The Journal of Systems & Software 186 (2022) 111187

Contents lists available at ScienceDirect

The Journal of Systems & Software

journal homepage: www.elsevier.com/locate/jss

Quality measurement in agile and rapid software development: A systematic mapping☆

Lidia López [a,e,*], Xavier Burgués [a], Silverio Martínez-Fernández [a], Anna Maria Vollmer [b], Woubshet Behutiye [c], Pertti Karhapää [c], Xavier Franch [a], Pilar Rodríguez [d], Markku Oivo [c]

[a] Universitat Politècnica de Catalunya, Spain
[b] Fraunhofer Institute for Experimental Software Engineering, Germany
[c] University of Oulu, Finland
[d] Universidad Politécnica de Madrid, Spain
[e] Barcelona Supercomputing Center, Spain

ARTICLE INFO

Article history:
Received 1 February 2021
Received in revised form 22 October 2021
Accepted 6 December 2021
Available online 10 December 2021

Keywords:
Quality requirements
Non-functional requirements
Quality indicators
Metrics
Agile software development
Rapid software development

ABSTRACT

Context: In despite of agile and rapid software development (ARSD) being researched and applied extensively, managing quality requirements (QRs) are still challenging. As ARSD processes produce a large amount of data, measurement has become a strategy to facilitate QR management.
Objective: This study aims to survey the literature related to QR management through metrics in ARSD, focusing on: bibliometrics, QR metrics, and quality-related indicators used in quality management.
Methods: The study design includes the definition of research questions, selection criteria, and snowballing as search strategy.
Results: We selected 61 primary studies (2001–2019). Despite a large body of knowledge and standards, there is no consensus regarding QR measurement. Terminology is varying as are the measuring models. However, seemingly different measurement models do contain similarities.
Conclusion: The industrial relevance of the primary studies shows that practitioners have a need to improve quality measurement. Our collection of measures and data sources can serve as a starting point for practitioners to include quality measurement into their decision-making processes. Researchers could benefit from the identified similarities to start building a common framework for quality measurement. In addition, this could help researchers identify what quality aspects need more focus, e.g., security and usability that have surprisingly few metrics reported.

1. Introduction

Agile and rapid[1] software development (ARSD) are currently the two most prominent software development approaches (Anon, 2020). In a recent systematic mapping study (Behutiye et al., 2020), we pointed out that, despite the importance that quality has in the development of successful software products (Anon, 2021a), the management of quality requirements (QRs, defined as the desired qualities of a system under development) is still an open challenge in ARSD. More precisely, we found a number of issues related to *(i)* a limited ability to handle QRs through typical agile artifacts such as user stories, *(ii)* a tendency to neglect QRs in favor of functional requirements, and *(iii)* time constraints due to short iteration cycles that often have a negative impact on products' quality (Behutiye et al., 2020).

This open challenge is not a consequence of a lack of data. Rather the opposite, ARSD produces a large amount of data as a result of the development process, much of it related to quality. For example, Jira,[2] GitLab,[3] and Jenkins[4] produce test and bug related data that, when properly interpreted, can be used to provide evidence on a system's quality and guide quality management. This is in line with the ISO 9000 standard (Anon, 2021b), which defines evidence-based decision making as one of

☆ Editor: Neil Ernst.

* Corresponding author.
E-mail addresses: llopez@essi.upc.edu (L. López), diafebus@essi.upc.edu (X. Burgués), silverio.martinez@upc.edu (S. Martínez-Fernández), anna-maria.vollmer@iese.fraunhofer.de (A.M. Vollmer), Woubshet.Behutiye@oulu.fi (W. Behutiye), Pertti.Karhapaa@oulu.fi (P. Karhapää), franch@essi.upc.edu (X. Franch), pilar.rodriguez@upm.es (P. Rodríguez), Markku.Oivo@oulu.fi (M. Oivo).

[1] Rapid software development (RSD) refers to rapid and continuous software engineering approaches focusing on the capability to develop, release, and learn from deployed software in short cycles, that can be from hours to a few weeks (e.g., continuous integration and continuous delivery, and DevOps).

[2] https://www.atlassian.com/software/jira.
[3] https://about.gitlab.com/.
[4] https://www.jenkins.io/.

https://doi.org/10.1016/j.jss.2021.111187

the seven quality management principles (Quality Management Principles, 2015). In the particular case of quality management, measurement is seen as a central feature to assess software quality characteristics (Malhotra, 2016). In our previous study (Behutiye et al., 2020), we learned that measurement, which was not our initial focus of attention, is particularly relevant in QR management. For this reason, we decided to extend our original study and further explore the literature that particularly focuses on quality measurement in ARSD.

Software metrics, which focus on collecting development data, play an important role in measurement. They have now been studied for decades (Kitchenham, 2010). Still, the scientific understanding on the use of software metrics in ARSD is rather limited (Ram et al., 2018; Kupiainen et al., 2015). The literature suggests that metrics used in agile software development may conflict with traditional measurement programs (Kupiainen et al., 2015). For example, traditional approaches often use metrics to track progress against a preconceived plan, instead of embracing change, as it should be the main focus in agile software development. Moreover, the Agile principle of simplicity may conflict with traditional measurement programs that are based on a rather heavyweight set of metrics (Kupiainen et al., 2015).

The overall picture of metrics used to manage quality in ARSD is unclear. On the one hand, QRs are more difficult to measure and monitor than functional requirements (Karhapää et al., 2021). On the other hand, quality is seen as an abstract concept that is not limited only to the characteristics considered under the umbrella of QRs. For example, aspects such as customer satisfaction or release on time are also important when considering quality but are not explicitly considered under any specific QR. Therefore, measuring QRs provides only a partial view on quality. For this reason, we decided to focus our study on two measurement aspects: *(a) QR metrics*, defined as metrics used to measure QRs, and *(b) Quality Management Indicators* (QMIs), defined as indicators used in the context of quality management, usually as input for some decision-making processes at a strategic level.

This study provides a thorough analysis of the scattered evidence that conforms the state of the art of quality measurement in ARSD and points out research gaps on the area, through a systematic mapping study that:

1. Identifies and classifies the existing scientific literature on quality measurement in the context of QR management in ARSD.
2. Assesses the quality of existing empirical studies on the area in terms of research rigor and industrial relevance.
3. Identifies and classifies QR metrics in the context of ARSD reported in the literature.
4. Identifies and classifies data sources used to compute QR metrics as well as the tools that produce them.
5. Identifies and classifies QMIs used in ARSD, their base metrics (referred as *QMI metrics*) and data sources, the tools used to visualize them, the entities that are their focus (i.e., product, process, project, resources) and the relationship that the identified QMIs have with regards to QRs, analyzing on top of which QRs are the QMIs metrics defined.

The rest of the paper is structured as follows. In Section 2, we explain the central concepts and introduce related work. Section 3 presents the research questions and details our research method. In Section 4 we present the results that answer our research questions. In Section 5 we discuss the implications of the results for researchers and practitioners. Finally, Section 6 concludes the study.

2. Background and related work

2.1. Measuring quality

One of the seven quality management principles stated in ISO 9000 is *Evidence-based decision making*, which states that effective decisions are based on the analysis of data and information (Anon, 2021b). In order to be able to apply this principle, organizations need to determine, measure and monitor key indicators to demonstrate the organization's performance (Quality Management Principles, 2015). In the context of software measurement, ISO/IEC/IEEE 15939 (Anon, 2017) identifies a process including the definition of a suitable set of measures that address specific information needs, but it does not provide the set of measures to be used as part of the measurement plan.

In this study, we are interested in measuring software quality in the context of ARSD processes. ARSD are particularly well-suited for evidence-based decision making, because they produce a large amount of data during the development processes, providing evidence that can be used for quality management. In ARSD, software quality demanded by stakeholders is characterized by quality requirements (QRs) documented in the software specification. The ISO/IEC 25010 standard provides a definition for QRs and quality models to help determine which quality characteristics should be considered when assessing certain quality properties (Anon, 2011). A QR is defined by ISO/IEC 25010 as a "requirement that a software quality attribute be present in software". Given this focus on the software, we adopt in the rest of the paper the *software product quality model* defined in the standard as a conceptual framework to classify QRs. This quality model is composed of eight characteristics that relate to static properties of software as an artifact, and dynamic properties of the software system in execution (see Table 1).

Table 2 contains some definitions from both aforementioned ISO standards that are used in the context of this study.

2.2. Measuring quality in ARSD

In the search for previous secondary studies on the topic of quality measurement in ARSD, we found four secondary studies after a keyword-driven automatic search (Kupiainen et al., 2015; Arvanitou et al., 2017; Biesialska et al., 2020; Arcos-Medina and Mauricio, 2019), and an additional one after performing snowballing on them (Mishra and Abdalhamid, 2018). None of them cover the same scope as ours (see Fig. 1). Kupiainen et al. (2015) report on using metrics in industrial lean and agile software development teams analyzing the reasons for and the effects of using them. Arvanitou et al. (2017) report on design-time quality attributes and related metrics. Biesialska et al. (2020) analyze how big data analytics is used in the agile development lifecycle. Arcos-Medina and Mauricio (2019) analyze critical success factors, metrics to measure quality, quality attributes, agile practices, quality models, and agile principles. Mishra and Abdalhamid (2018) explore quality in Scrum research, including attributes and metrics used for quality assessment.

Besides the secondary studies, we found a tertiary study on agile software development (Hoda et al., 2017). This tertiary study includes a table with a list of secondary studies. Again, as Hoda et al. revealed in their analysis (Hoda et al., 2017), none of them focuses on the particular topic of the present study. Likewise, one of the secondary studies (Arvanitou et al., 2017) refers to a set of other secondary studies in its related work analysis (Section 2), but they do not contain works on the specific area of quality measurement in ARSD either. Therefore, we have not found any study in the same topic as the present study (see Fig. 1).

Although not exactly in the same area, we can still find some common topics:

Table 1
ISO/IEC 25010:2011 software product quality model.

Characteristic	*Sub-characteristics*
Functional Suitability	Functional Completeness, Functional Correctness, Functional Appropriateness
Performance Efficiency	Time Behavior, Resource Utilization, Capacity
Compatibility	Co-existence, Interoperability
Usability	Appropriateness Recognizability, Learnability, Operability, User Error Protection, User Interface Aesthetics, Accessibility
Reliability	Maturity, Availability, Fault Tolerance, Recoverability
Security	Confidentiality, Integrity, Non-repudiation, Authenticity, Accountability
Maintainability	Modularity, Reusability, Analysability, Modifiability, Testability
Portability	Adaptability, Installability, Replaceability

Table 2
Quality measurement concepts used in this paper (with reference to the ISO standard defining the concept).

Concept	*Definition*
Base measure (15939)	Measure defined in terms of an attribute and the method for quantifying it
Entity (15939)	Object that is to be characterized by measuring its attributes. An entity can be a process, product, project or resource.
Indicator (15939)	Measure that provides an estimate or evaluation of specified attributes derived from a model with respect to defined information needs
Measure (noun) (25010)	Quantitative indication of extent, amount, dimension, capacity, or size of some attribute of a product or process
Measurement (15939)	Set of operations having the object of determining a value of a measure
Measurement process (25010)	The process by which numbers or symbols are mapped to attributes of entities in the real world in such a way as to describe them according to clearly defined rules.
Metric (25010)	Quantitative measure of degree to which a system, component or process possesses a given attribute. "A handle or guess about a given attribute."
Process (15939)	Set of interrelated or interacting activities that use inputs to deliver an intended result
Product (15939)	Result of a process
Project (15939)	Endeavor with defined start and finish criteria undertaken to create a product or service in accordance with specified resources and requirements
Quality model (25010)	Defined set of characteristics, and of relationships between them, which provides a framework for specifying quality requirements and evaluating quality
Quality requirement (25010)	Requirement that a software quality attribute be present in software
Resource[a]	A useful or valuable possession or quality that a person or organization has
Resource Type (9001)	Resources will often include raw materials, infrastructure, finance, personnel and IT[b].

[a]Cambridge Dictionary.
[b]https://www.iso9001help.co.uk/7.1%20Resources.html.

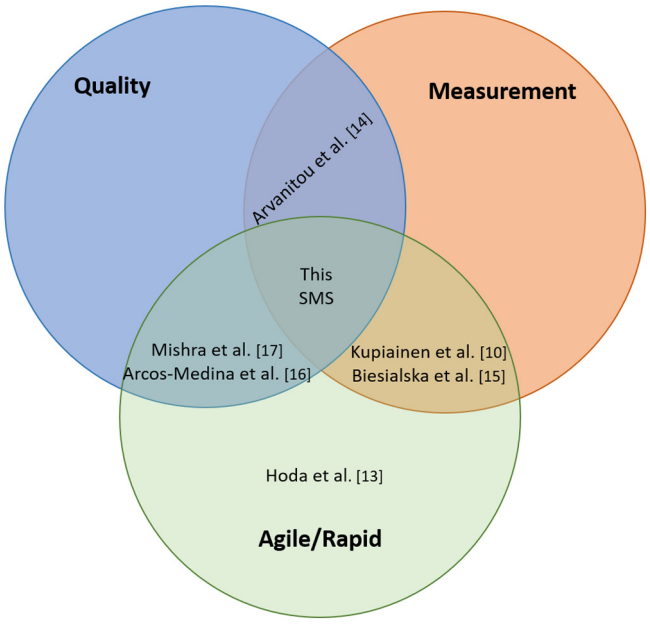


Fig. 1. Specific topic of related works.

- We have observed that many metrics are defined by the authors of the studies on their own, in some cases very similar but with different names. Kupiainen et al. also state that "it appears that practitioners add and invent new metrics according to their needs" (Kupiainen et al., 2015).
- Only Biesialska et al. address quality indicators in the form of Key Performance Indicators (KPIs) as we do (Biesialska et al., 2020). However, their work is framed in the context of the use of big data analytics in agile, their analysis is focused on the used data sources instead of metrics or KPI, which makes the focus of the study different. The rest of related works only report about metrics. In the case of Arvanitou et al. the focus is explicitly set out of KPIs (Arvanitou et al., 2017).
- Data sources identified in our study are included in those identified by Biesialska et al. who cover a broader context in which data is collected, embracing the whole software development process (Biesialska et al., 2020) . We have reported *issues, source code, version control system, test, system at runtime, builds* and *requirements* while they have identified, among others, *user feedback, logs* (including *tests, failures, monitoring, commit logs, pull requests*), *project artifacts* (including *user stories*), *issue reports* (including *experiments*), *source code & data model* and *project execution* (including *version control system*). No other related work provides information about data sources for measurement.
- Quality characteristics emerging in our study (suitability, maintainability, performance, reliability, security) are included in those identified by Arcos-Medina and Mauricio (2019) and mostly included in those reported by Mishra and Abdalhamid (suitability is missing) (Mishra and Abdalhamid, 2018).

From this analysis, we can conclude that, although the literature includes some antecedents that we consider in our work, none of the studies found covers the same scope as ours.

Table 3
Study research questions.

RQ1	How is the research on quality measurement characterized in the context of QR management in ARSD?	
	RQ1.1	What is the annual publication distribution trend?
	RQ1.2	What is the distribution of the venue of publications?
	RQ1.3	What is the distribution of research in terms of research type?
	RQ1.4	What is the authors' affiliation distribution?
	RQ1.5	What application domains are more prominent?
	RQ1.6	What is the level of quality of the primary studies which are empirical?
RQ2	What is the focus of software measurement in the context of QR management in ARSD?	
	RQ2.1	What metrics are reported in the scientific literature in the context of QR management in ARSD?
	RQ2.2	What data sources have been used to collect data to compute these metrics?
	RQ2.3	What tools produce the data needed to compute these metrics?
RQ3	What is the focus of QMIs and which are their relationships to software measurement metrics in the context of QR management in ARSD?	
	RQ3.1	What QMIs are reported in the scientific literature in the context of QR management in ARSD?
	RQ3.2	What tools are used to manage and visualize these QMIs?
	RQ3.3	What metrics have been used to measure QMIs?
	RQ3.4	What entities have been measured by these QMI metrics?
	RQ3.5	What data sources have been used to collect data to compute these QMI metrics?
	RQ3.6	What QRs are used to measure these QMIs?

3. Research method

The goal of this systematic mapping study (SMS) is to provide an overview of the state of the art of quality measurement in ARSD. To achieve this goal, we conducted a systematic mapping (Petersen et al., 2015) following snowballing guidelines (Wohlin, 2014).

3.1. Objective and research questions

We define the main objective of the systematic mapping study, using GQM (Caldiera and Rombach, 1994), as: Analyze *the scientific literature about QR management* for the purpose of *structuring the state of the art* with respect to *quality measurement* from the point of view of *SE researchers and practitioners* in the context of *ARSD*. QR management stands for the identification and classification of QRs, QR metrics and QMI.

In order to achieve this objective, we derive three Research Questions (RQ), decomposed into sub-questions as shown in Table 3. RQ1 focuses on bibliometrics about QRs management in ARSD based on the selected primary studies. RQ2 focuses on software measurement instruments (metrics, data sources, and tools). Finally, RQ3 focuses on QMI and their relationships with software measurement instruments.

