Design Approaches for Critical Embedded Systems: A Systematic Mapping Study

Daniel Feitosa[1], Apostolos Ampatzoglou[1], Paris Avgeriou[1(✉)], Frank J. Affonso[2], Hugo Andrade[3], Katia R. Felizardo[4], and Elisa Y. Nakagawa[5]

[1] Department of Mathematics and Computer Science, University of Groningen, Groningen, The Netherlands
{d.feitosa,a.ampatzoglou}@rug.nl, paris@cs.rug.nl
[2] Department of Statistics, Applied Mathematics and Computation, São Paulo State University (UNESP), Rio Claro, Brazil
frank@rc.unesp.br
[3] Department of Computer Science and Engineering, Chalmers University of Technology, Göteborg, Sweden
sica@chalmers.se
[4] Department of Computing, Federal Technological University of Paraná, Cornélio Procópio, Brazil
katiascannavino@utfpr.edu.br
[5] Department of Computer Systems, University of São Paulo, São Carlos, Brazil
elisa@icmc.usp.br

Abstract. Critical Embedded Systems (CES) are systems in which failures are potentially catastrophic and, therefore, hard constraints are imposed on them. In the last years the amount of software accommodated within CES has considerably changed. For example, in smart cars the amount of software has grown about 100 times compared to previous years. This change means that software design for these systems is also bounded to hard constraints (e.g., high security and performance). Along the evolution of CES, the approaches for designing them are also changing rapidly, so as to fit the specialized needs of CES. Thus, a broad understanding of such approaches is missing. Therefore, this study aims to establish a fair overview on CESs design approaches. For that, we conducted a Systematic Mapping Study (SMS), in which we collected 1,673 papers from five digital libraries, filtered 269 primary studies, and analyzed five facets: design approaches, applications domains, critical quality attributes, tools, and type of evidence. Our findings show that the body of knowledge is vast and overlaps with other types of systems (e.g., real-time or cyber-physical systems). In addition, we have observed that some critical quality attributes are common among various application domains, as well as approaches and tools are oftentimes generic to CES.

Keywords: Systematic mapping study · Critical embedded system Design

E. Damiani et al. (Eds.): ENASE 2017, CCIS 866, pp. 243–274, 2018.
https://doi.org/10.1007/978-3-319-94135-6_12

1 Introduction

Critical Embedded Systems (CESs) are among the most significant types of software-intensive systems, since they are extremely pervasive in modern society, being used from cars to power plants [1]. CESs are embedded systems in which runtime errors can potentially be catastrophic [2], causing serious damage to the environment or to human lives, or non-recoverable material and financial losses [3, 4]. Due to the criticality of such systems, the satisfaction of multiple quality constraints must be guaranteed. This is far from trivial, as it entails complex trade-offs, which to a large extent concern safeguarding the levels of critical against other non-critical qualities [5, 6]. As critical quality attributes (CQAs), we characterize qualities that, when not satisfied, may lead to catastrophic failures, as the aforementioned ones; typical examples are performance, security and reliability.

Engineering CES is particularly challenging, since it needs to guarantee the satisfaction of various critical qualities. One of the key solutions to alleviate this challenge is to design a sound architecture and validate it against the critical quality attributes. To this end, multiple approaches have been proposed, solving a variety of specific design problems. However, the plethora and diversity of available solutions has led to a difficulty on understanding, applying or even extending and combining such approaches. Thus, in order to support researchers and practitioners on CES design, it is important to have a comprehensive understanding of this field. To contribute towards a better understanding of design approaches for CES, we have conducted a systematic mapping study; this is a commonly used approach for assessing and describing the state of the art in a specific domain or problem (see Sect. 3 for more details). The contributions of this study are the following: (a) a classification of the existing approaches to design CES; (b) a list of tools for supporting existing approaches; (c) a list of domains for which approaches have been developed and used; (d) a list of the most commonly identified CQAs in the CES design; and (e) a classification of these approaches, based on the level of their empirical evidence.

2 Related Work

This section describes related Systematic Literature Reviews (SLRs) or Systematic Mapping Studies (SMSs), also known as secondary studies. To the best of our knowledge, there are no studies that focus on exactly the same topic as ours, i.e., designing of CESs. Thus, we searched for related work such as SMSs and SLRs that cover the entire software development process of CES, or a specific phase.

2.1 Development Processes

We identified two studies that discuss software development processes and are related to CESs [7, 8]. Although such processes do not focus or limit themselves to the design phase, they do have impact on the design phase. Cawley et al. [7] investigated Lean/Agile development processes on safety-critical systems, focusing on medical devices. For this purpose, an SLR based on the guidelines of Kitchenham and Charters

[9] was performed. The results of the SLR suggest that Lean/Agile methodologies are appropriate for the development of safety-critical systems, as they support several practices for regulated safety-critical domains (e.g., traceability and testing). However, the results also suggest a lack of adoption of Lean/Agile methods in these domains. This is not surprising as regulated environments typically involve activities that are not commonly used in these processes. Eklund and Bosch [8] investigated a holistic model for aligning software development processes with the architecture of embedded software. As part of this study, an SMS on development approaches for embedded systems was performed (based on the guidelines of Kitchenham and Charters [9]). The results of the study suggest that there is no single most common approach (or set of approaches) but, approaches are tailored for specific domains or products and may have different characteristics (e.g., incorporating agile practices). Despite the high customization of processes, the authors have been able to identify some similarities, e.g. activities are often executed sequentially and follow a V-model - [10] or stage-gate-like [11] process. In addition, the architectures created from these processes are often focused on supporting specific quality attributes, which are typically domain-specific (e.g., dependability for the space domain). Based on the identified approaches, the authors derived five archetypical developments processes, with their respective characteristics, aiming to support selection or migration between concrete archetypal development approaches.

2.2 Verification and Validation

Not all activities in the verification and validation of critical embedded software (V&V) are related to its design. However, a significant part concerns the verification and validation of design and are, therefore, relevant to the design phase. We identified two secondary studies that discuss aspects of V&V and are related to CES [12, 13]. Barbosa et al. [12] investigated software testing of CESs, checking the compliance level with the standard DO-178B, for the aviation industry. The aim was to identify primary studies that could be used to create a methodology for testing of CES. For this purpose, a SLR, based on Dybå and Dingsøyr [14], was performed to identify studies that implemented or applied V&V techniques in the context of CES. The results suggest that four techniques (functional, structural, mutation and model-based testing) are widely applied for testing of CES, from which the most recurrent technique is functional testing. In addition, all testing requirements of DO-178B have been investigated, with "structural coverage analysis" (e.g., dead code and deactivated code) being the most addressed requirement, likely due to its inherent complexity. Elberzhager et al. [13] investigated quality assurance techniques (i.e., analysis or test approaches) applied to Matlab Simulink models. These models are used in embedded software design, especially in critical domains. The aim was to develop an approach able to integrate different quality assurance techniques. For this purpose, an SMS was performed based on the guidelines of Petersen et al. [15], which presented different analysis and test techniques as well as some combined approaches. The results of the study suggest that formal methods, properties checking (e.g., rule-based analysis) and automatic test generation are the most common approaches for performing quality assurance for

embedded systems. The results also suggest a lack of research on combining analysis techniques with testing techniques for such models.

2.3 Software Architecture

The activity of architecture design for embedded systems was investigated by Antonio et al. [16], which aimed at establishing the state of the art on the topic by analyzing proposed architectures, available on the literature. For that, a SMS based on the guidelines of Petersen et al. [15] was performed. To understand the activity, various characteristics were collected from the architectures, and used for classifying them. Firstly, the architectures were grouped according to the type of modeling technique used to design them, namely formal, semi-formal and informal. Next, further classes were identified based on recurrent characteristics, e.g., level of abstraction and whether it is domain-specific. The results of the SMS suggest that the Architecture Analysis and Design Language (AADL) is the most used formal modeling approach, whereas UML stands out among the semi-formal and informal approaches. In addition, the most recurrent characteristic of these architectures is that they are designed to specific application domains.

Similar to the previous study, Guessi et al. [17] investigate the modeling of software architectures for embedded systems. However, this study focuses on architecture description languages (ADLs), as well as the concerns (e.g., quality attributes) being addressed and information (e.g., components, events) being represented in the designed architectures. The investigation was performed via a SLR based on the guidelines of Kitchenham and Charters [9]. The results suggest that UML is the most common language, while safety is the concern that is more often addressed. Despite the variety of approaches that currently exist, the results also suggest that more attention should be placed on the description of embedded system architectures. Among the reasons, Guessi et al. argue that there is a lack of consensus about the most adequate approach (es) for describing architecture, as well as whether existing approaches are sufficient for representing the variety of embedded systems.

Nakagawa et al. [18] present the state of the art on architecting approaches for systems of systems[1] (SoS), of which CES are among the most common examples. For that, an SLR based on the guidelines of Kitchenham and Charters [9] was performed, investigating the creation, representation, evaluation and evolution of these architectures. The results suggest the existence of several approaches, although most of them lack maturity and are neither adequately adapted nor widely adopted. In addition, several application domains (e.g., avionics and space) and quality attributes (e.g., security, reliability and performance) are common between SoS and CES.

[1] SoS are integrated solutions comprising operationally independent (non-trivial) systems, which are orchestrated in order to provide a more complex functionality.

2.4 Comparative Analysis

After presenting related work, it is important to highlight the differences between these studies and our work. To illustrate these differences, we compare them w.r.t. six characteristics (Table 1): review type; number of included primary studies; whether the study focuses on CES or is only indirectly related (i.e., with partial applicability to CES); whether it considered quality attributes (QA) in the investigation; whether it considered application domains in the investigation; and the main topic of the investigation. The review type is an indication of whether the study presents an overview or a detailed analysis over the main topic (SMS) or it examines more in-depth research questions (SLR). As presented in Table 1, three other SMSs were performed, although they were focused in different, yet related, topics. However, these three studies were not focused on CESs, which reinforces the purpose of our study, as it complements existing knowledge. Other important aspects of our study include the larger body of knowledge that has been investigated (due to the broader topic of research), as well as the consideration of quality attributes and application domains in the investigation. CESs are used in a variety of application domains and multiple factors affect the decision-making to select or reuse a design approach. Quality constraints are among the most relevant factors, as also suggested by related work [8, 17, 18]. Application domains may also play an important role, as each domain groups a set of common requirements, that are in turn related to specific quality attributes [8].

