Computer Standards & Interfaces 97 (2026) 104110

Contents lists available at ScienceDirect

Computer Standards & Interfaces

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

Quality assessment for software data validation in automotive industry: A systematic literature review

Gilmar Pagoto [a,*], Luiz Eduardo Galvão Martins [a], Jefferson Seide Molléri [b]

[a] UNIFESP – Federal University of São Paulo, São José dos Campos, SP 12.247-014, Brazil
[b] Kristiania University of Applied Sciences, Kirkegata 17, Oslo 0167, Norway

ARTICLE INFO

Keywords:
Software quality assessment
Software quality characteristics
Automotive software validation
Automotive software safety
Automotive software performance
Automotive software cybersecurity

ABSTRACT

Context: The complexity of automotive systems continues to grow, making software quality assessment crucial for vehicle performance, safety, and cybersecurity.
Objectives: This study explores Quality Assessment (QA) in this context, focusing on its key characteristics, practical implications, and expected deliverables.
Method: We performed a systematic literature review (SLR) by selecting 60 studies from digital libraries.
Results: This SLR highlighted essential QA characteristics that should be incorporated into a software validation phase. Our insights encourage the exploration of advanced techniques, such as Artificial Intelligence (AI), and Machine Learning (ML), to support safety-critical software quality assessments in the automotive domain.
Conclusion: The QA of software data validation requires a holistic approach that combines safety, security, and customer expectations, aligned with industry standards, requirements, and specifications. The relevance of AI and ML in managing complex technologies is evidenced, and the traditional real-world validation dependencies bring risks for safety-critical systems validation.

1. Introduction

Based on the IEEE 1012–2016 standard [1], software data validation involves an evaluation process, as part of a system or component, conducted during or after the development process to verify compliance with specified requirements. This standard establishes the guidelines for Verification and Validation (V&V) that provide an objective assessment of products throughout the life cycle. In this work, the target product level is the automotive software. Furthermore, ensuring quality in this evaluation process is crucial for vehicle quality, particularly in terms of safety and security. This is a complex task, and this complexity increases with the integration of emerging technologies. Advances such as autonomous driving, Advanced Driver Assistance Systems (ADAS), connectivity systems, electric propulsion vehicles, battery electric vehicles (BEV), and hybrid electric vehicles (HEV) are expanding the demands on software validation. Additionally, technologies like Artificial Intelligence (AI), and Machine Learning (ML) play a key role in enhancing safety system performance and improving the overall customer experience.

Traditional Quality Assessment (QA) tools, as Failure Mode and Effects Analysis (FMEA) [2], are no longer sufficient. Their limitations restrict coverage and prevent them from offering a fully comprehensive and efficient approach to safety and cybersecurity.

In addition, aspects related to driver experience pose unique challenges, as reference criteria are typically established by trained professionals rather than end-users, whose specific use cases may not be fully addressed. To fulfil these already-known application variations, minimise the uncovered use cases, and align with customer quality perception, more effort and resources are required [3].

Moreover, certain quality aspects are not directly visible to the end-user but are still important for ensuring the product's safety throughout its lifecycle. Poor software quality can lead to accidents, potentially resulting in material damage, financial loss, or even fatalities. ISO standards provide support and serve as foundational recommendations for software data validation [4–7], contributing to the process robustness. In summary, the extensive scope and growing complexity of QA for software data validation motivated this research.

The primary goals of this study are to identify and categorise software quality approaches used in the automotive industry according to their characteristics, applications, and focal areas. This study also seeks to enhance automotive professionals' and researchers' understanding of the critical role of quality in software data validation by outlining

* Corresponding author.
E-mail address: gilmar.pagoto@unifesp.br (G. Pagoto).

https://doi.org/10.1016/j.csi.2025.104110
Received 27 November 2024; Received in revised form 30 August 2025; Accepted 2 December 2025
Available online 4 December 2025

practical implications, expected deliverables, and the direct effects of QA efforts.

The contributions of this Systematic Literature Review (SLR) include the identification of the most applied interventions for QA in software data validation, their relevant characteristics, the impacts of applied interventions, and state-of-the-art trends that suggest future directions for research.

This paper is organised as follows: Section 2 presents the background and related work. Section 3 describes the adopted research methodology. The results and analysis based on research questions are presented in Section 4. In Section 5, we presented our conclusions.

2. Background and related work

This section reviews relevant international standards, discusses the limitations of traditional QA tools, and offers insights into the current state of research in software quality validation.

2.1. Software quality standards in the automotive industry

In software engineering (SE), the ISO/IEC 25,000 series [8], known as SQuaRE (Software Quality Requirements and Evaluation), provides guidelines for managing, modelling, measuring, specifying, and evaluating software quality. It outlines a framework for developing quality measures, including those focused on quality in use (ISO/IEC 25,022), system and software product quality (ISO/IEC 25,023), and data quality (ISO/IEC 25,024). Specifically, ISO/IEC 25,012 addresses software data validation, defining 15 quality characteristics grouped into inherent and system-dependent perspectives. Furthermore, the IEEE 1012 standard for System, Software, and Hardware Verification and Validation (V&V) provides recommendations for evaluating the compliance of development products with their intended use requirements throughout the life cycle.

In the automotive software domain, ISO 21,434 [9] focuses on cybersecurity assurance, addressing software defects, incorrect implementations, and vulnerabilities in security data, especially for embedded software controlling vehicle functions like stability control. Functional Safety (FuSa), outlined in ISO 26,262 [10], addresses potential hazards in electric and electronic safety-related systems, aligning with the system-dependent data characteristics of ISO/IEC 25,012 to support robust software data validation.

Table 1 shows a link between standards and their practical application examples.

Table 1
Practical contributions of standards for software data validation.

Standard	Short description	Use-case contributions
ISO/ IEC25000 Series	Comprehensive Structured guidelines for in-use software data quality.	Evaluation of software data in-use characteristics on inherent and/or system levels (e.g. Efficiency, Completeness, Accessibility, Consistency, etc.).
ISO21434	Cybersecurity in electrical and electronic systems, including software.	Software integrity analysis against potential external attacks through functional software tests, vulnerability scanning, fuzz testing, and/or penetration testing.
ISO26262 Series	Functional safety requirements management, design, implementation, verification, validation, and confirmation measures	Checking of software data for safety-related systems (e.g., autonomous systems) submitted to malfunctioning behaviour conditions.
IEEE1012	Support for Verification and Validation (V&V), including software.	Automotive function specification analysis based on vehicle requirements before software construction phase.

2.2. Automotive software data validation challenges

Selecting and ensuring software data validation in automotive systems is a complex task. Meeting end-user expectations often involves subjective factors that are directly or indirectly influenced by software. Another key challenge lies in linking objective software quality metrics, as determined by QA methods, with the subjective evaluations of experienced professionals [3].

Many Software Reliability Growth Models (SRGMs) rely heavily on large datasets of failure information and often make unrealistic assumptions about software quality estimation [11]. While SRGMs can measure quality by tracking remaining faults, failure rates, and overall software reliability, they struggle to capture customer perspectives and insights [12].

In Automotive Driver Assistance Systems (ADAS), validating software quality in real-world conditions is costly and time-consuming. A malfunction during this phase can lead to serious accidents. To mitigate such risks, vehicle-in-the-loop (ViL) testing can be employed. The main challenge with ViL is ensuring that the virtual data inputs to the vehicle's sensors are consistent with real-world inputs [13].

Traditional tools for QA, such as Fault Tree Analysis (FTA), Failure Mode and Effect Analysis (FMEA), and current software quality standards, often fall short when it comes to safety-critical systems, particularly machine learning-based systems. This occurs mainly because of the high complexity of data-driven systems without a widely accepted engineering solution to deal with that. As a result, more refined, adapted, and specialised QA approaches are required [14].

This work aims to provide a comprehensive picture of the most critical software data characteristics observed in recent years, discuss their contributions to the aforementioned challenges during software validation phases, and provide insights for handling the increasing complexity in the software-defined vehicles domain.

2.3. Related work

Feitosa et al. [15] conducted a systematic mapping study (SMS) with the main goal of "analyse existing software engineering literature for the purpose of characterising the state of the art concerning approaches (e.g., processes, methods, and tools) for designing critical embedded systems (CES) from the point of view of researchers and practitioners in the context of software-intensive systems engineering". The study identified seven critical quality characteristics (dependability, fault-tolerance, performance, reliability, safety, security, and timeliness). Its most relevant outcome is the identification and characterization of the created and/or used approaches to design CES.

In 2005, Bhansali [16] proposed an optimised subset for a universal software standard focused on safety. This paper identifies the minimum safety-critical quality characteristics set for multi-domain applications, such as: "commercial, military and space aviation; medical diagnostic and therapeutic instruments; automotive and transportation systems; industrial process control and robotics; nuclear power plants and weapons control; commercial appliances and ride electronics". The proposed benefits provided by a common safety standard are lower cost and quality software improvement, achieved through common processes and tools.

Rana et al. [17] presented a study of software prediction methods focused on defect detection. The authors presented an overview of 8 prediction models and described the applicability of each method overall software lifecycles. The integrity of software data monitoring is a critical factor not only for the software's performance but also for safety and security. This study pointed out that post-release software monitoring was limited in the past due to a lack of software monitoring during the in-use phase, or insufficient skills and the data retrieval capabilities.

Myllyaho et al. [18] presented an SLR on reported validation methods for AI-specific systems. The study classifies the applied methods into trial, simulation, model-centred validation, and expert opinion, and offers a detailed description of their application in realistic settings. Gezici and Tarhan [19] conducted an SLR to investigate the state-of-the-art practices on software quality (SQ) approaches for AI-based systems. They identified quality characteristics, models, reported challenges by selected studies, and suggested future work on definition/specification, design/evaluation, and process/socio-technical categories. Furthermore, Ali et al. [20] conducted a SMS on the AI systems, software, and components. This study identified no work on AI on software component quality models, a potential gap in that knowledge area.

Krichen [21] explored various formal methods and validation techniques focused on automotive systems security. The study provides a comprehensive overview of the state-of-the-art for systems security validation, including the automotive lifecycle, requirements engineering, design, implementation, and test phases. The challenges identified are scalability, efficiency, and applicability in real-world situations.

2.4. Definitions

Key terms are defined in Appendix A.

3. Methodology

The SLR was conducted followed the guidelines by Kitchenham and Charters [22] and Biolchini et al. [23]. Fig. 1 shows the steps of the SLR process, which are grouped into three main phases: planning, conducting, and reporting.

3.1. Planning

As its objective and motivation (step1), this SLR aims to provide the existing QA processes, methodologies, frameworks, models, or techniques applied for the software data validation phase in the automotive industry (Fig. 1, step 1).

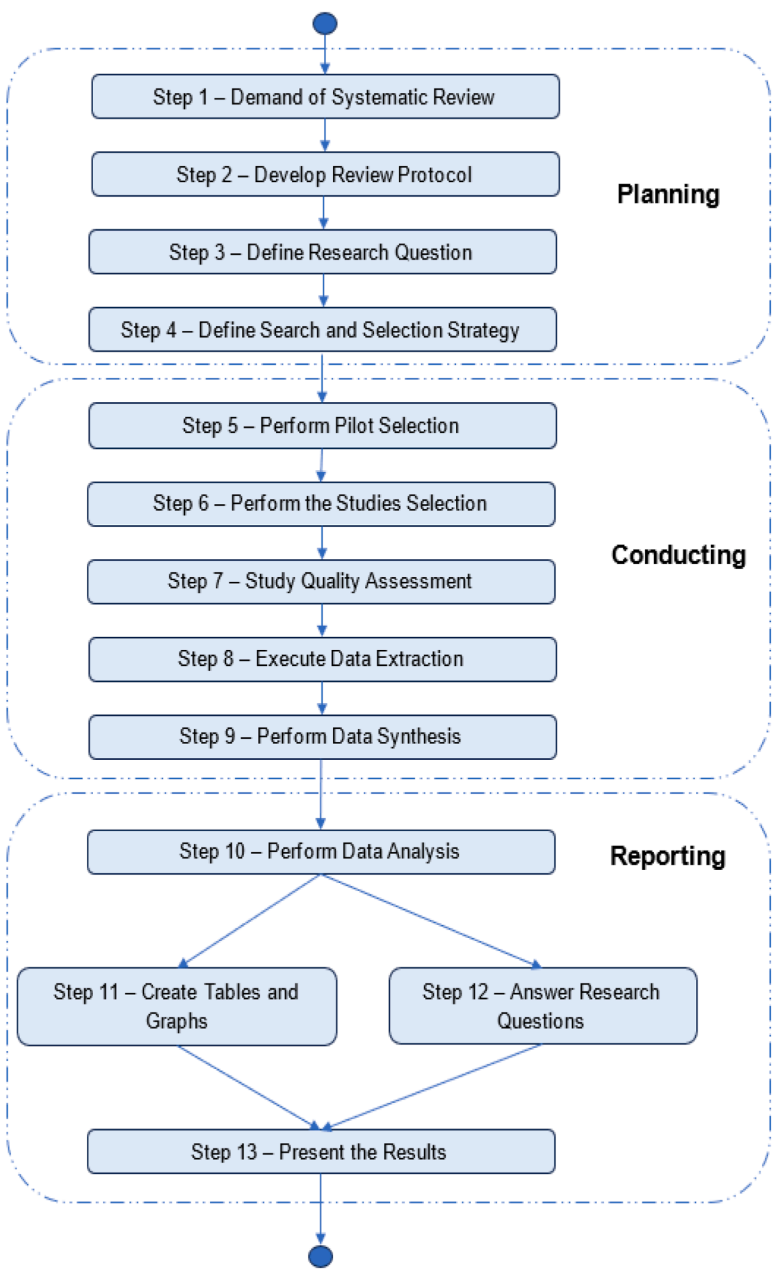


Fig. 1. SLR steps and phases.

