What we know about testing embedded software

Vahid Garousi, Hacettepe University and University of Luxembourg

Michael Felderer, University of Innsbruck

Çağrı Murat Karapıçak, KUASOFT A.Ş.

Uğur Yılmaz, ASELSAN A.Ş.

Abstract. Embedded systems have overwhelming penetration around the world. Innovations are increasingly triggered by software embedded in automotive, transportation, medical-equipment, communication, energy, and many other types of systems. To test embedded software in a cost effective manner, a large number of test techniques, approaches, tools and frameworks have been proposed by both practitioners and researchers in the last several decades. However, reviewing and getting an overview of the entire state-of-the-art and the –practice in this area is challenging for a practitioner or a (new) researcher. Also unfortunately, we often see that some companies reinvent the wheel (by designing a test approach new to them, but existing in the domain) due to not having an adequate overview of what already exists in this area. To address the above need, we conducted a systematic literature review (SLR) in the form of a systematic mapping (classification) in this area. After compiling an initial pool of 560 papers, a systematic voting was conducted among the authors, and our final pool included 272 technical papers. The review covers the types of testing topics studied, types of testing activity, types of test artifacts generated (e.g., test inputs or test code), and the types of industries in which studies have focused on, e.g., automotive and home appliances. Our article aims to benefit the readers (both practitioners and researchers) by serving as an "index" to the vast body of knowledge in this important and fast-growing area.

Keywords. Software testing; embedded systems; embedded software; systematic mapping; systematic literature review

1 INTRODUCTION

Embedded software is computer software, written to control machines or devices that are not typically thought of as computers, e.g., cars and TV. According to recent surveys, approximately 90% of all processors are part of embedded systems, computing systems that continually and autonomously control and react to the environment. The embedded system itself is an information processing system that consists of hardware and software components. Nowadays, the number of embedded computing systems— in areas such as telecommunications, automotive, electronics, office automation, and military applications—is steadily growing [1].

Since software is a major component of embedded systems, it is very important to properly and adequately test the embedded software, especially for safety-critical domains such as automotive and aviation. Due to the complex system context of embedded-software applications, defects can cause life-threatening situations, and delays can create huge costs [2].

To test embedded software in a cost effective manner, various test techniques, approaches, tools and frameworks have been proposed by both practitioners and researchers in the last several decades. However, reviewing and getting an overview of the entire state-of-the-art and –practice in this area is almost impossible for a practitioner or a new researcher since the number of studies is simply too many. Also, we have seen in several cases that, unfortunately, many companies sometimes spend a lot of effort to reinvent the wheel (by designing a test approach new to them, but existing in the domain) due to not having an adequate overview of what already exists in this area. Knowing that they can adapt/customize an existing test technique to their own context can potentially save companies and test engineers a lot of time and money. The other main reason why we conducted this review was that in our recent and ongoing collaborations with multiple industry partners in testing embedded software, our colleagues and we have constantly faced numerous challenges in testing embedded software and we were uncertain of whether certain techniques already exist or we shall develop new techniques ourselves to solve those challenges. In addition, based on the authors' experience and based on the opinion of other practitioners and researchers (e.g., [3, 4]) the majority of software practitioners do not (actively) read scientific papers and there are many papers relevant for practice in the area of testing embedded software. Thus, we have observed first-hand that there is a major need for review papers like the current one to provide a summary of the entire field and serve as an "index" to the vast body of knowledge in this area, so that a practitioner can get a snapshot of the knowledge out there without having to find and read through each of the 200+ papers in this area.

Although there have been state-of-the-practice papers such as [2] on embedded software engineering and 'review' papers such as [5] in other venues, no paper has studied the entire state-of-the-art and –practice in a holistic manner, which is essential for the field of testing embedded software that is equally driven by academia and industry.

To address the above need and to identify the state-of-the-art and–practice in this area and to find out what we, as a community, know about testing embedded software, we recently conducted a systematic mapping on the technical papers written by practitioners and researchers and we present a summary of its results in this article. Our review pool included 272 technical papers published in conferences and journals, and the earliest paper [6] was published in 1984. Previous 'review' (survey) papers like this article have

appeared in the IEEE Software and other venues previously on other topics, e.g., about Agile development [7], and developer motivation [8], and have shown to be useful in providing concise overviews on a given area.

By summarizing what we know in this area, our article aims to benefit the readers (both practitioners and researchers) by serving as an "index" to the vast body of knowledge in this important and fast-growing area. Our review covered the types of testing topics studied, types of testing activities, types of test artifacts generated, and the types of industries in which studies have focused on. We start with a brief review of our review procedure.

