ENACTEST - European Innovation Alliance for Testing Education

Beatriz Marín[1], Tanja E. J. Vos[1,2], Ana C. R. Paiva[3], Anna Rita Fasolino[4] and Monique Snoeck[5]

[1]*Universitat Politècnica de València (UPV), Camino de Vera s/n, Valencia, 46021, Spain*
[2]*Open Universiteit (OU), The Netherlands*
[3]*Faculty of Engineering of the University of Porto & INESC TEC, Rua Dr. Roberto Frias, s/n 4200-465 Porto, Portugal*
[4]*Università degli Studi di Napoli Federico II, DIETI, Via Claudio 21, Italy*
[4]*KU Leuven, Naamsestraat 69, box 3500, 3000 Leuven, Belgium*

Abstract

Testing software is very important, but not done well, resulting in problematic and erroneous software applications. The cause radicates from a skills mismatch between what is needed in industry, the learning needs of students, and the way testing is currently being taught at higher and vocational education institutes. The goal of this project is to identify and design seamless teaching materials for testing that are aligned with industry and learning needs. To represent the entire socio-economic environment that will benefit from the results, this project consortium is composed of a diverse set of partners ranging from universities to small enterprises. The project starts with research in sensemaking and cognitive models when doing and learning testing. Moreover, a study will be done to identify the needs of industry for training and knowledge transfer processes for testing. Based on the outcomes of this research and the study, we will design and develop capsules on teaching software testing including the instructional materials that take into account the cognitive models of students and the industry needs. Finally, we will validate these teaching testing capsules developed during the project.

Keywords

software testing, education, knowledge transfer, cognitive models

1. Introduction

Testing software is very important, but not done well, resulting in problematic and erroneous software applications. This has a noticeable impact on society, economy and innovation since software is at the basis of digital solutions and tools related to climate change, green economy, demography, digitalisation, artificial intelligence, and the recent COVID-19 pandemic. The reason that testing is not done well radicates from a skills mismatch between what is needed in industry, the learning needs of students and the way testing is currently being taught at Higher Education (HE) and Vocational Education and Training (VET) institutes. Testing needs to allocate multiple cognitive resources in students which makes it a challenge to learn and

Joint Proceedings of RCIS 2022 Workshops and Research Projects Track, May 17-20, 2022, Barcelona, Spain
bmarin@dsic.upv.es (B. Marín); tvos@dsic.upv.es (T. E. J. Vos); apaiva@fe.up.pt (A. C. R. Paiva); fasolino@unina.it (A. R. Fasolino); monique.snoeck@kuleuven.be (M. Snoeck)
0000-0001-8025-0023 (B. Marín); 0000-0002-6003-9113 (T. E. J. Vos); 0000-0003-3431-8060 (A. C. R. Paiva); 0000-0001-7116-019X (A. R. Fasolino); 0000-0002-3824-3214 (M. Snoeck)

teach [1]. The goal of this project is to identify and design seamless teaching materials for testing that are aligned with industry needs and which take into account also the learning needs and characteristics of students. This can only be achieved by a project in which HEs, VETs and companies cooperate to share knowledge and develop new approaches for education that take into account a broader socio-economic environment, including research into the sensemaking and cognitive models of learning software testing.

Why: *testing is important but not done well*

People and society are more and more dependent on software quality. Software more and more determines our daily lives, since our social and business lives are digitised. We see that software failures have increasingly more impact. A report from Failwatch [2] identified 548 recorded software failures impacting 4.4 billion people and $1.1 trillion in assets. And this is just scratching the surface—there are far more software bugs in the world than we will likely ever know about. In another study [3] it was found that, for the year 2020 in the US alone, the total Cost of Poor Software Quality (CPSQ) was $2.08 trillion. It must be clear that the cost of bad quality is getting out of hand and voices are already mentioning the coming software apocalypse [4].

Testing is, at the moment, the most important and most-used quality assurance technique applied in the software industry. The complexity of software and, hence of its development, is increasing. Modern systems get larger and more complex as they connect large amounts of components that interact in many different ways and must satisfy constantly changing and different types of requirements (functionality, dependability, security, etc.). Data processing that impacts all aspects of our life is increasingly distributed over clouds and devices. This leads to new emerging concerns, such as availability, security, and privacy.