We created a research protocol (Fig. 1, step 2) using the Parsifal online tool [24], which provides a template and facilitates interactive collaboration approach for researchers carrying out that task [25]. The protocol defines the strategies and criteria adopted during the SLR execution, ensuring a standardized SLR process, minimizing biases, and improving reproducibility [23].

To elaborate the research questions (Fig. 1, step 3) we applied the PICOC (population, intervention, comparison, outcomes, and context) approach as recommended by Kitchenham and Charters [22] and Wohlin et al. [26]. The PICOC terms are detailed in Table 2.

This work is part of a master's degree thesis for software validation and electronic systems integration for the automotive domain. Based on that scenario and the PICOC, we formulated the research questions shown in Table 3.

The search strategy for this study followed the guidelines of Kitchenham et al. [27] and included the recommended databases: ACM Digital Library, IEEE Digital Library, ScienceDirect, Scopus, and Springer Link.

To identify the relevant databases and create an appropriate search string, we conducted the Search and Selection Strategy phase (Fig. 1, step 4). A preliminary search in Google Scholar using general terms related to the topic of interest was performed to assess coverage in the research domain. We checked the results on the first four pages, looking for their potential relevance to our RQ's..

Since the automotive domain was previously defined in the PICOC context, the "automotive" term was included in the preliminary search string. Due to a large application of terms such as "quality" and "software" in many unrelated domains (e.g. domestic equipment, audio, and video), we refined the search string to include "automotive industry", which showed more accurate results in some digital libraries. After this initial exploration, we formulated a general search string -*"automotive industry" AND software AND ("quality assurance" OR "quality assessment") AND (test OR validation) AND (method OR framework OR model OR process"* - using an iterative process. We conducted trial searches to identify relevant studies and refine the search string based on their results. The general search string was adapted to the specifics of each database, as shown in Table 4.

Adaptations included using database-specific operators and filtering options, as follows:

- Selecting the "Full text" option for most databases, except Scopus, which uses a dedicated search method for title, abstract, and keywords.
- An initial search in Springer Link returned over 3000 papers. To refine the results, we filtered to "Articles and conference papers" only, which reduced the results to 602 candidate studies.

Table 2
PICOC for research questions and rationale for each element.

Population	Software testing, validation, and quality	In this study, we focus on software validation, tests, and quality phases within the automotive industry.
Intervention	Methodologies, processes, frameworks, models, methods, and techniques	Refers to the environments and methods applied, such as methodologies, processes, frameworks, models, methods, and techniques
Comparison	Not applied	Not in the scope of this SLR.
Outcome	Quality assessment characteristics, performance indicators, and validation impacts	Expected results include quality assessment characteristics, performance indicators, and impacts on validation.
Context	Automotive industry	The application domain is restricted to the automotive industry.

Table 3
Research questions and related objectives.

ID	Research question	Specific objectives
RQ1	What are the approaches involved in the quality assessment during software data validation in the automotive industry?	To identify and classify the software quality approaches based on proposed studies.
RQ1.1	How are the approaches applied in the software data validation?	To identify how the approach is applied based and where the application focus is placed.
RQ2	Which are the main quality assessment characteristics for software data validation in the automotive industry?	To extract the main aspects or characteristics taken into account in each study.
RQ3	What are the impacts of such characteristics in the validation, releasing and using phases of the software data validation process?	To identify and understand the influences of quality approaches on the software lifecycle.

Table 4
Final search strings resulting from our preliminary search.

Name	Search String	Results	Filter
ACM Digital Library	`"automotive" AND "software data" AND (test OR validation OR verification) AND ("quality assessment" OR "quality assurance") AND (method OR framework OR model OR process)`	26	Full year and text
IEEE Digital Library	`automotive AND quality AND software data AND (test OR validation) AND (method OR framework OR model OR process)`	89	Full year and text
Science@Direct	`"automotive industry" AND "software data" AND quality AND (assurance OR assessment) AND (test OR validation) AND (framework OR process)`	69	Full text and year
Scopus	`automotive AND software AND (test OR validation) AND ("quality assurance" OR "quality assessment") AND (method OR framework OR model OR process)`	77	TITLE-ABS-KEY
Springer Link	`"automotive industry" AND software AND ("quality assurance" OR "quality assessment") AND (test OR validation) AND (method OR framework OR model OR process)`	602	1. Full year, articles and conference papers 2. "Preview-only" selected

- In ACM and Springer Link, the original term "software data" to be too narrow and replaced it with the broader term "software". In the same way, we replaced "automotive industry" with the term "automotive" in IEEE, ACM, and Scopus.
- We added the keyword "verification" in ACM, to make the query more specific and retrieve additional studies.

We did not limit the studies by publication period in the search phase. We found out, in our preliminary searches, that the quality assessment for safety-related software has been reported from 1990 onwards. The first study concerned with safety in control functions is dated 1993, and out of 28 studies found before 1993, none focused on software quality assessments. Thus, we applied the exclusion criteria "E2 - Older than 1990s" during later screening. A complete list of our inclusion and exclusion criteria is provided in Table 5.

Table 5
Inclusion and Exclusion Criteria.

ID	Inclusion Criteria
I1	Primary studies;
I2	Papers that consider software data validation/test phases in the automotive industry AND:
I2.1	Discuss methodologies, processes, frameworks, methods, tools, or models of quality assessment/assurance for software data; OR
I2.2	Papers that address the impacts of quality assessment on software data
ID	**Exclusion Criteria:**
E1	Duplicated studies;
E2	Older than 1990′s;
E3	Papers that consider software development but not in automotive industry;
E4	Papers not related to software validation/test phases;
E6	Papers that are not articles from journals, conferences or magazines
E7	Papers that are not related to vehicle-embedded software
E8	Conference Proceedings
E9	Papers not written in the English language
E10	Not accessible in specific databases

3.2. Conducting

To assess the understanding of inclusion and exclusion criteria among the researchers, we performed the Pilot Selection (Fig. 1, step 5) - two rounds of individual selection and agreement check with a sample of 20 and 24 random candidate papers, respectively. We held meetings with all researchers aiming to identify the divergences and harmonise the criteria understanding. We performed two additional discussion rounds to clarify divergences and doubts regarding study selection. In the first one, we found cases within the automotive domain involving software validation, but not specific in-vehicle embedded software, which is the target of our research. For example, some papers addressed the QA of external sound equipment installed in the vehicle, which is out of scope. To solve this issue, we added the exclusion criteria "E7- Papers that are not related to vehicle embedded software" and "E8-Conference Proceedings". In the second round, we excluded extra papers based on the criteria "E9- Papers not written in English language" due to the low relevance of only two non-English papers identified.

To assess the agreement level among the researchers and ensure a rigorous selection, we computed the inter-rater agreement proposed by Landis and Koch [28]. The results were:

- Pilot selection 1 ($n = 20$): k-factor = 0.342 (Fair).
- Pilot selection 2 ($n = 24$): k-factor = 0.750 (Substantial).

After the first round, referred to as "pilot selection 1″, we reviewed and discussed the disagreements. Out of 20 randomly sampled studies, five disagreements occurred due to unclear inclusion/exclusion criteria. Specifically, the validation or test phases for automotive embedded software were not always clearly distinguished, and other out-of-scope cases, e.g., software applied in a vehicle assembly line or software for supporting vehicle development processes. After the researcher's discussion and refinement of exclusion criteria "E4" and "E7", we performed the "pilot selection 2", resulting in a substantial improvement in agreement.

We then started the Studies Selection phase (Fig. 1, step 6). We screened the candidate papers by reading the title and abstract, applying the inclusion and exclusion criteria listed in Table 5. In case of uncertainty, a full text screening was conducted. Out of 863 candidate papers, we included 157 (18,2 %) based on title and abstract. Table 6 describes the distribution of included and excluded studies by electronic database.

We further excluded 84 papers in full text screening that met the exclusion criteria E7 - "Papers that are not related to vehicle embedded software" and/or E4 - "Papers not related to software validation/test phases". Table 6 also details the distribution of final selected studies after the full text screening in the second selection round. We presented below some examples to illustrate the researchers' rationale:

Table 6
Selected studies by title and abstract.

Source	Candidate papers (database search results)	Included by screening title and abstract	Included in full text screening
ACM Digital Library	26	2 (7.7 %)	2 (7.7 %)
IEEE Digital Library	89	52 (58.4 %)	23 (25.8 %)
Science@Direct	69	5 (7.2 %)	1 (1.4 %)
Scopus	77	35 (45.5 %)	17 (22 %)
Springer Link	602	63 (10.5 %)	30 (5 %)
Total	**863**	**157 (18.2 %)**	**73 (8.4 %)**

- RLC circuits testing, verification, and validation (E7). Papers strictly focused on hardware and without qualitative contributions for the scope of SLR.
- Hardware requirement approaches (E7 and E4). The requirements for hardware, presented in some studies, did not have a relevant impact on software validation approaches. We opted for selecting software-specific studies when they cite the requirements for software in vehicle-embedded systems and present impacts on software quality assessments (I2.2).
- Description of software tooling quality evaluation (E7). The Software Assessment (SA) approaches for test tooling are out of this SLR scope.
- Data science solutions for IT software (E7 and E4). The data science application is not in scope unless the focus is placed on automotive-embedded software evaluations during validation phases.
- Software quality processes applied for production plants (E7 and E4). Several production software solutions are important for the software quality lifecycle, and they are out of SLR scope because of their contributions after software validation phases and release for series production.

As proposed by Kitchenham and Charters [22], we performed the Quality Assessment (Fig. 1, Step 7), aiming for an improved interpretation of selected studies. The quality assessment questions presented in Table 7 are based on SLR objectives and aimed to identify relevance for selected research questions (RQ's). The rationale behind those QA questions is described in Appendix A. The possible answers are 1 point (or "Yes"), 0.5 points (or "Partially"), and zero points (or "No").

A quality assessment of the selected studies was carried out after the full-text screening. To reach proper representativeness and quality results, we adopted a cutoff score greater or equal to 6.0. Thirteen out of 73 candidate papers below this threshold were further excluded, many of which do not fulfil our quality assessment questions Q6, Q7, and Q8. Fig. 2 shows the distribution of studies by achieved score.

Table 7
Quality Assessment Questions.

ID	Question
Q1	Is the study related to the proposed SLR objectives?
Q2	Does the study bring relevant information for answering research questions?
Q3	Is the quality application field clearly described and applicable to vehicle embedded software? (RQ1)
Q4	Are the methodology, processes, models or framework clearly presented? (RQ1.1)
Q5	Is there any technique comparison included? If yes, is that clearly presented and described?
Q6	Does the study present any real use-case investigation or is applicable for? (RQ2)
Q7	Does the study bring consistent results regarding the presented technique(s) approach? (RQ3)
Q8	Does the study present success, unsuccess examples or issues during the software validation phase?
Q9	Is the related study applicable to the commercial vehicles segment?
Q10	Does the study bring any coverage of quality issues related to software validation lacks?

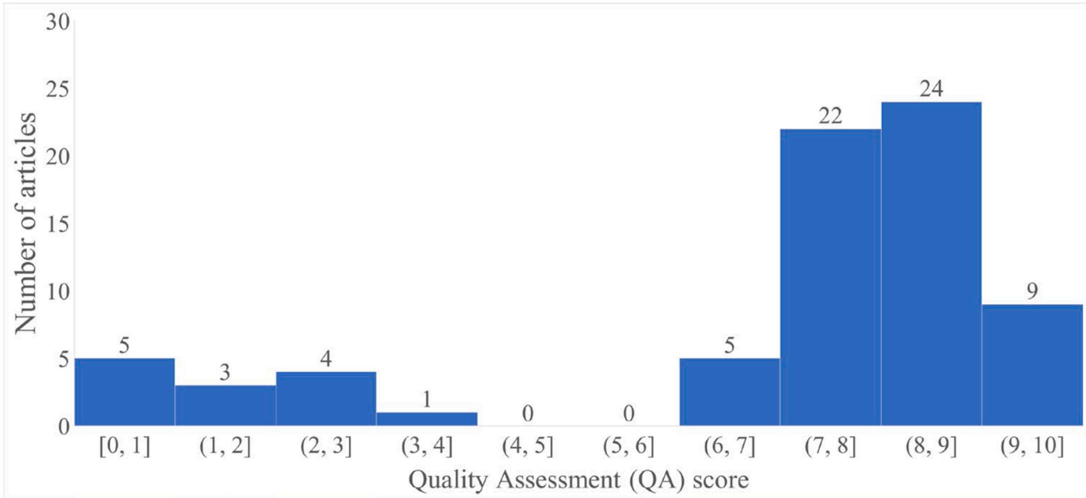


Fig. 2. Distribution of papers based on the achieved score.