2 THE REVIEW PROCEDURE

In our review and mapping, we followed the standard process for performing systematic literature review (SLR) and systematic mapping (SM) studies in software engineering [9, 10]. We performed the searches in the Google Scholar database. The four authors conducted all steps as a team. Our search string was: *(test OR testing OR validation OR verification) AND (embedded system OR embedded software)*. After compiling an initial pool of 560 papers, a systematic voting was conducted among the authors, and our final pool included 272 technical papers. The review aimed at addressing the following Review Questions (RQs):

- **RQ 1-** Types of testing activities: What types of testing activities have been conducted and proposed? Inspired by books such as [11] and our recent SM and SLR studies such as [12, 13], we categorized testing activities as follows: test planning and management, test-case design (criteria-based), test-case design (human knowledge-based), test automation, test execution, test evaluation (oracle), and other.
- **RQ 2-** Types of test artifacts generated: What types of testing artifacts are generated by the test techniques proposed? After reviewing a large subset of papers and in an iterative refinement manner, we categorized them as follows: test case requirements (not input values), test case input (values), expected outputs (oracle), test code (e.g., in xUnit) and other. Let us note that test requirements are usually not actual test input values, but the conditions that can be used to generate test inputs.
- **RQ 3-** Types of industries: We also wondered about the types of industries in which studies have focused on. While some test techniques are generic in that they can, in principle, be applied to all types of embedded software, some techniques are domain-specific. We categorized the types of industries (domains) into: generic; home appliances and entertainment; aviation, avionics and space; automotive; defense; industrial automation /control; medical; mobile and telecom; transportation; and other.

Both the academic community and industry are active in this area.

A more detailed description of our SM process is provided in the Web Extras of the article (goo.gl/by10jc). In addition, we discuss in the Web Extras how we identified and addressed the potential threats to validity to our review and its results. After discussing an overview of the pool of studies in the following sections, we present results for the review questions RQ 1 to 3.

3 POOL OF STUDIES: BOTH THE INDUSTRY AND ACADEMIC COMMUNITY ARE ACTIVE

The final pool of 272 papers and the online mapping repository can be found in an online Google spreadsheet (goo.gl/MhtbLD). In this article, we will use the format of "[Source 1]" to refer to the papers in the pool as listed in the online repository, as shown in Figure 1.

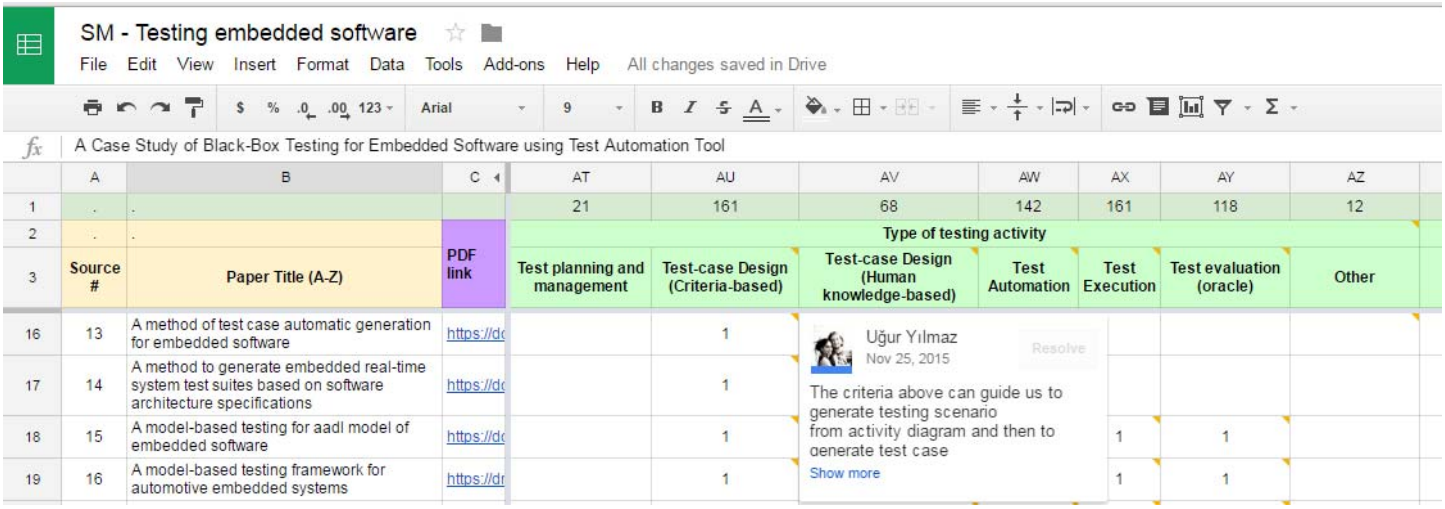

Figure 1- A screenshot from the online repository of papers

In Figure 2, we show (as a stack chart) the number of studies per year published solely by academic researchers (those working in universities and research centers/institutes), solely by practitioners (also including corporate research centers), or as collaborative work. As we can see, the attention level in this topic has steadily risen since early 2000's by both the research and practitioner communities. Note that since our study started in summer 2015 and finished in early 2016, we decided to include the papers published until end of 2014. The peak year in terms of number of papers was year 2012 in which 38 papers were published.

