Enabling Pragmatic Software Testing Education in Brazil through an Education Repository: An Initial Proposal

JOÃO GOMES, Universidade Federal do Maranhão, Brazil
KENNEDY NUNES, Universidade Federal do Maranhão, Brazil
LUIS RIVERO, Universidade Federal do Maranhão, Brazil
DAVI VIANA, Universidade Federal do Maranhão, Brazil
GERALDO BRAZ JUNIOR, Universidade Federal do Maranhão, Brazil
JOÃO ALMEIDA, Universidade Federal do Maranhão, Brazil
VANDECIA FERNANDES, Universidade Federal do Maranhão, Brazil
SIMARA ROCHA, Universidade Federal do Maranhão, Brazil
ARISTÓFANES SILVA, Universidade Federal do Maranhão, Brazil

When teaching software testing, it is essential that the students have access to the practical application of the theoretical concepts regarding software testing techniques. Thus, it is important that support materials and real artifacts are available for the practical application of the testing approaches. Despite this importance, undergraduate students from computer science courses still face difficulties in the practical application of the contents covered in the classroom. In this work in progress, our goal is to support the learning process by offering artifacts that allows students to improve their practical skills in software testing. For this, an analysis of the set of test techniques taught in classes and required by the job market was carried out. We also performed an analysis of the features of the repositories that provide support artifacts for teaching these techniques. Considering that the identified repositories are not in Portuguese, we present a high-fidelity prototype of a new Brazilian repository, aggregating the identified artifacts and proposing new ones for supporting testing teaching.

CCS Concepts: • **Software and its engineering → Software verification and validation**; • **Information systems**;

Additional Key Words and Phrases: Software Testing, Computer Science Education, Repository

ACM Reference Format:
João Gomes, Kennedy Nunes, Luis Rivero, Davi Viana, Geraldo Braz Junior, João Almeida, Vandecia Fernandes, Simara Rocha and Aristófanes Silva. 2022. Enabling Pragmatic Software Testing Education in Brazil through an Education Repository: An Initial Proposal. In LIQUE 2022: Life Improvement in Quality by Ubiquitous Experiences Workshop, together with IMX 2022: ACM International Conference on Interactive Media Experiences. June 22-24, 2022, Aveiro, Portugal. ACM, New York, NY, USA, 8 pages. https://doi.org/XXXXXXX.XXXXXXXA NY, USA.

Authors' addresses: João Gomes, Universidade Federal do Maranhão, Departamento de Informática, São Luis, Brazil, jpg.souza@discente.ufma.br; Kennedy Nunes, Universidade Federal do Maranhão, Departamento de Informática, São Luis, Brazil, kennedy.anderson@discente.ufma.br; Luis Rivero, Universidade Federal do Maranhão, Programa de Pós-Graduação em Informática, São Luis, Brazil, luisrivero@nca.ufma.br; Davi Viana, Universidade Federal do Maranhão, Programa de Pós-Graduação em Informática, São Luis, Brazil, davi.viana@ufma.br; Geraldo Braz Junior, Universidade Federal do Maranhão, Programa de Pós-Graduação em Informática, São Luis, Brazil, geraldo@nca.ufma.br; João Almeida, Universidade Federal do Maranhão, Programa de Pós-Graduação em Informática, São Luis, Brazil, jdallyson@nca.ufma.br; Vandecia Fernandes, Universidade Federal do Maranhão, Núcleo de Computação Aplicada, São Luis, Brazil, vandecia@ecp.ufma.br; Simara Rocha, Universidade Federal do Maranhão, Núcleo de Computação Aplicada, São Luis, Brazil, simara.rocha@ufma.br; Aristófanes Silva, Universidade Federal do Maranhão, Programa de Pós-Graduação em Informática, São Luis, Brazil, ac.silva@ufma.br.