Fig. 3 shows the quality of included and excluded papers by publication year. Papers published until 2005 accounted for <7 % (3 papers) of our dataset, whereas a significant concentration of studies emerged in the last decade. We observed an increase in high-quality studies in recent years, possibly linked to stricter regulatory requirements and/or the growing maturity of research fields such as safety-critical and cybersecurity domains. The growth trend identified cannot be fully confirmed for the last period (2021–2024), as the period is not yet complete, and the collection for the studies concluded in the first quarter of 2024.

The application of the protocol is summarized in the flowchart in Fig. 4. The specific data for each database is described in Appendix A.

We then carried out the Data Extraction phase (Fig. 1, step 8), considering a set of data characteristics (presented in Table 8) linked to the research questions, here called properties (from P1 to P8). An additional property, P9 was added later due to the relevance of new technology trends for software data assessment.

- **Proposed solutions (P1):** The proposed solutions for software data validation were categorised in the following groups: case study, framework, methodology/method, model, process, technique, and tool.
- **Software Assessment type (P2):** We analysed the focus of studies in terms of quality, safety, security, performance, or any other motivation for a proposal development solution.
- **Software Assessment phase (P3):** The objective here is to discern the software lifecycle phase, in accordance with Sommerville [29], within which the software assessment is applied: concept, design (including implementation), validation, development phases (except maintenance), or complete lifecycle coverage (including maintenance).
- **Involved software data source (P4):** To understand the possible paths and origins of software data, we extracted existent sources from which the software data is validated. Sources found dataset, dynamic tests, static tests, modelling/simulation, source code, and specification.

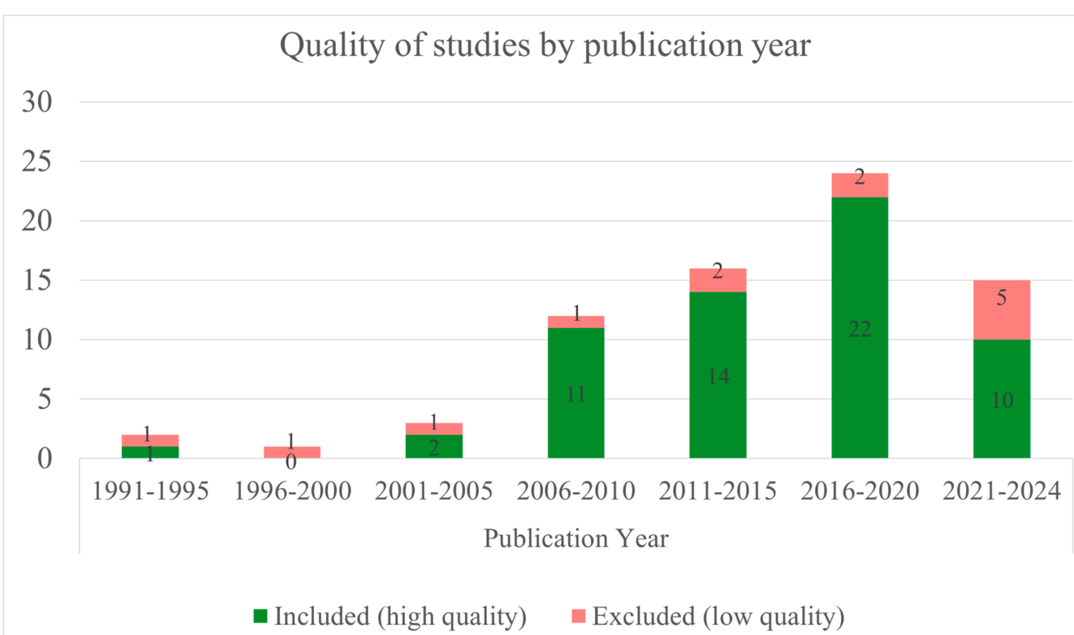


Fig. 3. Quality of studies by publication year.

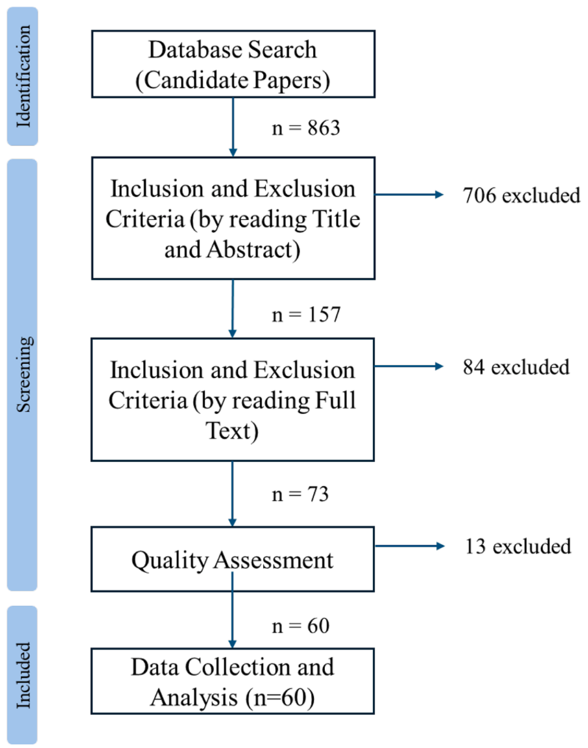


Fig. 4. Summarized protocol application.

Table 8
Extraction list for properties.

ID	Property	Research Question
P1	Proposed approaches/solutions	RQ1
P2	Software Assessment (SA) type	RQ1
P3	Software Assessment (SA) phase	RQ1.1
P4	Involved software data source	RQ2
P5	Other industry areas, segments, or product types than automotive also cited	RQ1.1
P6	Main software characteristics considered	RQ2
P7	Main assessment characteristics source	RQ2
P8	Main impacts of quality assessment	RQ3
P9	Application of Artificial Intelligence (AI), Neural Network (NN) or Machine Learning (ML) techniques	New technology trends

- **Cited industry areas, segments, or product types others than automotive (P5):** The Software Assessment (SA) in the automotive domain can bring relevant approaches valid for many other domains. This fact moved us to check which are the most cited in the selected studies. The possible categories are aerospace, avionics, chemical, automation, rail systems, nuclear, and software.
- **Main software characteristics considered (P6):** The target is to identify the software data characteristics described in the studies during the SA phase. We define such characteristic groups by applying the coding after the data extraction, without predefined categories. We conduct this analysis to support us in answering RQ2 properly.
- **Main Quality Assessment (QA) characteristics source (P7):** Similar to P6, but with the focus on identifying the origins and references of the studies to perform a software data assessment approach.
- **Main impacts of Quality Assessment (QA) (P8):** The assessment performed with a target to validate a software, in general, results in positive impacts such as cost, safety, security, error mitigation, performance, and user experience.
- **Application of Artificial Intelligence (AI), Neural Network (NN) or Machine Learning (ML) techniques (P9):** We intend to check how much the application of such techniques is cited to improve the software quality and provide support to the developers to mitigate the failures in the field.

The categorized groups and their definitions were as stated by the authors of the primary studies. Our classification was based on findings, and we did not intend to re-classify them into different definition groups, rather, we retained those stated in the primary studies. Appendix A lists the extracted terms for property groups used in the data extraction phase.

For the Data Synthesis (Fig. 1, step 9), we applied qualitative analysis and inductive coding to analyse the findings of the studies [30]. The inductive coding process allows the creation of codes directly from the collected data, instead of relying on a predetermined coding framework [31]. Each code represents similar textual contents in the studies and groups them according to the assigned category. When necessary, a second round of coding was conducted to merge similar codes into a common final categorization. Fig. 5 shows an example of the coding process.

3.3. Reporting

The Reporting phase (Fig. 1, steps 10 to 13) are presented in sections 4 (Results and Analysis), 5 (Discussion, Challenges, and Trends), and 6 (Conclusion).

3.4. Threats to validity and limitations

The term “quality” is largely applied to several areas of knowledge. In the software data validation and quality assessment process domains, we noted a large variation in how this term is used. The complexity and potential restrictions of the defined string may also have impacted the identification of relevant papers. We attempted to mitigate these risks by increasing the rigour during the studies selection phase (step 6), performing meetings and rounds of discussion among the researchers to reach a more assertive result. In the same way, the decision to state a publication date limitation after the title and abstract readings contributed to accessing all available papers and checking their real relevance before any exclusion step.

During the first selection phase, we noted several cases of quality assessment analysis focused on testing tools, which is out of our research scope. To exclude these papers, we added the exclusion-related criteria for non-embedded software-related papers (E7 in Table 5). This solution mitigates but does not eliminate completely the chance of finding studies mentioning testing tools in the context of embedded software quality assessment. In such cases, we opted to keep the papers that contributed to answering the research questions.

In Section 4 - Results and Analysis, we used classification terms as stated by the primary studies. That means our classification was based on direct findings, and we did not reclassify the studies based on our judgment. We recognise that overlaps or similarities among the different terms may introduce biases originating in the primary studies, which could reflect in our results.

4. Results and analysis

A total of 60 studies were included and considered for data extraction.

4.1. Software quality approaches based on proposed studies (RQ1)

The main software quality approaches stated in the included studies are Techniques (25 %), Methods (21.7 %), Frameworks (18.3 %), Models (16.7 %), Processes (6.7 %), and Case studies (5.0 %). Other

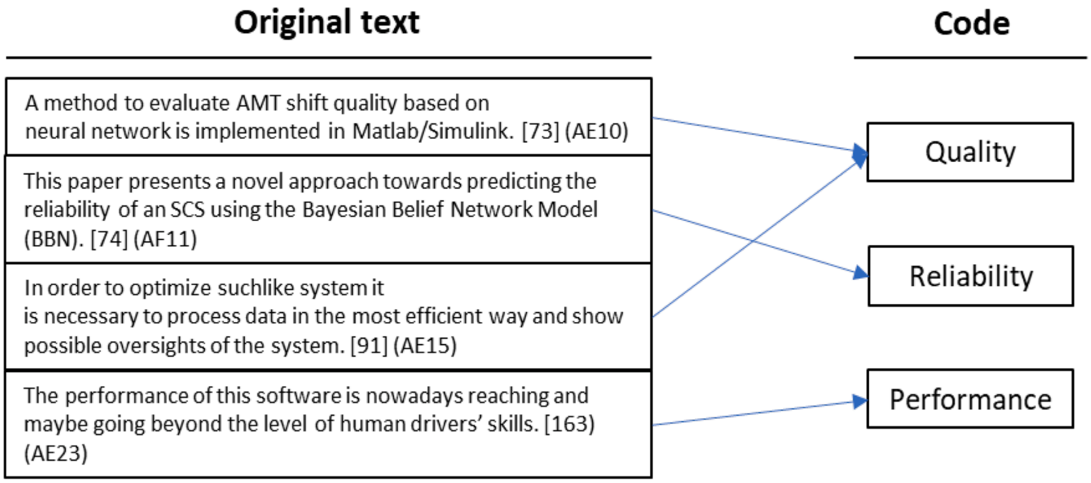


Fig. 5. Coding process examples for P2.

categories (Methodology, Standard, Systematic Mapping Study, and Topology) were mentioned just once (1.7 % each).

The predominance of Techniques and Methods (together over 45 %) suggests a practitioner-driven focus on implementation-ready tools. Frameworks and Models are academically grounded, but they might demand higher integration or customization effort. This indicates a prioritization of practical solutions in safety-critical automotive environments. The complete number of studies per software quality approach is illustrated in Fig. 6. **Models.** Models, as described by Takrouni et al. [32], are based on other already existing models to extend the applicability and cover the complexity of modern vehicles. This is possible through the addition of modules, focused on desired coverage areas, such as, reliability, safety, cyber security, cost optimization and others.

Techniques. Mateen et al. [33] propose a technique for an analysis work to provide a better way to evaluate the software quality level, having the comparison between manual testing vs automated testing aiming the minimum error rate within the time and lower target costs. Techniques based on software prediction are also cited in [34–37]. Ranasinghe et al. [35] emphasise that the quality of intelligent autonomous decision-making can be reached by applying techniques of the fusion of sensor data and historical health-state information.

