Need for Sleep: the Impact of a Night of Sleep Deprivation on Novice Developers' Performance

Davide Fucci, Giuseppe Scanniello, *Member, IEEE*, Simone Romano, and Natalia Juristo

Abstract—We present a quasi-experiment to investigate whether, and to what extent, sleep deprivation impacts the performance of novice software developers using the agile practice of test-first development (TFD). We recruited 45 undergraduates, and asked them to tackle a programming task. Among the participants, 23 agreed to stay awake the night before carrying out the task, while 22 slept normally. We analyzed the quality (i.e., the functional correctness) of the implementations delivered by the participants in both groups, their engagement in writing source code (i.e., the amount of activities performed in the IDE while tackling the programming task) and ability to apply TFD (i.e., the extent to which a participant is able to apply this practice). By comparing the two groups of participants, we found that a single night of sleep deprivation leads to a reduction of 50% in the quality of the implementations. There is notable evidence that the developers' engagement and their prowess to apply TFD are negatively impacted. Our results also show that sleep-deprived developers make more fixes to syntactic mistakes in the source code. We conclude that sleep deprivation has possibly disruptive effects on software development activities. The results open opportunities for improving developers' performance by integrating the study of sleep with other psycho-physiological factors in which the software engineering research community has recently taken an interest in.

Index Terms—sleep deprivation; psycho-physiological factors; test-first development.

1 Introduction

The loss for the American economy due to the lack of sleep of its workforce, whether chronic or not, is estimated to be 63 billion USD every year.[1] Nevertheless, as depicted in a scene from the movie *The Social Network* where Mark Zuckerberg writes code for 36 hours straight, forgoing sleep appears to be a badge of honor in the programmers and start-up communities.

The importance of sleep is nowadays recognized in the field of economics, where it is shown that sleep disturbances contribute to decreased the employees' performances at a high cost for the employers (e.g., [1], [2]). In management science, sleep loss was found to bewilder decision-makers activities [3].

In general, the lack of sleep affects working memory, creativity, decision making, multitasking ability, response time, and focus [4]. Not getting enough sleep prevents the brain from restoring its effectiveness, as it needs to work harder to accomplish the same amount of work [5]. In particular, the performance for tasks that require attention declines as a function of hours of sleep deprivation [6].

- *D. Fucci is with HITeC and the University of Hamburg, Hamburg, Germany*
  E-mail: fucci@informatik.uni-hamburg.de
- *G. Scanniello and S. Romano are with DiMIE - University of Basilicata, Potenza, Italy.*
  E-mail: {giuseppe.scanniello, simone.romano}@unibas.it
- *N. Juristo is with Technical University of Madrid, Madrid, Spain*
  E-mail: natalia@fi.upm.es

1. http://ti.me/xXa17G

In recent years, the software engineering community has been interested in the role played by factors related to human biology and physiology concerning several aspects of software development (e.g., [7], [8], [9]). A survey of 311 developers [10] found that *sleepiness* is perceived as one of the main causes of mental fatigue resulting in performance drop. However, *how* sleep affects software developers has not been studied so far. Given the link between sleep and cognitive performance showed in physiological research (e.g., [5], [11]), we believe that sleep deprivation can have serious repercussions for software developers' performance. The assessment of the effects of sleep deprivation will contribute to the broader investigation of the role of physiological factors for software engineers, with the goal of supporting them in their work; for instance by informing them when to take breaks so to avoid counterproductive actions.

In this paper, we investigate the following primary research question:

> To what extent does sleep deprivation impact developers' performance?

To answer this research question, we performed a quasi-experiment with 45 (final-year) Undergraduate Students in Computer science at the University of Basilicata in Italy. The participants in the experiment worked on a programming task requiring them to use the popular agile practice of test-first development (TFD) [12]; 23 of them did so while being totally sleep deprived—i.e., they forewent sleep the

night. We based our experiment on TFD because, together with unit testing, it was the main focus of the course in which the experiment was embedded. Moreover, TFD is well known and largely applied in software industry, its application requires discipline and rigor [13], [14], [15], and there exists non-invasive, validated tools that measure whether the process is followed correctly [16].

Participants were assigned to the treatment group (i.e., students who forewent sleep the night before the experiment) and to the control group (i.e., students who slept regularly the night before the experiment) by their availability to forgo one night of sleep, instead of a random assignment. Therefore, we consider our investigation a quasi-experiment rather than a randomized controlled experiment.

Comparing the two groups of participants, we found that:

- the quality of the software measured as functional-correctness produced by sleep-deprived software developers drop by half compared to developers under normal-sleep condition;
- sleep deprivation can have an impact on the engagement of developers, as well as their ability to follow the TFD practice.

The first contribution of this paper is to present and discuss the results of the first empirical study on the role of sleep deprivation in the software engineering field. Overall, the results of our quasi-experiment suggest that assessing sleep condition can provide indications on the quality of the source code that software developers write as well as their performance. Our second contribution is a replication package [2] and a series of lessons learned to foster replications and further studies. **Paper Structure.** In Section 2, we provide background information and present related work. In Section 3, we show the design of our experiment. The findings are reported in Section 4, and discussed in Section 5. Final remarks and future work conclude the paper in Section 6.

2 Background and Related Work

In this section, we report an overview of the software engineering research involved with the study of the biological and physiological facets of software developers (Section 2.1) and the key concepts used in our study from the medical research about sleep (Section 2.2).

2.1 Physiological factors software engineering

Investigating human cognitive endeavor through physiological measures is nowadays a standard practice in psychology [17], [18]. In recent years, the software engineering research community has taken an interest in studying how several aspects of software development (e.g., code comprehension [7], software quality [19]) impact the software developers' cognitive state—measured using physiological signals. Some of the first studies in this context are in the sub-field of program comprehension. For example, Sharif and Maletic [20] used an eye-tracking device to understand developers' naming conventions strategies. In a within-subjects controlled experiment, the authors observed differences in visual effort and elapsed time between developers comprehending source code identifiers written using camel case and the ones who read code written using underscore (i.e., snake case). They recommend novices to use the latter style, although the gap between the two narrows with experience. Similarly, Bednarik and Tukiainen [21] used eye-tracking to understand what part of source code expert developers look at, compared to novices.

Other investigations aims to understand the developers' brain. Floyd *et al.* [22] used medical imagining, through functional Magnetic Resonance Imaging (fMRI), to assess which areas of the brain are activated when reading source code. They found that the same areas of the brain are activated when reading source code as well as when reading prose text. However, different cerebral activities are associated with the two types of text. Moreover, their experiment showed that more experienced developers tend to read prose and code similarly at the neural level. Siegmund *et al.* [7] explored the feasibility to use fMRI to directly measure program comprehension. To that end, the authors conducted a controlled experiment with 17 participants, with the same level of programming experience, recruited among computer science and mathematics students. Participants were asked to comprehend six short source code snippets. Siegmund *et al.* found distinct activation patterns of five brain regions related to working memory, attention, and language processing [7]. The results of a different controlled experiment using fMRI on program comprehension were reported in [23]. The authors involved 11 participants and manipulated experimental conditions to isolate specific cognitive processes related to bottom-up comprehension and comprehension based on beacons providing hints about the purpose of a snippet (e.g., method signatures and common programming idioms). The results showed that beacons ease comprehension. Ikutani and Uwano [24] measured developers' brain activity when comprehending code. They used Near Infra-Red Spectroscopy (NIRS), a less invasive alternative to fMRI which measures the brain frontal-pole activity. Their small scale experiment shows that such area is activated when comprehending source code in which a variable needs to be memorized, but not when parsing *if-else* statements. Parnin [25] used subvocal utterances, emitted by developers while performing two tasks of different complexities. The author found a statistically significant difference between the number and intensity of subvocalization events corresponding to the two tasks. Fritz *et al.* [9] combined three physiological features (i.e., eye movement, the electrical signals of skin and brain) to distinguish programming tasks according to their difficulty. These results showed the potential of physiological techniques for characterizing development tasks according to their difficulty.

Developers' performance recently became the subject of research exploiting physiological measures, too. For example, Radevski [26] proposed a continuous monitoring of developers productivity based on brain electrical activities;

2. https://doi.org/10.6084/m9.gshare.5483974

whereas Müller and Fritz [8] used an ensemble of physiological metrics to measure the progress and incorruptibility of developers performing small development tasks.

Müller and Fritz [19] investigated the use of biometrics to identify code quality concerns in real time. In a longitudinal experiment with graduate students—then replicated on a smaller scale with professional software developers—they showed that heart rate, respiration rate, and skin temperature could determine the parts of the codebase in which developers are more likely to introduce a bug. Compared with this investigation, our quasi-experiment can be considered complimentary because we study the effect of sleep deprivation—usually associated with reduced blood flow in several regions of the brain, and changes in body temperature [27]—on the capacity of developers to write source code.

Software development, like many other intellectual activities, is ruled by emotions that can result in stressful situations (e.g., pressing deadlines, work within a restricted budget) [28]. Ostber *et al.* [29] proposed the use of salutogenesis[3] to support stressed developers. In their experiment, the authors make use of the concentration of cortisol and α-amylase in the saliva, as the physiological measure to infer the participants' stress level.

Sarkar and Parnin analyzed mental fatigue of software developers [10]. They surveyed 311 software developers and carried out an observational study with nine professionals. Their results show that fatigued developers have problems in focusing, coming up with optimal solutions and tend to make logical mistakes causing bugs. Moreover, fatigue hampered developers' creativity and motivation. They reported that one of the leading cause of mental fatigue for software developers is *sleepiness*.

Our research differs from previous work because our goal is to directly collect evidence about the role of sleep deprivation in software development. This study is the first in this respect.

2.2 Physiology of sleep

The daily hours of sleep needed vary depending on factors such as age and gender, with the average sleep duration being between seven and eight and half hours per night [31].

The empirical, medical research on sleep deprivation focuses on the effects of forcing the participants to sleep less than usual by keeping them awake between 24 and 72 hours [32]. These kinds of studies are very laborious and expensive to carry out leading to compromises in the study design [32]. For example, a small sample size can reduce the statistical power of the experiment, but a larger population may come at the expense of other methodological issues, such as a reduction in the cognitive test selection [32].

Medical research has shown that sleep deprivation decreases cognitive performance because of the wake-state instability due to microsleeps—i.e., very short periods of sleep-like state [33]. Moreover, sleep deprivation negatively affects the reactivity to stimuli from emotions [34]. The non-medical scientific communities studied the effects of sleep deprivation mainly in the field of manufacturing (e.g., [5], [2]) and decision making (e.g., [3], [35]), but not in the software engineering field. Sleep deprivation has detrimental effects on central features necessary for software development [36], such as working memory—that part of short-term memory in charge of manipulating transient information, which is fundamental for problem solving [11], [32]—attention, and decision making. Table 1 reports the main effects of sleep deprivation on cognitive performances that can have consequences for software development.

TABLE 1
Summary of cognitive performance effects of sleep deprivation

Cognitive state induced by sleep deprivation	References
Adverse mood changes	[37], [38], [39]
Loss of situational awareness	[40]
Reduced learning acquisition	[41], [42]
Deterioration of divergent thinking	[43], [44]
Perseveration on ineffective solutions	[33], [40]
Behavior compensatory effort	[45], [46]

There are several approaches to empirically measure sleep deprivation. These include self-assessment (e.g., on a scale with values from "completely sleepy" to "completely awake" [47]), sensor-based (e.g., smart band, polysomnography [48], PET scan), and psychometrics such as the psychomotor vigilance task (PVT) [49], [50]. PVT, employed by NASA to monitor astronauts' sleep condition [51], is cheaper and easier to administer compared to sensor-based approaches, while its measurements converge to the ones of more sophisticated tests [49]. A PVT task lasts for ten minutes during which the subject needs to react (e.g., by pressing a button on a keyboard) to a visual stimulus (e.g., a symbol appearing on a blank screen) before a given time threshold (usually 500 millisecond) [52]. An error (i.e., failing to react to the stimulus) or long reaction time (i.e., the timeframe between the stimulus appearing on the screen and the button being pressed) is attributable to attention lapses due to micro-sleep events indicating a condition of sleep deprivation [52]. The number of errors and the reaction time (RT) are then used to compute the following metrics [52]:

- *Performance score*: The percentage of correct reactions to stimuli;
- *Mean 1/RT*: The mean of the reciprocals of all RT;
- *10% Slowest 1/RT*: The mean of the reciprocals of the slowest 10% RT;
- *Minor Lapses*: The number of errors;
- *10% Fastest RT*: The mean of the fastest 10% RT.