Fig. 2 depicts the conceptual model that shapes this SMS, including the main concepts and the research questions introduced above. The two main concepts related to our study are *QR Metric* and *QMI*, representing the corresponding measurement aspects mentioned in the introduction. A *QR* represents the well-known requirements engineering concept of quality requirement (see Table 2), e.g. "The system shall respond to users' request in less than 3 s in the 95% of the interactions". These QRs can be classified under a *QR Type*, e.g., "Performance efficiency" is the type of the former QR, QR types correspond to the ISO/IEC 25010 characteristics (see Section 2.1). Therefore, a QR metric defines how the satisfaction of a QR that belongs to a certain QR type is measured, e.g., using response time metric for the previous example. An example of a QR metric for the requirement above is "Response Time". Related to QMIs, which are general indicators used in the context of quality management, we are also interested in the way these QMIs (e.g., delivery speed) are measured through *QMI metrics* (e.g., integration speed, such as average time to deliver features). The model also establishes what kind of *Data Source* is used to compute both kinds of metrics (QR metric and QMI metric), and which *Software Tool* produces the data. For QMI metrics, we are also interested in the measured *Entity*.

3.2. Search and study selection

This research originates from a previous SMS aimed at synthesizing the state of the art on quality management in the context of ARSD (Behutiye et al., 2020).

While conducting that study, we learned that measurement was particularly relevant and decided to further explore the literature that specifically focuses on measurement in ARSD. This is the goal of the current study. We used the papers found in our previous study with focus on measurement as seed papers. These seed papers are the result of an exhaustive search of scientific studies that contribute to the body of knowledge of QRs in ARSD (i.e., those that discuss different aspects of QRs in the context of ARSD). Therefore, as this study only focuses on the measurement aspect (a particular aspect of the initial study), we consider this set of seed papers to be complete. Then, we repeated the snowballing process that we followed for that study to further explore the measurement part of managing quality in ARSD. Details of this process are given below and in Fig. 3.

3.2.1. Seed papers identification

In order to select the seed papers to start the snowballing process, we followed the next steps: (1) Building an initial set of papers, (2) Formulating the inclusion/exclusion criteria, (3) Piloting inclusion/exclusion criteria on the initial set of papers, and (4) Applying inclusion/exclusion to identify the seed papers. The steps were conducted in the period October 2019–January 2020.

Initial set of papers. We collected the 156 primary studies selected in our previous secondary study (Behutiye et al., 2020), which we used as the initial set of papers.

Piloting inclusion/exclusion criteria. Six researchers piloted the inclusion/exclusion criteria (see below) on 36 randomly selected studies from the 156 studies of the initial set in two rounds following the next protocol for every paper:

1. Check the exclusion criteria. If it is not excluded, proceed to 2.
2. Review the title. If it is clear that the paper addresses at least one of the inclusion criteria, include it; if it does not, exclude it; if unsure, proceed to 3.
3. Review the abstract. If it is clear that the paper addresses at least one of the inclusion criteria, include it; if it does not, exclude it; if still unsure, proceed to 4.
4. Review the paper entirely. If it is clear that the paper addresses at least one of the inclusion criteria include it; if it does not, exclude it.

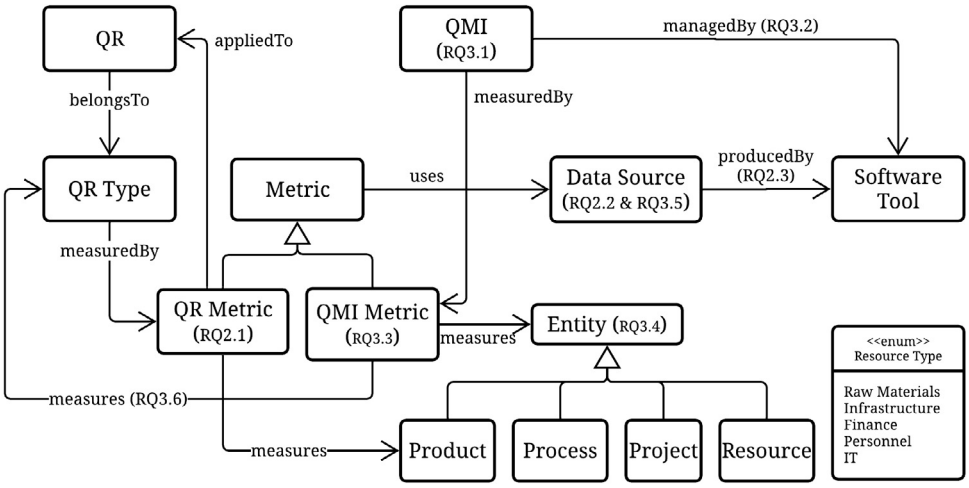


Fig. 2. SMS Conceptual model.

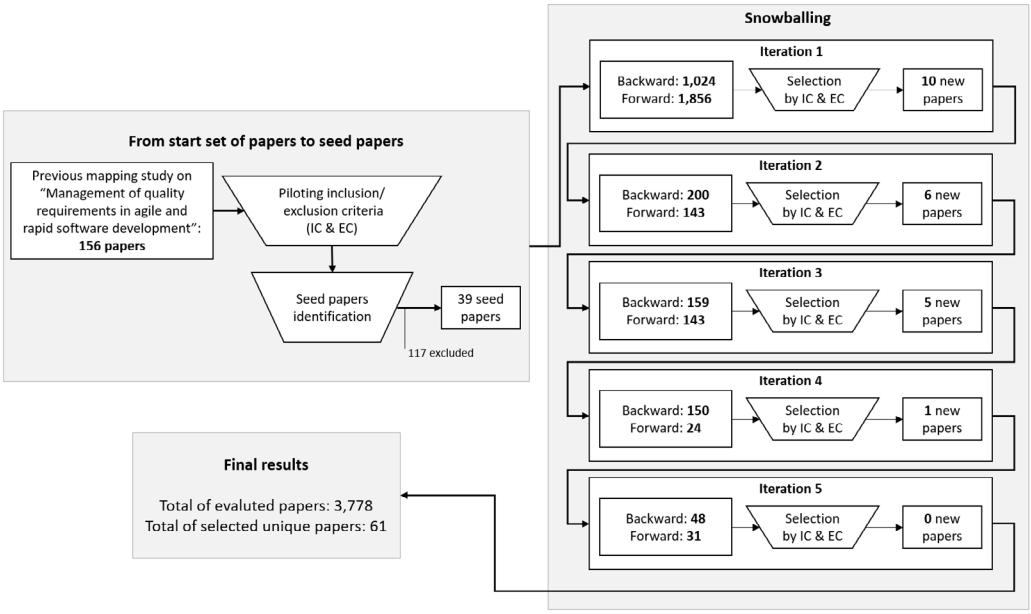


Fig. 3. Overview of the study selection process.

The results of the first piloting revealed that all six researchers agreed on the inclusion of two papers and exclusion of eight papers. However, there were differences in the inclusion decisions for the remaining 26 studies due to discrepancies in understanding the inclusion criteria. We discussed and resolved the issues and updated the inclusion criteria to clarify the misunderstandings. Then, we piloted the inclusion and exclusion criteria again. In this second pilot run, we agreed to include 11 additional studies and excluded the rest. Therefore, there was an agreement on the 36 studies, including 13 studies and rejecting 23 studies, and all researchers considered to understand inclusion and exclusion criteria in the same way:

- Inclusion criteria (*IC*):
  - (a) Studies reporting concrete metrics to measure QR in ARSD
  - (b) Studies reporting QMI and QMI metrics in the context of ARSD
- Exclusion criteria (*EC*):
  - a. Secondary studies
  - b. Studies presenting summaries of conferences
  - c. Non-peer reviewed studies
  - d. Studies not presented in English
  - e. Studies with non-accessible full text after checking with the authors
  - f. Books and grey literature
  - g. Duplicated studies

Seed papers identification The remaining 120 studies were divided into 3 groups of 40 studies each. In each group, two researchers applied the inclusion/exclusion criteria on the 40 studies individually and compared their results. This helped minimize researcher bias as both researchers were comparing their results and resolving disagreements, if any. In cases where the two researchers disagreed on the inclusion of a paper, all researchers reviewed the paper to resolve the disagreement. Only four studies required to be reviewed by all the researchers due to lack of consensus, with the result of including one of them. In total, from these 120 studies, we included 26 studies and excluded 94 studies. Overall, from the starting set of 156 papers, we kept

a total of 39 primary studies after applying the inclusion and exclusion criteria.

Next, we checked the adequacy of the set of papers as seed to conduct snowballing (i.e., minimize the possibilities of missing papers because we applied snowballing instead of automatic searches in databases). To do so, we performed the following quality checks on the group of 39 selected primary studies: (1) Include articles from different clusters not citing each other, and hence cannot be found through citation relationships, (2) Make a seed set which is not too small, (3) Are from different authors, years of publication, and also publisher, not limiting the breadth of the search, (4) Include keywords from the research questions. Since these quality checks were satisfied, we concluded that this group of 39 primary studies was adequate as seed papers to start snowballing.

3.2.2. Snowballing

In this study, we applied Wohlin's snowballing guidelines for systematic reviews (Wohlin, 2014) for searching and identifying primary studies. As argued by Petersen, "in the particular context studied, all relevant studies identified by the search could also be obtained by snowball sampling. The key for success is to define a good start set" (Petersen et al., 2015).

We performed backward and forward snowballing on the 39 seed papers to identify additional primary studies. In order to facilitate the snowballing process and provide visibility of the work in progress, we used a shared spreadsheet where researchers were able to track progress. As we were working in a distributed environment, this helped ensure that there were no duplicate efforts. For instance, researchers avoided working on the same paper simultaneously.

We used the Google scholar citations to perform forward snowballing and the reference list of studies to conduct backward snowballing, as recommended in Wohlin (2014). The snowballing search ran between February 2020 and April 2020. We divided the 39 studies among three researchers to perform the snowballing. In every iteration, one researcher proposed possible papers to be included based on title, and other two researchers checked the inclusion and exclusion criteria to either accept them or reject them. We needed 5 iterations to reach saturation. After checking inclusion and exclusion criteria, we included 22 additional studies, 17 from forward snowballing and 5 from backward snowballing. As a result, we found 61 primary studies (39 seed papers and 22 from snowballing), listed at Appendix A.

3.3. Data extraction

The data was extracted according to a predefined extraction form (see Appendix B) based on our RQs. Each primary study was assigned to a pair of researchers. Each researcher extracted the data from the half of the assigned papers to the pair s/he belongs to, and the result was reviewed by the other researcher of the same pair.

During the data extraction, we dealt with the two main concepts of our study, namely: QR metric (RQ2) and QMI (RQ3). The former, QR metric, refers to the set of metrics measuring a QR type. It also includes basic metrics, i.e., metrics that are computed directly to measure a concrete product quality property. For example, project size, number of bugs or complexity of the code. The latter, QMI, refers to: (1) Non-basic sets of metrics qualified explicitly as "indicators" or "KPIs" by the authors of the study, (2) Metrics measuring more than one QR type together (e.g., maintainability and reliability), and (3) Software quality metrics not measuring a QR type of a product, but rather the process, project, or resources.

Furthermore, a few primary studies included appendices detailing aspects of our data extraction form (e.g., QR metrics such as process performance metrics). In those cases, we have also extracted data from available appendices.

3.4. Quality assessment

Even if quality assessment is not mandatory in systematic mappings, we assessed the rigor and relevance of the empirical primary studies. We adapted Ivarsson and Gorschek's industrial assessment model (Ivarsson and Gorschek, 2011) to assess the quality of the studies presenting empirical results. Concretely, this model defines some aspects to be evaluated to assess the research rigor and industrial relevance (see Table 4). Research rigor evaluates three aspects: description of the context, description of the study design, and discussion of validity threats. Industrial relevance evaluates four aspects: representativeness of the subjects of the study, relevancy of the context, realistic size of subjects and experimental research method.

Following the rubrics included in the model, each rigor aspect can be rated by 1 (strong description), 0.5 (medium description), or 0 (weak description), and each relevance aspect can be rated by 1 (do contribute to relevance) or 0 (do not contribute to relevance). We assign the values for rigor and relevance as the sum of the rating of their aspects.

The quality assessment data has been extracted during the data extraction following the same protocol and the same extraction form (see Section 3.3).

3.5. Data analysis

For the data analysis, we combined quantitative and qualitative techniques. For quantitative results, we applied frequency and correlation analysis; for qualitative results, we applied coding and classification. We applied these techniques depending on the research question:

- *Frequency analysis*. This technique is used for providing quantitative results related to the total number of papers. We applied this technique to RQ1 (research characterization), RQ2 (reported quality factors), and RQ3 (reported QMIs).
- *Correlation analysis*. We applied correlation to RQ1 for the analysis of the primary studies quality (rigor and relevance properties), RQ2 for analyzing the relation between data sources and QRs, and RQ3 to analyze the relation between entities and data sources with QMIs.
- *Coding and classification*. This technique allows us to identify categories. We used deductive coding for domains in RQ1, QRs types in RQ2, and entities in RQ3; and inductive coding for data sources in RQ2 and RQ3. The first six authors performed this coding and classification.

For the deductive coding of domains (RQ1), we used the domain taxonomy in the Industry Classification Benchmark (ICB) (v3.4, February 2020) (Russell, 2017). ICB provides a detailed structure of classifying industries and their sectors. We chose the super sector level in the taxonomy (the second level in the hierarchy, as the first one is too general and lower ones would cause great dispersion) as the set of domain values to assign to the studies. For the deductive coding of QRs (RQ2), we used the software product quality model included in the ISO/IEC 25010 (Anon, 2011). We chose the quality model characteristics as the set of QR types to assign to the studies. For deducting coding of entities, we used the resource type definition included in Section 2.1 (see Table 2). For the inductive coding of data sources (RQ2, RQ3) and QMIs (RQ3), the codes arose directly from the studies. We mainly used the metric names and the context of the paper to specify corresponding data sources and entities because this information was not provided by most of the primary studies.

For each QMI metric, we identified the measured entity type (i.e., process, product, project, or resources). During the entity classification, we found some problems distinguishing entities. Therefore, we followed the next rules:

Table 4
Primary studies quality assessment attributes (Ivarsson and Gorschek, 2011).

Attribute	*Aspect*	*Description*
Rigor	Context described	It is considered strongly described when the reader can understand and compare it to others.
	Study design described	It is considered strongly described when the reader can fully understand the protocol, e.g., measured variables, treatments, sampling, etc.
	Validity discussed	It is considered strongly described when the validity of the evaluation discussion includes threats and the measures taken to mitigate them.
Relevance	Subjects	It is considered relevant when the subjects are representative of the intended users of the proposal.
	Context	It is considered relevant when the study has been performed in a representative setting of the usage of the proposal.
	Scale	It is considered relevant when the evaluation has been performed in a realistic size of subjects.
	Research method	There was a list of relevant (action research, lessons learned, case study, field study, interfere, descriptive/exploratory survey) and non-relevant (conceptual/mathematical analysis, laboratory experiment, other) methods.

- *Process*: Metrics related to the time used for performing some activities.
- *Personnel resources*: Metrics related to the effort used to perform some activities.
- *Project*:
  - Metrics for QMIs that include the word project, e.g., project success, project cost.
  - Metrics related to the quality of the product when it is already released, e.g., bugs delivered to the customer, post-release quality.
  - Metrics measuring if the product is ready to be released, e.g., on-time delivery, product readiness.
  - Metrics measuring time deviations, that implies a period of time, e.g., release delay, relative schedule deviation.

3.6. Threats to validity

We applied various mitigations to minimize the construct, internal, external and conclusion validity threats which are common in secondary studies, as defined in Ampatzoglou et al. (2019) and Zhou et al. (2016).

Construct validity entails identifying and applying appropriate operational measures for the concepts under study. We applied an SMS protocol with research objective, research questions, search method, study selection, data extraction, analysis, which was reviewed by researchers to guide the SMS, and summarized in this Section 3. This helped mitigate threats that could arise from imprecise description of the SMS setting.

Internal validity involves determining a causal relationship between factors within the context of a given study. We applied snowballing search to retrieve as many relevant primary studies as possible. In snowballing search, a potential threat comes from difficulties in identifying a good start set of papers (i.e., relevant and adequate number of starting set of papers), which has been claimed to be problematic for systematic reviews using snowballing search strategy (Wohlin, 2014). We minimized this threat by using an existing secondary study in QR management in ARSD to identify the initial set of papers (Behutiye et al., 2020) and by the aforementioned quality checks over the set of seed papers. Another threat to internal validity comes from study selection bias. We piloted the inclusion/exclusion criteria on 36 studies with six researchers to mitigate this threat. This helped to clarify differences and build a common understanding of the inclusion/exclusion criteria. Additionally, we performed researcher triangulation in order to minimize researcher bias when selecting the primary studies and performing the data analysis.

External validity defines the applicability of the findings of a study to other contexts. We defined the context of each primary study following the ICB domain classification, mainly from technology and telecommunication domains. In the discussions, as implications for SE practitioners, we offer a starting point for companies, which is only applicable depending on the data and context they have.

