Evaluating the impact of Software Testing Education through the Flipped Classroom Model in deriving Test Requirements

Leo Natan Paschoal
University of São Paulo
São Carlos, SP, Brazil
paschoalln@usp.br

Myke M. Oliveira
University of São Paulo
São Carlos, SP, Brazil
mykeoliveira@usp.br

Silvana M. Melo
Federal University of Grande Dourados
Dourados, MS, Brazil
silvanamelo@ufgd.edu.br

Ellen F. Barbosa
University of São Paulo
São Carlos, SP, Brazil
francine@icmc.usp.br

Simone R. S. Souza
University of São Paulo
São Carlos, SP, Brazil
srocio@icmc.usp.br

ABSTRACT

Recent studies have shown that software testing professionals, for the most part, conduct testing activity in a non-systematic way in software projects developed in the industry. In this sense, with the understanding that it is necessary to educate professionals capable of performing the test activity in a systematic way, the community of Software Engineering educators has paid attention to teaching test criteria. In particular, studies have been carried out in the context of different pedagogical models to recognize the models that manage to promote effective learning. In this perspective, research is being conducted on the flipped classroom pedagogical model. Although there are already experimental studies on this model, they did not investigate whether the student studying with the flipped classroom model can apply the test criteria effectively, deriving the test requirements satisfactorily. Our study emerges intending to analyze these issues. The study is configured using an experimental process that compares the test requirements derived by students who studied with the flipped classroom model, with test requirements derived by students who studied with the traditional teaching model. From the observed dependent variables (efficiency, efficacy, and effectiveness), it is possible to notice that students who study with the flipped classroom model derive test requirements that satisfy the equivalence partitioning criterion of the software under test with greater effectiveness compared to students who study with the traditional teaching model.

CCS CONCEPTS

• **Software and its engineering** → Software verification and validation;

KEYWORDS

Software Testing, Flipped Classroom Model, Computing Education

ACM Reference Format:
Leo Natan Paschoal, Myke M. Oliveira, Silvana M. Melo, Ellen F. Barbosa, and Simone R. S. Souza. 2020. Evaluating the impact of Software Testing Education through the Flipped Classroom Model in deriving Test Requirements. In *34th Brazilian Symposium on Software Engineering (SBES '20), October 21–23, 2020, Natal, Brazil.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3422392.3422489

SBES '20, October 21–23, 2020, Natal, Brazil

ACM ISBN 978-1-4503-8753-8/20/09…$15.00
https://doi.org/10.1145/3422392.3422489

1 INTRODUCTION

The failures produced by software during its execution are due to defects inserted by developers during the implementation of the software product [1]. The defects, in turn, comprise the incorrect commands that the programmer wrote [9]. The impact of software defects has gained repercussions, mainly because the generated failures can cause irreversible damage to the subjects and organizations directly involved or dependent on a given system [26, 30]. In this context, the software engineering community clearly understands that the software testing activity needs to be performed, given the possibility of identifying defects inserted during the software development process [9].

Although the testing activity is receiving attention from the software industry, the conduction of this activity within software development organizations still goes through a maturity process. The industry, despite performing the test activity, is still maintained by professionals without adequate qualifications in the area [13, 12]. From this perspective, the test activity ends up being performed randomly [10]. Therefore, test criteria[1] that could support testers in selecting test data[2] with quality are not used by industry professionals. Mostly, the professionals do not know the existing test criteria and this makes it difficult to perform the activity because they do not know when to stop testing and even to prioritize software functionality when it comes to carrying out the activity [10].

[1] Strategies that help define test input data, execution conditions, and expected outputs from condition execution [9].
[2] These are the input data from software and the conditions associated with a particular test case[9].

The skills and knowledge of professionals about software testing criteria and techniques[3] have been investigated in the literature. Recent studies showed that the percentage of professionals involved in testing who knows the activity accurately is very low [12, 13]. Few of these professionals obtained adequate academic training [18].In addition, software organizations generally do not offer formal training to their professionals [12, 19]. Usually, professionals can use a given time to learn more about the test activity, however, this does not guarantee that professionals will be studying how to establish a set of test cases in a systematic way.

Organizations have hired professionals without adequate knowledge about the test activity because students of Computer Science courses graduate without knowing how to test software [17]. This understanding is not new, but it is still a valid argument in the literature that has been discussed about the need to invest in the academic education of computing students. Computing students are expected to incorporate software testing concepts and practices through the development of initiatives. Efforts have been made in the past decade, even though the efforts made so far have not produced significant results. In this context, recent studies have discussed the need to change the strategies of how software testing content should be taught [3, 24].

Commonly, the teaching of software testing has occurred through the traditional teaching model [24]. In this model, students spend most of the in-class period watching lectures presented by the professor [2]. This scenario promotes demotivation on the part of the students concerning the content, produces results different from what is expected in terms of learning gain, and makes it difficult to carry out practical activities with professor supervision, which are crucial for the development of software testing skills [23]. Therefore, it is necessary to rethink the pedagogical models and practices used in the classroom. In this segment, some studies have emerged with the intention of recognizing pedagogical models with the potential to contribute to the learning of software testing [20, 21]. In particular, we also found studies that designed experimental studies to understand the feasibility of the flipped classroom pedagogical model [20, 21].

The flipped classroom model has been investigated in the context of teaching software testing because it gives priority to the moments when the student can perform active exercises with the help of the professor (e.g., laboratory activities, resolution of exercises and projects closer to the "real world", among others), given that students' doubts tend to arise when they are involved with the application of concepts in practice [2]. Thus, in this model, passive activities (i.e., readings and lectures given by the professor), which do not need as much supervision from the professor when compared to the performance of active exercises, are carried out outside the classroom [2].