1 INTRODUCTION

Software verification and validation is the activity of analyzing artifacts (documents, source code, diagrams, among others) produced in software development to identify and eliminate defects [3]. With the advancement of Software Engineering, the software produced becomes more complex, which can increase the occurrence of defects. Thus, the software development industry demands software engineers with knowledge in the testing area who know how to apply appropriate techniques for the conditions in which the software will be tested. However, there is a deficiency in terms of skilled labor in this area [2]. To keep up with this trend, it is necessary for software engineers to train themselves more and more in the test area.

To support software engineers in accessing teaching support materials aimed at the practice of software testing activities, this work presents the process of developing a proposal for a repository with these materials. For the development of the proposal, an analysis of online test teaching repositories was carried out regarding the presence of support materials that allow the practice of test techniques as close as possible to a real application. This evaluation aimed to assess the scope and quality of materials to support testing practice available in these repositories. Based on the results, this work in progress presents a new repository for the Brazilian community, containing artifacts generated from the analysis of materials used in the teaching of software testing techniques.

2 BACKGROUND

2.1 Software Testing Concepts

Initially, it is necessary to discover how the literature defines and classifies software testing techniques to investigate which of these techniques have material to support practical teaching available. For this, the software test study programs for the industry provided by ISTQB were used: *syllabi ISTQB*[1], *Foundation Level Specialist Syllabus Usability Testing*[2], *Specialist Syllabus Performance Testing*[3] and *Advanced Level Extension Syllabus Security Testing* (2016)[4]. As a complementary basis, the work by Barbosa et al. [1] which describes several approaches to software testing that are taught in undergraduate courses. After analyzing the surveyed techniques, a classification of software testing techniques was proposed, considering their occurrence in the analyzed documents. The classification is presented below, explaining each of the identified techniques.

Functional Test (Black Box): These are techniques that assess whether the system complies with its functional specification, namely: (a) **Partitioning into Equivalence Classes**: divides the input domain into valid and invalid equivalence classes, based on the input conditions identified in the program; (b) **Limit Value Analysis**: the test cases are selected at the boundaries of each class; (c) **Test Graph Cause Effect**: Causes (input conditions) and effects (actions) are identified and combined into a graph from which a decision table is built. From this table, test cases and outputs are derived; (d) **State Transition Test**: The system to be tested is represented as a state transition diagram. Test cases are defined in such a way that each diagram path is traversed; and (e) **Use Case Test**: Generate test cases by interpreting the system use case diagram.

Structural Testing (White Box): Uses implementation aspects to derive test requirements, based on knowledge of the implementation's internal structure. The techniques used in the White-Box test are: (a) **Complexity Based:** Uses information about the complexity of the program to determine test requirements; (b) **Control Flow Based:** Uses only program execution control features, such as commands or branches, to derive test requirements; and (c) **Based on Data Flow:** They use information about the data flow of the program to derive test requirements, that is, these criteria require that interactions involving variable definitions and references to these definitions be tested.

[1]Available at: https://bstqb.org.br/b9/doc/syllabus_ctfl $_2$018$br.pdf$
[2]Available at: https://bstqb.org.br/b9/doc/syllabus_ctfl_ut_2018br.pdf
[3]Available at: https://bstqb.org.br/b9/doc/syllabus_ctfl_en_2018br.pdf
[4]Available at: https://bstqb.org.br/b9/doc/syllabus_ctal_sec_2016br.pdf

Experience-Based Testing Techniques: In this type of techniques, test cases are derived from the tester's skill and intuition and his experience with similar systems. The experience-Based Testing techniques are: (a) **Error Assumption:** predicts the occurrence of defects based on the tester's knowledge of the defect history of the system under test and similar previous systems; and (b) **Exploratory Testing**: This technique is an iterative and empirical exploration activity that requires back and forth in an investigation process with the user's eye on it.