Conclusion validity shows the extent to which the procedures of a study are repeatable with the same result. We applied an SMS protocol, reviewed by all researchers, to guide the study. We mitigated the conclusion validity threat that may arise from data extraction bias by piloting the data extraction in two rounds with the six researchers who participated in the data extraction phase. We also performed a review process in which one researcher reviews the data extracted by another researcher. This helps minimize errors that can happen in the data extraction spreadsheet and possible bias of the researchers in the extracted data and its interpretation.

4. Results and analysis

In this section, we present the results for each of our RQs with an emphasis on visual maps and graphs, as is recommended for SMSs by Petersen et al. (2015). Section 4.1 provides an overview of the research on the topic. Section 4.2 presents the findings regarding QR metrics, and Section 4.3 presents the findings regarding QMI and QMI metrics. Appendix A includes the complete list of primary studies; the extracted data and the complete list of QR and QMI metrics is publicly available at López et al. (2021).

4.1. Overview of research on quality measurement in the context of QR management in ARSD (RQ1)

In this section we report bibliometrics characteristics and the results from the quality assessment.

4.1.1. RQ1.1 - What is the annual publication distribution trend?

The 61 selected primary studies were published between 2004 and 2019 with the only exception of one paper in 2001. Fig. 4 shows a moderate ascending trend from 2004 in four cycles (2005–2008, 2009–2011, 2012–2015, and 2016–2019), with 2014 as the year in which the field experienced a significant growth: 33 primary studies (54.1%) have been published in the six-year period from 2014 to 2019.

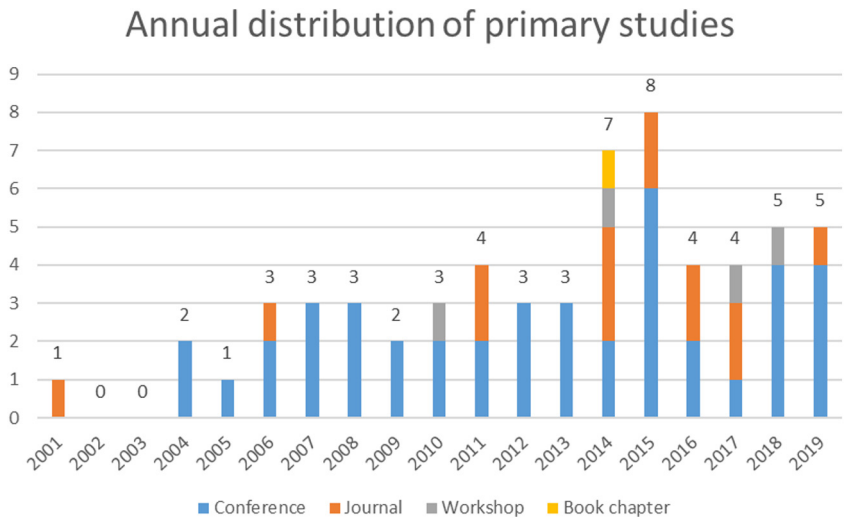


Fig. 4. Annual distribution of primary studies.

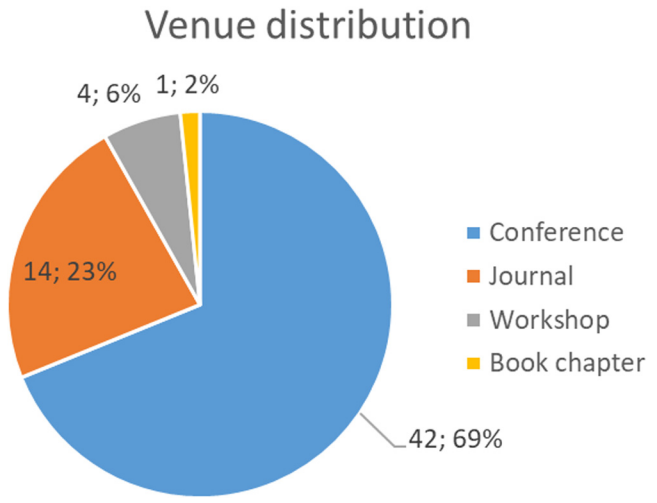


Fig. 5. Venue type distribution.

4.1.2. RQ1.2 - What is the distribution of the venue of publications?

The venues considered in this SMS are peer-reviewed (including journals, conferences, workshops, and book chapters). The distribution shows that conferences stand out as the main venue for primary studies in the area (see Fig. 5). However, the temporal distribution of the venues (depicted in Fig. 4) shows that a majority of journal papers (10 out of 14, i.e., 71.4%) were published in the last six years covered by our SMS (2014–2019).

The 61 primary studies have been published in 48 different venues. Looked the other way round, most of the venues (39 venues; 81.2%) only published one study. The most targeted venue is the *International Conference on Agile Software Development* (AGILE; five papers), which concentrates four papers but most published long ago, in the period 2004 to 2008 (the first cycle identified in the annual distribution in the previous subsection). All the venues publishing more than one paper are shown in Table 5.

4.1.3. RQ1.3 - What is the distribution of research in terms of type?

Fig. 6, left, shows how empirical studies are the prevalent type of research paper, dominant over experience reports and theoretical papers. Analyzing the 54 empirical studies in detail (Fig. 6, right), we see that more than half of these papers used case studies as empirical method, while surveys, experiments and action research were used in a similar share. The mixed study applies case study and experiment. It is worth mentioning that three of the primary studies applying surveys use both questionnaires and interviews, three only questionnaires, and two only interviews. From the studies classified as *other*, one is based on surveys (questionnaires and interviews) and observations, one on interviews and a workshop, one on a workshop, one on quantitative analysis of real-life project data, and one on observations.

4.1.4. RQ1.4 - What is the authors' affiliation distribution?

We analyzed the authors' affiliation to further understand the nature of the involved stakeholders. For each primary study, we extracted the affiliation information of all the authors and the involved companies. Fig. 7, left, reports the frequency of academia (including both universities and research institutes) and industry conducting research in this field. It is remarkable that 44% of the primary studies includes authors from industry (summing up Industry and Collaboration types), although only five papers have industry-only authors [PS11][PS20] [PS30] [PS41] [PS60]. Industry authors belonged to 25 companies. Fig. 7 (right), lists the companies that appear in more than one paper.[5]

Authors' organizations belong to 30 countries. Fig. 8 paints the countries in shades of blue according to the number of primary studies, and shows the number of primary studies (black bubbles.[6]) and number of primary studies including authors from industry (orange bubbles[7]) per continent (North and South America, Europe, and Asia) The figure shows the three countries with the highest number of primary studies (USA, Germany, and Sweden) and the concentration of research in Europe, with participation in 54.1% over the total of 61 primary studies, growing to 62.9% over the 27 papers with authors from industry.

4.1.5. Q1.5 - What application domains are more prominent?

Out of the 61 primary studies, 46 (75.4%) reported information related to the application domain. The most recurring domains

[5] In this figure, we count the number of papers, not the number of authors.

[6] Since [PS10] [PS26] [PS34] [PS56] have authors belonging to more than one continent, the sum of black bubbles is 65 instead of 61. Europe and Asia black bubbles do not correspond to the sum of their countries because there are papers with authors of different countries in the same continent.

[7] Since [PS26] is a collaboration between the USA and Japan, the sum of orange bubbles is 28.

Table 5
Venues publishing more than one primary study.

Venue	*#Papers*	*Years*
International Conference on Agile Software Development (AGILE)	5	2004–2006, 2008, 2012
International Conference on in Software Engineering (ICSE)	2	2006, 2007
International Symposium on Empirical Software Engineering and Measurement (ESEM)	2	2010, 2019
Information and Software Technology Journal (IST)	2	2011, 2014
Joint Conference of the International Workshop on Software Measurement and the International Conference on Software Process and Product Measurement (IWSM Mensura)	2	2013, 2017
International Journal of Secure Software Engineering (IJSSE)	2	2014, 2017
International Conference in Availability, Reliability and Security (ARES)	2	2015, 2016
Journal of Systems and Software (JSS)	2	2015, 2017
Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering (ESEC/FSE)	2	2015, 2019

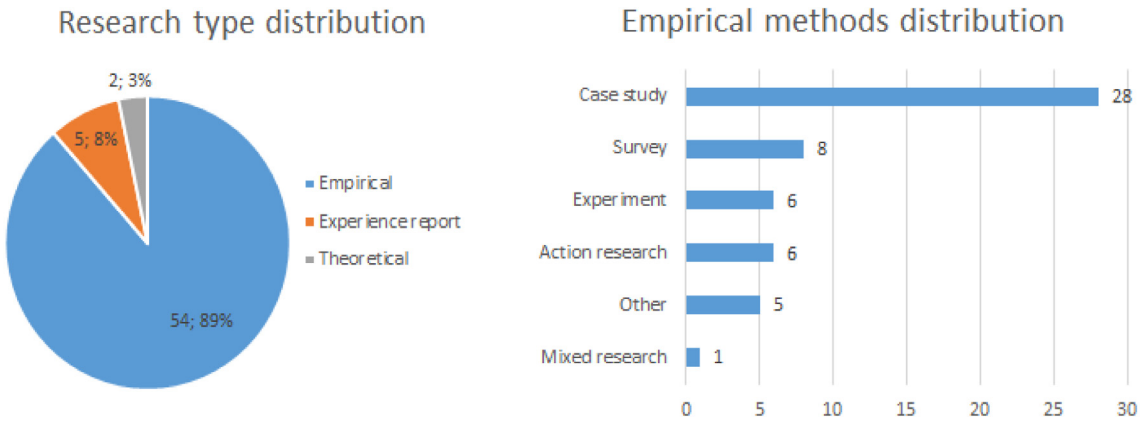


Fig. 6. Research type distribution.

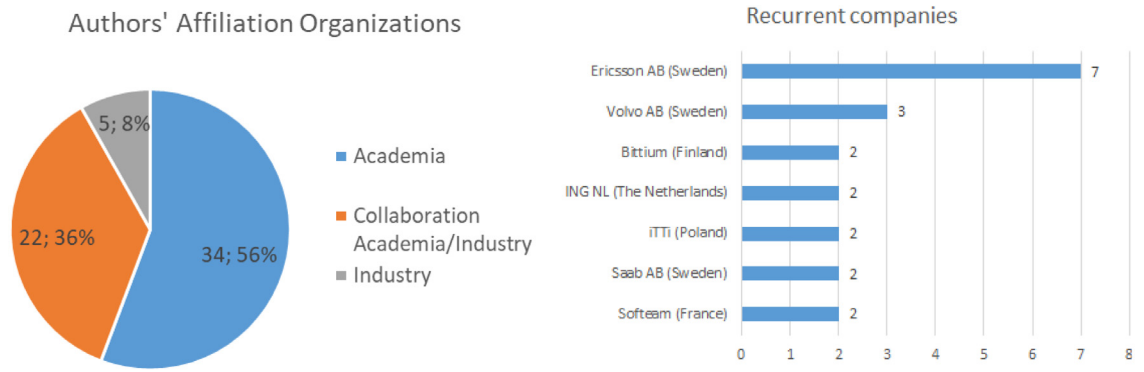


Fig. 7. Authors' affiliation organization.

are *technology* (code 1010; 24 primary studies) and *telecommunications* (1510; 13), followed by *Industrial Goods and Services* (5020; 3), *Banks* (3010; 2), *Consumer Products and Services* (4020; 2), *Financial Services* (3020; 1), and Health *Care* (2010; 1).

4.1.6. RQ1.6 - What is the level of quality of the empirical primary studies?

Fig. 9 maps the industrial relevance and research rigor of the 54 empirical primary studies. The bubble plot combines the industrial relevance and rigor, and the bubble size represents the number of primary studies corresponding to the concrete value for both attributes.

Analyzing industrial relevance, more than half of the assessed studies obtained the highest rate (i.e., relevance = 4; 34 studies, 63.0%), and this number grows until 43 studies (79.6%) when we consider studies with positive industrial relevance (i.e., relevance > 2). For research rigor, the rates are more distributed, but still a significant number of studies obtained positive rates (i.e., rigor > 1.5; 35 studies, 64.8%). The normalization of the average values into the interval [0,1] also shows how relevance (normalized average = 0.83) is better than rigor (normalized average = 0.67).

In order to show a global picture of the quality, Fig. 9 distributes the studies into five areas. The largest share of studies (30 studies, 55.6%) have both aspects positive (area A in the figure), while the converse case (both aspects negative) occurs only in two studies (3.7%; area E). On the other hand, the number of studies with one positive aspect and the other neutral is nine (16.7%; area B), while there are 10 studies with one negative aspect (18.5; area D). The remaining three studies (5.6%; area C) have both aspects neutral. The combined normalized averages of the two aspects is 0.75, both combined with arithmetic mean and geometric mean, indicating an overall good rate for relevance and rigor, despite a significant number of studies (15, 27.8%) are located in areas C, D and E.

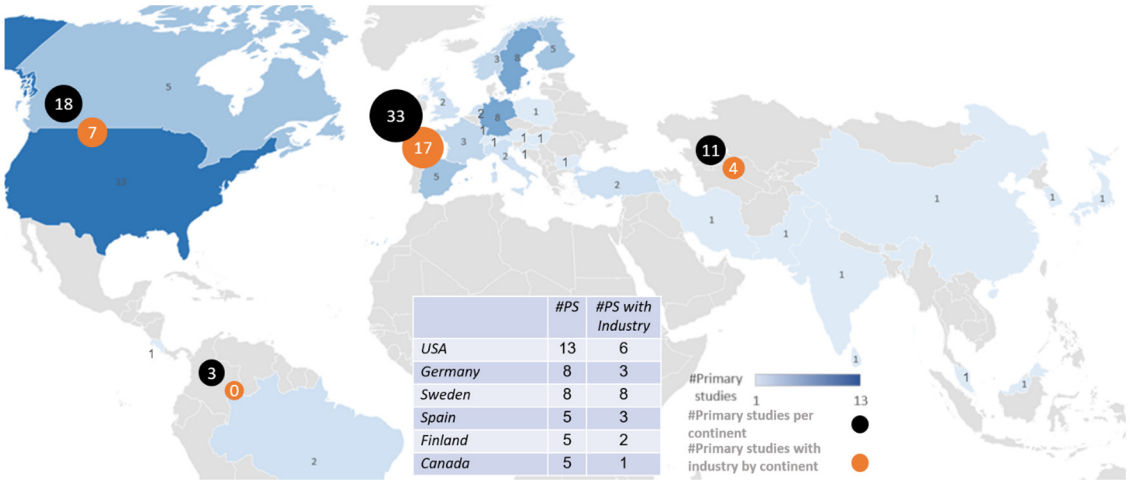


Fig. 8. Authors' affiliation country and continent. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

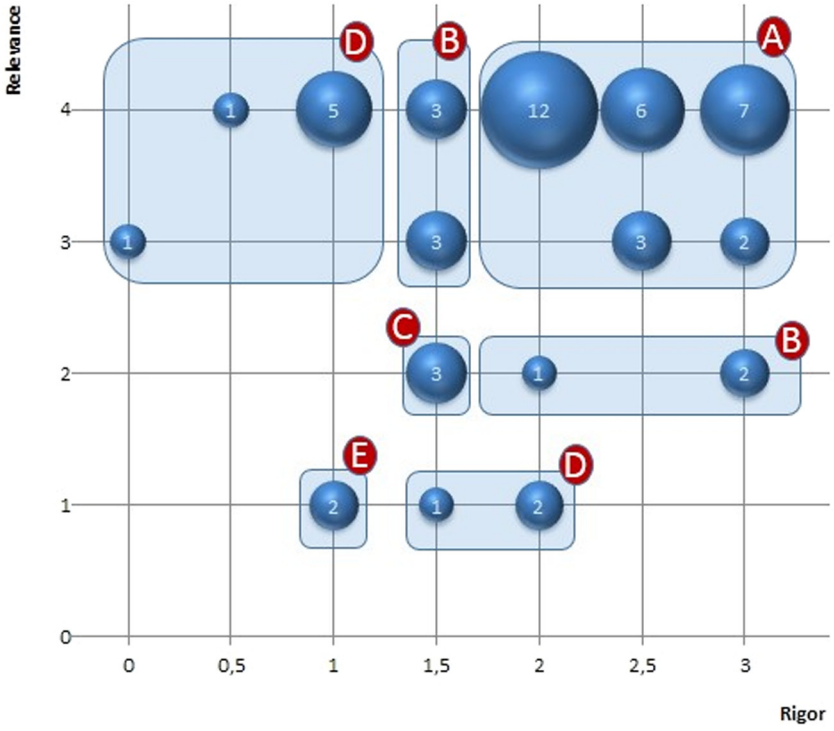


Fig. 9. Studies quality (research rigor and industrial relevance).

We also looked into the aspects that contribute to each attribute. We discovered that weak validation is the root cause for the lower score of rigor. Summing up the score of each of the 54 empirical studies on each aspect, we get 23 for *validity* while we get 43.5 for *context*, 41.5 for *design*, 43 for *subject*, 45 for *context*, 47 for *scale* and 45 for *method*. As long as the maximum score for each aspect is 1, these sums can achieve a maximum value of 54, which shows clearly that validity is the only aspect with a poor assessment.