We identified three studies in the literature on the use of the flipped classroom pedagogical model in teaching software testing. Two studies sought to analyze the viability of the model through controlled experiments [20, 21] and one study described an approach that aims to support the implementation of this model in testing or software engineering courses [23]. Although the carried out experimental studies showed positive results from the pedagogical point of view, they did not analyze aspects of software testing practice, such as the students' ability to recognize the test requirements that are capable of satisfying a given test criterion, considering a system under test (SUT). The test requirements characterize 'parts of the software' that will be exercised during the execution of the software and are derived by testing criteria [9].

Considering that the identification of the test requirements is a stage of the software testing activity, and the flipped classroom model is intended to promote the acquisition of technical skills during the lesson, to obtain an improved understanding of the model's impact on the development of practical testing skills is important. In particular, when we assume the particularities and assumptions of the flipped classroom model as well as the characteristics of the application domain (i.e., software testing), it is extremely relevant to understand whether students who study software testing with the flipped classroom model can identify test requirements of a SUT more effectively than students who study with the traditional model.

Our study emerges intending to present an experimental study that was conducted to analyze the students' effectiveness in identifying the test requirements. We compared the test requirements produced by students who studied with the flipped classroom model, with the test requirements produced by students who studied with the traditional teaching model. In our study, we taught students the functional testing technique[4] and the equivalence partitioning criterion[5]. Therefore, the test requirements analyzed aim to meet this test criterion, considering a given SUT. We justified the choice of this technique and criterion, given that: (i) the equivalence partitioning criterion is generally taught to computer students in the testing or software engineering course [22]; (ii) the experiments that precede this study also addressed this criterion providing comparable results [20, 21].

To present this study, we organized it as follows. Section 2 provides an overview of the works related to ours. In Section 3, we present the planning and describe the execution of the experimental study. In Section 4, we report the results obtained through the experiment. The threats to validity are presented throughout Section 5. In Section 6, we addressed some limitations of this study. Finally, Section 7 presents our final considerations and future work.

2 RELATED WORKS

The software testing education has been discussed in research papers through pedagogical approaches that are different from the traditional teaching model. There are studies exploring collaborative learning [4], game-based learning [5], case-based learning [27], among others. Considering that the focus of this study is the understanding of the flipped classroom model, this section aims to address previous studies that explored this pedagogical model in the theme of software testing. In this perspective, it is important to make it clear that the adoption of the flipped classroom pedagogical

[3]Testing techniques are composed of testing criteria and are characterized by directing how test cases will be defined, considering the source of the information[9].

[4]A software testing technique that uses information from the program specification to generate test cases.

[5]It is a test criterion that divides the set of values that can be used to run a given program into equivalence classes (valid and invalid ones), based on the program's entry conditions [9].

model in software testing courses, or even in software engineering, has been studied from different perspectives.

In 2005, even before the flipped classroom term was widespread, Kaner and Fiedler [15] reported a proposal to restructure a software testing course. This restructuring involved recording the lectures made by professors about the test content and making these recordings available through a website for students to watch them outside of class. In this way, the classroom meetings are aimed at developing a project, in which students get involved with development teams and apply the acquired knowledge. The development of the project is guided by the professor who will use the class meetings to supervise students, promote discussions, and listen to presentations made by students. The report prepared by Kaner and Fiedler did not focus on understanding the benefits or even the limitations of this pedagogical model.

Gannod et al. [11] also presented a proposal to use the model currently known as the flipped classroom. In particular, arguments are presented that aim to motivate the use of this model in computing courses. Regarding the software testing content, the authors described that face-to-face meetings can be dedicated exclusively to applications of software testing techniques by students in open source projects. According to the authors, this can provide students with the acquisition of experiences. Besides presenting suggestions for the use in different courses, the authors reported that they used the model only in software engineering content at that time (in teaching service-oriented architectures). Based on the teaching of this content, Gannod et al. presented feedback on students' acceptance of the model. In this feedback, the pedagogical model seems to have been well received by students, but acceptance was not considered unanimous.

In the study by Wu et al. [29], the authors addressed a project that aims to balance the knowledge acquired by students of software engineering with practical experience. In particular, the project involved the development of educational tools for teaching software verification and validation. They aimed to promote balance based on assumptions that characterize active activities. In addition to addressing how active activities were established, the authors reported that these activities and materials can be used in courses that follow the flipped classroom model.

In Paschoal et al. [21] a controlled experiment is presented intending to understand if the students who study with the flipped classroom are more motivated to learn software test content than the students who study with the traditional model. Also, the authors verified whether the students performance is better in practical or theoretical assessments. Based on the study, it was found that there was no difference in terms of motivation. However, there was a small difference in student's performance, which tended to be greater in students who studied with the flipped classroom model.

There was a second study reported by Paschoal and Souza [23], in which a preliminary approach was established to support the professor to use the flipped classroom model in teaching software testing. This approach included learning objectives, educational resources, activities that aim to promote active learning, and an artifact to support the implementation of the approach. Also, the approach has been defined for main software testing techniques.

Finally, another experimental study was conducted by Paschoal et al. [20], which intended to analyze aspects of learning gain and instruction time spent by students who studied with the flipped classroom model. Based on this study, the authors found that the students acquired more knowledge when learning software testing with the flipped classroom, but spent more time to study the content than the students who studied with the traditional teaching model.

3 STUDY SETUP

In this section, we reported the planning and conduction of the experimental study. The decisions and the process we followed were based on the experimental process described by Wohlin et al. [28].

3.1 Goal definition

This experimental study aims to analyze the flipped classroom model, for the purpose of verifying its impact when used to teach students to recognize the test requirements of a SUT to satisfy a certain test criterion, with respect to effectiveness, from the point of view of the researcher, in the context of software engineering students who are studying the software testing content.

3.2 Context selection

This experimental study held in April 2018, with students from the 5th semester of the computer science bachelor's degree program, which is offered by the Institute of Mathematics and Computer Sciences at the University of São Paulo. The experiment was performed after teaching the introductory content of software testing (i.e., the definition of software testing, its concepts and terminologies, and an overview of the techniques and sources of information used to conduct the test activity).