Additional Assessments: These techniques allow assessing the non-functional requirements of the system. Some of the additional testing techniques are: (a) **Usability Testing:** Usability testing assesses the degree to which the system can be used effectively, efficiently and satisfactorily by specific users in a specific context of use; (b) **Performance Testing:** Performance testing is an umbrella term that includes any type of testing focused on system or component performance under different load volumes; (c) **Security Test:** Assesses a system's vulnerability to threats when attempting to compromise the system's security policy; and (d) **Based on Checklist**: In the literature checklists are considered a category of inspection technique, which can be used to find defects at any stage of software development. A *checklist* has items that capture important lessons learned in previous reviews. Individual items can list, prioritize, or pose questions to help the reviewer discover characteristic defects.

2.2 Supporting Software Testing Teaching

The lack of trained professionals can be caused by the fact that software testing teaching is mainly taught from theoretical classes, with not enough practical materials to develop the [6] software testing content. In this context, the ISTQB® (*International Software Testing Qualifications Board*) is an organization that aims to formalize and certify the teaching of software testing in the world. In addition to providing professional assessment and issuance of certificates, this organization shares the *syllabi* that compiles the knowledge needed for each level of software testing expertise. Although these materials are a great source of information for software engineers interested in learning about software testing practices, these materials mainly present theoretical content or are not detailed enough to present, in practice, how to carry out the suggested testing activities.

The concern with analyzing and finding ways to organize teaching materials in software testing has already been addressed in other works. For example, Valle et al. [7] carried out a systematic mapping to verify the academic production in test teaching and came to the conclusion that there is a lack of virtual environments that motivate test learning. In addition, they also identified the need for studies that addressed all phases of testing. Finally, the authors proposed the creation of a digital game aimed at teaching test, to motivate student learning.

Garousi et al. [4] gathered a total of 204 articles dated between 1992 and 2019 that bring together concepts in software testing and experience reports from students and teachers in the area. The results were obtained from a systematic literature mapping and made available in a repository on the Google Drive platform. Most of the material gathered deals with reports of teachers' experiences and pedagogical approaches to test teaching. However, most of these approaches involve teaching in the classroom, where a teacher instructs or assists the student. As such, these approaches may not be effective for self-taught students or students seeking individual improvement.

Considering the works above, among the challenges in teaching software testing are the cognitive load in test learning and problems related to [4] tools. In addition, instructors report difficulties in aligning test teaching with the needs of the [4] industry. Additionally, students indicate difficulties with issues related to the application of specific [5] test techniques. The results reported in these works show the importance of support materials in test teaching, especially when the student tries to learn or reinforce the contents on their own. Given the above, in this work a repository will be proposed to include materials to support the practical teaching of testing. The methodology applied to develop this research is presented below.

3 METHODOLOGY

Considering that students and novice software engineers interested in being trained in the application of software testing techniques do not have access to support materials for the practical training of these techniques, we proposed to create a repository with these materials. To achieve this objective, the authors of this work carried out 3 main activities: (a) identification of existing repositories; (b) analysis of support materials made available by these repositories by type of software testing technique identified in the literature (see Subsection 2.1); and (c) proposal of a Brazilian repository considering the identified gaps. These activities are described in detail below.

Initially, to identify which software testing teaching repositories were available, a query was carried out in the Google search engine using keywords in Portuguese and English that referred to support material for the practical teaching of testing, such as: *"test code repository", "software testing practice", "test teaching repository", "test teaching support material", "Software testing demo repositories", "Software testing exercise", "Software testing demo", "Software testing learning" and "practicing software testing"*, among others. From the repositories returned, the first 10 that were free to access and created by educational institutions, teachers, software testing professionals or companies in the area were chosen. The choice of the first 10 repositories was motivated by the operating logic of the search engine, in which pages most related to the terms used are returned according to their degree of correspondence with the keywords used. The 10 identified repositories are presented below, ordered by the amount of support materials made available by them to support the practical teaching of software testing. It is worth mentioning that although the search used terms in Portuguese, the authors of this work did not identify free access repositories with material to support the practical teaching of software testing developed in Brazil.