3. It focuses on factors that support human health and well-being, rather than on factors that cause disease [30]. Specifically, this model is concerned with the relationship between health, stress, and coping.

TABLE 2
Summary of the experimental settings

Variable	Value
Participants	45 Computer Science Undergraduate Students
Groups size (control/treatment)	22 regular sleep/23 sleep deprivation (15 after the removal based on PVT)
Development env.	Java 8, JUnit 4, Eclipse 4.4.2
Training	Information System (IS) course (12 hours in lab, home assignments)
Experimental task	PigLatin (8 confirmations)
Task duration	90 minutes
Date	9 a.m. December 12, 2015
Place	DiMIE, University of Basilicata (Italy)

The reciprocal transform (1/RT) is one of the PVT outcomes most sensitive to total and partial sleep loss [53]. This metric emphasizes slowing in the optimum and intermediate response and it substantially decreases the contribution of long lapses, which is why the slowest and the fastest 10% of RTs are usually reciprocally transformed.

In this study, we use PVT, on top of a self-assessment questionnaire, to assess the adherence to the treatment—i.e., whether a participant slept or not. In the medical field, PVT is used not only to measure sleep deprivation [49], [50], [54] but also to perform data cleaning [55], [56]. We followed these good practices and used both PVT and the participants' self-declaration to get a more accurate assessment.

The human-computer interaction research community has previously used PVT to support the development of context-aware applications. For example, Abdullah *et al.* [57] conducted an ecological momentary assessment about the mobile phone usage of 20 participants over 40 days. The participants were prompted to take several PVT tasks during the day on their smartphone. By correlating the mobile phone usage with the PVT scores, the authors were able to predict, in an unobtrusive fashion, the participants cognitive performance and alertness at a given time of the day. With such information, a software system can improve its users productivity—e.g., by suggesting them to tackle the most important tasks while at the peak of their alertness. Leveraging a mechanism similar to PVT (i.e., measuring a subjects physical reaction to a visual stimuli), Althoff *et al.* [58] performed the largest study on the impact of sleep (and lack of thereof) on cognitive performance. The authors triangulated the sleep quality measurement, obtained from wearable devices, of more than 31.000 US-based Microsoft product users with their interactions with the Bing search engine[4] (75 millions keystrokes and mouse click). Their results align with laboratory-based sleep studies where PVT is used. They show that a single night of partial sleep deprivation (i.e., less than six hours) increases reaction time up to 4% with respect to normal sleeping condition (i.e., seven to nine hours). They also show that a decrease in cognitive performance due to two consecutive nights of partial sleep deprivation can last up to a period of six days before it is fully recovered.

4. https://www.bing.com

3 Study Design

We performed a quasi-experiment following the recommendations provided by Juristo and Moreno [59], Kitchenham *et al.* [60], and Wohlin *et al.* [61]. In Table 2, we show a summary of the experimental setting of this quasi-experiment —e.g., main variables representing the context of the study.

In the following, we first define the goal of our quasi-experiment and present the research questions (Section 3.1). Successively, we present the independent and dependent variables of our study (Section 3.2) and show the hypotheses defined to investigate the research questions (Section 3.3). We provide details on the participants in the experiment (Section 3.4), present its design (Section 3.5) and the infrastructure used to collect and analyze data (Section 3.6). We conclude by highlighting the experiment operation (Section 3.7) and experimental object (Section 3.8).

3.1 Goal

Following our main research question presented in Section 1, we defined the main goal of our quasi-experiment by applying the Goal Question Metrics (GQM) template [62] as follows:

Analyze developers' sleep deprivation
for the purpose of evaluating its effects
with respect to the quality of produced source code, the developers' engagement with the task, and their ability to follow TFD
from the point of view of the researcher
in the context of an Information System (IS) course involving novice developers and students in Computer Science and Software Engineering.

Accordingly, we defined and investigated the following research questions:

- RQ1. Does sleep deprivation decrease the quality of the solution to a programming task?
- RQ2. Does sleep deprivation decrease the developers' engagement with a programming task?
- RQ3. Does sleep deprivation decrease the ability of developers to apply TFD to a programming task?

The conceptual model and the operationalization of the constructs investigated in this experiment are presented in Figure 1. The rectangles in the upper part of this figure show the experiment constructs and their relationships with the research questions (in the ellipses). The bottom part shows the instruments used to allocate the treatments (left-hand side) and the metrics to measure the constructs (right-hand side).

3.2 Variables selection

Given the experiment design, the independent variable is *sleep*, a nominal variable with two levels, regular sleep (RS) and sleep deprived (SD). The control group (RS)

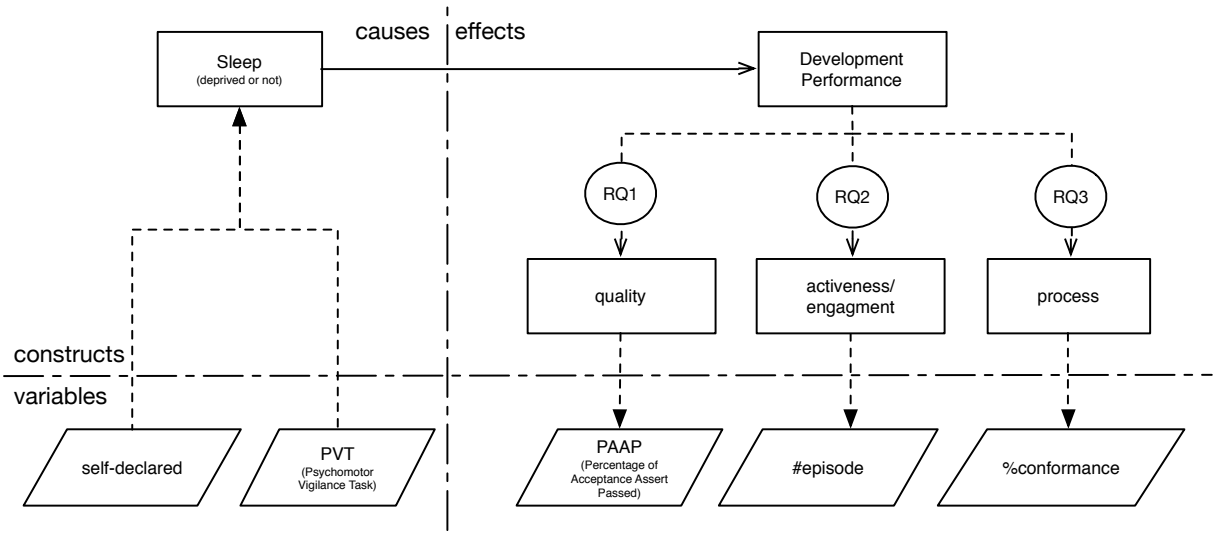


Fig. 1. Conceptual model and operationalization of our quasi-experiment

includes the participants who slept normally the night before carrying out the experimental task, while the treatment group (SD) includes the participants who forewent sleep the night before the experimental task.

The three dependent variables deal with three different constructs: quality, engagement, and process. Quality is intended here as a measure of how well the software satisfies the functional requirements. The metric we used is the percentage of acceptance asserts passed ($PAAP$)—the portion of the task correctly implemented based on an acceptance test suite, representing the compliance of the solution to the high-level requirements expressed in the user story. The acceptance test suite was developed by the researchers and hidden from the participants.

In particular, we calculated this metric as follows:

$$PAAP = \frac{\#ASSERT(PASS)}{\#ASSERT(ALL)} \times 100 \quad (1)$$

$PAAP$ measures the percentage of assert statements in the acceptance test suite passed by the production code delivered by a participant within the fixed duration of the task (i.e., 90 minutes). From a practical perspective, $PAAP$ represents how well the implementation provided by a participant fits the functional requirements expressed in the acceptance tests. The metric is equivalent to the concept of *functional correctness* reported in the ISO/IEC 25010. The total number of asserts included in the test suite—i.e., $\#ASSERT(ALL)$—is 13. $PAAP \in [0, 100]$; the higher its value, the better the quality of the implementation. The definition of this metric is founded on the recommendations to evaluate performance presented in Bergersen *et al.* [63].

The construct *engagement* (or also *activeness*) refers to how active the participants are in completing the task. A proxy to measure such construct is the number of activities a participant performs in the IDE while working. The activities we consider are a stream of low-level actions [64] taking place while programming. The first column of Table 3 reports the activities we consider and the related sequence of actions.

The metric $\#episodes$, which represents the total number of development activities performed within the duration of the experiment, is used to measure the participants' engagement. We considered the sensible development activities (see Table 3) for the specific process and task at hand (i.e., TFD applied to a simple task). $\#episodes$ assumes values in the interval $[0, +\infty]$ the larger the number of development episodes (i.e., the higher the value for $\#episodes$), the greater the engagement of a developer.

The *process* construct represents the extent to which a participant is able to follow TFD—i.e., develops a failing unit test, and then implements the production code to make it pass.[5] We study this construct because we have postulated that a night of sleep deprivation could negatively affect how developers adhere to a development technique (i.e., TFD) that requires them to be more focused than the traditional approach to software development. The process construct is measured through the $\%conformance$ as follows:

$$\%conformance = \frac{\#activities(\text{Test-first development})}{\#activities} \times 100, \quad (2)$$

where $\#activities$ is the total number of episodes recognized, while implementing the entire programming task. In other words, $\%conformance$ measures the percentage of development activities that were recognized as *Test-first development*. Therefore, $\%conformance \in [0, 100]$; the higher the value, the higher the adherence to TFD. We opted for this metric because it is well known and widely used in empirical studies on test-driven development (TDD) and TFD [65], [66].

3.3 Hypotheses formulation

Given the literature regarding the effects of sleep deprivation on cognition (see Section 2), it is expected to have

5. Although refactoring is one of the steps in TFD; here we do not explicitly address it due to the simplicity of the task.

TABLE 3
Heuristics used by Besouro to infer the development activities (from [64]). Number of recognized activities reported in parentheses.

Activity	Actions sequence
Test-first development (3006)	Test creation → Test compilation error → Code editing →Test failure → Code editing → Test pass Test creation → Test compilation error → Code editing → Test pass Test creation → Code editing → Test failure → Code editing → Test pass Test creation → Code editing → Test pass
Refactoring (772)	Test editing (file size changes ± 100 bytes) → Test pass Code editing (number of methods, or statements decrease) → Test pass Test editing AND Code editing → Test pass
Test addition (215)	Test creation → Test pass Test creation → Test failure → Test editing → Test pass
Production code (16)	Code editing (number of methods unchanged, statements increase) → Test pass Code editing (number of methods increase, statements increase) → Test pass Code editing (size increases) → Test pass
Test-last development (88)	Code editing → Test creation → Test editing → Test pass Code editing → Test creation → Test editing → Test failure → Code editing → Test pass

detrimental effects on software developers' performance. Therefore, we formulated the following null hypothesis to check the effect of sleep deprivation on the quality construct:

- $H0_{QLTY}$: The quality of source code produced by developers who stayed awake the night before is not better than the quality of source code produced by developers who slept normally.

We formulated the following null hypothesis to check the effect of sleep deprivation on the activeness construct:

- $H0_{ACTV}$: The activeness during an implementation task of developers who stayed awake the night before is not better than the activeness during the same task of developers who slept normally.

Finally, we formulated the following null hypothesis to check the effect of sleep deprivation on the process construct:

- $H0_{PROC}$: The adherence to TFD during an implementation task by developers who stayed awake the night before is not better than the adherence to TFD during the same task by developers who slept normally.

The alternative hypotheses are formulated as follows:

- $H1_{QLTY}$: The quality of the source code produced by developers who stay awake the night before is worse than the the quality of source code produced by developers who slept normally (i.e., $QLTY_{SD} < QLTY_{RS}$).
- $H1_{ACTV}$: Developers who stay awake the night before the execution of an implementation task are less active than developers who slept normally (i.e., $ACTV_{SD} < ACTV_{RS}$).
- $H1_{PROC}$: Developers who stay awake the night before the execution of an implementation task adhere less to TFD than developers who slept normally (i.e., $PROC_{SD} < PROC_{RS}$).

We have formulated $H0_{QLTY}$, $H0_{ACTV}$, and $H0_{PROC}$ to study RQ1, RQ2, and RQ3, respectively.