3.3 Hypotheses formulation

This experiment addresses a specific research question that has not yet been investigated in the literature of the area: "Are the students more effective in recognizing the test requirements of a SUT when studying with the flipped classroom instead of the traditional teaching model?". In seeking to answer this question, we formulated the following hypotheses:

- *Null Hypothesis:* there is no difference in terms of effectiveness concerning the identification of the test requirements made by the students who study with the flipped classroom model, and by the students who study with the traditional teaching model.
- *Alternative Hypothesis:* there is a difference in terms of effectiveness concerning the identification of the test requirements made by the students who study with the flipped classroom model, and by the students who study with the traditional teaching model.

3.4 Selection of Subjects

The students chosen to participate in the experiment were selected for convenience. We invited 93 students from the software engineering course of the Computer Science undergraduate program to participate in this experiment, which came from two classes.

Given the selection of subjects, we defined that one class would have the experience of studying the equivalence partitioning criterion content with the flipped classroom model, while the other one would study the same content with the traditional model.

3.5 Variable selection

The experiment investigated the values of a particular variable, the effectiveness. To understand the variable's values, we took some actions to control the factors that influence, directly or indirectly, this variable. In the following, we describe the variables that represent the influences that can affect the experiment results (dependent variables) and the variables that accommodate the values of the experiment results (independent variables).

Dependent Variables. Effectiveness is a variable that is defined by Cury [8] as the achievement of efficiency and efficacy at the same time by the subject during the performance of a certain activity. Thus, in order to recognize the values assumed by the effectiveness variable at the end of the experiment, we considered:

- *Efficiency:* the efficiency refers to the time that participants took to carry out the proposed activity. In other words, the time "spent" by students during the recognition of the elements required to satisfy the equivalence partitioning criterion.
- *Efficacy:* the efficacy refers to the number of test requirements that satisfy the equivalence partitioning criterion for the defined software project, which was identified by the participants. We emphasize that an element is only considered as correct if the participant identified it correctly and appropriately. To verify if the participant identified the test requirements correctly and appropriately, a test oracle was previously defined.
- *Effectiveness:* the effectiveness is a combination of efficiency and efficacy results.

Independent Variables. In this experiment, we consider the following input variables:

- *Pedagogical models:* refer to the pedagogical models used to instruct students during the teaching of the equivalence partitioning criterion. In this experiment, we investigated two treatments, namely:
  Treatment A: students are instructed using the flipped classroom model;
  Treatment B: students are instructed using the traditional teaching model.

3.6 Material

To support the execution of the experimental study, some materials were recovered from the experiments by Paschoal et al. [20] and Paschoal et al. [21] and additional necessary material were produced. The teaching materials we used to instruct, both the students in the control group and the students in the experimental group, were reused from the experiment carried out by Paschoal et al. [20] since they had been carefully selected and were positively appreciated by the students.

For the traditional class, we reused the didactic material used during the traditional classes of the experiment carried out by Paschoal et al. [21]. The learning management system (LMS) Moodle, in turn, was used in this experiment to support the publication of study materials for students in the experimental group. In this sense, we created groups within the environment, aiming to prevent students who should not have access to the study material, from gaining access to it.

We defined a specification for a software function that would later be used by the students during an activity in which they should recognize the test requirements necessary to satisfy the equivalence partitioning criterion. For that, we defined a specification on the function called "NextDate". This function returns a date (day/month/year) based on an entry date provided by the user. After the definition, we established an oracle that contains the minimum number of test requirements that needed to be identified by the students.

The activity for students was stipulated from the definition of the specification in which the students would have to analyze. Initially, the students would need to annotate the time spent on each step (reading the specification, recognizing test requirements). The activity consisted of reading the specification and defining the test requirements for the software specification.

Finally, a consent form was developed to clarify the research subjects about the purpose of the study and how it impacts society. The term contains information about the duties of the participants and the obligations of the researchers.

3.7 Experiment Planning

Our first action was to plan the number of classes that would be necessary to carry out the experimental study with the help of the professor. In this sense, we stipulated that the experiment would be performed as a regular activity of the course. Thus, all of the students would participate in the study, but they could choose to make the data available for analysis or not. A similar action was used in the experiment by Paschoal et al. [20] since the study could not hinder the progress of the course, and there would be no other opportunities in the course for the content to be resumed.

As the software engineering course was given to two different classes, we defined that each class would be taught with one of the pedagogical models. Thus, we defined that a class made up of 42 students would be taught by the traditional teaching model, while the other one made up of 51 students would be taught by the flipped classroom model. This definition led to the predetermination of the number of classes that would be required for each class.

For the group that would be taught with the traditional model, we foresaw an in-class for teaching the criterion and another one for conducting the activity. On the other hand, for the group that would study with the flipped classroom model, we foresaw an in-class period that precedes the doubt-solving class. This period was established so we could explain to the students that some teaching materials (instructional videos) had been made available and that they should study the content. Thus, they could get prepared for the in-class moment that would consist of solving questions and activities. In the flipped classroom model, we also foresaw two in-class periods, one so that students could solve doubts and activities for the application of the criterion, and another for the performance of activities.

After defining the instrumentation, all materials were prepared and revised. The consent form was prepared to be applied, as well as the LMS Moodle. In this case, as there were two different classes, we registered two undergraduate programs in the Moodle. We emphasize that students in one class did not interact with students in the other class. Thus, each class had a unique space in Moodle to access the teaching material.

We prepared a traditional in-class period for students in the control group, in which the professor taught the content throughout the class period (approximately 110 minutes). In this class period, the students would watch the lectures given by the professor, would solve their doubts about the lectures, and they would have time to solve some simple exercises that would be made available by the professor at the end of the set of slides. These exercises could be completed after class.

