to Validity	• Typos, synonyms and unspecified or implicit imperatives are not found by the automated procedure • Different project contexts can have significant impact on the reference values, because the authors writing style or the use of automated tools to generate SRS documents can influence the imperatives used.

B.52. Linked Communications completeness [Espana et al., 2009]

Assessed (Measured) Attribute	Feature Complete (Linked Communications completeness: Extent to which all communications between the system and external actors (persons, systems) are specified.)
Data Collection	The authors refer as "linked communication to the message conveyance that is triggered by the occurrence of an event (the use or activation of a function) and by which the IS informs an actor of this occurrence". During data collection, an expert modeling committee "analyze a given domain and agree a model that strictly follows best practices in modeling". This modeled is compared against the one specified in the SRS by counting the number of linked communications in each model.
Scale	Rational number (Ratio scale, The scale denotes the amount of linked communications specified in the SRS normalized w.r.t. a reference model, on a scale of [0;INF+] where 0 denotes that none are specified, and 1 that the SRS specifies as many linked communications as the reference model. A value above denotes that the SRS actually specifies more linked communications than the reference model.)
Interpretation	In the paper, the authors use the metrics to compare two approaches, and therefore, two SRS. Therefore, prescriptive reference values might be used for single SRS obtained by building a knowledge base of previous SRS and their completeness.
Threats to Validity	• No matching between the functions of the reference model and the SRS is performed; Therefore both, the SRS containing incorrect requirements or strongly fragmented SRS (see below) might suggest higher linked communications completeness. This hinders the use of strict thresholds.

B.53. Local completeness: Understood Requirements [Davis et al., 1993a]

Assessed (Measured) Attribute	Feature Complete (Understanding of Documented Requirements: The extent to which the documented requirements are indeed understood. The authors define understanding as the degree to which the documented requirements are "known and captured" and neither "poorly specified, abstractly stated, or not yet validated".)
Data Collection	All requirements specified in the SRS are partitioned into whether they are understood or not (by means of a manual review, although not specified in the publication), where the cardinality of the understood/not understood set is denoted n_A resp. n_B. The mathematical object is then obtained by $n_A/(n_A + n_B)$
Scale	Rational number (Ratio scale, Percentage of requirements understood compared to all requirements. Granularity is limited in the number of specified requirements.)
Interpretation	The values are interpreted in a range from 0 (totally incomplete) to 1 (complete). However, no reference value is given, and therefore, results between (0;1) are hard to interpret.
Recommendations for Action	Further investigate requirements which are not understood
Threats to Validity	• The metric definition is underspecified in terms of what is considered "understood", and is therefore highly subjective and non-repeatable. • A reference value other than 1 (which might not be feasible in practical situations) is not provided and might be hard to find and justify.

B.54. Local Feature Completeness (LFC) [Davis et al., 1993a]

Assessed (Measured) Attribute	Feature Complete
Data Collection	All requirements specified in the SRS are counted and denoted n_{AB}. Furthermore, a quality manager determines (not specified how) the number of requirements that "are needed but not (yet) specified" (n_C) and "potential requirements that are not understood well enough to be documented (n_D). The mathematical object is then obtained by calculating $n_{AB}/(n_{AB} + n_C + n_D)$.
Scale	Rational number (Ratio scale, Percentage of documented requirements compared to all identified requirements, independent whether they have been understood completely or not.)
Interpretation	The values are interpreted in a range from 0 (totally incomplete) to 1 (complete). However, no reference value is given, and therefore, results between (0;1) are hard to interpret.
Recommendations for Action	Document undocumented requirements, and if needed, investigate further if requirements are not understood to be completely specified.
Threats to Validity	• The metric definition is underspecified in what is considered "understood", and how to determine the number of potential and not-specified but known requirements. Therefore, it is highly subjective and non-repeatable. • A reference value other than 1 (which might not be feasible in practical situations) is not provided and might be hard to find and justify.

B.55. Loopholes Smell [Femmer et al., 2014a]

Assessed (Measured) Attribute	Unambiguous (Explicitly of the expectations on the requirements fulfillment: The extent to which the level of expectation associated with the requirements is explicitly and unambiguously captured in the requirements document)
Data Collection	In a first step, a natural language specification is parsed into single requirements. Afterwards, all sentences are analyzed for occurrences of certain phrases such as "if possible, as appropriate, as applicable". According to the tool presented in the paper, this process is automated and uses a dictionary (which may be altered over time) with a finite number of phrases. The number of loophole phrases corresponds to the number of dictionary matches.
Scale	Natural number (Absolute scale, Extent to which loophole words occur in the SRS.)
Interpretation	According to the authors, any occurrence of a loophole phrase is reported to the user, who ultimately judges whether the finding is indeed an error or not. Since the authors do not explicitly specify a scale, neither a prescriptive reference value or an interpretation procedure depending on the number of occurrences is given.
Recommendations for Action	Resolve identified terms, potentially with stakeholders
Threats to Validity	• Dictionary is incomplete, algorithm may not detect certain occurrences (e.g., because of typos)

B.56. Machine-Learned Nocuous Anaphoric Ambiguity Count [Yang et al., 2011]

Assessed (Measured) Attribute	Unambiguous (Nocuous Anaphoric Ambiguity: Extent to which the pronouns in a SRS are interpreted with respect to the same object (antecedent) by different readers)
Data Collection	In a first step, "the input requirements document is split into separate sentences and annotated with the individual words' part of speech." Afterwards, pronouns are identified together with a set of possible antecedents, which also includes noun-phrase co-reference resolution, i.e., that multiple noun phrases refer to the same object are treated as a unit. In a third step, the antecedents are classified based on human judgments of the most likely noun-phrase antecedent candidate and linguistic as well as statistical heuristics. Finally, each pronoun and its candidate antecedents are classified according to whether the anaphoric ambiguity is nocuous or innocuous, based on the previous classification (e.g., whenever the classification yields that one antecedent candidate is referred to significantly (based on a threshold tau of (0.5;1.0]) often, the anaphoric ambiguity is innocuous). All steps are implemented in a NLP tool, based on linguistic and statistic heuristics and machine-learning with human judgments as input. The mathematical object returned is the number of nocuous anaphoric ambiguities contained in the SRS.
Scale	Natural number (Absolute scale, The extent to which a SRS contains nocuous anaphoric ambiguities, on a scale of [0;INF+[. A value of 0 or N denotes that none respectively N nocuous anaphoric ambiguities are contained in the SRS.)
Interpretation	Authors to not specify an interpretation procedure other than to inspect every nocuous anaphoric ambiguity. However, they envision this task to be embodied in authoring tools, hence, not direct applicable to analytic SRS QA. No discussion on "bearable" anaphoric ambiguity is given, and hence, the presence of meaningful prescriptive reference values other than 0 must be investigated empirically.
Recommendations for Action	Anaphoric ambiguities are detected with a high precision (empirical results claimed by authors: 76pct)
Threats to Validity	• The WordNet database, as an socio-technical system, might fail or provide wrong results; Furthermore, typos might flaw results such that lookups do not work. • The validity of the interpretation depends on the selection and comprehensiveness of the human judgments used as input to the machine learning algorithm; Therefore, too small samples or judges not representing the actual readers of the SRS can hinder interpretation validity. • A broader generalizability wides the potential audience of a SRS, and demands a recalibration of the machine-learning approach, or, for a general external validity, replaces nocuous ambiguity but general ambiguity, assuming that at least one person associates a different meaning to a pronoun than all other persons.

B.57. Minimal Inconsistent Subset [Mu et al., 2005]

Assessed (Measured) Attribute	Semantically Consistent
Data Collection	The requirements in the SRS are assumed to be specified as a set of logical propositions, or have to be formalized. During data collection, all subsets (denoted INC) which include inconsistencies have to be detected, i.e., for which a contradiction (i.e., FALSE) can be derived. Although not specified in the paper, this step could be semi-automated using a theorem prover with automatic simplifier/prover (such as the Karlsruhe Interactive Verifier (KIV)). Finally, the number of minimal sets of INC are counted, i.e., all sets which are strict supersets of other members of INC(Delta) are removed and the resulting set's cardinality is counted.
Scale	Natural number (Absolute scale, The scale denotes the extent to which the SRS contains requirements which contradict other requirements. Specifically, the number of unsatisfiable (contradicting) cores with respect to all requirements. A value of 0 denotes no conflicting requirements, while a value equal to $2^{\text{number of requirements}}$ (= the cardinality of the powerset of all requirements) denotes maximal inconsistency (each possible set of statements has an "unique" conflict with any other possible set of statements).)
Interpretation	The authors do not provide an explicit interpretation procedure, due to the original use case being a comparison between two documents to measure improvement.
Recommendations for Action	Since the unsatisfiable cores are extracted, the inconsistency is made pretty clear and could be resolved with help from the stakeholders if needed.

B.58. Minimality of Vocabulary (MoV) [Duran et al., 2002]

Assessed (Measured) Attribute	Unambiguous (Minimality of Vocabulary: The extent to which glossary terms are used in the specification of requirements.)
Data Collection	Both the number of requirements specified and the number of times a term defined in the glossary is used are counted. Requirements are specified in XML based on a reference meta-model called REM, which allows the counting procedure to be automated by a tool (here, based on XSLT).
Scale	Rational number (Ratio scale, The ratio between the "number of references to glossary items in requirements and the number of requirements". A value of 0 denotes no term in the requirement desciription is defined in the glossary, and a value equivalent to the average number of words per requirement means that every word is actually defined in the glossary.)
Interpretation	The obtained MoV measurement is compared against prescriptive reference values per requirement. The requirements identified this way "should be checked for potential problems of ambiguity or understandability".
Reference Value	The authors state that requirements with "just a few (less than 4, for example)" glossary terms should be checked.
Threats to Validity	• Since the identification of glossary terms is based on a specific syntactic construct, the requirements have to be marked manually, which in turn is prone to human errors (omissions, incorrect associations with glossary terms) • The mean value is suspect to outliers, and the interpretation can be flawed (positively and negatively). Even for individual requirements, the relationship between the measured attribute and the qualities are neither well-understood nor empirically investigated. • The reference values are not empirically obtained but more or less a guess from the authors ("for example"). Different contexts would probably require different reference values.

B.59. Missing Conditions in Sentences (MCdS) [Kenett, 1996]

Assessed (Measured) Attribute	Information Complete (Necessary Condition Completeness: The extent to which conditions are specified wherever necessary in the SRS)
Data Collection	For a NL-SRS, the number of sentences are counted. Furthermore, for each sentence, a (presumably manual) review determines whether a condition is not specified but required. Conditions (for actions) are "the prerequisite states, activities, and/or data which are necessary for the action to occur. Examples include frequency of invocation, necessary completion of some preliminary processing sequence, and necessary processing state before entry into some operation."
Scale	Rational number (Ratio scale, The percentage of sentences which are missing conditions)
Interpretation	Values between 0 and 1 are obtainable, where 0 denotes no conditions are missing, and 1 that no conditions are specified but required for each sentence in the SRS. The values are to be interpreted against prescriptive reference values (thresholds), based on practical experiences but undisclosed in the publication.
Recommendations for Action	Missing conditions are identified as by-product of the review process.
Threats to Validity	• The number of missing conditions might be wrong since it is a quite difficult task. In particular, given the bounded knowledge of stakeholders on the actual requirements, the number of missing conditions might be indeed higher compared to the result of the review process. • The reference values seem to depend on the project, since the knowledge and boundedness of the problem space can significantly influence the number of missing attributes.

B.60. Missing Constraints in Sentences (MCsS) [Kenett, 1996]

Assessed (Measured) Attribute	Information Complete (Necessary Constraint Completeness: The extent to which constraints are specified wherever necessary in the SRS)
Data Collection	For a NL-SRS, the number of sentences are counted. Furthermore, for each sentence, a (presumably manual) review determines whether a constraint is not specified but required. Constraints are defined as "the boundary conditions enforced on the action after initiation. constraints bound influence, define termination criteria, and specify limits. Examples of constraints include numerical tolerances and time durations.
Scale	Rational number (Ratio scale, The percentage of sentences which are missing constraints.)
Interpretation	Values between 0 and 1 are obtainable, where 0 denotes no constraints are missing, and 1 that no constraints are specified but required for each sentence in the SRS. The values are to be interpreted against prescriptive reference values (thresholds), based on practical experiences but undisclosed in the publication.
Recommendations for Action	Missing constraints are identified as by-product of the review process.
Threats to Validity	• The number of missing constraints might be wrong since it is a quite difficult task. In particular, given the bounded knowledge of stakeholders on the actual requirements, the number of missing constraints might be indeed higher compared to the result of the review process. • The reference values seem to depend on the project, since the knowledge and boundedness of the problem space can significantly influence the number of missing attributes.

B.61. Missing Sentence Parts (MSP) [Kenett, 1996]

Assessed (Measured) Attribute	Information Complete
Data Collection	For a NL-SRS, each sentence is (presumably manually, but not specified in the publication) decomposed into attributes, e.g., the initiator of an action, the action, the object, etc. (9 types of attributes are given). Furthermore, each sentence is (again, presumably manually) reviewed to determine missing attributes (e.g., lack of object, conditions).
Scale	Rational number (Ratio scale, Ratio between the cardinality of two sets: the count of all missing attributes, and the count of all attributes of the SRS)
Interpretation	Values between 0 and +INF are obtainable in theory, with 0 being the interpreted as the best result (no missing attributes), and 1 meaning: for every attribute specified, there is also one attribute missing. The authors provide example thresholds (=reference values) to compare the results to for interpretation.
Reference Value	A value of <= 1/200 is considered excellent (above required performance level), <=5/200 as fair (complies with required perf. level), <=10/200 as high, and >10/200 as very high (contains some resp. Major deficiencies in performance level).
Recommendations for Action	Since the counting of missing attributes is done manually, the attributes are known and can be further investigated.
Threats to Validity	• The number of missing arguments might be wrong since it is a quite difficult task. In particular, given the bounded knowledge of stakeholders on the actual requirements, the number of missing attributes might be indeed higher compared to the result of the review process. • The reference values seem to depend on the project, since the knowledge and boundedness of the problem space can significantly influence the number of missing attributes.

B.62. Multiple Sentences (MS) [Fabbrini et al., 2000]

Assessed (Measured) Attribute	Organized (Number of compound sentences (>1 subject or verb): The extent to which a SRS contains sentences with more than one subject or verb)
Data Collection	Count the number of sentences with more than one subject or main verb, or with more than one direct or indirect complement that specified its subject. Although not described in detail, this step seems to be automated with the QUARS tool. In a second step, the tool results are validated manually by means of a manual inspection.
Scale	Natural number (Absolute scale, Number of multiple sentences)
Interpretation	The obtained number of sentences containing options is compared to restrictive prescriptive reference values (called "quality requirements" for the indicators presented in the paper).
Reference Value	The reference value is 0, i.e., no multiple sentences are allowed.
Recommendations for Action	Rephrase multiple sentences.
Threats to Validity	• Due to NL complexity, false positives and negatives are possible. While the former are addressed by a manual inspection, missed multiple sentences are still possible

B.63. Negative Statements [Femmer et al., 2014a]

Assessed (Measured) Attribute	Semantically Complete (Specification of Unintended Characteristics and Features: Extent to which negative statements, i.e., "statements of system capabilities not to be provided", are contained within the requirements specification)
Data Collection	In a first step, a natural language specification is parsed into single requirements. Afterwards, all sentences are analyzed for occurrences of certain negative phrases. According to the tool presented in the paper, this process is automated and uses a dictionary (based on the ISO 29148:2011 standard and which may be altered over time, but is not specified in the paper) with a finite number of phrases. The number of loophole phrases corresponds to the number of dictionary matches.
Scale	Natural number (Absolute scale, Extent to which negative statements occur in the SRS.)
Interpretation	According to the authors, any occurrence of a negative phrase is reported to the user, who ultimately judges whether the finding is indeed an error or not. Since the authors do not explicitly specify a scale, neither a prescriptive reference value or an interpretation procedure depending on the number of occurrences is given.
Recommendations for Action	Resolve identified terms, potentially with stakeholders
Threats to Validity	• Dictionary is incomplete, algorithm may not detect certain occurrences (e.g., because of typos)

B.64. Number of (Gapped) Clones in Use-Cases [Rago et al., 2014]

Assessed (Measured) Attribute	Concise (Use-Case Redundancy: Extent to which functionality described in use-cases is duplicated in the SRS)
Data Collection	The authors assume the presence of functional requirements described as use-cases, with steps expressed in natural language. During data collection, the textual use cases are initially divided into sentences, which are further divided into tokens. Each token is tagged by its part-of-speech, stemmed/lemmatized and analyzed for statistical significance (stop-word detection). Afterwards, the meaning of a sentence and its constituents is determined by a NLP module called "Semantic Role Labeling" (SRL), which recognizes the main predicate of a sentence and its attributes. Next, the identified main predicate and its attributes are classified into "domain actions", which are pre-defined 25 classes defined in the paper, such as "Input: Selection", "Output: Display", and "Write Data: Update"), using a machine-learning approach. Afterwards, gapped clones of such sequences of domain actions are identified among different use-cases, by a technique called "Sequence Alignment" which is described in more detail in the paper. Essentially, the approach identifies sequences which occur also as (sub-)sequences in other use cases and which contain only a limited number of additional or different actions compared to the length of the duplicated sequence in total, where in addition to being classified as the same domain action, the similarity of the sentences are also considered. Penalties for gaps are applied (larger penalty for gap occurrence at all, lesser penalty for the size of the gap), and the similarity thresholds also requires a sequence to contain at least 4 actions to be considered a clone. Finally, a number of gapped clones above this threshold are identified and counted.
Scale	Natural number (Absolute scale, The extent to which functionality is duplicated in textual use cases, on a scale of [0;INF+).)
Interpretation	The authors use the measurement as a constructive means during requirements specification, but do not specify an interpretation procedure. Implicitly, the metric is used to compare it against different versions of the same document to assess the improvement regarding redundancy.
Recommendations for Action	Part of the approach is a recommendation on how to fix (i.e., using which use-case modeling construct) the identified (gapped) clones, which was evaluated as spot-on in the paper
Threats to Validity	• The metric ignores smaller chunks of clones (e.g., less than 4 domain actions), leading to smaller number of clones found. Also, the simple count of clones does not consider the extent of cloning regarding more precise measures of size of the use-case (e.g., cloning blow up). Furthermore, redundancy is only one aspect of conciseness, so its absence does not guarantee that a SRS is as concise as possible. • Authors note that the approach works best for use-cases with lexically consistent use-cases; Domains or projects for which this property does not hold might lead to significantly less clones identified.

B.65. Number of Acronyms per Requirement [Génova et al., 2013]

Assessed (Measured) Attribute	Pragmatic Quality (Use of Acronyms: The extent to which the requirements specification contains acronyms)
Data Collection	The authors assume a NL-SRS in which individual requirements could be explicitly identified. During data collection, the occurrences of "words formed entirely or mostly with capital letters" are detected per requirement (however, no more detail specified here). Afterwards, the number of found acronyms is mapped to a numerical value 0,1,2 using a scoring function and two thresholds according to the definition of the Dec function. Finally, the arithmetic mean of all requirements' scores is calculated. This is implemented in specific tool proposed by the authors, and hence, automated.
Scale	Rational number (Ratio scale, The extent to which requirements contain no excessive amount of acronyms, on the scale of [0;2].)
Interpretation	According to the authors, individual requirements and a global interpretation are given. For each requirement individually, a decreasing interpretation function should applied be applied, "since the intention is to avoid an abuse". Each requirement is interpreted on three levels labeled "good", "medium" and "bad". The implication in case of "bad" is that the requirement "must" be changed (mandatory), while for "medium" the requirement "should" be changed (optional), replacing the negative terms by semantically equivalent grammatical constructions using positive descriptions since it "increases the risk of logical inconsistencies". Furthermore, a score for all analyzed requirements is obtained on a scale of [0;2] with the same interpretation as above applied but for the set of all requirements is indicated. Also, authors suggest to use more than one metric per quality attribute on an individual requirement basis first, and join those results for the set of requirements in a SRS.
Reference Value	Overall measure: [0, 0.5) is interpreted as "Bad", [0.5; 1.5) as Medium, and [1.5; 2] as "Good"; "Medium" and "Bad" suggest that requirements should respectively must be changed. In this case, the interpretation on the level of individual requirements is considered.
Recommendations for Action	Individual terms are identified which are candidates to be fixed
Threats to Validity	• Although the acronym detection procedure is vague it "maybe NOT [easy] in a fully deterministic way" according to the authors. • Acronyms are neither sufficient nor necessary conditions regarding pragmatic quality, and we consider this indicator to be of rather limited impact and hence, weak. Therefore, interpretation has to be careful in the extent the measurement value describes pragmatic quality at all, or even individual defects. Furthermore, the use of average as an aggregation function on scores can obscure a harmfully high number of acronyms of a few requirements. • We assume the effect of acronyms on pragmatic quality relies on a large degree on the domain experience of the SRS audience, since an experienced reader probably knows many or most acronyms, the domain itself (e.g., the extent to which acronyms are used can depend on the domain), the language and other project circumstances (e.g., whether or not the project partner is part of a prior collaboration)

B.66. Number of Anaphoric Terms per Requirement [Génova et al., 2013]

Assessed (Measured) Attribute	Unambiguous (Use of Anaphoric Terms: The extent to which the requirements contain anaphoric terms)
Data Collection	The authors assume a NL-SRS in which individual requirements could be explicitly identified. During data collection, the occurrences of words of a given dictionary are counted per requirement. Afterwards, the number of found terms is mapped to a numberical value 0,1,2 using a scoring function and two thresholds according to the definition of the Dec function. Finally, the arithmetic mean of all requirements' scores is calculated. This is implemented in specific tool proposed by the authors, and hence, the procedure is automated. No additional information on NLP techniques such as stemming are given, so we assume those are not applied.
Scale	Rational number (Ratio scale, The extent to which requirements contain no excessive amount of anaphoric terms on the scale of [0;2].)
Interpretation	According to the authors, individual requirements and a global interpretation are given. For each requirement individually, a decreasing interpretation function should applied be applied, "since the intention is to avoid an abuse". Each requirement is interpreted on three levels labeled "good", "medium" and "bad". The implication in case of "bad" is that the requirement "must" be changed (mandatory), while for "medium" the requirement "should" be changed (optional), resolving anaphors since it "increases the risk of imprecision and ambiguities". Furthermore, a score for all analyzed requirements is obtained on a scale of [0;2] with the same interpretation as above applied but for the set of all requirements is indicated.
Reference Value	Overall measure: [0, 0.5) is interpreted as "Bad", [0.5; 1.5) as Medium, and [1.5; 2] as "Good"; "Medium" and "Bad" suggest that requirements should respectively must be changed. In this case, the interpretation on the level of individual requirements is considered.
Recommendations for Action	Individual terms are identified which are candidates to be fixed
Threats to Validity	• The metric is based on a fixed dictionary, which in general and under practical conditions is at least incomplete and potentially incorrect. No additional NLP techniques are used, such as stemming, and typos would result in negative terms not being counted. All those threats can often result in a smaller measurement value in practice compared to the actual number of anaphoric terms, while false positives are still possible. • Anaphoric terms itself are neither sufficient nor necessary conditions regarding ambiguity, and we consider this indicator to be of rather limited impact because we expect to be anaphoric terms to be used frequently independent on the ambiguity of the SRS. Furthermore, the use of average as an aggregation function on scores can obscure a high number of anaphoric terms which are limited to few requirements. • The use of anaphoric terms depends on the writing style of the requirements engineer. Hence, at least prescriptive reference values have to be adapted.