A mix of software quality approaches could be also found in [38], comprising failure detection via two diagnostic protocols, vehicle data measurement, fault monitoring, and unexpected behaviour analysis. The authors proposed an application method based on existing methodologies of sensor networks and stream processing to optimise the amount of data to create a method for selectively preprocessing and recording measured data, which is the main intended target of their study. Along the same line, Touw [12] performed a case study merging three assessment models (CMMi, TMMi and A-SPICE) to establish a new model taking account of the quality aspects of complex systems.

4.1.1. How the software quality approach is applied and where the application focus is placed (RQ1.1)

The purpose of this research question is to understand how the software quality approach was applied in terms of the SA type domains. We found six different target groups, as illustrated in Fig. 7, many of them focused on a single quality characteristic. Several studies considered a combination of quality characteristics coming from multiple domains that bring a better compromise among performance, time, cost, use of technology limits, improved user experience, and others. For example, Sadio et al. [39] defined quality based on proper detection of Wi-Fi power thresholds, which improve customer experience and satisfaction.

Fig. 8 points out Safety and Performance as the most relevant characteristic for software quality evaluation, independent of the adopted approach. In the automotive domain, those attributes are often interrelated. For example, the capacity of a front radar for autonomous systems to maintain the coverage angle against powder, snow, exhaust gases, water, and fog is directly related to the safety-critical system operation. At the same time, a safety assessment for safety-critical

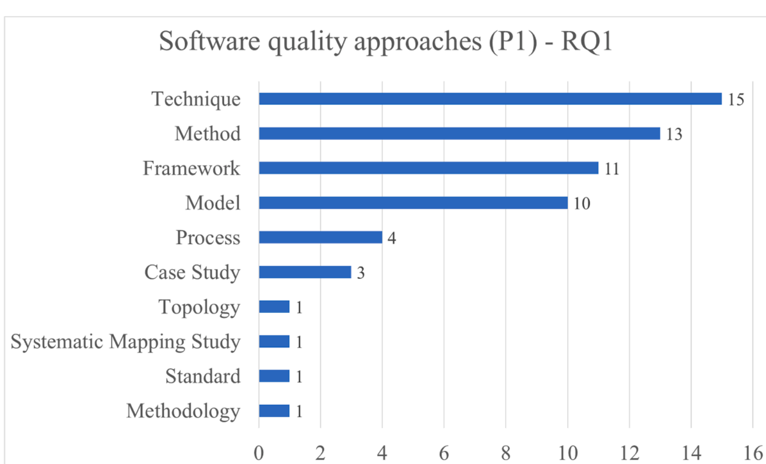


Fig. 6. Distribution of studies by software quality approach.

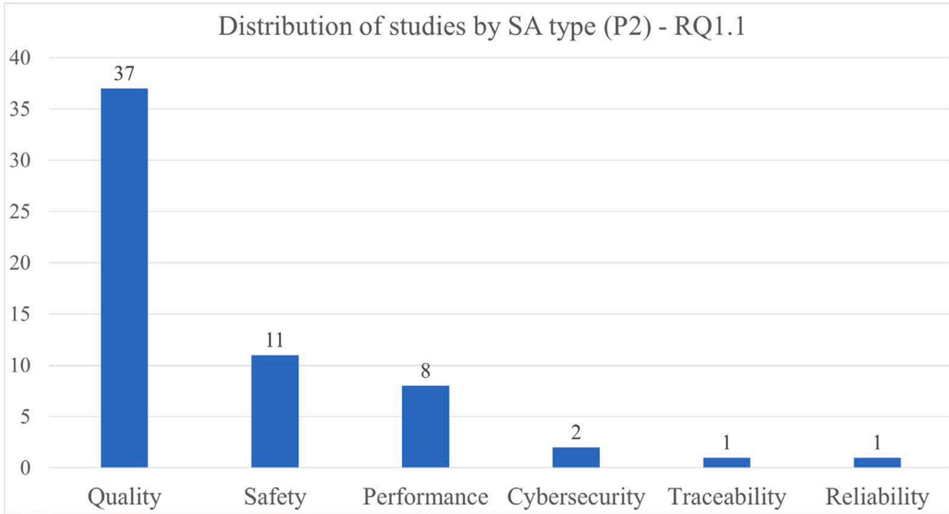


Fig. 7. Distribution of studies by SA type.

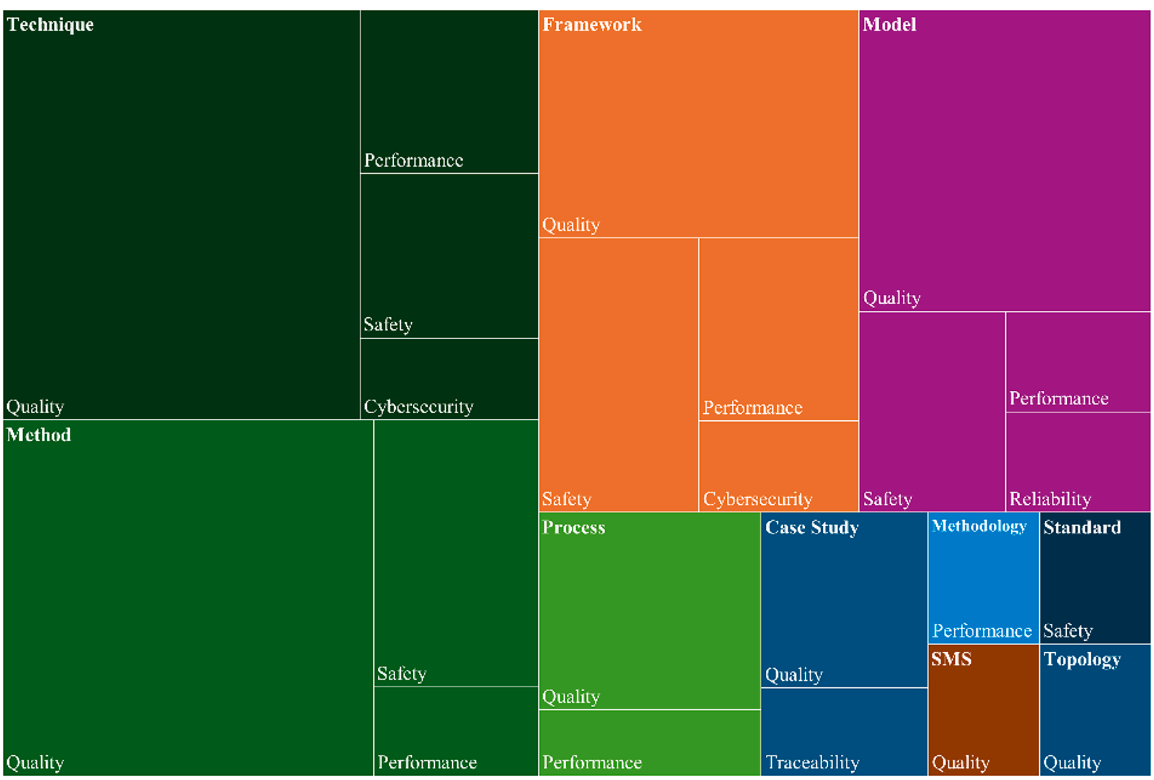


Fig. 8. Distribution of Assessment Focus (P2) by Software Quality Approaches (P1).

systems requires that the correct sensor data, processed by the software, is correct and available.

Taking account of the classification by SA phase (P3), we identified the software lifecycle phases in which software QA is placed (shown in Fig. 9). Not surprisingly, most of the studies are focused on the Validation phase (75 %) but others also mentioned Concept (11.7 %), Full lifecycle (5.0 %), Software development (3.3 %), among others. The predominance of Validation phase suggests that QA is often placed between Design and Deployment phases. This may reflect industry practices where late-stage validation is still a primary gatekeeper for compliance and cost, despite calls for shift-left approaches.

We also examined data sources for software QA (P4) to identify the origins of validation data, as illustrated in Table 9. Ten studies (16.7 %) did not describe clearly the source of their SA data. Some reasons for that are their focus on (1) the effects of the adopted approach or presented solution [40], (2) the comparison of approaches [33,36] or other similar targets. Other studies [41–45] reported more than one source of data.

The most relevant software quality approaches (techniques, methods, frameworks, and models) represent 81.7 % of the total selected studies. These approaches can often be implemented in combination, as proposed by [11,32,38]. The main reported motivation for adopting combined approaches is the need to fit them to a specific new demand such as increasing systems complexity [13,44,46,47] or cost [36,48,49]. Oka et al. [50] describe another solution that integrates tools and test cases. The proposed process uses information from Application Lifecycle Management (ALM) tools to configure the testing tools and, finally,

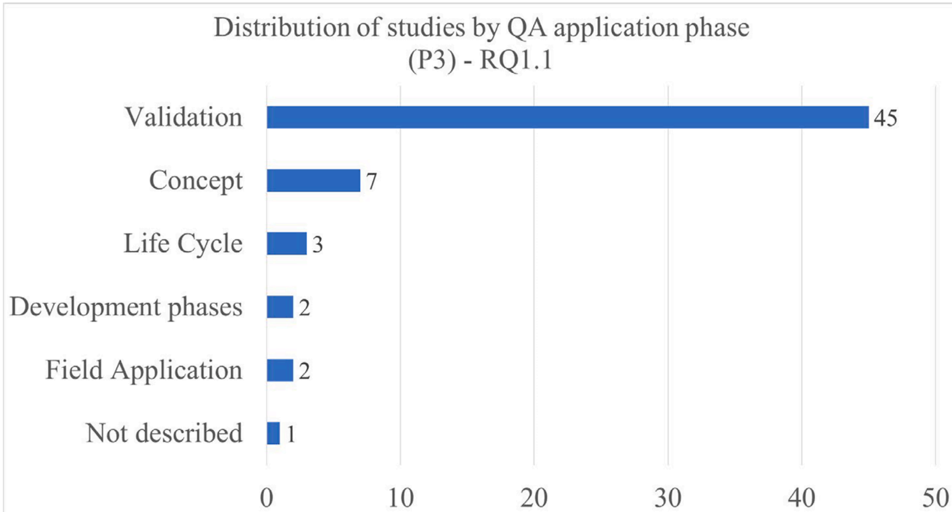


Fig. 9. Distribution of studies by software QA application phase.

Table 9
Software data sources.

Software data source	Occurrences
Specification	21 (35 %)
Dynamic test	13 (21.7 %)
Not clearly described	10 (16.7 %)
Dynamic and Static tests	3 (5 %)
Source code	3 (5 %)
Data validation set	2 (3.3 %)
Modelling	2 (3.3 %)
Static test	2 (3.3 %)
Dynamic test and Specification	1 (1.7 %)
Historic fault data	1 (1.7 %)
Simulation	1 (1.7 %)
Static test and Specification	1 (1.7 %)

collect the test results back to the ALM tools.

Similarities with other sectors (P5), such as aerospace, avionics, rail systems, chemical, and automation were cited by four (6.6 %) studies [11,35,40,51]. The reasons for this are the increasing relevance of safety in automotive software applications (e.g. autonomous driving), cyber-security requirements, high-voltage applications for battery-electric vehicles, and new standards linked to connectivity and over-the-air applications.

Furthermore, embedded software (48.3 %) leads the software data assessment in terms of product type segment, as illustrated in Fig. 10. In second place, autonomous systems (16.7 %) are increasing in relevance due to higher efforts concentrated over the last years, as pointed out by [7,13,14,35,37,52–56]. The electronic control system (13.3 %), and Powertrain (6.7 %) completed the most relevant product type segments. The dominance of embedded systems suggests that a deterministic behaviour could be more suitable for QA. The growth of sophisticated software products to attend very complex computer systems requires that QA starts as early as possible, thus reducing the correction costs [17, 36,42,57,58].

In Fig. 11, the Validation and Concept phases lead to the software Development phases, but their relevance is not the same from a product type perspective. The presence of the Concept phase for the Embedded Software, Electronic Control, and Autonomous Systems indicates that the assessment focus is also centered on base software or code, instead of the Powertrain. Assessment of Powertrain primarily involves evaluations of acceptance characteristics, like gear shifting software characteristics (comfort, shifting speed, gear scratches), more traditional mechanics evaluation approaches, and driver's feeling. Assessment of Electronic Control products focuses on microcontroller management, memory, drivers, standards, and proprietary communication. In the Autonomous Systems, the balance between Validation and Concept is expected to increase in the next years due to the importance of safety-critical systems issues being addressed early. Rework and additional costs, due to late identification of defective software, can be a no-go for

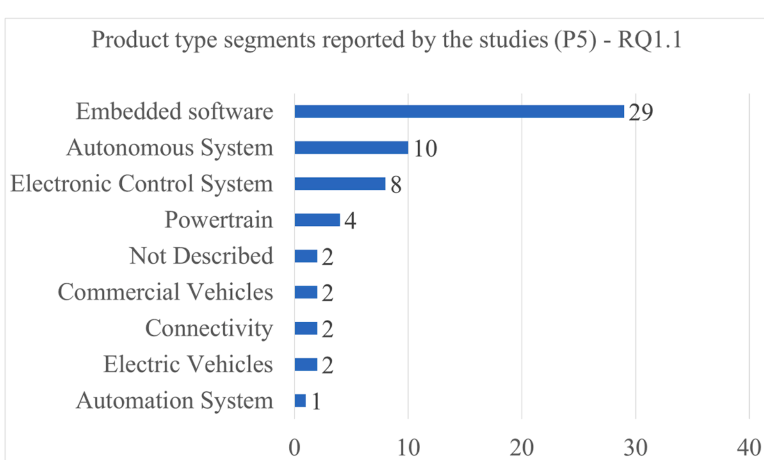


Fig. 10. Product type segments reported by the studies.