Table 1. Comparison between related work and our study.

Study	Review type	Number of studies	Focus on CES?	Investigated QAs?	Focus on domains	Main topic
[7]	SLR	19	Yes	No	No	Development process
[8]	SMS	23	No	Yes	Yes	Development process
[12]	SLR	97	Yes	No	No	Verification and validation
[13]	SMS	44	No	No	No	Verification and validation
[16]	SMS	104	No	No	No	Software architecture
[17]	SLR	24	No	Yes	No	Software architecture
[18]	SLR	60	No	Yes	Yes	Software architecture
Ours	SMS	258	Yes	Yes	Yes	Design

3 Review Methodology

Systematic Mapping Studies (SMSs) and Systematic Literature Reviews (SLRs) have been broadly adopted as systematic research methods to aggregate knowledge. As this study aims to outline the state-of-the-art on design approaches for CES in a broad sense, we decided to perform an SMS [15]. The rest of this section describes the protocol of our SMS, based on the guidelines of Petersen et al. [15].

3.1 Research Scope

The goal of this SMS is described using the Goal-Question-Metrics (GQM) approach [19], as follows: "**analyze** existing software engineering literature **for the purpose of** characterizing the state of the art **with respect to** approaches (e.g., processes, methods and tools) for designing critical embedded systems **from the point of view of** researchers and practitioners **in the context of** software-intensive systems engineering". Based on the goal we defined the following research questions (RQs):

RQ_1 - What are the proposed approaches for designing CES?
$RQ_{1.1}$ - Is the nature of these approaches industrial, academic or mixed?
$RQ_{1.2}$ - What is the purpose of the approach?
RQ_2 - What are the application domains where these approaches are applied?
RQ_3 - What are the most common critical quality attributes identified in CES design?
RQ_4 - What tools have been used to support CES design?
RQ_5 - What are the types of evidence provided in CES design research?

To achieve the aforementioned goal, we must analyze and present the existing body of knowledge from different perspectives. The most important outcome of this SMS is the identification and characterization of the approaches that were created and/or used to design CES (RQ_1). As a first step in characterizing the approaches, we consider their nature and purpose. Next, we look at the application domain (RQ_2) which influences CES design as it often imposes a number of constraints. For example, several application domains are bounded by international standards (e.g., DO-178B for aviation). In addition, these constraints commonly aim at defining critical quality values (e.g., safety); thus, design approaches are often targeting those values (e.g., fault tree analysis). Therefore, investigating the addressed quality attributes (RQ_3) is of paramount importance. Furthermore, multiple tools have been proposed or tailored to support the design of CES. As the number of CES grows, it is interesting to investigate how this reflects on the tooling (RQ_4), e.g., leading to news tools and adaptation of existing ones. Finally, it is important to not only classify the approaches, but also assess their maturity level to inform researchers and practitioners. For that, we analyze the types of evidence provided within the literature (RQ_5).

3.2 Search Strategy

Considering the research questions, we defined the search strategy, which comprises the selection of sources for collecting primary studies, as well as the definition of the scope for the collection.

Sources Selection. We decided to perform an automated search, as a manual search would be very time-consuming, thus not allowing us to search as many venues. In addition, by considering digital libraries (through an automated search) we might also include venues that otherwise we would not be aware of. The following criteria were adopted to select search sources (i.e., digital libraries): content update (publications are regularly updated); availability (full text of the papers is available); quality of results (accuracy of the results returned by the search); and versatility export (since a lot of information is returned through the search, a mechanism to export the results is required). These criteria are also discussed by Dieste et al. [20]. The selected sources for our SMS are: ACM Digital Library, IEEE Xplore, Science Direct, Springer Link and Scopus. According to Dybå et al. [21], the first four digital libraries are sufficient to conduct SMSs in the context of software engineering. Furthermore, Scopus was added, since it is considered to be the largest database of abstracts and citations [9].

Search Scope. As CESs have been the subject of research for a long time, we decided to not limit the start of the search period based on date of publication. However, we limit the end date of the search period in order to measure influence of the primary studies (see Sect. 3.5), considering primary studies published up to two years before the date of collection. We performed the data collection on March of 2015 and, thus, collected primary studies published up to March of 2013. Moreover, only primary studies written in English will be processed in this SMS. Due to automated search, we also defined a search string for filtering the studies to those that can be potentially included in the SMS. As we are interested in approaches for CES design, we selected two main keywords, "Critical Embedded System" and "Approach", with the respective related terms. The keywords were chosen to be simple enough to yield a large number of results and, at the same time, rigorous to cover only the desired research topic. The final search string is: *("Critical Embedded System" OR "Critical Embedded Systems" OR "Critical Embedded Software") AND ("Approach" OR "Approaches" OR "Method" OR "Methods" OR "Framework" OR "Frameworks" OR "Technique" OR "Techniques" OR "Process" OR "Processes" OR "Tool" OR "Tooling" OR "Guideline" OR "Guidelines").*

We clarify that we do not include terms such as "real-time", "hard real-time" or "cyber-physical systems", as they describe a broader range of systems, which extrapolates the scope of this SMS, and would make the paper selection process impractical. To validate the search string and, consequently, the papers collected by the automated search, we performed a manual search in a small number of venues, similarly to determining a *quasi-gold* standard as proposed by Zhang and Babar [22]. We selected the venues for the manual search based on their likelihood to publish studies on CES design: Real-time Systems journal, Digital Avionics Systems Conference (DASC), and International Conference on Computer Safety, Reliability, and Security (SAFECOMP). To filter the primary studies for the *quasi-gold standard*, we considered

the metadata (i.e., title, keywords and abstract) and full text (when necessary), resulting in the collection of 23 primary studies. Based on the *quasi-gold standard*, we adapted the search string to ensure that all 23 primary studies were included.

3.3 Study Selection

Based on the previously mentioned search strategy, we defined the procedure for filtering the results of the automated search, selecting the primary studies to be analyzed in the SMS. The study selection comprises the definition of the criteria for filtering the papers, both inclusion and exclusion criteria, as well as steps for applying them. We include a primary study if it: (a) proposes an approach to design CESs; (b) reports on the use of an approach to design CESs; (c) evaluates an approach to design CESs; or (d) discusses approach(es) to design CESs. A primary study is excluded if it is an editorial, position paper, keynote, opinion paper, tutorial, poster or panel. To promote a common understanding of the selection criteria among the three involved researchers, we performed a pilot selection on a small subset (50) of the papers collected from the sources. In this pilot, during a first review round, all researchers analyzed title, keywords and abstract of all papers and Cohen's Kappa was calculated between every pair of researchers (see Fig. 1). We clarify that no previous discussion was performed in order to evaluate the inclusion and exclusion criteria. Next, all researchers and authors discussed the criteria and their interpretation. Main points of this discussion included the boundaries of the design phase, hardware design and the inclusion of papers that do not propose approaches (e.g., use or discussion). Finally, in a second review round, the papers are analyzed again, but this time also considering introduction and conclusion sections (if necessary), and a new calculation of Cohen's Kappa was performed (see Fig. 1).

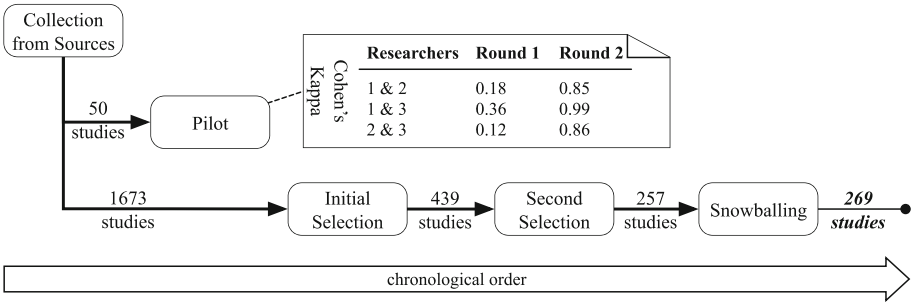

Fig. 1. Study selection.

To select the primary studies, we defined a three-step procedure. In every step, the papers were divided into three sets and three researchers were responsible for reviewing the papers of two sets. By doing this, we guarantee that every paper was reviewed by two different people while avoiding all three having to read all papers. When an inclusion/exclusion decision was conflicting or dubious (e.g., one or both reviewers

were not confident), the case was discussed among all authors. The selection steps were the following: (1) *Initial selection:* the search string was customized and applied to each publication source listed in Sect. 3.2. The string terms were searched in the title, abstract and keywords of all primary studies available in each database and search engine. As a result, a set of primary studies possibly related to the research topic was obtained. Based on this set, the title and the abstract of each primary study were read and evaluated based on the inclusion and exclusion criteria. The introduction and the conclusion may also be considered when necessary; (2) *Second selection:* each of the previously selected primary studies were read in full-text and analyzed according to inclusion and exclusion criteria. This step also included the data extraction, which is discussed in Sect. 3.5; and (3) *Snowballing:* the references of the studies selected in step 2 were used to identify extra literature, for which steps 1 and 2 are repeated.

3.4 Keywording

During the first two steps of the selection procedure (see Sect. 3.3), a set of keywords was collected from each primary study. As proposed by Petersen et al. [15], the keywording process occurs in two steps:

(1) *Identification of Context*: While reading the paper, the reviewer identifies any keywords and concepts that are relevant to describe that particular study. For example, words that describe the purpose of the approach, code of standards and names of quality attributes or tools were collected. During this step, reviewers share topics of keywords (e.g., code of standards) to maintain consistency and optimize the collection. Differently from Petersen et al. [15], we extended the searching of keywords to the whole paper, as some relevant keywords have been identified within the full text at early stages of the study.
(2) *Summarization*: The keywords are combined in order to create abstractions that support understanding the body of knowledge under investigation. Examples of such abstractions are the topics mentioned in the previous step (e.g., standards). The abstractions also support identifying categories and create a classification scheme for the primary studies.