B.67. Number of Clones [Juergens et al., 2010]

Assessed (Measured) Attribute	Concise (Text Redundancy: Extent to which text passages are duplicated in the SRS)
Data Collection	In a first step, all NL text from all physical documents comprising the SRS is extracted. Second, stop-words are removed and word stemming (according to the Porter stemming algorithm) is applied. Afterwards, a token-based clone detection algorithm is applied (for details, see B.S. Baker "On finding duplication and near-duplication in large software systems", WCRE'1995) with a minimal clone-length of 20 words (tokens). This algorithm yields a number of clones and clone groups, and is automated using the ConQAT tool according to the authors. Ina last and manual step, filters are iteratively applied to improve precision of the clone analysis. Filters are rules which ignore certain clones, e.g., because of their occurrence in document meta data, indexes, page decorations, of specification template information. This step is iterative, for which the authors result suggest that between 1 and 30 iterations are needed.
Scale	Natural number (Absolute scale, The extent to a SRS contains clones, i.e., duplication of text passages with 20 or more words. A value of 0 denotes that a SRS does not contain clones, while any positive integer is the count of clones contained.)
Interpretation	Authors do not specify an interpretation procedure as part of their paper; Empirical results reveal that the number of clones ranges from 0 (spec size: 40 pages, 8000 words) to 1818 (spec size: 3100 pages, 280k words); Therefore, we do not see a reliable prescriptive reference value that is applicable to practical situations.
Recommendations for Action	Clone pairs are identified; Can be used to include cross references or create glossary entries instead.
Threats to Validity	• The tailoring depends on the understanding and expertise of the person measuring in order to not filter out non-benign clones. • Depending on the SRS itself, a word-level granularity might be inappropriate. • The use of templates or generated documentation (e.g., from models) can harm external validity; The iterative filtering might not fully mitigate those effects.

B.68. Number of Clusters for Technical Terms [Matsuoka and Lepage, 2011]

Assessed (Measured) Attribute	Lexically Consistent (Interpretation consistency: Extent to which the interpretations (associated meanings) of occurrences of technical terms are equal within the SRS)
Data Collection	In a first step, "those words that are relevant for the requiremetns at hand, i.e., technical terms" have to be identified. Therefore, for a set of candidate terms (not specified in the paper, we assume all terms except stop words) a c-value measure (higher number= term is composed of more words and occurs frequently) is computed, and the terms with the highest value are extracted. For any such term, all N meanings are retrieved using the WordNet semantic database. In case N>1 (multiple meanings), the words similarity to other words in the sentence is investigated (using a similarity measure based on WordNets hierarchical structure) and each occurrence of the word is classified in a cluster according to the most significant meaning. In case the similarity measure is equal for more than one occurrence of the term, the occurrences are classified in a cluster associated with multiple meanings. Afterwards, Inter-sentence similarity is used to move sentences/occurrences of multiple-meaning clusters into cluster with only one meaning, according to a sentence similarity measure with a prescriptive threshold. Finally, the number of clusters obtained this way are counted.
Scale	Natural number (Absolute scale, The number of meanings associated with a technical term within a SRS, not necessarily individually ambiguous, on a scale from [1;INF+]. A value of 1 denotes that all occurrences of the technical word are consistently associated with the same (but potentially ambiguous) meaning in every sentence of the SRS, while a value of n denotes that the term is associated with n different (but potentially overlapping) interpretations within the SRS.)
Interpretation	No interpretations procedure is explicitly stated; In the paper, authors distinguish a value of 1 to not cause any action, while any value >1 is interpreted as lexically inconsistent and potentially ambiguous and should be inspected.
Reference Value	1
Recommendations for Action	Candidate terms are identified; Because of the limitations to the most important technical terms only, this approach seems to be applicable in practical situations
Threats to Validity	• The WordNet database, as an socio-technical system, might fail or provide wrong results; Furthermore, typos might flaw results such that lookups do not work. • Due to the complexity of language, the WordNet database is incomplete, and hence some additional interpretations might be missing. Furthermore, the hierarchical structuring of words as the only measure of similarity reduces relationships between objects to only one aspect, which might lead to incorrectly computed word similarities. • A source for disambiguation not considered in the paper is that of the background/domain knowledge of the person, which might allow to rule out certain ambiguities. Furthermore, additional interpretation assistance might be provided in terms of process or artifact knowledge, e.g., the use of templates and the interpretation of content because of certain artifact types.

B.69. Number of Control Flow Terms per Requirement [Génova et al., 2013]

Assessed (Measured) Attribute	Design-Independent (Use of Control Flow Terms: The extent to which the requirements specification contains control flow terms)
Data Collection	The authors assume a NL-SRS in which individual requirements could be explicitly identified. During data collection, the occurrences of words of a given dictionary (e.g., "while, when, if...then") are counted per requirement. Afterwards, the number of found terms is mapped to a numerical value 0,1,2 using a scoring function and two thresholds according to the definition of the Dec function. Finally, the arithmetic mean of all requirements' scores is calculated. This is implemented in specific tool proposed by the authors, and hence, the procedure is automated. No additional information on NLP techniques such as stemming are given, so we assume those are not applied.
Scale	Rational number (Ratio scale, The extent to which requirements contain no excessive amount of control flow terms on the scale of [0;2].)
Interpretation	According to the authors, individual requirements and a global interpretation are given. For each requirement individually, a decreasing interpretation function should applied be applied, "since the intention is to avoid an abuse". Each requirement is interpreted on three levels labeled "good", "medium" and "bad". The implication in case of "bad" is that the requirement "must" be changed (mandatory), while for "medium" the requirement "should" be changed (optional), removing the part introduced by the control flow construct since it "probably points to an excess of detail, [...] going beyond the limits of a must be abstract requirements specification". Furthermore, a score for all analyzed requirements is obtained on a scale of [0;2] with the same interpretation as above applied but for the set of all requirements is indicated. Also, authors suggest to use more than one metric per quality attribute on an individual requirement basis first, and join those results for the set of requirements in a SRS.
Recommendations for Action	Individual terms are identified which are candidates to be fixed
Threats to Validity	• The metric is based on a fixed dictionary, which in general and under practical conditions is at least incomplete and potentially incorrect. No additional NLP techniques are used, such as stemming, and typos would result in negative terms not being counted. All those threats can often result in a smaller measurement value in practice compared to the actual number of control flow terms, while false positives are still possible. • Control Flow terms are neither sufficient nor necessary conditions regarding design independence, and we consider this indicator to be of rather limited impact because we expect to be control flow terms to be present for design independent specifications often. Furthermore, the use of average as an aggregation function on scores can obscure a high number of control flow terms of limited to few requirements. • The use of control flow terms depends on the writing style of the requirements engineer. Hence, at least prescriptive reference values have to be adapted.

B.70. Number of Design Terms per Requirement [Génova et al., 2013]

Assessed (Measured) Attribute	Design-Independent (Use of Design Terms: The extent to which design terms are used in the requirements specification)
Data Collection	The authors assume a NL-SRS in which individual requirements could be explicitly identified. During data collection, the occurrences of words of a given dictionary for design terms (examples given are "method, parameter, database, applet") are counted per requirement. Afterwards, the number of found terms is mapped to a numerical value 0,1,2 using a scoring function and two thresholds according to the definition of the Dec function. Finally, the arithmetic mean of all requirements' scores is calculated. This is implemented in specific tool proposed by the authors, and hence, the procedure is automated. No additional information on NLP techniques such as stemming are given, so we assume those are not applied.
Scale	Rational number (Ratio scale, The extent to which requirements contain no excessive amount of design terms on the scale of [0;2].)
Interpretation	According to the authors, individual requirements and a global interpretation are given. For each requirement individually, a decreasing interpretation function should applied be applied, "since the intention is to avoid an abuse". Each requirement is interpreted on three levels labeled "good", "medium" and "bad". The implication in case of "bad" is that the requirement "must" be changed (mandatory), while for "medium" the requirement "should" be changed (optional), removing design decisions indicated by design terms. Furthermore, a score for all analyzed requirements is obtained on a scale of [0;2] with the same interpretation as above applied but for the set of all requirements is indicated. Also, authors suggest to use more than one metric per quality attribute on an individual requirement basis first, and join those results for the set of requirements in a SRS.
Reference Value	Overall measure: [0, 0.5) is interpreted as "Bad", [0.5; 1.5) as Medium, and [1.5; 2] as "Good"; "Medium" and "Bad" suggest that requirements should respectively must be changed. In this case, the interpretation on the level of individual requirements is considered.
Recommendations for Action	Individual terms are identified which are candidates to be fixed
Threats to Validity	• The metric is based on a fixed dictionary, which in general and under practical conditions is at least incomplete and potentially incorrect. No additional NLP techniques are used, such as stemming, and typos would result in negative terms not being counted. All those threats can often result in a smaller measurement value in practice compared to the actual number of design terms, while false positives are still possible. • Design terms as a metric depend highly on the quality of the dictionary, and the tradeoff between precision and recall can be tuned using different dictionaries. Furthermore, the use of average as an aggregation function on scores can obscure a high number of anaphoric terms which are limited to few requirements.

B.71. Number of Domain Terms per Requirement [Génova et al., 2013]

Measured Attribute	Singular (Atomic) (Use of Domain Terms: The extent to which domain terms are used in the requirements specification)
Data Collection	The authors assume a NL-SRS in which individual requirements could be explicitly identified. During data collection, all terms of the requirements are normalized, i.e., a canonical form is obtained by "verbal conjugation, singular/plural for nouns, gender for nouns in languages such as Spanish"Furthermore, "tools are also required to detect meaningful compound terms. Once nouns and verbs have been normalized, they are compared with the terms defined in the domain, which can be organized as a simple glossary of terms, thesauri or ontologies" per requirement. Afterwards, the number of found terms is mapped to a numerical value 0,1,2 using a scoring function and two thresholds according to the definition of the Dec function. Finally, the arithmetic mean of all requirements' scores is calculated. Afterwards, the number of found terms is mapped to a numerical value 0,1,2 using a scoring function and two thresholds according to the definition of the Dec function. Finally, the arithmetic mean of all requirements' scores is calculated. This is implemented in specific tool proposed by the authors, and hence, the procedure is automated. No additional information on NLP techniques such as stemming are given, so we assume those are not applied.
Scale	Rational number (Ratio scale, The extent to which requirements contain no excessive amount of domain terms on the scale of [0;2].)
Interpretation	According to the authors, individual requirements and a global interpretation are given. For each requirement individually, a convex (Increasing/Decreasing) interpretation function should applied be applied since "the number of domain terms [...] should be neither too high ([...] loss of atomicity) nor to low ([...]imprecise requirements or [....] not sufficiently cover the terms employed in the requirements)". Each requirement is interpreted on three levels labeled "good", "medium" and "bad". The implication in case of "bad" is that the requirement "must" be changed (mandatory), while for "medium" the requirement "should" be changed (optional). Depending on whether the number is too high or too low, the requirements should be split respectively the precision should be improved by using domain vocabulary. Furthermore, a score for all analyzed requirements is obtained on a scale of [0;2] with the same interpretation as above applied but for the set of all requirements is indicated. Also, authors suggest to use more than one metric per quality attribute on an individual requirement basis first, and join those results for the set of requirements in a SRS.
Recommend. for Action	The lack or excess of domain terms is identified for individual requirements to be inspected.
Threats to Validity	• The metric is based on a fixed dictionary, which in general and under practical conditions is at least incomplete and potentially incorrect. This often results in a smaller measurement value compared to the actual number of domain terms for practical situations, but can also include false positives. • Domain terms as a metric depend highly on the quality of the dictionary, and the tradeoff between precision and recall can be tuned using different dictionaries. However, whether or not a term is considered a domain term depends on the context of the sentence (e.g., some general terms have specific domain interpretations). Furthermore, the use of average as an aggregation function on scores can obscure a high number of anaphoric terms which are limited to few requirements. • Both, the terms and their frequency of use depends on the domain. Furthermore, the frequency of use of domain terms depends on the writing style, the language, and the relationship to the SRS audience.

B.72. Number of Leaf Goals of Agents (ANLG) [Espada et al., 2011, 2013]

Assessed (Measured) Attribute	Organized (Agent Responsibility Decomposition: The extent to which the responsibility for goal objects allocated to individual agents is decomposed.)
Data Collection	Authors assume that goals are specified according to the KAOS methodology. During data collection, the number of connected direct or indirect leaf goals are identified and counted, per and for all agents. Data collection was performed by hand in [espada2011measuring], while in [espada2013framework], a prototypical tool called modularKAOS was used. If there are no leaf goals, the metric is undefined.
Scale	Natural number (Absolute scale, The number of goals to which agents are associated to, on a scale of [0;N(LeafGoals)], where N(LeafGoals) is the number of leaf goals in a goal model.)
Interpretation	Authors propose to evaluate the (vector of) goal counts by comparing each agent's goal count against a single prescriptive reference value. This value shall be established by "sufficiently large a body of examples [considered as good]".
Recommendations for Action	Individual agents are identified. Therefore, agents must be investigated for "too much responsibility", i.e., that the agents is "too general, so more specialized agents should be considered."
Threats to Validity	• The metric measures complexity of a model, while for the extent a SRS is organized only the accidential complexity is relevant. Therefore, the intrinsic complexity must be judged by the interpreter, or it must be reflected in the training set regarding the reference value. • The proposed metric measures complexity of a goal-model, which inherently depends on the size of the system-to-be and the goal modeling process itself. Therefore, the metric's result must be carefully interpreted with respect to the size and modeling process (including the expertise of the modeler, both regarding KAOS and the system's domain). In particular, the prescriptive reference values might need to be adapted.

B.73. Number of Negative Terms per Requirement [Génova et al., 2013]

Assessed (Measured) Attribute	Pragmatic Quality (Use of Negative Statements: The extent to which the requirement contains negative statements)
Data Collection	The authors assume a NL-SRS in which individual requirements could be explicitly identified. During data collection, the occurrences of words of a given dictionary (e.g., "not, no, neither, never, nothing, nowhere") are counted per requirement. Afterwards, the number of found terms is mapped to a numerical value 0,1,2 using a scoring function and two thresholds according to the definition of the Dec function. Finally, the arithmetic mean of all requirements' scores is calculated. This is implemented in specific tool proposed by the authors, and hence, automated No additional information on NLP techniques such as stemming are given, so we assume those are not applied.
Scale	Rational number (Ratio scale, The extent to which requirements contain no excessive amount of negative terms on the scale of [0;2].)
Interpretation	According to the authors, individual requirements and a globel interpretation are given. For each requirement individually, a decreasing interpretation function should applied be applied, "since the intention is to avoid an abuse". Each requirement is interpreted on three levels labeled "good", "medium" and "bad". The implication in case of "bad" is that the requirement "must" be changed (mandatory), while for "medium" the requirement "should" be changed (optional), replacing the negative terms by semantically equivalent grammatical constructions using positive descriptions since it "increases the risk of logical inconsistencies". Furthermore, a score for all analysed requirements is obtained on a scale of [0;2] with the same interpretation as above applied but for the set of all requirements is indicated. Also, authors suggest to use more than one metric per quality attribute on an individual requirement basis first, and join those results for the set of requirements in a SRS.
Reference Value	Overall measure: [0, 0.5) is interpreted as "Bad", [0.5; 1.5) as Medium, and [1.5; 2] as "Good"; "Medium" and "Bad" suggest that requirements should respectively must be changed. In this case, the interpretation on the level of individual requirements is considered.
Recommendations for Action	Individual terms are identified which are candidates to be fixed
Threats to Validity	• The metric is based on a fixed dictionary, which in general and under practical conditions is at least incomplete and potentially incorrect. No additional NLP techniques are used, such as stemming, and typos would result in negative terms not being counted. All those threats result in a potentially smaller measurement value compared to the actual number of negative terms. • Negative statements are neither sufficient nor necessary conditions regarding pragmatic quality, and we consider this indicator to be of rather limited impact given recent empirical evidence (Mund et al., "Does Requirement Specification Quality matter?"). Therefore, interpretation has to be careful in the extent the measurement value describes pragmatic quality at all, or even individual defects. Furthermore, the use of average as an aggregation function on scores can obscure a harmfully high number of acronyms of a few requirements. • The use of negative/negated statements depends on the writing style of the requirements engineer. Hence, at least prescriptive reference values have to be adapted.

B.74. Number of Objects of Leaf Goals (GNO) [Espada et al., 2011, 2013]

Assessed (Measured) Attribute	Organized (Size of Leaf Goals: The size of leaf goals measured as the amount of objects associated with it)
Data Collection	Authors assume that goals are specified according to the KAOS methodology. During data collection, the number of objects directly connected to a goal are identified and counted individually for all leaf goals. Data collection was performed by hand in [espada2011measuring], while in [espada2013framework], a prototypical tool called modularKAOS was used. If there are no leaf goals, the metric is undefined.
Scale	Natural number (Absolute scale, The number of objects which are associated with each goal, on a scale of [0;N(Objects)], where N(Objects) is the number of objects occurring in a goal model.)
Interpretation	Authors propose to evaluate the (vector of) object counts by comparing each goal's object count against a single prescriptive reference value. This value shall be established by "sufficiently large a body of examples [considered as good]". However, the paper does not allow to propose any reference values for now.
Recommendations for Action	Individual goals are identified. "Too many objects associated with a goal should be avoided. I general, this is a hint for the need of decomposing a goal into sub-goals."
Threats to Validity	• The metric measures complexity of a model, while for the extent a SRS is organized only the accidental complexity is relevant. Therefore, the intrinsic complexity must be judged by the interpreter, or it must be reflected in the training set regarding the reference value. • The proposed metric measures complexity of a goal-model, which inherently depends on the size of the system-to-be and the goal modeling process itself. Therefore, the metric's result must be carefully interpreted with respect to the size and modeling process (including the expertise of the modeler, both regarding KAOS and the system's domain). In particular, the prescriptive reference values might need to be adapted.

B.75. Number of Optional Phrases (OP) [Fabbrini et al., 2000]

Assessed (Measured) Attribute	Unambiguous (Explicitly of the expectations on the requirements fulfillment: The extent to which the level of expectation associated with the requirements is explicitly and unambiguously captured in the requirements document)
Data Collection	All sentences in a NL-SRS are analyzed for occurrences of certain phrases such as "possibly, eventually, if appropriate, if needed, ...". According to the tool presented in the paper, this process is automated and hence uses a predefined list (which may be altered over time) with a fixed number of phrases. In a second step, every occurrence is found "is presented to a reviewer" who ultimately decides whether the sentence is really "incorrect".
Scale	Natural number (Absolute scale, Number of sentences which contain one or more option phrases.)
Interpretation	The obtained number of sentences containing options is compared to restrictive prescriptive reference values (called "quality requirements" for the indicators presented in the paper).
Reference Value	The reference value is 0. That is, no sentence is allowed to have unresolved options respectively none are allowed.
Recommendations for Action	Resolve options with stakeholders
Threats to Validity	• Dictionary is incomplete, algorithm may not detect certain occurrences (e.g., typos)

B.76. Number of Sentence Part Types used (NST) [Kenett, 1996]

Assessed (Measured) Attribute	Semantically Complete
Data Collection	For a NL-SRS, each sentence is (presumably manually, but not specified in the publication) decomposed into attributes, e.g., the initiator of an action, the action, the object, etc (9 types of attributes are given). The metric is obtained by counting the number of distinct attribute types used.
Scale	Natural number (Ratio scale, $[0; 9]$, since there exist nine distinct attribute types (sentence part types))
Interpretation	Values between 0 and 9 are obtainable, where more attributes used are linked to the completeness of the SRS. However, no reference values are given to interpret the metric.

B.77. Number of Steps of Use Case (NOS) [Duran et al., 2002, Bernárdez et al., 2004a, Ciemniewska et al., 2007]

Assessed (Measured) Attribute	Pragmatic Quality (Complexity of Use Cases: The extent of complexity in understanding a use-case from the point of view of the receiver.)
Data Collection	The authors assume that use-cases are specified according to a specific metamodel, allowing the identification and countability of individual use case steps and the associated action owner. Therefore, the data collection procedure is automated (e.g., using a XSLT-based implementation as in Duran et al. [2002]). In Bernárdez et al. [2004a], the metric was proposed to be refined to ignore Actor-Actor Actions within the use case for the sake of measurement.
Scale	Natural number (Absolute scale,)
Interpretation	The obtained *NOS* is compared to a prescriptive reference interval. If the *NOS* exceeds the interval, the use-case is (potentially) flawed. In case NOS is too low, the use case is considered incomplete; in case it is too high, it is considered too detailed/complex to understand.
Reference Value	According to Bernárdez et al. [2004a], the *NOS* should be in $[3, 9]$. The same reference value is specified in Ciemniewska et al. [2007], probably based on the same original argument by Adolph et al. [2002].
Recommendations for Action	If $NOS < 3$, verify that the use case is complete; If $NOS > 9$, restructure the use case using include/exclude relationships if possible
Threats to Validity	• The authors observed that the metric's applicability (with prescriptive reference values) depends on the writing style of the use case authors; It depends whether multiple actions by an actor or system which happen simultaneously or consecutively by the same actor/system are written as one step (e.g., concatenated by an "and") or as multiple steps. The metric was designed for the former style, and the latter one would require further adjustment.