3.4 Sampling and participants

The participants in the experiment were final-year undergraduate students enrolled in an Information System (IS) course in Computer Science at the University of Basilicata (Italy). The content of the course includes elements regarding software testing, software development processes, software maintenance, and agile development practices with a focus on TFD, regression testing, and refactoring. Participants had passed all the exams related to the following courses: Procedural Programming, Object-Oriented Programming I, and Databases. In these courses, the participants gained experience with C/C++, Java, and TFD. The experiment was conducted as an optional exercise at the end of the IS course. We informed the participants that their grade in the IS course would not be affected by their participation in the experiment. The research questions were not disclosed to the participants until the completion of the experiment. The participants were aware that their data would be treated anonymously and disclosed only in aggregated form.

Out of the 95 students enrolled in the IS course, 45 decided to take part in the experiment. The results of a pre-questionnaire—administered before the experiment—showed that they had an average experience of 0.5 years as professional programmers. In general, the average years of experience with programming (e.g., in university courses, own projects) was 3.6 years; whereas, their experience with software testing was a little over one and a half year. The pre-questionnaire was composed of Likert items to be rated on a five-points scale (i.e., from *Very Inexperienced* to *Very Experienced*). The answers are summarized in Table 4. The participants were in good health conditions and aged between 20 and 34 years (average 23.56 years).

3.5 Experiment design

The quasi-experiment was designed to have one factor (i.e., *sleep*) and two treatments—sleep-deprived condition (SD) and regular sleep condition (RS), where the latter is control

TABLE 4
Relevant experience levels of the participants (n = 45) in the dataset. Sleep-deprived participants (n=23) reported in parentheses

	Experience levels				
	Very Inexperienced	Inexperienced	Neither	Experienced	Very Experienced
Programming (general)	0 (0)	2 (1)	36 (20)	5 (2)	2 (0)
Object oriented programming	0 (0)	6 (3)	33 (18)	5 (2)	1 (0)
Unit testing	0 (0)	38 (20)	6 (3)	1 (0)	0 (0)
Test-first development	16 (8)	20 (12)	7 (2)	2 (1)	0 (0)
Eclipse IDE	35 (20)	8 (3)	2 (0)	0 (0)	0 (0)

group [61]. 22 participants (six females) slept normally before the experiment, while 23 participants (two females) stayed awake.

We do not consider our study as a controlled experiment because randomization was not possible. In experimental design theory, randomization involves the random allocation of experimental units (i.e., participants) to the experimental groups [67], [61]. That is, in a design that uses randomization, the participants have the same chance to be assigned to the control or treatment group. Randomization can help to reduce the systematic differences between the groups, except for the manipulated variable of interest [67]. In such settings, it is possible to identify a strong causal link between the manipulation of the treatments and observed outcomes. In our case, randomization was discouraged due to ethical reason. In particular, law and university regulations prevent the possibility to pay students to take part in the study, as well as forcing them to forgo sleep. Consequently, the participants are assigned to the experimental groups (i.e., treatment and control) based on their voluntary choice to forgo (or not) sleep for one night before the study took place. Although there is some medical evidence that males and females are affected by sleep deprivation in different ways (e.g., [32], [68], [69]), we could not include gender as a blocking factor. Forcing a balanced number of female and male participants in the experimental groups was not possible because participation was voluntary. Similar considerations can be done for the participants age as blocking factor.

The pre-existing conditions of the participants are of paramount importance for experiments with human participants [70]. In our case, we had to address the possibility that participants possessed different pre-existing skills and experience regarding software development. To that end, we sampled from a homogeneous population—i.e., students attending the same course, with a similar academic background. We assessed the two experimental groups according to their GPA[6] (Grade Point Average), used as a proxy to measure the ability of computer science students with software engineering tasks (e.g., [71], [72], [73]). The average GPA of the participants in the RS group was 23.92, while for the SD it was 24.3. We further analyze the participant's pre-existing skills and experience to substantiate participants homogeneity before the study (see Appendix A). Accordingly, we assume the differences between groups to be negligible, and participants to be homogeneous.

We discarded alternative designs, such as repeated measure[7], because they are more vulnerable to threats to validity prominent in our context. In particular, a learning effect [74] can interfere with the result once the participants' performance are measured under one experimental condition (i.e., regular sleep) and later under the other (i.e., sleep deprivation) In such settings, the latter measurement can be the result of prior practice under the former condition. Controlling or compensating for such effect is particularly risky as it implies the use of statistical techniques that hampers the interpretation of the results [75], [76].

Another discarded alternative involved creating of a baseline (i.e., a programming task different from the one used in the experiment) for the SD group *before* the experiment, and then comparing the participants' performances obtained in the two tasks. The introduction of possible bias is barely moved from the difference between participant's skills to the difference between tasks.

3.6 Data collection and analysis

To compute $\#episodes$ and $\%conformance$, we leverage Besouro [16], an Eclipse IDE plug-in able to identify development activities, which are assigned to the following categories: *test-first development*, *refactoring*, *test addition*, *production code*, and *test-last development*. The identification of these activities is based on the heuristics by Kou *et al.* [64] reported in Table 3. The set of heuristics matches sequences of actions (see second column of Table 3) with the information logged by the IDE during its usage. In total, the plugin registered 4097 activities and 6348 actions.

As for the adherence of the participants to the treatment, in addition to the method based on the participants self-declaration, (see left hand side of Figure 1), a few days before the quasi-experiment, each participant belonging to the SD group filled a questionnaire concerning their

6. In Italy, the exam grades are expressed as integers and assume values between 18 and 30. The lowest grade is 18, while the highest is 30.

7. In a repeated measure design, a participant receives one night of sleep or not in two different periods

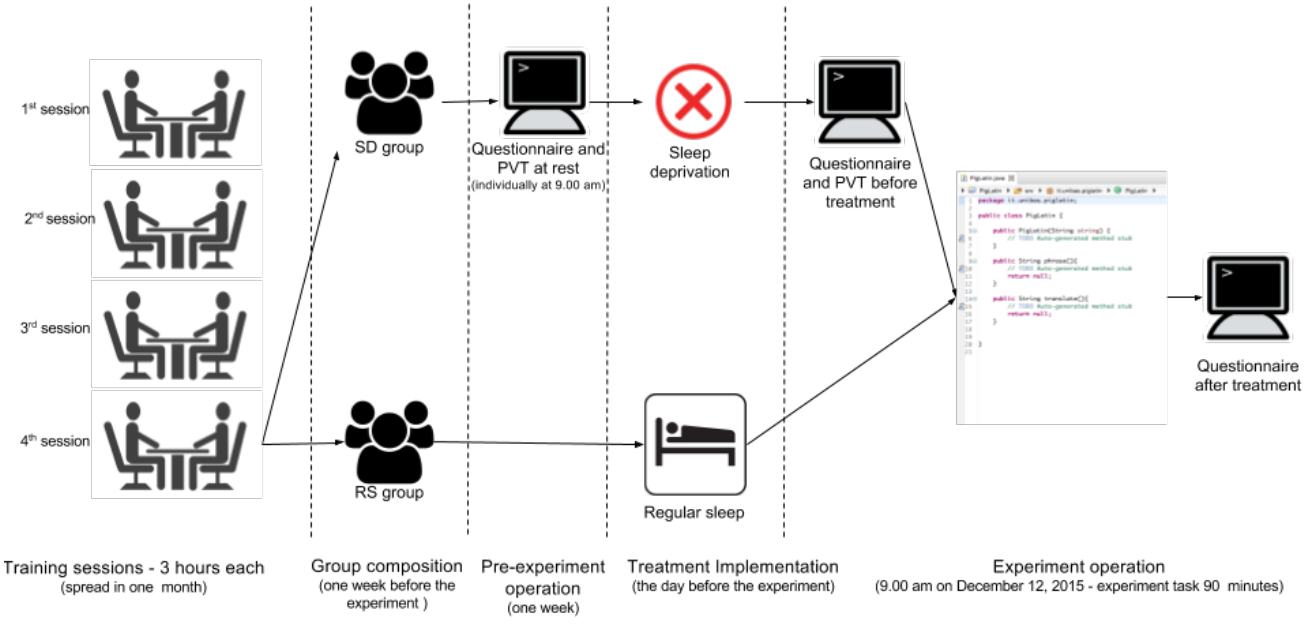


Fig. 2. Workflow of the study.

perceived quality of sleep during the previous night (pre-experiment operation in Figure 2). The participants declaring good quality of sleep took PVT at 9.00 a.m. in a research laboratory at the University of Basilicata. In case a participant perceived that she did not sleep well, we asked her to return at least one day later to retake both the questionnaire and PVT. This procedure allowed us to obtain baseline PVT scores (i.e., PVT scores at rest). PVT was administered to the participants using a software installed on a PC and a regular keyboard to register the participants' interaction [77]. The day of the experiment, before carrying out the experimental task, each participant in the SD group performed PVT again and filled the same questionnaire concerning their perceived quality of sleep during the previous night (see Figure 2).

All participants in the SD group carried out the experimental task independently from their PVT scores. We used the PVT scores to decide whether a participant in the SD group forewent a night of sleep. That is, we exploit PVT scores to assess the adherence to the sleep-deprivation treatment (see Figure 1). If a participant is sleep-deprived, we expect that her PVT values are worse on the day of the study than at rest. In Table 5, we report the differences between the PVT scores at rest and the ones obtained on the day of the study. Given how the PVT scores are calculated, a positive value of the differences for the following metrics *Performance score*, *Mean 1/RT*, *10% Slowest 1/RT* indicates a condition of sleep deprivation. Conversely, a negative value for *Minor Lapses* and *10% Fastest RT* indicates a condition of sleep deprivation. We followed a conservative approach to estimate if a participant stayed awake the night before the experiment. In particular, a participant in the SD group did not stay awake if at least one PVT score measured the day of the experiment was better than the score at rest. Following this criteria, we marked eight participants as non-compliant to the treatment (i.e., the gray rows in Table 5). For example, P7 had one more lapse (i.e., an error) when taking PVT at rest, vis-á-vis after the treatment; P7 also gave faster answers after the treatment compared to normal rest condition, as indicated by the 10% fastest RT values in Table 5. Such unexpected PVT score values led us to separately consider P7 and other seven participants during further analysis.

All the participants in the SD group (included those highlighted in Table 5) declared that the night before the experiment they were awake between 16 and 24 hours. On average, they did not sleep for 20.7 hours ($\pm$ 4.3 hours) before carrying out the experimental task. Participants in the RS group declared an average sleep time of 6.5 hours ($\pm$ 1 hour) the night before that experiment.

To improve the reliability of treatment implementation and reduce possible threats to conclusion validity, we considered two instances for the SD group, namely $SD_{Uncleaned}$ and $SD_{Cleaned}$. The participants who declared to forego sleep the night before the task were included in the $SD_{Uncleaned}$ dataset. On the other hand, $SD_{Cleaned}$ comprised the participants in the SD group after removing the eight participants marked in Table 5. As a result of data cleaning, $SD_{Uncleaned}$ contained all the 23 participants in SD, while $SD_{Cleaned}$ included 15 participants. The average GPA of the participants in $SD_{Cleaned}$ was 24.37 (vs. 24.3 in $SD_{Uncleaned}$)—the removal of participants did not affect the homogeneity between treatment and control groups regarding pre-existing skills.

TABLE 5
PVT score differences between the values at rest and those obtained on the day of the study for the sleep-deprived group. Participants in gray background are not included in the cleaned dataset ($SD_{Cleaned}$).

ID	Performance Score	Mean 1/RT	10% Slowest 1/RT	Minor Lapses	10%Fastest RT
P1	**-2**	0.01	**-0.14**	**1**	0
P2	5	0.94	1.42	-6	-21
P3	**-4**	0.31	0.35	**1**	-30
P4	0	**-0.09**	**-0.47**	0	**1**
P5	0	0.58	0.12	0	-10
P6	25	1.62	0.75	-26	-67
P7	**-1**	**-0.16**	**-0.72**	**1**	**14**
P8	5	0.34	1.22	-5	**39**
P9	0	0.77	0.05	0	-74
P10	1	0.26	0.63	0	-12
P11	3	1.03	0.88	-3	-66
P12	**-2**	**-0.17**	0.12	**2**	**23**
P13	5	0.31	0.77	-4	-8
P14	0	0.29	0.18	0	-15
P15	14	1.27	1.12	-16	-72
P16	1	0.75	1.06	-1	-38
P17	2	0.84	1.72	-2	-12
P18	1	0.52	0.44	-3	-23
P19	5	0.4	0.42	-5	-26
P20	**-4**	**-0.39**	0.41	**3**	0
P21	**-2**	0.32	**-0.47**	**1**	-27
P22	10	0.6	0.46	-13	-34
P23	3	0.35	0.04	-5	-50

We performed data analysis[8] considering both $SD_{Uncleaned}$ (from here onward also referred to as uncleaned dataset) and $SD_{Cleaned}$ (from here onward also referred to as cleaned dataset). In particular, we carried out the following steps:

1) Compute the descriptive statistics for all the dependent variables;
2) Use violin plots to summarize and visualize the gathered data;
3) Apply Bonferroni correction [78] (when needed) to mitigate the family-wise error rate;
4) Test our null hypotheses using the Mann-Whitney U test [79] due to the non-normality of the data;
5) Study the magnitude of the differences between two groups using Cliff's d [80] as a measure of the effect size. The confidence intervals for the effect size were also calculated to interpret its precision;
6) Provide the means percentage reduction as a less robust—though more intuitive—effect size indicator.[9]

3.7 Operation

In the following subsections, we describe the steps taken during the training and experimental operation.