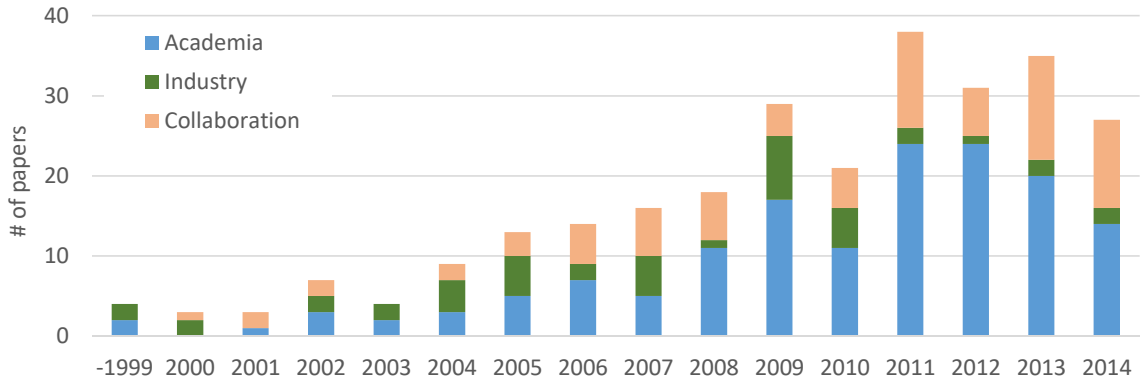

Figure 2-Growth of the field and affiliation types of the study authors across years

In terms of breakdown of papers by affiliation types of the authors, 151 papers were authored solely by academic researchers, 43 paper by practitioners only, and 78 papers as joint (collaborative) work between researchers and practitioners. In terms of the most active companies in conducting research and publishing papers in this area, the top three were: (1) Daimler (14 papers in the pool), (2) Samsung (8 papers), as well as (3) Nokia and Berner & Mattner (4 papers each).

4 VARIOUS TOPICS ARE BEING STUDIED

Before getting into details of extracting the data and classifying each paper according to our three RQs (discussed above), we first wanted to get an overview of the topics covered in the papers. Word clouds are a suitable tool for this purpose. Figure 2 shows a word cloud of all paper titles denoting the popularity of the topics covered (we used the online tool www.wordle.net). As we can see in this bird's eye view, topics such as model-based and automated / automatic (testing), (test-case) generation, and control systems are among the most popular topics.

Figure 3-Popularity of the topics shown by the word cloud of all paper titles

5 TYPES OF TEST ACTIVITIES

Many companies are active in conducting research and publishing papers in this area, especially major players in the automotive and home appliances industry.

Inspired by classifications made in software testing books such as [11] and our recent SM and SLR studies on software testing such as [12, 13], we categorized testing activities based on the generic test process shown in Figure 4.

Similar to other types of software systems, testing embedded software usually starts with test-case design (either criteria-based or human knowledge-based). Using the derived test cases, test execution follows afterwards in which the System Under Test (SUT) is tested (exercised). Test evaluation (using test oracles) is the final phase in which the results of testing are evaluated (pass or fail), and test verdicts are made. Three cross-cutting activities are also shown in Figure 4: test management (planning, control, monitoring, etc.), test automation (could be conducted in any of the phases), and other activities (e.g., regression testing, and test prioritization). Note that many people think of test automation only for automated execution of test cases, but test automation has been successfully implemented in other test activities too [14], e.g., test-case design and test evaluation.

In Figure 4 we also show for each of the test activities the number of papers in our study pool which have proposed contributions it, e.g., 161 papers (out of the 272) have proposed techniques for criteria-based test-case design. As we can see in Figure 4, there is a good mix of papers proposing techniques and tools for each of the test activities. The most frequently addressed test activities in the papers are test execution, automation and criteria-based test-case design (e.g., based on code coverage). Next, we discuss a selected summary of studies classified under each test activity type. Note that, due to space constraints, we do not provide extensive summaries of technical

approaches, methods and tools in this area, but only provide a selected summary of a few studies for each test activity type. Practitioners and (new) researchers can easily use the online Google spreadsheet (goo.gl/MhtbLD) of the study as an "index" to the body of knowledge in this area and to access each of the 272 primary studies.