4.2. Quality requirements measurement in the context of QR management in ARSD (RQ2)

We found QR metrics in 28 primary studies, which corresponds to 45.9% of the total of 61 primary studies. This section provides information about what measurable QRs have the focus of attention in the literature and how they are measured for software quality management in the context of ARSD.

4.2.1. RQ2.1 - What QR metrics are reported in the scientific literature in the context of ARSD?

The 28 primary studies reported a total of 107 QR metrics; the complete list is included in the public dataset (López et al., 2021).

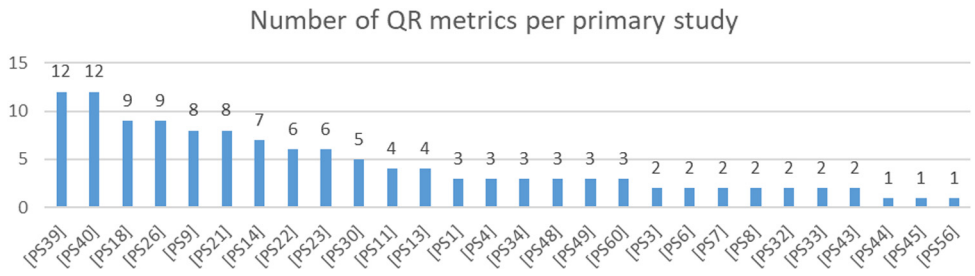


Fig. 10. The number of metrics measuring QRs per primary study.

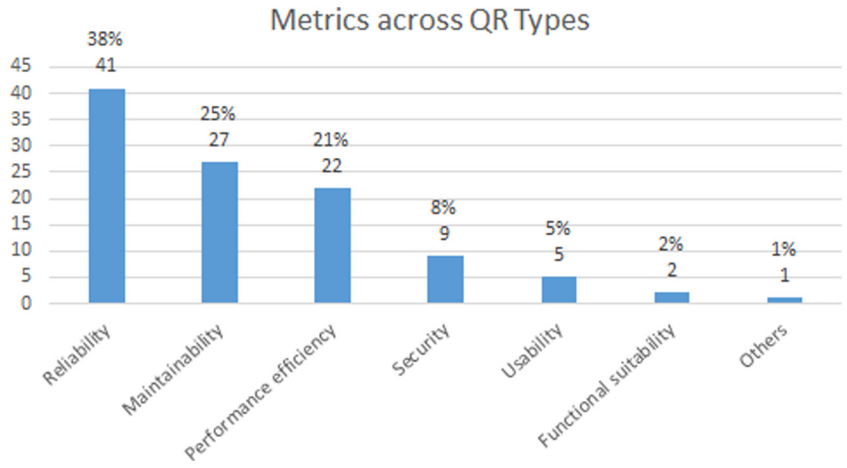


Fig. 11. Number of QR metrics based on the mapping onto ISO 25010 characteristics.

Most of the metrics (91 out of 107) are reported only once, with only 16 metrics reported in more than one study. More precisely, *Code duplication* and *Probability of successful attack* are reported in three studies and the other 14 metrics (*Complexity*, *Comments*, *Non-blocking files*, *Passed tests*, *Fast tests*, *Bug density*, *Weighted methods of a class*, *Lack of cohesion in methods*, *Impact of attack*, *Memory usage*, *Number of vulnerabilities*, *Test coverage*, *Number of transactions per second* and *Throughput*) are reported twice. Out of the primary studies, Martínez et al. is most active in reporting metrics (two times 12 metrics [PS39] [PS40]) as illustrated in Fig. 10, i.e., they report nearly a fifth of our identified occurrences of QR metrics in all primary studies (24 out of 125).

We applied deductive coding using as predefined codes the eight quality characteristics defined in ISO 25010 (see Table 1, Section 2.1), Fig. 11 shows the number of metrics per each QR type, the complete mapping is included in the public dataset (López et al., 2021). A first observation is that all except one QR metrics are mapped to six quality characteristics, with *reliability* having the highest number of reported metrics. Examples of *reliability* metrics include *passed tests*, *bug density*, *number of tested classes*, and *mean time to failure*. The other two dominant characteristics are: *maintainability*, with metrics such as *complexity*, *tight class cohesion*, *coupling between objects* and *weighted methods of a class*; and *performance efficiency*, including metrics such as *memory usage*, *throughput*, *response time* and *operational performance*.

We identified one paper by Jinzenji et al. that explicitly reported the metric *lines of code* in relation with the QR type *reliability* [PS26]. However, we mapped *lines of code* reported by Concas et al. [PS14] and similar *total LOC*, reported by Bakota et al. [PS4], to the QR type *maintainability* according to the definition in ISO 25010. For us it is difficult to mark both *lines of code* metrics as duplicates since the paper by Jinzenji et al. does not provide additional information. To avoid biasing our results, we mapped the metric by Jinzenji et al. to the *others* category of the QR types and did not count it as a duplicated metric.

4.2.2. RQ2.2 - What data sources have been used to collect data to compute these metrics?

We obtained the following types of data sources associated with QR metrics: *source code, system (at runtime), issues, tests, version control system, and builds*.

As Fig. 12 shows, *source code* is the most used data source, appearing in 37 QR metrics. Examples of metrics using source code are *complexity*, *comments*, and *coupling between objects*. Other relevant data sources are: (i) *system (at runtime)* appearing in 30 QR metrics, used for measuring e.g., *average CPU usage*, *memory usage*, *mean time to failure*, and *number of transactions per second*; (ii) *issues* appearing in 22 QR metrics, used for measuring *list of open defects*, *critical issues ratio*, and *number of usability issues met by user*, among others; (iii) *tests* appearing in 17 QR metrics, used for measuring for instance *unit tests*, *passed tests*, and *fast tests*.

It is worth to mention that 14 out of the 34 usages of *source code* were in combination with other data sources (eight with tests, six with issues); for instance, *defect density (released defects/KLOEC)* combines source code and issues.

Moreover, we cross analyzed these data sources with the reported QRs by making use of our mapping to the QR characteristics specified in ISO 25010 (see results of RQ2.1, QR types). The corresponding result is provided in Fig. 13. For example, *reliability* metrics make use of *issues*, *tests, system (at runtime)*, *source code*, and *builds*. However, none of the *reliability* metrics are computed based on the version control system (VCS). Looking at the data sources, it appears that every data source except *builds* is primarily used in QR metrics of a particular type. For instance, all except one of the 22 QR metrics measuring *performance efficiency* need the data source *system (at runtime)*. It is worth remarking that we found 10 QR metrics whose data source cannot be identified; we

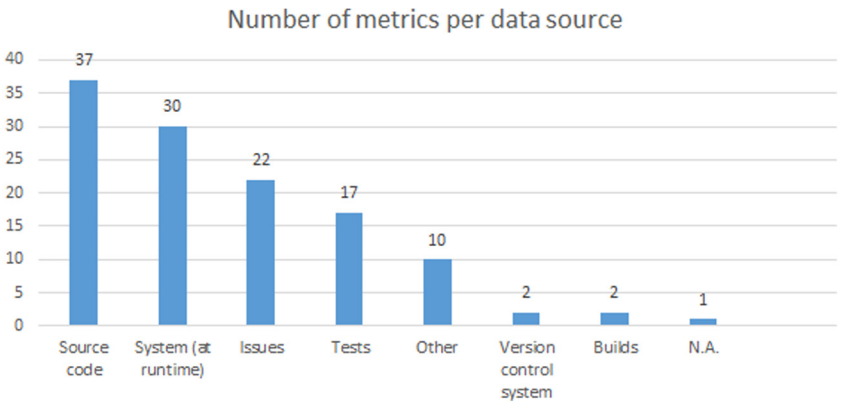


Fig. 12. Number of QR metrics per data sources.

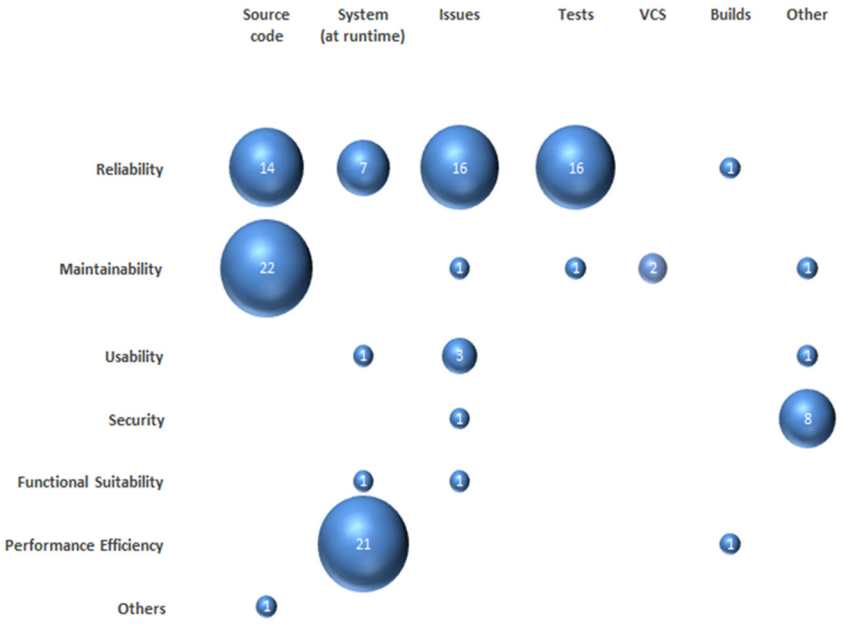


Fig. 13. Data sources used for measuring QR Types.

classified them as *other*. The cross analysis shows that eight out of these 10 metrics are security metrics.

4.2.3. RQ2.3 - What tools are reported together with the metrics measuring QRs?

Only three primary studies (out of the 28 studies that propose metrics for QRs) provide information about tools regarding their reported QR metrics. In total, these primary studies reported 12 different tools together with metrics measuring either reliability, maintainability or functional suitability. Jira is reported by all of the three primary studies whereas Gerrit, Nagios, SVN, and Zabbix are reported just by one primary study. Table 6 provides the whole mapping with details on the corresponding metrics and the data sources.

4.3. Quality management indicators measurement in the context of QR management in ARSD (RQ3)

We found QMIs in 46 primary studies, which corresponds to 73.8% of the total of 61 primary studies. This section provides information about what QMIs have the focus of attention and how they are assessed for software quality management in the context of ARSD.

4.3.1. RQ3.1 - What are the reported QMIs in the scientific literature in the context of QR management in ARSD?

We collected 86 different QMI from the 46 primary studies and classified them into 10 generic QMIs, as shown in Fig. 14. The frequency distribution shows two prevalent QMIs (*Product Quality* appearing in 65.2% of the studies, and *Productivity* in 39.1%) and other eight QMIs mentioned by nine primary studies (19.6%) at most. We use the 10 generic QMIs to present the analysis for this RQ. Research dataset includes the categorized QMIs (López et al., 2021).

Fig. 15 shows the number of QMIs identified in each primary study. A similar number of studies introduce one single indicator (19 studies; 41.3%) and two (21; 45.7%), with the rest of studies (6; 13%) defining up to six QMIs.

The analysis of QMI per primary study (Fig. 15) shows that QMIs are often not reported in isolation: 29 primary studies, out of the 46 reporting QMIs (63.0%), report more than one QMI. In order to provide a better understanding about the synergies among QMIs, we analyzed the pairs of QMIs that are reported together (see Table 7), and the number of times that the QMIs are reported alone (Table 7, diagonal). The last row shows the total number of QMIs reported in the same paper per each QMI. We can observe that most QMIs are reported together with others

Table 6
Overview of tools reported together with metrics measuring QR Types.

Tools (# metrics)	*# Primary studies*	*Data sources*	*Metrics*	*QR Types*
Jira (5)	3	tests source code & issues issues	unit test coverage for the developed code bug density, non-bug density open defect severity index well defined issues	Reliability Reliability Reliability Maintainability
CodeSonar (1)	2	source code	non-blocking files	Maintainability
Coverity (1)	2	source code	non-blocking files	Maintainability
GitLab (5)	2	tests source code & issues issues	passed tests, fast tests bug density, non-bug density well defined issues	Reliability Reliability Maintainability
Jenkins (4)	2	tests builds	passed tests, fast tests, test coverage (testing status) build stability	Reliability Reliability
Mantis (6)	2	source code & issues issues	bug density, non-bug density postponed issues ratio, critical issues ratio well defined issues end user feedback	Reliability Reliability Maintainability Functional suitability
Redmine (3)	2	source code & issues issues	bug density, non-bug density well defined issues	Reliability Maintainability
SonarQube (5)	2	source code tests	complexity, comments, duplication, non-blocking files test coverage (testing status)	Maintainability Reliability
Gerrit (1)	1	version control system	highly changed files	Maintainability
Greenhopper (2)	1	tests issues	unit test coverage for the developed code open defect severity index	Reliability Reliability
Nagios (1)	1	system (at runtime)	availability uptime	Reliability
SVN (1)	1	version control system	highly changed files	Maintainability
Zabbix (1)	1	system (at runtime)	availability uptime	Reliability

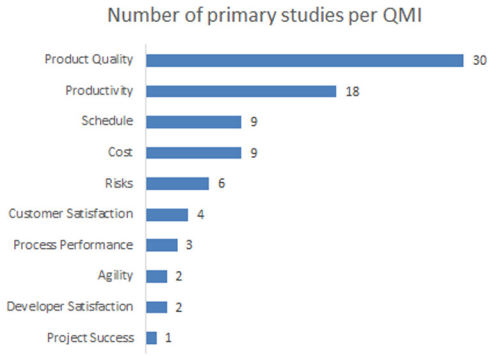


Fig. 14. The number of QMIs per primary study.

and in the extreme case, three of them (*Customer Satisfaction*, *Developer Satisfaction* and *Process Performance*) are always reported with some other QMI. Only *Product Quality* is defined alone in a high number of studies (cf. value eight in the diagonal) although percentual speaking, *Project Success* (one out of one) and *Agility* (one out of two) have a higher rate of individual definition. It is worth noticing how the two most recurrent QMIs, namely *Product Quality* and *Productivity*, are also frequently introduced together, as much as 15 times, which in the case of *Productivity* represents 83.3% of the total number of occurrences.

4.3.2. RQ3.2 - What tools are used to manage and/or visualize these QMIs?

We were also interested in tools used for QMIs management, i.e. tools that can be used to monitor the QMIs, presenting the QMI metric results. From the 46 papers reporting QMIs, 12 papers mentioned tools (26%): Q-Rapids Tool [PS37] [PS39] [PS40] [PS59], MS Vista Gadget [PS50] [PS51] [PS52], Dashing Tool [PS41], Failure Modes and Effects Analysis (FMEA) [PS55], SoReady [PS53], and some papers including visualizations but not as part of a tool, classified as *ad-hoc dashboard* [PS1] [PS9] [PS51].

Most of these studies use a single tool to visualize QMIs, except [PS51], which uses MS Vista Gadget and an ad-hoc dashboard.

4.3.3. RQ3.3 - What metrics have been used to measure QMIs?

Not all the 46 primary studies reporting QMIs are reporting QMI metrics. A total of 38 primary studies reported 223 QMI metrics, including descriptive measurement (e.g., cost as money, time, and effort), or concrete formulas (e.g., number of lines of code, cost computed as `((real cost - planned cost)/planned cost) * 100`)). Fig. 16 shows the number of metrics proposed in the primary studies reporting QMI metrics (blue series) and the number of QMIs that these metrics are measuring (red series). It is remarkable to mention that eight primary studies (21.1%) are reporting 10 or more metrics; altogether, they report more than half of the reported metrics (57%). The number of measured QMIs per study reveals that most of the studies are devoted to measuring concrete QMIs, 36 primary studies (94.7%) are measuring one or two QMIs. The complete list of metrics per each QMI is included in the public dataset (López et al., 2021)

Most QMI metrics (199, 89.2%) are reported only once. We found that authors measuring the same quality characteristic use slightly different metrics. For example, for measuring the number of bugs, the primary studies report metrics such as *Number of reported bugs* (total number of bugs), *Bug ratio* (number of open bugs/total number of issues), and *Number of post-release bugs* (total number of bugs reported after the release).

Fig. 17 shows the number of papers mentioning the same metric, i.e., metrics reported in more than one paper, together with the QMIs that they measure. In contrast to RQ2, in which we observed that QR metrics are devoted to measure always QRs belonging to the same quality characteristic, the same QMI metric can measure different QMIs. For example, *Number of open defects*, *Size (LOC)* and *Spent bug fixing effort ratio* are used to measure *Product Quality* and *Productivity*, *Timely feature specification delivery* and *Non-issue component commits* measure *Process Performance* and *Schedule*; and *Fast test builds* measures *Process Performance* and *Product Quality*.