We applied keywording not only to classify the primary studies but also to identify relevant concepts for all research questions, e.g., purpose of tools, application domains standards and safety integrity levels (SILs).

3.5 Data Extraction and Mapping

During the *second selection* procedure (see Sect. 3.3), a set of variables were collected from each primary study to answer the research questions. Similar to selection procedure, the data collection of every paper involved two researchers and conflicts were discussed among all authors. The extracted variables are described in Table 2.

The mapping between variables and research questions is provided in Table 3, accompanied by the analysis method used on the data. The type of evidence (V14) evaluates the level of evidence of the proposed approach. For that, we adopted the classification proposed by Alves et al. [23] in order to make the assessment more

Table 2. Extracted variables.

Variable	Description	Variable	Description
V1	Author(s)	V8	Type of paper (conference/journal/book)
V2	Year	V9	SMS keywords
V3	Title	V10	Approaches to design CES
V4	Source	V11	Application domain(s)
V5	Venue	V12	Critical quality attributes
V6	Author(s) keywords	V13	Nature of the approaches (industrial/academic/mixed)
V7	Number of citations per year	V14	Tools to support the approaches
		V15	Type of evidence used to develop the approach

Table 3. Mapping of variables to RQs.

Research question	Variables used	Analysis method
RQ_1 (Approaches)	V1–V3, V6, V7, V9–V10	Descriptive Statistics (sum, average, frequency analyses, etc.) Classification based on keywording Heatmap based on classification and year Crosstabs on classification vs. nature
RQ_2 (Application domains)	V1–V3, V10, V11	Descriptive Statistics (sum, average, frequency analyses, etc.) Heatmap based on application domain and year Crosstabs on application domain vs. approaches (classification)
RQ_3 (Critical quality attributes)	V1–V3, V9–V12	Descriptive Statistics (sum, average, frequency analyses, etc.) Heatmap based on critical quality attribute and year Bubble chart on critical quality attribute vs. approaches (classification) vs. application domain Spearman correlation between critical quality attribute and approaches (classification), and application domain
RQ_4 (Tools)	V1–V3, V9, V10, V14	Descriptive Statistics (sum, average, frequency analyses, etc.) Classification based on keywording
RQ_5 (Evidence type)	V1–V3, V9, V10, V15	Descriptive Statistics (sum, average, frequency analyses, etc.) Heatmap based on type of evidence and year Bubble chart on type of evidence vs. approaches (classification) vs. application domain Spearman correlation between type of evidence and approaches (classification), and application domain

practical. From weakest to strongest, the classes are: (i) no evidence; (ii) evidence obtained from demonstration or working out toy examples; (iii) evidence obtained from expert opinions or observations; (iv) evidence obtained from academic studies (e.g., controlled lab experiments); (v) evidence obtained from industrial studies (i.e., studies are done in industrial environments, e.g., causal case studies); and (vi) evidence obtained from industrial application (i.e., actual use in industry).

4 Results

In this section, we present the results of the mapping study, highlighting the most important observations. We note that the complete information from data extraction is publicly available as part of the supplementary material for this paper [24]. We clarify that, when necessary, we cite specific primary studies using an "S" (e.g., [S134]). Due to space limitations, we do not provide the list the primary studies in this manuscript, but we have made it available as a supplementary material [24].

4.1 Demographic Overview

The distribution of studies, per year, among the different types of publication (conference, journal and book) is depicted in Fig. 2. We clarify that we collected studies published up to March of 2013 (see Sect. 3.2), resulting on the observed smaller number in that year. We notice a linear growth in the number of conference papers. The number of journal articles experiences a growth as well, but not as high. We note that conference proceedings published as books were counted as conferences, explaining the small number of book chapters in the chart.

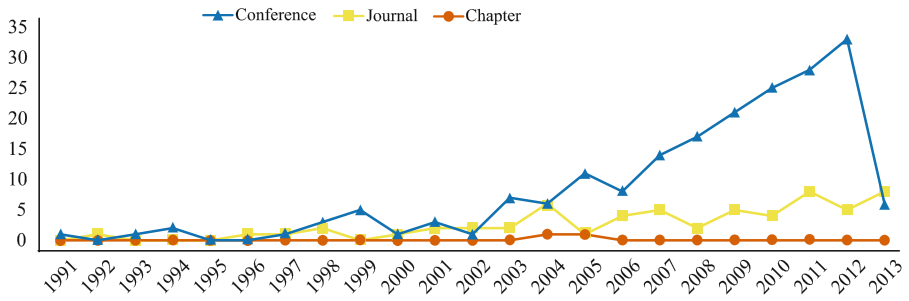

Fig. 2. Number of filtered studies per year, per type of paper.

To investigate further potential reasons for the aforementioned growth, we looked at the venues and checked whether they focus on CES alone, or have a broader scope (e.g., embedded systems) and only include CES as one of the topics of interest. We observed that, although a few venues do focus on CES (e.g., Brazilian symposium on CES), most of the studies were published in other venues, suggesting a shift or growing interest of the respective (broad) communities towards CES. In addition, we can try to

identify the most relevant venues, by looking at their distribution according to two metrics: number of included studies (Fig. 3a), and number of citations (Fig. 3b). We chose these metrics, because they reflect distinct features that may draw the attention of researchers to venues: the size of the CES community within the venue, and the potential visibility of the study. To investigate the venues, we analyzed how they are distributed statistically, identifying the high outliers, which in this case indicate popular venues for CES. We used the software IBM SPSS Statistics to create the box-plots as well as to identify the outliers, using the stem-and-leaf diagram.

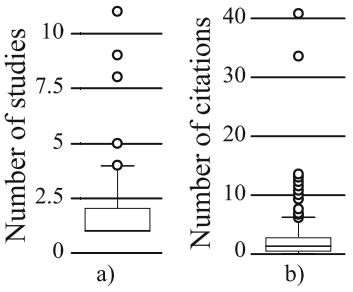

Fig. 3. Box-plot of venues based on (a) number of studies and (b) citations per paper per year.

On the one hand, Fig. 3a shows that the vast majority of venues contributed with one or two papers only, respectively 111 (approx. 70%) and 28 (17.5%). The analysis suggests that venues that contributed with four papers or more (nine venues) are exceptional in our dataset. On the other hand, Fig. 3b shows that most venues (85%) exhibit a maximum average of four citations per paper per year. The analysis of this metric suggests that venues with an average citation rate of 6.2 or more (15 venues in total) are also exceptional. Thus, we identified a set of 22 exceptional venues, which, due to space limitations, is presented in the supplementary material [24].

4.2 Design Approaches

As shown in the previous section we were able to collect a large number of studies. Therefore, it is infeasible to present all collected approaches here. For that reason, we decided to present the results as a summary based on the types of approaches that were found, which are based on a classification scheme (presented below). In addition, we present some details on the most relevant approaches, i.e., those with the most citations, identified by using the number of citations according to Google Scholar. To avoid omitting relatively new papers (i.e. those that did not have enough time to receive citations), we considered the number of citations per year. In the next subsections, we elaborate on this classification scheme and results.

Classification Scheme. The design phase in a development lifecycle is often elusive, in the sense that it is typically hard to determine the boundaries of design with respect to the other lifecycle phases. In embedded systems development, including systems

with harder constraints such as CES, this is no exception. However, in order to classify the design approaches, it is necessary to identify the parts of the development lifecycle that approaches belong to, i.e., their purpose. It is widely accepted that the design phase includes activities that translate requirements into software/hardware elements, with their respective responsibilities, excluding the actual implementation of these elements (source code) [1, 25, 26]. To initialize our classification scheme, we collected the keywords obtained from the keywording process (see Sect. 3.4) and filtered those that regard the purpose of approaches. Next, we grouped the keywords by similarity, trying to organize them in a hierarchical fashion, also creating a generic design flow[2]. However, it was not possible to derive such hierarchical organization, as we were not able to identify or define a flow that was sufficiently generic to accommodate the extracted approaches. This is due to the high heterogeneity of domains, requirements, and platforms for which CES are designed [1]. Therefore, we decided to organize our keywords based on a simplified design flow proposed by Marwedel [1], which is meant to generically represent the design activities of an ES.

To create our classification scheme, we successfully mapped the identified keywords into some elements of the design flow proposed by Marwedel [1], and assessed whether or not the relationship between the keywords were consistent with the description of the simplified design flow. By the end of the keyword mapping, we were able to derive five types of activity representing general purposes, as well as scope them and their relationships. The final classification scheme is presented in Fig. 4, in which rectangles represent each general purpose, and arrows show the flow of design artifacts. Moreover, smaller rectangles (i.e., Optimization and Test) represent auxiliary purposes that are special for the design of embedded systems. The approaches are grouped according to how they modify the system's design, rather than based on a logical sequence of activities. In addition, common activities in embedded system design are also clearly placed within the classification (e.g., scheduling is placed within Application mapping). The main characteristic of this kind of classification is that it is artifact-centric, i.e., the artifacts dictate what activities may be performed (i.e., what

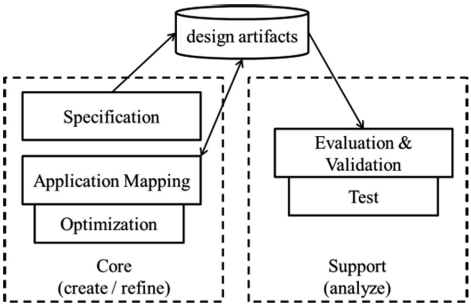

Fig. 4. Classification scheme.

