Test Process Improvement Through Test Maturity Models, Test Process Quality, Embedded Systems and DevSecOps Movement: A Tertiary Study

Sara Gračić
Faculty of Economics
University of Novi Sad
Subotica, Serbia
saritta4u@gmail.com

Abstract — **The quality of developed software is especially influenced by the quality of test processes. The purpose of this study was to give answers to research questions regarding quality, topics and trends in the field of test process improvement. To determine the-state-of-the-art achievements in the above mentioned field, it was necessary to summarize the existing body of knowledge by conducting a tertiary study. 5 scientific databases were chosen; search string, inclusion, exclusion and quality criteria were defined and applied and the search resulted in 9 final papers. All sources reported the phenomenon of test process improvement from different perspectives: test maturity models, quality of testing process (test breakages, test smells and software quality management), embedded systems and DevSecOps movement.**

Keywords-test process improvement; test maturity models; SLR; quality; embedded software

I. INTRODUCTION

The quality of developed software is strongly influenced by the quality of the development process and testing process, especially, contributes to software quality. However, test process requires significant effort [1].

During development, software evolves, so breakage of certain developed test cases is inevitable. As a consequence, discarding broken test cases is necessary, which causes "significant waste of effort and leads to test suites that are less effective and have lower coverage". To eliminate wasted effort, test repair approaches have been developed to evolve test suites parallel with applications by repairing broken test cases [2].

Causes of wasted effort are also "test smells"; they are considered as anti-patterns and poorly designed tests. Because their presence can have negative effects on the quality of test suites and production code, test smells are actively being discussed among practitioners and researchers. As a result of discussions, various guidelines for dealing with test smells are being offered to ensure their prevention, detection and correction [3].

Including security in DevOps has been challenging, because traditional security methods were no match to DevOps' agility and speed. Therefore, development and integration of modern security methods that can keep up with DevOps was the main goal of DevSecOps movement. DevSecOps is an expansion of DevOps with integrated security controls and processes into software development by promoting collaboration among security, development and operations teams [4].

"With the proliferation of embedded ubiquitous systems in all aspects of human life" [5] such as automotive, transportation, medical-equipment, communication, energy and other system types [6], challenges are constantly set when developing embedded systems (e.g. quality, time to market) [5] and that has also triggered many innovations [6].

Embedded software requires effective and efficient testing, for which many techniques, approaches, tools and frameworks have been proposed by practitioners and researchers [6].

On the other hand, it has been stated that implementation of many software processes in Embedded Systems Development (ESD) has had both advantages and disadvantages, so presenting the-state-of-the-practice could be beneficial for academia and industry [5].

After the Introduction, the rest of the paper is structured in the following order: Section II presents research methodology (research questions, search strings, inclusion and exclusion criteria, chosen scientific databases and how the review was conducted); Section III extracts the review data; Section IV discusses findings, while conclusion is given in Section V.

II. RESEARCH METHODOLOGY

Systematic literature review is becoming a vital part of scientific research [7]. It is a mean of identification, evaluation and interpretation of available research relevant to [8] a specific problem, question, phenomenon or area of interest. Literature review has to be thorough and fair, otherwise it will be of little scientific value [8]. Although systematic reviews require significant effort, their main advantage is providing detailed information regarding a specific phenomenon in focus of a research. A systematic review includes several discrete activities, divided into three main phases: planning a review, conducting a review and reporting review findings [8].

A. Review Planing

The main objective of this paper is to summarize the existing systematic literature reviews regarding software test process improvement and determine current trends in the previously mentioned field and the quality of published studies.

1) Defining Research Questions

Based on the set objective, following research questions were constructed. **RQ1**: *How many literature review studies regarding test process improvement were published until December 30th 2019?* **RQ2**: *What is the quality of the papers published?* **RQ3**: *What were the main topics discussed in these papers?* **RQ4**: *What are current trends in the field of test process improvement?*

2) Defining Search Strings for Scientific Databases

To provide answers to research questions, a tertiary study was conducted to identify trends and quality of the published studies. Five electronic databases were chosen for conducting the research: IEEE xplore, AIS library, Science Direct, Springer Link and ACM Digital library. Research string was set in the following way: "test process improvement" AND systematic literature review. It was applied in all of the above stated electronic databases.