- R1 - SoftwareTestingHelp[5] (Total Materials: 34)
- R2 - Guru99[6] (Total Materials: 27)
- R3 - Software Testing Material [7] (Total Materials: 16)
- R4 - ArtOfTesting[8] (Total Materials: 14)
- R5 - TestBytes[9] (Total Materials: 9)
- R6 - SeleniumEasy [10] (Total Materials: 9)
- R7 - Edureka [11] (Total Materials: 8)
- R8 - TutorialsPoint [12] (Total Materials: 3)
- R9 - Octoperf [13] (Total Materials: 3)
- R10 - MartinFowler [14] (Total Materials: 1)

Table 1 shows which materials were made available by the repositories according to the type of technique analyzed. In this table, the techniques are organized according to their categories and the repositories presented according to the amount of support materials provided. When analyzing the degree of supply of materials from the repositories, most repositories do not provide support material that allows for an in-depth study of the identified techniques in general. In addition, repositories that support the teaching of various techniques do not have all the materials necessary for their practice. Finally, none of the repositories was able to concentrate material, even theoretical, on all the techniques identified in this work. Based on the identified problems, and

[5]Available at: https://www.softwaretestinghelp.com/
[6]Available at: https://www.guru99.com/
[7]Available at: https://www.softwaretestingmaterial.com/
[8]Available at: https://artoftesting.com/
[9]Available at: https://www.testbytes.net/mobile-app-testing-blog/
[10]Available at: https://www.seleniumeasy.com/
[11]Available at: https://www.edureka.co/blog/what-is-software-testing/
[12]Available at: https://www.tutorialspoint.com/software_testing/index.htm
[13]Available at: https://octoperf.com/blog/
[14]Available at: https://martinfowler.com/testing/

considering that there are no repositories in Portuguese, a new repository was proposed to assist in dealing with these problems. The repository prototype will be described next.

Table 1. Analysis of the repositories of materials to support the practical teaching of software testing, accounting for the amount of materials made available per test technique

Black-box testing																																
	Partitioning into Equivalence Classes						Limit Value Analysis						Test Graph Cause Effect							State Transition Test						Use Case Test						
Provided Support Materials	Theoretical Basis of the Technique	System Specification	Definition of Equivalence Classes	Test Cases	System to be Tested	Oracle with Defects List	Theoretical Basis of the Technique	System Specification	Definition of Equivalence Classes and Limit Values	Test Cases	System to be Tested	Oracle with Defects List	Theoretical Basis of the Technique	System Specification	Cause-effect graph	Decision Table	Test Cases	System to be Tested	Oracle with Defects List	Theoretical Basis of the Technique	System Specification	State Diagram	Test Cases	System to be Tested	Oracle with Defects List	Theoretical Basis of the Technique	System Specification	Interaction flow diagram	Test Cases	System to be Tested	Oracle with Defects List	Number of Features
R1	X						X						X	X	X	X	X			X	X	X	X			X	X	X	X			15
R2	X	X	X	X	X		X	X	X	X	X									X	X	X				X	X					15
R3	X		X	X			X		X	X										X		X	X									9
R4	X	X	X				X	X	X				X							X						X						9
R5	X						X						X							X						X						5
R6					X						X							X						X						X		5
R7	X		X				X		X																							4
R8																																0
R9																																0
R10																																0

White-box testing																			
	Complexity Based						Control Flow Based						Based on Data Flow						
Provided Support Materials	Theoretical Basis of the Technique	System Specification	System Code to be Tested	Control Flow Graph	Test Cases based on Complexity	Oracle with Defects List	Theoretical Basis of the Technique	System Specification	System Code to be Tested	Control Flow Graph	Test Cases based on Control Flow	Oracle with Defects List	Theoretical Basis of the Technique	System Specification	System Code to be Tested	Control Flow Graph	Test Cases based on Data Flow	Oracle with Defects List	Number of Features
R1	X		X	X			X		X	X			X						7
R2	X		X	X	X		X		X	X	X								8
R3							X	X		X									3
R4																			0
R5																			0
R6																			0
R7																			0
R8																			0
R9																			0
R10																			0