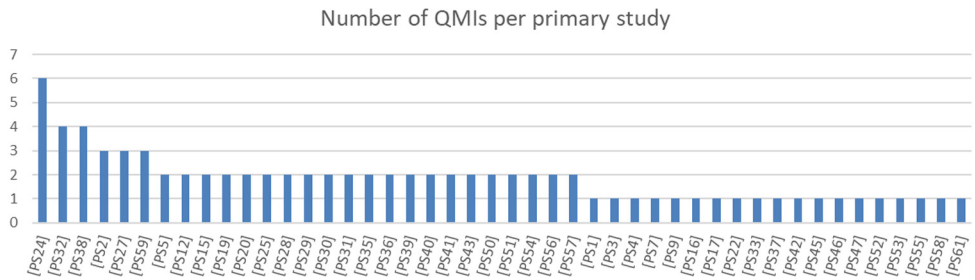


Fig. 15. Number of QMIs per primary study.

Table 7
Pairs of QMIs reported jointly (reported alone in the diagonal).

	Agility	*Cost*	*Customer Satisfaction*	*Developer Satisfaction*	*Process Performance*	*Product Quality*	*Productivity*	*Project Success*	*Risk*	*Schedule*
Agility	*1*	1					1			
Cost	1	*3*	1	1		2	3		1	3
Customer Satisfaction		1	***0***	2		3	3			3
Developer Satisfaction		1	2	***0***		2	2			1
Process Performance					***0***	3	1			
Product Quality		2	3	2	3	***8***	**15**		1	**5**
Productivity	1	3	3	2	1	**15**	*1*			2
Project Success								*1*		
Risks		1				1			*1*	1
Schedule		3	3	1	0	**5**	2		1	1
Number of related QMIs	*2*	*7*	*5*	*5*	*2*	*7*	*7*	*0*	*3*	***6***

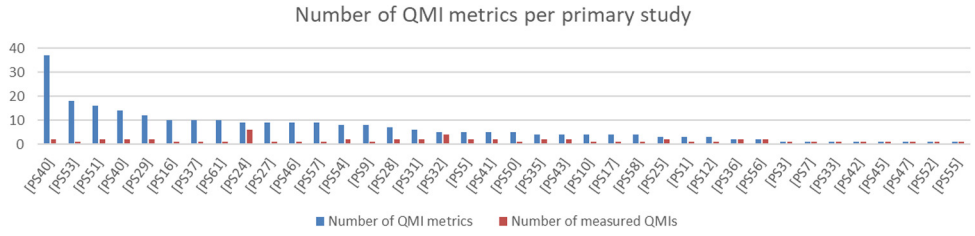


Fig. 16. The number of QMI metrics and QMIs per primary study. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

To complete the results related to this sub-RQ, Fig. 18 shows the number of metrics reported per QMI (blue series) combined with the number of papers mentioning QMIs (red series) and the number of papers measuring the QMIs (green series). The most measured QMI is *Product Quality* with half of the measures (112 metrics; 50.2%) and near two thirds of the primary papers (30 studies, 65.2%).

4.3.4. RQ3.4 - What entities have been measured by these QMI metrics?

As QMIs can be used to monitor different organization aspects, we analyzed the entity measured by each metric. An entity can be classified as process, *product*, *project* or *resource* (see Section 2.1).

Fig. 19 (left) shows the entities measured by QMI metrics. The majority of the metrics focus on the *product* (121 measures out of 223), measuring different software product properties (e.g., size, complexity, test coverage). Next, we find: *process*, measuring mainly time spent (e.g., bug correction time, commit review duration, and timely feature delivery); *personnel resource*, measuring

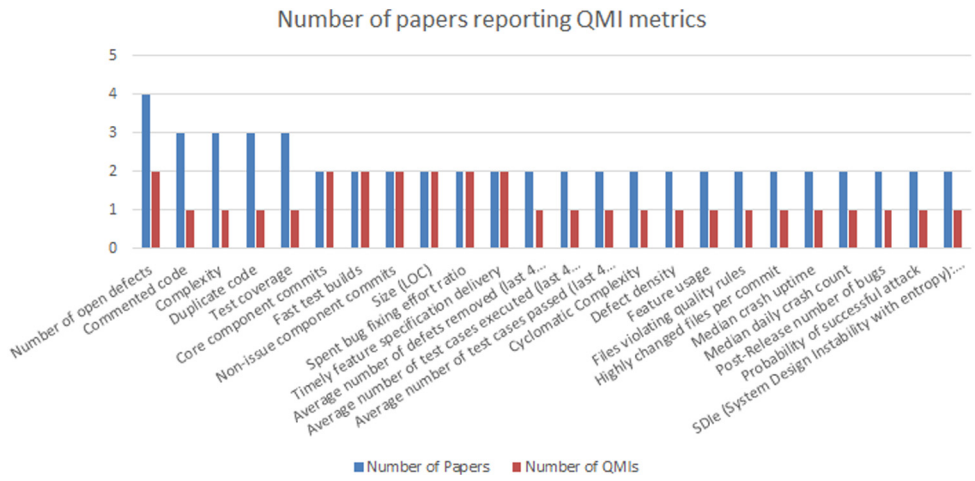


Fig. 17. Number of papers reporting QMI metrics.

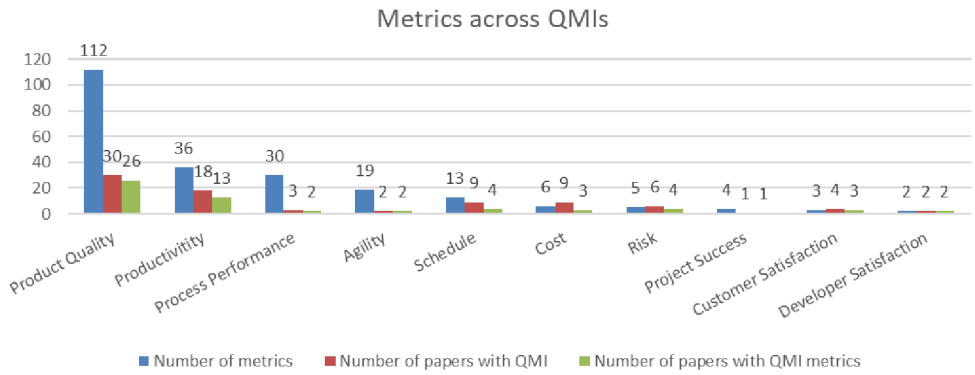


Fig. 18. Number of metrics and primary studies per QMI. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

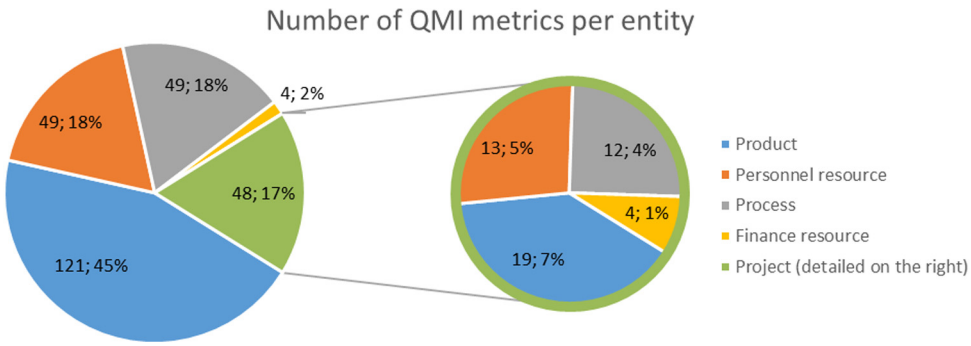


Fig. 19. QMI measured entities frequency.

effort spent on concrete activities (e.g., testing effort, LOC per person-month); and *project*. The least measured entity is *finance resource*, none of the papers include details about how to measure them. Our analysis shows that all the project metrics are also measuring another entity, as shown in the detailed chart on the right in Fig. 19.

In order to understand the impact of these entities in the QMIs, we analyzed which entities are measured for each QMI (see Fig. 20). Most of the QMI metrics have a type of entity as the main target, i.e., with at least half of their metrics applied to such type. Not surprisingly, *Product quality* metrics measure fundamentally *Product* entities (in 76.6% of the cases), *Productivity* metrics apply mainly to *Personnel resource* (56.4%) and *Schedule* metrics to *Project* entities (50.0%). On the other hand, not all *Process performance* metrics apply to *Process* entities (only 39.4% of occurrences). The most diverse is Agility, with similar numbers for *Process*, *Project* and *Personnel resource* entities.

4.3.5. RQ3.5 - What data sources have been used to collect data to compute these metrics?

As done in RQ2.2., we investigated the data sources that were used to compute each QMI metric. Similar to data sources used to compute QR metrics, this information was not provided by most of the primary studies and it should be inferred from the QMI metric's description and/or formula. We identified the data source for 187 metrics. By doing that, we found the same list of data sources that we found for RQ2.2 (*builds*, *issues*, *source code*, *system at runtime*, *tests*, and *version control system* data) and a new

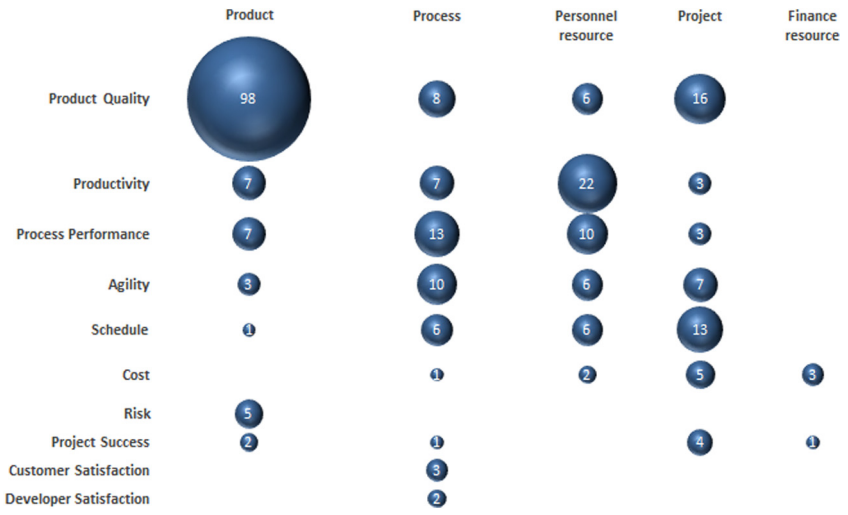


Fig. 20. Entities used in QMI Metrics.

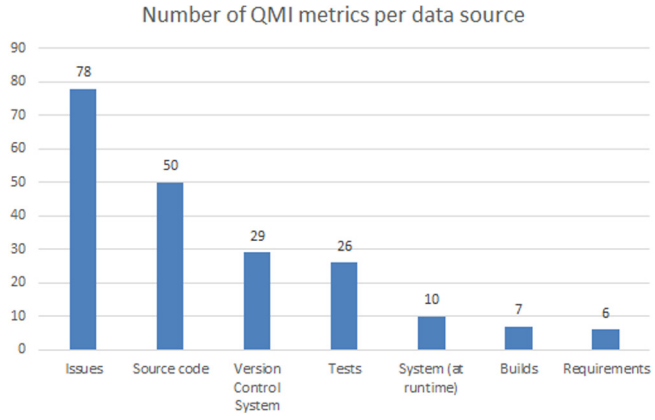


Fig. 21. Data sources used in QMI Metrics.

one, *requirements*. The most used data source is *issues* (78 metrics, 35%), with 64 QMI metrics using only issue data (e.g., *#Defects injected per sprint*, *Amount of programming effort*, and *Bug correction time*), nine using issues and source code (e.g., *Defect density (Defects/KLOC)*, *Size/Effort*, and *Defect rate (number of errors/complexity)*), four using issues and tests (e.g., *Test Effectiveness (defect#/person-hour)*, *Test Speed (TS#/person-hour)*), and one using issues and VCS data (*Productivity (LoC/person-hour)*). The number of QMI metrics per each data source is shown in Fig. 21 (some metrics need more than one data source).

Fig. 22 shows the results of the analysis of which data sources are providing data for measuring the entities (blue bubbles) and QMIs (white bubbles). Entities and QMIs not using data sources are not included in the figure (*Finance resource* entity and *Customer and developer satisfaction* QMIs). For the entities, only *Product* is using all the reported data sources; the most used are *Source code* and *Issues*, followed by *Tests* and *Version control systems* data. The other entities are measured by a similar number of data sources (four or five) except *Finance resources*, for which we could not identify any data source. In the case of QMIs, we found a very diverse situation, ranging from *Product Quality*, which uses all data sources, to *Customer Satisfaction* and *Developer Satisfaction*, which do not use any data source but instead they are measured qualitatively through questionnaires.

Metrics for *Project Success*, *Risk*, and *Agility* were reported in a qualitative way; the primary studies do not include details about how they were computed, this is the reason why there is so little information about the data sources that can be used.

4.3.6. RQ3.6 - What QR Types are used to measure QMIs?

This subRQ addresses the relationship of QR measurement and QMIs. Given that QRs are defined as software quality attributes that must be present in the software product, we classified the 121 QMI metrics related to the entity *product* (see Fig. 19) using the software product quality defined by the ISO/IEC 25010, as we did in RQ2. Fig. 23 shows which QR Types are used to measure the 10 QMIs identified in RQ3.1.

We identified five QR Types used in the QMI metrics: *Functional Suitability*, *Maintainability*, *Performance*, *Reliability*, and *Security*. None of the QMIs uses all the QRs in the reported metrics. The QMI related to more QRs is *Product Quality*, as we could expect because QR by definition is defined to improve the software quality. Curiously enough, *Security* is not measured as part of *Product Quality* QMI; the few metrics related to security are used in the context of risk management, assessed by *Risk* QMI.

The other QMIs impacted by QRs are *Process Performance, Productivity*, *Project Success*, and *Risk*. *Process Performance* uses metrics measuring the old issues (Functional Suitability) and metrics related to bugs like bugs ratio and leakage (Reliability). *Productivity* uses metrics for code size and complexity (*Maintainability*), and number of defects and passed tests (*Reliability*). *Project Success* is a tailored indicator that is assessed with four metrics, two of them related to quality factors (size assessing *Maintainability* and defect rate assessing *Reliability*) and time and budget not related to product quality factors. Finally, metrics used for *Risk* are measuring the risk of having complex files (*Maintainability*) and the risk of suffering an attack (*Security*).

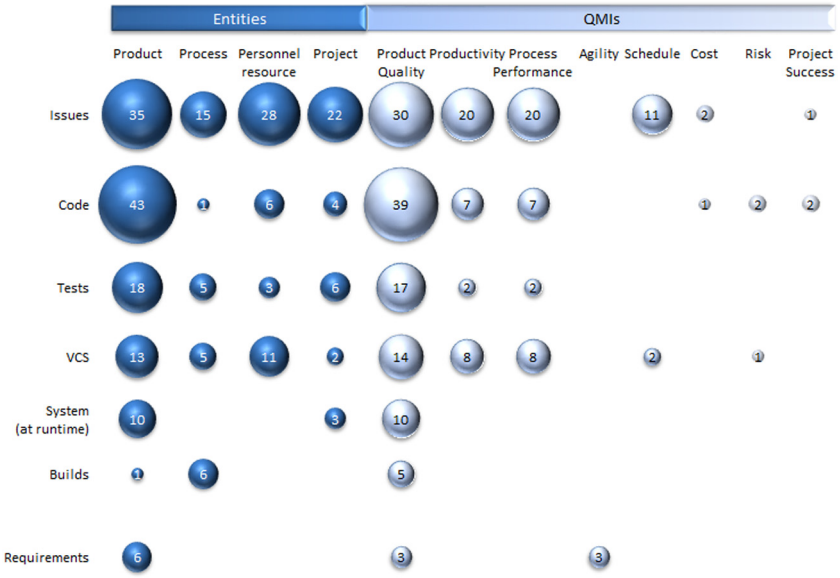


Fig. 22. Data sources used for measuring entities and QMIs. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

Fig. 23. QR Types used in QMI Metrics.

5. Discussion

The analysis of the type of research methods applied and the quality assessment results show that the research in the field is quite relevant for practitioners. All the primary studies, except for three, apply empirical methods, and their quality assessment proves high industrial relevance scores. But, in spite of that, from a research point of view, there are some signs that suggest a lack of maturity in the field. For example, we did not find any standardized models for metrics, neither a solid-holistic view (e.g., only one paper measuring "project success", and very few metrics related to customer/user satisfaction).

This section presents the implications of the findings of the mapping study for software engineering researchers (Section 5.2) and practitioners (Section 5.3). We precede this analysis by a summary of the most relevant observations on bibliometrics (Section 5.1).

5.1. Analysis of bibliometrics

5.1.1. Publication trends

We found 61 primary studies related to QR management through metrics in ARSD. From these primary studies, 28 relate to QR metrics (45,9%) and 46 relate to QMI (73, 75.4%). From the 46 papers reporting QMIs, 38 also report QMI metrics (i.e., 82.6% of these 46 papers provide means to measure the QMIs). A corollary of the numbers above is that 12 primary studies (19.7%) target both QR metrics and QMIs, providing a comprehensive analysis of QR management through metrics in ARSD.