3) Defining Inclusion and Exclusion Criteria

There were no limitations regarding publishing period (all studies available till December 30th 2019 were taken into consideration). All types of publications were included in the initial population - journals, conferences and books. Both published and approved papers were taken into consideration. Grey literature (e.g. blog posts, white papers) was not included in the initial population. Only systematic literature review studies in the domain of test process improvement were examined. Studies that were not systematic literature review in the field of test process improvement were excluded. Language was not set as a barrier for entering the initial population, although it could be for entering in the next level (if title or abstract are not in English).

4) Defining Quality Criteria

To evaluate the quality of studies that will enter in the review population, they firstly have to satisfy inclusion and exclusion criteria. Secondly, chosen scientific databases should provide papers that were published or are in press with respectable publishers e.g. Springer.

B. Conducting the Review

Applying the search string in all 5 databases resulted in building an initial population of total of 87 hits that are either published or approved for publishing. Hits were structured in the following order: IEEE xplore (1), AIS library (1), Science Direct (19), Springer Link (38) and ACM Digital Library (28).

To ensure that a study is a systematic literature review in the field of test process improvement and therefore in function of answering research questions, titles, keywords and abstracts for all 87 papers were evaluated.

It was determined that 86 articles were published in English completely; 1 had abstract in Portuguese and English, while the rest of the paper was in Portuguese. Since the paper published in Portuguese was not a systematic literature review of test process improvement, its' exclusion represents no threat to validity for this tertiary review.

Research paper from IEEE xplore was included in the second population. Research paper from AIS library was discarded, because it was not in the relevant field (architecture). Out of 19 research papers from Science Direct, only 5 were included. 3 out of 38 research papers from Springer Link were included. 1 of the 28 hits in ACM Digital Library met the above set criteria and was included in the second population.

After "quick reading" of second population, it was determined that Science Direct had more detailed version of the paper that was found in IEEE xplore. Therefore, final population for this tertiary study includes 9 papers.

III. Extracting Review Resuts

Data collected from 9 studies are summarized in the following categories: maturity of test processes, test process quality, embedded systems and DevOps.

A. Maturity of test processes

Authors in [9], [10] conducted a systematic literature review of the scientific literature (e.g. journals, conference papers) and practitioners' gray literature (e.g. blog posts, white papers) called multivocal literature review (MLR) regarding test maturity models. Their search resulted in 181 sources, of which 51 sources (29%) were grey literature, while 130 sources (71%) were formally published literature. Summary of MLR resulted in identification of 58 test maturity models and a lot of sources with varying degree of empirical evidence. The most popular test maturity models were Test Maturity Model Integration (TMMi), its predecessor, Testing Maturity Model (TMM), Test Process Improvement (TPI) and its successor, TPI Next. It is also evident that out of 58 models, no one model fits all test process improvement needs, possibly because some models from academics are not based on industrial needs, but "hypothetically argued motivations" or because some researchers do not fully take into account best practices from academia and industry. With so many available models, choosing the most appropriate is becoming a challenge. These authors summarized drivers, challenges and benefits in the field of test process improvement. **Drivers** were classified into several categories: process and operational drivers (present in 46 sources), software quality drivers (25 sources), cost-related drivers (23 sources), schedule-related drivers (12 sources) and other drivers (15 sources). Improving test processes poses **challenges**, which were classified into several categories: lack of (required) resources (17 sources), resistance to change (12 sources), improvement was seen as additional effort (9 sources), lack of competencies (7 sources), unclear scope and focus (7 sources), no owner of the improvement process (5 sources), no clear benefits from improvement activities (4 sources), while 23 sources mentioned other challenges. Although some participants in various studies do not see clear **benefits**, authors in [9] classify them into following categories: operational benefits (present in 48 sources), technical benefits (37 sources) and

business benefits (27 sources). Although TMA and TPI have evolved and have gained importance, there is still need for empirical studies to provide evidence for TMA and TPI in specific context – e.g. "by taking into account the domains of the systems under test".