3.7.1 Training

The IS course was accompanied by four hands-on training sessions of three hours each (see Figure 2). The sessions took place in a didactic laboratory at the University of Basilicata over a period of one month, and all the participants in our experiment attended the sessions. During the sessions, the students improved their knowledge regarding the development of unit tests in Java using the JUnit framework, refactoring, and TFD. They also familiarized themselves with the Eclipse IDE and the Besouro plug-in, later used in the experiment. Throughout the training sessions, the participants were asked to use TFD (although it was not mandatory) to solve several code katas of increasing difficulty, and worked on home assignments to further practice the contents of the course. All participants received the same training.

The material used in the training session (e.g., slides, practice tasks) was the same used in [81]. This material was translated from the English language to the Italian language to avoid that different familiarity levels of the participants with the English would affect experimental results.

The participants followed a five-steps procedure during each training session:

1) import a starting Eclipse project containing a stub of the expected API signature for the assigned programming task;[10]
2) start the Besouro Eclipse plugin;
3) implement a user-story card[11] (or simply card, from here on) using TFD;
4) stop the Besouro Eclipse plugin.

Regarding 3), we asked the participants to implement *confirmations* (i.e., conditions that need to be satisfied) for the card. The confirmations were of incremental difficulty, each building on the results of the previous one. However, we did not impose any order the participants had to follow to implement the confirmations, but we suggested them to follow the order in which they appeared. We also did not suggest any approach to browse, run, execute, and debug source code. That is, participants could freely use all the features present in the Eclipse IDE.

We had information on the composition of the experimental groups after the last training session, before the experiment, was over (left side of Figure 2). At the end of this session (November 30th, 2015), we administered the questionnaire to measure participants' self-perceived experience level (see Table 4).

3.7.2 Experiment operation

The experimental task was tackled at 9.00 a.m. on December 12, 2015, under controlled conditions in a laboratory at the University of Basilicata. The experimental operation followed the same five-steps procedure of the training session. We provided the participants with a hard copy of the task card shown in Figure 3. This card was never shown to the students during the training sessions.

8. The analysis was carried out using R version 3.3.1.

9. In particular, given μ_{RS} and μ_{SD}, corresponding respectively to the mean values of the control and treatment groups for a given dependent variable, the means' percentage reduction is computed as $\frac{(\mu_{RS}-\mu_{SD})}{\mu_{RS}}\%$.

10. This was done to avoid mundane tasks, but also to have a consistent naming convention.

11. A card is usually a very high-level definition of a requirement; it contains just enough information for the developers to reasonably estimate the effort required to implement it [82].

The experimental session lasted for 90 minutes, after which the participants returned source code of the solution they implemented. The projects were later used to extract the metrics necessary to assess the constructs. We impose a time limit to perform the task in the experimental session because this allows a better evaluation of developers' performance [63].

We allowed all the participants to use the Internet to accomplish the task, but we forbid them to use the Internet to communicate with one another. Two supervisors made sure that no interaction among participants took place.

Pig Latin is a language game in which words in English are altered. The objective is to conceal the words from others not familiar with the rules. The reference to Latin is a deliberate misnomer, as it is simply a form of jargon, used only for its English connotations as a strange and foreign-sounding language. We ask the participants to implement the following confirmations:

1) Create a PigLatin class that is initialized with a string
   - The string is a list of words separated by spaces: "hello world"
   - The string is accessed by a method named phrase
   - The string can be reset at any time without re-initializing
   - PigLatin.new('hello world')
2) Create a TranslateMethod, namely a method that translates the phrase from English to pig-latin.
   - The method will return a string.
   - The empty string will return nil.
   - " " translates to nil
3) Translate words that start with vowels.
   - Append "ay" to the word if it ends in a consonant.
     example: "ask" translates to "askay"
   - Append "yay" to the word if it ends with a vowel.
     example: "apple" translates to "appleyay"
   - Append "nay" to the word if it ends with "y".
     example: "any" translates to "anynay"
4) Translate a word that starts with a single consonant.
   - Removing the consonant from the front of the word.
   - Add the consonant to the end of the word.
   - Append "ay" to the resulting word.
     example: "hello" translates to "ellohay"
     example: "world" translates to "orldway"
5) Translate words that start with multiple consonants.
   - Remove all leading consonants from the front of the word.
   - Add the consonants to the end of the word.
   - Append 'ay' to the resulting word.
     example: "known" translates to "ownknay"
     example: "special" translates to "ecialspay"
6) Support any number of words in the phrase.
   example: "hello world" translates to "ellohay orldway"
   - Each component of a hyphenated word is translated separately.
     example: "well-being" translates to "ellway-eingbay"
7) Support capital letters.
   - If a word is capitalized, the translated word should be capitalized.
     example: "Bill" translates to "Illbay"
     example: "Andrew" translates to "Andreway"
8) Retain punctuation.
   - Punctuation marks should be retained from the source to the translated string
     example: "fantastic!" translates to "anfasticfay!"
     example: "Three things: one, two, three." translates to "Eethray ingsthay: oneyay, otway, eethray."

Fig. 3. Card administered to the participants

3.8 Experimental object

The experimental object is a programming exercise (i.e., a code kata), that consisted in implementing the PigLatin program commonly used to demonstrate TFD principles [83] . This card (Figure 3) contains eight confirmations. Each confirmation has a short description and at least one example of input and expected output. In this sense, confirmations are a sort of acceptance tests for the card. The acceptance tests used to calculate $PAAP$ were different from the ones shown in Figure 3. To implement the card, the participants used the Java programming language (version 6) and JUnit (version 4). We provided the participants with a template Eclipse project containing a stub of the expected API signature. Both groups tackled the same experimental task.

4 RESULTS

In this section, we first present the descriptive statistics for the dependent variables (Section 4.1) and then the results of the statistical hypothesis testing (Section 4.2). We conclude by reporting results from additional analyses (Section 4.3).

4.1 Descriptive statistics

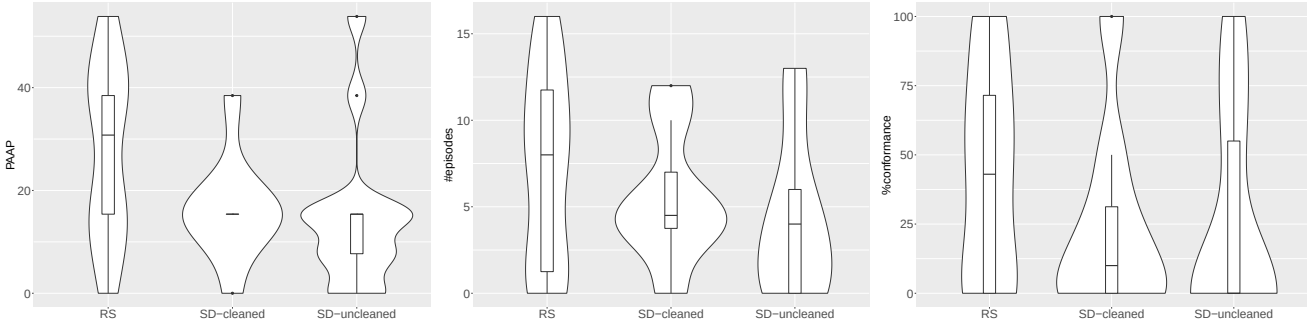
Table 6 reports the descriptive statistics for the metrics used to measure the dependent variables; the violin-plots (Figure 4) graphically summarize their distributions and the superimposed boxplots report the summary statistics. On average, the RS group performed twice as well as the SD group regarding $PAAP$, although their variation (i.e., standard deviation) is similar. Figure 4(a) visually shows that the quality of the software produced by sleep-deprived developers is below that of the developers working under normal-sleep condition. The boxplots representing the two groups do not overlap; therefore, we expect a clear difference. The difference is less clear for $\#episodes$. Figure 4(b) shows that sleep-deprived participants are less active. However, the observations at the bottom of the boxplots are similar. Six participants in the SD group (one in the RS group) were not active—i.e., they did not accomplish any of the development activities, yielding a $\#episode$ value of zero. Figure 4(c) shows that sleep-deprived developers were less capable of following TFD, although such difference is not as remarked as for the other two dependent variables. The participants in the SD group follow the TFD process 20% of the time less compared to those in the RS group (i.e., 25% vs. 45% in $\%conformance$ score); however, there is a considerable variation in both groups. In the SD group, 12 participants did not follow TFD (seven in the RS group)—i.e., these participants had a $\%conformance$ score equals to zero (see Appendix B for a post-mortem analysis of this result). The results are similar when all the original participants are considered in the SD group—i.e., no data cleaning based on PVT scores was applied. In general, Figure 4 shows that the distributions of the dependent variables are more uniform for RS, whereas they tend to be skewed towards the bottom (i.e., lower values) for SD.

4.2 Statistical inference

We applied Bonferroni correction when testing the hypotheses. Therefore, the α considered as a threshold to reject the

TABLE 6
Descriptive statistics for the considered metrics. Values in parentheses refer to the original dataset ($SD_{Uncleaned}$).

Metric	RS							SD						
	Min	Max	Mean	Q1	Median	Q3	St. Dev.	Min	Max	Mean	Q1	Median	Q3	St. Dev.
PAAP	0	53.85	28.57	15.38	38.46	38.46	16.344	0 (0)	53.85 (53.85)	14.36 (15.05)	7.692 (7.692)	15.38 (15.38)	15.38 (15.38)	14.498 (13.013)
#episodes	0	16	7.571	2	9	12	5.39	0 (0)	13 (13)	4.267 (4.696)	0 (1)	4 (4)	6 (6.5)	4.59 (4.3)
%conformance	0	100	45.43	0	50	73	37.682	0 (0)	100 (100)	25.27 (24.96)	0 (0)	0 (0)	55 (50)	38.653 (36.738)



(a) (b) (c)

Fig. 4. Violin-plots representing the dependent variables distribution for (a) quality ($PAAP$), (b) engagement ($\#episodes$), (c) test-first development conformance ($\%conformance$) for the regular (RS) and sleep-deprived (SD) groups (cleaned and uncleaned datasets).

TABLE 7
Shapiro-Wilk test results to assess the normality of dependent variable distributions (α = .05)

DV	Shapiro-Wilk	p-value	Normally distributed
PAAP	.48	2.02e-11	No
#episodes	.91	.02	No
%conformance	.79	2.35e-06	No

null hypotheses was .016 (i.e., the standard α .05 divided by the three hypotheses being tested). We used the non parametric Mann-Whitney U test because the dependent variables are not normally distributed according to the result of Shapiro-Wilk test [84], reported in Table 7.

In this section, we discuss the results based on Table 8 which reports the test statistics, p-values, and effect sizes.
RQ1: PAAP. The test result allowed us to reject the null hypothesis $H0_{QLTY}$, the sleep-deprived group (SD) performs worse than regular sleep group (RS) regarding the percentage of assert passed ($PAAP$). The estimated effect size is to be considered *medium*, and its negative sign is consistent with the direction of the alternative hypothesis. As the null hypothesis is rejected, the confidence interval of the effect size does not include zero, but its range is quite wide. In other words, an effect of sleep deprivation on external software quality exists and it is showed to be medium in our quasi-experiment. However, a more precise estimation of its real size, which could be as large as 1.58, requires further replications. From a practical perspective, sleep deprivation can deteriorate software quality, measured through $PAAP$, of about 50% as indicated by the percentage reduction estimator.

RQ2: ACTV. The null hypothesis about developers' engagement ($H0_{ACTV}$) could not be rejected at the considered p-value threshold. Although the medium effect size is consistent with the direction postulated in the alternative hypothesis, its wide confidence interval includes zero. Therefore, we cannot be sure that an effect of sleep on developers' engagement exists in reality, but if it does, it would likely be detrimental (as large as 1.31). Considering that, sleep-deprived developers were less engaged with the task—i.e., perform 43% fewer development activities with respect to developers working under normal-sleep condition.