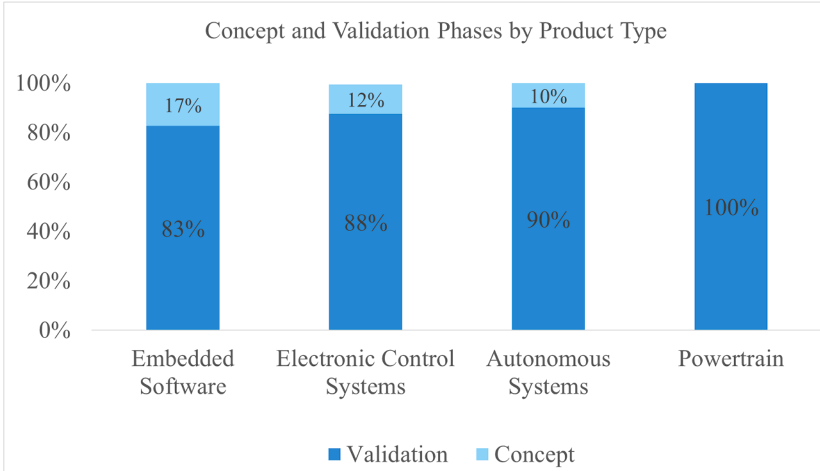


Fig. 11. Concept and Validation Phases by Product Type.

the start of production releases.

4.2. *Main aspects or characteristics considered in each study (RQ2)*

The main assessment characteristics considered in QA for software data validation are the scope of this research. Furthermore, we also intended to identify the origins of such characteristics and understand their relationship (P4, P6, and P7).

Out of 60 included studies, 17 (28.3 %) mentioned code quality. The main cited characteristics are Reliability, Maturity, Maintainability, Fault Tolerance, Availability, Safety, Cybersecurity, and Security. Despite the automotive domain's particularities, they are mainly placed on the requirements specification, not in the software. This means that a QA for software data validation can be established by generic software quality standards such as ISO/IEC25010, where Functional Suitability can also cover the Functional Safety (FuSa) automotive-specific requirements [46].

From the characteristics illustrated in Fig. 7, we can highlight that Safety and Cybersecurity have been increasingly cited in recent years. A potential explanation for this is the spread of regional regulations implementation, such as UN-ECE R155 for Cybersecurity and UN-ECE R156 for software update management, and new standards, such as ISO26262 – Functional Safety, and ISO21434 – Cybersecurity, to establish a baseline for new complex upcoming technologies such as autonomous driving, e-mobility applications, and telematics. Furthermore, other cited characteristics (code quality, misfunction, performance, communication quality, data accuracy, data interpretation, and software health) will also be affected by those standards and regulations.

In terms of data sources, we identified the following origins: specification with 22 occurrences (36.7 %); requirements with 14 (23.3 %); measured data with seven (11.7 %); fault data with three (5.0 %); simulated with two (3.3 %); CAN databases, emulated data, source code and stored project data with one (1.7 %) occurrence each one. Eight studies (13.3 %) did not report a specific data source or did not simply emphasise that. This could be because some internal documents are confidential or protected by non-disclosure agreements.

Lastly, but not less important, we tracked software data sources. Fig. 12 shows the distribution of the occurrences for each software data source. The majority of studies concentrates in specification (21 occurrences, 35 %) and dynamic and/or static tests (20 occurrences, 33.3 %). Ten (16.7 %) of selected studies did not clearly mention the applied source for QA. The causes for this may be the same as described above for reported data sources.

4.3. *The impacts of QA approaches on the software lifecycle (RQ3)*

It may be obvious that the main impact of a well-performed software QA process is improved quality itself. As expected, the word "quality"

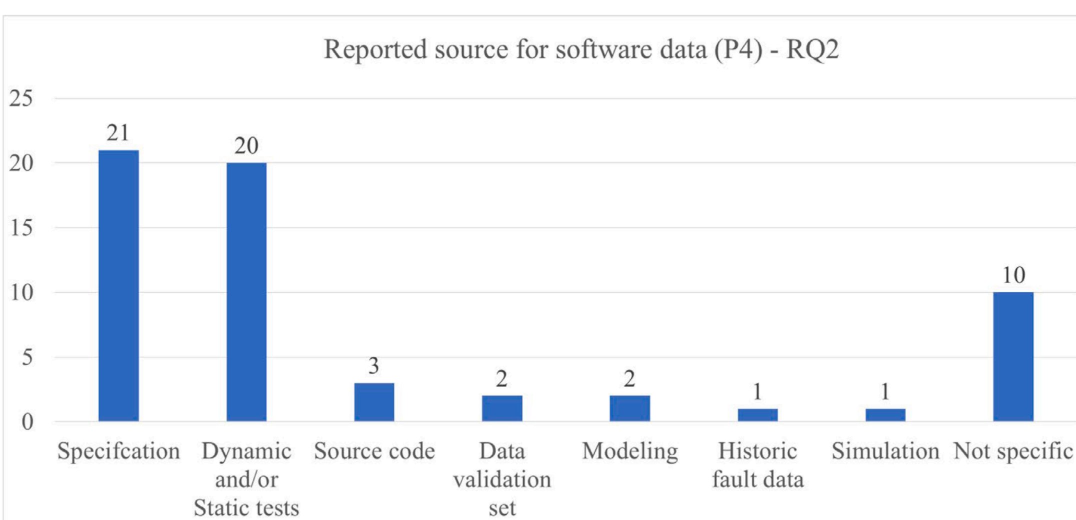


Fig. 12. Reported sources for software data.

was the most cited impact, it was reported by 22 studies (36.7 %).

We aimed in RQ3 to identify other potential impacts and understand how they are perceived. Thus, we analysed additional cited terms beyond "quality". The complete overview of the reported impact distribution is presented in Fig. 13.

The reported techniques, frameworks, and methods are mostly applied to safety-critical systems. Autonomous systems are predominant (e.g., brake assist, adaptive cruise controller) as well as safety-related embedded software products (ISO26262 compliance). These reported characteristics reinforce the concern of automotive software developers in recent years. The complete distribution by approaches is presented in Fig. 14.

Table 10 presents the comparison between the SA type (P2) and the main perceived impacts on Quality Assessment (P8). As illustrated, there is a clear link between the main perceived impacts (P8) and the software quality assessment type (P2). Ranking first in both is quality, followed by safety, and performance in third. The key messages here: 1) Efforts can be applied directly to collect the desired outcomes (quality, safety, performance, and cybersecurity); 2) Side effects are not restricted to the directly aimed SA type, the benefits extend further; and 3) QA can also have positive impacts on other aspects, such as user experience, efficiency, and cost.

The growing emphasis on cybersecurity aligns with regulatory shifts such as UNECE-R155 and UNECE-R156. This points out concerns with public trust and liability management. Despite their lower impact ranking on QA (7th position), cybersecurity topics are now more cited than UX as SA types, revealing an increasing priority gap between security and user-centred validation.

In terms of cost and efficiency, even though they are not directly cited in the SA phase as primary targets, they remain an integral part of the entire software lifecycle and listed by studies as perceived impacts in the 5th and 6th positions, respectively.

4.4. Application of new techniques and their benefits for software quality assessment

Out of 60 included studies, nine (15 %) mentioned at least one of the following techniques: artificial intelligence (AI), machine learning (ML), and neural networks (NN). Those upcoming trends are also relevant to mention.

In terms of product type segment, four out of nine (44 %) focused on autonomous systems. These studies emphasized the validation complexity and the impact on fulfilling functional safety requirements [7,14,35,37]. Two studies (22 %) addressed Powertrain systems, highlighting relevant challenges in the integration among electronic, electrical, and mechanical components (e.g., automated manual transmissions). Electric vehicles were cited in one study (11 %), as was general embedded software (11 %).

Regardless of the intended characteristic being evaluated in an SA phase, AI, ML, and NN can be widely applied. Some examples of their use are as follows:

Artificial intelligence (AI). In addition to conventional vehicle tests, AI can be used for automated test generation and creation and failure-revealing scenarios for safety-systems evaluation, as reported by [37].

Machine learning (ML). System-healthy assessment can be optimised by ML models, as pointed out by Ranasinghe et al. [35]. For example, ML has been used for state of charge (SoC) battery estimation in blind modelling tools [41]. Another example is the use of data-driven ML for dual-clutch transmission software, where input data - including actuator constraints and performance/comfort characteristics – are used to evaluate optimal gear-shifting settings [34].

Neural network (NN). A variant called Recurrent Neural Network (RNN), described by Kollmeyer et al. [41], is a type of supervised ML method that uses short-term memory to process battery state of charge (SoC) quality metrics. It can be used for time-dependent applications with sequential datasets, where the goal is to apply the best-trained network in the next processing step. Another study [3] trained NNs with quality parameter metrics to evaluate the software quality and produce performance ratings for Powertrain applications.

The distribution of quality assessment characteristics by these techniques is as follows: Safety level with three occurrences (33 %), code quality and performance with two each (22 %), data accuracy and state of health one each (11,1 %).

These techniques are suitable for high-complexity systems during Verification & Validation (V&V) activities. However, their application present challenges for software developers, such as higher costs due to additional required efforts, the need for advanced infrastructure and expertise. Their focus on autonomous and other safety-critical systems makes sense, as these areas demand superior product quality and reliability that traditional validation processes cannot achieve [36]. Additionally, their scalability can mitigate undesired events during tests (e.g., accidents) and minimize the need for physical tests, reserving prototype vehicle trials in controlled or public road conditions only at higher software maturity levels.

In terms of applicability, these techniques can support several software and data quality **assessment** tasks. For software code, they aid in specification from requirements, code creation through reuse, identification of code deviations from fault storage data, test monitoring, and result analysis. For software functions, especially the safety-related ones, benefits include driver behaviour recognition (e.g., detecting fatigue, misuse of cell phones while driving, etc.). They can also extend into post-production, enabling quality experts to monitor the software quality performance in end-user applications and apply proper corrections when needed.

However, there are constraints. AI- and ML- techniques face challenges in meeting safety-critical requirements in real-world conditions [7]. This limitation prevents the exclusive application of those techniques without additional efforts and funding for complementary validation in the "real-life" scenarios. Furthermore, variations in product characteristics can affect system responses, requiring a worst-case evaluation to identify critical variants for testing and ensure a proper functional performance while mitigating safety and security risks.

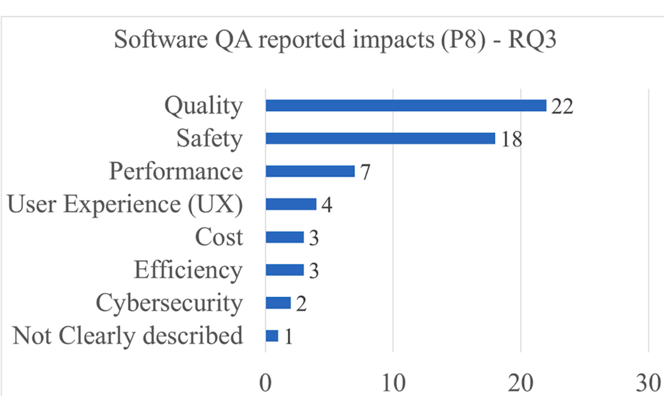


Fig. 13. Software QA reported impacts.