Authors in [11] conducted systematic literature review on software test process improvement (STPI) approaches. These approaches are frameworks for providing guidelines for improving software test processes in software development organizations. Authors identified 18 STPI approaches and investigated their characteristics: completeness of development, availability of information, assessment instruments and domain limitations. They selected TPI NEXT and TMMi and these were "evaluated with respect to their content and assessment results in industry". Content comparison of TPI NEXT and TMMi was done. The result of their systematic literature review has showed two important problems with many STPI approaches: insufficient information and lack of assessment instruments. Approximately 61% of STPI approaches have been developed as concepts. This means that practical implementation of these approaches in industry is quite difficult. Some approaches have been developed for specific domains e.g. Emb-TPI, while only a few have included case studies, experiments or surveys as a form of validation. STPI approaches were divided into four groups. The first group included TMM-based approaches; the second included TPI-based approaches; the third included approaches based on standards, while the fourth included approaches without a testing model. Comparison between TPI NEXT and TMMi showed that the number of *similarities* was high, while there were only a few differences between these two approaches. After examination, the authors concluded that, although numerous STPI approaches were available, many of them were not applicable in industry. Another classification divided approaches based on their *model representation*: approaches without a model, with continuous model and with staged model representation. They emphasized that model representation of available STPI approaches was a major difference that caused variations in the assessment results, despite their "strong similarities". STPI approaches were also divided into *qualitative* and *quantitative*, but only one approach used qualitative data for assessment. *Maturity mapping* conducted in their case study showed that some aspects of level 2 maturity of TMMi are spread across all three maturity levels of TPI NEXT. This means that if an organization achieves the highest level of TPI NEXT, it will not surpass level 2 of TMMi. Authors concluded that for successful implementation of STPI approaches, "extended knowledge in software testing is essential".

For the purpose of identifying developed, extended and adapted test process models from 1990 till 2014, authors in [1] conducted a systematic literature review. Their review resulted in identifying 23 test process models, of which many represent adaptation or extension of TMMi and TPI, "which have different architecture and the new ISO/IEC 29119 with an architectural approach aligned to other ISO/IEC software process models". TMMi and TPI are often used in research and are considered as "mainstream" models; other models e.g. TOM, MMAST have low adoption rate. TPI and TMMi are maturity models, while ISO/IEC-29119 in conjunction with ISO/IEC 33063 is a capability model. CMMI was the base for TMMi; TPI uses a test maturity matrix, while ISO/IEC 29119 architecture is defined by Process Reference Models (PRMs) and Process Assessment Models (PAMs). 13 examined models are general, while 9 are related to specific domains. 17% of investigated models are based on TMM, 19% on TPI, "followed by a 12% of practical experience models". These authors also analyzed minimal model information and determined that TOM, MMAST, TAP and TCMM were process models mentioned in one source; no information was available in scientific databases, minimal information was obtained from non-scientific sources.

B. Test process quality

In [12], authors conducted a literature review of software quality management (SQM), which resulted in 92 papers. Among their findings was shown that "SPI in the context of SQM is equally focused on software testing as well as on complementing (support) activities". A trend of utilization of individual testing approaches is preferred compared to implementing/following standards. Test activity, non-functional testing and level of testing were well covered in literature. On the other hand, test maturity models were not very well covered. Little information was provided, which suggests reluctance towards standardization of testing and lack of clarity of test process. 3 out of 8 studied projects followed a defined process; a basic testing strategy was defined, but not implemented by most of teams. Working in small agile teams and implementing agile practices were considered as strengths. It was reported that "these findings also depend on project/team size, i.e. teams of different size might go for different solution e.g. comprehensive test case management tools are considered more valuable for larger teams". Lack of "unified testing process, unawareness of the process, different process-related constrains represent process issues problematic for teams of any size". According to them, testing was not that presented in the study data, because of specialized (grey) literature on test process improvement (TPI), which is not adequately linked to SPI. They did not find detailed data of the impact of switching to an alternative test approach, but found that quality improving can be achieved by reducing the number of defects. They reported a lack of standardized testing approaches and putting more effort in improving support activities, with a doubt whether a "broader" perspective SPI is more beneficial then optimizing a "technical" test method.