B.78. Number of Steps of Use Case (NOS) [Duran et al., 2002, Bernárdez et al., 2004a, Ciemniewska et al., 2007]

Assessed (Measured) Attribute	Feature Complete (Completeness of Individual Features: The extent to which individual features are described to match the stakeholders requirements and expectations about the specific features.)
Data Collection	The authors assume that use-cases are specified according to a specific meta-model, allowing the identification and countability of individual use case steps and the associated action owner. Therefore, the data collection procedure is automated (e.g., using a XSLT-based implementation as in Duran et al. [2002]). In Bernárdez et al. [2004a], the metric was proposed to be refined to ignore Actor-Actor Actions within the use case for the sake of measurement.
Scale	Natural number (Absolute scale,)
Interpretation	The obtained *NOS* is compared to a prescriptive reference interval. If the *NOS* exceeds the interval, the use-case is (potentially) flawed. In case *NOS* is too low, the use case is considered incomplete; in case it is too high, it is considered too detailed/complex to understand.
Reference Value	According to Bernárdez et al. [2004a], the *NOS* should be in $[3, 9]$. The same reference value is specified in Ciemniewska et al. [2007], probably based on the same original argument by Adolph et al. [2002].
Recommendations for Action	If $NOS < 3$, verify that the use case is complete; If $NOS > 9$, restructure the use case using include/exclude relationships if possible
Threats to Validity	• The authors observed that the metric's applicability (with prescriptive reference values) depends on the writing style of the use case authors; It depends whether multiple actions by an actor or system which happen simultaneously or consecutively by the same actor/system are written as one step (e.g., concatenated by an "and") or as multiple steps. The metric was designed for the former style, and the latter one would require further adjustment.

B.79. Ontology-Based Completeness [Kaiya and Saeki, 2005]

Assessed (Measured) Attribute	Feature Complete (Ontological Coverage: The extent to which the requirements cover all elements from an associated ontology)
Data Collection	The authors assume a formal domain ontology and a mapping from the requirements to elements of this ontology (denoted Fint) for the system under consideration to be present. The measure counts the the number of concepts (Con) and relationships (Rel) in the ontology which are not referenced by any terms (ReqItem) of the SRS. The extraction of terms in requirements specifications is based on "lexical decomposition techniques" (reference given in paper), and authors mentioned that all steps were performed manually.
Scale	Rational number (Ratio scale, The scale is a rational number between 0 and 1, denoting the extent to which the SRS mentions elements (concepts and relationships) of the ontology. A value of 0 denotes that no requirements mentions any element of the ontology, while a value of 1 denotes that every concept and relationship of the ontology is mentioned at least in one requirement.)
Interpretation	The authors do no provide an explicit interpretation procedure. In the paper, a single value is computed, however, no preference values/thresholds are specified. According to our judgment, a value of 1 might neither be feasible nor desirable (not all domain knowledge must be mentioned in the SRS) in practice, and hence, empirical investigations are required.
Recommendations for Action	According to [kaiya2006using], "when the completeness measure is low, the analyst should find and add new items to the requirements list". This can be achieved either "by focusing on the mapped concepts typed with "function", "object", and "environment" and by then tracing "apply" and "perform relationships", or "by tracing 'is-a', 'has-a' and 'require' relationships".

B.80. Ontology-Based Consistency [Kaiya and Saeki, 2005]

Assessed (Measured) Attribute	Semantically Consistent (Ontologically Consistent: The extent to which the requirements' ontological interpretations are free from contradictions identified in the ontological basis.)
Data Collection	The authors assume a formal domain ontology (including a "contradict" relationship between ontological concepts) and a mapping from the requirements to elements of this ontology for the system under consideration to be present. The measure counts the ratio of contradiction-relationships w.r.t. all relationships in the ontology which are also mentioned as terms in the SRS. The extraction of terms in requirements specifications is based on "lexical decomposition techniques" (reference given in paper), and authors mentioned that all steps were performed manually.
Scale	Rational number (Ratio scale, The scale is a rational number between 0 and 1, denoting the normalized amount of contradictions in the SRS w.r.t. an ontological basis. A value of 1 denotes the absence of such contradictions, while a value of 0 denotes that all terms occurring in the SRS which are also captured in the ontology are contradicted by another term in the SRS.)
Interpretation	The authors do no provide an explicit interpretation procedure. In the paper, a single value is computed, however, no preference values/thresholds are specified. According to our judgment, a value of 1 does not seem appropriate (nor feasible) for practical SRS, because of contradicting goals of software systems, especially for systems with multiple (heterogeneous) stakeholders.
Recommendations for Action	According to [kaiya2006], "when the consistency measure is low, one of two inconsistent items is selected to be removed. The inconsistent items can be detected by searching contradict relationships in the mapped [ontology] elements. The priority relationship among the items can help the analyst and stakeholders to decide which item should be removed."

B.81. Ontology-Based Correctness (ONT_COR) [Kaiya and Saeki, 2005]

Assessed (Measured) Attribute	Semantically Correct (Ontological Completeness: The extent to which the requirements are based on elements from an associated ontology)
Data Collection	The authors assume a formal domain ontology and a mapping from the requirements to elements of this ontology for the system under consideration to be present. The measure counts the the number of terms in requirements specifications which are mapped to at least one concept or relationship in the associated domain ontology. The extraction of terms in requirements specifications is based on "lexical decomposition techniques" (reference given in paper), and authors mentioned that all steps were performed manually.
Scale	Rational number (Ratio scale, The scale is a rational number between 0 and 1, denoting the extent to which the terms within the SRS are based on elements from the associated domain ontology. A value of 0 or 1 denote that no resp. all requirements are based on ontological elements.)
Interpretation	The authors do no provide an explicit interpretation procedure. In the paper, a single value is computed, however, no preference values/thresholds are specified. According to our judgment, a value of 1 might not be feasible in practice, and hence, empirical investigations are required.
Recommendations for Action	According to Kaiya and Saeki [2006], "when the correctness measure is low, the analyst confuses on the mapped concepts of the incorrect items, and asks stakeholders whether the items are really necessary or not".

B.82. Ontology-Based Unambiguity [Kaiya and Saeki, 2005]

Assessed (Measured) Attribute	Unambiguous (Ontologically Partitioned: The extent to which terms are interpreted with more than one weakly connected components within the associated ontology.)
Data Collection	The authors assume a formal domain ontology and a mapping from the requirements to elements of this ontology for the system under consideration to be present. The measure counts the the number of terms in requirements specifications, for which all concepts and relationships of the associated domain ontology are semantically connected. Semantically connected means that all concepts are (transitively) related to each other with relation other than "contradiction" or "antonym". The extraction of terms in requirements specifications is based on "lexical decomposition techniques" (reference given in paper), and authors mentioned that all steps were performed manually.
Scale	Rational number (Ratio scale, The scale is a rational number between 0 and 1 denoting the extent to which the terms of a requirements specification are mapped to exclusively semantically related elements. A value of 0 (1) denotes that no (all) element are mapped to semantically related elements only.)
Interpretation	The authors do no provide an explicit interpretation procedure. In the paper, a single value is computed, however, no preference values/thresholds are specified.
Recommendations for Action	According to [kaiya2006], "when the unambiguous measure is low, the analyst should find ambiguous items and then modify, remove or split them into several items. The analyst looks for a requirements item that is mapped to more than one "island" and it is a candidate of ambiguous item. In the case of splitting it, the analyst does it into several items so that each of the items is mapped to only one "island".

B.83. Open-Ended Terms Smell [Femmer et al., 2014a]

Assessed (Measured) Attribute	Unambiguous (Use of Open-Ended Phrases: The extent to which phrases which "offer a choice of possibilities" are used within the requirements specification)
Data Collection	In a first step, a natural language specification is parsed into single requirements. Afterwards, all sentences are analyzed for occurrences of certain phrases such as "provide support, not limited to, as a minimum". According to the tool presented in the paper, this process is automated and uses a dictionary (based on the ISO 29148:2011 standard and which may be altered over time) with a finite number of phrases. The number of loophole phrases corresponds to the number of dictionary matches.
Scale	Natural number (Absolute scale, Extent to which the SRS contains open-ended terms.)
Interpretation	According to the authors, any occurrence of a open-ended term is reported to the user, who ultimately judges whether the finding is indeed an error or not. Since the authors do not explicitly specify a scale, neither a prescriptive reference value or an interpretation procedure depending on the number of occurrences is given.
Recommendations for Action	Resolve identified terms, potentially with stakeholders
Threats to Validity	• Dictionary is incomplete, algorithm may not detect certain occurrences (e.g., because of typos)

B.84. Partiality of Specified Functionality (PSF) [Davis et al., 1993a]

Assessed (Measured) Attribute	Feature Complete (Definition Completeness of Documented System Functions: The extent to which a specification defines the systems' function in terms of their definition completeness, i.e., the degree to which system functions are specified for any possible input and system state.)
Data Collection	Measures whether a system behavior is specified for all states and stimuli that can occur during the interaction with the system. Therefore, the "inputs or stimuli specified in the SRS" (n_i), states of the system (n_s) "stated in or implied y the SRS", and unique functions specified in the SRS (n_u), where unique functions account to the fact that functions may be redundantly specified in the SRS. The metric then computes a rational number as n_u / (n_i * n_s).
Scale	Rational number (Ratio scale, Percentage of functions specified in the SRS compared to the "total number of function values that must be specified", i.e., n_s * n_i. Therefore, granularity is limited to the product of specified inputs and specified or implied system states)
Interpretation	The metric is interpreted as follows: As closer as the value is to 1, the more likely the specification is "feature complete" (for state-based systems), i.e., the more likely all required system functionality is specified. In such terms, the percentage obtained by the metric directly reflects the extent to which the functionality is "completely" specified. However, no reference values for "complete enough" is given, neither by argument nor by empirical evidence.
Recommendations for Action	The metric result yields combinations of system states and stimuli (inputs) for which no function is specified. The recommended action would be to specify this functions
Threats to Validity	• The metric definition is underspecified in terms of what is considered an elementary input and how to count all possible system states. An implementation of those functions is not easy and can be highly subjective and non-repeatable in practical settings. • A reference value other than 1 (which might not be feasible in practical situations) is not provided and might be hard to find and justify.

B.85. Pathfinder network similarity coefficient [Kudikyala and Vaughn, 2005]

Assessed (Measured) Attribute	Pragmatic Quality (Similarity of Requirements Clustering based on Semantic Coherence between Stakeholders and Developers: The extent to which stakeholders and developers cluster requirements according to semantic coherence similarly)
Data Collection	The data collection procedure consists of two main steps, (i) constructing a pathfinder network (graph) out of individual requirements from stakeholders and the intended audience of the SRS, and (ii) computing a correlation coefficient expressing the similarity between those two pathfinder networks. For obtaining the pathfinder network, a label is chosen by the quality manager for every single requirement in the SRS, and this label is written on index cards. Next, a group of stakeholders and a group of developers/project participants (representing the audience of the SRS) is identified which participate in further steps to obtain a pathfinder network per group. Each participant is asked to read the SRS to become familiar with the requirements on the index cards. Next, each group is asked to "group the index cards into different stacks based on perceived similarities and dissimilarities among the requirements", allowing requirements to be duplicated if the participants "[think] it belongs in more than one stack". Based on those clusters a NxN similarity matrix is constructed (N: number of requirements), where each cell denotes the number of times a pair of requirements co-occur in a stack, which is then transformed into a Pathfinder network as described in (Dearholt and Schvaneveldt, 1990), completing step (i). Next, similarity is assessed by computing the adjacency vectors for each node (requirement) v as the distance to all other nodes and computing its correlation coefficient. The similarity between the two Pathfinder networks is then computed as the mean of the correlations of all nodes v (described as C_7 in (Goldsmith and Davenport, 1990)).
Scale	Rational number (Ratio scale, The scale is a rational number between $[-1; 1]$ denoting the similarity measured as a correlation coefficient between the Pathfinder networks of the stakeholders and the SRS audience ("developers"). A value of -1 denotes the maximum dissimilarity, while a value of $+1$ denotes maximum similarity.)
Interpretation	Reference thresholds to compare the measurement value against are provided.
Reference Value	A correlation coefficient of below [-1,0.4) indicates "little or no similarity", [0.4,0.7] "moderate degree of similarity" and (0.7, 1] "very good to strong similarity", which according to the authors, translates into whether requirements are understood the same by the stakeholders and the developers (audience). Although not explicitly stated by the authors, the thresholds can be interpreted as the SRS must be revised, should be inspect or no further action regarding common understanding is necessary.
Recommend. for Action	If below a threshold, the correlation coefficients for individual requirements should be inspected, starting from the one with the least correlation coefficient, and discussed in conjunction with the stakeholder to identify misunderstandings.
Threats to Validity	• Results may be flawed when stakeholders or developers do not judge similarity exclusively based on the SRS but on background knowledge. For instance, a stakeholder who participated in the elicitation may understand a requirement in the SRS differently than a stakeholder who did not. For assessing pragmatic quality, the stakeholder should cluster requirements due to the intended ("correct") interpretation, while developers should represent the actual audience as close as possible, i.e., bring the same domain knowledge, RE participation and experience. • Since "similarity" between requirements is a vague relationship, it must be ensured that both stakeholders and developers have the same understanding of when requirements are considered "similar" and when not (e.g., similar from a domain in contrast to a technical perspective).

B.86. Percentage Leaf Goals with Object (PLGWO) [Espada et al., 2011, 2013]

Assessed (Measured) Attribute	Information Complete (Object Specification Completeness: Extent to which objects have been specified for all leaf goals)
Data Collection	Authors assume that goals are specified according to the KAOS methodology. During data collection, the number of leaf goals in total and those which have (at least) one object specified, are counted, and the ratio is computed according to the pseudo formula. Data collection was performed by hand in [espada2011measuring], while in [espada2013framework], a prototypical tool called modularKAOS was used. If there are no leaf goals, the metric is undefined.
Scale	Rational number (Ratio scale, The extent to which objects are specified for all leaf goals as demanded by the KAOS modeling guidelines on a scale of [0;1].)
Interpretation	According to the authors in [espada2013framework], any value below 1 is against the KAOS modeling guidelines and should be regarded as incomplete.
Reference Value	Authors suggest to interpret 1 as complete, and [0;1) as incomplete strictly following the KAOS modeling guidelines.
Recommendations for Action	During data collection, the goals without objects are identified, and should be reported to be completed (with help of the stakeholders if necessary).
Threats to Validity	• Empirical results in [espada2013framework] suggest that the reference values are too strict. Therefore, any real project would be classified as incomplete, rendering the metrics useless. Furthermore, the interpretation levels lack discriminative power (two levels only). • The number of specified operations and agents depends on the problem domain, and hence, the metrics result might be determined or constraint by the domain, limiting ist usefullness for individual projects (might still be used to characterize the SRS for a given domain).

B.87. Percentage Leaf Goals with Operation (PLGWOp) [Espada et al., 2011, 2013]

Assessed (Measured) Attribute	Feature Complete (Operation Specification Completeness: Extent to which operations have been completely specified for all leaf goals)
Data Collection	Authors assume that goals are specified according to the KAOS methodology. During data collection, the number of leaf goals in total and those which have (at least) one operations specified, are counted, and the ratio is computed according to the pseudo formula. Data collection was performed by hand in [espada2011measuring], while in [espada2013framework], a prototypical tool called modularKAOS was used. If there are no leaf goals, the metric is undefined.
Scale	Rational number (Ratio scale, The extent to which operations are specified for all leaf goals as demanded by the KAOS modeling guidelines on a scale of [0;1].)
Interpretation	According to the authors in [espada2013framework], any value below 1 is against the KAOS modeling guidelines and should be regarded as incomplete.
Reference Value	Authors suggest to interpret 1 as complete, and [0;1) as incomplete strictly following the KAOS modeling guidelines.
Recommendations for Action	During data collection, the goals without operations are identified, and should be completed (with help of the stakeholders if necessary).
Threats to Validity	• Empirical results in [espada2013framework] suggest that the reference values are too strict. Therefore, any real project would be classified as incomplete, rendering the metrics useless. Furthermore, the interpretation levels lack discriminative power (two levels only). • The number of specified operations and agents depends on the problem domain, and hence, the metrics result might be determined or constraint by the domain, limiting ist usefullness for individual projects (might still be used to characterize the SRS for a given domain).

B.88. Percentage Leaf Obstacles with Goal Resolution (PLOWS) [Espada et al., 2011, 2013]

Assessed (Measured) Attribute	Feature Complete (Obstacle Resolution Completeness: The extent to which resolutions are specified for all leaf goals)
Data Collection	Authors assume that goals are specified according to the KAOS methodology. During data collection, the number of leaf obstacles in total and those which have (at least) one resolution associated are counted, and the ratio is computed according to the pseudo formula. Data collection was performed by hand in [espada2011measuring], while in [espada2013framework], a prototypical tool called modularKAOS was used. If there are no leaf goals, the metric is undefined.
Scale	Rational number (Ratio scale, The extent to which resolutions are proposed for all leaf obstacles as demanded by the KAOS modeling guidelines on a scale of [0;1].)
Interpretation	According to the authors in [espada2013framework], any value below 1 is against the KAOS modeling guidelines and should be regarded as incomplete.
Reference Value	Authors suggest to interpret 1 as complete, and [0;1) as incomplete strictly following the KAOS modeling guidelines.
Recommendations for Action	During data collection, the obstacles without resolutions are identified, and requirement-level resolutions should be identified and documented (with help of the stakeholders if necessary).
Threats to Validity	• Empirical results in [espada2013framework] suggest that the reference values are too strict. Therefore, any real project would be classified as incomplete, rendering the metrics useless. Furthermore, the interpretation levels lack discriminative power (two levels only). • The number of specified operations and agents depends on the problem domain, and hence, the metrics result might be determined or constraint by the domain, limiting ist usefullness for individual projects (might still be used to characterize the SRS for a given domain).

B.89. Percentage of Descriptive Information (PDI) [Kenett, 1996]

Assessed (Measured) Attribute	Concise (Relative Amount of Descriptive Information: The amount of descriptive information in relation to the amount of descriptive and prescriptive information contained in the SRS.)
Data Collection	For a NL-SRS, every sentence is judged as being descriptive or prescriptive (presumably by a manual review process).
Scale	Rational number (Ratio scale, The percentage of sentences which are descriptive compared to all (descriptive and prescriptive) sentences.)
Interpretation	Values between 0 and 1 are obtainable. The values are to be interpreted against prescriptive reference values (thresholds), based on practical experiences but undisclosed in the publication.
Threats to Validity	• To understand whether a sentence is prescriptive or not, the stakeholders intention behind a sentence in the SRS must be clear. • Unclear domains / new products may impact the desirable/allowed percentage of descriptive statements.

B.90. Percentage of Operations with Agent (POpWA) [Espada et al., 2011, 2013]

Assessed (Measured) Attribute	Information Complete (Agent Specification Completeness: Extent to which agents have been completely specified for all leaf operations)
Data Collection	Authors assume that goals are specified according to the KAOS methodology. During data collection, the number of leaf operations in total and those which have (at least) one agent specified, are counted, and the ratio is computed according to the pseudo formula. Data collection was performed by hand in [espada2011measuring], while in [espada2013framework], a prototypical tool called modularKAOS was used. If there are no leaf goals, the metric is undefined.
Scale	Rational number (Ratio scale, The extent to which agents are specified for all operations as demanded by the KAOS modeling guidelines on a scale of [0;1].)
Interpretation	According to the authors in [espada2013framework], any value below 1 is against the KAOS modeling guidelines and should be regarded as incomplete.
Reference Value	Authors suggest to interpret 1 as complete, and [0;1) as incomplete strictly following the KAOS modeling guidelines.
Recommendations for Action	During data collection, the operations without agents are identified, and should be completed (with help of the stakeholders if necessary).
Threats to Validity	• Empirical results in [espada2013framework] suggest that the reference values are too strict. Therefore, any real project would be classified as incomplete, rendering the metrics useless. Furthermore, the interpretation levels lack discriminative power (two levels only). • The number of specified operations and agents depends on the problem domain, and hence, the metrics result might be determined or constraint by the domain, limiting its usefulness for individual projects (might still be used to characterize the SRS for a given domain).

B.91. Percentage of Strong Imperatives (PSI) [Wilson et al., 1997]

Assessed (Measured) Attribute	Unambiguous (Explicitly of the expectations on the requirements fulfillment: The extent to which the level of expectation associated with the requirements is explicitly and unambiguously captured in the requirements document)
Data Collection	Imperatives ("shall", must (not)", "is required to", "are applicable", "responsible for", "will" and "should", ordered by their "strength as a forceful statement of a requirement" in descending order) are automatically extracted and counted. Then, the number of each imperative is counted.
Scale	Natural number per Imperative(Ratio scale, A count of imperatives per type)
Interpretation	According to the authors, "requirements documents that were judged to be the most explicit had the majority of their imperative counts associated with the upper list items" and the numbers are compared to (two) previous projects. However, no specific counts (or normalized counts, i.e., ratios) are specified.-
Recommendations for Action	For every "weak" imperative, consult stakeholders to resolve them.
Threats to Validity	• Typos, synonyms and unspecified or implicit imperatives are not found by the automated procedure • Different project contexts can have significant impact on the reference values, because the authors writing style or the use of automated tools to generate SRS documents can influence the imperatives used.