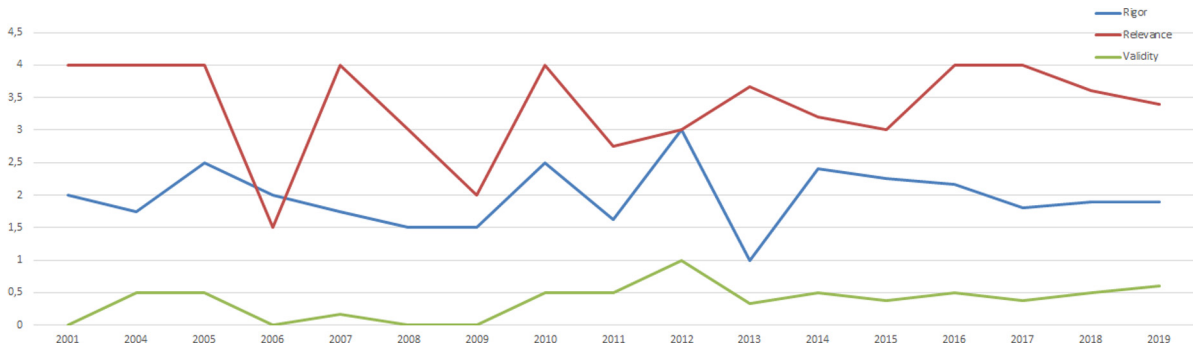


Fig. 24. Trends on relevance (0–4), rigor (0–3) and validity (0–1).

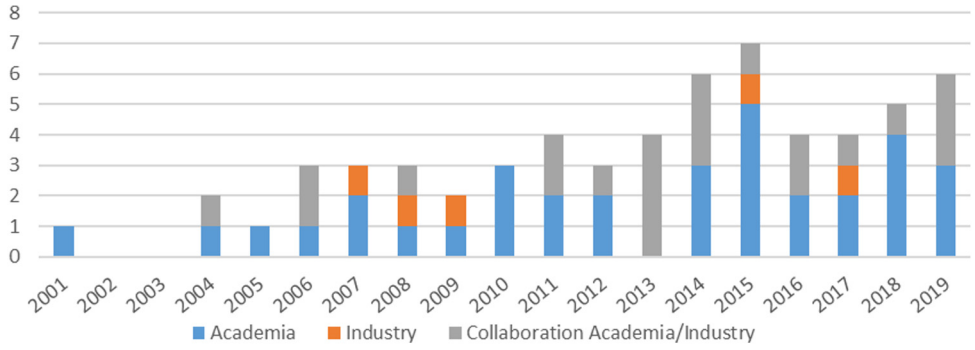


Fig. 25. Annual distribution of academia/industry papers.

While RQ1.1 reveals an increasing publication trend, we can arguably affirm that there are not many papers on the topic, considering the relevance of quality management in software development. We interpret the results of RQ1.2 somehow aligned to this observation: although there are some software quality and measurement conferences and journals, our results make evident that the literature in this field is really scattered in terms of venue. The absence of flagship venues may be an impediment for a community to grow and boost the research on the topic.

5.1.2. Quality assessment

A positive outcome of our SMS reported in the answer to RQ1.6 is that the assessment of the primary studies' quality yields high scores, especially on relevance (0.83 over 1 on average). The average score on rigor is not as high (0.67) as a consequence of the poor rating on validity. If we ignore this aspect and consider only the remaining ones, rigor score achieves 0.79. It seems that quality has improved if we compare with Kupiainen et al.'s paper (Kupiainen et al., 2015), where we can read "Even though there were many low scoring studies, they were included". Also, Hoda et al. remark that "Combining research rigor and industrial relevance still remains a 'grand challenge' for ASD[8] research" (Hoda et al., 2017). This improvement would be higher if more attention is put on validity, the weak point of the primary studies.

This observation on improved quality seems to be supported if we analyze the trends over time (see Fig. 24). We can see that the scores[9] tend to be more stable in the last 5-year period. The observation reported in RQ1.2 about the higher share of journal papers published since 2014, as well as the wide use of empirical methods in the primary studies (cf. RQ1.3), aligns to this increase in quality too.

5.1.3. Industrial collaboration

The quality assessment performed in RQ1.4 reveals high relevance for industry. This practitioners' perception is confirmed by the high number of authors from industry as reported in RQ1.4: up to 44% of the primary studies include authors from industry. This percentage is significantly higher than usual; as an example, it is higher than the percentage found in the previous SMS on QR management (Behutiye et al., 2020), which was 28% only.

As a result of the analysis of the number of publications including authors from industry (see Fig. 25), we can remark that it is quite stable. Years including industry authors (industry and collaboration), have between one and three publications, with the exception of 2013 with four. What is remarkable is that industry authors publish almost all years, except for 2005 and 2013, in 2013 there were only papers with industry authors.

Regarding the geographical distribution, it is remarkable that almost half of the companies involved in primary studies belong to Sweden (48%); in fact, all the primary studies from Sweden include some authors from industry. However, this fact also raises an observation already reported by Kupiainen et al. (2015) that they called "Ericsson bias": in our SMS, 25.9% of papers with participation of industry authors had Ericsson affiliation (even higher than the 22% reported by Kupiainen et al.). Kupiainen et al. literature review analyzes primary studies until 2013, our study reveals that Ericsson continues the research on the field, we found four publications in the 6-year period 2014–2019.

5.2. Implications for researchers in quality measurement in ARSD

5.2.1. Elusive terminology

When we started this mapping study, and in spite of our work in the previous related mapping on QRs in ARSD (Behutiye et al., 2020), we were not aware of the existence of any specific term for the indicators used in decision-making processes. The analysis of the primary studies reporting indicators we named

[8] ASD: Agile Software Development.

[9] The values assigned to each year are computed as the averages of the papers from that year.

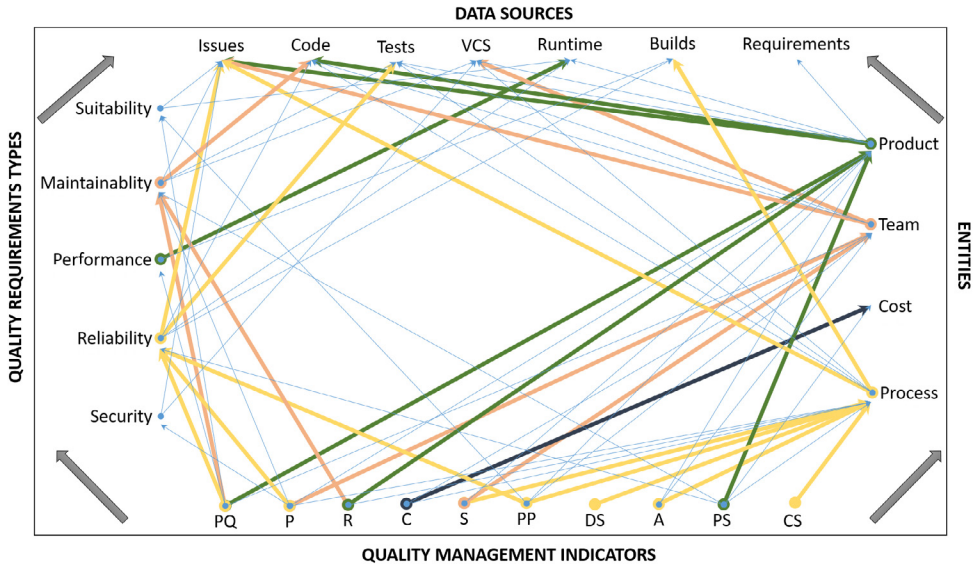


Fig. 26. Potential generic measurement model.

as QMI (RQ3.1), confirmed this lack of concrete terminology, at least in the context of QR management. Most of the primary studies (32; 71.1%) use the word "indicator", a too generic term to be included in a search string. Only one paper [PS51] includes a definition from the JCGM Vocabulary in metrology standard: *"Indicator – measure that provides an estimate or evaluation of specified attributes derived from a model with respect to defined information needs"*. (International Bureau of Weights and Measures., 1993)". Four primary studies use "key performance indicator" [PS17] [PS39] [PS45] [PS50], but only [PS17] also includes the KPI acronym. Q-Rapids project papers use their own terminology, namely "strategic indicator" to highlight the fact that these indicators refer to high-level issues [PS37] [PS38] [PS40]. Others terms used in the primary studies are "factors" or "influencing factors" referring to quality requirements of software quality [PS2]; "quality attribute" related to agile process in [PS25]; and only "metrics" or "software metrics" in [PS5] [PS16] [PS24].

In the case of QRs, while the term itself is well-established in RE (notwithstanding the everlasting controversy with the term "non-functional requirements"), discrepancies arise when it comes to measurement terminology. Some of them mention "quality metrics" (9 primary studies from the 28 reporting QR metrics; 32.1%) or "software quality metrics" (5 primary studies from 28; 17.8%) in the body. The rest of the papers, normally include the word "metric", usually followed by an adjective referring to the measured quality, e.g. "coverage metrics" [PS60], "performance metrics" [PS49] [PS60], "process metrics" [PS56], or "security metrics" [PS45]. We found one of the primary studies that does not even include the word "metric" [PS33]; in this case, the authors use verbs "measures" and "calculate" to introduce the metrics. Looking at the paper keywords, we only found "software quality" in four primary studies [PS21] [PS33] [PS39] [PS40]; "software metrics" in three [PS9] [PS14] [PS43]; "quality metrics" in one [PS26], and "software" and "metric" in one [PS1]. Most of the papers are measuring a QR Type, and they include the name of the characteristic from the ISO/IEC 25010 quality model (e.g., maintainability, reliability, security), but only five primary studies are referring explicitly to the standard.

A consequence of this lack of standard terminology is that searching this kind of literature can be challenging. In the case of systematically surveying the literature, we recommend applying a snowballing search strategy instead of using a search string, as we have done in this SMS.

5.2.2. Low level of detail

Our findings revealed an important observation regarding the reported information on metrics for QRs and QMIs: despite the high-quality assessment, we observed that, in general, primary studies did not include all the necessary details to fully understand the metrics. For example, some recurrent missing information is a definition or the concrete formula. Moreover, most of the papers do not explicitly mention the data source or tool used to get the data needed for the computation. We believe that researchers and practitioners can benefit from definitions and detailed explanations of metrics and their calculation, and therefore we recommend that future studies on the topic take it into consideration.

The lack of details on the metrics reveals that, although the authors of the primary studies are reporting measurement plans for specific quality attributes, the metrics themselves are not the most important outcome. The authors of the primary studies focus on how the metrics are defined and/or selected and how they are used by the companies. However, reporting the concrete data sources and tools that are actually used in practice could be beneficial to get a more precise understanding on how metrics are actually operationalized. A lot of challenges related to measurement are related to a lack of data and actual operationalization of metrics (Ram et al., 2018). As a recommendation for future studies, we believe that reporting this information could shed light on this area.

5.2.3. Quality measurement efforts and gaps

We found that reliability, maintainability and performance efficiency are the top three QR types with most reported metrics. On the other hand, our findings revealed that the literature reports a small number of usability and security QR metrics. Dominance of these three types occurs despite the fact that security and usability are widely reported QRs in the literature of QR management in ARSD (Behutiye et al., 2020). One reason could be the difficulty in quantifying and measuring security and usability QRs thus leading to a small number of metrics reported in the literature. However, this hypothesis does not align well with the huge body of knowledge that exists on these two types of QR. Therefore, we finally interpret it as an indicator of a research gap in these two particular areas in the context of the ARSD research community.

Table 8
Metrics for QMIs and QR Types per data source.

Data Source	*Metric Types*	*QMI (RQ3)*	*QR Type (RQ2 & RQ3.6)*
Builds	Average builds between quality builds, Average time between quality builds, Build performance, Builds feedback time, Deployment speed, Quality branches ratio, Quality builds ratio, Successful builds ratio	Process Performance, Product Quality	Performance Efficiency, Reliability
Issues	Big issues, Bug Types, Bugs (total), Bugs fixing effort, Bugs fixing speed, Bugs ratio (#bugs/#issues), Closed bugs, Closed issues, Closed tasks, Complete data in issues, Critical issues, Development effort, Development speed, Development time, Effort estimation, Fixing bugs effort, Issues (total), New issues, Old issues, Open Bugs, Schedule deviation, Team communication, Team productivity, Testing Effort, Usability Issues, User issues	Cost, Process Performance, Product Quality, Productivity, Schedule	Functional Suitability, Maintainability, Reliability, Security, Usability
Requirements	Requirement changeability, Requirements ambiguity, Requirements analysis, Requirements design, Requirements risk, Specification change	Agility, Product Quality	Functional Suitability
Runtime	CPU usage, Feature usage, Graphics card usage, Memory usage, Operational performance, Rendering speed, Rendering time, Response time, Screen switch time, System availability, System crashes, System errors, System latency, System performance, System resources, System throughput, Transactions rate	Product Quality	Functional Suitability, Performance Efficiency, Reliability, Usability
Source Code (code)	Branch coverage, Code changed, Code cohesion, Code complexity, Code coupling, Code effectiveness, Code extensibility, Code flexibility, Code functionality, Code incorrectness, Code reusability, Code reviews, Code smells, Code status, Code understandability, Code violations, Commented code, Cost, Duplicated code, Product size	Cost, Product Quality, Productivity, Project Success, Risk	Maintainability, Reliability
Tests	Failed test ratio, Failed tests fault content function, Path coverage, State coverage, Successful test ratio, Test execution rate, Test performance, Test ratio, Test speed, Tests, Transition coverage	Process Performance, Product Quality, Productivity	Reliability Maintainability
Version Control System (VCS)	Check-ins, Code changed, Commit comments, Commit speed, Commits (total), File revisions, Issue Comments, Merge speed, Pull requests frequency, Pull requests speed, Team activeness, Technical debt	Process Performance, Product Quality, Productivity, Schedule, Risk	Maintainability
Issues & Code	Bugs ratio (#bugs/LOC), Code changed, Code changed performance, Team productivity	Product Quality, Productivity, Project Success	Reliability
Issues & Tests	Bug Types, Bugs identification productivity, Test speed (per person-hour)	Process Performance, Product Quality	Reliability
Issues & VCS	Team productivity	Process Performance	
Code & Tests	Path coverage, State coverage, Test coverage, Transition coverage	Product Quality	Reliability
Code & VCS	Team productivity	Productivity	
Code & Tests & VCS	Test coverage (per pull request)	Product Quality	Reliability

On the other hand, we observed that product quality and productivity are the top two QMIs reported in our study, respectively. Similarly, our findings revealed the highest number of metrics for these two QMIs. The findings may imply that these two QMIs are widely adopted and relevant when considering quality measurement in ARSD. Given that Agile software development promotes close collaboration with customers to deliver value and ensure customer satisfaction (Beck et al., 2001), one could expect more metrics focused on customer satisfaction in order to have a data-based understanding of customers. However, we only found four works mentioning QMIs related to this concept, far from the numbers of the topmost QMIs (product quality and productivity). This gap calls for further research in this particular topic.

5.2.4. Lack of common measurement model

We have not found any common baseline that could be used to build up a measurement model, that is, a comprehensive quality management strategy including all the elements defined in the SMS conceptual model introduced in Fig. 2, nor to choose or define the elements to be taken into account in the quality management activity.

This lack of common base leads to remarkable dispersion at several levels. For instance, most of the metrics (85% of QR metrics and 89.7% of QMI metrics) are reported only once. We found that some papers measuring the same quality element use specific metrics, all of them really similar but still different. We have also observed that most of the time, the literature does not report the data source used to compute a metric.

This seems to be a gap that should be filled in order to help practitioners to carry out their job in a more rapid way, and to avoid repeating the same reasonings each time a quality management process is carried out and allowing reusing or adapting previously designed models. The existence of some common accepted framework would also increase the confidence of the indicators providing a more solid rationale to the management decisions. The framework, of course, should be flexible enough to allow its customization to each management context. It could be built upon existing proposals or standards, e.g., the ISO/IEC 25010 standard for defining types of QRs, but it should cover aspects like process not addressed by this standard.

Although there is no explicit model stated and used, a certain pattern can be extracted from the primary studies that might be

Table 9
Tools providing data sources.

	Builds	Issues	Req.	Runtime	Code	Tests	VCS	Total DS
Cantata++					X	XX		2
CodeSonar					X			1
Coverity					X			1
Jira		X				X *(plug-in)*		2
Jenkins	X					X		2
Gerrit							X	1
Git					X		X	2
GitHub		X					X	2
GitLab	XX	X			X	X	X	5
Greenhopper		X						1
Mantis		X						1
Nagios				X				1
OpenProject		X						1
Redmine		X						1
SonarQube					X	X		2
SVN							X	1
TravisCI	X					XX		2
Zabbix				X				1

a starting point of research to fill this gap. It is depicted in Fig. 26 and it consists of a directed graph connecting QMIs, data sources, QR types and entities.

Using data in Fig. 20, each QMI is linked to the entities it depends on, being the one linked with the thickest arrow the most important to the QMI. In a similar way, QMIs are linked to QR types according to data in Fig. 23. The edges from QR types to data sources come from Fig. 13, and, finally, Fig. 22 contains the data to establish the edges going from entities to data sources. Interestingly enough, we have seen that, with the exception of risk, cost and agility (with very little use in the primary studies) given a QMI, the sources reached by the thickest arrows include the most important sources to the QMI according to the primary studies (Fig. 22). Moreover, the data sources reached starting from QMIs linked to some QR type following the left path also include the most important sources reported in Fig. 22 with the exception, in this case, of productivity.