Consequently, testing is getting more and more important and complex too. Companies need to apply systematization and automation of testing throughout the software and system life-cycle so as to keep up with the increasing quality requirements.

Despite the pressing need for good testing practices, there is a lack of testing culture and awareness amongst practitioners in companies [5] and students in academia [6], [7]. In many cases, programmers do not test even though they theoretically understand the value of testing [6],[8]. Practitioners put off writing effective test cases because they are under pressure to deliver the product as quickly as possible, which finally results in low quality software. Students do not test because testing has not been sufficiently integrated into computer science curricula and hence they still can get away with it [9].

Furthermore, knowledge transfer among the projects rarely happens, so that testing challenges related to test case design, scripting, test execution, reporting, management, automation, or even the lack of general knowledge of testing must be faced again every time people start a new project, as it can be evidenced in the industry testing challenges presented in [5]. In addition, taking into account the cognitive process that must be performed when people test, and that the quality of test cases designed is affected by the domain knowledge and testing expertise of people [10], companies also need strategies to transfer the knowledge inside the teams.

➜ **WHAT is needed**: *the problem should be tackled at the root: Education*

We see more and more interest in the topic of Testing in education, for instance the systematic literature reviews papers specifically focused on testing education [6] and [7], and also in workshops (https://testedworkshop.github.io/) dedicated entirely to testing in education.

Academy and industry permanently remark the importance of software-testing techniques to improve software quality and to reduce development and maintenance costs. In many cases, novice software engineers argue that they do not know if they are well prepared to do those tasks.

In academia, several efforts have been made to include testing in the curricula and to properly teach software testing techniques to have students better prepared for industry. Nevertheless, teaching - and learning - testing techniques are not easy tasks. Software testing is a complex and intellectual activity based on analysis, reasoning, decision making, abstraction and collaboration. Testing needs to allocate multiple cognitive resources in students which makes it a challenge to teach [1].

An extensive mapping review of 273 scientific works published from 2000 on the topic of inclusion of testing in academia was published in 2019 [6]. This work states that including testing in curricula is not straightforward, and it presents nine topics that have been researched to improve testing in academia. These topics correspond to the inclusion of testing in the curricula as a separated topic or jointly with programming courses; the use of teaching methods that integrate testing when teaching programming; the creation of course material about testing; the creation of programming assignments that include testing practices; the definition of declarative programming processes for novices that include testing; the use of supporting tools for testing; quality assessment of students' code by testing their code; students' attitude towards software testing; and assessment of students' knowledge about testing.

Even though the inclusion of testing in academia has evident benefits such as the improvement of the students programming performance, timely feedback for students, objective assessment, and better understanding of the programming process by the students, there are still drawbacks. All these existing initiatives focus on testing as an isolated topic, they do not take into account the training needs of industry nor those of the students, and they are positioned too late in the curriculum. This leads to disconnection and a gap between theory and practice, less interest from students and hence a negative attitude towards testing, students who are not confident of their testing skills because of the lack of measurement of students performance at testing, and hence, absence of testing activities by students and the future practitioners they become. Moreover, course staff feels that they have considerably more workload when including separate testing topics.

2. ENACTEST: a recently granted ERASMUS+ project

As described in the previous section, we need to tackle the problems identified in testing at the root, i.e. during education where future professionals are prepared. Our hypothesis is that software testing should be integrated throughout the whole computer science curriculum in the following 4 ways:

1. First, it should be done as early as possible.
2. Second, it should be done seamlessly in a smooth and continuous way as an inherent part of programming, and not as a separate activity.
3. Third, it should take into account the needs of industry.
4. Fourth, it should be based on testing experts' mental models and take into account students' sensemaking and learning models to tackle the inherent complexity and intellectual challenges that are faced when teaching testing.

Thus, the objective of the ENACTEST project (2022-2025) is to identify and to design early and seamless teaching materials for testing that are aligned with industry needs and which take into account also the learning needs and characteristics of students, with the purpose of analyze the feasibility to integrate them and apply them in the curricula of the HE and VET partners and in the training process of SMEs partners in order to improve the learning performance of students and also reduce training needs about testing of industry. As a final result, this will improve the knowledge transfer of testing amongst teams, academia, professionals and novice engineers.