We also prepared an in-class period with the flipped classroom model, based on the recommendations for implementing the model described by Bergmann and Sams [2]. During the class period, students would have time to solve their doubts and perform some exercises to apply the criterion, under the supervision of the professor and a post-graduate student who was a teaching assistant.

3.8 Experiment Execution

As we described in the previous section, this experiment was carried out with two classes in the software engineering course. Next, we present all the activities carried out in each of the classes, throughout the experiment:

Class A – Experimental Group. The experimental group performed activities in four distinct stages:

Stage 1: At the end of a traditional in-class period in the software engineering course, in which software testing concepts and terminologies had been taught, the course professor explained that she would take a different approach when teaching the equivalence partitioning criterion in the course. The professor explained that she would use a pedagogical model that provided the moment available in the classroom to solve doubts about the theoretical content, teaching assistant student activities, and solve doubts that may arise during the activity. To do so, students would have to study the content preliminarily before class.
In this stage, before releasing students from the traditional class, they were instructed to access the didactic material available on LMS Moodle. Students were also notified that they could watch the videos as many times as they think they needed, at the speed they preferred.

Stage 2: The second stage comprises the moment when students should study the content to prepare for the lesson. Considering that the students were instructed in the first stage on Friday and the next class of the course would be on Monday, students would have a weekend (two days) to study the content.

Stage 3: In the class on Monday, the professor asked the students to make questions about the content. It was emphasized that at that moment could solve doubts that could have arisen during the study in Stage 2. The moment that dedicated to solving doubts lasted approximately 30 minutes. Subsequently, some exercises on the application of the equivalence partitioning criterion were made available to students. These exercises should be performed by the students during the class period and the professor was available to solve doubts about the resolution of the exercise. At the end of the activity, the students were released.

Stage 4: In the class on Friday, students were notified that they would carry out an individual activity, without support from the professor and colleagues, and that they would not be able to consult the course material. On the occasion, students were challenged to carry out the activity on the identification of test requirements. After solving the activity, students were notified that the initiative was part of an experimental study. Thus, they were introduced to the objective of the experiment and were able to inform whether they would accept to make available the data collected for the analysis of the experiment.

Class B – Control Group. The control group participated in activities in two distinct stages:

Stage 1: On Tuesday, the professor gave a class in the traditional model, in which most of the class period was held by lectures supported by slides and examples of applications. At the end of the lectures, the professor presented some activities that should be performed by the students, which could be done at home, "after class".

Stage 2: After exactly three days, at the beginning of the class, students were notified that they should carry out an individual activity and without accessing any type of consultation material. The activity of identifying the test requirements was applied to students in this group. At the end of the activity, students were notified that an experimental study was being carried out in the context of the course and that each student could decide whether or not to make their data available.

4 DATA ANALYSIS AND RESULTS

After executing the experiment, we analyzed the data associated with the activities. To support the analysis of the results, we used descriptive and inferential statistics. This section is intended to present these analyzes.

4.1 Characterization of the subjects

From the execution of the experiment, it was planned that the 93 students declared their interest in making the data available for analysis or not. Many students chose not to make it available, as they did not feel comfortable being evaluated. This caused a reduction in the number of students per group. To contribute to reducing the number of students, not all of them participated in all the activities of the experiment, and some missed the classes in which the experiment was carried out. Therefore, the final analysis was attended by 79 students, who participated in all activities and agreed to make their data available.

Of the 79 subjects, 35 (44%) studied the criterion through classes based on the traditional teaching model, which were part of the control group. The rest of the subjects (equivalent to 56%) studied with an initiative that implemented the flipped classroom model. The

subjects who participated in the study with the flipped classroom model were described as members of the experimental group.

4.2 Effectiveness recognition

As described in Section 3, effectiveness can only be observed from the analysis of efficiency and efficacy. Efficiency can be calculated through the time the student "spends" during the recognition of the test requirements. Thus, we performed a calculation to recognized the efficiency that was based on the following equation:

$$Efficiency_{(i)} = (t_{MAX} - t_{(i)})/t_{MAX}$$

where:
t = represents the total time spent by the student during the activity;
i = represents the student, i.e., {student $_1$, student $_2$, student $_3$, ..., student $_n$};
t_{MAX} = represents the maximum time spent by the whole group of students during the activity accomplishment.

The equation we adopted for efficiency was the same that Souza [25] used in his study. With the definitions of t and t_{MAX}, if the participant spent 100 minutes during the identification of the test requirements, the maximum time would be allocated to t. Thus, a participant who spent 50 minutes receives an efficiency value of 0.5, that is, 50% of efficiency in that activity. On the other hand, the student who spent the most time in the group receives an efficiency value of 0.0, that is, 0% of efficiency in the activity.

Regarding efficacy, we considered the number of test requirements that we defined in a test oracle and the test requirements that were identified by the participants. Based on this information, these data were applied in an equation that calculates the participant's efficacy in recognizing the test requirements.

$$Efficacy_{(i)} = n_{(i)}/n_{Quantity}$$

where:
n = represents the number of test requirements that were correctly identified, that is, that were before identified in the oracle and were recognized;
i = represents the student, i.e., {student $_1$, student $_2$, student $_3$, ..., student $_n$};
$n_{Quantity}$ = represents the number of test requirements provided by the test oracle.

This equation was also based on the calculation to identify the student's efficacy, used in the study by Souza [25]. Thus, the student who was able to identify all the test requirements by the criterion that were foreseen by the oracle received an efficacy value of "1.0", that is, 100% efficacy. In contrast, if the student was unable to correctly identify any of the test requirements, he received an efficacy value of 0.0, that is, 0% efficacy.