RQ3: PROC. The null hypothesis regarding developers' adherence to the TFD process ($H0_{PROC}$) could not be rejected at the considered p-value threshold. The estimated medium effect size is negative as postulated by the alternative hypothesis. As for the case of $\#episodes$, given the evidence collected in this experiment, we cannot be sure whether an effect of sleep deprivation exists in reality for $\%conformance$. The result suggests that the effect is likely to be negative and as large as 1.18. From the effect-size estimate in this study, sleep-deprived software developers

TABLE 8
Null-hypothesis testing, and effect size results. The $*$ indicates values significant after applying the Bonferroni correction ($\alpha = 0.016$). Values in parentheses refer to the original dataset ($SD_{Uncleaned}$).

Research question	Hypothesis	Mann-Whitney U	p-value	effect size	% reduction	95% effect size CI
RQ1	$H0_{QLTY}$	208 (229)	.005* (.006*)	-0.498 (-0.466)	-49.73 (-47.32)	[-1.58, -0.20] ([-1.41, -0.20])
RQ2	$H0_{ACTV}$	167 (187)	.035 (.040)	-0.359 (-0.308)	-43.64 (-37.97)	[-1.31, 0.04] ([-1.17, 0.01])
RQ3	$H0_{PROC}$	163 (179)	.046 (.027)	-0.321 (-0.325)	-44.37 (-45.08)	[-1.18, 0.15] ([-1.13, 0.05])

perform TFD 44% less compared to software developers under normal-sleep condition.

4.3 Additional results

In this section, we report the results of additional analyses which were not originally planned when the experiment was designed, but emerged only after we looked at the data. We report the additional hypotheses and results in a separate section to mitigate researcher bias [85].

Differences in source code edits. The data collected from the participants' IDE allowed us to gather further insights about their behavior. We conjecture that the participants in the SD group are likely to commit syntactical mistakes (e.g., illegal sequence of tokens) when writing code due to their susceptibility to distractions [86]. Such mistakes need to be fixed by editing the source code (e.g., by renaming or removing tokens) so that the project can be compiled and the unit tests can be correctly executed. Therefore, from the log files registered by the Besouro Eclipse plugin, we gathered the editing actions[12] that involved deleting or renaming identifiers (e.g., variable and method names) which fix syntactical mistakes. We define FIX as the ratio of such actions over the total amount of actions registered.

We formulated the following one-tailed null hypothesis $H0_{FIX}$ to check the effect of sleep deprivation on the amount of fixing actions.

- $H0_{FIX}$: The amount of fixing actions performed by developers who stay awake the night before an implementation task is not larger than the amount of fixing actions performed by developers who slept normally.

The alternative hypothesis follows:

- $H1_{FIX}$: The amount of fixing actions performed by developers who stay awake the night before an implementation task is larger that the amount of fixing actions performed by developers who slept normally.

For the analysis, we used the same steps presented in Section 3.6. Using the Shapiro-Wilk test (W = .941, p-value = .023), we could not show that the FIX is normally distributed; therefore, we compared the two groups in terms of FIX using the same non-parametric test used for the previous hypotheses, i.e., the Mann-Whitney U test.

It appears that sleep-deprived participants perform more fix actions than the ones at rest (W = 82.5, $p - value$ = .008). The effect size is .41 , CI = [.2, .46], and mean percentage reduction is 54%. The *medium* effect size and its confidence interval allow us to conclude that sleep-deprived developers tend to do more fixing actions (approximately 54%) than developers who slept normally. Our explanation is that sleep deprivation has an effect on driving changes to the source code due to inattention.

TABLE 9
Sprearman correlations between the number of waking hours ($AWAKE$) and the three dependent variables.

	Spearman ρ	p-value
$PAAP$	-.382	.151
$\#episodes$	-.345	.206
$\%conformance$	-.108	.700

Correlations with waking hours. In the pre-questionnaire, we asked the participants to report the number of waking hours before performing the experimental task. Considering the SD group, the average number of awake hours ($AWAKE$) was 20.73 (median = 18, sd = 4.31, range = [16, 28]). The result of a Shapiro-Wilk test shows that the variable is not normally distributed (W = 0.803, $p - value$ = .004). Subsequently, we assess the correlations between $AWAKE$ and the three dependent variables of our study using Spearman correlation coefficient, deemed appropriate for non-normally distributed variables. In Table 9, we report the correlation analysis results.

Although the Spearman coefficients show a negative correlation—i.e., the longer a participant is awake, the less she performs—none was statistically significant. Nevertheless, this preliminary outcome can be used to create a baseline for future experiments—e.g., when deciding the different levels of sleep deprivation in further studies.

Perceptions of sleep-deprived participants. In a post-questionnaire, we gathered open-form feedback from the participants after the completion of the experimental task. In particular, three participants in the SD elaborated on the experience of being sleep deprived and the perceived effect of sleep deprivation on their performance during the task[13].

P2 - "It is difficult to undertake such task without any sleep. I was lacking focus, attention, and the ability of reading and understanding properly."

P6 - "In my opinion, under a sleep deprivation condition you quickly lose focus. Like in my case, I easily had doubts about pieces of code which I would otherwise grasp."

P17 - "Sleepiness remarkably slowed me down in each

12. Actions are the basic interactions within the IDE which together form the development activities presented in Table 3

13. The quotations were originally recorded in Italian and translated into English.

aspect, especially when it came to logical thinking. The task problem is surely simple but it was so difficult for me to think under a sleep deprivation condition."

It appears that sleep-deprived participants experience loss of focus and attention. From P6, it appears that sleep deprivation provokes a loss in self-confidence and causes uncertainties in the participant's. Further studies can survey this effect and try to quantify to which extents it applies to software development.

5 Discussion

In the following subsections, we answer the research questions (Section 5.1) and discuss limitations to be considered when interpreting our results (Section 5.2). We also delineate practical implications and future directions for research (Section 5.3). We conclude showing some lessons we learned from the execution of our experiment (Section 5.4).

5.1 Answers to the research questions

- ***RQ1***. *Does sleep deprivation decrease the quality of the solution to a programming task?*
  Since we were able to reject $H0_{QLTY}$ we answer RQ1 as follows: **developers who forewent one night of sleep write code which is approximately 50% more likely to not fulfill the specification with respect to the code written by developers under normal sleep condition.** This outcome is the most important result of our study, and it is in line with similar studies, in other disciplines, dealing with measuring the impact of sleep on performance with other types of cognitive tasks [87], [88]. The answer to RQ1 supports the theory that sleep deprivation has detrimental effects on the quality (i.e., functional correctness) of software developed by novices. In our additional analysis, we showed that sleep-deprived developers are more prone to perform editing actions to address syntax issues and, from preliminary qualitative evidence, that even simple operations can become difficult under such condition. It is perhaps not overly surprising, but evidence needs to be obtained through empirical studies to move from opinions and common sense to facts (e.g., [60], [89], [90]), as well as to have a first understanding of the size of the impact [91] that sleep deprivation can have on software development activities.
- ***RQ2***. *Does sleep deprivation decrease the developers' activeness in writing source code?* Although the results are not conclusive, we have evidence that one night of sleep deprivation can be harmful to the developers' activeness, with a loss of about 43% reported in this study.
- ***RQ3***. *Does sleep deprivation decrease the ability of developers to follow the TFD process?* In this study, we could not conclude that the ability to apply TFD is impacted by sleep. However, the evidence shows that sleep-deprived developers can encounter some difficulties applying TFD.

The main take-away from our quasi-experiment can be summarized as follows:

> One night of sleep deprivation is detrimental for software developers. In particular, sleep-deprived developers produce software of lower quality (i.e., functional correctness).

To increase our confidence in the results, it would be advisable to replicate this study with a design that takes into consideration individual characteristics of the participants, such as gender and age. It would be also important to conduct replications with developers with different levels of programming experience (i.e., more/less experienced) over multiple days with less/no sleep over a longer duration (e.g., one working week), or at different levels of sleep deprivation. In such regard, our additional results showed that (relatively) short awake-time differences among participants in the SD group (approximately four hours in the case of our experiment) may not be enough to show differences in performance. Based on this preliminary results, we suggest that a sensible second level of sleep deprivation condition, under which software developers can be studied, is between 25 and 28 hours of awake time [92].

5.2 Threats to validity

Sleep-deprivation studies are laborious and expensive to carry out leading to compromises in the design [32]. We discuss the threats that can affect the results following the recommendations in Wohlin *et al.* [61]. We ranked the validity threats from the most to the least relevant for this study. In particular, as we are testing a theory regarding sleep in the context of software engineering, we prioritize internal validity (i.e., warrant that a causal relationship exists) and construct validity (i.e., the metrics and instruments represent the constructs specified in the theory) rather than external validity (i.e., the generalization of the results to a context wider than that of this study).

5.2.1 Internal validity

The lack of randomization is the hallmark of quasi-experiment, and it is common practice in physiology studies where several treatments cannot be allowed to subjects (e.g., [67]). In our case, this can have an impact on the causal relationship between sleep and the dependent variables. For instance, the participants who are more accustomed to sleeping for a shorter amount of time may have opted to be included in the SD group (i.e., compensatory rivalry). The lack of randomization also prevented the possibility to apply blocking. For instance, the participants' age and gender can impact the causal relation between sleep deprivation and the response variables. However, after the participants decided to be included in one experimental group, we observed that the SD group was approximately two years older than the RS group. This difference should not affect the outcomes as the natural sleep-awake cycle

is not altered for subjects in the age range of the participants [32]. According to the available medical evidence, such alteration becomes apparent with aging (i.e., in subject older of approximately 55 years) [32], [93].

The *selection* threat of letting volunteers take part in the study can influence the results (45 out of 95 students enrolled in the IS course) as the sample can include participants with specific characteristics, for instance, more motivated developers. To deal with this kind of threat, we administered a questionnaire at the beginning of the course to assess the homogeneity of participants regarding the relevant pre-existing skills that might impact the study results. We did not find any statistically significant difference between the RS and SD groups regarding their knowledge of Java programming, unit testing, TFD, and Eclipse IDE. Similarly, we did not observe differences among the two groups when analyzing the assignment performed before the experiment took place (see Appendix A).

5.2.2 Construct validity

In this study, the adherence to treatment can be problematic. We measured the independent variable, *sleep*, not only through self-assessment, but also using a standard test from the clinical literature, namely the PVT test [49].

The dependent variables may suffer from the *mono-method bias*—i.e., a single type of measure is used to assess each construct. However, we exploited metrics existing in the literature (e.g., [66], [94], [95]), thus strengthening their reliability. The metrics were calculated automatically and were less prone to human measurement errors and rater subjectivity.

The participants were aware of being part of a study regarding the effects of sleep deprivation on software development practices; thus, posing the risk of *hypotheses guessing*. However, since the participants were not aware of the specific constructs being investigated, this threat can be considered addressed.

5.2.3 Conclusion validity

We addressed the threat to conclusion validity by using robust statistical methods and carefully checking the assumptions of the statistical tests used in the analysis. Moreover, we controlled for the error rate by applying Bonferroni correction when inferring the results.

Another threat to the conclusion validity is the *reliability of treatment implementation*. In our case, we could not observe the sleep treatment directly—i.e., we could not require the participants to be present in a controlled environment (e.g., a laboratory) where we could observe them sleeping (or not) during the night before the experimental task. However, rather than relying on self-assessment, we adopted PVT as it is the commonly used test in sleep research.

5.2.4 External validity

We sampled the participants by convenience from a population of students at the University of Basilicata. Therefore, generalizing the results to a different population (e.g., professional software developers) might pose a threat of *interaction of selection and treatment*. Our experiment involved basic programming skills; therefore, using of students as participants allow us to obtain reliable results [96]. Using students participants brings various advantages, such as homogeneous prior knowledge, the availability of a large sample [97], and the possibility to cheaply test experimental design and initial hypotheses [98]. Correspondingly, a threat of *interaction of setting and treatment* exists due to the non-real-world experimental task used. This would equally affect the results of the participants in the participants in both groups. That is, if there is a difference on a non-real-world task, we could speculate that this difference could increase in case of more complex development tasks. The settings are made more realistic by including TFD, a software development process used in industrial settings.

5.3 Implications and future extensions

In this section, we delineate how the results of our quasi-experiment fit in the emerging area of software engineering that focuses on physiological aspects.