5. Discussion, challenges & trends

Many studies focus on quality-specific metrics but fail to comprehensively address all the characteristics of inherent and system-dependent software quality assessment as defined by ISO/IEC25012. While data accuracy, data interpretation (often referred to as "understandability"), and efficiency are frequently cited, terms like "code quality" can also align with this international standard as part of a more comprehensive software quality assessment approach. This is further supported by studies that identify safety (ISO 26,262) and cybersecurity (ISO21434) as key assessment characteristics, with "compliance" being

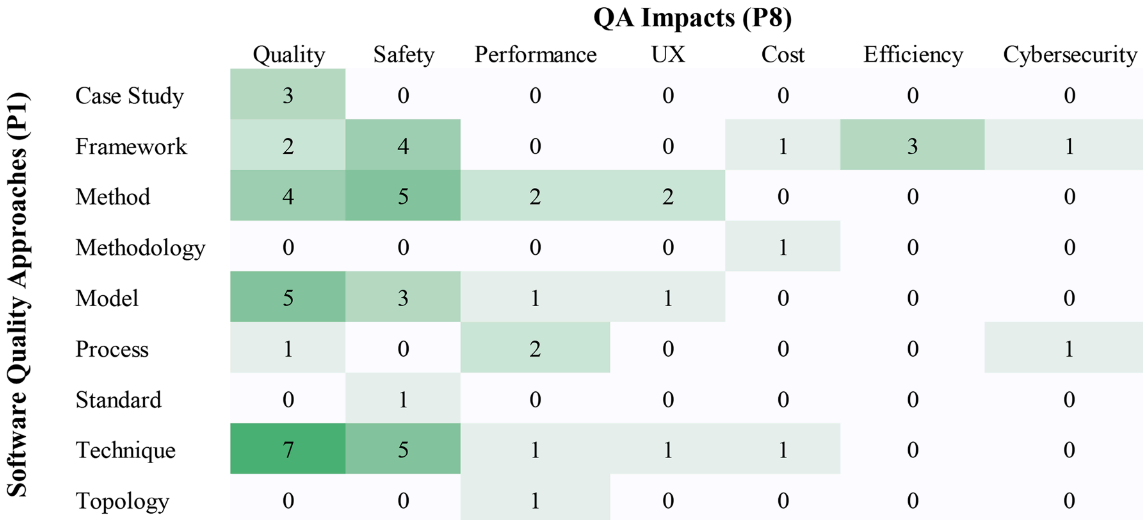


Fig. 14. Perceived QA Impacts (P8) by Software Quality Approaches (P1).

Table 10
Comparison between SA type (P2) and Main perceived impacts of QA (P8).

	P2 - SA Type	P8 - Main Impacts of QA
Quality	***1°***	***1°***
Safety	***2°***	***2°***
Performance	***3°***	***3°***
User Experience (UX)	-	4°
Efficiency	-	5°
Cost	-	6°
Cybersecurity	4°	***7°***
Traceability	5°	-
Reliability	6°	-

a particularly notable focus. Another group of studies emphasises portability, reliability, security, user experience (often referred to as "usability"), and performance, all based on the principles of the product quality model established by ISO/IEC25010. This model serves as the foundation for software requirements specification and quality evaluation.

Best practices in software development processes significantly enhance the efficiency of quality assessment, especially given that approximately 85 % of vehicle functionalities are software-based today.

We examined quality characteristics and the impacts of these assessment approaches on the software lifecycle, as detailed below:

How each approach is applied and where the application focus is placed. The distribution of software quality assessment approaches was concentrated in Techniques, Methods, Frameworks, and Models. Notably, these categories are not dependent on the specific application focus. The three most relevant aspects identified are: Quality, Safety, and Performance. Regarding the application phase, Validation was cited in 75 % of the 60 studies included in this review. Other phases, such as Conceptualization, Development, and Field Application, are expected to increase in relevance, particularly due to the complexity of cybersecurity and safety-critical systems that necessitate monitoring during the process to enhance failure prediction during vehicle operation.

Quality characteristics. Code quality was the most frequently cited characteristic, as the structure of the software code is critical to various aspects, including complexity, efficiency, execution time, and reliability. Additionally, code quality is intrinsically linked to other characteristics such as malfunction, performance, communication quality, data accuracy, data interpretation, and software health.

The other two significant topics identified were safety level and cybersecurity level, both of which are guided by standards like ISO26262 (Functional Safety) and ISO21434 (Cybersecurity). These standards are further regulated by regional implementations, such as UN-ECE R155 for Cybersecurity and UN-ECE R156 for software update management. They apply particularly, though not exclusively, to safety-critical features such as autonomous systems, electric propulsion, and Advanced Driver Assistance Systems (ADAS). Therefore, to ensure proper code quality, all the aforementioned aspects must be considered from the software concept phase. If these characteristics are neglected, they can negatively impact not only software quality but also safety, security, and compliance with regional regulations.

Impacts on the software lifecycle. Quality was identified as the most significant impact throughout the software lifecycle. Ensuring software quality over its lifetime involves focusing on traceability, applying quality tools (such as FMEA and FTA), and implementing safety approaches. The second most significant impact is Safety, which arises from software quality assessments that prioritise safety and performance. This is reinforced by functional safety requirements outlined in ISO26262. Additionally, User Experience (UX) was reported as an important factor, encompassing all user perceptions of the final product, which can greatly influence customer satisfaction. Attention to UX has increased in recent years with the incorporation of interactive solutions, such as digital and touch screens, connectivity features, and voice interaction.

The findings of RQ1 brought an overview of applied approaches, the most relevant targets, and motivations for an automotive software quality evaluation. The contributions of RQ1.1 provided the automotive software segment groups and identified the phases of software development where the approaches are concentrated. The RQ2 comprises the characteristics considered, the software source data (acquisition type), and the provided sources for those characteristics (requirements, standards, and/or specifications). Finally, the RQ3 improves the understanding of perceived impacts and results of a well-implemented automotive software quality assessment.

5.1. Challenges & trends

The studies also revealed several challenges and issues, such as:

- **Ensuring safety-critical requirements:** Addressing these challenges in real-world conditions, particularly with the introduction of new AI and ML techniques, requires identifying and clarifying safety and fail-operational requirements as the foundation for an effective testing strategy [7]. The requirements specification should ensure all intended application environments and cover the possible software variances during the entire lifecycle.

- **Adapting to rapidly changing software requirements and environments:**
- Frequent changes hinder alignment with standards and processes, leading to increased costs and efforts in correcting software defects. To facilitate early defect detection, a risk-based approach that identifies high-potential failures is essential. This should be supported by exploratory testing based on ISO 29,119 [59]. The software documentation, including requirements, code specification, tests, and related reports, must be maintained. Lifecycle modes, like agile, support the software development teams in this challenging task. This does not, by itself, prevent software issues but contributes to a structured basis for improved software traceability throughout the whole development phase.
- **Bridging the gap between simulated and physical systems:** Simulations are often constrained by what is known as the “reality gap.” This issue can be mitigated by incorporating real measurements into the simulation and validating the relevant components of the physics engine using modelling software [52]. Additionally, the use of digital twins - virtual representations of physical systems updated with real-time data - can further narrow this gap. It allows continuous validation of existing physical tests using real-world measurements in prototype vehicles or customer fleets (under an official customer’s agreement) via upcoming over-the-air technologies for data acquisition. The existing infrastructure may be insufficient to fulfil the requirements for the validation task, and complementary funding shall be considered only for “in-vehicle” equipment, not for the acquisition of infrastructure from scratch.
- **Improving quality assessment testing procedures and metrics:** The absence of clear software quality assessment and testing metrics in the automotive industry results in vulnerability issues. To address this, a weight-based approach targeting the most vulnerable software components is proposed [53]. The application of robust approaches based on quality, safety, and security standards, internal documented lessons learned, and state-of-the-art practices certainly contributes to improved software quality. Nevertheless, the accumulated knowledge, expertise, and adequate resources for continuous processes optimization are critical success factors for the long-term for software quality.
- **Reducing cost and effort in safety-critical embedded software quality assessment:** A set of software defect prediction approaches to mitigate these impacts is presented in [36]. Reducing the quality assessment does not mean, in principle, eliminate spending, but allocating it properly. As aforementioned, the implementation of a robust software validation process reduces the rework costs. In addition to that, flexibility for corrections, modularity for the addition of new software functions blocks, and applying the defect detection techniques as early as possible, can contribute to mitigating the final software validation costs.
- **Addressing problems in safety-critical machine learning systems:** Kuwajima et al. [14] emphasised the need for improved data-driven requirements specification and design. Both test data and training data should accurately reflect real application scenarios. Additionally, they propose future research to address gaps in current quality standards, such as ISO/IEC 25,000. As pointed out for the gap between simulated and physical systems, the use of real-world data can stimulate and teaching machine learning systems. The combination of measured data and simulated environments can be provided, for example, a more comprehensive scan of radars and cameras from an active brake assist system. The integration of such benefits can also provide a reduced development cost by shared resources.
- **Overcoming poor documentation and scalability challenges:** Software testing strategy decisions are often based on experience, and unclear textual specifications can negatively impact software quality. Therefore, improved documentation practices are essential for enhancing software quality management. An example is the use of tools for software and systems management, such as the Engineering Lifecycle Management (ELM) that provides support for requirements, tests, and design specifications. This integrated documentation environment enables the software development teams to deal with complexity, software compliance, and auditable documentation.

While these findings are centred on the automotive domain, they can also benefit other industries, such as aviation, electronic control systems, automation, railway transport, and autonomous systems, by enhancing their software quality processes through similar or adapted approaches.

6. Conclusion

Higher software quality positively influences customer acceptance of a product. Over the past 20 years, the term ’software quality assessment’ has evolved to include new assessment criteria to support the automotive software developers in managing growing systems’ complexity. The primary reason for this evolution in the automotive industry is the rapid expansion of complex technologies, including Advanced Driver Assistance Systems (ADAS), connectivity systems (such as over-the-air features), electric propulsion vehicles (e-mobility), battery electric vehicles (BEV), and hybrid electric vehicles (HEV). To effectively address the challenges posed by this complexity, the quality assessment must align with new requirements and standards. Additionally, a more comprehensive approach can enhance understanding and better equip software developers and testers to tackle the challenges.

This SLR identified 60 primary studies on automotive software quality assessment published over the past 30 years. It synthesized the findings about four main types of approaches (techniques, methods, frameworks, and models) and their focus across the software lifecycle, with Validation being the most commonly addressed phase (75 %). The review also mapped QA characteristics, with the most cited being code quality (28.3 %) and safety (20 %). Safety and cybersecurity (3.33 %) figure in alignment with standards such as ISO 26,262 and ISO 21,434. It also revealed that AI and ML techniques are emerging in the field, though currently applied in only 15 % of the studies, mostly in autonomous and powertrain systems. Finally, our SLR offers actionable guidelines (Section 5) for practitioners and researchers to improve future software quality processes by applying standards like ISO/IEC 25,010 and 25,012, and regional regulations such as UN-ECE R155 and R156.

We explored the software quality approaches used in the automotive industry, clarified their applications, and identified their specific areas of focus. Our results indicate that new software developers should adopt a more comprehensive quality approach throughout the entire software development process, beginning with the concept phase, to enable earlier detection of non-quality issues, reduce correction costs, and avoid showstoppers on vehicle assembly lines. The quality characteristics and impacts outlined in this SLR ensured through the use of quality standards (ISO25000 series), proper specifications and requirements, encompassing key pillars such as safety, cybersecurity, user experience, and performance. Additionally, regional regulations and their impacts on technical specifications must be considered during the conceptualization phase. Quality tools remain essential for the entire software development lifecycle. Furthermore, it is crucial to involve customers as much as possible in the conceptualization phase and to gather their feedback during the initial pre-series (pilot production) evaluations to enhance user experience (e.g., gearshift strategy for automated transmission software). The relevance of AI, and ML in managing complex technologies is evidenced, but their integration depends on traditional real-world validation, which introduces safety risks. The availability of expert workforce, infrastructure costs, and day-by-day improvements will likely define which organizations will stay and which ones will perish in this domain.

The SLR outcomes can be used as a guideline for software developers to implement and/or improve the automotive SA approaches by using documentation-supported tools (e.g.: ELM), focusing on the most relevant characteristics highlighted in state-of-the-art practices, and aligned with recommended quality, safety, and security standards.

We encourage the research directions focused on how the more comprehensive real-world data analysis can contribute to mitigate the existing test modelling and simulation gaps, especially for safety-critical automotive software (e.g., autonomous systems). Second specific research is a deeper investigation into the potential contributions of the over-the-air automotive data acquisition techniques for AI-based systems training activities, focused on software quality assessments. Furthermore, the research in the aforementioned techniques – AI and ML– to gain deeper understanding of future trends in supporting safety-critical quality assessments. Comparison with existing systematic reviews can support this goal. Future studies should aim for a more comprehensive quality coverage that includes characteristics not fully addressed by traditional standards, such as the ISO/IEC 25,000 series, through a comparative analysis of emerging studies and traditional approaches.

CRediT authorship contribution statement

Gilmar Pagoto: Writing – review & editing, Writing – original draft, Methodology, Investigation, Conceptualization. **Luiz Eduardo Galvão Martins:** Writing – review & editing, Validation, Supervision, Methodology, Formal analysis, Conceptualization. **Jefferson Seide Molléri:** Writing – review & editing, Validation, Methodology, Formal analysis, Conceptualization.

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.

Appendix A

1. *Terms and definitions*

To provide a better understanding of the application context in our SLR to the readers, we described the several key terms used in this work. The terms are alphabetically ordered:

ADAS or Advanced Driver-Assistance System. These systems are focused on improving safety driving, offering features that can warn the driver in case of misdriving (lane departure), distracted attention, or assume vehicle control (steering, braking, accelerating).