From this, based on the study by Souza [25], we used the following equation to calculate the effectiveness:

$$Effectiveness_{(i)} = Efficiency_{(i)} * Efficacy_{(i)}$$

where:
i = represents the student, i.e., {student $_1$, student $_2$, student $_3$, ..., student $_n$};
Efficiency = represents the student's efficiency in identifying the test requirements to satisfy the criterion;
Efficacy = represents the student's efficacy in identifying the test requirements to satisfy the criterion.

After calculations to identify the efficiency, efficacy, and effectiveness of each student, we analyzed the results in order to compare them between groups. In the following, we present an analysis that was based on measurements of position and measurements of dispersion.

4.3 Efficiency analysis

Table 1 shows the main results we obtained regarding efficiency. The results were opposed based on measurements of position, more specifically the minimum, maximum, mean, and median. We also calculated the sample standard deviation as measurements of dispersion. Observing the means, we noticed that the group that studied with the flipped classroom model obtained better results, but not significantly different from the control group. The other values related to the measurements of position also reveal proximity. However, the sample standard deviation associated with this variable reveals that the experimental group showed a greater variation concerning the efficiency of students in the control group. This indicates that some students were more efficient at identifying the test requirements while other students were less efficient.

Table 1: Descriptive statistics of efficiency values

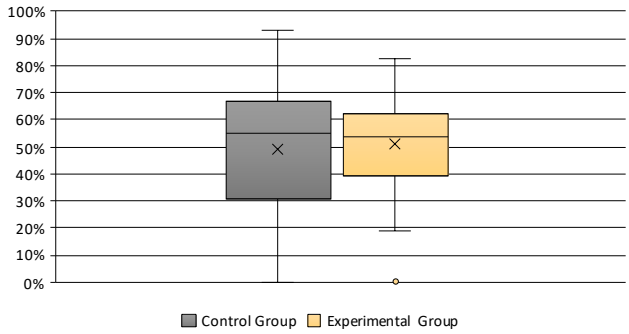
	Mean	SD	Min	Max	Median
CG	49%	25%	0%	93%	55%
EG	51%	17%	0%	83%	54%

Legend: CG - Control Group; EG - Experimental Group

To complement the analysis, we generated a box plot type chart. In Figure 1 it is possible to observe, in a summarized form, the values of students' efficiency in recognizing the test requirements of a software project to satisfy the test criterion. Regarding efficiency, the graph reveals a greater amplitude in the data of the control group than in the data of the group of students who studied with the flipped classroom model. It was also possible to observe that the experimental group obtained an efficiency value considered as an outlier concerning the other members.

After the descriptive analysis of the data, we carried out inference tests to find out if students who study with the flipped classroom model were more efficient in recognizing the test requirements of a SUT than students who study with the traditional teaching model. For this purpose, we initially established a confidence interval for α= 0.05. The normality of the data, in turn, was verified using the Shapiro-Wilk test. From this test, we identified that the distribution of data related to efficiency was organized in a Gaussian way, since the p-value for efficiency of the experimental group was equal to 0.2537 (i.e., p-value > α) and the p-value for efficiency of the control group was 0.3186 (i.e., p-value > α). From this, the Student's t-test was applied, revealing that there is no significant difference between the efficiency of the control group and the efficiency of the experimental group when identifying the test requirements, given that p-value = 0.2990.

Figure 1: Box plots with efficiency data



4.4 Efficacy analysis

To analyze efficacy, initially, we also look at position and dispersion measurements. As can be seen in Table 2, the efficacy values of the experimental group are higher than the efficacy values of the control group. The measures of position reveal that, on average, efficacy was twice as high in the experimental group. It is possible to observe that 50% of the efficacy values of the control group were less than 7%, while 50% of the efficacy values of the experimental group were greater than 54%. Observing the minimum efficacy values, we noticed that some students in the control group were unable to correctly identify the test requirements to satisfy the criterion.

Table 2: Descriptive statistics of efficacy values

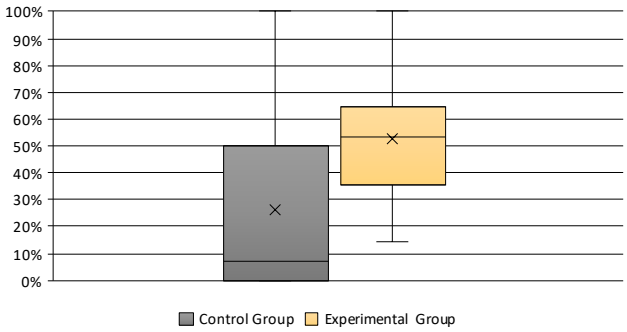
	Mean	SD	Min	Max	Median
CG	26%	30%	0%	100%	7%
EG	53%	22%	14%	100%	54%

Legend: CG - Control Group; EG - Experimental Group

We also generated a box plot chart with the efficacy data (See Figure 2). From this, we observed that the measure of the central tendency of the experimental group exceeds the value corresponding to the third quartile of the control group. Note the existence of asymmetric values in both groups. While the control group showed a positive asymmetry, the experimental group showed a negative one. Thus, the box plot corresponding to the control group indicates that the values that are more repeated are lower than the median, which in turn is lower than the mean efficacy. On the other hand, the values that characterize the measures of position of the experimental group efficacy revealed that the mean value is lower than the median, which in turn is lower than the values that are more repeated in the group (i.e., mode).

We also conducted an inferential test in the context of the efficacy analysis. We used the Shapiro-Wilk test to verify the distribution of the data, considering the same confidence interval used in the efficiency analysis (i.e., α= 0.05). From the Shapiro-Wilk test, we noticed that the data do not follow a normal distribution. The p-value for the control group was equivalent to 0.0087 (i.e., p-value < α), while the p-value for the experimental group was 0.0100 (i.e., p-value < α). Thus, we used the Mann-Whitney test. Through this test, we noticed that there is a significant difference in effectiveness between the groups, since the p-value was equal to 0.0001.