- We complement the results from the work by Müller and Fritz [19] by showing that sleep deprivation—usually associated with reduced blood flow in several regions of the brain and changes in body temperature [27]—causes a severe decrease in the capacity of developers to write code that matches functional requirements. Medical research has shown that sleep deprivation negatively affects the responsiveness to stimuli from the same kind of emotions [34]. These outcomes can explain our results regarding sleep-deprived developers low engagement in performing even a small development task.
- We provided some evidence supporting the idea that sleep deprivation can hinder the application of TFD. In fact, to be adequately applied, TFD requires discipline and rigor [13], [14]. The detrimental effects of sleep deprivation on the developers' attention level [32] may explain our results. The perception of the participants in our study, reported in the post-questionnaire, are in line with the results of Sarkar and Parnin's [10] work investigating fatigue. Fatigue, as a result of sleep deprivation, caused participants to lose focus and hindered them from thinking logically even for a simple problem—both effects are also reported by Sarkar and Parnin's survey. The lack of focus was manifested in the larger number of syntax fix the sleep deprived participant needed to perform. Sleep deprivation can be detrimental for development practices and software engineering tasks for which developers' attention level is crucial. This point deserves further empirical investigations and it is relevant for practitioners and researchers.
- Sleep deprivation is a phenomenon that commonly occurs in software development, for example when deadlines are approaching [99], [100], [101]. Consequently, the result of this research can be exploited to inform

practitioners about the adverse effects of sleep deprivation from a technical perspective. Although our results apply to situations of total sleep deprivation, medical evidence shows similar (if not worse) results after a few nights of partial sleep restriction—e.g., less than five hours of sleep over several nights [102]. Sleep-deprived software developers should be aware that they are likely to produce buggy code and that such condition is likely to affect their programming performance. Therefore, information regarding the quality of their sleep (for example, gathered using PVT) can be utilized, in addition to the other physiological and biological metrics proposed recently (e.g., [9], [8]), to better support them. Alongside, fitness trackers could be used to measure the quality of developers' sleep supporting or replacing PVT and self-assessment. How and whether this kind of devices could help the research delineated in this paper will be the subject of our future work.

- Results seem to suggest that no sleep in a night reduces the quality of work, measured as functional correctness, of novice developers (i.e., undergraduate students). Mark *et al.* [86] conducted an *in-situ* study with students (76 undergraduates: 34 males and 42 females) on how different sleep duration can affect the use of information technology. The authors observed that students with less sleep: *i)* have significantly shorter focus suggesting higher multitasking, *ii)* may seek out activities requiring less attentional resources, and *iii)* tend to have a bad mood and use social media more than usual. On the basis of these results, we can speculate that to limit the detrimental effect of sleep deprivation, the daily work of a novice developer should be customized to adapt to her sleep-wake pattern—for example, assigning her less demanding or critical tasks. Further studies can look at how novice developers plan their tasks—e.g., decide whether more resources are needed when reviewing code implemented by a sleep-deprived developer.
- Future work should initially replicate this study to address the limitations presented in Section 5.2. As the medical evidence shows that different sleep deprivation time leads to various responses in the participants [102], a first improvement over the current design is to have treatments groups at different levels of sleep deprivation. Likewise, the cumulative effects of sleep deprivation can be assessed. These future directions need specific experimental infrastructures that are often not available in software engineering laboratories. For example, the effect of partial sleep restriction can be evaluated in a laboratory setting in which experimenters control that developers sleep for the right amount of time [32]. Further replications of our study could involve researchers from fields such as medicine—specifically, somnipathy—where sleep laboratories are commonplace. Other directions for further work include the study of the motivations behind sleep deprivation in software development, under which circumstances such condition happens (e.g., [99]), and how it is perceived in different software engineering phases.
  Our agenda includes the evaluation of *i)* what types of mistakes are more likely to be made by sleep-deprived developers; *ii)* how long developers should sleep to avoid the adverse impact of sleep deprivation on their performances taking into account their physiological traits; *iii)* the effect of sleep deprivation on other software engineering activities and practices (e.g., software maintenance and requirements elicitation).

5.4 Lessons learned

Software engineering experiments involving humans are quite common [103], however, ethical concerns in participants selection and allocation to experimental groups do not involve health-related issues which are commonplace in medicine. In our case, we not only had to follow a voluntary-based selection of the experimental sample as commonly happens in software engineering [103], particularly with student participants, but also for groups allocation. In medical studies, and in particular for sleep deprivation studies, this is the only available strategy due to ethical reasons [32]. From our experience, this aspect:

- makes it is costly to implement a dry-run of the study to assess the appropriateness of constructs, tasks, and experimental design. To avoid *wasting* resources (e.g., participants), we recommend using constructs which have been previously validated in studies targeting the same response variable(s), experimental objects, and measurement tools.
- makes it difficult to check for treatment conformance (i.e., assuring that participants were actually sleep-deprived) compared to traditional software engineering experiment in which the treatment is applied over a short period of time and can be automatically checked or enforced. In our case, we use PVT which does not require special equipment but does not scale well, in terms of time, once the pool of participants to be tested becomes large—for a single experimenter, testing a sample of 20 participants will take approximately four hours. Traditionally, in medical sleep deprivation studies, participants are observed in special chambers, implying that the experimenter(s) should invest the same time necessary to implement the treatment just to check that it is correctly followed. The cost of checking the conformance to treatment could be reduced using actimetry devices which are nowadays embedded in smart watches and health trackers.
- imposes a delicate tradeoff in the selection of an experimental design. We decided to employ a between-subjects design by verifying that participants in the experiment would be as homogeneous as possible in terms of existing skills relevant for the implementation of the experimental task. We excluded a within-subject design, although this seems to be the ordinary choice for medical sleep deprivation experiments. In

contrast to software engineering, medical experiments focus on factors such as chemical reactions associated with sleep conditions (e.g., brain receptors [44]), self-reported measure about psychiatric disorders (e.g., anxiety [1]), and cognitive tasks (e.g., attention span and learning rate [11]). Such constructs are not affected (or are to a less extent) by carryover and learning effects, typical of software engineering tasks, as they do not require specific skills (e.g., Java programming). Although repeated measure designs can be afforded in medicine, software engineering experiments emphasize tasks to observe a construct. Until a nomenclature of software engineering tasks—allowing researcher to compare them, for example, in terms of difficulty—will not available, we recommend a between-subjects design.

6 Final Remarks

In this paper, we presented the results of a first investigation about the effects of one night of sleep loss on the performance of software developers. We have asked the participants to implement a small Java application using the test-first development practice. One group of participants did not sleep the day before the experiment. The results indicated that sleep deprivation has a negative effect on the capacity of software developers to produce a software solution that meets given requirements. In particular, developers who forewent one night of sleep write code which is approximately 50% more likely not to fulfill the functional requirements with respect to the code produced by developers under normal sleep condition. We observed that sleep deprivation decreases developers' engagement with the development task and hinders their ability to apply the test-first development practice. For example, we observed a difference of about 44% in the engagement with the task of developers who forewent one night of sleep and developers under normal sleep condition. Moreover, the results showed that sleep-deprived developers performed 54% more fixing addressing syntactic mistakes compared to developers who slept regularly.

Our investigation i) has implications for education, research, and practice particularly when functional correctness is relevant—e.g., it might be useless to ask students, experiment participants, and developers to implement not-trivial task as their performances could be negatively affected in case of sleep deprivation. This paper adds to the new research direction in software engineering that focuses on programmers performance and the forces that impact it from a physiological perspective; ii) it represents a starting point for improving researchers and practitioners understanding of how software quality can benefit from monitoring developer's physiology (e.g., devising corrective actions to avoid that quality decreases due to sleep-deprived developers); and iii) provides a stepping store for follow-up studies in industry with different samples and using different kinds of empirical investigations (e.g., case studies)

Given the results of this study, we have reasons to believe that the community interested in assessing physiological factors for software developers performance should consider sleep quality in their research.

Acknowledgment

We would like to thank the participants in our study, especially those in the sleep-deprivation group. We would also like to thank Angelo Mecca for his precious support with PVT.

References

[1] M. R. Rosekind, K. B. Gregory, M. M. Mallis, S. L. Brandt, B. Seal, and D. Lerner, "The cost of poor sleep: workplace productivity loss and associated costs." *Journal of Occupational and Environmental Medicine*, vol. 52, no. 1, pp. 91–98, 2010.

[2] S. Vidaček, L. Kaliterna, B. Radošević-Vidaček, and S. Folkard, "Productivity on a weekly rotating shift system: circadian adjustment and sleep deprivation effects?" *Ergonomics*, vol. 29, no. 12, pp. 1583–1590, 1986.

[3] Y. Harrison and J. A. Horne, "One night of sleep loss impairs innovative thinking and flexible decision making," *Organizational behavior and human decision processes*, vol. 78, no. 2, pp. 128–145, 1999.

[4] J. J. Pilcher and A. J. Huffcutt, "Effects of sleep deprivation on performance: a meta-analysis." *Sleep: Journal of Sleep Research & Sleep Medicine*, 1996.

[5] D. F. Dinges, "An overview of sleepiness and accidents," *Journal of sleep research*, vol. 4, no. s2, pp. 4–14, 1995.

[6] B. Kolb and I. Q. Whishaw, "Why do we sleep and dream?" in *An Introduction To Brain and Behavior. Fourth Edition.* Worth Publishers, 2012, pp. 481–522.

[7] J. Siegmund, C. Kästner, S. Apel, C. Parnin, A. Bethmann, T. Leich, G. Saake, and A. Brechmann, "Understanding understanding source code with functional magnetic resonance imaging," in *Proceedings of the 36th International Conference on Software Engineering*. ACM, 2014, pp. 378–389.

[8] S. C. Müller and T. Fritz, "Stuck and frustrated or in flow and happy: Sensing developers' emotions and progress," in *Proceedings of the 37th International Conference on Software Engineering-Volume 1*. IEEE Press, 2015, pp. 688–699.

[9] T. Fritz, A. Begel, S. C. Müller, S. Yigit-Elliott, and M. Züger, "Using psycho-physiological measures to assess task difficulty in software development," in *Proceedings of the 36th International Conference on Software Engineering*. ACM, 2014, pp. 402–413.

[10] S. Sarkar and C. Parnin, "Characterizing and predicting mental fatigue during programming tasks," in *Proceedings of the 2nd International Workshop on Emotion Awareness in Software Engineering*. IEEE Press, 2017, pp. 32–37.

[11] L. Linde and M. Bergström, "The effect of one night without sleep on problem-solving and immediate recall," *Psychological research*, vol. 54, no. 2, pp. 127–136, 1992.

[12] K. Beck, *Test Driven Development: By Example*. Addison Wesley, 2003.

[13] R. C. Martin, "Professionalism and test-driven development," *Ieee Software*, vol. 24, no. 3, p. 32, 2007.

[14] R. Jeffries and G. Melnik, "Guest editors' introduction: Tdd–the art of fearless programming," *IEEE Software*, vol. 24, no. 3, pp. 24–30, 2007.

[15] V. One, "9th annual state of agile survey," Technical report, Version One, Tech. Rep., 2015.

[16] K. Becker, B. de Souza Costa Pedroso, M. S. Pimenta, and R. P. Jacobi, "Besouro: A framework for exploring compliance rules in automatic TDD behavior assessment," *Information and Software Technology*, vol. 57, pp. 494–508, 2015.

[17] J. L. Andreassi, *Psychophysiology: Human behavior & physiological response*. Psychology Press, 2013.

[18] A. F. Kramer, "Physiological metrics of mental workload: A review of recent progress," *Multiple-task performance*, pp. 279–328, 1991.

[19] S. C. Müller and T. Fritz, "Using (bio) metrics to predict code quality online," in *Proceedings of the 38th International Conference on Software Engineering*. ACM, 2016, pp. 452–463.

[20] B. Sharif and J. I. Maletic, "An eye tracking study on camelcase and under_score identifier styles," in *Program Comprehension (ICPC), 2010 IEEE 18th International Conference on*. IEEE, 2010, pp. 196–205.

[21] R. Bednarik and M. Tukiainen, "An eye-tracking methodology for characterizing program comprehension processes," in *Proceedings of the 2006 symposium on Eye tracking research & applications*. ACM, 2006, pp. 125–132.

[22] B. Floyd, T. Santander, and W. Weimer, "Decoding the representation of code in the brain: An fmri study of code review and expertise," in *Proceedings of the 39th International Conference on Software Engineering*. IEEE Press, 2017, pp. 175–186.

[23] J. Siegmund, N. Peitek, C. Parnin, S. Apel, J. Hofmeister, C. Kästner, A. Begel, A. Bethmann, and A. Brechmann, "Measuring neural efficiency of program comprehension," in *Proceedings of the 2017 11th Joint Meeting on Foundations of Software Engineering*. ACM, 2017, pp. 140–150.