At this point, it is important to mention that the ENACTEST project does not intend to design brand new curricula or to develop ready-made courses. Neither it intends to oblige teachers to change their entire courses or curricula. We understand that these solutions are not feasible and will not be taken up by a wide audience. Yet ENACTEST will offer bite-sized testing teaching materials (that we will call capsules) that educators can easily integrate into their courses without the need to change their initial programs and adding additional workload. Moreover, counting with these teaching materials will allow industry to improve their training process for novice engineers. Two important characteristics of the developed teaching materials, and hence the reason for us to call them capsules, are that they are bite-sized such that they allow seamless integration. These two properties allow the widespread take-up of the results of this project.

2.1. Objectives

We identified several skills mismatches around testing education. For instance, industry needs students that are better prepared to do testing and hence improve software production. In order to improve education, HE needs to understand what they should teach considering industry needs of testing; and finally, in order to teach well, HE needs to understand the students' learning processes to improve the education of testing. We summarize all these skills mismatches as three gaps in testing education: the gap between academy and students, the gap between industry and academy and the gap between graduated students and industry.

Considering the complex intellectual processes that are performed when testing software, and taking into account the intrinsic characteristics of students that they use to identify what to test and how to test; we identify a gap between academia and students since there is a disconnection between the teaching methods and the learning needs of students. Testing is often seen as an isolated (boring) topic and is positioned too late in the curriculum. Moreover, from the students, we need to understand their learning models when students perform testing. What cognitive abilities do we need to design tests and explore different scenarios to check if the program meets the requirements, as well as, to identify possible bugs of the programs. To do that, they need to use their knowledge, skills, creativity, strategy and cognitive resources to perform testing. Therefore, it is important to identify the students' learning models while making sense of the problems in order to properly present teaching materials that maximize the learning performance of students.

The gap between industry and academia is a result of the low collaboration between these two stakeholders, which can be observed in the low relevance of the contributions done by several researchers in academia and the difficulties that industry has when they want to take advantage of them or use these contributions in their daily work. Even though the global software industry and the academic software engineering (SE) are two large communities, still, the level of joint activities in an SE industry-academia collaboration (IAC) is low; compared to the amount of activities in each of the two communities [11], [12]. This also holds for the level of IACs in the software testing [13], [12]. For example, a survey of software testing practices involving 246 practitioners from Canada [13] revealed that 56% of the respondents had never interacted with researchers in academia during their career, and 32% reported "seldom" interactions with researchers only. A small portion of respondents (12%) mentioned interacting with researchers once a year or more. Thus, there is a "gap" in industry-academia collaborations and knowledge exchange in software testing research, education, and training across the globe and in Europe [14]. In the research front, the gap leads to low relevance of research conducted by universities, and also inability of companies to benefit from/utilize the innovations by researchers [15], [16], [12]. In the education/teaching front, the gap leads to inadequate software-testing skills of students and university graduates [17]. Furthermore, there is a need for companies to share best practices and knowledge among themselves.

The low collaboration between industry and academia results in the fact that testing knowledge and skills needed by novice engineers are not taught to the students, so that, at the end of their career, they are not prepared with the skills that they need to work in industry. We call this situation the gap between students and industry.

From the companies, we need to understand their needs and pains about testing education, which are related to technical issues as well as skills issues. Currently, companies need to create their own solutions to train people about testing techniques, skills, and to transfer knowledge from expert testers to the novice ones, as well as the knowledge transfer among the team members of different projects. Thus, we need to categorize and identify possible solutions for this issue in order to provide companies with graduate students better prepared and also to provide approaches to facilitate the training and knowledge transfer about testing inside the companies.

From the academy, we need to investigate new techniques to educate, such as gamification [18], modelling [19], and early testing [20] in order to respond to the needs of companies and students, and also taking into account the learning models of students and the training methods in companies.

Therefore, this project wants to fill in three gaps of testing education: the gap between industry and academy, the gap between academy and students, and the gap between graduated students and industry. To do that, this project will look from 3 perspectives: students, companies and teachers as shown in figure 1.