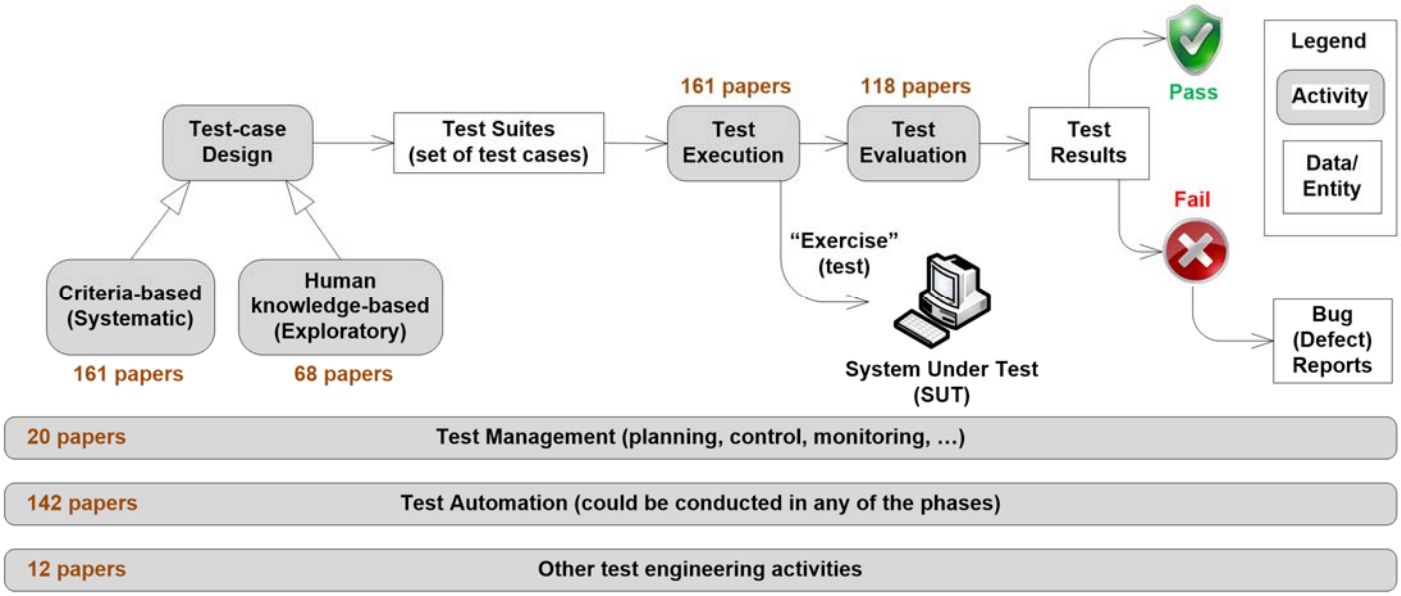

Figure 4- A generic test process showing different types of test activities

Two example papers in the test-case design category are as follows. In [Source 34] '*Adaptation of state/transition-based methods for embedded system testing*', test sequences were generated from Extended Finite State Machine (EFSM). In [Source 57] '*Automated generation of test cases from output domain and critical regions of embedded systems using genetic algorithms*', test cases were automatically generated using genetic algorithms from partitioned input spaces. The approach was applied to the temperature monitoring and controlling system of a nuclear reactor. Many of the papers (overall 161) covered test execution as it is the actual phase of software testing, i.e., test-case design and the other activities in Figure 4 are actually 'supporting' test execution.

Overall 118 papers covered test evaluation. For instance, in '*An Approach for Test Derivation from System Architecture Models applied to Embedded Systems*' [Source 39] the test evaluation algorithm correlates the SUT's architectural information with the functional requirements and checks, in an interesting manner, that a certain side effect cannot occur. In the example SUT of that study [Source 39] (a mirror control system), the test programmatically validated that the SUT does not move the mirror vertically in case the mirror position setting button is pulled to right. In [Source107] '*Effective test driven development for embedded software*', a popular embedded unit test framework (in C) is used and test scripts are written, which include test evaluation (assertions), e.g., `TEST_ASSERT_EQUAL_INT( 7, Model_FeedbackVoltage)`. The case-study SUT in that study is a conductor hardware control unit developed for automated guided vehicles.

In 20 papers, test management and planning were addressed. For instance, in [Source 31] a test process improvement model for embedded software development was proposed. In [Source 212] [15] *'Taming the embedded tiger: agile test techniques for embedded software*', the authors argued that, even with evolving hardware in the picture, Agile methods work well provided that testers use proper test strategies. The study then proposed an experience-based test strategy for that purpose.

As Figure 4 shows, test automation is also very active in the embedded testing community (142 papers). Many automated tools and frameworks have been proposed in this domain. Two example papers in the test automation category are: [Source 214],*'Teaching cross-platform design and testing methods for embedded systems using DICE',* a unit test framework called DICE was used. [Source 155] *'Model-based testing of embedded automotive software using MTest'* proposed a model-based testing toolset called MTest.