Tests Based on Experience									
	Error Assumption				Exploratory Testing				
Provided Support Materials	Theoretical Basis of the Technique	Error History in Previous System Versions	System to be Tested in New Version	Oracle with Defects List	Theoretical Basis of the Technique	System to be Tested	Test Cases Identified Based on Exploration	Oracle with Defects List	Number of Features
R1					X	X	X		3
R2					X				1
R3					X				1
R4					X	X			2
R5					X				1
R6						X			1
R7									0
R8									0
R9						X			1
R10					X				1

Additional Tests																		
	Usability Testing				Performance Testing				Security Test				Based on Checklist					
Provided Support Materials	Theoretical Basis of the Technique	Usability Criteria	Prototype or System to be Tested	Oracle with Defects List	Theoretical Basis of the Technique	System Specification with Defined Performance Criteria	Automated Test Case Scripts	System to be Tested	Theoretical Basis of the Technique	System Specification with Security Criteria	Automated Test Case Scripts	System to be Tested	Theoretical Basis of the Technique	System Overview and Inspection Objectives	Checklist	Artifact to be inspected	Oracle with Defects List	Number of Features
R1	X	X	X		X	X	X		X	X			X					9
R2	X				X				X									3
R3	X				X				X									3
R4	X				X				X									3
R5	X				X				X									3
R6				X				X				X						3
R7	X				X	X	X											4
R8	X				X				X									3
R9					X				X									2
R10																		0

4 INITIAL RESULTS

Considering the needs identified from the analysis of the repositories to support the practical teaching of software testing, we chose to create the "Broken" repository, alluding to the breakdown of systems when they present catastrophic defects. The repository has been implemented as a website, available online[15].

The purpose of the Broken repository is to support the practice of software testing, providing support materials for the techniques identified in the literature, both manually and automatically. To date, support materials are being developed considering the needs raised for the practical application of the techniques identified in Subsection 2.1. In this context, systems, source codes and/or artifacts involved in the development process (documentation, diagrams, among others) with seeded defects are being produced to feed the repository. To exemplify the use of the repository, an example will be used considering the technique of functional testing based on partitioning by equivalence classes.

Initially, the repository presents a side menu with a summary of the software test subjects present in the repositories. One idea is for the repository to be *open source*, which will encourage community interaction, allow novice testers to connect with more experienced testing professionals, and add more support material over time. Also, when searching for a software testing technique, the system will open a new page where the support materials will be presented following the logical order of their creation to simulate and/or practice the application of the selected technique.

Figure 1 presents an exercise in which all the materials needed to perform a partitioning test by equivalence class are made available. In this example, a specification of a banking system is presented. In the specification details about how the system works and the types of input it accepts are given. For example, the system should not allow null or negative withdrawal amounts; or greater than the balance. In addition to this information, the repository will also provide a fictitious systems considering the documentation presented. In this example, the system will be an executable code of a banking system containing some seeded defects.

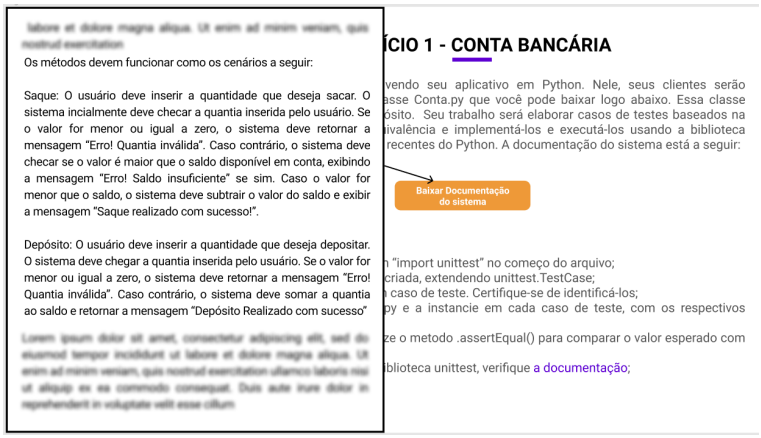

Fig. 1. Exercise to test a simple banking system to be implemented in Python with its documentation