[2] A design flow is the sequence of specific activities (with respective approaches) to design a system.

purposes they serve), rather than the other way around [1]. The five general purposes are described as follows:

- **Specification:** these activities formalize constraints (e.g., safety requirements) in the design. They define the scope/boundaries of the design. To draw a parallel, this type of activity is similar to the analysis in a software architecture design flow [27]. Common examples are formal specification languages, such as Z.
- **Application Mapping:** these activities generate new (partial) design information. A series of mappings are applied in order to refine the design from a more abstract representation to platform-specific design. In a software architecture design flow, this type of activity is similar to architecture synthesis [27]. Common approaches encompass: mapping of operations to concurrent tasks; mapping of operations to HW/SW; compilation; or scheduling.
- **Evaluation & Validation:** similarly to the evaluation in a software architecture design flow [27], these activities evaluate design elements w.r.t. the objectives (e.g. provide a proper scheduling of tasks) and validate a design description against other descriptions. Examples of approaches are algorithms or analysis frameworks for comparing models that tackle different quality attributes, as well as simulations.
- **Optimization:** these activities perform design tuning according to stated objectives. Examples of approaches are HL transformation and energy optimizations.
- **Test:** these activities include test generation and testability evaluation. They are included in design iterations if testability issues are already considered during the design steps. Tests are run after the design phase.

This classification is sufficiently robust for expressing different software, hardware and SW/HW design flows, including prominent ones such as the V-Model [28] and the design flow provided with SpecC [29]. Finally, it is important to clarify that approaches may serve several purposes. For example, some architecture modeling languages are able to perform both application mapping and specification.

Summary of Design Approaches. To analyze the extracted approaches, we classified each of them into one or more of the aforementioned general purposes. In addition, some studies presented entire design flows and, therefore, we also considered it as a category for the classification. Figure 5 depicts a heat map that shows the number of studies, per year, discussing approaches from each category.

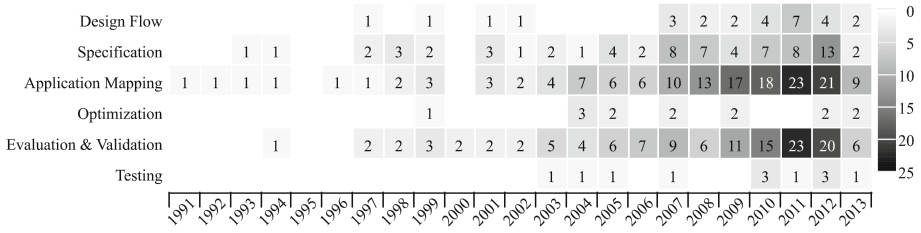

Fig. 5. Number of studies, per year, containing approaches from each category.

In this heat map, darker shades of grey represent bigger numbers, which are presented as well. For example, in 2011, 23 studies that contain approaches for application mapping were published. One can notice that most attention has been given to approaches for Application Mapping and Evaluation & Validation, which is understandable because approaches that serve this purpose encompass most of the design flow of an embedded system. Approaches for Specification of CES design were also presented in a considerable number of studies. Such interest is explained by the necessity of unambiguously representing the different aspects of CES (e.g., safety, components, security) in a variety of platforms (e.g., time/event-triggered and mixed architectures, and communication protocols). Table 4 presents the number of studies in each category, grouped by nature (i.e., academic, industrial or mixed). The table also presents the number of citations per year, for the entire set of studies. By exploring this table, one can notice that most of the studies were performed in an academic setting, followed by mixed and industrial settings, respectively; this is understandable as the included venues are more academic than industrial. In addition, solutions are normally proposed and explored in academic studies before they are applied in industry. However, there is one interesting observation to highlight. The mixed setting does not follow the same trend of the academic and industrial settings (which are in accordance to Fig. 5): studies performed in collaboration between academia and industry were mostly focused on Evaluation & Validation approaches, rather than Application Mapping, suggesting that the main interest of academic-industrial collaborations may be for evaluation & validation approaches. This finding may be partially explained by analyzing the number of citations per year. This number tends to follow the number of studies in the categories (i.e., more studies would result in more citations). However, there is one exception to that: industrial studies have more citations than mixed studies, w.r.t. approaches for Application Mapping, possibly due to increased industrial interest. By investigating the approaches we observed that: (a) almost all studies propose or consider formal approaches; (b) model-driven and component-based approaches are

Table 4. Classification of included studies by type of activity and nature.

Type of activity	Metric	Nature			Total
		Academic	Industrial	Mixed	
Design flow	Number of studies	16	6	6	28
	Citations/year	*65,05*	*8,71*	*18,48*	*92,25*
Specification	Number of studies	44	11	16	71
	Citations/year	181,84	31,30	39,50	252,64
Application mapping	Number of studies	97	21	32	150
	Citations/year	*298,42*	*85,97*	*72,33*	*456,72*
Evaluation & validation	Number of studies	74	17	36	127
	Citations/year	*232,66*	*22,33*	*73,50*	*328,49*
Optimization	Number of studies	11	1	2	14
	Citations/year	*28,81*	*0,12*	*3,19*	*32,11*
Testing	Number of studies	7	2	4	13
	Citations/year	*31,96*	*2,40*	*6,83*	*41,19*

preferred for tackling CES problems, specially due to the facilitation of (semi-) automatic verification and code generation; and (c) one of the most prominent challenges in designing CES, is the design of systems with mixed-criticality (i.e., critical and non-critical elements co-existing within the same system). In the following, we present the most important observations regarding each of the categories.

Multiple *design flows* have been proposed so far, which is in accordance to the high heterogeneity of CES. Each design flow aims at tackling specific problems, such as multi-tasking in multi-periodic synchronization [S206] or reliability-driven design in CES with mixed criticality [S257]. The most important observation is that the majority of the design flows didn't provide a complete lifecycle. They rather described how to tackle the specific issue within the system design. These incomplete flows are not surprising because every single CES entails a rather unique set of requirements that are tackled by combining different approaches. The most relevant studies are a generic design flow (from 1997) that served as inspiration to other flows [S16] and a safety-oriented and component-based design flow for vehicular systems [S102]. Approaches for design *specification* consist mostly of (semi-)formal languages or notations for representing different types of problems, such as specific forms of scheduling [S117, S225], or classes of constraints (commonly related to quality attributes such as safety or reliability) [S87, S244]. We highlight that most studies presenting specification approaches (approx. 80%) also presented approaches with other purposes (e.g., application mapping or evaluation & validation). The most relevant studies include the specification of time constraints in systems with mixed criticality [S225] and formal specification of safety constraints on higher-level design [S180].

The majority of the studies involve a variety of approaches for *Application Mapping*. Among these studies, approx. 30% proposed architectural approaches, i.e., architectures [S35, S94] or approaches for designing architectures (e.g., styles or patterns) [S121, S166]. We highlight that in the context of CES, communication architecture (e.g., time-triggered architecture [S35]) is a more relevant kind of architecture, due to its relevance on evaluating the hard constraints CES are subject to. In fact, this relevance is also evident by another common topic: scheduling of tasks/components, which corresponds to approx. 21% of the studies. Scheduling poses several challenges, from guaranteeing of time allocation to specific components, to integration with other models (e.g., fault-tolerance) to provide more accurate scheduling. Another common topic is software patterns, corresponding to approx. 9% of the studies, among which, design patterns were the most investigated [S105, S106, S137, S160, S259], followed by architectural [S121, S201], fault-tolerance [S191] and process patterns [S240]. As for the remainder of the studies, other scattered topics can be observed, from which the most recent/recurrent encompass approaches for modeling components w.r.t. various critical constraints (e.g., safety) and integration of models. The most relevant studies include the time-triggered architecture [S35], remote agent architecture [S13], a component-based approach for modeling safety [S102] and an approach for scheduling of mixed-criticality workload [S164].

Approaches for *Evaluation & Validation* comprise mostly formal methods for evaluating specific aspects of the design, such as scheduling of tasks [S51, S140, S225], fault-tolerance [S151, S192] and safety requirements [S74, S102]. In addition, there is a growing interest on model-driven approaches and object-oriented design.

Classical approaches for verifying safety and reliability (e.g., fault-tree analysis – FTA – and failure mode and effects analysis) have been adapted to new design paradigms. For example, a component-based FTA was proposed in [S128] aiming at facilitating the certification of systems by reusing certified components. In addition, exploratory-based evaluation approaches (e.g., prototyping and simulation) are also broadly explored in order to evaluate designs [S21, S102, S168, and S216]. The most relevant studies present formal approaches for evaluating reliability and safety [S8, S225], as well as safety evaluation based on simulation [S102].

Finally, regarding *Optimization* and *Testing* approaches, the approaches are used for the same reason: improving the evaluation & validation of the designed systems [S51, S186, and S261]. Most of the approaches, including the most relevant approaches, tackle time constraints [S51, S248] and fault-tolerance issues [S48, S151].

4.3 Application Domains

The results on application domains suggest that the most studies (approx. 57%) report generic approaches, from which approx. 9% showed examples on specific application domains, e.g., automotive [S149, S257] and avionics [S225, S166]. Figure 6 presents the distribution of the studies, per year, according to the application domains. For comparison purposes, we plot the amount of studies reporting generic approaches. We note that studies that report approaches for specific domains often refer to more than one domain, e.g., support the design of avionic and space systems [S161].

By observing Fig. 6, we notice that, besides constituting the majority, the number of studies reporting generic approaches is growing more than for any specific domain. This may suggest a trend or intention to work on unified technologies for developing CES. However, we also notice that the combined number of studies that focus on specific domains comprise almost half (approx. 48%) of the papers. Among the specific domains: avionics and automotive present the biggest growth. On the one hand, avionics is historically among the first application domains of CES and contains special regulations, which make the interchange of approaches more difficult. On the other hand, the automotive industry has been going through a series of technological innovations to provide several new features such as autonomous driving.

