## Research questions or hypothesis, studied in the papers

### Supplementary material for the Systematic Literature Review (SLR) on software-testing education

| Paper ID | RQs |
|---|---|
| [P8] | • RQ1: Does the integration of SEP-CyLE have a significant and quantitative impact on the programming and testing knowledge gained by the students?<br>• RQ2: Are SEP-CyLE testing-related assignments aligned with the needs and interests of the learners in understanding underlying programming concepts? |
| [P9] | • RQ1: Is it possible to take existing materials and tack on a TDD approach?<br>• RQ2: Is giving credit to tests the best way to teach TDD? Do students write more, higher quality tests if they get feedback through grades on tests?<br>• RQ3: Does the TDD approach affect the amount of time spent on projects, since students have to write test-code?<br>• RQ4: Does writing tests lead to higher quality code with respect to the number of acceptance tests passed?<br>• RQ5: If in-class examples are developed using a TDD approach, does it have a higher impact on students than those who do not see testing in class? |
| [P13] | • RQ1: The main question that needs to be answered is whether (1) investing in an expensive (25.000 euros) physical infrastructure really creates a substantial positive effect on ST learning.<br>• RQ2: A less critical question is whether one month offers enough time to overcome the non-technical background of CS students and really get focus on testing. |
| [P15] | • RQ1: What are the knowledge gaps in testing topics faced by graduates with respect to industry needs? |
| [P18] | • RQ1: Can unit testing improve the quality of human computer interaction projects?<br>• RQ2: When introducing unit testing, what additional steps must be taken to ensure a positive learning experience?<br>• RQ3: What potential for regression of students' unit testing model is possible, and how can that potential be mitigated? |
| [P20] | • Alternative hypothesis: post-test scores are significantly higher than pre-test scores on average. (Pre-test and post-test covered all the learning objectives of the course described in the study. The description includes the pedagogical approach taken.) |
| [P26] | • RQ1: Do unit-testing practices in CS1 assignments and labs really improve code quality?<br>• RQ2: Do CS1 students enjoy writing test cases?<br>• RQ3. Do unit-testing practices in CS1 enhance the student's learning process? |
| [P27] | • RQ1: Students' attitude toward accepting non-traditional educational module is more positive than toward accepting traditional one?<br>• RQ2: If subjects were given training using non-traditionally-produced educational module would behave more uniformly, in the sense of fault detection rate, than if they were given training using traditionally-produced module?<br>• H0: There is no difference in the fault detection rates uniformity of subjects given training using non-traditionally-produced module as compared to subjects given training using traditionally-produced module. |
| [P28] | • H1: Students who received test sets T1 and T2 will produce higher quality programs than students who received only the program specifications. |
| [P31] | • RQ1: Is peer testing more effective than individual testing for the construction of test cases?<br>• RQ2: Is peer testing more efficient than individual testing for the construction of test cases? |
| [P38] | • RQ1: How can (i) participation and (ii) performance in agile testing be measured? |
| [P40] | • RQ1: Which test quality measures actually assess how much of the expected behavior is checked by the tests?<br>• RQ2: What are the practical obstacles of using identified test quality measures in an educational setting?<br>• RQ3: How can we resolve the obstacles to apply the measures in classroom tools? |