Figure 2: Box plots with efficacy data



4.5 Effectiveness analysis

After a comparative analysis between the two variables that characterize effectiveness, considering measurements of position and dispersion, we observed that the experimental group showed greater effectiveness in carrying out the activity to which they were exposed. The student who presented the best result for effectiveness was part of the experimental group. The median values, shown in Table 3, reveal a wide difference between the effectiveness of the experimental group and the control group, since 50% of the students who studied with the flipped classroom model had effectiveness greater than 28%, while at least 50% of students in the control group achieved effectiveness equivalent to 0%.

Because some students presented 0% efficiency or 0% efficacy, some students in the control group and also in the experimental group obtained 0% effectiveness. As a result, the values of the groups varied widely. Despite the wide variation, the control group showed a greater variation in the results of effectiveness compared to the experimental group.

Table 3: Descriptive statistics of effectiveness values

	Mean	SD	Min	Max	Median
CG	11%	16%	0%	57%	0%
EG	27%	14%	0%	63%	28%

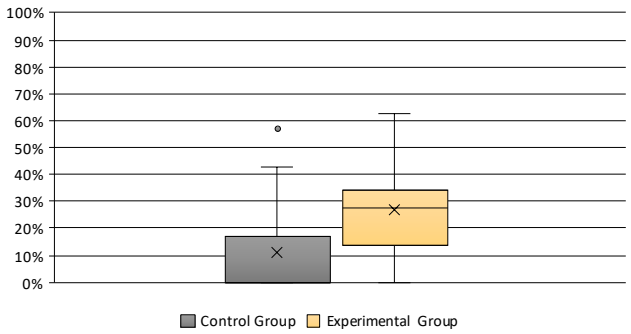
Legend: CG - Control Group; EG - Experimental Group

We also generated and analyzed box plot charts with data on effectiveness (See Figure 3). As it is possible to observe, the groups present different mean levels of effectiveness, in which the experimental group presented effectiveness superior to the control group. In this perspective, the measure of the central tendency of the control group is lower than that of the experimental group. Besides, it is also possible to observe that there is a presence of an outlier, this time in the data of the control group.

Finally, to better understand the effectiveness, we applied the hypothesis test. The Shapiro-Wilk test revealed that the effectiveness data that correspond to the control group does not follow

a Gaussian distribution, since the p-value is equal to 0.0081 (i.e., p-value < α). In contrast, the effectiveness data that comprise the experimental group has a Gaussian distribution since the p-value is equal to 0.0693 (i.e., p-value > α). Thus, we used the Mann-Whitney test to perform the inferential test. Based on the results (i.e., p-value = 0.000001), the null hypothesis was rejected, indicating that there is a difference in terms of effectiveness concerning the identification of the test requirements made by students who study with the flipped classroom model and by students who study with the traditional model.

Figure 3: Box plots with effectiveness data



As we identified a statistically significant difference in the effectiveness variable, it was necessary to evaluate the meaning of the results on the differences found between the effectiveness values in the groups. Thus, to calculate the effect size we used the Cohen's *(d)* test. As a result, we obtained a value of *(d)* = 1,02. Considering the argument of Cohen [6], this means that the size of the effect is large, that is, the statistical difference is effectively significant. To conduct the calculation, we used the following equation:

$$d = \frac{\overline{x_1} - \overline{x_2}}{\sqrt{\frac{(n_1-1)s_1^2+(n_2-1)s_2^2}{n_1+n_2}}}$$

where:
$\overline{x_1}$ and $\overline{x_2}$ represent the means of the experimental group and the control group;
s_1^2 and s_2^2 represent the sample variances of the experimental group and the control group;
n_1 and n_2 represent the sample sizes of the experimental group and the control group.

4.6 Synthesis

Effectiveness is a variable that is interconnected with the variables of efficiency and efficacy. Thus, to better understand what had been established in the research question during its definition, these other variables were considered. Regarding efficiency, descriptive statistics indicated higher rates for the group of students who studied with the flipped classroom model. Despite this, the values of the control group were close to those of the experimental group. With the Student's t-test, it was possible to infer that the efficiency in identifying the test requirements was equivalent between groups.

On the other hand, the descriptive analysis on efficacy indicated that students who study with the flipped classroom model performed with more efficacy the recognition of the test requirements than students who study with the traditional model. This value reflects how well the students performed the activity. The hypothesis test proved that there is a significant difference in the efficacy values between the groups.

Finally, to understand whether students recognize the test requirements in the way that they should be identified, reaching the objectives set, and using the available time in the best possible way, effectiveness was observed. The descriptive analysis indicated that the experimental group had higher effectiveness rates than the control group, indicating that they are more efficient and effective at the same time. The hypothesis test in conjunction with the Cohen's *(d)* test revealed that there is indeed a significant difference in terms of effectiveness between the groups. Therefore, from the experimental study, we can conclude that students who study with the flipped classroom model are more effective in recognizing the test requirements of a SUT than students who study with the traditional teaching model.

5 THREATS TO VALIDITY

Every controlled experiment is subject to threats to validity. In this sense, during the planning and execution of our experiment, we raised threats that could invalidate the study. We organized these threats into four categories that were defined by Cook and Campbell [7]. In addition to presenting the threats, we discuss strategies we used to mitigate them.

Conclusion validity: The conclusions were obtained through the results, which were analyzed through statistical tests. These tests, if defined and used incorrectly, can produce erroneous conclusions. Thus, to mitigate a possible threat to the results, in addition to the descriptive analysis of the data, we carried out the hypothesis test. Additionally, to mitigate possible threats in the definition of the appropriate test, we analyzed the data normalization.

Internal validity: The conclusions about the causes and their effects in the study can be influenced by factors that are not predicted in the study. Thus, the effects caused by the variables analyzed (efficiency, efficacy, effectiveness) may not be the result of defined factors (with different treatments) but attributed to other causes, such as the software project selected for the experiment or the defined criterion. In order to mitigate the threats that permeate the experiment, we took some measures.