5.3. Implications for practitioners in ARSD processes

As we can see in the results section, the industrial relevance of the primary studies is quite high, even higher than the research rigor classification. We summarize the evidence below, as a starting point for companies that want to integrate quality measurement as part of their decision-making processes in the context of ARSD.

The results of this SMS can be used by the organizations to understand which key indicators (QRs and QMIs) they need to measure and monitor to improve the quality of their products and processes. In order to automate the key indicators monitoring, the organizations need to use the data produced by the tools used during their development process. In this section we summarize the most relevant data produced by the tools that are used by the development team as part of their daily tasks.

As part of the results of this SMS, we found that builds, issues, requirements, system (at runtime), source code, version control system, and tests are the most reported data sources in the context of measuring quality in ARSD processes. We found that the most relevant data sources are: issues (99 QR and QMI metrics), source code (85), tests (40), system (at runtime) (40) and version control system (31). In the following table (Table 8), we summarize the metric types that can be measured per each data source, we also include which QMIs (RQ3) and QR Types (RQ2 and RQ3.5) can be measured using these metrics. For the QR Types, we include the QR extracted from RQ2 and the QRs identified in the analysis of RQ3.6 (analyzing QMI metrics), the last ones are underlined.

Table 9 summarizes concrete tools providing data, i.e., data sources, which can be used to measure both QRs and QMIs. It includes data sources used by the tools reported in the primary studies (marked as X) and the data sources available on the tools that are not used in the primary studies (XX), this complementing data has been extracted from tools websites and documentation. We have excluded specific tools that are used for specific organizations or products, e.g., Mozilla Wiki, Mozilla Bug Repository and Mozilla Source Code Repository used for the management of Mozilla products, or ServiceNow an ad-hoc tool for backlog management.

The tool providing more kind of data is GitLab, providing all the data sources related to the development phase (builds, issues, source code, tests, and version control system data), the data sources not covered are requirements (before development) and runtime data (after release). Other remarkable tools are GitHub and Jira. GitHub, which provides issues and version control system data, can be integrated with external tools that would eventually provide source code metrics, builds and tests metrics embedded in the pull request data (e.g., static source code analysis from SonarCloud, and test and build from TravisCI). Even Jira, which is mainly an issue tracker system, can be complemented with some apps, provided in the Atlassian marketplace (Jira's producer), e.g., providing test management or continuous integration (TM4J, Zephir, Jenkins), tools providing test and builds information.

6. Conclusions

We have conducted a SMS to survey the literature related to QR management through metrics in ARSD focusing on three aspects: bibliometrics, QR metrics, and QMIs. Our methodology relies on Petersen et al. (2015) and includes the definition of research questions, selection criteria and backward and forward snowballing as a search strategy. The search returned 61 primary

Table 10
Primary studies.

[PS1] Antinyan, V., Staron, M., Meding, W., Österström, P., Wikstrom, E., Wranker, J., …& Hansson, J. (2014, February). Identifying risky areas of software code in Agile/Lean software development: An industrial experience report. In 2014 Software Evolution Week-IEEE Conference on Software Maintenance, Reengineering, and Reverse Engineering (CSMR-WCRE) (pp. 154-163). IEEE.
[PS2] Asghar, A. R., Bhatti, S. N., Tabassum, A., Sultan, Z., & Abbas, R. (2016). Role of requirements elicitation & prioritization to optimize quality in scrum agile development. work, 7(12).
[PS3] Baca, D., Boldt, M., Carlsson, B., & Jacobsson, A. (2015, August). A novel security-enhanced agile software development process applied in an industrial setting. In Availability, Reliability and Security (ARES), 2015 10th International Conference on (pp. 11-19). IEEE.
[PS4] Bakota, T., Hegedűs, P., Ladányi, G., Körtvélyesi, P., Ferenc, R., & Gyimóthy, T. (2012, September). A cost model based on software maintainability. In 2012 28th IEEE International Conference on Software Maintenance (ICSM) (pp. 316-325). IEEE.
[PS5] Bartels, R. A., Rodriguez, J., & Jenkins, M. (2009). Implementing software metrics in an agile organization: a case study. In COMPEDES09: II Congreso Computación para el Desarrollo.
[PS6] Bartsch, S. (2011, February). Authorization enforcement usability case study. In International Symposium on Engineering Secure Software and Systems (pp. 209-220). Springer, Berlin, Heidelberg.
[PS7] Boldt, M., Baca, D., & Carlsson, B. (2017). Introducing a novel security-enhanced agile software development process. International Journal of Secure Software Engineering, 8(2).
[PS8] Bourimi, M., & Tesoriero, R. (2014, July). Non-Functional Requirements for Distributable User Interfaces in Agile Processes. In Proceedings of the 2014 Workshop on Distributed User Interfaces and Multimodal Interaction (pp. 54-66). ACM.
[PS9] Çalikli, G., Staron, M., & Meding, W. (2018, May). Measure early and decide fast: transforming quality management and measurement to continuous deployment. In Proceedings of the 2018 International Conference on Software and System Process (pp. 51-60).
[PS10] Camacho, C. R., Marczak, S., & Cruzes, D. S. (2016, August). Agile team members perceptions on non-functional testing: influencing factors from an empirical study. In Availability, Reliability and Security (ARES), 2016 11th International Conference on (pp. 582-589). IEEE.
[PS11] Cannizzo, F., Clutton, R., & Ramesh, R. (2008, August). Pushing the boundaries of testing and continuous integration. In Agile, 2008. AGILE'08. Conference (pp. 501-505). IEEE.
[PS12] Chen, H. M., Kazman, R., & Haziyev, S. (2016). Strategic prototyping for developing big data systems. IEEE Software, 33(2), 36-43.
[PS13] Clark, S., Collis, M., Blaze, M., & Smith, J. M. (2014, November). Moving targets: Security and rapid-release in firefox. In Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security (pp. 1256-1266).
[PS14] Concas, G., Di Francesco, M., Marchesi, M., Quaresima, R., & Pinna, S. (2008, June). Study of the evolution of an agile project featuring a web application using software metrics. In International Conference on Product Focused Software Process Improvement (pp. 386-399). Springer, Berlin, Heidelberg.
[PS15] De Diana, M. J. D. O., Kon, F., & Gerosa, M. A. (2010). Conducting an Architecture Group in a Multi-team Agile Environment. Proceedings of I Workshop Brasileiro de Métodos Ágeis, Porto Alegre - RS.
[PS16] Domah, D., & Mitropoulos, F. J. (2015, April). The NERV methodology: A lightweight process for addressing non-functional requirements in agile software development. In SoutheastCon 2015 (pp. 1-7). IEEE.
[PS17] Dragicevic, S., & Celar, S. (2013). Method for elicitation documentation and validation of software user requirements (MEDoV). 18th IEEE International Symposium on Computers and Communications (ISCC 2013), 956-961.
[PS18] Gary, K., Enquobahrie, A., Ibanez, L., Cheng, P., Yaniv, Z., Cleary, K., …& Heidenreich, J. (2011). Agile methods for open source safety-critical software. Software: Practice and Experience, 41(9), 945-962.
[PS19] Ghanbari, H., Similä, J., & Markkula, J. (2015). Utilizing online serious games to facilitate distributed requirements elicitation. Journal of Systems and Software, 109, 32-49.
[PS20] Graham, T. N., Kazman, R., & Walmsley, C. (2007, May). Agility and experimentation: Practical techniques for resolving architectural tradeoffs. In Software Engineering, 2007. ICSE 2007. 29th International Conference on (pp. 519-528). IEEE.
[PS21] Haindl, P., & Plösch, R. (2019, March). Towards Continuous Quality: Measuring and Evaluating Feature-Dependent Non-Functional Requirements in DevOps. In 2019 IEEE International Conference on Software Architecture Companion (ICSA-C) (pp. 91-94). IEEE.
[PS22] Hanssen, G., Yamashita, A. F., Conradi, R., & Moonen, L. (2010, January). Software entropy in agile product evolution. In System Sciences (HICSS), 2010 43rd Hawaii International Conference on (pp. 1-10). IEEE.
[PS23] Ho, C. W., Johnson, M. J., Williams, L., & Maximilien, E. M. (2006, July). On agile performance requirements specification and testing. In Agile Conference, 2006 (pp. 6-pp). IEEE.
[PS24] Ilieva, S., Ivanov, P., & Stefanova, E. (2004, August). Analyses of an agile methodology implementation. In Euromicro Conference, 2004. Proceedings. 30th (pp. 326-333). IEEE.
[PS25] Jeon, S., Han, M., Lee, E., & Lee, K. (2011, August). Quality attribute driven agile development. In 2011 Ninth International Conference on Software Engineering Research, Management and Applications (pp. 203-210). IEEE.
[PS26] Jinzenji, K., Hoshino, T., Williams, L., & Takahashi, K. (2013, November). An experience report for software quality evaluation in highly iterative development methodology using traditional metrics. In Software Reliability Engineering (ISSRE), 2013 IEEE 24th International Symposium on (pp. 310-319). IEEE.
[PS27] Keramati, H., & Mirian-Hosseinabadi, S. H. (2008, March). Integrating software development security activities with agile methodologies. In Computer Systems and Applications, 2008. AICCSA 2008. IEEE/ACS International Conference on (pp. 749-754). IEEE.
[PS28] Khomh, F., Dhaliwal, T., Zou, Y., & Adams, B. (2012, June). Do faster releases improve software quality? an empirical case study of mozilla firefox. In 2012 9th IEEE Working Conference on Mining Software Repositories (MSR) (pp. 179-188). IEEE.
[PS29] Khomh, F., Adams, B., Dhaliwal, T., & Zou, Y. (2015). Understanding the impact of rapid releases on software quality. Empirical Software Engineering, 20(2), 336-373.
[PS30] Krishna, A. B., & Abraham, S. E. (2015, May). Statistical analysis of memory and performance non functional requirements in real time embedded system development for agile methodology. In Industrial Instrumentation and Control (ICIC), 2015 International Conference on (pp. 300-305). IEEE.
[PS31] Kula, E., Rastogi, A., Huijgens, H., Deursen, A. V., & Gousios, G. (2019, August). Releasing fast and slow: an exploratory case study at ING. In Proceedings of the 2019 27th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering (pp. 785-795).

(*continued on next page*)

Table 10 (*continued*).

[PS32] Layman, L., Williams, L., & Cunningham, L. (2004, June). Exploring extreme programming in context: an industrial case study. In Agile Development Conference, 2004 (pp. 32-41). IEEE.
[PS33] Li, J., Moe, N. B., & Dybå, T. (2010, September). Transition from a plan-driven process to scrum: a longitudinal case study on software quality. In Proceedings of the 2010 ACM-IEEE international symposium on empirical software engineering and measurement (p. 13). ACM. [PS34] Li, X., Li, Y. F., Xie, M., & Ng, S. H. (2011). Reliability analysis and optimal version-updating for open source software. Information and Software Technology, 53(9), 929-936.
[PS35] MacCormack, A. (2001). Product-development practices that work: How Internet companies build software. MIT Sloan Management Review, 42(2), 75.
[PS36] Mann, C., & Maurer, F. (2005, July). A case study on the impact of scrum on overtime and customer satisfaction. In Agile Conference, 2005. Proceedings (pp. 70-79). IEEE.
[PS37] Manzano, M., Gómez, C., Ayala, C., Martínez-Fernández, S., Ram, P., Rodríguez, P., & Oriol, M. (2018, August). Definition of the on-time delivery indicator in rapid software development. In 2018 IEEE 1st International Workshop on Quality Requirements in Agile Projects (QuaRAP) (pp. 1-5). IEEE.
[PS38] Manzano, M., Mendes, E., Gómez, C., Ayala, C., & Franch, X. (2018, October). Using Bayesian networks to estimate strategic indicators in the context of rapid software development. In Proceedings of the 14th International Conference on Predictive Models and Data Analytics in Software Engineering (pp. 52-55).
[PS39] Martínez-Fernández, S., Jedlitschka, A., Guzmán, L., & Vollmer, A. M. (2018, August). A quality model for actionable analytics in rapid software development. In 2018 44th Euromicro Conference on Software Engineering and Advanced Applications (SEAA) (pp. 370-377). IEEE.
[PS40] Martínez-Fernández, S., Vollmer, A. M., Jedlitschka, A., Franch, X., López, L., Ram, P., …& Partanen, J. (2019). Continuously assessing and improving software quality with software analytics tools: a case study. IEEE access, 7, 68219-68239.
[PS41] Meding, W. (2017, October). Effective monitoring of progress of agile software development teams in modern software companies: an industrial case study. In Proceedings of the 27th International Workshop on Software Measurement and 12th International Conference on Software Process and Product Measurement (pp. 23-32).
[PS42] Olague, H. M., Etzkorn, L. H., Li, W., & Cox, G. (2006). Assessing design instability in iterative (agile) object-oriented projects. Journal of Software Maintenance and Evolution: Research and Practice, 18(4), 237-266.
[PS43] Padmini, K. J., Bandara, H. D., & Perera, I. (2015, April). Use of software metrics in agile software development process. In 2015 Moratuwa Engineering Research Conference (MERCon) (pp. 312-317). IEEE.
[PS44] Pohl, C., & Hof, H. J. (2015). Secure Scrum: Development of secure software with SCRUM. arXiv preprint arXiv:1507.02992.
[PS45] Poller, A., Kocksch, L., Türpe, S., Epp, F. A., & Kinder-Kurlanda, K. (2017, February). Can Security Become a Routine?: A Study of Organizational Change in an Agile Software Development Group. In CSCW (pp. 2489-2503).
[PS46] Roden, P. L., Virani, S., Etzkorn, L. H., & Messimer, S. (2007, September). An empirical study of the relationship of stability metrics and the qmood quality models over software developed using highly iterative or agile software processes. In Source Code Analysis and Manipulation, 2007. SCAM 2007. Seventh IEEE International Working Conference on (pp. 171-179). IEEE.
[PS47] Rundle, P. J., & Dewar, R. G. (2006, May). Using return on investment to compare agile and plan-driven practices in undergraduate group projects. In Proceedings of the 28th international conference on Software engineering (pp. 649-654). ACM.
[PS48] Schwartz, L. (2014). Agile-User Experience Design: Does the Involvement of Usability Experts Improve the Software Quality?. International Journal on Advances in Software Volume 7, Number 3 & 4, 2014.
[PS49] Shu, X., & Maurer, F. (2007, August). A Tool for Automated Performance Testing of Java3D Applications in Agile Environments. In Software Engineering Advances, 2007. ICSEA 2007. International Conference on (pp. 35-35). IEEE.
[PS50] Staron, M., Meding, W., & Palm, K. (2012, May). Release readiness indicator for mature agile and lean software development projects. In International Conference on Agile Software Development (pp. 93-107). Springer Berlin Heidelberg.
[PS51] Staron, M., Meding, W., Hansson, J., Höglund, C., Niesel, K., & Bergmann, V. (2013). Dashboards for continuous monitoring of quality for software product under development. System Qualities and Software Architecture (SQSA).
[PS52] Staron, M., Hansson, J., Feldt, R., Henriksson, A., Meding, W., Nilsson, S., & Höglund, C. (2013, October). Measuring and visualizing code stability–a case study at three companies. In 2013 Joint Conference of the 23rd International Workshop on Software Measurement and the 8th International Conference on Software Process and Product Measurement (pp. 191–200). IEEE.
[PS53] Syed-Mohamad, S. M., & Akhir, N. S. M. (2019, December). SoReady: An Extension of the Test and Defect Coverage-Based Analytics Model for Pull-Based Software Development. In 2019 26th Asia-Pacific Software Engineering Conference (APSEC) (pp. 9-14). IEEE.
[PS54] Tarhan, A., & Yilmaz, S. G. (2014). Systematic analyses and comparison of development performance and product quality of Incremental Process and Agile Process. Information and Software Technology, 56(5), 477-494.
[PS55] Tetmeyer, A., Hein, D., & Saiedian, H. (2014). A tagging approach to extract security requirements in non-traditional software development processes. International Journal of Secure Software Engineering (IJSSE), 5(4), 31-47.
[PS56] Vasilescu, B., Yu, Y., Wang, H., Devanbu, P., & Filkov, V. (2015, August). Quality and productivity outcomes relating to continuous integration in GitHub. In Proceedings of the 2015 10th Joint Meeting on Foundations of Software Engineering (pp. 805-816). ACM.
[PS57] Vassallo, C., Zampetti, F., Romano, D., Beller, M., Panichella, A., Di Penta, M., & Zaidman, A. (2016, October). Continuous delivery practices in a large financial organization. In 2016 IEEE International Conference on Software Maintenance and Evolution (ICSME) (pp. 519-528). IEEE.
[PS58] Vassallo, C., Palomba, F., Bacchelli, A., & Gall, H. C. (2018, September). Continuous code quality: are we (really) doing that?. In Proceedings of the 33rd ACM/IEEE International Conference on Automated Software Engineering (pp. 790-795).
[PS59] Vollmer, A. M., Martınez-Fernández, S., Bagnato, A., Partanen, J., López, L., & Rodrıguez, P. (2019, September). Practical experiences and value of applying software analytics to manage quality. In 2019 ACM/IEEE International Symposium on Empirical Software Engineering and Measurement (ESEM) (pp. 1-6). IEEE.
[PS60] Yuksel, H. M., Tuzun, E., Gelirli, E., Biyikli, E., & Baykal, B. (2009, September). Using continuous integration and automated test techniques for a robust C4ISR system. In Computer and Information Sciences, 2009. ISCIS 2009. 24th International Symposium on (pp. 743-748). IEEE.
[PS61] Zhu, M., & Pham, H. (2017). Environmental factors analysis and comparison affecting software reliability in development of multi-release software. Journal of Systems and Software, 132, 72-84.

studies for analysis and data extraction, from the 3.778 evaluated papers. 28 papers contain QR metrics (45.9%) and 46 contain QMI (75.4%), 38 (62.3%) among the 46 contain QMI metrics also.