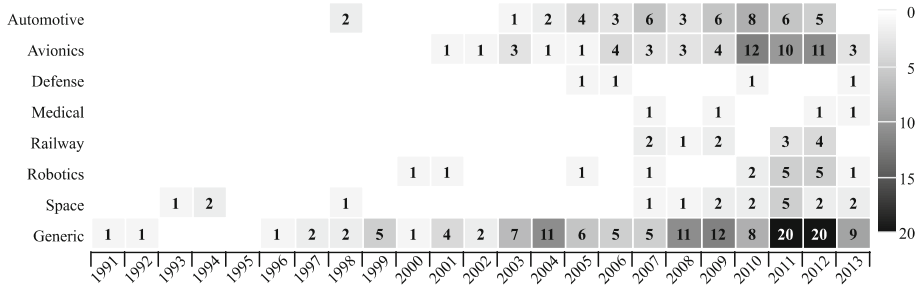

Fig. 6. Number of studies per application domain, per year.

To further analyze the influence of application domains on design approaches, we classified the primary studies according to their purpose. Table 5 presents the distribution of studies in each application domain among the five general purposes. We note that approaches serving more than one general purpose are counted for each of them. Based on Table 5, we observe that the distribution of studies on the application domains tend to be similar to the general distribution (Table 4). However, there is an exception for the medical and defense domains, as most studies report approaches for evaluation & validation rather than for application mapping. This may be either related to the low number of studies, or suggests a focus on this type of activity, perhaps motivated by specific industry standards or requirements of these domains. Another exception is that in the robotics domain the number of approaches for application mapping is quite higher (almost double) compared to evaluation & validation. Such disparity may be related to a larger variety of potential systems designs (large design space), which could result in more possibilities for mapping elements of the system. The disparity may also be related to a less regulated application domain that could in turn facilitate new design ideas to be implemented or experimented with.

Table 5. Classification of primary studies by domain and purpose.

Domain	Purpose					
	Design flow	Specification	Application mapping	Evaluation & validation	Optimization	Test
Automotive	7	11	31	22	2	2
Avionics	7	20	32	30	0	4
Defense	0	1	1	4	0	1
Medical	0	1	1	3	0	0
Railway	3	5	7	7	0	2
Robotics	2	3	13	6	0	1
Space	5	8	13	12	0	3
Generic	13	36	77	61	13	5

4.4 Quality Attributes

CES are subject to constraints on critical quality attributes (CQA). In this section, we report on the CQAs that are tackled within each primary study, using the original terms of CQAs that are used in the studies (i.e. those terms used by the authors). Even though some qualities are similar (e.g. dependability, fault-tolerance and reliability) we have not tried to merge them. Our goal is not to create a new quality model, but to simply present how authors express the hard-constraints of CES. However, we checked whether each term has the same or similar definition among the authors (e.g., if security is always used to convey the same concerns). We further discuss the relationship between CQAs and their definition in Sect. 5.1. We note that each study may tackle one or more CQAs. In Fig. 7, we present the number of studies, per year, tackling each critical quality attribute. We excluded two CQAs from this chart (power constraints and correctness) due to low number of papers (6 and 7, respectively).

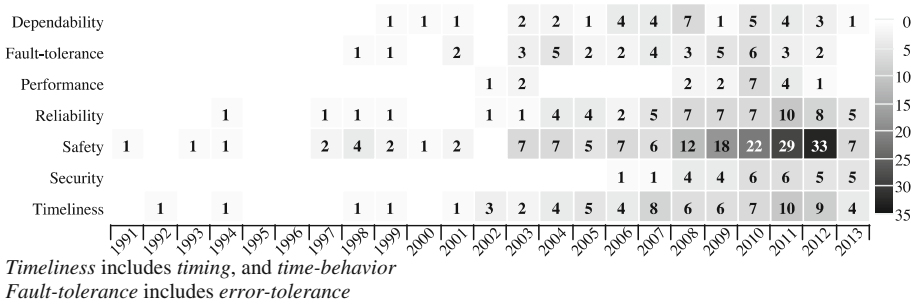

Fig. 7. Number of studies tackling quality attributes, per year.

By observing Fig. 7, one can notice that the interest in the different CQAs has grown in a similar fashion, except for safety, which shows higher growth. Such interest is not surprising, as safety is a very common and challenging concern among CQAs. In addition, the emergence and/or growth of application domains such as automotive, home automation, unmanned vehicles (e.g., drones) that are intrinsically centered on safety, have likely contributed to the observed growth. It is also relevant to point out that, although less intense, the interest in timeliness and reliability has also grown more than the remaining CQAs. The aforementioned arguments regarding safety, may also explain this observation. For example, the interest in multi-core platforms, as well as systems with mixed-criticality requires careful scheduling of tasks, and assurance that no interference between system parts with different criticality.

To further characterize the primary studies, we investigate them with respect to purpose and application domain. In Fig. 8, we present a bubble chart that depicts the distribution of the studies, based on CQAs (Y axis), with regards to the general purpose

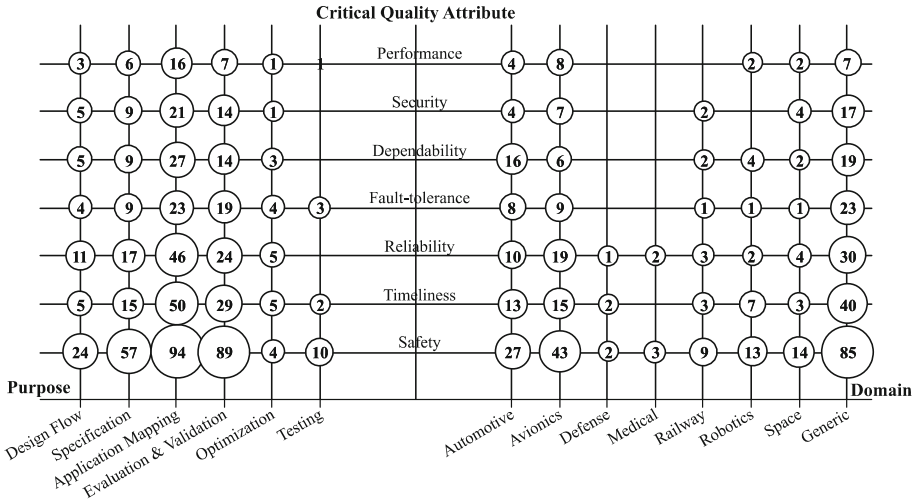

Fig. 8. Classification of studies based on quality attribute, purpose, and application domain.

(X axis—left side) and the application domain (X axis—right side). The size of the bubble represents the number of studies, which is shown inside the bubble. On the one hand, the distribution of studies among purposes, for each CQA, is similar compared to each other as well as compared to the general data (see Sect. 4.2). To confirm that, we calculated the spearman correlation between every pair of CQA and against the general data. All results were statistically significant and showed strong correlation (minimum coefficient of 0.899). This suggests that the distribution of research effort among different purposes is independent of CQAs. On the other hand, it is possible to observe a variation in the distribution of studies among application domains. For example, we notice that dependability displays a higher interest on the automotive domain (i.e., approx. 20% of the papers tackle this CQA), when compared against the average number of papers on dependability across domains (9%). We further investigated this observation by calculating the correlation between every pair of CQA, which showed that dependability has a weaker correlation with other CQAs (e.g., 0.667 with performance). This may suggest that these application domains are characterized by different constraints for the respective CQAs.

4.5 Tools

During the data extraction, we observed that approx. 53% of the papers either proposed or explicitly mentioned the use of specific tools. We also identified several Reference Technology Platforms (RTPs) [30], which consist of a set of approaches (e.g., methods, workflows) and tools providing a generic solution that can be tailored to various applications. The RTPs extracted in our study are all part of large projects involving multiple partners from both academia and industry. In total, we identified 147 tools of different kinds (e.g., CAD, model checkers, tool suites, etc.) and with various purposes (e.g., specification, application mapping, etc.). In addition, we noticed that some specification and/or modeling languages are an important part for many of these tools, e.g., serving as input format and base of the tool, or as exchange format between different tools. Therefore, we considered it relevant to include these languages in the results. Due to the number of identified tools, we summarized the results based on the general purposes presented in Sect. 4.2.

Table 6 shows the number of tools identified for each category (i.e., purpose). Within each category, we were able to define certain subcategories of tools representing specific purposes. We note that we include RTPs and IDEs (Integrated Development Environment), into the *Design Flow* category, as they support entire sets of activities. We also note that similar to approaches every tool may be classified in more than one category, e.g., a modeling tool that can import and export different models (i.e., *Application Mapping* category) as well as analyze them (i.e., *Evaluation & Verification* category). Furthermore, we note that the number of tools for subcategories do not necessarily add up to the number of the parent category. On the one hand, we only present subcategories with at least 3 tools (i.e. there were more subcategories with only 1 or 2 tools). On the other hand, tools may serve more than one purpose, which also affects subcategories. For example, SPIN is a verification tool with model checking and simulation capabilities, thus, counting for two subcategories. In the following we provide a brief description and the purpose of some relevant tools/languages, which we

Table 6. Summary of identified tools.

Purpose	Number of tools
Design flow	12
IDE	6
RTP	6
Specification	15
Notation/specification language	*12*
Programming language	*3*
Application mapping	35
Cad	*14*
Model transformation	*5*
Evaluation & validation	32
Simulation	*9*
Model checking	*9*
Optimization	1
Testing	2

identified based on the number of studies referring to the tool/language, as well as on the amount of citations these studies have. Due to space limitations a detailed discussion of tools and languages is omitted from this manuscript, but discussed in detail, in the supplementary material [24].

Summary of Languages. In Table 7, we list the top five recurrent languages within the primary studies, i.e., those discussed by three or more papers. We consider these languages relevant also due to the amount of citations obtained by the studies that refer to them. We observed that most languages are mentioned indirectly, i.e. not being the focus of the paper. For example, the Promela language is recurrent because researchers are interested in the SPIN verification tool, which defines models in Promela. In addition, most languages are also not specific to CES, although they are heavily used for this class of systems. Languages (e.g., Z) were created to enable representation of formal/mathematical constraints, which are common to CES.

Table 7. Highlighted languages.