B.92. Percentage of Universally Understandable Requirements (PUUR) [Davis et al., 1993a]

Assessed (Measured) Attribute	Pragmatic Quality (Understandability: The extent to which the actual readers are able to derive the knowledge specified in the SRS.)
Data Collection	All requirements in the SRS are counted. In addition, the SRS is manually reviewed. The number of requirements is counted for which for all "classes of reviewers" ("project managers, software developers, testers, users and customers" are explicitly mentioned) "thought they understood" the requirement.
Scale	Rational number (Ratio scale, Percentage of requirements "for which all reviewers thought they understood", compared to all documented requirements)
Interpretation	The values are interpreted in a range from 0 (no requirement is understandable by all reviewers) and 1 (every single requirement is understood by all reviewers). However, no reference value in-between is given.
Threats to Validity	• Due to underspecification of how understanding is measured, the internal validity depends on this factor. Therefore, a measurement technique must be able to thoroughly "test" the reviewers' understandings. • A reference value other than 1 is not provided and might be hard to justify.

B.93. Percentage of Validated Requirements (PVR) [Davis et al., 1993a]

Assessed (Measured) Attribute	Semantically Correct (Validation Coverage: The extent the SRS was validated against the stakeholders expectations, at the granularity level of individual requirements.)
Data Collection	We count the number of requirements specified in the SRS, and furthermore, the number of requirements which have been validated and are still contained in the SRS. This information must be disclosed either by the SRS directly (e.g., a status flag) or by the RE process.
Scale	Rational number (Ratio scale, Percentage of validated requirements in SRS compared to all documented requirements.)
Interpretation	The values are interpreted in a range from 0 (completely unvalidated) to 1 (all requirements were validated). However, no reference value is given.
Recommendations for Action	Valid the other requirements
Threats to Validity	• The metric definition is underspecified how it is disclosed that a requirements was validated or not. Therefore, incorrect flagging regarding validation can result in incorrect measurements. • A reference value other than 1 is not provided and might be hard to justify.

B.94. Positive Goal Contribution Rate (POS) [Kaiya et al., 2002]

Assessed (Measured) Attribute	Semantically Correct (Absence of conflicting requirements: The extent to which the SRS does not include requirements which conflict with stakeholder goals.)
Data Collection	The authors assume the presence of an attributed goal-graph according to their specific meta-model, with root elements (no predecessors) representing stakeholder demands and leaf elements representing finalised goals, i.e., refined requirements requiring certain functionality or characteristics of the (software) system. For any edge, an assigned contribution value denotes the degree to which a goal contributes to the achievement of the parent goal, expressed as an integer from -10 (maximal negative) to +10 (maximal positive contribution). Therefore, model conforming to this meta-model can be analyzed automatically, counting all paths (with and without only positive contributions attached to all edges of the paths)
Scale	Rational number (Ratio scale, The scale is a rational number between 0 and 1 which expresses the share of goals refinements (=path in goal tree) which contribute to the stakeholder needs, i.e., the initial goals (root nodes).)
Interpretation	The authors do not specify an interpretation procedure. Since a single value is obtained, prescriptive thresholds might work. However, because of the metric's complexity and the inherent trade-off associated with goals in practice (see also threats to external validity), the applicability of prescriptive reference values should be investigated empirically.
Threats to Validity	• Goal contribution is at least project-specific, i.e., the degree of achievable contribution depends for every single project (and can even change during projects in practice). Therefore, external validity seems hardly achievable. Therefore, we could imagine the interpretation procedure to include a derivation of project-specific reference values.

B.95. Positive Goal Contribution Rate (POS) [Kaiya et al., 2002]

Assessed (Measured) Attribute	Agreed (Absence of conflicting requirements: The extent to which the SRS does not include requirements which conflict with stakeholder goals.)
Data Collection	The authors assume the presence of an attributed goal-graph according to their specific meta-model, with root elements (no predecessors) representing stakeholder demands and leaf elements representing finalized goals, i.e., refined requirements requiring certain functionality or characteristics of the (software) system. For any edge, an assigned contribution value denotes the degree to which a goal contributes to the achievement of the parent goal, expressed as an integer from -10 (maximal negative) to +10 (maximal positive contribution). Therefore, model conforming to this meta-model can be analyzed automatically, counting all paths (with and without only positive contributions attached to all edges of the paths)
Scale	Rational number (Ratio scale, The scale is a rational number between 0 and 1 which expresses the share of goals refinements (=path in goal tree) which contribute to the stakeholder needs, i.e., the initial goals (root nodes).)
Interpretation	The authors do not specify an interpretation procedure. Since a single value is obtained, prescriptive thresholds might work. However, because of the metric's complexity and the inherent trade-off associated with goals in practice (see also threats to external validity), the applicability of prescriptive reference values should be investigated empirically.
Threats to Validity	• Goal contribution is at least project-specific, i.e., the degree of achievable contribution depends for every single project (and can even change during projects in practice). Therefore, external validity seems hardly achievable. Therefore, we could imagine the interpretation procedure to include a derivation of project-specific reference values.

B.96. Quality Requirements Spectrum Comparison [Kaiya and Ohnishi, 2011]

Assessed (Measured) Attribute	Semantically Correct (Difference in number of quality requirements specified in and required of the SRS: Extent (i) to which the SRS specifies quality requirements for the SuC, and (ii) of requirements which must be specified in the SRS in order for the SuC to satisfy its quality demands)
Data Collection	In a first step, the SRS is transformed into what the authors call "conceptual requirements description (CRD)": every noun is classified according to a predefined types (e.g., human, function, data, device, system, extsystem), and every sentence is classified according to predefined "concepts" (e.g., data flow, control flow, data creation, file manipulation) based on the nouns and verbs occurring in the sentence. Each such concept is structured according to a "case structure"; For instance for data flow, nouns for source, goal, object and instrument are extracted from the sentence. This step appears to be manual in general, but supported by tools. In a second step, two kinds of categorization rules have to be specified: IS? and SHOULD?. Both specify a certainty (either 0.5 or 1) and conditions for a CRD for predefined categories of quality requirements e.g., ISO9126). Finally, two vectors are obtained by summing the certainties of the IS? and SHOULD? functions for all requirement statements (CRDs), where each component of the vectors are given by the quality requirements categories.
Scale	2 vectors of natural numbers(Ratio scale, The metric yields two vectors of natural numbers; The vector maps categories of quality requirements (e.g., ISO9126) onto a weight which denotes the extend of requirements of the particular category to be present (IS) respectively should (SHOULD) be specified in the SRS. Each weight is natural number on a rational scale ranging from 0 (not present resp. not desired) to the number of statements in the SRS (maximum presence resp. desirable))
Interpretation	Authors propose to visualize both vectors in one bar chart ("spectrum"), where the categories constitute the horizontal axis, and the sums the vertical axis; Based on this graph, the requirements engineer selects the categories for which the IS and SHOULD values differ significantly (no further details provided by authors). If within a category the IS bar is higher compared to the SHOULD bar, the specification contains incorrect statements, and if the SHOULD bar is higher, the specification is incomplete since requirements are missing, with respect to the quality requirements category.
Threats to Validity	• The categorization of sentences into "concepts" is not straightforward and subject to misinterpretation. • The rule-based approach does potentially yields many false positives (requirements not related to a particular quality requirement) and negatives (i.e., requirements related to a quality requirement but not covered by the rules). Furthermore, the SHOULD? rules are used to express the exact extent to which quality requirements should be specified, and any deviation is considered harmful (either that statements are missing or that statements in the SRS are incorrect). According to our understanding, a rule-based system based on a (weighted) count of requirement sentences already present in the SRS seems hardly capable to estimate this demand accurately and reliably. Finally, the presence of partial incompleteness might masque the inclusion of incorrect requirements due to mutual compensation during aggregation. • Authors state that the set of rules depends on the project, therefore, it must be constructed and validated per project/domain/etc. The reuse of rules therefore is a threat to external validity.

B.97. Quality Requirements Spectrum Comparison [Kaiya and Ohnishi, 2011]

Assessed (Measured) Attribute	Semantically Complete ((Amount of) Quality Requirements (i) specified in and (ii) required of the SRS: Extent (i) to which the SRS specifies quality requirements for the SuC, and (ii) of requirements which must be specified in the SRS in order for the SuC to satisfy its quality demands)
Data Collection	In a first step, the SRS is transformed into what the authors call "conceptual requirements description (CRD)": every noun is classified according to a predefined types (e.g., human, function, data, device, system, extsystem), and every sentence is classified according to predefined "concepts" (e.g., data flow, control flow, data creation, file manipulation) based on the nouns and verbs occurring in the sentence. Each such concept is structured according to a "case structure"; For instance for data flow, nouns for source, goal, object and instrument are extracted from the sentence. This step appears to be manual in general, but supported by tools. In a second step, two kinds of categorization rules have to be specified: IS? and SHOULD?. Both specify a certainty (either 0.5 or 1) and conditions for a CRD for predefined categories of quality requirements e.g., ISO9126). Finally, two vectors are obtained by summing the certainties of the IS? and SHOULD? functions for all requirement statements (CRDs), where each component of the vectors are given by the quality requirements categories.
Scale	2 vectors of natural numbers(Ratio scale, The metric yields two vectors of natural numbers; The vector maps categories of quality requirements (e.g., ISO9126) onto a weight which denotes the extend of requirements of the particular category to be present (IS) respectively should (SHOULD) be specified in the SRS. Each weight is natural number on a rational scale ranging from 0 (not present resp. not desired) to the number of statements in the SRS (maximum presence resp. desirable))
Interpretation	Authors propose to visualize both vectors in one bar chart ("spectrum"), where the categories constitute the horizontal axis, and the sums the vertical axis; Based on this graph, the requirements engineer selects the categories for which the IS and SHOULD values differ significantly (no further details provided by authors). If within a category the IS bar is higher compared to the SHOULD bar, the specification contains incorrect statements, and if the SHOULD bar is higher, the specification is incomplete since requirements are missing, with respect to the quality requirements category.
Threats to Validity	• The categorization of sentences into "concepts" is not straightforward and subject to misinterpretation. • The rule-based approach does potentially yields many false positives (requirements not related to a particular quality requirement) and negatives (i.e., requirements related to a quality requirement but not covered by the rules). Furthermore, the SHOULD? rules are used to express the exact extent to which quality requirements should be specified, and any deviation is considered harmful (either that statements are missing or that statements in the SRS are incorrect). According to our understanding, a rule-based system based on a (weighted) count of requirement sentences already present in the SRS seems hardly capable to estimate this demand accurately and reliably. Finally, the presence of partial incompleteness might masque the inclusion of incorrect requirements due to mutual compensation during aggregation. • Authors state that the set of rules depends on the project, therefore, it must be constructed and validated per project/domain/etc. The reuse of rules therefore is a threat to external validity.

B.98. Rate of Action Steps (NOAS_RATE) [Duran et al., 2002, Bernárdez et al., 2004a]

Assessed (Measured) Attribute	Feature Complete (Completeness of Individual User-Visible Features: The extent to which individual features are described to match the stakeholders requirements and expectations about the specific features.)
Data Collection	The authors assume that use-cases are specified according to a specific metamodel, allowing the identification and countability of individual use case steps and the associated action owner. Therefore, the data collection procedure (counting all steps and actor steps) is automated (e.g., using a XSLT-based implementation as in Duran et al. [2002]). In Bernárdez et al. [2004a], the metric was proposed to be refined to count only Actor-System Actions but ignore Actor-Actor Actions within the use case for the sake of measurement.
Scale	Rational number (Ratio scale)
Interpretation	The obtained measurement is compared to a prescriptive reference interval. If the measured value exceeds the interval, the use-case is (potentially) flawed. In case the is too low, the use case is considered incomplete; in case it is too high, it is considered too detailed/complex to understand.
Reference Value	According to Bernárdez et al. [2004a], the NOAS_RATE should be in [30pct, 60pct].
Threats to Validity	• Low NOASRATEs were sometimes associated with batch processes, i.e., processes without user involvement. The authors state that this is indeed a defect, since "the use case is not actually a use case and it should have been expressed as a functional requirement in plain text". • The authors observed that the metric's applicability (with prescriptive reference values) depends on the writing style of the use case authors; It depends whether multiple actions by an actor or system which happen simultaneously or consecutively by the same actor/system are written as one step (e.g., concatenated by an "and") or as multiple steps. The metric was designed for the former style, and the latter one would require further adjustment.

B.99. Rate of Exception Steps (NOE_RATE) [Duran et al., 2002]

Assessed (Measured) Attribute	• Feature Complete (Completeness of Individual Features: The extent to which individual features are described to match the stakeholders requirements and expectations about the specific features.) • Pragmatic Quality (Complexity of Use Cases: The extent of complexity in understanding a use-case from the point of view of the receiver.)
Data Collection	The authors assume that use-cases are specified according to a specific meta-model, allowing the identification and countability of individual use case steps and the associated action owner as well as the use cases exceptions. Therefore, the data collection procedure (counting all steps and exceptions) is automated (e.g., using a XSLT-based implementation as in Duran et al. [2002]).
Scale	Rational number (Ratio scale,)
Interpretation	The obtained measurement is compared to a prescriptive reference interval. If the measured value exceeds the interval, the use-case is (potentially) flawed. In case the is too low, the use case is considered incomplete; in case it is too high, it is considered too detailed/complex to understand.
Reference Value	According to Duran et al. [2002], the NOERATE should be in [5pct, 45pct] according to experiments with students.

B.100. Rate of Use Case Action Steps (NOUS_RATE) [Duran et al., 2002, Bernárdez et al., 2004a]

Assessed (Measured) Attribute	Pragmatic Quality (Complexity of Use Cases: The extent of complexity in understanding a use-case from the point of view of the audience.)
Data Collection	The authors assume that use-cases are specified according to a specific meta-model, allowing the identification and countability of individual use case steps and the associated action owner. Therefore, the data collection procedure (counting all steps and use case actions/steps) is automated (e.g., using a XSLT-based implementation as in Duran et al. [2002]).
Scale	Rational number (Ratio scale,)
Interpretation	The obtained measurement is compared to a prescriptive reference interval. If the measured value exceeds the interval, the use-case is (potentially) flawed. In case the is too low, the use case is considered incomplete; in case it is too high, it is considered too detailed/complex to understand.
Reference Value	According to Bernárdez et al. [2004a], the NOUSRATE should be in [0%, 25%].
Recommendations for Action	Common defects reported were a "menu-like" or "programmer-like" structuring of use-cases. In the former case, multiple unrelated use-cases were merged into one use-case, while in the latter case, use cases were split into small chunks to improve "modularity". Both such defects could be resolved by reversing the splitting or use-cases.
Threats to Validity	• High NOUSRATEs were reported due to the limited REM meta-model, which forced participants to use inclusion/exclusion mechanisms for conditionals (due to technical limitations; unable to group steps in a conditional block)

B.101. Rationale Rate [Kaiya et al., 2002]

Assessed (Measured) Attribute	Metadata Complete (Rationale Complete: The extent to which rationale is provided)
Data Collection	The authors assume an attributed goal-model according to a strict metamodel. Such models allow the any graph element to be annotated by a rationale using a special syntactic construct. The counting of graph elements and the associated rationales can be automated.
Scale	Rational number (Ratio scale, The scale is a rational number between 0 and 1, denoting the extent to which the model's elements are annotated with rationales. A value of 0 denotes that the model does not include any rationales, while a value of 1 denotes that every element has at least on rationale provided.)
Interpretation	The authors do not specify an interpretation procedure, and no prescriptive reference values are provided. Furthermore, according to our judgment, a reference value of 0 might be too conservative (too many false negatives) while a value of 1 might be inappropriate for practical use. Therefore, we would advocate to derive prescriptive reference values from empirical evidence and motivate this research.

B.102. Reference Model Coverage [Cherfi et al., 2007, 2002]

Assessed (Measured) Attribute	Semantically Complete (Model expressiveness: According to the authors, a model is "said to be expressive when it represents users requirements in a natural way".)
Data Collection	The metric is applied on a conceptual model diagram consisting of entities and more-dimensional/N-ary relationships among them, with the inheritance relationship being a special, designated kind, and weights representing relative importance are associated with each diagram entity type. Not much detail on obtaining those weights except that they "result from a simulation process conducted on several examples" is given; The weights are 1 for both entities and relationships with cardinality (UML: multiplicity) 1:1, 3 for a M:N relationship-type; 4 for both ternary M:N:P relationships and inheritance links. In addition, a number of conceptual diagrams for the same domain/problem setting exist. During data collection, the weighted sum of all elements contained in the diagram under investigation is computed and divided by the weighted sum of all elements contained in the unified set of all conceptual diagrams for the same domain/problem setting, where the unified set denotes the mathematical set union operation is understood, i.e., joining all sets but not creating duplicates.
Scale	Rational number (Ratio scale, The scale denotes the extent to which a given model includes any elements also included in a reference set of models describing the same domain/problem setting. A value of 1 means that all elements are included, while a value of 0 denotes that no element of the reference set is included. If the model under consideration is itself included in the reference set and does contain elements (is not empty), a value of 0 is impossible.)
Interpretation	The authors do not provide an explicit interpretation procedure. In the paper, the metric is used to establish an order between a set of models. Therefore, a quality improvement could assessed using this metric, but no reference value of "good enough" is given.
Threats to Validity	• Since matching is assumed to be done by name, spelling errors and synonyms/homonyms can skew the results.

B.103. References to Incomplete Material (RIM) [Costello and Liu, 1995]

Assessed (Measured) Attribute	Semantically Complete (Incompleteness of stated identified demands: The amount of (SRS) content identified to be required but (yet) not specified)
Data Collection	The SRS is scanned for phrases of the form TBx, which stands for any acronym or full phrase of "to be determined" ("TBD"), "to be specified", "to be supplied" ("TBS"), or similar. The metric is obtained by counting those occurrences.
Scale	Natural number (Absolute scale, Number of references to nonexistent or incomplete material)
Interpretation	No interpretation procedure provided. However, it is stated that the metric "cannot provide a complete picture of the work remaining [...] [because] typically varying amours of effort to address [the identified areas are required]. Thus, TBC metrics reports should be presented with an analysis of the difficulty (technical and organizational) of resolving each item".
Recommendations for Action	Contribute to the identified missing parts of the SRS
Threats to Validity	• For the number of broken references, a valid decision procedure must be specified to objectively classify "incompleteness" of materials. • A reference value is not provided. 0 is the optimum, but may not be feasible in practice

B.104. Refined Customer Needs Ratio [Kaiya et al., 2002]

Assessed (Measured) Attribute	• Semantically Correct (Superfluous Requirements: The extent the SRS states requirements which do neither address nor conflict with the stakeholders needs or demands) • Metadata Complete (Rationale Complete: The extent to which a rationale is provided)
Data Collection	The authors assume an attributed goal-model according to a strict meta-model. Such models include specific syntactic constructs to identify initial goals (stakeholder needs) and finalized goals (requirements), and can therefore be analyzed automatically, according to the pseudo-formula.
Scale	Rational number (Ratio scale, The scale is a rational number between 0 and 1 which expresses the share of final goals linked to initial goals (linked=path exists).)
Interpretation	The authors do not specify an interpretation procedure. Since a single value is obtained, prescriptive reference values can be used. According to our understanding, any finalized goal (requirement) should be linked with customer needs (initial goals), and therefore a reference value of 1 seems reasonable.
Recommendations for Action	Finalized goals not connected to initial goals might suggest missing edges in the graph and hence the model should be checked for completeness. Other than that, finalized goals not connected to initial goals might also be incorrect requirements (requirements not requested/needed by stakeholders)

B.105. Relative Cloning Blow-Up [Juergens et al., 2010]

Assessed (Measured) Attribute	Concise (Enlargement due to Text Redundancy: Extent to which the SRS is enlarged due to duplicated text passages contained in it)
Data Collection	In a first step, all NL text from all physical documents comprising the SRS is extracted. Second, stop-words are removed and word stemming (according to the Porter stemming algorithm) is applied. Afterwards, a token-based clone detection algorithm is applied (for details, see B.S. Baker "On finding duplication and near-duplication in large software systems", WCRE'1995) with a minimal clone-length of 20 words (tokens). This algorithm yields a number of clones and clone groups, and is automated using the ConQAT tool according to the authors. Ina last and manual step, filters are iteratively applied to improve precision of the clone analysis. Filters are rules which ignore certain clones, e.g., because of their occurrence in document meta data, indexes, page decorations, of specification template information. This step is iterative, for which the authors result suggest that between 1 and 30 iterations are needed.
Scale	Rational number (Ratio scale, The extent to which an SRS is enlarged due to the contained duplication (clones), on a scale of [1;INF+[. A value of 1 denotes that cloning does not occur, while a value of close to 1 can be interpreted that cloning does not significantly increase the size of the SRS, due to (a) a small number of clones in the SRS and (b) the duplicated junks are short, or a combination of both.)
Interpretation	Authors do not specify an interpretation procedure as part of their paper; However, authors report empirical results for blowup between 0.0pct and 129.6pct, with an average of 13.5pct. Therefore, we propose two reference values: A value of about 10pct or less can be interpreted as slight blowup with no actions required, while a value of about 50pct or less denotes moderate blow-up and action should be considered; A value of more than 50pct denotes that action is required because substantial blow-up does limit conciseness substantially.
Recommendations for Action	Clone pairs are identified; Can be used to include cross references or create glossary entries instead.
Threats to Validity	• The tailoring depends on the understanding and expertise of the person measuring in order to not filter out non-benign clones. • Depending on the SRS itself, a word-level granularity might be inappropriate. • The use of templates or generated documentation (e.g., from models) can harm external validity; The iterative filtering might not fully mitigate those effects.