Finally, 12 papers covered other test engineering activities. For instance, in [Source 138] *'Integrating test levels for embedded systems',* test-case specifications were reused throughout the test process (the activity was test reuse). In [Source 180] *'Rapid embedded system testing using verification patterns',* a pattern approach to testing embedded systems was proposed which lets developers customize test script templates.

The sum of the numbers in Figure 4 are more than the number of papers (272), since many papers made contributions in more than one test activity, e.g., the paper '*Improving the accuracy of automated GUI testing for embedded systems'* [16] [Source 135 in the online sheet] made contributions under four different test activities: human knowledge-based test-case design, test automation, test execution and test evaluation.

To know about the existing advances and to potentially adopt/customize them for usage in their own testing needs, we advise test practitioners in the embedded software industry to review the list of 272 studies in our online Google spreadsheet (goo.gl/MhtbLD) as categorized by the above six types of test activities. For example, if a test team intends to conduct test-case design, it is advised to review

the 161 papers in the "criteria-based" category or the 68 papers in the "human knowledge-based" test-case design category to see if the existing approaches can be adopted/customized into their needs. This would prevent them from "reinventing the wheel" (developing an already-existing test technique).

6 TYPES OF TEST ARTIFACTS GENERATED

Different test techniques have been proposed to generate different types of test artifacts. Ordered by frequency (number of papers), the largest ratio of papers (131, 48.1%) proposed techniques to derive test-case inputs (values), e.g., the paper '*Applying Model-Based Testing in the Telecommunication Domain*' [17] [Source 52] applied model-based coverage criteria to derive test cases. In that study, UML models of the SUT were transformed to models in a domain-specific language (DSL) called Qtronic Modeling Language (QML) and then a tool called Conformiq Qtronic tool was used to derive test cases. The test-case generation approach was applied to and evaluated on the mobility management feature of a mobile services switching center server. Another work in this category is the paper *'Model-based testing of embedded automotive software using MTest*' [Source 155], a joint work between two major industrial firms in Germany, proposing the model-based testing approach MTest which uses classification trees to model test data and was developed to specify the requirements and then used to derive test sequences. The approach was evaluated on a real automotive system in DaimlerChrysler, i.e., the "lane change" feature of a driving maneuver controller.

In 93 papers, the generation of automated test code (e.g., in xUnit) was addressed. For example, the paper '*A model-based testing framework for automotive embedded systems'* [18] [Source 16] developed an approach to generate test scripts in Python based on a specific form of abstract test-cases. The approach was applied to the ABS function implementation of the Brake-by-Wire system prototype from Volvo and it showed 'encouraging results'.

85 papers presented approaches for generation of test-case requirements (not explicit input values). As discussed above, test requirements are usually not actual test input values, but the conditions (e.g., for control flow paths in code) that can be used to generate test inputs. For example, the paper '*Automatic Test Case Generation and Test Suite Reduction for Closed-Loop Controller Software*' [19] [Source 72] presented an automatic test-case generation and test-suite reduction approach based on path coverage for closed-loop controller software. To evaluate the proposed approach, the study applied the approach to five controller programs that are based on implementations of real-world medical protocols and used at the Hospital of the University of Pennsylvania to determine the amount of insulin to pump.

In 73 papers, expected outputs (test oracle) were discussed or generated. For example, the paper '*A model-based testing framework for automotive embedded systems*' [Source 16] proposed an approach for generating expected outputs using the specifications based on the Architecture Analysis and Design Language (AADL). The approach was applied to an experimental smart curtain control system. The approaches proposed in 9 other papers proposed 'Other' types of test artifacts, e.g., test patterns in [Source 83] and test documentation in [Source 163].

7 MAJOR FOCUS ON MODEL-BASED TESTING

In terms of techniques used to derive test artifacts, requirement-based testing is very popular in this area as it was used in 159 papers (58.4% of the papers in the pool). A large portion of these papers (142 of 159) used model-based testing (52.2% of the pool).

Figure 5 shows the general process of model-based testing, in which models are developed using either a forward engineering or a backward engineering manner. Once models are validated themselves, they can be used for test-case generation (test-case design), e.g., Finite State Machines (FSM) and its extensions are frequently used to derive test-case sequences using coverage criteria such as all-transitions coverage. Given the large wealth of knowledge and industrial evidence in model-based testing of embedded systems (142 papers), we recommend companies, who are planning to implement systematic testing, to review and consider this vast body of knowledge to potentially adopt some of the ideas in model-based testing.