[24] Y. Ikutani and H. Uwano, "Brain activity measurement during program comprehension with nirs," in *Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing (SNPD), 2014 15th IEEE/ACIS International Conference on*. IEEE, 2014, pp. 1–6.

[25] C. Parnin, "Subvocalization-toward hearing the inner thoughts of developers," in *Program Comprehension (ICPC), 2011 IEEE 19th International Conference on*. IEEE, 2011, pp. 197–200.

[26] S. Radevski, H. Hata, and K. Matsumoto, "Real-time monitoring of neural state in assessing and improving software developers' productivity," in *Proceedings of the International Workshop on Cooperative and Human Aspects of Software Engineering*. IEEE Press, 2015, pp. 93–96.

[27] M. Thomas, H. Sing, G. Belenky, H. Holcomb, H. Mayberg, R. Dannals, J. Wagner, D. Thorne, K. Popp, L. Rowland *et al.*, "Neural basis of alertness and cognitive performance impairments during sleepiness. i. effects of 24 h of sleep deprivation on waking human regional brain activity," *Journal of sleep research*, vol. 9, no. 4, pp. 335–352, 2000.

[28] D. Wastell and M. Newman, "The behavioral dynamics of information system development: A stress perspective," *Accounting*, 1993.

[29] J.-P. Ostberg, D. Graziotin, S. Wagner, and B. Derntl, "Towards the assessment of stress and emotional responses of a salutogenesis-enhanced software tool using psychophysiological measurements," in *Proceedings of the 2nd International Workshop on Emotion Awareness in Software Engineering*. IEEE Press, 2017, pp. 22–25.

[30] A. Antonovsky, *Health, stress, and coping*, 1st ed. Jossey-Bass Publishers San Francisco, 1979.

[31] T. E. Weaver, G. Maislin, D. F. Dinges, T. Bloxham, C. F. George, H. Greenberg, G. Kader, M. Mahowald, J. Younger, and A. I. Pack, "Relationship between hours of cpap use and achieving normal levels of sleepiness and daily functioning," *SLEEP-NEW YORK THEN WESTCHESTER-*, vol. 30, no. 6, p. 711, 2007.

[32] P. Alhola and P. Polo-Kantola, "Sleep deprivation: Impact on cognitive performance," *Neuropsychiatric disease and treatment*, vol. 3, no. 5, p. 553, 2007.

[33] J. S. Durmer and D. F. Dinges, "Neurocognitive consequences of sleep deprivation," in *Seminars in neurology*, vol. 25, no. 01. Copyright© 2005 by Thieme Medical Publishers, Inc., 333 Seventh Avenue, New York, NY 10001, USA., 2005, pp. 117–129.

[34] J. J. Pilcher, C. Callan, and J. L. Posey, "Sleep deprivation affects reactivity to positive but not negative stimuli," *Journal of psychosomatic research*, vol. 79, no. 6, pp. 657–662, 2015.

[35] Y. Harrison and J. A. Horne, "The impact of sleep deprivation on decision making: a review." *Journal of experimental psychology: Applied*, vol. 6, no. 3, p. 236, 2000.

[36] G. R. Bergersen and J.-E. Gustafsson, "Programming skill, knowledge, and working memory among professional software developers from an investment theory perspective," *Journal of Individual Differences*, 2011.

[37] J. M. Taub and R. J. Berger, "Performance and mood following variations in the length and timing of sleep," *Psychophysiology*, vol. 10, no. 6, pp. 559–570, 1973.

[38] P. Totterdell, S. Reynolds, B. Parkinson, and R. B. Briner, "Associations of sleep with everyday mood, minor symptoms and social interaction experience," *Sleep*, vol. 17, no. 5, pp. 466–475, 1994.

[39] M. M. David, A. MacLean, J. Knowles, and M. Coulter, "Rapid eye movement latency and mood following a delay of bedtime in healthy subjects: do the effects mimic changes in depressive illness?" *Acta Psychiatrica Scandinavica*, vol. 84, no. 1, pp. 33–39, 1991.

[40] T. H. Monk, "Practical consequences of fatigue-related performance failures," *Sleep*, vol. 30, no. 11, p. 1402, 2007.

[41] S.-S. Yoo, P. T. Hu, N. Gujar, F. A. Jolesz, and M. P. Walker, "A deficit in the ability to form new human memories without sleep," *Nature neuroscience*, vol. 10, no. 3, pp. 385–392, 2007.

[42] B. Rasch and J. Born, "About sleep's role in memory," *Physiological reviews*, vol. 93, no. 2, pp. 681–766, 2013.

[43] J. A. Horne, "Sleep loss and divergent thinking ability," *Sleep*, vol. 11, no. 6, pp. 528–536, 1988.

[44] N. Goel, H. Rao, J. S. Durmer, and D. F. Dinges, "Neurocognitive consequences of sleep deprivation," in *Seminars in neurology*, vol. 29, no. 04. © Thieme Medical Publishers, 2009, pp. 320–339.

[45] M. Engle-Friedman, "The effects of sleep loss on capacity and effort," *Sleep Science*, vol. 7, no. 4, pp. 213–224, 2014.

[46] G. R. J. Hockey, D. G. Wastell, and J. Sauer, "Effects of sleep deprivation and user interface on complex performance: a multilevel analysis of compensatory control," *Human Factors: The Journal of the Human Factors and Ergonomics Society*, vol. 40, no. 2, pp. 233–253, 1998.

[47] K. Kaida, M. Takahashi, T. Åkerstedt, A. Nakata, Y. Otsuka, T. Haratani, and K. Fukasawa, "Validation of the karolinska sleepiness scale against performance and eeg variables," *Clinical Neurophysiology*, vol. 117, no. 7, pp. 1574–1581, 2006.

[48] N. Douglas, S. Thomas, and M. Jan, "Clinical value of polysomnography," *The Lancet*, vol. 339, no. 8789, pp. 347–350, 1992.

[49] D. F. Dinges and J. W. Powell, "Microcomputer analyses of performance on a portable, simple visual rt task during sustained operations," *Behavior research methods, instruments, & computers*, vol. 17, no. 6, pp. 652–655, 1985.

[50] S. P. Drummond, A. Bischoff-Grethe, D. F. Dinges, L. Ayalon, S. C. Mednick, M. Meloy *et al.*, "The neural basis of the psychomotor vigilance task," *SLEEP-NEW YORK THEN WESTCHESTER-*, vol. 28, no. 9, p. 1059, 2005.

[51] D. Dinges, D. Mollicone, and M. Basner, "Psychomotor vigilance self test on the international space station (reaction_self_test)," 2012.

[52] M. Basner, D. Mollicone, and D. Dinges, "Validity and sensitivity of a brief psychomotor vigilance test (pvt-b) to total and partial sleep deprivation," *Acta Astronaut*, vol. 69, no. 11-12, pp. 949–959, 2011.

[53] M. Basner and D. F. Dinges, "Maximizing sensitivity of the psychomotor vigilance test (pvt) to sleep loss," *Sleep*, vol. 34, no. 5, pp. 581–591, 2011.

[54] S. Brown, P. Matsangas, and N. L. Shattuck, "Comparison of a circadian-based and a forward rotating watch schedules on sleep, mood, and psychomotor vigilance performance," *Proceedings of the Human Factors and Ergonomics Society Annual Meeting*, vol. 59, no. 1, pp. 1167–1171, 2015.

[55] N. Shattuck, P. Matsangas, and E. Powley, "Sleep patterns, mood, psychomotor vigilance performance, and command resilience of watchstanders on the "five and dime" watchbill," 2 2015. [Online]. Available: https://calhoun.nps.edu/handle/10945/44713

[56] N. Shattuck and P. Matsangas, "Work and rest patterns and psychomotor vigilance performance of crewmembers of the uss jason dunham: a comparison of the 3/9 and 6/6 watchstanding schedules," 2014. [Online]. Available: https://calhoun.nps.edu/handle/10945/44348

[57] S. Abdullah, E. L. Murnane, M. Matthews, M. Kay, J. A. Kientz, G. Gay, and T. Choudhury, "Cognitive rhythms: Unobtrusive and continuous sensing of alertness using a mobile phone," in *Proceedings of the 2016 ACM International Joint Conference on Pervasive and Ubiquitous Computing*, ser. UbiComp '16. New York, NY, USA: ACM, 2016, pp. 178–189. [Online]. Available: http://doi.acm.org/10.1145/2971648.2971712

[58] T. Althoff, E. Horvitz, R. W. White, and J. Zeitzer, "Harnessing the web for population-scale physiological sensing: A case study of sleep and performance," in *Proceedings of the 26th International Conference on World Wide Web*, ser. WWW '17. Republic and Canton of Geneva, Switzerland: International World Wide Web Conferences Steering Committee, 2017, pp. 113–122. [Online]. Available: https://doi.org/10.1145/3038912.3052637

[59] N. Juristo and A. Moreno, *Basics of Software Engineering Experimentation*. Englewood Cliffs, NJ: Kluwer Academic Publishers, 2001.

[60] B. Kitchenham, S. Pfleeger, L. Pickard, P. Jones, D. Hoaglin, K. El Emam, and J. Rosenberg, "Preliminary guidelines for empirical research in software engineering," *IEEE Trans. Softw. Eng.*, vol. 28, no. 8, pp. 721–734, 2002.

[61] C. Wohlin, P. Runeson, M. Höst, M. Ohlsson, B. Regnell, and A. Wesslén, *Experimentation in Software Engineering*. Springer, 2012.

[62] V. Basili, G. Caldiera, and D. H. Rombach, *The Goal Question Metric Paradigm, Encyclopedia of Software Engineering*. John Wiley and Sons, 1994.

[63] G. Bergersen, D. I. K. Sjøberg, and T. Dybå, "Construction and validation of an instrument for measuring programming skill," *IEEE Transactions on Software Engineering*, 2014.

[64] H. Kou, P. M. Johnson, and H. Erdogmus, "Operational definition and automated inference of test-driven development with zorro," *Automated Software Engineering*, vol. 17, no. 1, pp. 57–85, 2010.

[65] D. Fucci, B. Turhan, N. Juristo, O. Dieste, A. Tosun-Misirli, and M. Oivo, "Towards an operationalization of test-driven development skills: An industrial empirical study," *Information and Software Technology*, vol. 68, pp. 82–97, 2015.

[66] A. Höfer and M. Philipp, "An Empirical Study on the TDD Conformance of Novice and Expert Pair Programmers," in *Agile Processes in Software Engineering and Extreme Programming*. Berlin, Heidelberg: Springer Berlin Heidelberg, May 2009, pp. 33–42.

[67] W. R. Shadish, T. D. Cook, and D. T. Campbell, *Experimental and quasi-experimental designs for generalized causal inference.* Houghton, Mifflin and Company, 2002.

[68] K. Blatter, P. Graw, M. Münch, V. Knoblauch, A. Wirz-Justice, and C. Cajochen, "Gender and age differences in psychomotor vigilance performance under differential sleep pressure conditions," *Behavioural brain research*, vol. 168, no. 2, pp. 312–317, 2006.

[69] M. Ferrara, A. Bottasso, D. Tempesta, M. Carrieri, L. De Gennaro, and G. Ponti, "Gender differences in sleep deprivation effects on risk and inequality aversion: evidence from an economic experiment," *PloS one*, vol. 10, no. 3, p. e0120029, 2015.

[70] E. M. Tucker-Drob, "Individual differences methods for randomized experiments." *Psychological Methods*, vol. 16, no. 3, pp. 298–318, 2011.

[71] S. Abrahao, C. Gravino, E. I. Pelozo, G. Scanniello, and G. Tortora, "Assessing the effectiveness of sequence diagrams in the comprehension of functional requirements: Results from a family of five experiments," *IEEE Trans. on Soft. Eng.*, vol. 99, no. PrePrints, 2012.

[72] F. Ricca, M. Di Penta, M. Torchiano, P. Tonella, and M. Ceccato, "How developers' experience and ability influence Web application comprehension tasks supported by UML stereotypes: A series of four experiments," *Trans. on Soft. Eng.*, vol. 36, no. 1, pp. 96–118, 2010.

[73] G. Scanniello, C. Gravino, M. Risi, G. Tortora, and G. Dodero, "Documenting design-pattern instances: A family of experiments on source-code comprehensibility," *ACM Trans. Softw. Eng. Methodol.*, vol. 24, no. 3, pp. 14:1–14:35, 2015.