B.106. Relative Line Crossings [Cherfi et al., 2007, 2002]

Assessed (Measured) Attribute	Organized (Model Topology Organization: The extent to which lines in a diagram cross, normalized)
Data Collection	The metric is applied on a conceptual model diagram consisting of entities and more-dimensional/N-ary relationships among them, with the inheritance relationship being a special, designated kind. During measurement, sums the dimensions of all relationships between entities (dim(R)), and counts all inheritance links between entities as well as all line crossings in the diagram are collected. A data collection procedure is neither specified in Cherfi et al. [2002] nor in Cherfi et al. [2007], but the authors claim that the measure can be "computed automatically".
Scale	Rational number (Ratio scale, The scale denotes the extent to which lines denoting relationships among entities cross, normalized w.r.t. the maximal amount of line crossings. A value of 1 denotes no line crossings, while a value of 0 means that all relationships and inheritance link exactly once cross another line on average. According to the formula, the measurement value has no lower bounds.)
Interpretation	The authors do not provide an explicit interpretation procedure. In the paper, the metric is used to establish an order between a set of models. Therefore, a quality improvement could assessed using this metric, but no reference value of "good enough" is given. A value of 1 might be infeasible.

B.107. Relative Model Minimality (Absence of Redundancy) [Cherfi et al., 2007, 2002]

Assessed (Measured) Attribute	Concise (Model Redundancy: The extent to which a model does not contain redundant information)
Data Collection	The metric is applied on a conceptual model diagram consisting of entities and more-dimensional/N-ary relationships among them, with the inheritance relationship being a special, designated kind, and weights representing relative importance are associated with each diagram entity type. Not much detail on obtaining those weights except that they "result from a simulation process conducted on several examples" is given; The weights are 1 for both entities and relationships with cardinality (UML: multiplicity) 1:1, 3 for a M:N relationship-type; 4 for both ternary M:N:P relationships and inheritance links. During data collection, the weighted sum of all redundant elements (NB_R for element C_i) is computed and normalized w.r.t. the weighted sum of all elements in the conceptual schema. The detection of redundant elements, however, is not shared. Due to the claim that all measures can be "computed automatically" and by the example given in the paper, we assume that a matching of entities and relationships by name is done.
Scale	Rational number (Ratio scale, The scale denotes the extent to which a conceptual model contains redundant elements, weighted by relative importance of element types and normalized w.r.t. all elements. A value of 1 denotes that no redundant element is contained, while a value of 0 denotes that for every element in the model another redundant one exists on average. According to the formula, the scale has no lower bound.)
Interpretation	The authors do not provide an explicit interpretation procedure. In the paper, the metric is used to establish an order between a set of models. Therefore, a quality improvement could assessed using this metric, but no reference value of "good enough" is given.
Recommendations for Action	Redundant elements are obtained. The authors claim that redundant items are due to a lack of factorization, which therefore constitutes the recommendation for action.
Threats to Validity	• Since matching is assumed to be done by name, spelling errors and synonyms/homonyms can skew the results.

B.108. Relative Occurrence of Continuances (ROC) [Wilson et al., 1997, Fantechi et al., 2002]

Assessed (Measured) Attribute	Organized
Data Collection	Continuances ("below:", "as follows:", "following:", "listed:", "in particular:" and "support:") are identified and counted. As mentioned by the authors, "after much experimentation [...], lines of text [are used] as a normalization".
Scale	Rational number (Ratio scale, [0; inf[, phrases count (continuances) normalized by size measure (lines of text))
Interpretation	According to the authors, the extent of continuances used was interpreted after normalization. The authors further suggest to compare this number against SRS of previous projects. However, no specific reference value was given.
Reference Value	Specified in Fantechi et al. [2002]: The reference value (interval) of 0.1 - 0.2 is "derived from the good practices of NL requirements".
Threats to Validity	• Typos, synonyms and unspecified or implicit imperatives are not found by the automated procedure • Authors mention that "in some instances, extensive use of continuances was found to indicate the presence of very complex and detailed requirements specification statements." • Different project contexts can have significant impact on the reference values, because the authors writing style or the use of automated tools to generate SRS documents can influence the imperatives used.

B.109. Relative Occurrence of Directives (ROD) [Wilson et al., 1997, Fantechi et al., 2002]

Assessed (Measured) Attribute	Pragmatic Quality (Extensiveness of Descriptions: The extensiveness of information provided in terms of the "amount" but not the "precision" of provided information.)
Data Collection	Directives ("figure, table, for example, note:") are identified and counted. As mentioned by the authors, "lines or text [are used] as a normalization".
Scale	Rational number (Ratio scale, [0;inf), phrases count (continuances) normalized by size measure (lines of text))
Interpretation	According to the authors, the extent of directives thath was used was interpreted after normalization. The authors further suggest to compare this number against SRS of previous projects. However, no specific reference value was given.
Reference Value	Specified in Fantechi et al. [2002]: The reference value (interval) of 0.1 - 0.3 is "derived from the good practices of NL requirements".
Threats to Validity	• Typos, synonyms and unspecified or implicit directives are not found by the automated procedure • Different project contexts can have significant impact on the reference values, because the authors writing style or the use of automated tools to generate SRS documents can influence the imperatives used.

B.110. Relative Occurrence of Directives (ROD) [Wilson et al., 1997, Fantechi et al., 2002]

Assessed (Measured) Attribute	Information Complete (Extensiveness of Descriptions: The extensiveness of information provided in terms of the "amount" but not the "precision" of provided information.)
Data Collection	Directives ("figure, table, for example, note:") are identified and counted. As mentioned by the authors, "lines or text [are used] as a normalization".
Scale	Rational number (Ratio scale, 0; inf), phrases count (continuances) normalized by size measure (lines of text))
Interpretation	According to the authors, the extent of directives that was used was interpreted after normalization. The authors further suggest to compare this number against SRS of previous projects. However, no specific reference value was given.
Reference Value	Specified in Fantechi et al. [2002]: The reference value (interval) of 0.1 - 0.3 is "derived from the good practices of NL requirements".
Threats to Validity	• Typos, synonyms and unspecified or implicit directives are not found by the automated procedure • Different project contexts can have significant impact on the reference values, because the authors writing style or the use of automated tools to generate SRS documents can influence the imperatives used.

B.111. Relative Occurrence of Weak Phrases (ROWP) [Wilson et al., 1997]

Assessed (Measured) Attribute	Quantitatively Precise
Data Collection	Weak phrases ("adequate, as applicable, as appropriate, be capable, capability of, effective, normal, timely, as a minimum, easy, be able to, but not limited to, capability to, if practical, provide for, tbd") are identified and counted. As mentioned by the authors, "lines or text [are used] as a normalization".
Scale	Rational number (Ratio scale, $[0; +\infty)$, phrases count (continuances) normalized by size measure (lines of text))
Interpretation	According to the authors, the extent of weak phrases that was used was interpreted after normalization. The authors further suggest to compare this number against SRS of previous projects. However, no specific reference value was given.
Recommendations for Action	Correct identified "weak phrases"
Threats to Validity	• Typos, synonyms and unspecified weak phrases are not found by the automated procedure • Different project contexts can have significant impact on the reference values, because the authors writing style or the use of automated tools to generate SRS documents can influence the imperatives used.

B.112. Relative Punctuation per Sentence [Génova et al., 2013]

Assessed (Measured) Attribute	Syntactically Correct (Correct Punctuation: The extent to which punctuation is used according to grammatical rules of the language used)
Data Collection	The authors assume a NL-SRS in which individual requirements could be explicitly identified. The metric measures, for each requirement, the size of the requirement in words. This is implemented in specific tool proposed by the authors, and hence, automatable. However, we assume the data collection procedure to be automated for any modern word processing tool with minimal manual overhead (e.g., applying a word count function to individual requirements)
Scale	Rational number (Ratio scale, The extent to which punctuation signs are used in a sentence, per sentence. Scale is [0;INF).)
Interpretation	According to the authors, a convex (increasing-decreasing) value function was applied, for which the requirements atomicity is interpreted on three levels labeled "good", "medium" and "bad". The implication in case of "bad" is that the requirement must be changed (mandatory), while for "medium" the requirement should be inspected and changed if necessary (optional)
Threats to Validity	• The amount of punctuation used is not related to the amount that should be used according to grammatical rules, but approximated via sentence length, which harms both precision and recall. Furthermore, the correct use of punctuation is not measured, and hence, a measurement interpreted as "good" might be wrong and thus taken with a a grain of salt. • The metric was specified for English as the SRS language and the writing style of the author.

B.113. Relative Use of Simple Model Constructs [Cherfi et al., 2007, 2002]

Assessed (Measured) Attribute	Pragmatic Quality (Model Complexity: The extent to which a model is complex respectively simple.)
Data Collection	The metric is applied on a conceptual model diagram consisting of entities and more-dimensional/N-ary relationships among them, with the inheritance relationship being a special, designated kind. For data collection, the number of entity elements (elements of type entity) are counted and divided by the number (count) of any element in the diagram, i.e., the sum of the counts of relationships, inheritance links and entities.
Scale	Rational number (Ratio scale, The scale denotes the extent to which the model consists of entities w.r.t. the whole model. A value of 0 denotes that no entities were used at all, while a value of 1 denotes that the model consists exclusively of entities.)
Interpretation	The authors do not provide an explicit interpretation procedure. In the paper, the metric is used to establish an order between a set of models. Therefore, a quality improvement could assessed using this metric, but no reference value of "good enough" is given.

B.114. Relevant BPM Elements Ratio [Overhage et al., 2012]

Assessed (Measured) Attribute	Concise (Business Process Description Element Relevancy: The extent to which the elements of the business process description are relevant for the SRS audience)
Data Collection	Authors do not explicitly specify a measurement process; Therefore, we will assume a manual measurement process by means of quality managers (e.g., requirements engineers) on the business process description (both, using natural and graphical languages) as suggested in the paper's evaluation. Therefore, any element is analyzed for relevance, i.e., whether or not it can be removed from the model with loss of information for the SRS audience according to the quality manager. Finally, the ratio of relevant elements to all elements is established by counting both sets.
Scale	Rational number (Ratio scale, The extent to which the business process description represent those parts of the real world which are relevant for the SRS audience, on a scale of $[0; 1)$. A value of 0 denotes that all elements are relevant for the audience, while a value of 1 denotes that no element of the model contains valuable information for the SRS audience.)
Interpretation	Authors do not explicitly specify an interpretation procedure. Results during evaluation yielded rates between 0.00 and 10.1 (mean: 1.413, median: 0.37), also depending on the model element. Therefore, assuming that the models used in the evaluation resemble the models for the system under consideration, one may interpret the results particular to this "training set". For both kinds of interpretation, absolute and relative to a training set, further empirical evidence is required.
Recommendations for Action	Irrelevant elements are identified and should be removed from the diagram (or moved to an appendix)
Threats to Validity	• The metric is based on the quality manager's judgment about the relevancy of the model for the SRS audience. Therefore, the measurement is subjective and hence potentially lack robustness and consistency, due to the quality manager's perception or knowledge of the real-world domain. Furthermore, misinterpretations may also lead to flawed results. This is amplified in case no precise guidelines for judgment are given, which are rare for semantic model issues. • The metric measures perceived relevance from the viewpoint of the quality manager, therefore a potential gap between actual and perceived quality has to be considered. Furthermore, the metric does not account for the extent to which a model element is relevant i.e., it does not distinguish between a "tiny bit" and a critical loss of information, for which the former might be acceptable while the latter would result in severe problems for subsequent activities. Therefore, the metric can only provide a limited view on the relevancy of the business process description. Furthermore, an interpretation relative to a training set bears the problem that it may lead to false conclusion, e.g., because of the limited expressiveness of the training set on the general quality. • If relative prescriptive reference values are used, it must be made sure that the training set adequately resembles the system under consideration and project context.

B.115. Requirements Goal Coverage (COV) [Kaiya et al., 2002]

Assessed (Measured) Attribute	Feature Complete
Data Collection	The authors assume the presence of an attributed goal-graph according to their specific meta-model, with root elements (no predecessors) representing stakeholder demands and leaf elements representing finalized goals, i.e., refined requirements requiring certain functionality or characteristics of the (software) system. For any edge, an assigned contribution value denotes the degree to which a goal contributes to the achievement of the parent goal, expressed as an integer from -10 (maximal negative) to +10 (maximal positive contribution). Therefore, model conforming to this meta-model can be analyzed automatically, counting all paths (with and without only positive contributions attached to all edges of the paths)
Scale	Rational number (Ratio scale, The scale is a rational number between 0 and 1 which expresses the rate of initial goals which is refined into at least one requirement (=finalized goal) on at least one exclusively positive (contributing) path.)
Interpretation	The authors do not specify an interpretation procedure. Since a single value is obtained, prescriptive reference values might work. However, because of the metric's complexity and limited current understanding of causality, empirical investigations are needed.
Recommendations for Action	Not provided by the authors; The metric identifies stakeholder needs which may not have been refined into requirements. All identified requirements must therefore be checked for refinement completeness.
Threats to Validity	• Goal contribution is at least project-specific, i.e., the degree of achievable contribution depends for every single project (and can even change during projects in practice). Therefore, external validity seems hardly achievable. Therefore, we could imagine the interpretation procedure to include a derivation of project-specific reference values.

B.116. Resource Presence [Etien and Rolland, 2005]

Assessed (Measured) Attribute	Feature Complete (Business Resource Coverage: The extent to which all relevant business resources are specified in the system's description of the SRS)
Data Collection	The authors assume the presence of a description/model of the business domain and the systems implementation according to specific meta-models (ontologies) as well as a mapping between elements of those models in terms of mapping between elements (equivalent concepts) and a representation mapping (a system element may represent certain aspects of a business element). For this particular metric, the business resources (resources which are used in the business process supported by the system under consideration), both all and only those for which a system class (i.e., an object in the data model) exists, are counted. Depending on the actual tool, this process can be automated.
Scale	Rational number (Ratio scale, The scale denotes the extent to which business resources are also represented in the system to be build. A value of 0 and 1 denotes all respectively no resources are also represented in the system.)
Interpretation	The authors do no provide an explicit interpretation procedure. In the paper, a single value is computed, however, no preference values/thresholds are specified.
Recommendations for Action	The metric identifies business resources not specified in the SRS. Those resources have to be then checked if they have to be known for the system under consideration by consulting the stakeholders/domain experts, and then added to the specification to increase completeness if necessary.

B.117. Score Ordering [Mu et al., 2005]

Assessed (Measured) Attribute	Semantically Consistent
Data Collection	The requirements in the SRS are assumed to be specified as a set of logical propositions, or have to be formalized. The measurement description provides an ordering based on the Minimal Inconsistent Subset (MIS) defined in the same paper, which is used to compare a given SRS to reference values. Therefore, during data collection, the requirements engineer establishes a bijective mapping between the two sets of requirements. Afterwards, for any subset of requirements of the system under consideration, the MIS for all requirements except this subset must be checked to be less than of the comparator requirements mapped to by the bijective mapping. If this holds for all subsets, the comparison result is true, i.e., the requirements of the system under consideration is less inconsistent than those of the compared SRS. Otherwise, the comparison result is false.
Scale	Comparison Result(s) (Ordering Relation)(Dichatominal scale, The scale denotes whether a given SRS is less inconsistent compared to a reference SRS. It consists of two modes: "true" iff the SRS is less consistent than a reference SRS, or "false" otherwise.)
Interpretation	The authors intent is to compare if a chance of a SRS is indeed an improvement in consistency, and therefore propose the compare an altered SRS to its previous version. Although not explicitly mentioned in the paper, we assume that inconsistency should be improved upon during SRS changes and therefore would recognize "false" as triggering an action to resolve newly introduced inconsistency. This might not be true if new requirements are elicited which were not previously included, however, the metric is not applicable under those conditions since one is unable to establish a bijective mapping this way.
Recommendations for Action	The metric could yield the minimal satisfiable core introduced in the altered SRS, which is subject to resolve inconsistency.

B.118. Sentence-Level Syntax Correctness Ratio [Overhage et al., 2012]

Assessed (Measured) Attribute	Syntactically Correct (Sentence-level syntactical correctness: The extent to which a business process description contains (unique) grammar violations on the sentence-level.)
Data Collection	Authors do not explicitly specify a measurement process; Therefore, we will assume a manual measurement process by means of quality managers (e.g., requirements engineers) on the business process description (both, using natural and graphical languages) as suggested in the paper's evaluation. Therefore, during data collection, the number of sentence-level ("rules of grammar that prescribe the combination of individual words to larger units (sentences)") grammatical rules which are applied in the model and, in addition, those which violate these rules, are counted. Only first time occurrences are counted, i.e., multiple occurrences of the same violation to the grammatical rule are counted as one.
Scale	Rational number (Ratio scale, The extent to which a business process description contains unique violations to the grammatical rules concerning the sentence-level of the description language, relative to all grammatical rules applied in the description. The result is a rational number of the ratio scale [0;1], where 1 denotes that all grammatical rules are violated (by an unique fault), whereas a value of 0 denotes that the business process description does not contain any grammatical violations on the sentence-level.)
Interpretation	Authors do not explicitly specify an interpretation procedure. Results during evaluation yielded rates between 0.75 and 1.95 (median: 1.25). Therefore, assuming that the models used in the evaluation resemble the models for the system under consideration, one may interpret the results particular to this "training set". For both kinds of interpretation, absolute and relative to a training set, further empirical evidence is required.
Recommendations for Action	Individual grammar violations which are judged as incorrect are identified during the process and constitute candidates to be fixed.
Threats to Validity	• Languages with implicit, overwhelming or grammatical rules unknown to the quality manager may lead to a false judgment about the correctness of its application. • One has to be careful to not draw unwarranted conclusion from the measurement. First, the metric does consider only unique occurrences, and therefore, the business process description might be significantly less correct than suggested by a measurement interpreted as highly correct. And second, an interpretation relative to a training set bears the problem that it may lead to false conclusion, e.g., because of the limited expressiveness of the training set on the general quality. • If relative prescriptive reference values are used, it must be made sure that the training set adequately resembles the system under consideration and project context.

B.119. Shape of Text Structure (STS) [Wilson et al., 1997]

Assessed (Measured) Attribute	Organized
Data Collection	For each hierarchical level (sections, etc.) of the requirements document, the number of statements identifiers found are counted. Afterwards, the number of statements at each depth (i-th level) are summed. The obtained sums per level STS(L) are investigated if they can be classified as pyramidial, hour-glass or diamond, by checking the fulfillment of each categories constraints (see scale description).
Scale	Classification (Pyramid, Hour-Glass, Diamond)(Nominal scale, Three nominal values are provided: (i) Pyramidal shape (few number statements at level 1 and each lower level having more numbered statements), (ii) Hour-glass shaped (many numbered statements at high levels, few at mid levels and many at lower levels), and (iii) diamond shape (a pyramid followed by decreasing statement counts at levels below the pyramid))
Interpretation	If the text structure can be classified into the nominal scale, an interpretation for each of the classes is provided.
Reference Value	Pyramidal shape: "The text structure of documents judged to be well organized and having a consistent level of detail"; Hour-glass shape: "Usually contains a large amount of introductory and administrative information"; Diamond shape: "indicate that subjects introduced at higher levels were addressed at different levels of detail"
Threats to Validity	• As identified by the authors, the identification of levels is not straightforward and can thus count statements for the wrong hierarchical levels. • Different project contexts can have significant impact on the reference values, because the authors writing style or the use of automated tools to generate SRS documents can influence the imperatives used.

B.120. Shape of Text Structure (STS) [Wilson et al., 1997]

Assessed (Measured) Attribute	Pragmatic Quality (Extensiveness of Descriptions: The extensiveness of information provided in terms of the "amount" but not the "precision" of provided information.)
Data Collection	For each hierarchical level (sections, etc.) of the requirements document, the number of statements identifiers found are counted. Afterwards, the number of statements at each depth (i-th level) are summed. The obtained sums per level STS(L) are investigated if they can be classified as pyramidial, hour-glass or diamond, by checking the fulfillment of each categories constraints (see scale description).
Scale	Classification (Pyramid, Hour-Glass, Diamond)(Nominal scale, Three nominal values are provided: (i) Pyramidal shape (few number statements at level 1 and each lower level having more numbered statements), (ii) Hour-glass shaped (many numbered statements at high levels, few at mid levels and many at lower levels), and (iii) diamond shape (a pyramid followed by decreasing statement counts at levels below the pyramid))
Interpretation	If the text structure can be classified into the nominal scale, an interpretation for each of the classes is provided.
Reference Value	Pyramidal shape: "The text structure of documents judged to be well organized and having a consistent level of detail"; Hour-glass shape: "Usually contains a large amount of introductory and administrative information"; Diamond shape: "indicate that subjects introduced at higher levels were addressed at different levels of detail"
Threats to Validity	• As identified by the authors, the identification of levels is not straightforward and can thus count statements for the wrong hierarchical levels. • Different project contexts can have significant impact on the reference values, because the authors writing style or the use of automated tools to generate SRS documents can influence the imperatives used.

B.121. Similarity of Sentences based on Lexical Affinities [Kim et al., 1999, Park et al., 2000]

Assessed (Measured) Attribute	Feature Complete (Feature Completeness with respect to higher-level documentation (inter/intra document): The extent to which all features and characteristics are included in the SRS, limited to those features and characteristics documented in higher-level descriptions, both document-internal (e.g., requirements and goal models), and external documentation.')
Data Collection	The data is collected on natural-language specifications. To this end, sentences of high-level and low-level specifications (e.g., goals and software requirements) have to be extracted. Each high-level sentence is compared to all low-level sentences, and the number of sentences above a similarity threshold are counted (the comparison is based on lexical affinities (LA), i.e., words often occurring next to each, using a Sliding-Window technique; Details are presented in the publication).
Scale	Natural number (Absolute scale, The number of low-level statements which are similar for a high-level statement)
Interpretation	For data interpretation, the authors give the limited advice to "check consistency [...] because the system automatically displays which sentence in a low level document is most similar to the given sentence in a high level document" and "improving the completeness of SRS [if] there are not sentences in the low-level document associated with the given sentence". This manual improvement procedure however does not specify a quantitive evaluation
Reference Value	Authors implicitly suggest that any high-level sentence with no similar low-level sentences should be checked, therefore a reference value of "0" for each high-level statement,
Recommendations for Action	Identified potentially incomplete requirements
Threats to Validity	• Sliding window algorithm is based on a fixed (5) window size, meaning that lexical affinities apart more than 5 words are not captured. Furthermore, a normalization for often occurring words (e.g., "be") is done by normalizing the LA score based on the number of occurrences in the document, which in itself is only an indicator of entropy in a SRS. • The present computation is not perfect, since lexical affinities do miss some semantic affinities, e.g., by reference ("it"), etc.