In 46 papers, Finite State Machines (FSM) and its extensions (e.g., timed-FSM) were used for model-based testing, e.g., [Sources 16, 18, 21]. 28 papers used MATLAB Simulink models, e.g., [Sources 28, 29, 54]. 75 papers used 'Other' types of models, e.g., Method Definition Language (MeDeLa) in [Source 170] and UML sequence diagrams in [Source 173].

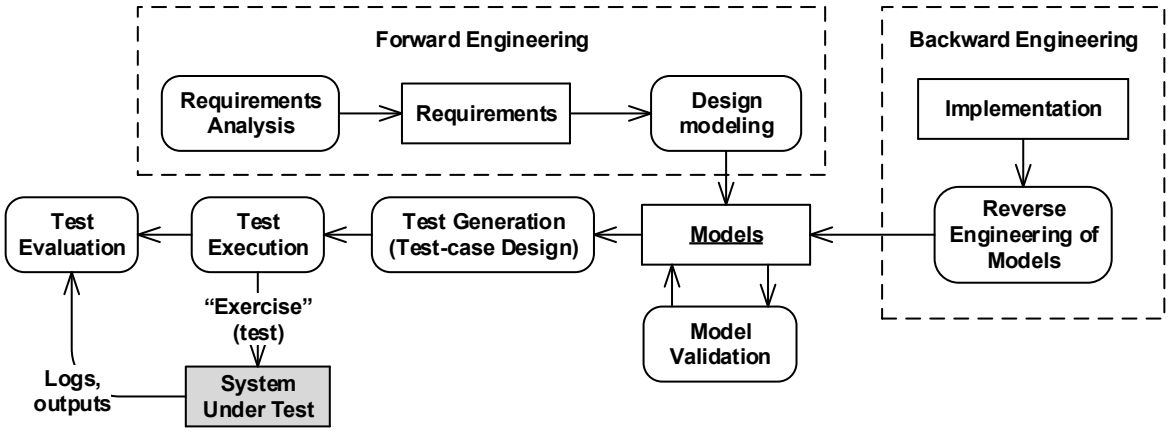

Figure 5- General process of model-based testing

Model-based testing actually fits in the scope of a larger development concept for embedded systems, i.e., X-in-the-loop development, simulation and testing [Sources 160, 161, 194, 208]: which consist of Model-in-the-Loop (MiL), Software-in-the-Loop (SiL), Processor-in-the-Loop (PiL), Hardware-in-the-Loop (HiL), and System-in-the-Loop (SYSiL). X-in-the-loop testing has gained acceptance due to the increased adoption of model-based development in industry, especially in the automotive domain [20].

8 Different Industries

While some test techniques can be applied to all types of embedded software, some techniques are domain-specific. According to the categorization made earlier in the paper, Figure 6 depicts the breakdown of the different industries. Research in the automotive sector is the most active in this field (in 81 papers). Industrial automation and control domain is the second (33 papers). Aviation, avionics and space are the third. Among the 'Other' domains are: banking, public transport and e-government in [Source 43], and fire-safety systems in [Source 219].

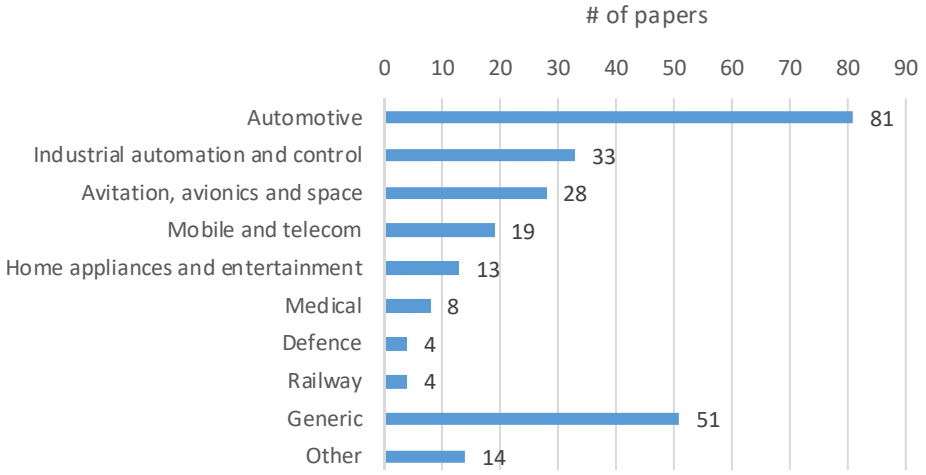

Figure 6- Breakdown of the different industries (domains) represented in the papers

9 Benefits of This Review

The authors have already started to benefit from the results of this review. In our ongoing collaborations with several industry partners in Turkey and Austria in the area of testing embedded software, our colleagues and we were quite uncertain of whether certain techniques already exist or we shall develop new testing techniques ourselves to solve the existing challenges/gaps. In this context, thanks to our review study, we are currently assessing several existing model-based techniques selected from the literature based on the review at hand for possible adoption/extension in our ongoing industry-industry collaborative projects.