[74] L. Kuzniarz, M. Staron, and C. Wohlin, "An empirical study on using stereotypes to improve understanding of UML models," in *Workshop on Program Comprehension*. Bari, Italy: IEEE Computer Society, 2004, pp. 14–23.

[75] S. Vegas, C. Apa, and N. Juristo, "Crossover designs in software engineering experiments: Benefits and perils," *IEEE Transactions on Software Engineering*, vol. 42, no. 2, pp. 120–135, 2016.

[76] L. Madeyski and B. Kitchenham, "Effect sizes and their variance for ab/ba crossover design studies," *Empirical Software Engineering*, pp. 1–36, 2017.

[77] M. Y. Khitrov, S. Laxminarayan, D. Thorsley, S. Ramakrishnan, S. Rajaraman, N. J. Wesensten, and J. Reifman, "Pc-pvt: a platform for psychomotor vigilance task testing, analysis, and prediction," *Behavior research methods*, vol. 46, no. 1, pp. 140–147, 2014.

[78] O. J. Dunn, "Multiple comparisons among means," *Journal of the American Statistical Association*, vol. 56, pp. 52–64, 1961.

[79] W. J. Conover, *Practical Nonparametric Statistics*, 3rd ed. Wiley, 1998.

[80] V. B. Kampenes, T. Dybå, J. E. Hannay, and D. I. K. Sjøberg, "A systematic review of effect size in software engineering experiments," *Infor. & Soft. Tech.*, vol. 49, no. 11-12, pp. 1073–1086, 2007.

[81] D. Fucci and B. Turhan, "A Replicated Experiment on the Effectiveness of Test-First Development," in *2013 ACM/IEEE International Symposium on Empirical Software Engineering and Measurement (ESEM)*. IEEE, Oct. 2013, pp. 103–112.

[82] L. Williams, "The xp programmer: the few-minutes programmer," *IEEE Software*, vol. 20, no. 3, p. 16, 2003.

[83] J. Elkner. Using Test Driven Development in a Computer Science Classroom: A First Experience. [Online]. Available: http://www.elkner.net/jeff/testFirst/

[84] S. Shapiro and M. Wilk, "An analysis of variance test for normality," *Biometrika*, vol. 52, no. 3-4, pp. 591–611, 1965.

[85] L. K. John, G. Loewenstein, and D. Prelec, "Measuring the prevalence of questionable research practices with incentives for truth telling," *Psychological science*, vol. 23, no. 5, pp. 524–532, 2012.

[86] G. Mark, Y. Wang, M. Niiya, and S. Reich, "Sleep debt in student life: Online attention focus, facebook, and mood," in *Proceedings of the CHI Conference on Human Factors in Computing Systems*. New York, NY, USA: ACM, 2016, pp. 5517–5528.

[87] M. P. Walker, "The Role of Sleep in Cognition and Emotion," *Annals of the New York Academy of Sciences*, vol. 1156, no. 1, pp. 168–197, Mar. 2009.

[88] P. Maquet, "The Role of Sleep in Learning and Memory," *Science*, vol. 294, pp. 1048–1052, 2001.

[89] V. Basili, F. Shull, and F. Lanubile, "Building knowledge through families of experiments," *IEEE Trans. Softw. Eng.*, vol. 25, no. 4, pp. 456–473, 1999.

[90] F. Shull, J. C. Carver, S. Vegas, and N. J. Juzgado, "The role of replications in empirical software engineering," *Empirical Software Engineering*, vol. 13, no. 2, pp. 211–218, 2008.

[91] G. M. Sullivan and R. Feinn, "Using effect sizeor why the p value is not enough," *Journal of graduate medical education*, vol. 4, no. 3, pp. 279–282, 2012.

[92] D. F. DINGES, *Sleep debt and scientific evidence*. Sleep, 2004.

[93] P. Philip, J. Taillard, P. Sagaspe, C. Valtat, M. Sanchez-Ortuno, N. Moore, A. Charles, and B. Bioulac, "Age, performance and sleep deprivation," *Journal of sleep research*, vol. 13, no. 2, pp. 105–110, 2004.

[94] H. Erdogmus, M. Morisio, and M. Torchiano, "On the effectiveness of the test-first approach to programming," *IEEE Transactions on Software Engineering*, vol. 31, no. 3, pp. 226–237, March 2005.

[95] L. Madeyski, *Test-driven development: An empirical evaluation of agile practice*. Springer Science & Business Media, 2009.

[96] D. G. Feitelson, "Using students as experimental subjects in software engineering research - A review and discussion of the evidence," *CoRR*, vol. abs/1512.08409, 2015.

[97] J. Verelst, "The influence of the level of abstraction on the evolvability of conceptual models of information systems," in *Procedeedings of the International Symposium on Empirical Software Engineering*. Washington, DC, USA: IEEE Computer Society, 2004, pp. 17–26.

[98] D. Falessi, N. Juristo, C. Wohlin, B. Turhan, J. Münch, A. Jedlitschka, and M. Oivo, "Empirical software engineering experts on the use of students and professionals in experiments," *Empirical Software Engineering*, pp. 1–38, 2017.

[99] M. Claes, M. Mäntylä, M. Kuutila, and B. Adams, "Abnormal working hours: effect of rapid releases and implications to work content," in *Proceedings of the 14th International Conference on Mining Software Repositories*. IEEE Press, 2017, pp. 243–247.

[100] M. A. Chilton, B. C. Hardgrave, and D. J. Armstrong, "Person-job cognitive style fit for software developers: The effect on strain and performance," *Journal of Management Information Systems*, vol. 22, no. 2, pp. 193–226, 2005.

[101] M. Mäntylä, B. Adams, G. Destefanis, D. Graziotin, and M. Ortu, "Mining valence, arousal, and dominance: possibilities for detecting burnout and productivity?" in *Proceedings of the 13th International Conference on Mining Software Repositories*. ACM, 2016, pp. 247–258.

[102] D. F. Dinges, F. Pack, K. Williams, K. A. Gillen, J. W. Powell, G. E. Ott, C. Aptowicz, and A. I. Pack, "Cumulative sleepiness, mood disturbance and psychomotor vigilance performance decrements during aweek of sleep restricted to 4-5 hours per night." *Sleep: Journal of Sleep Research & Sleep Medicine*, 1997.

[103] D. I. K. Sjoberg, J. E. Hannay, O. Hansen, V. B. Kampenes, A. Karahasanovic, N. Liborg, and A. C. Rekdal, "A survey of controlled experiments in software engineering," *IEEE Transactions on Software Engineering*, vol. 31, no. 9, pp. 733–753, 2005. [Online]. Available: http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=1514443

APPENDIX A
ANALYSIS OF PARTICIPANTS' HOMOGENEITY

We collected the participant's perceived skills using a questionnaire administered before the experiment session. As the items were rated using a Likert-scale, we used the Kolgorov-Smirnov (K-S) test to identify differences between the RS and SD groups. It is a test of goodness of fit for the univariate case when the scale of measurement is ordinal. The defined statistical hypotheses are:

- $H0_{SKILL}$: There is no skill difference between sleep-deprived developers and developers who sleep regularly.
- $H1_{SKILL}$: There is skill difference between sleep-deprived developers and developers who sleep regularly ($SKILL_{SD} \neq SKILL_{RS}$).

Where $SKILL$ is one of the Likert items measuring perceived experience with general programming, object-oriented programming, unit testing, test-first development, use of Eclipse. The test results are reported in Table 10.

Moreover, we asked the participants to evaluate their perceived experience with respect to the rest of their classmates. The summary of the answers is reported in Figure 5.

We assessed the differences between the two groups with respect to the dependent variables of this study using the score obtained from the homework assignment they carried out before the experimental task. The task required the implementation of a formula to calculate the distance between two points on a geoid. It was completed over a varying time span (20 hours $\pm$ 5 hours). Table 12 reports the descriptive statistics for the dependent variable of this study calculated using the homework dataset.

Considering $H0P$, the same set of null hypotheses presented in Section 3.3 tested using the new dataset, none was rejected except for $H0P_{PROC}$ (see Table 11).

TABLE 10
Results of the Kolgorov-Smirnoff (K-S) test comparing subjects in the sleep-deprived (SD) and regular sleep (RS) groups in terms of experience with the relevant skills. The results for the cleaned dataset ($SD_{Cleaned}$) are reported in parentheses.

Experience	K-S test	p-value
Programming (general)	.044 (.029)	.832 (.863)
Object oriented programming	.036 (.003)	.849 (.954)
Unit testing	.178 (.027)	.672 (.867)
Test-first development	.388 (.947)	.533 (.330)
Eclipse IDE	.206 (.057)	.649 (.810)

TABLE 11
Null-hypothesis test results ($\alpha = 0.016$).

Hypothesis	Mann-Whitney U	p-value
$H0P_{QLTY}$	176.5	.30
$H0P_{ACTV}$	167	.036
$H0P_{PROC}$	129	.013

APPENDIX B
PARTICIPANTS ANSWERS TO POST-QUESTIONNAIRE REGARDING TFD

After the experimental session, we administered a post-questionnaire to the participants. We asked them to rate how difficult it was to apply TFD to the experimental task (Figure 6a), and which development approach they used (Figure 6b)—TFD or test-last development (TLD).

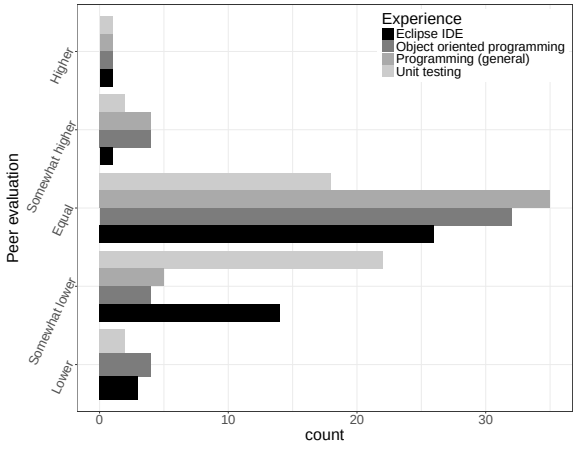
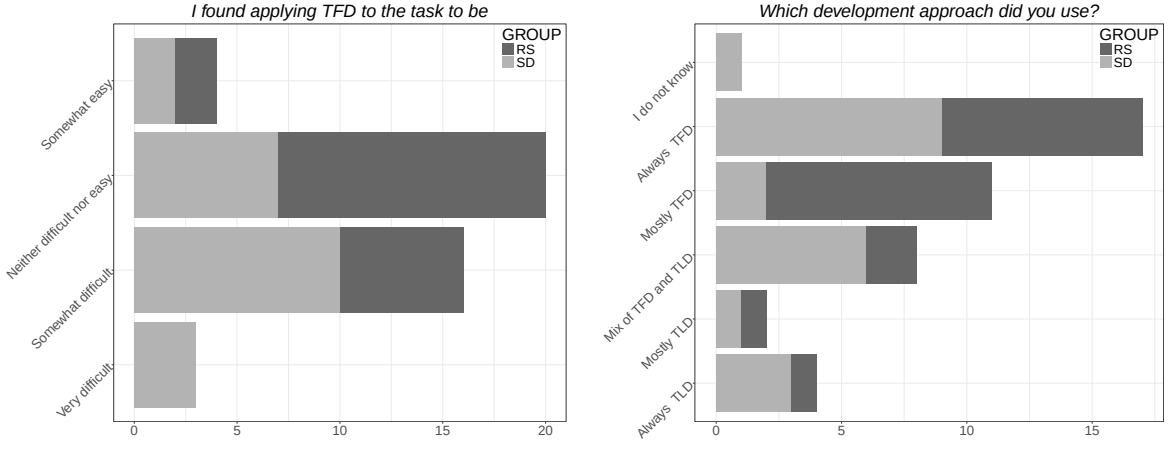


Fig. 5. Participants' perception of their experience with respect to their peers.

TABLE 12
Descriptive statistics for the dependent variable measured using the home assignment dataset.

Metric	RS							SD						
	Min	Max	Mean	Q1	Median	Q3	St. Dev.	Min	Max	Mean	Q1	Median	Q3	St. Dev.
$PAAP$	0	100	76.8	75.0	81.2	87.5	21.3	0	100	70.4	61.5	75.0	87.5	22.6
$\#episodes$	0	17	6.84	4	5	7	5.12	0	51	10.5	2.50	11	12.8	11.0
$\%conformance$	0	67	22.9	13	20	31	18.1	0	100	10.1	0	0	7.5	23.1



(a) (b)

Fig. 6. Answers to the post-questionnaire questions regarding the use of test-first development (TFD) during the experimental task, divided by group.