B.122. Similarity of SRS Sentences based on Lexical Affinities [Kim et al., 1999, Park et al., 2000]

Assessed (Measured) Attribute	Semantically Consistent (Statements of the same fact: The amount of statements which describe the same fact, e.g., a system's characteristic or feature.)
Data Collection	The data is collected on natural-language specifications. To this end, sentences of high-level and low-level specifications (e.g., goals and software requirements) have to be extracted. Each high-level sentence is compared to all low-level sentences, and the number of sentences above a similarity threshold are counted (the comparison is based on lexical affinities (LA), i.e., words often occurring next to each, using a Sliding-Window technique; Details are presented in the publication).
Scale	Natural number (Absolute scale, The number of low-level statements which are similar for a high-level statement)
Interpretation	For data interpretation, the authors give the limited advice to "check consistency [...] because the system automatically displays which sentence in a low level document is most similar to the given sentence in a high level document" and "improving the completeness of SRS [if] there are not sentences in the low-level document associated with the given sentence". This manual improvement procedure however does not specify a quantitive evaluation
Reference Value	Authors implicitly suggest that any high-level sentence with no similar low-level sentences should be checked, therefore a reference value of "0" for each high-level statement,
Recommendations for Action	Identified potentially incomplete requirements
Threats to Validity	• Sliding window algorithm is based on a fixed (5) window size, meaning that lexical affinities apart more than 5 words are not captured. Furthermore, a normalization for often occurring words (e.g., "be") is done by normalizing the LA score based on the number of occurrences in the document, which in itself is only an indicator of entropy in a SRS. • The present computation is not perfect, since lexical affinities do miss some semantic affinities, e.g., by reference ("it") et cetera.

B.123. Size of Requirements [in Words] [Génova et al., 2013]

Assessed (Measured) Attribute	Singular (Atomic) (Size of Requirement: The extent to which a SRS consists of individual requirements of the "right" size, measured in number of words)
Data Collection	The authors assume a NL-SRS in which individual requirements could be explicitly identified. The metric measures, for each requirement, the size of the requirement in words. This is implemented in specific tool proposed by the authors, and hence, automatable. However, we assume the data collection procedure to be automated for any modern word processing tool with minimal manual overhead (e.g., applying a word count function to individual requirements). Afterwards, a convex (increasing/decreasing) scoring function is applied to map the size to a score 0,1,2. Finally, the arithmetic mean of all scores is calculated.
Scale	Rational number (Ratio scale, The extent to which requirements are within certain bounds regarding its size in words, on the scale of [0;2].)
Interpretation	According to the authors, individual requirements and a global interpretation are given. For each requirement individually, a decreasing interpretation function should applied be applied, "since the intention is to avoid an abuse". Each requirement's atomicity is interpreted on three levels labeled "good", "medium" and "bad". The implication in case of "bad" is that the requirement "must" be changed (mandatory), while for "medium" the requirement "should" be changed (optional). Furthermore, a score for all analyzed requirements is obtained on a scale of [0;2] with the same interpretation as above applied but for the set of all requirements is indicated.
Reference Value	In the example, individual requirement's atomicity is interpreted as "good" if the measurement value is in [40;200), "medium" if the measurement value is in either [10;40) or [80;200), and "bad" otherwise (i.e., [0;10) or [200;INF)). Scoring is applied based on the individual thresholds and a mean value is obtained this way on a scale of [0;2], where [0;05) denotes "Bad", [0.5;1.5) denotes "Medium", and [1.5;2] denotes "Good" atomicity of the SRS.
Recommendations for Action	Candidates violating singularity
Threats to Validity	• First and foremost, size and atomicity are only in a probabilistic relationship, hence any interpretation (level) has to be taken with a salt of grain. Special care has to be taken into account for the reference values used, since according to our experience there is a significant probability any requirement item will have between [10;200) words in practice. Therefore, align with the suggestion by the authors, the values should be tailored to the specific (project) context. Furthermore, a measurement value in [0;10) seems to be a special case: Here, the argument of the authors is that the requirement is "too short", which were unsure how to interpret: On the one hand, it can be seen as violating atomicity (notice that the term "atomic" is contra-intuitive here) in the sense that a single atomic requirement is split into 2 or more requirement items. On the other hand, it can be interpreted as an incomplete, and therefore invalid, requirement. Furthermore, using average as an average function can lead to overlook atomicity problems of individual requirements. • The metric's external validity is threatened by various factors. First, the metric was specified for English as the SRS language. In case of other languages, the prescriptive reference values may need to change. Other prominent examples we can think of are whether or not the requirement items contain descriptive information, such as domain descriptions, or not, the writing style of the author or the inherent complexity of the system-to-be.

B.124. Stakeholder-Perceived Lexical Ambiguity [Huertas et al., 2011]

Assessed (Measured) Attribute	Unambiguous (Stakeholder-perceived Lexical ambiguity: Extent to which words have multiple interpretations, given the understanding of the particular stakeholders involved with the SRS)
Data Collection	The authors propose a manual data collection procedure; All lexicals which are not (a) numbers, (b) special symbols (not part of a word) and (c) or words "only used as nexus, to connect others words or elements" are (uniquely) extracted of a NL-SRS (denoted Words). For each of those words, the number of meanings ([1;INF+], denoted SH Meanings) is obtained by asking each stakeholder for the number of meanings she associated with the word; The authors do not specify a merge process here, we therefore assume that the MAX function is applied to all results (other procedures are possible, e.g., naming the meanings, removing duplicates, and finally, counting). Finally, the number of ambiguous meanings are counted, divided by the number of words, and multiplied by the number of stakeholders minus 1 (see pseudo formula)
Scale	Rational number (Ratio scale, The extent to which a SRS contains lexical ambiguity as specified by all involved stakeholders, weighted by both the size of the SRS and the number of stakeholders (readers) of the SRS, on a scale of $[0; \infty+)$.)
Interpretation	Authors propose prescriptive reference values for the interpretation of the SRS.
Reference Value	Authors state that 0 is interpreted as "a completely non ambiguous requirement for the given set of stakeholders", and "any value equal or greater 1 a non acceptable ambiguity degree that indicates that you are dealing with a requirement specification that is dangerous and must be fixed before advancing to any other step". Therefore, any value (0;1) can be interpreted as not perfect but acceptable ambiguity.
Recommendations for Action	The number of lexical ambiguities associated with the words are obtained, thus, the most ambiguous words can be used as candidates for disambiguation.
Threats to Validity	• The metric relies on a good identification of ambiguity in terms of all meanings for a given word by the stakeholders; During data collection, this value can be influenced by external factors, e.g., the capabilities of retrieving all meanings for a given word by and/or time constraints of the stakeholders. Furthermore, authors explicitly state that if a stakeholder does not associate any meaning with a word (i.e., the word is unknown to her/him), the metric does not work. Therefore, for this border case, the data collection should be declared incorrect. • The metric has several threats to validity: (a) First, the specification of the number of meanings and the merge process using the mathematical max-function can be a lower bound only since meanings do not necessarily "overlap" (for a "perfect overlap", the SUM function would be appropriate). (b) Second,the measure does not take into account factors such as the number of occurrences of a word (for instance, a word occurring only once has the same impact as a word which occurs in every sentence) or any measure of "distance" between the identified meanings.

B.125. Subjective Language Smell [Femmer et al., 2014a]

Assessed (Measured) Attribute	Unambiguous (Use of Subjective Phrases: The extent to which "words of which the semantics are not objective" are used within the requirements specification)
Data Collection	In a first step, a natural language specification is parsed into single requirements. Afterwards, all sentences are analyzed for occurrences of certain phrases such as "user friendly, easy to use, cost effective". According to the tool presented in the paper, this process is automated and uses a dictionary (based on the ISO 29148:2011 standard and which may be altered over time) with a finite number of phrases. The number of subjective phrases corresponds to the number of dictionary matches.
Scale	Natural number (Absolute scale, Extent to which certain subjective phrases occur in the SRS.)
Interpretation	According to the authors, any occurrence of a subjective phrases is reported to the user, who is responsible to ultimately judge whether the finding is indeed an error or not. Since the authors do not explicitly specify a scale, neither a prescriptive reference value nor an interpretation procedure depending on the number of occurrences is given.
Recommendations for Action	Resolve identified terms, potentially with stakeholders
Threats to Validity	• Dictionary is incomplete, algorithm may not detect certain occurrences (e.g., because of typos)

B.126. Subjective Sentences (SS) [Fabbrini et al., 2000]

Assessed (Measured) Attribute	Unambiguous (Lexically Unambiguous (limited to subjective phrases): The extent to which a SRS contains lexicals (words) which are ambiguous)
Data Collection	All sentences in a NL-SRS are analyzed for occurrences of certain phrases, such as "having in mind, similar, better, similarly, worse, as ... as possible" based on a fixed dictionary. According to the tool presented in the paper, this process is automated and hence uses a pre-defined list (which may be altered over time) with a fixed number of phrases. In a second step, every occurrence is found "is presented to a reviewer" for validation.
Scale	Natural number (Absolute scale, Number of sentences which contain a personal opinion or feeling)
Interpretation	The obtained number of sentences containing options is compared to restrictive prescriptive reference values (called "quality requirements" for the indicators presented in the paper).
Reference Value	The reference value is 0, meaning the absence of any key phrase.
Recommendations for Action	Resolve subjective statements with stakeholders (e.g., remove from SRS)
Threats to Validity	• List of phrases is incomplete, algorithm may not detect certain occurrences (e.g., typos)

B.127. Superlatives [Femmer et al., 2014a]

Assessed (Measured) Attribute	Quantitatively Precise (Use of Adverbs and Adjectives in Superlative Form: The extent to which adverbs and adjectives are used in superlative form within the requirements specification)
Data Collection	In a first step, a natural language specification is parsed into single requirements. Afterwards, the part-of-speech (POS) for each word of every sentence is determined using a NLP tool. Next, every word marked as an adverb and adjective is analyzed and counted if it is in superlative form. According to the tool presented in the paper, this process is automated using the Stanford POS tagger and a morpological analysis.
Scale	Natural number (Absolute scale, Extent to which adverbs and adjectives in superlative form occur in the SRS.)
Interpretation	According to the authors, any occurrence of an adverb or adjective in superlative form is reported to the user, who is responsible to ultimately judge whether the finding is indeed an error or not. Since the authors do not explicitly specify a scale, neither a prescriptive reference value nor an interpretation procedure depending on the number of occurrences is given.
Recommendations for Action	Resolve identified terms, potentially with stakeholders

B.128. TBD Frequency (TBDF) [Kenett, 1996]

Assessed (Measured) Attribute	Information Complete
Data Collection	For a NL-SRS, each sentence is (presumably manually, but not specified in the publication) decomposed into attributes, e.g., the initiator of an action, the action, the object, etc. (9 types of attributes are given). Furthermore, the number of "TBD" (and potentially synonyms, not specified in publication) contained in the SRS is counted. Although unspecified in the publication, we assume that each TBD is counted as an ordinary attribute with special syntax (i.e., the term "TBD"). Therefore, it is included in the number of attributes.
Scale	Rational number (Ratio scale, The number of occurrences normalized to the size of the SRS in terms of identified attributes (=sentence parts according to predefined categories))
Interpretation	Values between 0 and 1 are obtainable in theory, with 0 being interpreted as there are no TBDs, and 1 being interpreted as every attribute is a TBD.
Recommendations for Action	Identified TBDs can be targeted by corrective action.
Threats to Validity	• The reference values seem to depend on the project, since the knowledge and boundedness of the problem space can significantly influence the number of TBDs.

B.129. Text Syntax Correctness Ratio [Overhage et al., 2012]

Assessed (Measured) Attribute	Syntactically Correct (Text-level syntactical correctness: The extent to which a business process description contains (unique) grammar violations on the text-level.)
Data Collection	Authors do not explicitly specify a measurement process; Therefore, we will assume a manual measurement process by means of quality managers (e.g., requirements engineers) on the business process description (both, using natural and graphical languages) as suggested in the paper's evaluation. Therefore, during data collection, the number of text-level grammatical rules ("combination of sentences to form complex expressions, i.e., the transitive combination of words to form texts") which are applied in the model and, in addition, those which violate these rules, are counted. Only first time occurrences are counted, i.e., multiple occurrences of the same violation to the grammatical rule are counted as one. An example for BPMN-models is that the same operator used for a control-flow split is used for latter merging.
Scale	Rational number (Ratio scale, The extent to which a business process description contains unique violations to the grammatical rules concerning the text-level of the description language, relative to all grammatical rules applied in the description. The result is a rational number of the ratio scale [0;1], where 1 denotes that all grammatical rules are violated (by an unique fault), whereas a value of 0 denotes that the business process description does not contain any grammatical violations on the text-level.)
Interpretation	Authors do not explicitly specify an interpretation procedure. Results during evaluation yielded rates between 0.46 and 1.11 (median: 1.08). Therefore, assuming that the models used in the evaluation resemble the models for the system under consideration, one may interpret the results particular to this "training set". For both kinds of interpretation, absolute and relative to a training set, further empirical evidence is required.
Recommendations for Action	Individual grammar violations which are judged as incorrect are identified during the process and constitute candidates to be fixed.
Threats to Validity	• Languages with implicit, overwhelming or grammatical rules unknown to the quality manager may lead to a false judgment about the correctness of its application. • One has to be careful to not draw unwarranted conclusion from the measurement. First, the metric does consider only unique occurrences, and therefore, the business process description might be significantly less correct than suggested by a measurement interpreted as highly correct. And second, an interpretation relative to a training set bears the problem that it may lead to false conclusion, e.g., because of the limited expressiveness of the training set on the general quality. • If relative prescriptive reference values are used, it must be made sure that the training set adequately resembles the system under consideration and project context.

B.130. To-Be-X Contained in SRS (TBx) [Costello and Liu, 1995]

Assessed (Measured) Attribute	Semantically Complete (Incompleteness of stated identified demands: The amount of (SRS) content identified to be required but (yet) not specified)
Data Collection	The SRS is scanned for "references to materials in other sections of the sepcification that is blank, has been omitted, [... or] that contains TBxs or [...] nonexistent or incomplete material", and the metric is obtained by counting those occurrences.
Scale	Natural number (Absolute scale, Number of TBx placeholders)
Interpretation	No interpretation procedure provided. However, it is stated that the metric "cannot provide a complete picture of the work remaining [...] [because] typically varying amounts of effort to address [the identified areas are required]. Thus, TBC metrics reports should be presented with an analysis of the difficulty (technical and organizational) of resolving each item".
Recommendations for Action	Contribute to the identified missing parts of the SRS
Threats to Validity	• The set of keywords scanned for must be complete and valid for the SRS at hand. • A reference value is not provided. 0 is the optimum, but may not be feasible in practice • Must be adapted for each language

B.131. Under-referenced Sentences [Fabbrini et al., 2000]

Assessed (Measured) Attribute	Organized (Completeness of References: The extent to which the references present in the SRS are complete, i.e., that the reference target exists and is explicitly stated.)
Data Collection	The number of sentences containing one or more explicit references (i.e., for any occurrence of phrases such as "according to, on the basis of, relatively to, compliant with, confirming to") that reference unnumbered sentences, documents not referenced in the SRS, or entities not defined nor described within the SRS. The collection procedure itself is not described.
Scale	Natural number (Absolute scale, Number of under-referenced sentences in the SRS.)
Interpretation	The obtained number of sentences containing options is compared to restrictive prescriptive reference values (called "quality requirements" for the indicators presented in the paper).
Reference Value	The reference value is 0, i.e., no under-referenced sentences are allowed.
Recommendations for Action	In case under-referenced sentences are detected, an explicit reference must be added.
Threats to Validity	• Depending on the measurement procedure

B.132. Underspecified Sentences (US) [Fabbrini et al., 2000]

Assessed (Measured) Attribute	Semantically Correct (Underspecified subjects: The extent to which a SRS contains statements about a general class of objects rather than a specific one)
Data Collection	All sentences in a NL-SRS are analyzed for "subjects [which] contain a word identifying a class of objects without a modifier specifying an instance of such class" (e.g., "The testers shall" is underspecified, but "the module testers shall" is not.). We do not know if the tool indeed is able to detect such anomalies automatically, hence we suspect a manual analysis process here.
Scale	Natural number (Absolute scale, Number of sentences with "underspecified" classes of objects as the subject)
Interpretation	The obtained number of sentences containing options is compared to restrictive prescriptive reference values (called "quality requirements" for the indicators presented in the paper).
Reference Value	The reference value is 0, meaning no subject denotes a class of objects as a whole.
Recommendations for Action	Resolve classes by determining the concrete instances that are contained respectively not contained.
Threats to Validity	• Rating a subject as a "too coarse class" can dependent on the reviewer, and hence, might be subjective.

B.133. Unexplained Acronyms in Sentences (UeS) [Fabbrini et al., 2000]

Assessed (Measured) Attribute	Unambiguous (Definition Coverage of Acronyms: The extent to which all acronyms present in the SRS are explained/for which a definition is provided.)
Data Collection	For every sentence in a NL-SRS, the occurrence of acronyms not explicitly and completely explained within the SRS are identified. The data collection procedure is not described.
Scale	Natural number (Absolute scale, Number of sentences with unexplained acronyms.)
Interpretation	The obtained (sentences with) unexplained acronyms are compared to restrictive prescriptive reference values (called "quality requirements" for the indicators presented in the paper)
Reference Value	The reference value is 0, i.e., no unexplained acronyms are allowed.
Recommendations for Action	Define unexplained acronyms in the SRS. Stakeholder involvement may be necessary therefore.
Threats to Validity	• Depending on the measurement procedure

B.134. Universal Quantification in Indicative Statements (UQ-IS) [Berry and Kamsties, 2000]

Assessed (Measured) Attribute	Semantically Correct (Incorrect universal quantification in domain descriptions: The amount of incorrect statements due to universal quantification in domain descriptions ("world" according to Jackson [1995]))
Data Collection	First, all statements which are indicative (descriptions of the domain) have to be extracted from the SRS, either manually by a reviewer or by considering specific content items of the SRS only (e.g., the domain model). Second, universality phrases ("all", "each", "every", "always", "none", "never" are implicitly mentioned) are identified. Although not explicitly specified, we suggest counting those statements.
Scale	Natural number (Absolute scale, Number of universality phrases in indicative (domain description) statements)
Interpretation	No reference value is given by the authors, but "any occurrence of universal quantification in indicative statements is potentially dangerous", although some are indeed true ("every human is mortal").
Recommendations for Action	Correct universal quantification by explicit mentioning of exceptions.
Threats to Validity	• Not all forms of universality phrases may be identified. Furthermore, not all indicative statements might be identified (e.g., because of review mistakes or insufficient adherence to the artifact model)

B.135. Unspecific Adverbs Smell [Femmer et al., 2014a]

Assessed (Measured) Attribute	Quantitatively Precise (Use of Unspecific Adverbs and Adjectives: The extent to which "unspecific adverbs and adjectives" are used within the requirements specification)
Data Collection	In a first step, a natural language specification is parsed into single requirements. Next, all sentences are analyzed for occurrences of certain phrases such as "almost always, significant, minimal". According to the tool presented in the paper, this process is automated and uses a dictionary (based on the ISO 29148:2011 standard and which may be altered over time) with a finite number of phrases. The number of loophole phrases corresponds to the number of dictionary matches.
Scale	Natural number (Absolute scale, Extent to which certain adverbs and adjectives defined as ambiguous occur in the SRS.)
Interpretation	According to the authors, any occurrence of a ambiguous adverb or adjective phrase is reported to the user, who is responsible to ultimately judge whether the finding is indeed an error or not. Since the authors do not explicitly specify a scale, neither a prescriptive reference value or an interpretation procedure depending on the number of occurrences is given.
Recommendations for Action	Resolve identified terms, potentially with stakeholders
Threats to Validity	• Dictionary is incomplete, algorithm may not detect certain occurrences (e.g., because of typos)

B.136. Use-Case Conditional Steps [Ciemniewska et al., 2007]

Assessed (Measured) Attribute	Pragmatic Quality (Complexity of Use Cases: The extent of complexity in understanding a use-case from the point of view of the receiver.)
Data Collection	Authors presume a dictionary for keywords regarding conditionals, such as "if, whether, when". The measure is specified on individual steps of use cases, both in the main flow and the exceptions/alternatives. During data collection, the number of words in the step's sentences which (i) occur in the dictionary and (ii) are a subordinating conjunction are counted. This step is automated based on a NLP tool, e.g., the Stanford parser.
Scale	Natural number (Absolute scale, The scale denotes the extent to which conditionals are used in a use-case step.)
Interpretation	The authors do not provide an explicit interpretation procedure. A value of 0 might not be achievable in practice and/or yield too many false positives. Authors mention that nested conditionals are even worse. Therefore, we would advocate to empirically investigate reference values for goals associated with decision making, e.g., acceptable and unacceptable levels.
Recommendations for Action	Conditionals should be moved from the main flow to the extensions/alternatives part.
Threats to Validity	• Because of typos, stemming etc., several words from the dictionary might be missed. • The metric should only be applicable of the writing rules for use cases should be respected (no conditionals).

B.137. Use-Case Similarity Factor [Ciemniewska et al., 2007]

Assessed (Measured) Attribute	Concise (Requirement Duplication: The number of clones or duplicated requirements occurring in a requirement specification.)
Data Collection	The data collection procedure is two-phased. "In the first stage a signature [...] of each use case is computed" which is "a combination of a main actor identifier (e.g. its number) and a number of steps a use case contains", according to which use cases are clustered. In a second step, all use-cases of each cluster are pair-wise compared. To this end, a StepSimilarity is computed for every step of the two use-cases, and the metric is the product of all StepSimilarities per use-case. The StepSimilarity for the i-th step of each use-case is determined as follows: If all the corresponding verbs and (i) all nouns are the same, or all but one corresponding nouns are the same, the conflicting pair is the same for all use-case steps of both use-cases (e.g., the use-cases are the same except UC-A always uses "bill" instead of "invoice" compared to UC-B), the StepSimilarity is defined as '0'. In all other cases, the StepSimilarity is defined as '1'.
Scale	$\{0, 1\}$ (Dichatominal scale, The scale denotes whether all steps of a use-case are considered similar (value equals 1), or not (value equals 0).)
Interpretation	The authors do not provide an explicit interpretation procedure. Because of the nominal scale we assume a value of 0 being interpreted as duplicated, while a value of 1 is interpreted as not duplicated.
Reference Value	1 = triggering recommended action; 0 = no action required
Recommendations for Action	Since duplicated pairs are identified, the pairs have to be checked if they are unintentional, and if yes, merged or removed.