Language	Number of studies	Number of citations	CES specific
AADL	20	294	Yes
Promela	7	162	No
SystemC	7	51	No
Z	5	153	No
EAST-ADL	3	19	Yes

Summary of Tools. The top five tools according to the number of studies and citations are presented in Table 8. We observe that most tools are not specific to designing CES. We believe this is related to the fact that most tools in this list have Evaluation & Validation purposes. Tools from this category, are mainly focused on ensuring the hard

constrains imposed w.r.t. meeting critical quality attributes; such CQAs are not particular to CES only. Finally, we notice that the tools focused on CES are mostly (a) from the Application Mapping category (e.g., modelling tools and schedulers), which are specialized for one or a group of application domains; and (b) RTPs and IDEs, which are tailored for this class of systems, and normally include some tools that are not specific to CES (e.g., verification tools).

Table 8. Highlighted tools.

Tool	Number of studies	Number of citations	CES specific
Simulink	15	132	No
UPPAAL	8	79	No
DECOS	7	164	Yes
SPIN	7	162	No
NuSMV	4	112	No

4.6 Evidence Type

To investigate the maturity of the primary studies, we considered the type of evidence they provide. For that, we use the classification proposed by Alves et al. [23], as mentioned in Sect. 3.5. At the lowest level, the primary study does not provide any evidence, whereas at the highest level, the study provides evidence from actual use of the approach within an industrial application. In Fig. 9, we present the distribution of the primary studies, per year, according to the evidence type. By observing Fig. 9, one can notice that the amount of studies that provide evidence from academic studies has been growing considerably, exhibiting the highest growth among the six types of evidence. This also reflects the fact that most primary studies (approx. 55%) are supported by such type of evidence. This result is understandable, as studies performed in academic settings usually have a lower threshold to conduct than those performed in industrial settings. In addition, considering the hard constraints of CES, multiple studies may need to take place before a mature technology emerges and industrial studies can be performed. Interestingly, the second most common type of evidence is industrial studies (approx. 20%), which is one step further according to the classification of Alves et al. [23], and may suggest successful transition of a fair number of technologies to industrial maturity level.

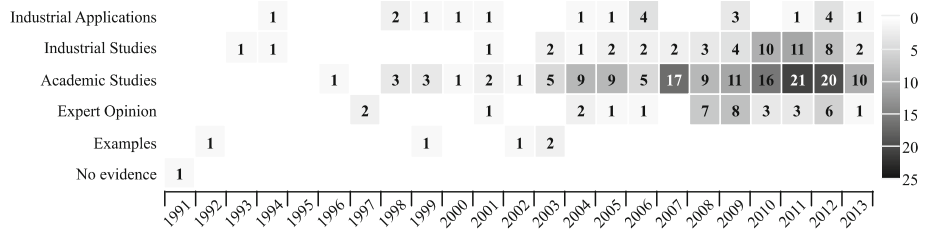

Fig. 9. Number of studies per type of evidence, per year.

Another interesting observation is that most studies are distributed among higher levels of evidence (academic studies, industrial studies and industrial applications). This may be, again, a consequence of the hard constraints imposed to CES, as tackling them would require stronger evidence to support the reported results. Another complimentary reason may be that embedded systems have been extensively investigated already, and management of hard constraints is not a new research topic for this class of system. Therefore, much of the exploratory research that has been done for embedded systems is now reused to investigate CES. To further investigate the evidence type, we classified the studies according to the purpose that their approaches serve, as well as the application domain. Similar to Figs. 8, Fig. 10 depicts the distribution of the studies, based on evidence type (Y axis), with respect to the purpose (X axis—left side) and the application domain (X axis—right side).

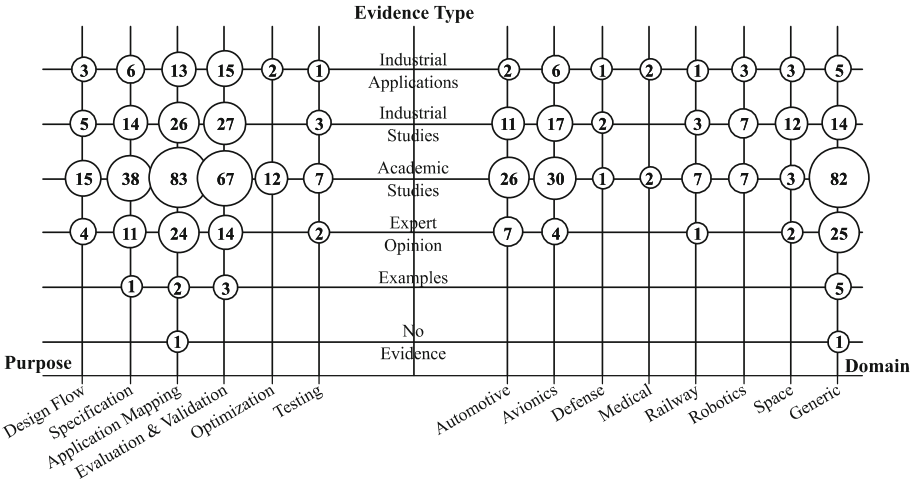

Fig. 10. Classification of studies based on evidence type, purpose, and application domain.

When verifying the distribution according to purpose, we observe that it follows a similar trend to that of the general data (presented in Sect. 4.2). We checked this hypothesis by calculating the correlation between each pair of evidence type, which showed a minimum correlation coefficient of 0.900. Conversely, while a visual inspection of the distribution according to domain suggests similarities between evidence types, the statistical correlation reveals minor differences between types of evidence, with coefficients varying from 0.500 to 0.927. These minor differences suggest that the application domain may affect what kind of research is performed.

5 Discussion

5.1 Relationship Between Quality Attributes

The approaches investigated in this mapping study tackle various CQAs, as presented in Sect. 4.4. While investigating this research question (RQ_3), we recorded the CQAs as used by the authors, i.e., we neither grouped nor merged any quality attributes, based on the definition used or implied in the primary studies. However, it is undeniable that some CQAs are related and, therefore, the identified quality attributes should be further investigated/synthesized. In this subsection, we group CQAs that have a similar or related meaning and map them to a quality model. For this purpose, we consider: (a) the SQuaRE quality model [31] which is a well-known quality model adopted by both researchers and practitioners; and (b) the ISO/IEC/IEEE vocabulary for system and software engineering [32], which is used within SQuaRE and provides additional definitions. We note that other quality models could be used to map the CQAs and that we do not assume that SQuaRE is the best model. We selected this model due to our experience with it and the possibility to fit all our recorded CQAs and observed terminologies. In Table 9 we present the CQAs identified in this study (presented in Sect. 4.4) on the right, and the characteristic (i.e., quality attribute) from SQuaRE to which they are mapped on the left. We note that SQuaRE presents a set of characteristics (left column of Table 9) and sub-characteristics (e.g. sub-characteristics of Performance Efficiency are Time Behavior, Resource Utilization and Capacity), which were both used to map CQAs. In addition, a CQA can be directly related if the terms are equivalent (e.g., *safety* maps to *freedom from risk*), or indirectly related if it is one of the aspects of the main quality attribute (e.g., *correctness* is a sub-characteristic of Functional suitability) or if it is related to one of them (e.g., *energy efficiency* regards Resource utilization, i.e., sub-characteristic of performance).

Table 9. Grouping and mapping of critical quality attributes.

CQA from SQuaRE	Identified CQA
Functional suitability	Correctness
Security	Security
Performance efficiency	Performance Energy efficiency Timeliness
Reliability	Reliability Fault tolerance Dependability
Freedom from risk	Safety

Correctness and *security* are directly mapped, since they similarly referred in the primary studies. However, the grouping of the remainder CQAs is not as straightforward. Performance efficiency is defined as the degree to which functionalities are delivered within given constraints [31], i.e., how well the system uses its resources to

accomplish the designed functions. This definition encompasses the interpretations of *performance*, *energy efficiency*, and *timeliness* among the primary studies. *Fault tolerance* is a well-known aspect of reliability and the interpretations of the authors meet the definition of the sub-characteristic in SQuaRE (also named Fault tolerance). Although dependability is commonly addressed as a separate quality attribute, we decided to map it to Reliability. Dependability is not part of SQuaRE but it is explained within the description of reliability. It comprises a more subjective definition, which is not easily quantifiable, and reflects whether or not a system can be trusted [32]. Due to its subjective definition, dependability is commonly improved through addressing other, more objective, quality attributes that can contribute to the trustworthiness of the system, in particular, reliability, maintainability, and availability. By observing the primary studies of our mapping, it is also clear that dependability is commonly used as proxy to other quality attributes, in particular, aspects of reliability, such as fault tolerance. Therefore, since the primary studies exploit dependability mostly as a proxy to reliability, we decided to group them together. *Safety* is another subjective CQAs, which is mentioned within SQuaRE's model for quality in use, i.e., how well the product can be used by specific users [31]. Similar to dependability, safety is commonly used as a proxy to other quality attributes, although not always the same ones. Particularly, safety is related to the avoidance of hazardous situations (i.e., that lead to endangerment of humans, environment or properties), which can originate from various sources, depending on the system. In our study, we identified connections between safety and various aspects: security [S215], performance, correctness [S50, S198] and fault-tolerance [S50, S84]. For example, when using a Time-Triggered Architecture (TTA) for communication (instead of an event-triggered one), timeliness become a safety threat.

In summary, CQAs as defined in primary studies are uniformly understood (i.e. their definitions are the same or similar across the studies) and that some can be grouped based on similarity. This culminated into the identification of five attributes: Functional Suitability, Security, Performance efficiency, Reliability, and Safety (Freedom from risk). We acknowledge that other CQAs may exist in individual cases depending on application-specific constraints. However, these five QA are by far the most recurrent ones. We also noticed that Safety is more abstract, since it depends on other CQAs. Therefore, is achieved by meeting requirements related to other CQAs. Furthermore, we note that identifying these CQAs is not always a trivial task as different components in the same systems may pose different constraints, i.e., may be subject to different kinds of hazards. A common approach to handle this mixed criticality is the use of integrity levels [33], which reflect the degree of compliance within a certain characteristic. Components with different integrity levels will be subject to different safety checks, which may also reflect the different concerns of that level. For example, the drive-by-wire feature is subject to hard reliability checks, while GPS navigation should only be assured to not interfere with the critical components. Therefore, it is important to identify and monitor the CQAs that are tightly related to safety.