| | |
|---|---|
| | • RQ4: Which approach is more appropriate for open-ended assignments?<br>• RQ5: What measure works better for close-ended assignments?<br>• RQ6: What combination of the approaches works well as a hybrid measure to separately evaluate tests of the assignments having variable amounts of design freedom? |
| [P41] | • RQ1: How many bugs does each team find? (In a software testing competition using Bug Catcher – a web-based system for running software testing competitions)<br>• RQ2: Do the students recommend this event for future students? (The software testing event using Bug Catcher)<br>• RQ3: Do the students report an increased interest in Computer Science? (After the event)<br>• RQ4: What are suggestions for improving the system? (Bug Catcher) |
| [P46] | • Hypothesis: Including software security testing techniques as part of the typical software testing exercises used in CS classrooms will expand students' programming toolset and make them better equipped to tackle programming tasks.<br>• RQ1: Were student submissions unique? (Students wrote both submissions (defence programs) and test cases (attack programs) for an assignment given in an introductory security class.)<br>• RQ2: Do multiple attacks benefit performance? (Did students acquire a better score if they submitted multiple attack submissions?)<br>• RQ3: What accounts for the difference between max and overall SAQ? (SAQ score: student attack quality as a student's overall ability to attack all monitors.)<br>• RQ4: Are attack/defense abilities correlated? |
| [P47] | • RQ1: Whether either checked coverage or object branch coverage is a better indicator of test suite quality than a number of alternative measures—that is, is either a more accurate predictor of a test suite's ability to detect faults? |
| [P51] | • RQ1: How to make writing tests more reasonable in the educational context? |
| [P63] | • RQ1: How good are student-written tests at finding real bugs?<br>• RQ2: How much variation is there in the software tests written by students? |
| [P64] | • RQ1: Does the Testing Game have good quality regarding motivation, user experience and learning, from students' point of view? (Testing Game: an educational game addressing the following topics: functional testing, structural testing and mutation testing.)<br>• RQ2: Does the Testing Game have good usability from the student's point of view? |
| [P66] | • RQ1: Is there any different in relative learning in the game higher than in the group that did not play?<br>• RQ2: Is the education game considered appropriate in terms of content relevancy, correctness, and degree of difficulty? Is the game considered engaging? |
| [P67] | • RQ1 (Testing Strategies): How did students test their software products?<br>• RQ2 (Enabling and Inhibiting Factors): What factors supported students in testing methodically and what factors hindered them?<br>• RQ3 (Testing Attitude): What did students think of testing methodically?<br>• RQ4 (Testing in the SWP process): How did students incorporate testing in their engineering process? |
| [P71] | • RQ1: Can the mutation testing criterion facilitate the learning process of novice students in programming courses?<br>• RQ2: What are the trade-offs and recommendations of using mutation testing to support the learning process in programming courses? |
| [P72] | • RQ1: Can ST knowledge help developers improve their programming skills in terms of delivering more reliable implementations?<br>• RQ2: Does ST knowledge impact on the effort invested by developers on their implementations?<br>• RQ3: Does ST knowledge impact on the complexity of the produced code? |
| [74] | • RQ1: What are the beneficial on-line services for successful testing course?<br>• RQ2: To what extent can a technically challenging CSE course be offered online? |
| [79] | • RQ1. Is there a significant difference between the students' performances under different testing techniques?<br>• RQ2. Does there exist a noticeable relationship between the tests results under different testing techniques? |

| | |
|---|---|
| | • RQ3. What is the importance of the programming background when applying different testing techniques?<br>• RQ4. What is the influence of the gender factor on success in software testing assessments?<br>• RQ5. How do various teaching strategies over the years affect the exam results in the software testing? |
| [80] | • H1: Students will rate importance of skills and their corresponding strengths with a positive correlation<br>• H2: Students will rate helpfulness of and their adherence to behaviors with a positive correlation<br>• H3: Students more likely to adhere to TDD principles will rate TDD's helpfulness more positively<br>• H4: Students with higher programming anxiety (according to WTAS) will adhere less to starting work early and to principles of TDD<br>• H5: Students with higher programming anxiety will rate Web-CAT as more helpful<br>• H6: Students with higher evaluation anxiety (according to BFNES) will rate Web-CAT as less helpful. |
| [83] | • RQ1: How do students engage with the game? (The Code Defenders game: Students compete over code under test by either introducing faults ("attacking") or by writing tests ("defending") to reveal these faults.)<br>• RQ2: Does student performance improve over time?<br>• RQ3: Does student engagement correlate with exam grades?<br>• RQ4: Do students appreciate using Code Defenders in class? |
| [84] | • RQ1: Which testing tools and technologies are most used in the industry?<br>• RQ2: What are the current issues related to testing in the industry?<br>• RQ3: How should the learning goals, teaching methods and evaluation methods in a software testing course constructively aligned with current industry practices? |
| [88] | • RQ1: Is the level of CS program exposure related to the quality of test cases generated with black-box and white-box methods by undergraduate and graduate students? |
| [89] | • RQ1: If the availability and knowledge of the use of code coverage tools positively impacts and increases students' propensity to improve the quality of their black-box test suites.<br>• RQ2: If an increase in code coverage during white-box testing results in an increase in the number of bugs students find during testing.<br>• RQ3: If students find WReSTT a useful learning resource for testing techniques and tools.<br>• RQ4: If students find that WReSTT supports collaborative learning. |
| [90] | • H1: The experimental group will have significantly greater average TMSM and average coverage than the control group. (TMSM: average test-methods-per-solution-method. The experimental group used a plugin for Web-CAT that provides adaptive feedback based on how well the student is adhering to incremental unit testing.)<br>• H2: The experimental group will have significantly greater project correctness and coverage scores than the control group.<br>• H3: The experimental group's average TMSM and average coverage will increase over time relative to the control group's average TMSM and average coverage trends.<br>• H4: Students' perceptions of the helpfulness of test-first and unit testing will have a positive correlation with their self-reported adherence to the same behaviors.<br>• H5: The experimental group will value the helpfulness of test- first and unit testing behaviors significantly higher than the control group.<br>• H6: The experimental group will score significantly lower on WTAS (project anxiety) scale relative to their BFNES (fear of negative evaluation) scale when compared to the control group.<br>• H7: The experimental group will respond more positively to following TDD in the future than the control group. |
| [93] | • RQ1: Can TDD be integrated into early programming courses with minimal effort on the part of instructors?<br>• RQ2: What effect does the grading of test-code have on students' tests?<br>• RQ3: What effects does TDD have on quality of code and productivity of students? |
| [95] | • H1: Written test cases based on pair programming increase the number of killed mutants.<br>• H2: Written test cases based on pair programming provide better code coverage. |