Also, as per our observations, unfortunately, many companies do not have an adequate overview of state-of-the–art and -practice in the area of testing embedded software, which is driven by model-based approaches often developed in an academic or industrial research context. As a consequence, companies often reinvent the wheel in this area by designing a test approach new to them, but existing in the literature. The main reason for this is, that companies do not have an adequate overview of state of the practice in this area. This review paper and its supplementary material (the online repository and classification of all the 272 papers) close this gap and are intended to support the large world-wide community of software engineers and testers in the embedded software industry.

To further assess the benefits of this review, we asked several active test engineers in the Turkish embedded software industry to review this short paper and the online spreadsheet of papers, and let us know what they think about the potential benefits of this review paper. Their general opinion was that a review paper like this article is an invaluable resource and can actually serve as an "index" to the body of knowledge in this area. Here is what one of the practitioners explicitly mentioned: "*Our company conducts embedded system testing for software-intensive systems in the military domain. In such a context, one of our major problems is to borrow the actual Systems Under Test (SUT) from our customers because of security concerns. After reviewing this study, I wonder about "why we do not have any model of these SUTs". If we can create models of these SUTs (e.g., model-in-the loop, MIL), in the future we would not need to borrow the real systems from our customers and models may be sufficient for our test-case design and other test activities. There are a lot of studies in the pool of this review study, which would benefit us. I think this idea is a major benefit for companies like ours. And I hope we may realize this idea [in near future]. And as you have said in the paper, I believe that many companies are exactly reinventing wheel.*"

10 SUMMARY OF FINDINGS AND ROAD AHEAD

By reviewing the state-of-the-art and the –practice, this paper summarized the types of testing topics studied, types of testing activity, types of test artifacts generated, and the types of industries in which studies have focused, in embedded software testing based on a systematic literature review including 272 papers. Testing embedded software is a growing field of research where not only academia but also many companies especially from the automotive industry are active. By summarizing what we know in this active area, this paper can serve as an "index" to the vast body of knowledge in this important area. Practitioners and readers who are interested in each of the topic areas, discussed in this paper, can conveniently use the online Google spreadsheet (goo.gl/MhtbLD) to navigate to each of the papers to study that topic in more depth.

We recommend companies, who plan respective improvement measures, to take the collected body of knowledge from this SLR into account. Although there has been a high interest and progress in embedded software testing, more than half of the available papers only propose solution or report experiences, but do not provide empirical evidence on the effectiveness and efficiency of specific embedded software testing approaches. Therefore in the future, there is a need for more rigorous empirical studies providing industrial evidence on the effectiveness (e.g., measured in terms of defect detection rate) and efficiency of embedded software testing approaches in specific contexts to further improve decision support on the selection of testing approaches beyond this SLR.

Also given that the existing literature is extensive already (i.e., 270+ papers), the goal of this paper was to survey and present what we know in this area. As a follow-up work, we invite researchers to synthesize the major "open" technical issues and needed solutions in this area. We also invite practitioners to utilize and benefit from the vast body of knowledge in the available large number of studies of this area in their industrial projects and to assess the effectiveness of approaches, tools and techniques, to collaborate with researchers on the open issues, and to then report back in the form of papers about their experiences and lessons learned in using the vast body of knowledge in the area of testing embedded software.

Last but not the least, we are aware that new research and new results are and will regularly be published in this area. Thus, it will be important to maintain / update the pool of studies on a regular basis. Since our dataset and pool are open access and publicly available, we invite other researchers and practitioners to maintain / update the pool after a while when there is need to re-conduct a review study on this subject (perhaps after a few years).

REFERENCES

[1] S. Schulz, K. J. Buchenrieder, and J. W. Rozenblit, "Multilevel testing for design verification of embedded systems," *Design & Test of Computers, IEEE,* vol. 19, pp. 60-69, 2002.

[2] C. Ebert and C. Jones, "Embedded Software: Facts, Figures, and Future," *IEEE Computer,* vol. 42, pp. 42-52, 2009.

[3] A. Tang and R. Kazman, "On the Worthiness of Software Engineering Research," in *http://shidler.hawaii.edu/sites/shidler.hawaii.edu/files/users/kazman/se_research_worthiness.pdf*, Last accessed: June 2016.

[4] D. Lo, N. Nagappan, and T. Zimmermann, "How practitioners perceive the relevance of software engineering research," presented at the Proceedings of the 2015 10th Joint Meeting on Foundations of Software Engineering, Bergamo, Italy, 2015.

[5] A. Banerjee, S. Chattopadhyay, and A. Roychoudhury, "On Testing Embedded Software," *Advances in Computers,* vol. 101, pp. 121–153, 2016.