5.2 Domain-Specific Research for CES

In Sect. 4.3 through Sect. 4.6, we presented an overview of the primary studies with respect to application domains, as well as how other facets (e.g., evidence type) related to domains. In summary, we did not notice major differences across application domains regarding which CQAs are the most relevant. This observation might be an indication that CQA-related challenges in CES are common to all application domains and have similar relevance. The only difference we observed was that studies focused on the automotive domain seem more concerned about dependability rather than reliability. However, these two fall under the umbrella quality of reliability in the SQuaRE model (see Sect. 5.1). Furthermore, we also notice that domains may influence the kind of research that is performed; for example, most studies on medical and defense domains focused on approaches for evaluation & validation rather than application mapping (as the general trend).

The difference between domains becomes clearer when looking at the type of evidence that studies provide (see Sect. 4.6). We separated the studies into three groups and verified their distribution among the different types of evidence (see Fig. 11). The three groups consist of studies that: (a) focus on a specific domain; (b) do not focus on any domain but present an example of application on a specific domain; and (c) neither focus nor present an example on specific domains. We notice that application domains become more relevant when a technology is being transferred to industry, as the two rightmost types of evidence (Industrial Study and Industrial Application) account mostly for studies that focus on application domains.

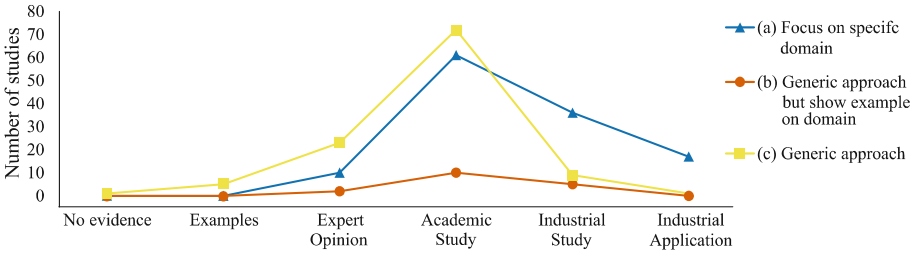

Fig. 11. Distribution of studies according to type of evidence and application domain.

It is understandable that studies conducted with industrial partners or in an industrial setting are focused on specific domains, as companies are by and large interested into applying approaches on certain products, which in turn fall under specific domains. As expected, generic approaches that solve domain-independent problems are first validated in academic settings, and subsequently find applications in industry that in turn customize and validate them in specific application domains. The opposite is also possible: there are also technologies that initially emerge as domain-specific solutions and are later applied to other domains. For example, the Architecture

Analysis and Design Language (AADL) was standardized by the Society of Automotive Engineers (SAE) with focus on the avionics domain[3] and is currently being applied in other CES domains.

5.3 Relationships Among Approaches, Tools, and Languages

The data analysis in this SMS resulted in the identification of many concepts related to the research questions, namely approaches, tools, languages, critical quality attributes, and application domains, as well as relationships between them. While we were able to present and discuss all CQAs and application domains found in the primary studies (see Sects. 4.3, 4.4, 5.1 and 5.2), the amounts of approaches, tools and languages was too large to present and discuss all concepts and relationships. To tackle this issue, we created a concept map to help us visualize these approaches, tools, and languages and identify relevant findings.

The concept map was created as a webpage that features an interactive interface, which is available[4]. To avoid loss of information, we also created a text version of the concept map. The text version and source code of the web version are available within the supplementary material [24]. In Fig. 12, we show a screenshot of the concept map and its interface. The concept map consists of a network in which nodes represent concepts and edges relationships. Each type of concept (i.e., approach, tool or language) is represented by an icon for easy identification. Upon clicking on a concept, an information panel is prompted on the right side, showing: (a) name of the concept, which is a link if a URL (Uniform Resource Locator) is available (shown by the chain icon next to the name); (b) a brief description of the concept; (c) the list of purposes, according to our classification scheme; and (d) a list of relationships (i.e., links) attributed to the concept. The relationship between concepts can be of two types:

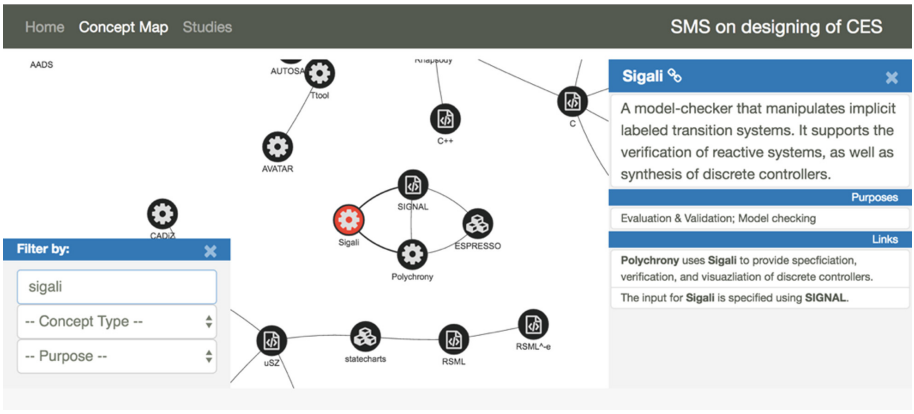

Fig. 12. Screenshot of the concept map interactive interface. (Color figure online)

[3] Note that SAE does not limit itself to the automotive domain.

[4] http://feitosa-daniel.github.io/sms-ces-design.

"use/is used" (e.g., "Polychrony uses Sigale to provide specification … of discrete controllers"), or "is kind of" (e.g., "SystemC is a subset of C++").

The interface also provides a feature to filter concepts based on name, type of concepts, or purpose. Upon typing on the name field or selecting type of concept or purpose, the filtered items are highlighted in red (see Fig. 12). For example, in the screenshot we typed "sigali" and the tool "Sigali" was automatically highlighted (the search looks for partial matches and is not case sensitive). After that, we clicked on the node, which prompted the information panel on the right. Finally, the interface is responsive, i.e., it adapts to different screen sizes (e.g., smartphones), which improves the usability of the concepts map.

Based on the concept map, we can make several observations. However, due to space limitations, we provide only one of them, also explaining how we identified it. We note that the main purpose of the concept map is to support the investigation of its concepts by third-parties and, therefore, we encourage the reader to further analyze it. The Architecture Analysis and Design Language (AADL) appears to be a rather mature technology. The results of the study showed that AADL is cited in multiple papers (see Sect. 4.5). In addition, by looking at the concept map we notice a fair number of related concepts (see Fig. 13) when compared against the average of 2.13 edges per node, and we notice that there are related concepts that serve different purposes: (a) specification, (b) application mapping, and (c) evaluation & validation. In particular, there is a toolset that is able to read AADL models, tools to evaluate AADL models and a language (EAST-ADL) that is partially derived from AADL.

Fig. 13. Part of the concept map surrounding AADL.

5.4 Implications to Researchers and Practitioners

The results and discussion presented in this SMS have potential value for both researchers and practitioners. The information compiled in this study may support readers that want to get acquainted with the design process of CES or may be interested in specific outcomes, e.g., identified CQAs and how they are tackled by primary studies. Researchers can use the information in this SMS to identify work that is related or that can contribute to theirs, as well as identify opportunities for future work. For example, researchers interested in a specific application domain have access pointers to

the existing literature, as well as how studies are distributed within the domain. We envisage similar learning opportunities to practitioners, through a more practical perspective. For example, practitioners can investigate a tool that is being considered for the designing of a new system or investigate the ecosystem around an approach, i.e., tools and related approaches.

In addition, we specifically aimed at the reuse of the information collected during our SMS when we created the concept map, which contains the complete set of approaches, tools and languages. Based on the information and features provided by the user interface, we believe that the concept map is valuable to both practitioners and researchers. Regarding practitioners, it can be used to support the exploration of problem and solution spaces while designing CESs. For example, using filters, one is able to search for approaches and or tools that fit the requirements of the systems (e.g., model-checking of models specified in SIGNAL). Also, if one has decided for a specific approach or tool, she can also explore related concepts and identify alternatives or tools that support the approach (e.g., tools that evaluate Binary Decision Diagrams). Regarding researchers, the concept map helps identifying potential links between different research results. For example, researchers interested into investigating a certain approach can use the concept map to easily visualize some of the involved approaches and tools that support it. We note that despite our great effort on collecting and analyzing the selected studies, the concepts and relationships presented in this map do not present the entire set of approaches, tools and languages available to design CES. Therefore, we hope that by providing access to the concept map, we can support others on developing it even further.

6 Threats to Validity

Concerning studies identification, the main threat is that the automatic search may not have been able to collect all relevant primary studies, i.e., the search string was not as inclusive as necessary or the considered digital libraries did not include all relevant venues. To mitigate this risk, we defined a gold standard and ensured that the automatic search returned all papers in the gold standard. In addition, we included digital libraries of the main publishers in the topic, and Scopus, which indexes papers from additional venues. Another potential threat is that the inclusion and exclusion criteria may have left relevant studies out of the final set of primary studies. This was mitigated not only by the usage of the gold standard but also by having key points of our protocol (e.g., inclusion and exclusion criteria) inspected by other external researchers with experience in CES. To mitigate risks related to data collection and analysis, we considered several strategies. The filtering of papers and data extraction involved at least two researchers on every step, while there were extensive discussions on topics such as selection criteria and understanding of CES terminology. In addition, the alignment of researchers involved in these steps where verified by calculating the Cohen's kappa coefficient between them. For data analysis, we applied frequency analysis, cross-tabulation and statistical tests, which are less prone to researcher bias. However, we acknowledge that our results are limited to the set of design approaches, CQAs, and application domains that were discussed in the collected primary studies. Although

considering non-peer-reviewed literature was out of the scope of our SMS, we argue that the digital libraries we considered, do catalog most of the work relevant to the research of CES design.