Approach. Based on the context of this SLR, we refer to the methodologies, processes, frameworks, models, methods, and techniques focused on software data assessment phases.

Automotive SPICE (A-SPICE). It is a reference model for automotive process assessment [12].

Capability maturity model integration (CMMi). Is a bundle of practices to evaluate product development or services [12].

Characteristics impacts. The reported effects or influences on the software quality, such as safety, performance, user experience (UX), cost, efficiency, and cyber-security.

Critical embedded systems (CES). The systems, in case of failures, can hardly impact the safety, security, and performance causing potential damage to the environment or to human lives [15].

FuSa or Functional Safety. Is the "absence of unreasonable risks due to hazards caused by malfunctioning behaviour of Electric/Electronic systems" [10]. ISO26262 is a standard that offers a framework to provide guidelines for automotive safety-related Electric/Electronic systems.

Property. In this work, the extracted data that contributes to research questions answering and/or aggregates value for the results analysis step.

Quality assessment characteristics. The main assessed attributes in the software (e.g. code quality, safety level, data accuracy). They were extracted from selected studies.

Safety. Is a system attribute with the focus on software as the critical component [16].

Software data. In this work, "software data" means any software-related information that can be used for checking, measuring, comparing, or evaluating via static (e.g., documentation, software code) or dynamic (e.g., test benches, vehicle, simulation) tests. The term software applied in this work is based on the IEEE 1012–2016, which includes firmware and code. Software includes documentation.

Test maturity model integration (TMMi). Is a reference model for test process assessment and improvements [12].

User experience (UX). In the context of this SLR, is the impact created by a software quality improvement on the product in terms of facility, day-by-day use, and efficiency perceived by the final customer [3,17,34,41].

Validation. This term is based on IEEE 1012–2016, which defines "validation" as the evaluation process of the software (as part of a system or component) during or after the development process concerning the specified requirements. Validation processes include analysis, review, inspection, assessment, and testing.

2. *Categories description*

The definitions were self-stated by the authors of the primary studies. Our classification was based on findings, and we did not intend to re-classify them into other definition groups, rather the one stated in the primary studies. We acknowledge that overlaps or similarities exist among the different terms and now discuss them in our Section 3.4-Threats to Validity and Limitations.

Approaches

We considered the self-stated approaches integrally.

- Techniques
- Methods
- Frameworks
- Models
- Processes

- Case study
- Methodology
- Standard
- Systematic Mapping Study
- Topology

Application Focus / Target Groups

- Quality. When the studies present this general term, we understand that a bundle that was not described or split by the studies is behind. The content of this bundle is detailed in standards, such as ISO 25,012, which support the critical characteristics for software quality evaluation.
- Safety. The term is covered by ISO 26,262 or only self-stated by studies.
- Performance. The combination of characteristics is covered by standards (ISO25012) and others, such as comfort.
- Cybersecurity. The term is covered by ISO 21,434, or only self-stated by studies.
- Traceability. The term is covered by ISO 25,012, or only self-stated by studies.
- Reliability. Term self-stated by studies.

Quality Assessment Application

- Validation. In same studies, the Validation is cited as Verification and Validation (V&V).
- Concept. Term self-stated by studies.
- Life Cycle. This classification defines the software phases from concept to decommissioning.
- Development Phases. The automotive V-Model establishes that "development phases" are Requirements & System Specification, Architecture & Component Design, Coding or Implementation, and Tests (Component, Integration, System, and Acceptance).
- Field Applications. The end-user environment.

Product Type

- Embedded Software. The embedded software is an automotive software under evaluation and the focus of the studies' assessment.
- Autonomous Systems. The Advanced Driver Assistance Systems are in scope. Examples: Active Brake Assist and Lane Keep Assist.
- Electronic Control Systems. The combination of embedded software and hardware.
- Powertrain. This term includes the engine, gearbox, and axles.
- Commercial Vehicles. Focused on vehicle-centered approaches.
- Connectivity. Cloud-based systems and others that use over-the-air solutions.
- Electric Vehicles. All categories of vehicles equipped with electric propulsion.
- Automation Systems. The automated gearbox is an example.

Software Data sources

- Specification. When the assessment data is based on specifications.
- Static/Dynamic Tests. When the Static/Dynamic tests provide reference data for other approaches.
- Source Code.
- Data Validation Set. When the storage data from tests is applied.
- Modelling. The result of modelling tasks is in focus.
- Historic Fault Data. When the storage fault data is applied.
- Simulation. The result of modelling tasks is in focus.

Reported Assessment Impacts

- Quality. When the studies present this general term, we understand that a bundle that was not described or split by the studies is behind. The content of this bundle is detailed in standards, such as ISO 25,012, which support the critical characteristics for software quality evaluation.
- Safety. The term is covered by ISO 26,262 or only self-stated by studies.
- Performance. The combination of characteristics covered by standards (ISO25012) and others, such as comfort.
- User Experience (UX)
- Cost
- Efficiency. The term is covered by ISO 25,012 or is only self-stated by studies.
- Cybersecurity. The term is covered by ISO 21,434 or is only self-stated by studies.

3. *Instructions for quality assessment criteria*

We have created the instructions below to provide a better understanding of the quality assessment defined criteria.
Q1 - Is the study related to the proposed SLR objectives?
The target is to identify the presence of research questions topics in the studies.

- 1 point (Yes): The study contributes to all the research questions.
- 0,5 point (Partially): The study contributes at least one or more research question topic.
- 0 point ("No"): The study does not contribute to the research questions.

Q2 - Does the study bring relevant information for answering research questions?
The target is to identify the detailing level of research question topics in the studies.

- 1 point (Yes): The study contributes to all the research questions.
- 0,5 point (Partially): The study contributes at least one or more research question topic.
- 0 point ("No"): The study does not contribute to the research questions.

Q3 - Is the quality application field clearly described and applicable to vehicle embedded software? (RQ1)
The target is to identify the description level of the quality approach field.

- 1 point (Yes): The study offers a dedicated section for application description.
- 0,5 point (Partially): The application description is present and can be identified in the text.
- 0 point ("No"): The study does not contribute to the research questions.

Q4 - Are the methodology, processes, models or framework clearly presented? (RQ1.1)
The target is to identify the description level of the adopted quality approaches.

- 1 point (Yes): The study offers a dedicated section for description.
- 0,5 point (Partially): The description is present and can be identified in the text.
- 0 point ("No"): The study does not contribute to the research questions.

Q5 - Is there any technique comparison included? If yes, is that clearly presented and described?
The target is to identify the presence and description level of the technique's comparison.

- 1 point (Yes): The study offers a dedicated section for description.
- 0,5 point (Partially): The description is present and can be identified in the text.
- 0 point ("No"): The study does not contribute to the research questions.

Q6 - Does the study present any real use-case investigation, or is it applicable for? (RQ2)
The target is to identify the presence and description level of real use-cases.

- 1 point (Yes): The study offers a real use-case investigation and description.
- 0,5 point (Partially): The study presents example(s) of use-cases without detailing.
- 0 point ("No"): The study does not present real use-cases.

Q7 - Does the study bring consistent results regarding the presented technique(s) approach? (RQ3)
The target is to identify the presence and description level of the presented techniques.

- 1 point (Yes): The study offers results and description.
- 0,5 point (Partially): The study presents results without detailing.
- 0 point ("No"): The study does not present results.

Q8 - Does the study present success, unsuccess examples or issues during the software validation phase?
The target is to identify the presence and description level of critical results evaluation and challenges faced.

- 1 point (Yes): The study presents success, unsuccess, and challenges.
- 0,5 point (Partially): The study presents at least one of them.
- 0 point ("No"): The study does not present any results.

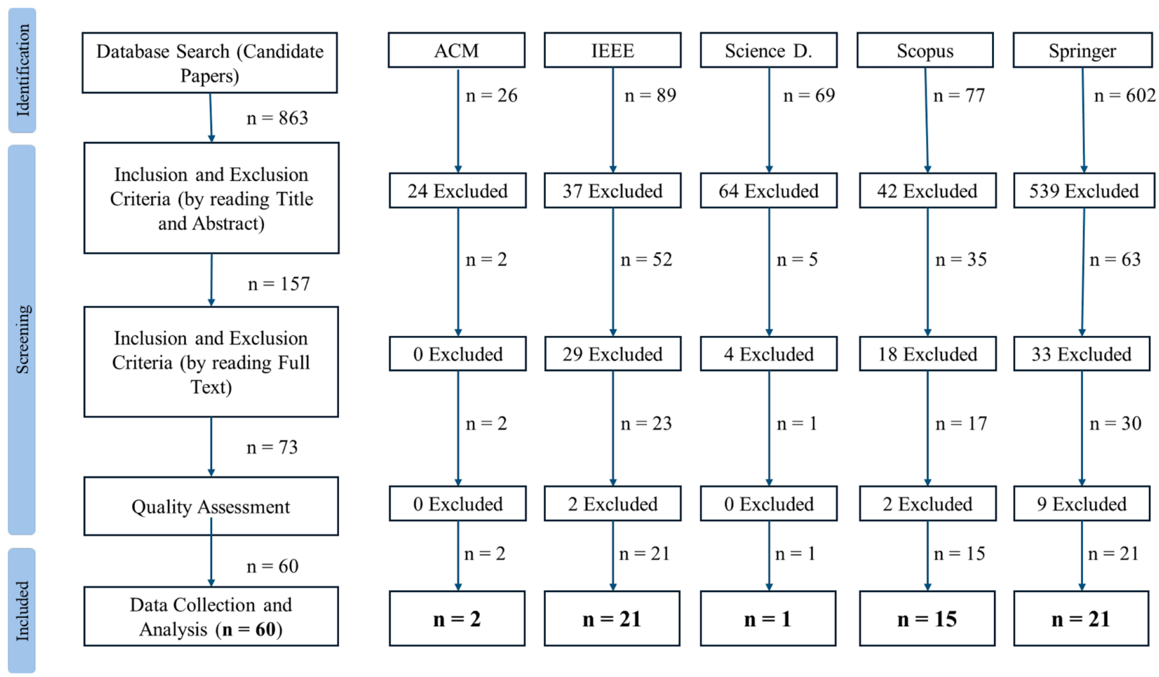
Q9 - Is the related study applicable to the commercial vehicles segment?
The target is to identify the applicability level for the commercial vehicles segment.

- 1 point (Yes): The study states the applicability for commercial vehicles.
- 0,5 point (Partially): The study presents general applicability coverage.
- 0 point ("No"): The study does not specify the applicability coverage.

Q10 - Does the study bring any coverage of quality issues related to software validation lacks?
The target is to identify the presence of examples of software validation lacks and their consequences.

- 1 point (Yes): The study presents a root cause relationship.
- 0,5 point (Partially): The study presents a general lack and/or consequences.
- 0 point ("No"): The study does not mention them.

4-. Protocol application: candidate studies identification, screening process, and inclusion/exclusion results by database



Data availability

No data was used for the research described in the article.

References

[1] IEEE Standards Association, IEEE 1012-2016: IEEE standard for system, Software, and hardware verification and validation (2017).

[2] P.L. Goddard, Validating the safety of embedded real-time control systems using FMEA, in: Annual Reliability and Maintainability Symposium 1993 Proceedings. IEEE, 1993, https://doi.org/10.1109/RAMS.1993.296851.

[3] J. Zhang, Y. Lei, X. Hua, et al., Proposed shift quality metrics and experimentation on AMT shift quality evaluation, in: Third International Conference on Natural Computation (ICNC 2007) 2, 2007, https://doi.org/10.1109/ICNC.2007.582. IEEE.

[4] L. Mäurer, T. Hebecker, T. Stolte, et al., On bringing object-oriented software metrics into the model-based world–Verifying ISO 26262 compliance in Simulink. System analysis and modeling: models and reusability: 8th international conference, SAM 2014, Valencia, Spain, September 29-30, 2014. Proceedings 8, Springer International Publishing, 2014, https://doi.org/10.1007/978-3-319-11743-0_15.

[5] M. Nyberg, D. Gurov, C. Lidström, et al., Formal verification in automotive industry: enablers and obstacles." leveraging applications of Formal methods, verification and validation. Industrial practice: 8th international symposium, ISoLA 2018, Limassol, Cyprus, November 5-9, 2018, Proceedings, Part IV 8, Springer International Publishing, 2018, https://doi.org/10.1007/978-3-030-03427-6_14.

[6] I. Stürmer, E. Salecker, H. Pohlheim, Reviewing software models in compliance with ISO 26262, in: Computer safety, reliability, and security: 31st international conference, SAFECOMP 2012, Magdeburg, Germany, September 25-28, 2012. Proceedings 31, Springer Berlin Heidelberg, 2012, https://doi.org/10.1007/978-3-642-33678-2_22.

[7] F. Wotawa, B. Peischl, F. Klück, et al., Quality assurance methodologies for automated driving, Elektrotech. Inf. 135 (4–5) (2018) 322–327, https://doi.org/10.1007/s00502-018-0630-7.