[6] M. A. J. Burford and F. Belli, "CADAS: A Tool for Designing Reliable Embedded Software and Supporting Testing "in the Large"," in *Fehlertolerierende Rechensysteme*. vol. 84, K. E. Großpietsch and M. Dal Cin, Eds., ed: Springer Berlin Heidelberg, 1984, pp. 101-112.

[7] T. Dyba and T. Dingsoyr, "What Do We Know about Agile Software Development?," *Software, IEEE,* vol. 26, pp. 6-9, 2009.

[8] T. Hall, H. Sharp, S. Beecham, N. Baddoo, and H. Robinson, "What Do We Know about Developer Motivation?," *Software, IEEE,* vol. 25, pp. 92-94, 2008.

[9] K. Petersen, S. Vakkalanka, and L. Kuzniarz, "Guidelines for conducting systematic mapping studies in software engineering: An update," *Information and Software Technology,* vol. 64, pp. 1-18, 2015.

[10] B. Kitchenham and S. Charters, "Guidelines for Performing Systematic Literature Reviews in Software engineering," 2007.

[11] P. Ammann and J. Offutt, *Introduction to Software Testing*: Cambridge University Press, 2008.

[12] V. Garousi, Y. Amannejad, and A. Betin-Can, "Software Test-Code Engineering: A Systematic Mapping," *Journal of Information and Software Technology*, vol. 58, pp. 123–147, 2015.

[13] S. Doğan, A. Betin-Can, and V. Garousi, "Web Application Testing: A Systematic Literature Review," *Journal of Systems and Software*, vol. 91, pp. 174-201, 2014.

[14] V. Garousi and F. Elberzhager, "Test automation: not just for test execution," *In Press, IEEE Software*, 2016.

[15] N. V. Schooenderwoert and R. Morsicato, "Taming the embedded tiger: agile test techniques for embedded software," in *Agile Development Conference*, 2004, pp. 120-126.

[16] Y. D. Lin, E. T. H. Chu, S. C. Yu, and Y. C. Lai, "Improving the Accuracy of Automated GUI Testing for Embedded Systems," *IEEE Software*, vol. 31, pp. 39-45, 2014.

[17] A. Fredrik, A. Veli-Matti, K. Jani, T. Risto, and T. Dragos, "Applying Model-Based Testing in the Telecommunication Domain," in *Model-Based Testing for Embedded Systems*, ed: CRC Press, 2011, pp. 486-524.

[18] R. Marinescu, M. Saadatmand, A. Bucaioni, C. Seceleanu, and P. Pettersson, "A Model-Based Testing Framework for Automotive Embedded Systems," in *2014 40th EUROMICRO Conference on Software Engineering and Advanced Applications*, 2014, pp. 38-47.

[19] C. Murphy, Z. Zoomkawalla, and K. Narita, "Automatic Test Case Generation and Test Suite Reduction for Closed-Loop Controller Software," in *Technical Report, University of Pennsylvania*, 2013.

[20] J. Day, "Virtual Hardware In the Loop (vHIL): Earlier and Better Testing for Automotive Applications," in *http://johndayautomotivelectronics.com/virtual-hardware-in-the-loop-earlier-and-better-testing-for-automotive-applications/*, Last accessed: June 2016.

AUTHOR BIOGRAPHIES

Vahid Garousi is an Associate Professor of Software Engineering in Hacettepe University in Ankara, Turkey and a senior consultant at Maral Software Consulting Corporation. Prior to that, he worked as an Associate Professor at the University of Calgary, Canada from 2006 until 2014. His research interests include software test engineering, software quality, empirical software engineering and improving industry-academia collaborations in software engineering. He was selected a Distinguished Visitor (speaker) for the IEEE Computer Society's Distinguished Visitors Program (DVP) for the period of 2012-2015. Contact him at vahid.garousi@hacettepe.edu.tr.

Michael Felderer is a senior researcher at the University of Innsbruck, Austria, and a senior consultant at QE LaB Business Services GmbH. He holds a PhD and habilitation degree in computer science. His research interests include software and security testing, software processes, requirements engineering, empirical software engineering, as well as improving industry-academia collaboration. Besides his research activities, he transfers his research results into practice as a consultant and speaker on industrial conferences. Contact him at michael.felderer@uibk.ac.at.

Çağrı Murat Karapıçak is a software engineer in KUASOFT A.Ş., Ankara, Turkey. He is also a PhD student in Middle East Technical University (METU). Contact him at cmkarapicak@kuasoft.com.

Uğur Yılmaz is a test engineer in ASELSAN A.Ş., Ankara, Turkey. He is also a MSc student in Hacettepe University. Contact him at uguryilmaz@aselsan.com.tr.