To help the student if he does not know how to apply the technique and identify defects, the answer of the applied exercise will be made available. In this example, the tester would need to generate the information related to the various inputs and outputs and their respective partition. In addition, the test cases and the failures of

[15] Available at: https://petcompufma.org/dexters/broken/

these cases will be made available to the user with respect to the system that would be tested. Figure 2 presents the partitions for identifying test cases.

Fig. 2. Resolution of the proposed exercise, showing definition of equivalence classes and test cases

5 CONCLUSIONS AND FUTURE WORK

This paper presented an analysis of online test practice teaching repositories based on the literature. We identified a the lack of artifacts to support the testing practice. As a solution, the creation of the Broken collaborative repository was proposed, focusing on testing practice, offering all the necessary artifacts.

The presented analysis allowed us to identify which repository currently allow to partially meet the support of test teaching. In addition, we analyzed what types of techniques could be supported by the proposed repository. Although the development of the repository is in progress, several of the materials to support the teaching of software testing for different types of techniques have already been prepared.

This work in progress will be improved by carrying out empirical studies to validate the applicability of the repository in teaching software testing, in addition to collecting data on the opinion of teachers and students regarding the materials available and their form of presentation. Also, we intend to continue in the production of materials, allowing the collaboration of teachers and professionals. Furthermore, this ongoing work aims to incorporate current software development scenarios, in which systems with contemporary software development procedures are considered (e.g. mobile applications, chatbots, systems with artificial intelligence, among others). Additionally, we intend to approach the artifacts development from a pedagogical perspective, since the artifacts can be viewed as didactic objects to support the Software Testing teaching. Finally, we intend to support the teaching of current technologies used in the execution of tests, such as automation tools for various platforms. We hope that this research will contribute to improving the quality of teaching and training practitioners with an interest in the software testing area.

REFERENCES

[1] Ellen Francine Barbosa, José Carlos Maldonado, Auri Marcelo Rizzo Vincenzi, Márcio Eduardo Delamaro, SRS Souza, and Mario Jino. 2000. Introdução ao teste de software. *Minicurso apresentado no XIV Simpósio Brasileiro de Engenharia de Software (SBES 2000)* (2000).

[2] Draylson Micael de Souza, José Carlos Maldonado, and Ellen Francine Barbosa. 2012. Aspectos de Desenvolvimento e Evolução de um Ambiente de Apoio ao Ensino de Programação e Teste de Software. In *Brazilian Symposium on Computers in Education (Simpósio Brasileiro de Informática na Educação-SBIE)*, Vol. 23.

[3] Marcio Delamaro, Mario Jino, and Jose Maldonado. 2013. *Introdução ao teste de software.* Elsevier Brasil.

[4] Vahid Garousi, Austen Rainer, Per Lauvås Jr, and Andrea Arcuri. 2020. Software-testing education: A systematic literature mapping. *Journal of Systems and Software* 165 (2020), 110570.
[5] Neil B Harrison. 2010. Teaching software testing from two viewpoints. *J. Comput. Sci. Coll* 26, 2 (2010), 55–62.
[6] Joanna Smith, Joe Tessler, Elliot Kramer, and Calvin Lin. 2012. Using peer review to teach software testing. In *Proceedings of the ninth annual international conference on International computing education research*. 93–98.
[7] Pedro Valle, Ellen Francine Barbosa, and José Maldonado. 2015. Um mapeamento sistemático sobre ensino de teste de software. In *Brazilian Symposium on Computers in Education (Simpósio Brasileiro de Informática na Educação-SBIE)*, Vol. 26. 71.