Finally, to mitigate replicability threats, the steps of our study were clearly stated in our protocol and can be reproduced by other researchers. However, we acknowledge that the reproduction of the SMS by other researchers may lead to slight different sets of primary studies due to biases, e.g., when applying the inclusion and exclusion criteria. We mitigated this threat to some extent by comprehensively documenting faced challenges and decisions made upon them. Thus, despite some potential minor differences, we believe that the results and observations would be predominantly similar in replication studies.

7 Conclusions

In this paper, we presented a Systematic Mapping Study (SMS) on designing Critical Embedded Systems (CES) that investigated five facets: (a) approaches for designing CES; (b) application domains for which these approaches are developed; (c) Critical Quality Attributes (CQAs) considered on these approaches; (d) tools used for designing CES; and (e) type of evidence provided by these approaches. We considered five digital libraries and collected an initial amount of 1673 primary studies, which were then filtered, resulting in 269 selected primary studies. Subsequently, we extracted and analyzed all data necessary to answer our research questions.

The results of our SMS show that the body of knowledge on designing CES is vast, and this is partially due to the overlap of knowledge with other classes of systems such as hard real-time systems. Results also suggest that the CQAs that are relevant to the design of CES, are common for this whole class of systems, i.e. they are mostly independent of application domain. The main contributions of our work are the classification scheme for approaches and tooling, the provided collection of CQAs and approaches (with associated tools), as well as the webpage that supports exploring this information. We believe that both researchers and practitioners can benefit from these contributions, taking advantage of our provided overview of this vast body of knowledge; they can thus focus on more relevant tasks such as identification of related and future work, and exploration of problem and solution spaces. Based on our results and observations we envisage several opportunities for future work. Among them, we highlight the possibility of investigating approaches that might be potentially beneficial to CES and have not being thoroughly explored yet, like using design patterns to improve levels of CQAs. The body of knowledge presented in this SMS has considerable overlap with other classes of system, thus we find it relevant to continue exploring such related classes (e.g., hard-real time systems) and seek approaches that can be applied to the designing of CES.

Acknowledgements. The authors would like to thank the financial support from the Brazilian and Dutch agencies CAPES/Nuffic (Grant N.: 034/12), CNPq (Grant N.: 204607/2013-2), as well as the INCT-SEC (Grant N.: 573963/2008-8 and 2008/57870-9).

References

1. Marwedel, P.: Embedded System Design: Embedded Systems Foundations of Cyber-Physical Systems. Springer, Dordrecht (2010). https://doi.org/10.1007/978-94-007-0257-8
2. Bate, I.: Systematic approaches to understanding and evaluating design trade-offs. J. Syst. Softw. **81**, 1253–1271 (2008)
3. Medikonda, B.S., Panchumarthy, S.R.: A framework for software safety in safety-critical systems. ACM SIGSOFT Softw. Eng. Notes. **34**, 1 (2009)
4. Aguiar, A., Filho, S.J., Magalhães, F.G., Casagrande, T.D., Hessel, F.: Hellfire: a design framework for critical embedded systems' applications. In: 11th International Symposium on Quality Electronic Design, pp. 730–737 (2010)
5. Ampatzoglou, A., Gkortzis, A., Charalampidou, S., Avgeriou, P.: An embedded multiple-case study on OSS design quality assessment across domains. In: Seventh ACM/IEEE International Symposium on Empirical Software Engineering and Measurement, pp. 255–258. IEEE (2013)
6. Linares-Vásquez, M., Klock, S., McMillan, C., Sabané, A., Poshyvanyk, D., Guéhéneuc, Y.-G.: Domain matters: bringing further evidence of the relationships among anti-patterns, application domains, and quality-related metrics in Java mobile apps. In: 22nd International Conference on Program Comprehension, pp. 232–243. ACM Press (2014)
7. Cawley, O., Wang, X., Richardson, I.: Lean/agile software development methodologies in regulated environments – state of the art. In: Abrahamsson, P., Oza, N. (eds.) LESS 2010. LNBIP, vol. 65, pp. 31–36. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-16416-3_4
8. Eklund, U., Bosch, J.: Archetypical approaches of fast software development and slow embedded projects. In: 39th Euromicro Conference Series on Software Engineering and Advanced Applications, pp. 276–283 (2013)
9. Kitchenham, B., Charters, S.: Guidelines for performing systematic literature reviews in software engineering. Engineering **2**, 1051 (2007)
10. Karlström, D., Runeson, P.: Integrating agile software development into stage-gate managed product development. Empir. Softw. Eng. **11**, 203–225 (2006)
11. Selim, G.M.K., Wang, S., Cordy, James R., Dingel, J.: Model transformations for migrating legacy models: an industrial case study. In: Vallecillo, A., Tolvanen, J.-P., Kindler, E., Störrle, H., Kolovos, D. (eds.) ECMFA 2012. LNCS, vol. 7349, pp. 90–101. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-31491-9_9
12. Barbosa, J.R., Delamaro, M.E., Maldonado, J.C., Vincenzi, A.M.R.: Software testing in critical embedded systems: a systematic review of adherence to the DO-178B standard. In: Third International Conference on Advances in System Testing and Validation Lifecycle, pp. 126–130 (2011)
13. Elberzhager, F., Rosbach, A., Bauer, T.: Analysis and testing of matlab simulink models: a systematic mapping study. In: 2013 International Workshop on Joining AcadeMiA and Industry Contributions to testing Automation, pp. 29–34 (2013)
14. Dybå, T., Dingsøyr, T.: Empirical studies of agile software development: a systematic review. Inf. Softw. Technol. **50**, 833–859 (2008)
15. Petersen, K., Feldt, R., Mujtaba, S., Mattsson, M.: Systematic mapping studies in software engineering. In: 12th international conference on Evaluation and Assessment in Software Engineering, pp. 68–77 (2008)
16. Antonio, E.A., Ferrari, F.C., Ferraz Fabbri, S.C.P.: A systematic mapping of architectures for embedded software. In: Second Brazilian Conference on Critical Embedded Systems, pp. 18–23 (2012)

17. Guessi, M., Nakagawa, E.Y., Oquendo, F., Maldonado, J.C.: Architectural description of embedded systems: a systematic review. In: Third International ACM SIGSOFT Symposium on Architecting Critical Systems, pp. 31–40. ACM (2012)
18. Nakagawa, E.Y., Gonçalves, M., Guessi, M., Oliveira, L.B.R., Oquendo, F.: The state of the art and future perspectives in systems of systems software architectures. In: 1st International Workshop on Software Engineering for Systems-of-Systems, pp. 13–20 (2013)
19. Basili, V.R., Caldiera, G., Rombach, H.D.: Goal question metric paradigm. In: Marciniak, J. J. (ed.) Encyclopedia of Software Engineering, pp. 528–532. Wiley, New York (1994)
20. Dieste, O., Grimán, A., Juristo, N.: Developing search strategies for detecting relevant experiments. Empir. Softw. Eng. **14**, 513–539 (2009)
21. Dybå, T., Dingsøyr, T., Hanssen, G.K.: Applying systematic reviews to diverse study types: an experience report. In: First International Symposium on Empirical Software Engineering and Measurement, pp. 225–234 (2007)
22. Zhang, H., Babar, M.A.: On searching relevant studies in software engineering. In: 14th International Conference on Evaluation and Assessment in Software Engineering, pp. 111–120. British Computer Society, Keele (2010)
23. Alves, V., Niu, N., Alves, C., Valença, G.: Requirements engineering for software product lines: a systematic literature review. Inf. Softw. Technol. **52**, 806–820 (2010)
24. Feitosa, D., Ampatzoglou, A., Avgeriou, P., Affonso, F.J., Andrade, H., Felizardo, K.R., Nakagawa, E.Y.: Supplementary Material: "Design Approaches for Critical Embedded System: A Systematic Mapping Study". https://doi.org/10.5281/zenodo.996480
25. Bass, L., Clements, P., Kazman, R.: Software Architecture in Practice. Addison-Wesley Professional, Upper Saddle River (2012)
26. Sommerville, I.: Software Engineering. Addison Wesley, Boston (2000)
27. Hofmeister, C., Kruchten, P., Nord, R.L., Obbink, H., Ran, A., America, P.: A general model of software architecture design derived from five industrial approaches. J. Syst. Softw. **80**, 106–126 (2007)
28. Bartelt, C., Bauer, O., Beneken, G., Bergner, K., Birowicz, U., Bliß, T., Cordes, N., Cruz, D., Dohrmann, P., Friedrich, J., Gnatz, M., Hammerschall, U., Hidvegi-Barstorfer, I., Hummel, H., Israel, D., Klingenberg, T., Klugseder, K., Küffer, I., Kuhrmann, M., Kranz, M., Kranz, W., Meinhardt, H.-J., Meisinger, M., Mittrach, S., Neußer, H.-J., Niebuhr, D., Plögert, K., Rauh, D., Rausch, A., Rittel, T., Rösch, W., Saas, E., Schramm, J., Sihling, M., Ternité, T., Vogel, S., Wittmann, M.: V-Modell XT Gesamt 1.3 (2010)
29. Gajski, D.D., Zhu, J., Dömer, R., Gerstlauer, A., Zhao, S.: SPECC: Specification Language and Methodology. Springer, New York (2000). https://doi.org/10.1007/978-1-4615-4515-6
30. Kacimi, O., Ellen, C., Oertel, M., Sojka, D.: Creating a reference technology platform - performing model-based safety analysis in a heterogeneous development environment. In: Second International Conference on Model-Driven Engineering and Software Development, pp. 645–652 (2014)
31. ISO/IEC: ISO/IEC 25010:2011 - Systems and software engineering – Systems and software Quality Requirements and Evaluation (SQuaRE) – System and software quality models (2011)
32. ISO/IEC/IEEE: ISO/IEC/IEEE 24765-2010 - Systems and software engineering – Vocabulary (2010)
33. ISO/IEC: ISO/IEC 15026-3:2015 Systems and software engineering – Systems and software assurance – Part 3: System integrity levels (2015)