In [2], authors stated that because plenty of studies are being published regarding the topic of test repair approaches, it was important to summarize and consolidate the existing knowledge in order to provide researchers and practitioners a set of guidelines in the field of test repair approaches. The authors followed a standard protocol for conducting systematic literature review of prevention and repair of test breakages. Research goals and corresponding questions were formulated using the Goal Question Metric (GQM) and metrics were extracted from 41 included papers that satisfied set criteria. There were 5 papers published in journals and 36 conference papers. They presented taxonomy of test case

breakages consisted of 3 groups: code level test breakages, web test breakages and GUI-related changes. **Code level test breakages** include class-level changes, method-level changes and attribute-level changes. **Web test breakages** include: locator-based breakages, value/action related changes, JavaScript popup boxes, page reloading and session related changes. **GUI-related changes** include: event-related changes and structural changes. They also proposed a set of test repair tools, with types of modifications necessary to repair scripts, domain and availability and observed that only four of them were publicly available, thus lowering their adoption by industry practitioners and that "most studies evaluated their approaches on open-source case studies". They concluded that evaluating approaches on large scale open source studies is present, but there is also absence of study results performed in real industrial context.

Test smells are an important topic in vast grey, as well as scientific literature and that abundance is not "practical for practitioners and researchers to locate and synthesize", so authors in [3] conducted a multivocal literature mapping (classification) that includes scientific and grey literature. With 166 sources, 120 from industry and 46 from academia, their review presents a "catalogue of test smells", summarizing guidelines/techniques and tools to deal with smells and an "index" to the body of knowledge regarding test smells, with the purpose of providing help in developing high-quality test scripts and minimizing negative consequences in large test automation projects. Various sources dealt with topic like: smells, tools for dealing with smells, models, metrics and processes for smell handling and empirical studies were conducted. 109 sources contributed with guidelines/technics for dealing with smells, while 8 proposed metrics. Authors found a vast number of test smell types and, although many are discussed in the practitioners' community, not many were empirically assessed. Based on their review, approaches for dealing with smells are classified as: prevention, detection, corrections, issues that generate smells and discussion regarding smells. Although test smells are not necessarily harmful, they generate undesirable negative consequences and only 38 papers directly discussed negative consequences of test smells (e.g. tests are fragile, cause-and-effect relationship is compromised, unnecessary work is done, test runs are slower, dead fields of class or its super class, lower stability, maintenance etc.). Out of 81 sources that pointed out new smells, only 8 were from academia. Dependency of test smells and frameworks/tools was also discussed and it was observed that some smells were generic and were in focus of 54 sources; other smells were framework specific – JUnit specific smells were discussed in 69 sources. Manual prevention, detection and correction of smells is challenging, especially for large test suits, so tools for handling test smells are "more than welcome". 24 studies presented these tools, of which only 12 are available. Test smell discussions started around 2001 and are still an important topic in software testing.

C. Embedded systems

Authors in [6] stated that embedded software requires effective and efficient testing, but that many companies invent the invented by designing a new test approach that already exists in the domain, because they do not have an overview of the existing knowledge. Therefore, the authors conducted a systematic literature review (SLR) in the form of a systematic literature mapping (SLM). Their initial pool had 588 papers. After applying inclusion and exclusion criteria, the final pool included 312 technical papers. The review pointed out that most of the primary studies are focused on *system testing*, while the rest are focused on *unit* and *integration testing*. **Types of test activity** include test-case design, test execution, test evaluation, test management, test automation and other activities (e.g. regression testing). **Generated test artifacts** include: test case inputs (values), test case requirements (conditions), automated test code and expected output. **Techniques for deriving test artifacts** included: requirement-based testing, white-box testing, risk/fault-based testing, search-based testing, random testing and others. **Non-functional tests** (e.g. performance tests, load (stress) tests, real-time and reliability tests) were also in focus of some primary studies. 106 studies used existing, while 98 proposed new **test tools**. **Types of industries** cover automotive sector as the most active (96 papers, 30.8%), than industrial automation and control (40 papers, 12.8%), aviation, avionics and space industries (34 papers), mobile and telecommunication (22), appliances and entertainment (18), medical and transport (each 8), defense (5), other e.g. banking, public transport, e-government and fire-safety systems (16). The most popular testing **topics** cover model-based and automated/automatic (testing), (test-case) generation and control systems. They assessed the benefits of their review through consultation with one active test engineer in the Turkish embedded software industry. One benefit was seen in choosing the right test techniques/approaches for embedded software testing challenges. Other is seen as presentation of the latest trends in this area necessary for identifying topics which need further research. Companies that develop embedded software have an "index" of the-state-of-the-art achievements and can avoid "reinventing the wheel" as authors in [6] stated.