[8] ISO/IEC, ISO/IEC 25000: 2014 systems and software engineering-systems and software Quality Requirements and Evaluation (SQuaRE)-Guide to SQuaRE, Inf. technol. stand. (2014).

[9] International Organization for Standardization, ISO/SAE 21434: 2021: Road Vehicles: Cybersecurity Engineering, ISO, (2021).

[10] International Organization for Standardization (ISO). ISO 26262-1: 2018–road vehicles—functional safety. Geneva, Switzerland. 2018.

[11] P. Kumar, L.K. Singh, C. Kumar, et al., A Bayesian belief network model for early prediction of reliability for computer-based safety-critical systems, in: 2021 2nd International Conference on Range Technology (ICORT). IEEE, 2021, https://doi.org/10.1109/ICORT52730.2021.9581624.

[12] E. Touw, Multi-faceted reliability assessment techniques: an industrial case study, in: 2017 IEEE International Conference on Software Architecture Workshops (ICSAW). IEEE, 2017, https://doi.org/10.1109/ICSAW.2017.66.

[13] T. Genevois, J.B. Horel, A. Renzaglia, et al., Augmented reality on lidar data: going beyond vehicle-in-the-loop for automotive software validation, in: 2022 IEEE Intelligent Vehicles Symposium (IV). IEEE, 2022, https://doi.org/10.1109/IV51971.2022.9827351.

[14] H. Kuwajima, H. Yasuoka, T. Nakae, Engineering problems in machine learning systems, Mach. Learn 109 (5) (2020) 1103–1126, https://doi.org/10.1007/s10994-020-05872-w.

[15] D. Feitosa, A. Ampatzoglou, P. Avgeriou, et al., Design approaches for critical embedded systems: a systematic mapping study, in: Evaluation of Novel Approaches to Software Engineering: 12th International Conference, ENASE 2017, Porto, Portugal, April 28–29, 2017.

[16] P.V. Bhansali, Universal software safety standard, ACM SIGSOFT Softw. Eng. Notes 30 (5) (2005) 1–4, https://doi.org/10.1145/1095430.1095440.

[17] R. Rana, M. Staron, J. Hansson, et al., Defect prediction over software life cycle in automotive domain state of the art and road map for future, in: 2014 9th International Conference on Software Engineering and Applications (ICSOFT-EA). IEEE, 2014.

[18] L.S. Myllyaho, et al., Systematic literature review of validation methods for AI systems, J. Syst. Softw. 181 (Nov. 2021) 111050, https://doi.org/10.1016/j.jss.2021.111050.

[19] B. Gezici, A.K. Tarhan, Systematic literature review on software quality for AI-based software, Empir. Softw. Eng. 27 (3) (Mar. 2022), https://doi.org/10.1007/s10664-021-10105-2. Art. 66.

[20] M.A. Ali, et al., A systematic mapping of quality models for AI systems, software and components, Appl. Sci. 12 (17) (Aug. 2022), https://doi.org/10.3390/app12178700. Art. 8700.

[21] M. Krichen, Formal methods and validation techniques for ensuring automotive systems security, Information 14 (12) (Dec. 2023) 666, https://doi.org/10.3390/info14120666.

[22] B. Kitchenham, and S. Charters. "Guidelines for performing systematic literature reviews in software engineering." 2007, 1051.

[23] J. Biolchini, P.G. Mian, A.C.C. Natali, et al., Systematic review in software engineering, Syst. Eng. Comput. Sci. Dep. COPPE/UFRJ Tech. Rep. ES 679.05 (2005) 45.

[24] V. Parsifal, 2021. Available online: https://parsif.al (accessed on 24.03.2024).

[25] D. Stefanovic, S. Havzi, D. Nikolic, et al., Analysis of the tools to support systematic literature review in software engineering. IOP conference series: materials science and engineering. Vol. 1163. No. 1, IOP Publishing, 2021.

[26] C. Wohlin, P. Runeson, M. Höst, et al., Experimentation in software engineering, Springer Science & Business Media, 2012.

[27] Kitchenham, B.A., Budgen, D., & Brereton, P. "Evidence-based software engineering and systematic reviews." CRC press., 2015.
[28] J.R. Landis, G.G. Koch, An application of hierarchical kappa-type statistics in the assessment of majority agreement among multiple observers, Biometrics (1977) 363–374.
[29] I. Sommerville, Software engineering, 9"'ed, Engl.: Educ. Ltd. (2010).
[30] J. Saldaña, The coding manual for qualitative researchers, Sage (2015).
[31] D.M. Bailey, J.M. Jackson, Qualitative data analysis: challenges and dilemmas related to theory and method, Am. J. Occup. Ther. 57 (1) (2003) 57–65. Publisher: American Occupational Therapy Association.
[32] M. Takrouni, M. Gdhaifi, A. Hasnaoui, et al., Design and implementation of a simulink DDS blockset and its integration to an active frame steering blockset conformed to SAE ElectricVehicle, in: 2017 IEEE/ACS 14th International Conference on Computer Systems and Applications (AICCSA). IEEE, 2017, https://doi.org/10.1109/AICCSA.2017.52.
[33] A. Mateen, Q. Zhu, S. Afsar, Comparitive analysis of manual vs automotive testing for software quality, in: Proceedings of the 7th International Conference on Software Engineering and New Technologies, 2018, https://doi.org/10.1145/3330089.3330121.
[34] X. Lu, H. Chen, B. Gao, et al., Data-driven predictive gearshift control for dual-clutch transmissions and FPGA implementation, IEEE Trans. Ind. Electron. 62.1 (2014) 599–610, https://doi.org/10.1109/TIE.2014.2312312.
[35] K. Ranasinghe, R. Sabatine, A. Gardi, et al., Advances in integrated system health management for mission-essential and safety-critical aerospace applications, Prog. Aerosp. Sci. 128 (2022), https://doi.org/10.1016/j.paerosci.2021.100758.
[36] M.K. Thota, F.H. Shajin, P. Rajesh, Survey on software defect prediction techniques, Int. J. Appl. Sci. Eng. 17 (4) (2020) 331–344, https://doi.org/10.6703/IJASE.202012_17(4).331.
[37] H. Ebadi, M.H. Moghadam, M. Borg, et al., Efficient and effective generation of test cases for pedestrian detection-search-based software testing of Baidu Apollo in SVL, in: 2021 IEEE International Conference on Artificial Intelligence Testing (AITest). IEEE, 2021, https://doi.org/10.1109/AITEST52744.2021.00030.
[38] H. Schweppe, A. Zimmermann, D. Grilly, Flexible in-vehicle stream processing with distributed automotive control units for engineering and diagnosis, in: 2008 International Symposium on Industrial Embedded Systems. IEEE, 2008, https://doi.org/10.1109/SIES.2008.4577683.
[39] O. Sadio, I. Ngom, C. Lishou, Design and prototyping of a software defined vehicular networking, IEEE Trans. Veh. Technol. 69 (1) (2019) 842–850, https://doi.org/10.1109/TVT.2019.2950426.
[40] M. Kläs, T. Bauer, A. Dereani, et al., A large-scale technology evaluation study: effects of model-based analysis and testing, in: 2015 IEEE/ACM 37th IEEE International Conference on Software Engineering. Vol. 2. IEEE, 2015, https://doi.org/10.1109/ICSE.2015.141.
[41] P.J. Kollmeyer, M. Naguib, F. Khanum, et al., A blind modeling tool for standardized evaluation of battery state of charge estimation algorithms, in: 2022 IEEE Transportation Electrification Conference & Expo (ITEC). IEEE, 2022, https://doi.org/10.1109/ITEC53557.2022.9813996.
[42] N. Kovačić, V. Ilić, A. Korać, et al., Validation of process execution on automotive embedded devices, in: 2017 13th International Conference on Advanced Technologies, Systems and Services in Telecommunications (TELSIKS). IEEE, 2017, https://doi.org/10.1109/TELSKS.2017.8246318.
[43] R. Jiang, Required characteristics for software reliability growth models. 2009 WRI world congress on software engineering. Vol. 4, IEEE, 2009, https://doi.org/10.1109/WCSE.2009.157.
[44] P. Herber, S. Glesner, Verification of embedded real-time systems, in: Formal Modeling and Verification of Cyber-Physical Systems: 1st International Summer School on Methods and Tools for the Design of Digital Systems, Bremen, Germany 2015, September 2015, pp. 1–25, https://doi.org/10.1007/978-3-658-09994-7_1.
[45] E. Bringmann, A. Krämer, Model-based testing of automotive systems, in: 2008 1st international conference on software testing, verification, and validation. IEEE, 2008, https://doi.org/10.1109/ICST.2008.45.
[46] C. Kugler, S. Kowalewski, J. Richenhagen, et al., Metrics-based strategies for quality assurance of automotive embedded software. 17. Internationales stuttgarter symposium: Automobil-und Motorentechnik, Springer, Fachmedien Wiesbaden, 2017, https://doi.org/10.1007/978-3-658-16988-6_56.
[47] J. Jürjens, D. Reiß, D. Trachtenherz, Model-based quality assurance of automotive software. International conference on model driven engineering languages and systems, Springer Berlin Heidelberg, Berlin, Heidelberg, 2008, https://doi.org/10.1007/978-3-540-87875-9_59.
[48] M. Ellims, R. Evans, K.M. Hobley, et al., When is software ready for production? Parallels with automotive QS9000 methods, in: Aspects of safety management: proceedings of the ninth safety-critical systems symposium, Bristol, UK 2001, Springer London, London, 2000, https://doi.org/10.1007/978-1-4471-0713-2_9.
[49] X. Li, W.E. Wong, R. Gao, et al., Genetic algorithm-based test generation for software product line with the integration of fault localization techniques, Empir. Softw. Eng. 23 (2018) 1–51, https://doi.org/10.1007/s10664-016-9494-9.
[50] D.K. Oka, T. Makila, R. Kuipers, Integrating application security testing tools into ALM tools in the automotive industry, in: 2019 IEEE 19th International Conference on Software Quality, Reliability and Security Companion (QRS-C). IEEE, 2019, https://doi.org/10.1109/QRS-C.2019.00021.
[51] O. Kovalenko, E. Serral, M. Sabou, et al., Automating cross-disciplinary defect detection in multi-disciplinary engineering environments, Knowledge engineering and knowledge management: 19th international conference, EKAW 2014, Linköping, Sweden, November 24-28, 2014. Proceedings 19 (2014), https://doi.org/10.1007/978-3-319-13704-9_19.
[52] J.R.V. Rivero, T. Gerbich, B. Buschardt, et al., The effect of spray water on an automotive LIDAR sensor: a real-time simulation study, IEEE Trans. Intell. Veh. 7.1 (2021) 57–72, https://doi.org/10.1109/TIV.2021.3067892.
[53] L.J. Moukahal, M. Zulkernine, M. Soukup, Vulnerability-oriented fuzz testing for connected autonomous vehicle systems, IEEE Trans. Reliab. 70 (4) (2021) 1422–1437, https://doi.org/10.1109/TR.2021.3112538.
[54] E.Y. Kang, D. Mu, L. Huang, et al., Verification and validation of a cyber-physical system in the automotive domain, in: 2017 IEEE International Conference on Software Quality, Reliability and Security Companion (QRS-C). IEEE, 2017, https://doi.org/10.1109/QRS-C.2017.62.
[55] M. Mauritz, F. Howar, A. Rausch, Assuring the safety of advanced driver assistance systems through a combination of simulation and runtime monitoring. Leveraging applications of formal methods, verification and validation: discussion, dissemination, applications: 7th international symposium, ISoLA 2016, Imperial, Corfu, Greece, October 10-14, 2016, Proceedings, Part II 7, Springer International Publishing, 2016, https://doi.org/10.1007/978-3-319-47169-3_52.
[56] P. Skruch, M. Długosz, P. Markiewicz, A formal approach for the verification of control systems in autonomous driving applications, in: Trends in Advanced Intelligent Control, optimization and automation: proceedings of KKA 2017—the 19th Polish control conference, Kraków, Poland, June 18–21, 2017, Springer International Publishing, 2017, https://doi.org/10.1007/978-3-319-60699-6_18.
[57] K. Liu, W. Kong, G. Hou, et al., A survey of formal techniques for hardware/software co-verification, in: 2018 7th International Congress on Advanced Applied Informatics (IIAI-AAI). IEEE, 2018, https://doi.org/10.1109/IIAI-AAI.2018.00033.
[58] N. Mellegård, M. Staron, F. Törner, A light-weight defect classification scheme for embedded automotive software and its initial evaluation, in: 2012 IEEE 23rd International Symposium on Software Reliability Engineering. IEEE, 2012, https://doi.org/10.1109/ISSRE.2012.15.
[59] D.K. Kim, J.H. Kim, M.C. Lee, Improvement of HILS using advanced exploratory and optimization techniques for system qualification test, Int. J. Automot. Technol. 24 (3) (2023) 901–911, https://doi.org/10.1007/s12239-023-0074-x.