Thus, we define the following specific objectives for the ENACTEST project:

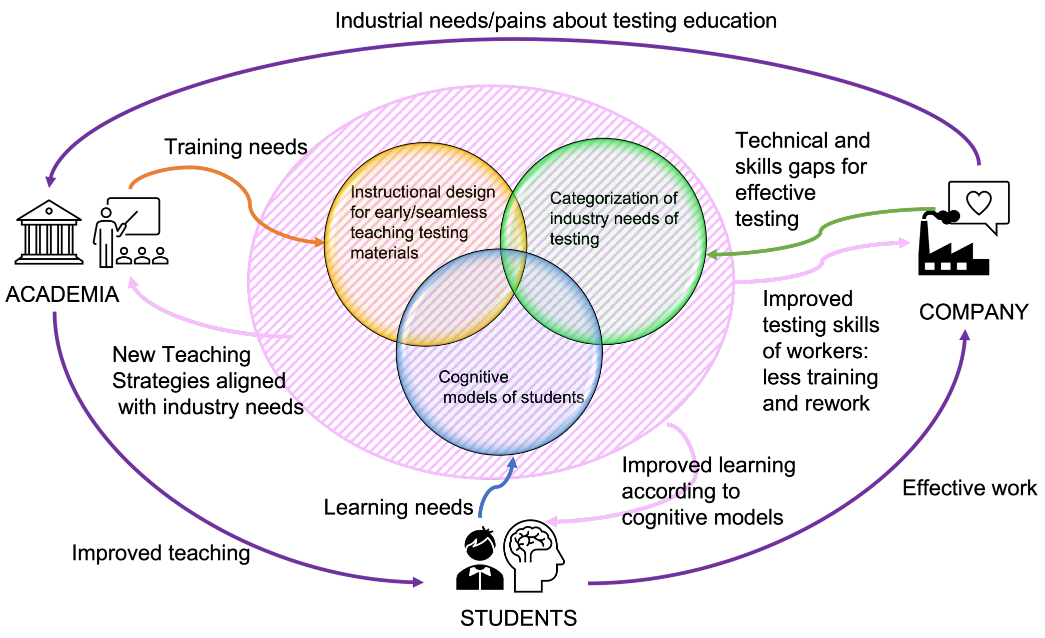
Industrial needs/pains about testing education
Training needs
ACADEMIA
Instructional design for early/seamless teaching testing materials
Categorization of industry needs of testing
Technical and skills gaps for effective testing
COMPANY
Cognitive models of students
New Teaching Strategies aligned with industry needs
Improved testing skills of workers: less training and rework
Improved learning according to cognitive models
Learning needs
Effective work
Improved teaching
STUDENTS

Figure 1: ENACTEST ecosystem

SO1: To identify the cognitive models for testing that are used by students and experts when they need to deal with testing, specifically for test case design.

SO2: To categorize the industry needs and pains with regard to technical and skills of testing in order to identify topics and skills that are fundamental to include in academic curricula.

SO3: To design and develop new specific bite-sized testing materials (capsules) for including them into education as early and seamlessly as possible. These take into account the cognitive models of students and, at the same time, the needs of industry.

SO4: To provide evidence about the improvement in learning performance of testing by the students and the improvement of knowledge transfer in industry.

2.2. Expected results

ENACTEST will develop and implement new approaches for teaching/learning software testing with students at the center of the process, using real problems of companies. The proposed solution will be delivered through innovative bite-sized capsules that take into account 3 different views (professors, students, and industry) involving all the key actors in this process.

At Local level, the project expects to steer the educational community to find the gaps in the curricula, as well as involve companies to identify good testing practices. These two worlds previously not strongly linked in the software testing area, will come together and work in local clusters that research for excellence in software testing. This can mean that new positions will be created in companies as they will realise the importance and benefits of testing, and thus employment will rise.

At the National level, it is expected that the produced capsules for teaching will be highlighted and adopted by several educational institutions making the curricula more responsive to the market needs. That will breed a new generation-solution to the rising problem of diminishing quality software and boost the market competitiveness of the countries by inducing a sector growth in software testing.

At European level, the cross-border collaboration will guarantee that the developed solution combats problems in software quality that are pan European, and thus raise the quality innovation. Moreover, it presumes that better software products mean better market share in the IT sector worldwide.

Finally, we advocate ENACTEST will improve the learning performance of students and improve their testing skills as they are increasingly important in digital job profiles across the entire labour market. In the long run this will improve the quality of the software on which our digitalised society relies.

3. Relevance to RCIS

Software is at the core of the information systems that provide digital services to society. The quality of these systems is of fundamental importance and determined by the processes used for engineering these systems. Testing is one of these engineering processes. However, testing is not done well because there is a skills mismatch between what is needed in the industry, the learning needs of students, and the way testing is currently being taught.