From the results found, we have identified research gaps and challenges that we think that should be addressed to increase the maturity of the field and to provide a solid base to practitioners:

Table 11
Data extraction form.

Data item	*Value*	*RQ*
PublicationID	First author's surname plus publication year	
Publication Year	Integer	RQ1.1
Authors' affiliation	In terms of academia, industry or industry-academia collaboration. In the case of industry, also the list of authors' company names (all of them)	RQ1.4
Authors' countries	The list of countries of the authors' organizations (all of them)	RQ1.4
Venue Type	Conference, Journal, Workshop or Book Chapter	RQ1.2
Research Method	Empirical (Action Research, Case Study, Controlled Experiment, Experiment, Survey, Other), Theoretical, Experience Report	RQ1.3
Research rigor	The research method of an empirical study, the context, study design and validity aspects. Adopted from Ivarsson and Gorschek (2011).	RQ1.6
Industrial relevance	Industrial relevance of an empirical study: aspects related to the industrial relevance (subjects, context, scale, and research method) are evaluated. Adopted from Ivarsson and Gorschek (2011).	RQ1.6
Domain	Application domain of the study as reported in the study	RQ1.5
For each QR Metric		
QR Name	Name of the quality requirement(s) as reported in the study	RQ2.1
QR Metric	Name of the metric used to compute the QR as reported in the study	RQ2.1
QR Metric Data Source	Source of data used to compute the metric	RQ2.2
QR Metric Data Source Tool	The name of the tool producing the data used to compute the metric	RQ2.3
For each QMI		
QMI Name	Name of the indicator as reported in the study	RQ3.1
QMI Metric	Name and/or formula of the factors used to compute the QMI	RQ3.3
QMI Metric's QR	Name of the QRs (if mentioned) that are measured by the QMI Metric	RQ3.6
QMI Metric Measured Entity	Type of measured entity (i.e., product, process, project, resources)	RQ3.4
QMI Metric Data Source	Source of data used to compute the metric (e.g., issues, source code, …)	RQ3.5
QMI Management Tool	The name of the visualization tool	RQ3.2

lack of a common framework and terminology, lack of metrics and tools for some QR and QMI and low level of detail in the definition of metrics.

We think that the information provided by this study may be valuable to the research on quality models in ARSD as well as Arcos-Medina and Mauricio, who claim that their results can contribute to the definition of a comprehensive quality model (Arcos-Medina and Mauricio, 2019). We have gone a little forward and sketch a possible way to link quality concepts at different levels in the discussion section.

From the industry point of view, our collection of measures and data sources can serve as a starting point for practitioners who want to include quality measurement into their decision-making processes. The results of this SMS can be used by the organizations to understand which key indicators (QRs and QMIs) they need to measure and monitor to improve the quality of their products and processes as well as the data sources and tools that they can use.

Conducting this SMS, some questions arise that could be faced by the research community in the field complementing the gaps presented in the discussion section. The analysis reported in this SMS could be complemented by studying which metrics are applied to which phases, activities, and stakeholders of the ARSD. This study could be also replicated for the context of non-agile methods in order to provide a wider view identifying which metrics are specific for ARSD.

CRediT authorship contribution statement

Lidia López: Conceptualization, Methodology, Investigation, Data curation, Analysis, Supervision, Paper writing, Writing - original draft, Visualization, Discussion section. **Xavier Burgués:** Conceptualization, Methodology, Investigation, Data curation, Analysis, Writing - original draft, Visualization, Discussion section. **Silverio Martínez-Fernández:** Conceptualization, Methodology, Investigation, Data curation, Analysis, Writing - original draft, Visualization, Discussion section. **Anna Maria Vollmer:** Conceptualization, Methodology, Investigation, Data curation, Analysis, Writing - original draft, Visualization, Discussion section. **Woubshet Behutiye:** Conceptualization, Methodology, Investigation, Data curation, Analysis, Writing - original draft, Visualization, Discussion section. **Pertti Karhapää:** Conceptualization, Methodology, Investigation, Data curation, Analysis, Writing - original draft, Visualization, Discussion section. **Xavier Franch:** Conceptualization, Methodology, Initial SMS activities (search and Analysis), Supervision, Discussion section, Writing – review & editing. **Pilar Rodríguez:** Conceptualization, Methodology, Discussion section, Writing – review & editing. **Markku Oivo:** Conceptualization, Methodology, Discussion section, Writing – review & editing.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

This work has been funded by the European Union's Horizon 2020 research and innovation program through the Q-Rapids project (grant no. 732253). This research was also partially supported by the Spanish Ministerio de Economía, Industria *y* Competitividad through the DOGO4ML project (grant PID2020-117191RB-I00). Silverio Martínez-Fernández worked in Fraunhofer IESE before January 2020.

Appendix A. Primary studies

See Table 10.

Appendix B. Data extraction form

See Table 11.

References

Ampatzoglou, A., Bibi, S., Avgeriou, P., Verbeek, M., Chatzigeorgiou, A., 2019. Identifying, categorizing and mitigating threats to validity in software engineering secondary studies. Inf. Softw. Technol. 106, 201–230.

Anon, 2011. ISO/IEC 25010 - systems and software engineering - systems and software quality requirements and evaluation (SQuaRE) - system and software quality models. https://www.iso.org/standard/35733.html (accessed January 2021).

Anon, 2017. ISO/IEC/IEEE 15939 - systems and software engineering — Measurement process. https://www.iso.org/standard/71197.html (Accessed January 2021).

Anon, 2020. 14Th annual state of agile report. https://explore.digital.ai/state-of-agile/14th-annual-state-of-agile-report (Accessed January 2021).

Anon, 2021a. World Quality Report 2015-16, 7th edition https://www.capgemini.com/thoughtleadership/world-quality-report-2015-16 (Accessed January 2021).

Anon, 2021b. ISO 9000:2015 - quality management systems - fundamentals and vocabulary. https://www.iso.org/obp/ui/#iso:std:iso:9000:ed-4:v1:en (Accessed January 2021).

Arcos-Medina, G., Mauricio, D., 2019. Aspects of software quality applied to the process of agile software development: a systematic literature review. Int. J. Syst. Assur. Eng. Manag. 10 (5), 867–897.

Arvanitou, E.M., Ampatzoglou, A., Chatzigeorgiou, A., Galster, M., Avgeriou, P., 2017. A mapping study on design-time quality attributes and metrics. J. Syst. Softw. 127, 52–77.

Beck, K., et al., 2001. The Agile Manifesto. Agile Alliance., http://agilemanifesto.org/ (Accessed January 2021.

Behutiye, W., Karhapää, P., López, L., Burgués, X., Martínez-Fernández, S., Vollmer, A.M., Oivo, M., 2020. Management of quality requirements in agile and rapid software development: a systematic mapping study. Inf. Softw. Technol. 123, 106225.

Biesialska, K., Franch, X., Muntés-Mulero, V., 2020. Big data analytics in Agile software development: A systematic mapping study. Inf. Softw. Technol. 106448.

Caldiera, V.R.B.G., Rombach, H.D., 1994. The goal question metric approach. Encycl. Softw. Eng. 52, 8–532.

Hoda, R., Salleh, N., Grundy, J., Tee, H.M., 2017. Systematic literature reviews in agile software development: A tertiary study. Inf. Softw. Technol. 85, 60–70.

Ivarsson, M., Gorschek, T., 2011. A method for evaluating rigor and industrial relevance of technology evaluations. Empir. Softw. Eng. 16 (3), 365–395.

Karhapää, P., Behutiye, W., Rodriguez, P., Oivo, M., Costal, D., Franch, X., Abherve, A., 2021. Strategies to manage quality requirements in agile software development: a multiple case study. In: Accepted in Empirical Software Engineering. http://dx.doi.org/10.1007/s10664-020-09903-x.

Kitchenham, B., 2010. What's up with software metrics? – a preliminary mapping study. J. Syst. Softw. 83 (1), 37–51.

Kupiainen, E., Mäntylä, .M.V., Itkonen, J., 2015. Using metrics in Agile and lean software development–a systematic literature review of industrial studies. Inf. Softw. Technol. 62, 143–163.

López, L., Burgués, X., Martínez-Fernández, S., Vollmer, A.M., Behutiye, W., Karhapää, P., Franch, X., Rodríguez, P., Oivo, M., 2021. Quality Measurement in Agile and Rapid Software Development: A Systematic Mapping [Dataset], V.1. Universitat Politècnica de Catalunya, https://doi.org/10.5821/data-2117-335910-1.

Malhotra, R., 2016. Empirical Research in Software Engineering: Concepts, Analysis, and Applications. CRC Press.

Mishra, D., Abdalhamid, S., 2018. Software quality issues in SCRUM: A systematic mapping. J. UCS 24 (12), 1690–1716.

Petersen, K., Vakkalanka, S., Kuzniarz, L., 2015. Guidelines for conducting systematic mapping studies in software engineering: An update. Inf. Softw. Technol. 64, 1–18.

Quality Management Principles, 2015. International Organization for Standardization, Edition 2 https://www.iso.org/publication/PUB100080.html (Accessed January 2021).

Ram, P., Rodriguez, P., Oivo, M., 2018. Software process measurement and related challenges in agile software development: a multiple case study. In: International Conference on Product-Focused Software Process Improvement. Springer, Cham.

Russell, 2017. FTSE. In: Industry Classification Benchmark: Structural Enhancements To the Industry Categorization Framework. https://content.ftserussell.com/sites/default/files/research/industry_classification_benchmark-final.pdf (Accessed January 2021).

Wohlin, C., 2014. Guidelines for snowballing in systematic literature studies and a replication in software engineering. In: Proceedings of the 18th international conference on evaluation and assessment in software engineering, pp. 1–10.

Zhou, X., Jin, Y., Zhang, H., Li, S., Huang, X., 2016. A map of threats to validity of systematic literature reviews in software engineering. In: 2016 23rd Asia-Pacific Software Engineering Conference. APSEC, IEEE, pp. 153–160.

Lidia López received her Ph.D. degree in Computing from the Universitat Politècnica de Catalunya (UPC-BarcelonaTech), Spain, in 2013; Currently, she is a senior research engineer at the Barcelona Supercomputing Center (BSC) and Associate Professor at UPC-BarcelonaTech. From 2007 to 2012, she worked as an Assistant teacher at the same University. Her research interests are software engineering, focused in requirements engineering, open source software, empirical software engineering, and data-driven decision-making processes in agile software development. She has been PC co-chair in the international conferences and workshops (CIbSE, iStar) and has been PC member on several international conferences like RCIS, ICSOFT, SAC, and CIbSE. She also reviewed papers for journals including IST, JSS, and IEEE Software. She has participated in several international research projects, e.g., Q-Rapids (H2020, work-package leader) and RISCOSS (FP7, work-package leader and partner representative). Details at http://www.essi.upc.edu/~llopez/.

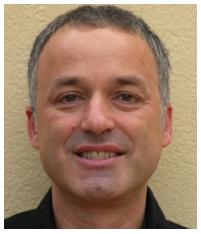
Xavier Burgués is currently working as University Teacher at the Universitat Politècnica de Catalunya (UPC-BarcelonaTech), Spain. He has been participating in several research projects and industrial collaborations and is currently working on the formalisation of quality models as UML models and on the definition of ontologies by means of models, metamodels and instatiation relationships. He has been actively involved in the H2020 Q- Rapids project, "Quality aware rapid software development" work package as a researcher, mainly focusing in quality assessment and measurement models.

Silverio Martínez-Fernández is an Assistant Professor at Universitat Politècnica de Catalunya (UPC-BarcelonaTech) since January 2020. He received his Ph.D. degree in Informatics from UPC in 2016. He was a Post-Doctoral Fellow of the European Research Consortium for Informatics and Mathematics (2016-2018) and operative project manager (2018-2019) in Fraunhofer IESE (Germany). Researcher with more than 50 peer- reviewed publications and H-factor 14 (according to Google Scholar). His interests include Empirical Software Engineering, AI Engineering, Reference Architectures, and Software Analytics. In EU framework programmes, he acted as Evaluation WP leader in Q-Rapids (H2020, RIA), and participated in DESIRA (H2020, RIA). He has participated in the organization of several conferences and workshops: PROFES 2019 (PC Co-chair), CESI@ICSE 2018 (PC-co chair) and QuASD@PROFES 2017-2018 (PC co-chair). He is Editorial Board Member of the SCI-indexed journal IET Software (IEE). He has also been reviewer of multiple journals (e.g. IST, JSS, EMSE, ASE) and PC member of international conferences (e.g. ESEM, ICSME, ECSA, CIbSE). More information at http://www.essi.upc.edu/~smartinez/.

Anna Maria Vollmer received her M.Sc. in Computer Science with the major subject in empirical software engineering from the University of Kaiserslautern in 2016. Since then, she has been working as a researcher in the data science department at the Fraunhofer Institute for Experimental Software Engineering IESE in Germany. Her professional interests range from identifying and modelling relevant data to the evaluation of data-based systems. The connecting element is the focus on quality aspects in the context of data-driven improvements and AI-based systems. As a senior data engineer, she planned and conducted data analysis and evaluation activities in industrial as well as in research projects.

Woubshet Behutiye is a doctoral student and researcher in the M3S research unit, at the faculty of information technology and electrical engineering, University of Oulu. He received his MSc in Information Processing Science from the University of Oulu in 2015. His research interests include agile software development, continuous software engineering, requirements engineering, quality requirements, technical debt management, and evidence based software engineering. He has been actively involved in the H2020 Q-Rapids project, "Quality aware rapid software development" work package as a researcher.

Pertti Karhapää is currently working as University Teacher at the University of Oulu, he is also a doctoral student and researcher in the M3S research group in the Department of Information Processing Science, University of Oulu. His research interests are Requirements Engineering, agile software development processes, and management of quality requirements. He teaches discrete mathematics for computing and data structures and algorithms. He is also teaching students of Nanjing Institute of Technology. He has been actively involved in the H2020 Q-Rapids project, "Quality aware rapid software development" work package as a researcher.



Xavier Franch is a professor in Software Engineering at the Universitat Politècnica de Catalunya (UPC-BarcelonaTech). He received his Ph.D. degree in Informatics from UPC in 1996. His research interest embraces many fields in software engineering, including requirements engineering, empirical software engineering, open source software, and agile software development. He is a member of the IST, REJ, and Computing editorial boards, Journal First chair of JSS, and Deputy Editor of IET Software. He served as a PC chair at CAiSE'22, RE'16, ICSOC'14, CAiSE'12, and REFSQ'11, among others, and as GC for RCIS'22, PROFES'19 and RE'08. More information at https://www.essi.upc.edu/~franch.

Pilar Rodríguez is an Assistant Professor at the Universidad Politécnica de Madrid, Spain. She received her Ph.D. in Computer Science from University of Oulu (Finland) in 2013. Her research interest includes agile and lean software development, software quality, value-based decision-making, and human factors in software engineering. She is a member of the TSE Review Board and has served as a reviewer for other leading SE journals such as ESEM, IST, JSS and SQJ. She has been a PC member of conferences such as ESEM and XP, where she received the XP2021 Best Reviewer Award. Recently, she has been WP2 leader in the H2020 Q-Rapids project. More information at https://scholar.google.fi/citations?hl=es&user=wZQtLdUAAAAJ&view_op=list_works&sortby=pubdate.

Markku Oivo is professor and head of the M3S research unit at the University of Oulu, Finland since 2002. He has worked both in academy and industry in Finland, US, France, Germany, Italy and Spain. During 2000-2002 he was Vice President and director of R&D at Solid Co. He held several positions at VTT in 1986-2000, University of Maryland (1990-91), Schlumberger Ltd. (Paris 1994-95), Fraunhofer IESE (1999-2000), University of Bolzano (2014-15), Universidad Politécnica de Madrid (2015), Kone Co. (1982-86). He has initiated and managed more than 100 national and international research projects. He has served as chair and committee member in organizing numerous international conferences and has been a reviewer for top scientific journals. He is a founding member and chair of the steering committee of ISERN.