Authors in [5] conducted a systematic literature review of the embedded systems with the purpose of identifying common challenges and "how software processes and practices address them" as well as to investigate opportunities for improvement from academic and industrial perspective by analyzing sources from 2000 to 2013. Quasi-Gold Standard (QGS) was applied: 5 publication venues and 4 search engines were chosen; search query, inclusion and exclusion criteria were defined and the review resulted in 45 individual studies. Four groups of challenges were identified: **development factors** (e.g. lack of suitable tools, methodologies), **human factors** (e.g. lack of interaction among software and hardware developers), **external factors** (e.g. marketing requirements) and **internal factors** (e.g. hardware dependence, resource constraints, non-defect policy, reliability). Authors noticed that suitable processes for tackling challenges in ESD are frequently missing in the examined studies. "One possible reason for this may be the many-to-many mapping between the challenging factors and the process elements and complex context of ESD as well lead to difficulties to isolate cause-and-effect relationships between them." Solutions for tackling challenging factors seem to be diverse and were grouped into

two categories: **technical solutions** (improved testing process, model-driven engineering, object-oriented methodology, component-based engineering, design patterns, simulations, architectural framework) and **managerial solutions** (strengthened V-model (and its variants), agile methods, improved lifecycle management with more ESD-specific steps).

D. DevOps

Authors in [4] conducted a multivocal literature review in the first quarter of 2017 that resulted in 52 sources: 2 from Google Scholar and 50 from Google search engine. DevSecOps is based on several principles: culture, automation, measurement, and sharing, but with adding of adding security from the start. It is applied in: threat modeling and risk assessments, continuous testing, monitoring and logging, security as code and red-team and security drills. Reported benefits include: shifting security to the left, automating security and value; challenges organizations face are related to keeping up with DevOps, organizational and tools and practices. The review also point out that the number of sources regarding DevOps is increasing. Authors conclude that "implementing security that can keep up with DevOps is a challenge, but it can gain great benefits if done correctly".

IV. DISCUSSION

To answer **RQ1**: *How many literature review studies regarding test process improvement were published until December 30th 2019?*, research string - "test process improvement" AND systematic literature review - was applied in five electronic databases (IEEE xplore, AIS library, Science Direct, Springer Link and ACM Digital library). After applying inclusion and exclusion criteria, "quick reading" of title, key words and abstract and eliminating one duplicate paper, 9 literature review studies were included into final review population and results were extracted.

Answer to **RQ2**: *What is the quality of the papers published?* is that 1 paper was published by IEEE society; detailed version of that paper was published by Elsevier, together with 4 more papers; 3 papers were published by Springer, while 1 paper was published by ACM. Also, all final review papers fully satisfied inclusion and exclusion criteria.

Answers to **RQ3**: *What were the main topics discussed in these papers?* and **RQ4**: *What are current trends in the field of test process improvement?* are given in the following paragraphs.