The software specification was selected because it depicts a particular function that receives input data that can be classified into different subdomains. As a result, this project has been used by professors at different universities worldwide, as part of specific books in the area [14]. The function mentioned in the project specification is also used in experimental studies in the area [16].

Regarding the criterion used in the experiment, it was defined as being relatively simple [23], it is a criterion that is taught by professors who teach software engineering and/or software testing courses in Computer courses in Brazil [22],

in addition to offering conditions for a subset of data to be defined [9].

Construct validity: The instruments used in the experiment can offer threats to the validity of the study. In this perspective, the data related to the time spent by the student to identify the test requirements, as well as the test requirements that were identified by the student reflect on a threat associated with the bias of a single method. We defined only one activity for students to perform (identify the test requirements and annotate the time). It is possible that the use of more than one activity, as was the case in the study by Souza [25], may offer more accurate results. Because the experiment was carried out in a course with a defined schedule and limited deadlines, it was not possible to carry out more than one activity.

In the study, we also used only a single criterion. The use of different criteria may offer other results for the analyzed variables. Despite this, the criterion is directed to the software project that was defined and used during the execution.

External validity: This experiment was planned and executed to enable and stimulate its replication to compare different criteria and test techniques. However, some elements may offer threats to the generalization of the data obtained for another context. The software design used and the defined criterion, for example, may not be representative of all types of activities performed in real teaching settings. Despite this, they were defined cautiously, aiming at making it possible to generalize the results. Thus, the generalization of results for students in learning situations similar to those used in the context of the experiment is not a threat to this study.

6 LIMITATIONS OF THIS STUDY

Although the controlled study followed an experimental process widely recognized in the literature, which provides for the mitigation of possible threats to validity, our work was not free of restrictions. These restrictions are characterized as limitations of the study, which do not threaten its validity. Below, we list the limitations that we recognize and that persist in this study:

- Test requirements: The software test activity involves different steps. One of these steps is the identification of the test requirements. In this study, we observed the results of the performance of this activity by software engineering students, who studied with the pedagogical models. Therefore, we did not analyze the set of test cases produced by these students. An analysis of test coverage that considers the set of cases produced by students can be interesting to obtain a more accurate understanding of the skills that students developed during the study of the software test with the different pedagogical models. We emphasize that an experiment that involves this type of analysis will involve other variables such as: (i) students' knowledge in a programming language, as test cases need to be coded; (ii) the appropriate knowledge of the student to handle a given test tool because the student will have to learn how to use a tool that will enable the execution of test cases. Also, it is possible that other variables have to be observed or controlled.
- Test criterion: The test requirements derived by the students during the experiment were based on the equivalence partitioning criterion. Therefore, we cannot generalize that similar results can be found if another test criterion is used. In this perspective, if another subject is taught (e.g., control flow based criteria [9]), it is possible that the effectiveness result differs from that we found in this study. Thus, experiments that include other criteria and software testing techniques are required to check whether the results produced by the model, regarding the effectiveness of students in recognizing test requirements, are independent of the test criteria the students have learned.
- Study participants profile: Our experiment was performed in the context of students in a Computer Science undergraduate program. Most of these students are fully dedicated to the undergraduate program and, therefore, do not engage in professional activities, only in research and university extension activities. In this sense, our study has the limitation of not understanding whether the same effects produced by the experiment configuration will be achieved if the student profile changes. Thus, we believe that it is extremely valid to carry out a replication of this study in the context of students who do not have an exclusive dedication to the course (e.g., students who study at night and work during the day). This type of understanding is important because the flipped classroom promotes an increase in the instruction time of the student who studies software testing [20] and if the student does not have an exclusive dedication to the course, he may struggle to follow the classes, because he was not adequately prepared for the face-to-face classes.

Given these limitations, we prepared a laboratory package intending to enable new studies to face the limitations of this work as opportunities for further studies. The laboratory package we developed was registered in the Zenodo repository, under a creative commons license, and can be accessed by the DOI number, available at[6]: <http://doi.org/10.5281/zenodo.3997923>. We included all the materials we used in the experiment, the data collected and the analyzes performed.

7 CONCLUSIONS

In this study, we reported the planning, execution, results and data analysis obtained from a controlled experiment that we carried out with the intention of gaining a better understanding of the impact of the flipped classroom model in teaching software testing. In particular, we focused our attention on students' effectiveness in identifying test requirements after studying with the flipped classroom model. We take into account previous studies in the field to define our contribution and use a real teaching context to carry out the experimental study. Therefore, we compared the results produced by students who have learned software testing with the flipped classroom, with the results generated by students who learn the same subject with the traditional teaching model. The implementation of each model was also based on the literature. In particular, we instantiated all the pedagogical moments of the flipped classroom, which were reported in the study by Bergmann

[6] In Portuguese.

and Sams [2]. The traditional teaching model was also configured to characterize the traditional classes, considering the definition of Bergmann and Sams [2]. From this, with the consolidation of the experiment results, we can infer that the student who studies software test content with the flipped classroom model recognizes the test requirements more effectively than the student who studies with the traditional teaching model, in terms of the number of test requirements identified by the students (efficacy) and the time spent in this activity (efficiency). Although we obtained positive results for the flipped classroom pedagogical model, other more specific studies need to be carried out. We look forward to furthering studies on the subject, investigating particular analyzes of the test activity (e.g., generation of test data [9]). The theme also allows the investigation of experimental studies that analyze the flipped classroom with other pedagogical models, for example, those that promote game-based learning.

ACKNOWLEDGEMENTS

This study was financed by the University of São Paulo and the Brazilian funding agencies: the State of São Paulo Research Foundation - FAPESP (under processes no. 2013/07375-0, 2017/10941-8, and 2018/26636-2), Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) - Finance Code 001, and CNPq (under process no. #312922/2018-3).