B.138. Use-Case Step Actor Completeness [Ciemniewska et al., 2007]

Assessed (Measured) Attribute	Information Complete (Use-Case Actor Completeness: The extent to which use-case steps include the step's actor.)
Data Collection	Authors presume the prior specification of all actors (Actors). The measure is specified on individual steps of use cases, both in the main flow and the exceptions/alternatives. For each sentence of individual steps, the subject if extracted and checked for inclusion in the actors specification. If it is not included, a "warning" is issued, otherwise not. The measurement itself is automated in a tool based on the Stanford analyzer for POS tagging.
Scale	Incomplete(Warning), Complete(NoWarning) (Dichatominal scale, The scale denotes whether the subjects of all sentences of individual steps of the use-case are defined as actors.)
Interpretation	The authors do not provide an explicit interpretation procedure. Because of the nominal scale we assume a warning being interpreted as potentially flawed triggering the recommended action, while an absence of a warning is interpreted as no action required.
Reference Value	Warning = triggering recommended action; NoWarning = no action required
Recommendations for Action	In case a warning is issued, the individual step shall be completed by an actor, or potentially rephrased such that the actor is the sentence's subject.
Threats to Validity	• The metric should only be applicable of the writing rules for use cases should be respected (no passive sentences/actors as subjects).

B.139. Use-Case Step Linguistic Complexity [Ciemniewska et al., 2007]

Assessed (Measured) Attribute	Syntactically Correct (Sentence Structural Complexity: The complexity of a sentence used for describing each step of use case.)
Data Collection	The measure is specified on individual steps of use cases, both in the main flow and the exceptions/alternatives. Each sentence is investigated if it contains more than (i) one sentence, (ii) one subject or predicate, (iii) one coordinate clause, or (iv) one subordinate clause. Only if any of those conditions are present in a sentence, a "warning" is issued. The measurement itself is automated in a tool based on the Stanford analyzer for POS tagging.
Scale	Linguistically complex(Warning), Linguistically Simple (NoWarning)(Dichatominal scale, The scale denotes whether individual steps of the use-case are (potentially) too complex, or not.)
Interpretation	The authors do not provide an explicit interpretation procedure. Because of the nominal scale we assume a warning being interpreted as potentially flawed triggering the recommended action, while an absence of a warning is interpreted as no action required.
Reference Value	Warning = triggering recommended action; NoWarning = no action required
Recommendations for Action	In case a warning is issued, the individual step shall be checked for understandability for the reader of the SRS.
Threats to Validity	• The metric is a very rough heuristic. In both cases, false positives and negatives are possible and may be frequent. • Writing style and languages might require an adjustment of the measurement procedure (limits) and/or the prescriptive reference values

B.140. Use-Case Technical Jargon [Ciemniewska et al., 2007]

Assessed (Measured) Attribute	Design-Independent
Data Collection	Authors presume a dictionary of technical terms "typical for specific technologies and use interface (e.g. button, web page, database, edit box"). The measure is specified on individual steps of use cases, both in the main flow and the exceptions/alternatives. The metric counts the number of words of this dictionary which are also contained in all sentences of the use-case step. Although not explicitly mentioned, we assume from the context of the paper that this task is automated using linguistic tools such as the Stanford analyzer.
Scale	Natural number (Absolute scale, The scale denotes the extent to which technical jargon terms are used in a use-case step.)
Interpretation	The authors do not provide an explicit interpretation procedure. A value of 0 might not be achievable in practice, depending on the dictionary. Therefore, we would advocate to empirically investigate reference values for goals associated with decision making, e.g., acceptable and unacceptable levels.
Recommendations for Action	The measurement identifies technical terms in the SRS. For those terms, it must be verified that it is indeed a technical term and necessary. If not, it should be rephrased in the problem space rather than in the solution space.
Threats to Validity	• Because of typos, stemming etc., several words from the dictionary might be missed.

B.141. Use Case Cyclomatic Complexity (UCCC) [Duran et al., 2002, Bernárdez et al., 2004a]

Assessed (Measured) Attribute	Pragmatic Quality (Cyclomatic Complexity of Use Cases: The number of alternative paths of a use case ("defined in the same sense that McCabe [1976] defined CC for source code))
Data Collection	The authors assume that use-cases are specified according to a specific meta-model, allowing the identification and countability of individual use case steps and the associated action owner. Therefore, the data collection procedure (counting all steps and use case actions/steps) is automated (e.g., using a XSLT-based implementation as in [duran2002supporting]).
Scale	Rational number (Ratio scale,)
Interpretation	If the measured value is within the reference interval, the use-case is well understandable; Otherwise, it should be simplified.
Reference Value	According to Bernárdez et al. [2004a], the UCCC should be in [1,4].
Threats to Validity	• High UCCCs were reported due to the limited REM meta-model, which forced participants to use inclusion/exclusion mechanisms for conditionals (due to technical limitations; unable to group steps in a conditional block)

B.142. Vague Pronouns Smell [Femmer et al., 2014a]

Assessed (Measured) Attribute	Unambiguous (Use of Substituting pronouns: The extent to which substituting pronouns are used within the requirements specification)
Data Collection	In a first step, a natural language specification is parsed into single requirements. Next, the part-of-speech (POS) for each word of every sentence is determined using a NLP tool. Last, all words tagged as a substituting pronoun, i.e., a pronouns that do no repeat the original noun, is identified and counted. According to the tool presented in the paper, this process is automated.
Scale	Natural number (Absolute scale, Extent to which substituting pronouns occur in the SRS.)
Interpretation	According to the authors, any occurrence of a substituting pronoun is reported to the user, who is responsible to ultimately judge whether the finding is indeed an error or not. Since the authors do not explicitly specify a scale, neither a prescriptive reference value or an interpretation procedure depending on the number of occurrences is given.
Recommendations for Action	Replace substituting pronouns with the original noun. However, this might limit conciseness.

B.143. Vague Sentences (VS) [Fabbrini et al., 2000, Génova et al., 2013]

Assessed (Measured) Attribute	Unambiguous (Lexically unambiguous (limited to vague phrases): The extent to which a SRS contains lexicals (words) which are ambiguous)
Data Collection	All sentences in a NL-SRS are analyzed for occurrences of vague phrases, such as (i) intrinsic characteristics ("clear, well, easy, strong, weak, good, ..."), (ii) environmental characteristics ("useful, significant, ...") etc. According to the tool presented in the paper, this process is automated and hence uses a pre-defined list (which may be altered over time) with a fixed number of phrases. In a second step, every occurrence is found "is presented to a reviewer" who ultimately decides whether the sentence is really "incorrect". [Genova2013framework] provides an additional but abbreviated list of weak words, called "imprecise terms".
Scale	Natural number (Absolute scale, Number of sentences with vague phrases)
Interpretation	The obtained number of sentences containing options is compared to restrictive prescriptive reference values (called "quality requirements" for the indicators presented in the paper).
Reference Value	The reference value is 0, i.e., no vague terms are allowed in any sentence.
Recommendations for Action	Resolve vague phrases (with stakeholders)
Threats to Validity	• List of phrases is incomplete, algorithm may not detect certain occurrences (e.g., typos)

B.144. Weak Sentences (WS) [Fabbrini et al., 2000]

Assessed (Measured) Attribute	Unambiguous (Explicitly of the expectations on the requirements fulfillment: The extent to which the level of expectation associated with the requirements is explicitly and unambiguously captured in the requirements document)
Data Collection	Count the number of sentences containing weak words ("may, can, could") using an automated analysis. The results are validated by means of a manual inspection.
Scale	Natural number (Absolute scale, Number of sentences containing weak phrases)
Interpretation	The obtained number of sentences containing options is compared to restrictive prescriptive reference values (called "quality requirements" for the indicators presented in the paper).
Reference Value	The reference value is 0, i.e., no weak sentences are allowed.
Recommendations for Action	Resolve weak sentences (with stakeholders)
Threats to Validity	• List of phrases is incomplete, algorithm may not detect certain occurrences (e.g., typos)

B.145. Weighted Ambiguity (WA) [Kim et al., 1999, Park et al., 2000]

Assessed (Measured) Attribute	Unambiguous (Lexically Unambigous: The extent to which a SRS contains lexicals (words) which are ambiguous)
Data Collection	The natural-language specification is scanned for ambiguous words from a predefined set. For each word identified in the SRS this way, a manual review determines whether the word is indeed ambiguous in the context of this sentence, and updates an associated weight (>= 1) with this word. For future findings, only ambiguous words above a given threshold weight are counted. Following the argument of similar approaches (e.g., [wilson1997automated]), a normalization of the measure is desirable.
Scale	Natural number (Absolute scale, The number of ambiguous words, multiplied by a weight (>=1) for that word; The value itself is not normalized to the size of the SRS)
Interpretation	For data interpretation, no prescriptive thresholds for interpretation are specified and no normalization is performed. Therefore, a value of 0 can be interpreted as free of lexical ambiguity, while any value equal or larger than one is potentially problematic. Two (versions of) SRS can be compared when the same weights are used for both SRS, and the SRS are the same size; in this case, a higher value indicates a worse SRS.
Recommendations for Action	Ambiguous words are presented
Threats to Validity	• Since the data collection is automated, not all forms of the ambiguous words may be captured.

B.146. Weighted Ambiguity (WA) [Kim et al., 1999, Park et al., 2000]

Assessed (Measured) Attribute	Quantitatively Precise
Data Collection	The natural-language specification is scanned for ambiguous words from a predefined set. For each word identified in the SRS this way, a manual review determines whether the word is indeed ambiguous in the context of this sentence, and updates an associated weight (>= 1) with this word. For future findings, only ambiguous words above a given threshold weight are counted. Following the argument of similar approaches (e.g., [wilson1997automated]), a normalization of the measure is desirable.
Scale	Natural number (Absolute scale, The number of ambiguous words, multiplied by a weight (>=1) for that word; The value itself is not normalized to the size of the SRS)
Interpretation	For data interpretation, no prescriptive thresholds for interpretation are specified and no normalization is performed. Therefore, a value of 0 can be interpreted as free of lexical ambiguity, while any value equal or larger than one is potentially problematic. Two (versions of) SRS can be compared when the same weights are used for both SRS, and the SRS are the same size; in this case, a higher value indicates a worse SRS.
Recommendations for Action	Ambiguous words are presented
Threats to Validity	• Since the data collection is automated, not all forms of the ambiguous words may be captured.

B.147. Word Syntax Correctness Ratio [Overhage et al., 2012]

Assessed (Measured) Attribute	Syntactically Correct (Syntactical correctness of individual signs (words): The extent to which a business process specification includes individual signs (words) which do not adhere to the syntactic lexicon of the individual language.)
Data Collection	Authors do not explicitly specify a measurement process; Therefore, we will assume a manual measurement process by means of quality managers (e.g., requirements engineers) on the business process description (both, using natural and graphical languages) as suggested in the paper's evaluation. Therefore, during data collection, the number of language constructs that are modeled incorrectly, i.e., are not "formally correct" with respect to a lexicon for the language used. Only first time occurrences are counted, i.e., multiple occurrences of the same lexical error is counted only once.
Scale	Rational number (Ratio scale, The extent to which a business process description contains unique "words", ("signs" according to semiotic theory) which are not correct with respect to a lexicon, relative to its size in words. The result is a rational number of the ratio scale [0;1], where 1 denotes that all words are incorrect (and unique), whereas a value of 0 denotes that the business process description does not contain any incorrect signs.)
Interpretation	Authors do not explicitly specify an interpretation procedure. Results during evaluation yielded rates between 0.78 and 8.89 (mean: 3.89; median: 2.00). Therefore, assuming that the models used in the evaluation resemble the models for the system under consideration, one may interpret the results particular to this "training set". For both kinds of interpretation, absolute and relative to a training set, further empirical evidence is required.
Recommendations for Action	Individual words which are judged as incorrect are identified during the process and constitute candidates to be fixed.
Threats to Validity	• Languages with implicit, overwhelming or dictionaries unknown to the quality manager may lead to a false judgment about the correctness of words/signs. • One has to be careful to not draw unwarranted conclusion from the measurement. First, the metric does consider only unique occurrences, and therefore, the business process description might be significantly less correct than suggested by a measurement interpreted as highly correct. And second, an interpretation relative to a training set bears the problem that it may lead to false conclusion, e.g., because of the limited expressiveness of the training set on the general quality. • If relative prescriptive reference values are used, it must be made sure that the training set adequately resembles the system under consideration and project context.

Appendix C

Intrinsic Quality Properties

In this chapter we present the quality attributes identified by applying the ABRE-QM approach to the Unified Process [Jacobson et al., 1999] as described in Chap. 4 (pp. 63). The attributes are organized according to the activity they impact. Individual activities are specified in the order of occurrence within the Unified Process, and identified attributes for each activity are ordered by their ID in ascending order. For more details on the activities performed, including the involved audience roles and obtained artifacts, please refer directly to Jacobson et al. [1999]. For each attribute, we provide the following information:

ID An identifier used throughout this thesis to uniquely refer to individual properties

Intrinsic Property A very brief description of the intrinsic property.

Activity The activity that the property impacts.

Impact The property's impact on the quality-in-use of the activity it is associated with; The quality-in-use characteristics considered here are efficiency, i.e., the efforts associated with successfully performing the activity, and effectiveness, i.e., the quality of the artifacts which result from the activity

Rationale A short rationale why the intrinsic property indeed impacts the quality-in-use of the associated activity

Furthermore, the online companion material[1] additionally specifies the artifact and entity the intrinsic property is associated with as well as necessary conditions that must hold for the context of use for the property to be a valid quality attribute.

[1] `www4.in.tum.de/~mund/metrics-companion.zip`

C.1. Requirements Refinability

Structure UCs in diagrams

ID	Intrinsic Property	Impact	Rationale
9	The business process associated with activities and objects is specified	+Effectiveness & Efficiency	Since use-cases are derived from activities and may be structured according to the business processes they are contained in [Jacobson et al., 1999], knowing what activities belong to what business process helps the system analyst to save time in structering the system, and potentially resulting in a better structure compared that this information is indisclosed to the system analyst.
69	Inclusion and extension relationships between business activities specified	+Effectiveness & Efficiency	Since a use-case model contains inclusion and extension relationship between use-cases, it would be helpful to the system analyst if such relationships are already specified for business activities, resulting in a potentially superior use-case model (compared to that this information is indisclosed to the system analyst) in less time.

Name and describe actors

ID	Intrinsic Property	Impact	Rationale
10	All actors using or depending on the system are specified	+Effectiveness	Results in a more complete, hence superior, use-case model
53	Structured description of stakeholders operating with the system	+Effectiveness & Efficiency	A structured description of stakeholders makes them easiy identifiable, resulting in less efforts and more complete intial specification of use-cases
55	For each actor, at least one user who can enact it is specified	+Effectiveness	A user who can enact an actor constitutes a source if something is unclear and therefore potentially results in less flaws in the use-case model
56	Relationship between stakeholders specified	+Effectiveness	IF the relationship is known, it could be made explicit part of the model, resulting in a superior use-case or stakeholder model.
57	Provides navigation means to locate stakeholders	+Efficiency	It is not always straightforward to locate the stakeholders interacting with the system, and hence, if such stakeholders are documented, a navigation means should be provided to this documentation
58	Stakeholders are documented in one place	+Efficiency	Documenting stakeholders explicitly in one place saves the system analyst time to read the complete document while looking for stakeholders
60	Individual users aggregated to roles	+Efficiency	If already aggregated, this step must not be done by system analyst, and hence saves time.
62	Stakeholder names are lexically consistent	+Effectiveness	Homonyms and synonyms can result in missing important relationships (including equality) between stakeholders, resulting in an inferior model.
63	Stakeholder names syntactically correct and unambiguous	+Efficiency	Names which are not understood by the system analyst would require additional documentation or stakeholder/domain expert feedback, hence slowing down the identification of actors.

Identify use-cases from business model

ID	Intrinsic Property	Impact	Rationale
64	Detailed description of business processes supported by the system	+Effectiveness	Marking activities of the business process explicitly as to be supported by the information system, in contrast to activities which are not, saves the system analyst to understand all activities but only those relevant to the system development. Furthermore, if those activities are specified in detail, i.e., HOW they should be supported and with what quality, makes it easier for him to spot functionality and derive use-cases and extra-functional requirements, resulting in a superior use-case model.
64	Detailed description of business processes supported by the system	-Efficiency	If the business processes activities are specified in detail can result in additional overhead for experienced business analysts.
65	Business objects to be modified by the system specified	+Effectiveness & Efficiency	Same as for activities, business objects are candidates for use-cases. As such, the ones to be represented and handeled (e.g., the system must provide CRUD operations for it) should be designated in contrast to any business object which occurs in the business process the it system directly or indirectly supports. This way, the system analyst can identify use-cases more efficiently and with better results.
66	Activity names syntactically correct and unambiguous	+Efficiency	Names which are not understood by the system analyst would require additional documentation or stakeholder/domain expert feedback, hence slowing down the identification of actors.

Describe use-cases briefly

ID	Intrinsic Property	Impact	Rationale
67	Activities start with verb and are named after what shall be achieved	+Efficiency	As use-cases should start with a verb and be named after what shall be achieved, the system analyst would not need to interpret the activitiy himself but re-use the one already provided for the business activity, given that ist meaningful.

Structure use-case model

ID	Intrinsic Property	Impact	Rationale
13	Actors, Actions and Inputs/Outputs are lexically consistent	+Effectiveness	Avoiding synonyms and homonyms for actors, actions and inputs/outputs both increases the likelihood to find valid inclusion/extension relationships and decreases the likelihood that invalid inclusion/extension relationships are wrongly identified. Hence, a superior use-case model is obtained.
69	Inclusion and Extension between activities specifed	+Effectiveness & Efficiency	Since a use-case model contains inclusion and extension relationship between use-cases, it would be helpful to the system analyst if such relationships are already specified for business activities, resulting in a potentially superior use-case model (compared to that this information is indisclosed to the system analyst) in less time.

Prioritize use-cases

ID	Intrinsic Property	Impact	Rationale
31	All actions and flows specified	+Effectiveness	To understand the architectural impact of the use-cases, all actions and flows must be specified.
37	All capabilities and characteristics to be provided specified	+Effectiveness	All features (use-cases) must be known to the architect to select the most important ones from an architectural perspective.
49	Inputs/Outputs specified at system boundary	+Effectiveness & Efficiency	To understand the architectural impact of the use cases, all inputs and outputs must be specified at the system boundary, because otherwise the architect has to refine phenomena in the system's environment to concrete inputs and outputs (resulting in more resources and times spend), or could only approximate the architectural impacts from the enviroment phenomena, resulting in a potentially worse architecture description.

Identify (common, implementation) special requirements

ID	Intrinsic Property	Impact	Rationale
33	Quality requirements specified at function granularity	+Effectiveness & Efficiency	Extra-functional requirements may not be system wide, but be applicable to individual functions (use-cases or steps therein) only. Therefore, the specification of extra-functional requirements at function granularity enables the use-case engineer to more precisely specify the extra-functional ("special requirements" in RUP terminology) within the use-case.
34	Quality requirements specified quantitatively precise	+Effectiveness	Precise, quantitative values for extra-functional requirements, e.g., performance or reliability, enable the use-case engineer to specify those values in the use-case (realizations, both analysis and design).
35	Quality requirements are valid and unambiguous	+Effectiveness	If extra-functional requirements are invalid, a use-case engineer might include those in the detailed use-cases respectively the realizations of analysis/design.

Develop Glossary

ID	Intrinsic Property	Impact	Rationale
70	Terms used according to glossary or standard	+Effectiveness & Efficiency	If the terms of the business model can be referenced to an established standard, either a general one (e.g., ISO international standards) or a company-wide, the system analyst can reference those in the use-case standard.

Prototype User Interface

ID	Intrinsic Property	Impact	Rationale
9	The business process associated with activities and objects is specified	+Effectiveness	Understanding the business case surrounding the use-case can guide the UI designer to select more approriate UI elements, resulting in a better UI prototype.
31	All actions and flows specified	+Effectiveness	The UI designer must know all features (actions and flows) in order to designe the optimal user interface.
37	All capabilities and characteristics to be provided specified	+Effectiveness	The UI designer must know all features (actions and flows) in order to designe the optimal user interface.
47	Usage scenarios specified (including criticality and frequency)	+Effectiveness	Knowing how the system will be used, including how frequent certain functions are to be executed, in what order, and how critical they are, enables the UI engineer to better evaluate the suitability of certain UI element alternatives, resulting in a superior UI prototype.
49	Inputs/Outputs specified at system boundary	+Effectiveness & Efficiency	The UI designer must know which information is entered from and presented to the user. Therefore, the information exchange at the system boundary, i.e., the human computer (machine) interface, must be given. If instead the system interaction is specified as events in the environment, the UI designer must interpret those, resulting in a loss of time and potentially introducing mistakes.
74	End users are described in detail	+Effectiveness	The UI designer must understand the stakeholder's needs when interacting with the system. Therefore, she must know what background knowledge the user has and what information and guidance he requires of the system, and what information he can provide to the system in ways that fit into the user's tasks.
74	End users are described in detail	-Efficiency	The UI designer must understand the stakeholder's needs when interacting with the system. Therefore, she must know what background knowledge the user has and what information and guidance he requires of the system, and what information he can provide to the system in ways that fit into the user's tasks.
75	All end users are specified	+Effectiveness	The UI designer must know all users to design the best user interface.
79	(Parts of) UI specified	+Efficiency	If UI details are specified, either directly e.g. in terms of mockups, or indirectly, e.g., constrained to a limited subset of UI elements due to a particular technology being used, the UI designer has to choose from less alternatives, potentially saving time.
79	(Parts of) UI specified	-Effectiveness	If UI details are specified, either directly e.g. in terms of mockups, or indirectly, e.g., constrained to a limited subset of UI elements due to a particular technology being used, the UI designer may not choose the best UI elements to support the stakeholder in an optimal way.
80	Use-case is mapped to business process activity	+Effectiveness	Understanding the business case surrounding the use-case can guide the UI designer to select more approriate UI elements, resulting in a better UI prototype.