Sources [1], [9], [10] and [11] examined *test maturity models*. Multivocal literature review (MLR) in [9] and [10] resulted in identification of 58 test maturity models with different levels of empirical evidence. The review in [1] identified 23 test process models, of which many are an adaptation or extension of TMMi and TPI maturity models, which are considered as "mainstream" models. This is also confirmed by [9] and [10], which report that the most popular models are TMMi, its predecessor, TMM, TPI and its successor, TPI Next, which is also confirmed with the fact that authors in [11] selected and compared TPI NEXT and TMMi in their review, which showed high number of similarities and only a few differences. However, their maturity mapping showed that if an organization achieves the highest level of TPI NEXT, it will not surpass level 2 of TMMi. In [1] was stated that 17% of the investigated models are based on TMM, 19% on TPI, "followed by a 12% of practical experience models". Other models e.g. TOM and MMAST have low adoption rate. 13 examined models are general, 9 are domain specific. This division is also present in [9] and [10], were it was concluded that no one model fits all test process improvement needs, from various reasons e.g. not taking into account real industrial needs or because best practices from academia and industry are not taken into account by some researchers. 58 maturity models also mean that finding the most suitable model is becoming quite challenging. These authors summarized drivers, challenges and benefits in the field of test process improvement, while [1] analyzed minimal model information and reported that TOM, MMAST, TAP and TCMM were process models mentioned in one source; no information was found in scientific databases, minimal was obtained from non-scientific sources. Authors in [11] identified and investigated characteristics (completeness of development, availability of information, assessment instruments and domain limitations) for 18 STPI approaches of which many are not applicable in industry because of insufficient information, lack of assessment instruments, conceptualization of 61% of STPI approaches and domain specificity of some approaches. For now, only a small number of approaches are validated empirically. Classification based on model representation showed that model representation of STPI approaches was a major difference that caused variations in the assessment results, despite their "strong similarities". STPI approaches were also divided into qualitative and quantitative, but only one approach used qualitative data for assessment. Authors concluded that for successful implementation of STPI approaches, "extended knowledge in software testing is essential".

Sources [2], [3] and [12] examined the *quality of testing process* from different perspectives. Authors in [2] conducted a SLR from perspective of ***test case breakages*** and presented a detailed taxonomy consisted of 3 breakage groups: code level test breakages, web test breakages and GUI-related changes. They also proposed a set of test repair tools, with an important observation that only four of them were publicly available. This means that their adoption rate by practitioners will be lower and that there is generally absence of study results performed in real industrial surroundings.

In [3] quality of test process is observed from aspect of ***test smells.*** Multivocal literature mapping showed that only 27.71% sources covering this topic were academic, while 72.29% were grey sources. Most of the review sources (109) provided guidelines/technics for dealing with smells; only 8 proposed smell metrics. It was noted that, although many smells do exist and are an important topic in the practitioners' community, just a few are empirically assessed. Smell actions include prevention, detection, corrections, issues that generate smells and discussion regarding smells. Despite negative consequences (e.g. tests are fragile, cause-and-effect relationship is compromised), only 38 papers directly addressed them. Detection of new smells is by far more present in industry that in academia, since only 8 sources, out of 81,

were from academia. Some smells are generic, others are framework based. Although tools for handling smells do exist, only half (12) are publicly available.

In [12] testing is discussed from ***software quality management*** perspective. It is reported that utilizing individual testing approaches is preferred as opposed to implementing/following standards and that test maturity models were not very well covered, which is partially true, because TMMi is well covered in [9], [10], [11] and [1]. However, only [1] reports certain models with scares non-academic and no academic information at all. Unwillingness for accepting test standardization and unclear perception of test processes resulted in not implementing test strategy, although it does exist. The authors claimed that TPI was not that present in the study data, because of vast grey literature.

Sources [5] and [6] examined test process through ***embedded systems***. In [6] is emphasized that embedded software requires effective and efficient testing, but that many companies "reinvent the wheel". SLR pointed out that most of the primary studies are focused on system testing, while the rest are focused on unit and integration testing. It also summarizes: types of test activity, generated test artifacts, techniques for deriving test artifacts, non-functional tests, test tools (98 sources suggested new ones), types of industries (where the most popular is automotive sector and embedded systems have application in banking as well) and testing topics. Benefits of their SLR include choosing the right test techniques/approaches for embedded software testing challenges, presentation of the latest trends, the-state-of-the-art achievements and elimination of repeating what has already been done.

Authors in [5] identified challenges regarding human, external, internal and development factors and that suitable processes for handling those challenges in ESD are frequently missing, because of complexity of ESD and difficulties in isolating cause-and-effect. They summarized solutions into technical (e.g. improved testing process) and managerial.