REFERENCES

[1] P. Ammann and J. Offutt. 2008. *Introduction to Software Testing*. (1st ed.). Cambridge University Press, New York, NY, USA.

[2] J. Bergmann and A. Sams. 2012. *Flip your classroom: Reach every student in every class every day*. International Society for Technology in Education, Washington, D.C., EUA.

[3] J. F. P. Cheiran, E. de M. Rodrigues, E. L. de S. Carvalho, and J. P. S. da Silva. 2017. Problem-based learning to align theory and practice in software testing teaching. In *Brazilian Symposium on Software Engineering*, 328–337.

[4] P. J. Clarke, D. Davis, T. M. King, J. Pava, and E. L. Jones. 2014. Integrating testing into software engineering courses supported by a collaborative learning environment. *ACM Transactions on Computing Education*, 14, 3, 1–33.

[5] B. S. Clegg, J. M. Rojas, and G. Fraser. 2017. Teaching software testing concepts using a mutation testing game. In *International Conference on Software Engineering*, 33–36.

[6] J. Cohen. 1992. A power primer. *Psychological bulletin*, 112, 1, 155.

[7] T. D. Cook and D. T. Campbell. 1979. *Quasi-Experimentation: Design & Analysis Issues for Field Settings*. Houghton Mifflin, Boston, Massachusetts, United States.

[8] A. Cury. 2006. *Organização e métodos: uma visão holıstica*. Atlas.

[9] M. Delamaro, M. Jino, and J. Maldonado. 2016. *Introdução ao Teste de Software*. Elsevier, Rio de Janeiro, Brasil.

[10] A. C. Dias-Neto, S. Matalonga, M. Solari, G. Robiolo, and G. H. Travassos. 2017. Toward the characterization of software testing practices in south america: looking at brazil and uruguay. *Software Quality Journal*, 25, 4, 1145–1183.

[11] G. C. Gannod, J. E. Burge, and M. T. Helmick. 2008. Using the inverted classroom to teach software engineering. In *International Conference on Software Engineering*, 777–786.

[12] V. Garousi and J. Zhi. 2013. A survey of software testing practices in canada. *Journal of Systems and Software*, 86, 5, 1354–1376.

[13] T. Hynninen, J. Kasurinen, A. Knutas, and O. Taipale. 2018. Software testing: survey of the industry practices. In *International Convention on Information and Communication Technology, Electronics and Microelectronics*, 1449–1454.

[14] P. C. Jorgensen. 2013. *Software testing: a craftsman's approach*. Auerbach Publications.

[15] C. Kaner and R. L. Fiedler. 2005. Inside out: a computer science course gets a makeover. In *The National Convention of the Association for Educational Communications and Technology*, 254–264.

[16] S. K. Khalsa, Y. Labiche, and J. Nicoletta. 2016. The power of single and error annotations in category partition testing: an experimental evaluation. In *International Conference on Evaluation and Assessment in Software Engineering*, 28:1–28:10.

[17] O. A. L. Lemos, F. F. Silveira, F. C. Ferrari, and A. Garcia. 2017. The impact of software testing education on code reliability: an empirical assessment. *Journal of Systems and Software*.

[18] B. Maqbool, F. U. Rehman, M. Abbas, and S. Rehman. 2018. Implementation of software testing practices in pakistan's software industry. In *International Conference on Management Engineering, Software Engineering and Service Sciences*, 147–152.

[19] S. P. Ng, T. Murnane, K. Reed, D. Grant, and T. Y. Chen. 2004. A preliminary survey on software testing practices in australia. In *Australian Software Engineering Conference*, 116–125.

[20] L. N. Paschoal, B. R. N. Oliveira, E. Y. Nakagawa, and S. R. S. Souza. 2019. Can we use the flipped classroom model to teach black-box testing to computer students? In *Brazilian Symposium on Software Quality*, 158–167.

[21] L. N. Paschoal, L. R. Silva, and S. R. S. Souza. 2017. Abordagem flipped classroom em comparação com o modelo tradicional de ensino: uma investigação empírica no âmbito de teste de software. In Simpósio Brasileiro de Informática na Educação, 476–485.

[22] L. N. Paschoal and S. R. S. Souza. 2018. A survey on software testing education in brazil. In *Brazilian Symposium on Software Quality*, 334–343.

[23] L. N. Paschoal and S. R. S. Souza. 2018. Planejamento e aplicação de flipped classroom para o ensino de teste de software. *RENOTE - Revista Novas Tecnologias na Educação*, 16, 2, 1–10.

[24] A. Soska, J. Mottok, and C. Wolff. 2016. An experimental card game for software testing: development, design and evaluation of a physical card game to deepen the knowledge of students in academic software testing education. In IEEE Global Engineering Education Conference, 576–584.

[25] D. M. Souza. 2017. *Subsídios à integração de ferramentas de avaliação automática e sistemas de gerenciamento de aprendizagem*. Ph.D. Dissertation. Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo, São Carlos.

[26] G. Tan. 2016. A collection of well-known software failures. (2016). http://www.cse.psu.edu/~gxt29/bug/softwarebug.html.

[27] S. Tiwari, V. Saini, P. Singh, and A. Sureka. 2018. A case study on the application of case-based learning in software testing. In *Innovations in Software Engineering Conference*, 1–5.

[28] C. Wohlin, P. Runeson, M. Hst, M. C. Ohlsson AND B. Regnell, and A. Wessln. 2012. *Experimentation in Software Engineering*. Springer Publishing Company, Incorporated, New York City, USA.

[29] P. Y. Wu, P. Manohar, and S. Acharya. 2016. The design and evaluation of class exercises as active learning tools in software verification and validation. *Information Systems Education Journal*, 14, 4, 4–12.

[30] M. Zhivich and R. K. Cunningham. 2009. The real cost of software errors. *IEEE Security Privacy*, 7, 2, 87–90.