| | |
|---|---|
| [115] | • RQ1: What are the possible strengths and weaknesses of mutation analysis when compared to code coverage based metrics?<br>• RQ2: Can mutation analysis be used to give meaningful grading on student-provided test suites requested in programming assignments? |
| [P123] | • RQ1: Can POPT help students to obtain more correct implementations than traditional approach based on blind<br>• testing? (POPT: A Problem-Oriented Programming and Testing Approach for Novice Students)<br>• RQ2: Do students adopting POPT submit fewer versions than the ones using traditional approach?<br>• RQ3: Do POPT programmers spend more time to deliver the implementation than traditional programmers? |
| [125] | • RQ1: What common mistakes do students make when learning software testing?<br>• RQ2: Which software testing topics do students find hardest to learn?<br>• RQ3: Which teaching methods do students find most helpful? |
| [126] | • RQ1: Can we lead students towards the habit of writing tests in software projects using introductory programming exercises?<br>• RQ2: Can these exercises be implemented in a highly automated, yet student-centered manner? |
| [137] | • RQ1: Is the effectiveness in the detection of defects affected by the use of a CVE? (CVE: collaborative virtual environment) |
| [143] | • RQ1: Does the use of code coverage tools motivate students to improve their test suites during testing?<br>• RQ2: Do the results generated by the code coverage tools support the subsumes relation between branch coverage and statement coverage, i.e., does branch coverage subsume statement coverage?<br>• RQ3: Do students find WReSTT a useful learning resource for testing techniques and tools?<br>• RQ4: Do students find the features in WReSTT support collaborative learning? |
| [145] | • H1: Students who used WebIDE perform better on programming tasks than students who used traditional static labs.<br>• H2: Students who used Web-IDE spend more time on labs (because of the lock-step aspect) than students who used traditional static labs. |
| [157] | • H1: Through the use of a mutation testing game, students will be able to grasp all relevant mutation testing concepts while having fun, and in the end become better software developers and testers, who produce higher quality software. |
| [158] | • H1: The proposed tool (ProgTest) helps novice programmers to increase the quality of their programs and test suites. |
| [167] | • RQ1: Does the use of Coding Dojo methodology to teach TDD improve the code coverage of students when compared to solo programming?<br>• RQ2: Does the use of Coding Dojo methodology improve motivation and grow the interest in learning TDD when compared with solo programming? |
| [168] | • RQ1: Does TFD have any effect on the learning process?<br>• RQ2: Does it impact the way inexperienced students code?<br>• RQ3: Will this experience have long-lasting effects on students? |
| [182] | • RQ1: Can ST knowledge help developers improve their programming skills in terms of delivering more reliable implementations?<br>• RQ2: Does ST knowledge impact on the effort invested by developers on their implementations?<br>• RQ3: Does ST knowledge impact on the complexity of the produced code? |
| [190] | • H0: CorrectnessIT = CorrectnessST (Is the code correctness using instructor-provided test cases (IT) equal to that with student-written test cases (ST)?) |
| [204] | • H0: There would be no significant difference between the performances in terms of scores of students on the first programming assignment from 2001 and 2003. (To assess the effectiveness of Web-CAT and TDD.) |