Source [4] presents testing from the perspective of ***DevSecOps movement***, where security is present from the start. It is applied in: threat modeling and risk assessments, continuous testing, monitoring and logging, security as code and red-team and security drills, while benefits include: shifting security to the left, automating security and value. Although challenging, proper implementation of security that can keep up with DevOps is rewarding.

V. CONCLUSION

The quality of developed software is especially influenced by the quality of test processes, so their improvement is of at-most-importance. To determine the-state-of-the-art achievements in test process improvement, it was necessary to summarize existing body of knowledge by conducting a tertiary study.

5 scientific databases were chosen, search string, inclusion, exclusion and quality criteria were applied and the search resulted in 9 final papers. All of the papers reported the phenomenon of test process improvement from various perspectives, thus giving answers to research questions regarding topics and trends.

Sources [1], [9], [10] and [11] examined *test maturity models* stating that more must be done to overcome huge gap between industry and academia, since a lot of maturity models that do exist are generally not applicable in practice. Sources [2], [3] and [12] examined the *quality of testing process* from test breakages, test smells and software quality management. Various tools should become publicly available, academia should pay more attention to discovering test smells and more should be done in the field of acceptance of standardized test processes. Sources [5] and [6] examined test process through *embedded systems*, emphasizing that testing should be more effective and efficient, that companies should turn to academia to eliminate reinventing the existing and more attention should be paid to identified challenges. Source [4] presented testing from aspect of DevSecOps movement, clearly stating that companies should be more focused on proper implementation of security, because of potential benefits.

REFERENCES

[1] C. Garcia, A. Dávila, and M. Pessoa, "Test Process Models: Systematic Literature Review," SPICE 2014, CCIS 477, pp. 84–93, 2014

[2] J. Imtiaz, S. Sherin, M. U. Khan, and M. Z. Iqbal, "A systematic literature review of test breakage prevention and repair techniques," Information and Software Technology, vol. 113, pp.1-19, 2019

[3] V. Garousi, and B. Küçük, "Smells in software test code: A survey of knowledge in industry and academia," The Journal of Systems and Software, vol. 138, pp.52-81, 2018

[4] H. Myrbakken, and R. Colomo-Palacios, "DevSecOps: A Multivocal Literature Review," SPICE 2017, CCIS 770, pp. 17–29, 2017

[5] G. Rong, T. Liu, M. Xie, J. Chen, C. Ma, and D. Shao, "Processes for Embedded Systems Development: Preliminary Results from a Systematic Review," ICSSP 2014, pp. 94–98, 2014

[6] V. Garousi , M. Felderer , C. M. Karapıcak, and U. Yılmaz, "Testing embedded software: a survey of the literature," Information and Software Technology, 2018

[7] A. A. Khan, J. Keung, M. Niazi, S. Hussain, and H. Zhang, "Systematic Literature Reviews of Software Process Improvement: A Tertiary Study," EuroSPI 2017, CCIS 748, pp. 177–190, 2017

[8] B. Kitchenham, "Procedures for Performing Systematic Reviews," Joint Technical Report, Computer Science Department, Keele University (TR/SE-0401) and National ICT AustraliaLtd. (0400011T.1), 2004

[9] V. Garousi, M. Felderer, and T. Hacaloglu, "What We Know about Software Test Maturity and Test Process Improvement," IEEE Software, 2018

[10] V. Garousi, M. Felderer, T. Hacaloglu, "Software test maturity assessment and test process improvement: A multivocal literature review," Information and Software Technology, pp.1-27, 2017

[11] W. Afzal, S. Alone, K. Glocksien, and R. Torkar, "Software test process improvement approaches: A systematic literature review and an industrial case study," The Journal of Systems and Software, vol. 111, pp.1-33, 2016

[12] J. W. Jacobsen, M. Kuhrmann, J. Munch, P. Diebold, and M Felderer, "On the Role of Software Quality Management in Software Process Improvement," PROFES 2016, LNCS 10027, pp. 327–343, 2016