This project focuses on bridging the gap among industry, academia, and computer science students regarding testing skills and knowledge. This topic is very relevant for the RCIS community since it aims to bring together researchers, engineers and professionals and provide opportunities for sharing and disseminating knowledge about software testing.

ENACTEST will provide an holistic approach to fill in these gaps through testing capsules. This will improve the learning performance of students and improve their testing skills as they are increasingly important in digital job profiles across the entire labour market. In the long run this will improve the quality of the software on which our digitalised society relies.

Acknowledgments

This project has been funded by ERASMUS-EDU-2021-PI-ALL-INNO under the number 101055874.

References

[1] E. Enoiu, G. Tukseferi, R. Feldt, Towards a model of testers' cognitive processes: Software testing as a problem solving approach, in: 2020 IEEE 20th International Conference on Software Quality, Reliability and Security Companion (QRS-C), IEEE, 2020, pp. 272–279.

[2] The cost of poor software quality in the us: A 2020 report, 2020. URL: https://www.it-cisq.org/pdf/CPSQ-2020-report.pdf.

[3] The software fail watch, 2018. URL: https://www.tricentis.com/blog/software-fail-watch-q2-2018/.

[4] J. Somers, The coming software apocalypse, a small group of programmers wants to change how we code—before catastrophe strikes, 2016.

[5] V. Garousi, M. Felderer, M. Kuhrmann, K. Herkiloğlu, S. Eldh, Exploring the industry's challenges in software testing: An empirical study, Journal of Software: Evolution and Process 32 (2020) e2251.

[6] L. P. Scatalon, J. C. Carver, R. E. Garcia, E. F. Barbosa, Software testing in introductory programming courses: A systematic mapping study, in: Proceedings of the 50th ACM Technical Symposium on Computer Science Education, 2019, pp. 421–427.

[7] V. Garousi, A. Rainer, P. Lauvås Jr, A. Arcuri, Software-testing education: A systematic literature mapping, Journal of Systems and Software 165 (2020) 110570.

[8] A. Afzal, C. Le Goues, M. Hilton, C. S. Timperley, A study on challenges of testing robotic systems, in: 2020 IEEE 13th International Conference on Software Testing, Validation and Verification (ICST), IEEE, 2020, pp. 96–107.

[9] T. E. J. Vos, Zoeken naar fouten: op weg naar een nieuwe manier om software te testen, 2017.

[10] K. Juhnke, M. Tichy, F. Houdek, Challenges concerning test case specifications in automotive software testing: assessment of frequency and criticality, Software Quality Journal 29 (2021) 39–100.

[11] R. L. Glass, Software Creativity 2.0, developer.* Books, 2006.

[12] V. Garousi, M. M. Eskandar, K. Herkiloğlu, Industry–academia collaborations in software testing: experience and success stories from canada and turkey, Software Quality Journal 25 (2017) 1091–1143.

[13] V. Garousi, J. Zhi, A survey of software testing practices in canada, Journal of Systems and Software 86 (2013) 1354–1376.

[14] V. Garousi, M. Felderer, Worlds apart: industrial and academic focus areas in software testing, IEEE Software 34 (2017) 38–45.

[15] V. Garousi, K. Petersen, B. Ozkan, Challenges and best practices in industry-academia collaborations in software engineering: A systematic literature review, Information and Software Technology 79 (2016) 106–127.

[16] P. Runeson, S. Minör, The 4+ 1 view model of industry–academia collaboration, in: Proceedings of the 2014 international workshop on Long-term industrial collaboration on software engineering, 2014, pp. 21–24.

[17] S. M. et al., The international dimension of e-skills and the impact of globalisation, 2014. URL: www.cepis.org/media/FINAL_INTERNATIONAL_e_Skills_report_Aug_141.pdf.

[18] A. Lapeña, B. Marín, T. E. J. Vos, Gipppy: Game for introductory programming practice in python, in: 16th annual International Technology, Education and Development Conference, 2022, pp. 224–229.

[19] B. Marín, S. Alarcón, G. Giachetti, M. Snoeck, Tescav: An approach for learning model-based testing and coverage in practice, in: International Conference on Research Challenges in Information Science, Springer, 2020, pp. 302–317.

[20] N. Doorn, T. Vos, B. Marín, Test Informed Learning with Examples assignments, 2021. URL: https://tile-repository.github.io/TILES/.