C.2. Analyzeability

Allocate system functionality (use-cases) to analysis packages

ID	Intrinsic Property	Impact	Rationale
1	Relationships (Generalization, Extension, Similarity) between use-cases are specified	+Effectiveness	Knowing the relationships between use-cases can improve the allocation to analysis package by achieving higher coherence within analysis packages
9	All business processes to be supported by the system are specified	+Effectiveness	Structuring functionality according to business processes, which Jacobson et al. [1999] mention as an allocation criterion, requires that all business processes are specified. Otherwise, the architect's allocation might not reflect the actual business structure, and hence the analysis package model might not localize changes less effectively.
10	All actors using or depending on the system are specified	+Effectiveness	Structuring functionality according to actors is another criterion for allocation, and hence, requires that all actors are specified. Otherwise, the obtained package model might localize changes less effectively.
11	Matching level of abstraction in business process and use-case, or explicit links are specified	+Effectiveness & Efficiency	If the business process is too abstract, the architect may be unable to map the activities to (parts of) use cases, resulting in either wrong or incomplete allocation of use cases to analysis packages. In contrast, when the business process is too detailed, the architect may spend more time than necessary to determine the use-cases involved in a business process.
13	Use Case names, actors, actions and inputs/outputs are lexically consistent to each other	+Effectiveness	Homonyms can lead to architects not allocating coherent use-cases into a common analysis package, while synonyms can lead to use-cases being separated despite being coherent
14	Use-case names, actors, actions and inputs/outputs are lexically consistent with terms in the business model	+Effectiveness	Different terms which are unnoticed by the architect lead to missing that a use-cases is part of a given business process, which in turn can result in an suboptimal allocation of use-cases to analysis packages

Identify and extract commonality among analysis packages

ID	Intrinsic Property	Impact	Rationale
13	Use Case names, actors, actions and inputs/outputs are lexically consistent to each other	+Effectiveness	Homonyms can lead to miss some commonalities, while synonyms can lead to falsely judge entities as being shared despite being different.
14	Use-case names, actors, actions and inputs/outputs are lexically consistent with terms in the business model	+Effectiveness	The relationship between the business objects (entities) and its usage within the use-cases must be understood by the architect. Therefore, either the same terms have to be used for the same or closely related/derived concepts in use-cases, or explicit links between entities in business model and use-cases must be provided.
15	Common Business Entities specified	+Effectiveness	Jacobson et al. [1999] state that shared classes that represent commonalities are very likely to be entity classes that can be traced to domain or business entity classes. It is thus worth studying the domain or business classes if they are shared". Hence, if unknown to the architect, thus business entities must exist and be linked to the various uses of them (in the business or domain model), which in turn results in more commonalities being identified.

Identify service packages

ID	Intrinsic Property	Impact	Rationale
1	Explicit specification of relationships between (parts of) use-case flows	+Efficiency	If commonalities of use-case flows are explicitly specified, the architect does not need to take care and save time
13	Common vocabulary for use case actions, inputs and outputs	+Effectiveness	A common vocabulary for actions enables easier identification of shared functionality and services
18	Structured description of use-case flows	+Efficiency	A structured description (delimitable steps, different intent according to primary/secondary actor) is easier to compare to each other, saving time to identify common functionality in use-cases which can be made part of a service packages
20	Consistent level of abstraction of use-cases	+Effectiveness	Comparing use-cases yields better results when on the same level of abstraction, potentially resulting in more common features found
38	Criticality of requirements specified	+Effectiveness	Explicitly marking (e.g., using a predefined vocabulary) features as mandatory or optional helps the architect to decide which services should be part of an optional service package

Find use-case flow beginning

ID	Intrinsic Property	Impact	Rationale
18	Structured description of use-case flows	+Efficiency	A structured description (delimitable steps, different intent according to primary or secondary actor) is easier to read, saving time to identify individual steps and responsibilities in use-cases.

Identify entity classes from use-cases

ID	Intrinsic Property	Impact	Rationale
11	Matching level of abstraction in business data model and use-case, or explicit links are specified	+Effectiveness & Efficiency	If the business process is too abstract, the architect may be unable to map the activities to (parts of) use cases, resulting in either wrong or incomplete allocation of use cases to analysis packages. In contrast, when the business process is too detailed, the architect may spend more time than necessary to determine the use-cases involved in a business process.
24	Objects referred to in use cases are lexically consistent to the data model	+Effectiveness	Homonyms cause one analysis to stand for multiple concepts, while synonyms cause duplicate objects referring to the same concept.
25	Detailed data model, specifying all relevant attributes and non-technical type	+Effectiveness	A detailed datamodel helps to identify entity objects and its relevant attributes, enabling the use case engineer to provide a more detailed outline of the analysis object
25	Detailed data model, specifying all relevant attributes and non-technical type	-Efficiency	A detailed datamodel will require more time of the use-case engineer to outline of the analysis object

Identify analysis classes

ID	Intrinsic Property	Impact	Rationale
13	Use Case names, actors, actions and inputs/outputs are lexically consistent to each other	+Effectiveness	Inconsistent vocabulary the architect is not aware of can result in either single analysis classes introduced for different concepts (in terms of unaware homonyms in the requirements specification) respectively multiple analysis classes for the same concept (in terms of unaware synonyms in the requirements specification). This is based on the description that "analysis classes already in the analysis model should, of course, be taken into consideration".
13	No homonyms/synonyms used for actors, actions and objects	+Effectiveness	Homonyms cause one analysis to stand for multiple concepts, while synonyms cause duplicate objects referring to the same concept (i.e., reduce reuse of analysis classes).
30	Source of requirements provided	+Efficiency	A source can be used to efficiently contact the stakeholder the requirement is originating from, e.g., to resolve questions about an objects importance, interpretation, etcetera.
37	All necessary features and characteristics specified	+Effectiveness	Omitted features (use-cases) can lead to omitted analysis classes

Identify analysis class interactions step by step in the use-case

ID	Intrinsic Property	Impact	Rationale
18	Structured description of use-case flows	+Efficiency	A structured description (delimitable steps, different intent according to primary/secondary actor) is easier to read, saving time to identify individual steps and responsibilities in use-cases.
31	All actions and flows specified	+Effectiveness	If any step is missing of ambiguous to the use-case engineer, the use-case realization description can be flawed

Identify central boundary classes from use-case

ID	Intrinsic Property	Impact	Rationale
26	All human actors interacting with system specified	+Effectiveness	Since an human actor who is not specified may remain unconsidered by the use case engineer, resulting in a missing analysis class (boundary class)
27	All external systems interacting with system specified	+Effectiveness	Since an external system which is not specified may remain unconsidered by the use case engineer, resulting in a missing analysis class (boundary class)

Detail Use Cases

ID	Intrinsic Property	Impact	Rationale
33	Extra-functional requirements specified at function granularity	+Effectiveness & Efficiency	Extra-functional requirements may not be system wide, but be applicable to individual functions (use-cases or steps therein) only. Therefore, the specification of extra-functional requirements at function granularity enables the use-case engineer to more precisely specify the extra-functional ("special requirements" in RUP terminology) within the use-case.
34	Extra-functional requirements specified quantitatively precise	+Effectiveness	Precise, quantitative values for extra-functional requirements, e.g., performance or reliability, enable the use-case engineer to specify those values in the use-case (realizations, both analysis and design).
35	Extra-functional requirements are valid	+Effectiveness	If extra-functional requirements are invalid, a use-case engineer might include those in the detailed use-cases respectively the realizations of analysis/design.

C.3. Designability

Identify nodes and network configurations

ID	Intrinsic Property	Impact	Rationale
27	All external systems interacting with system specified	+Effectiveness	Knowledge of external system the system under development must interoperate with is required to choose appropriate nodes and network configurations.
32	Nodes and network configurations predefined	+Efficiency	If some nodes and/or network configurations are already specified as constraints, the architect saves time to identify and evaluate alternatives
32	Nodes and network configurations predefined	-Effectiveness	A predefined node and/or network configuration can result in a deployment model which is inferior to the one proposed by the architect
33	Quality requirements specified at function granularity	+Effectiveness & Efficiency	Extra-functional requirements may not be system wide, but be applicable to individual functions only. Therefore, the specification of extra-functional requirements at function granularity enables the architect to adapt her evaluation to better match the intended purpose of the system. If such requirements are specified at the system level, the architect either require considerable effort to break down those requirements to individual functions (resulting in an efficiency loss) or proposes a less optimal deployment model (resulting in an Effectiveness loss).
34	Quality requirements specified quantitatively precise	+Effectiveness	Precise, quantitative values for extra-functional requirements, e.g., performance or reliability, enable the architect evaluate alternatives more profoundly, in particular, rule out solutions not able to satisfy the extra-functional requirements.
35	Quality requirements are valid and unambiguous	+Effectiveness	The architect must match the "limits and possibilites of the nodes and their connections" to the supplementary requirements, e.g., performance, reliability, security. Therefore, the proposed nodes and network configurations depend on those extra-functional requirements to be valid

Identify middleware and system-software subsystems

ID	Intrinsic Property	Impact	Rationale
27	All external systems interacting with system specified	+Effectiveness	Knowledge of external system the system under development must interoperate with is required to identify and choose appropriate middleware and system-software subsystems

Identify subsystems and their interfaces

ID	Intrinsic Property	Impact	Rationale
13	Central objects lexically consistent among use-cases	+Effectiveness	A common vocabulary for central objects (actions, inputs, outputs) enables easier identification of shared functionality and services
36	Subsystems (e.g., COTS) predefined	+Efficiency	If some subsystems/components (including COTS) are already specified as constraints or within use-cases, the architect saves time to identify and evaluate alternatives
36	Subsystems (e.g., COTS) predefined	-Effectiveness	A predefined subsystem/components (including COTS) can result in a design which is inferior to the one proposed by the architect
81	All use-cases follow a consistent structure	+Effectiveness & Efficiency	A common structure of use-cases allows an easier and faster identification of commonalities, enabling the architect to obtain a better subsystem design in less time
89	Value ranges and required precision specified in data model	+Effectiveness & Efficiency	Knowing the range (in particular, upper and lower bounds) and precision required of domain data enables the architect to choose the optiomal representation in the information system (i.e., the variable types). The extensive definition (defining all elements contained rather than define common characterstics, e.g., the interval from 1 to 100) saves the architect time because this definition allows a straightforward mapping to data types.

Identify new exceptions not found during requirements engineering and analysis

ID	Intrinsic Property	Impact	Rationale
30	Source of requirements provided	+Effectiveness	If the source is specified, the use-case engineer can consult the source (e.g., a stakeholder or standard) in order to support his design decisions.
31	All actions and flows specified	+Effectiveness & Efficiency	If a requirements specification indeed does specify all necessary features, including all actions and flows, the use-case engineer can choose freely any option as a refinement for design as long as such system does not contradict any specified feature. In contrast, if it is not complete, the use-case engineer can not be sure if the underspecification is intended (i.e., the stakeholder does value all alternatives roughly equivalent) or because a crucial requirement was not specified, either by mistake or because of a lack of understanding at the point in time the specification was written. In the first case, the use-case engineer can create better results in less time because he does not need to reconsider the stakeholders.
37	All necessary features and characteristics specified	+Effectiveness & Efficiency	If a requirements specification indeed does specify all necessary features, including all actions and flows, the use-case engineer can choose freely any option as a refinement for design as long as such system does not contradict any specified feature. In contrast, if it is not complete, the use-case engineer can not be sure if the underspecification is intended (i.e., the stakeholder does value all alternatives roughly equivalent) or because a crucial requirement was not specified, either by mistake or because of a lack of understanding at the point in time the specification was written. In the first case, the use-case engineer can create better results in less time because he does not need to reconsider the stakeholders.
38	Criticality of requirements specified	+Effectiveness	If the use-case engineer knows how critical a certain part of functionality is, she could propose solutions for less critical requirements while consulting domain experts or stakeholders for crucial requirements. This would result in a more suitable, hence superior, UC realization – design.
40	All system goals specified	+Effectiveness	If the systems' goals are known, the use-case engineer can base his design decisions with those goals in minds, resulting in a potentially better refinement from analysis to design, and hence, a superior UC realization–design.

Identify active classes

ID	Intrinsic Property	Impact	Rationale
33	Quality requirements specified at function granularity	+Effectiveness & Efficiency	Extra-functional requirements may not be system wide, but be applicable to individual functions only. Therefore, the specification of extra-functional requirements at function granularity enables the architect to more precisely determine which classes need to be active, resulting in an improved architecture.
34	Quality requirements specified quantitatively precise	+Effectiveness	Precise, quantitative values for extra-functional requirements, e.g., performance or reliability, enable the architect evaluate which classes should be active more profoundly, resulting in an superior architecture.
35	Quality requirements are valid and unambiguous	+Effectiveness	If extra-functional requirements are invalid, some classes might falsely become active (or not active), resulting in an inferior architecture.

Identify generic design mechanisms

ID	Intrinsic Property	Impact	Rationale
33	Quality requirements specified at function granularity	+Effectiveness & Efficiency	Extra-functional requirements may not be system wide, but be applicable to individual functions only. Therefore, the specification of extra-functional requirements at function granularity enables the architect to more precisely determine which design mechanisms to introduce for what parts of the system.
34	Quality requirements specified quantitatively precise	+Effectiveness	Precise, quantitative values for extra-functional requirements, e.g., performance or reliability, enable the architect evaluate which design alternatives more profoundly, resulting in an superior architecture.
35	Quality requirements are valid and unambiguous	+Effectiveness	If extra-functional requirements are invalid, design mechanisms might be falsely (not) introduced, resulting in an inferior architecture.

Propose design classes from analysis classes and architectural design

ID	Intrinsic Property	Impact	Rationale
89	Value ranges and required precision specified extensively	+Effectiveness & Efficiency	Knowing the range and precision required of domain data enables the architect to choose the optiomal representation in the information system (i.e., the variable types). This step is not already contained in the analysis model, since it does only specify domain data types but not technical aspects of its representation. The extensive definition (defining all elements contained rather than define common characterstics, e.g., the interval from 1 to 100) saves the architect time because this definition allows a straightforward mapping to data types.

Identify and report missing design classes

ID	Intrinsic Property	Impact	Rationale
31	All actions and flows specified	+Effectiveness	Missing design classes are found with respect to the systems' features, in particular ist actions and flow. Hence, in order to identify all missing design classes for the optimal system, the features must be known to the use case engineer, which in turn requires them to be specified (or otherwise communicated)
37	All necessary features and characteristics specified	+Effectiveness	Missing design classes are found with respect to the systems' features, in particular ist actions and flow. Hence, in order to identify all missing design classes for the optimal system, the features must be known to the use case engineer, which in turn requires them to be specified (or otherwise communicated)

C.4. Implementability

Understand the (limits of) functionality encapsulated per use-case

ID	Intrinsic Property	Impact	Rationale
1	Shared sections in use cases explicitly specified	+Effectiveness & Efficiency	If the inclusion and inheritence relationships are specified, the system integrator saves time because he does not need to determine those on his own, and it improves the result because he has a better understanding of shared parts, and can hence select use-cases which share a large portion of functionality to be integrated in the subsequent build.
2	Provides navigation means to locate features and characteristics	+Efficiency	requirements specification documents may contain more than prescriptive specifications of the system's features and characteristics. Therefore the time spent for the activity is also based on how fast the system integrator can locate the specification of the system's functionality. To this end, providing means to quickly navigate to features and characteristics can reduce searching for browse the whole requirements specification.
4	Requirements specified at system boundary scope	+Effectiveness & Efficiency	If features and characteristics are described as events in the environment rather than phenomena observable directly at the system's boundary, the system's boundary becomes blurred. Consequently, depending on the system integrator's knowledge, he is either unable to map those events to the system's boundary or requires additional analysis / stakeholder feedback, resulting in more time spent or a limited understanding of the system's functionality.
37	All necessary features and characteristics specified	+Effectiveness	If desired features or characteristics are not (correctly) specified, they will not be part of any subsequent build, and therefore not (correctly) implemented. Therefore, the implementation model is worse than if it would address those features or characteristics.

Determine features and characteristics to be implemented in next iteration

ID	Intrinsic Property	Impact	Rationale
2	Provides navigation means to locate features and characteristics	+Efficiency	requirements specification documents may contain more than prescriptive specifications of the system's features and characteristics. Therefore the time spent for the activity is also based on how fast the system integrator can locate the specification of the system's functionality. To this end, providing means to quickly navigate to features and characteristics can reduce searching for browse the whole requirements specification.
5	Priorities for requirements specified	+Effectiveness	If the priorities are unknown to or misunderstood by the system integrator, less important features might be selected for subsequent build, resulting in an integration build plan which in turn can lead to problems when the project runs out of time or budget, because at that point in time, less important features might be implemented while critical features are not, resulting in less acceptance from the customer.
6	Fine-grained prioritization	+Effectiveness	If the priorities are too coarse-grained (i.e., many features are classified as equally important) the system integrator may not be able to include all features in a given build and must select features and characteristics from the same priority. This can lead to a worse integration plan compared to a more fine-grained prioritization.
7	Subsystem decomposition specified	+Efficiency	If the subsystem decomposition is part of the requirements specification, the system integrator does not need to look into the system documentation and therefore safe time.

C.5. Testability

Plan Test

ID	Intrinsic Property	Impact	Rationale
1	Relationships between (parts of) features and characteristics are explicitly specified	+Effectiveness	Identifying shared functionality helps to understand cost reductions by sharing, enabling the test engineer to derive a more efficient testing plan
2	Provides navigation means to locate features and characteristics	+Efficiency	Navigation means, such as a table of contents or bookmarks in a written document, enable the test engineer to skip unnecessary parts of the requirements specification and hence save time.
5	Priorities for requirements specified	+Effectiveness	Jacobson et al. [1999] suggests to test those requirements with "the highest return on invest", for which both the importance of the requirement and the risk of not satsisfying it (e.g., due to a bug during implementation) must be considered.
6	Fine-grained prioritization	+Effectiveness	Given limit ressources, fine-grained prioritization potentially result in a superior test plan compared to the situation that a test engineer must decide between several requirements which are all characterized with the same priority.
46	Target system environment specified	+Effectiveness	A specification of the target system environment, i.e., the software and hardware the system under development shall be operating with, enables to create better test plans (because the resources and time required must be estimated) and test cases (because necessary preconditions on the system environment can be specified)
47	Usage scenarios specified (including criticality, frequency and parallel tasks)	+Effectiveness & Efficiency	Knowing requirements dependencies in the sense that a function must be processed in parallel to, before, etcetera another function enables the test engineer to propose a superior plan (more applicable test case and procedure), or force him to derive such information on his own.
47	Usage scenarios specified (including criticality and frequency)	+Effectiveness	Knowing how the system will be used, including how frequent certain functions are to be executed, in what order, and how critical they are, enables the test engineer to better evaluate which tests provide the best "return on investement" respectively which order of test cases should be considered (e.g., a user shall log in first).
52	Only prescriptive statements	+Efficiency	In order to derive the test cases, no explanations and rationales are required but the preskriptive statements must be translated into system actions and responses. If the requirements mix prescriptive and descriptive statements, additional effort to distinguish those must be overcome by the test engineer.

Design system test cases

ID	Intrinsic Property	Impact	Rationale
2	Provides navigation means to locate features and characteristics	+Efficiency	Navigation means, such as a table of contents or bookmarks in a written document, enable the test engineer to skip unnecessary parts of the requirements specification and hence save time.
46	Target system environment specified	+Effectiveness	A specification of the target system environment, i.e., the software and hardware the system under development shall be operating with, enables to create better test plans (because the resources and time required must be estimated) and test cases (because necessary preconditions on the system environment can be specified)
47	Usage scenarios specified (including criticality, frequency and parallel tasks)	+Effectiveness & Efficiency	Knowing requirements dependencies in the sense that a function must be processed in parallel to, before, etcetera another function enables the test engineer to propose a superior plan (more applicable test case and procedure), or force him to derive such information on his own.
47	Usage scenarios specified (including criticality and frequency)	+Effectiveness	Knowing how the system will be used, including how frequent certain functions are to be executed, in what order, and how critical they are, enables the test engineer to better evaluate which tests provide the best "return on investement" respectively which order of test cases should be considered (e.g., a user shall log in first).
48	Inputs/Outputs specified quantitatively	+Effectiveness & Efficiency	Quantitative values removes the burden to refine abstract values to concrete values from the test engineer, resulting in less time required and probably better results compared to qualitative values which the test engineer is unsure how to interpret.
49	Inputs/Outputs specified at system boundary	+Effectiveness & Efficiency	Specifying values at the system boundary instead of as phenomena in the environment removes the need to refine those values to the system boundary. This task requires additional resources and time, and may introduce faults into the test cases and procedures, resulting in inferior quality of those.

Design regression test cases

ID	Intrinsic Property	Impact	Rationale
50	Stability of requirements specified	+Effectiveness	Knowing how stable certain requirements are enables the test engineer to more profoundly select the test cases for regression testing which offer the "highest return on investment", i.e., the tradeoff between making test cases suitable for regression testing could be contrasted to the (predicted) frequency of changes of certain use-cases

Identify and structure test procedures

ID	Intrinsic Property	Impact	Rationale
31	All actions and flows specified	+Effectiveness	Only when all actions and flows are known to the test engineer, which can but must not be due to the specification, the optimal test cases and procedures could be selected and specified.