Measuring the Cognitive Load of Software Developers: A Systematic Mapping Study

Conference Paper · March 2019

DOI: 10.1109/ICPC.2019.00018

Lucian Gonçales
Universidade do Vale do Rio dos Sinos
32 PUBLICATIONS **106** CITATIONS

Kleinner Farias
Universidade do Vale do Rio dos Sinos
91 PUBLICATIONS **608** CITATIONS

Bruno C. da Silva
California Polytechnic State University, San Luis Obispo
43 PUBLICATIONS **237** CITATIONS

Measuring the Cognitive Load of Software Developers: A Systematic Mapping Study

Lucian Gonçales, Kleinner Farias
University of Vale do Rio dos Sinos
São Leopoldo, Brazil
lucianj@edu.unisinos.br, kleinnerfarias@unisinos.br

Bruno da Silva, Jonathan Fessler
California Polytechnic State University
San Luis Obispo, United States
bcdasilv@calpoly.edu, jvfessle@calpoly.edu

***Abstract*—*Context:* In recent years, several studies explored different facets of the developers' cognitive load while executing tasks related to software engineering. Researchers have proposed and assessed different ways to measure developers' cognitive load at work and some studies have evaluated the interplay between developers' cognitive load and other attributes such as productivity and software quality. *Problem:* However, the body of knowledge about developers' cognitive load measurement is still dispersed. That hinders the effective use of developers' cognitive load measurements by industry practitioners and makes it difficult for researchers to build new scientific knowledge upon existing results. *Objective:* This work aims to pinpoint gaps providing a classification and a thematic analysis of studies on the measurement of cognitive load in the context of software engineering. *Method:* We carried out a Systematic Mapping Study (SMS) based on well-established guidelines to investigate nine research questions. In total, 33 articles (out of 2,612) were selected from 11 search engines after a careful filtering process. *Results:* The main findings are that (1) 55% of the studies adopted electroencephalogram (EEG) technology for monitoring the cognitive load; (2) 51% of the studies applied machine-learning classification algorithms for predicting cognitive load; and (3) 48% of the studies measured cognitive load in the context of programming tasks. Moreover, a taxonomy was derived from the answers of research questions. *Conclusion:* This SMS highlighted that the precision of machine learning techniques is low for realistic scenarios, despite the combination of a set of features related to developers' cognitive load used on these techniques. Thus, this gap makes the effective integration of the measure of developers' cognitive load in industry still a relevant challenge.**

***Index Terms*—Cognitive Load; Software Engineering; Program Comprehension; Systematic Mapping Study;**

I. Introduction

The measure of cognitive load plays a key role in tasks of software engineering. Software developers are involved in activities that affect and demand their attention. Developers apply cognitive effort to mentally process the structures of the source code (which is a cognitive process known as source code comprehension) [1], [2], on resolving arithmetic problems [3], and on interpreting different abstractions levels of software artifacts [4]. The measurement of cognitive load generates valuable information (such as the level of expertise of developers) for software engineering purposes [1], [2], e.g, for accounting developers programming experience, and classification of the perceived difficulty [5], [6] during a coding task. Previous research has pointed out that measuring cognitive load in software development activities is still a problem [7]. But even so, researchers and practitioners in the industry need to select a set of cognitive load measures available in the literature and adapt them to their purpose.

However, choosing a proper cognitive load measure is difficult. There are a limited number of studies in the literature that have systematically classified the measures and their technologies related to cognitive load. Some researchers have produced literature reviews and surveys [8]–[10] related to the measurement of cognitive load. These include a general view of how mobile electroencephalograms (EEGs) could be classified according to the activity of the users [8], an analysis of biometric measures used to predict personal characteristics [9], and a review of the literature about wearable biometric recognition systems [10]. Moreover, they are far from covering the research field of software engineering. The second reason why the choice for a proper cognitive load measure is difficult is that, while some measures about cognitive load were proposed between 2008 and 2018, academic research and software industry have neglected a careful classification of such measures produced in that period [7] [6] [11]. In addition, there is still a lack of understanding about important issues, such as, which sensors and measures relate to cognitive load, the machine learning techniques used to measure cognitive load in software engineering, and which empirical methods are used to assess the cognitive load. In general, an understanding of the state-of-the-art approaches remains limited. Moreover, some studies points that measuring code comprehension through cognitive load may be a more reliable measure to classifying developer expertise rather than time of completion tasks, or correctness of produced software artifacts [1], [2], [12].

Therefore, this article aims at (1) providing a classification and a thematic analysis of studies on the measurement of developers'cognitive load, and (2) pinpoint gaps and research directions for further investigations. To this end, we carried out a Systematic Mapping Study (SMS) based on well-established guidelines (e.g., [13]–[15]). A review protocol was established by combining automatic and heuristic search in eleven widely recognized electronic databases and running a careful filtering process over a sample of 2,612 potentially relevant studies. In total, 33 articles were selected for answering nine research questions which are solved and discussed in the remainder of this article.

Section II describes the methodology used to map the current literature. Section III presents the results of the nine research questions. Section IV discusses future challenges and Section V contrasts parts of this work with related work. Section VI discusses decisions taken to minimize threats to the validity of our results. Finally, Section VII outlines final remarks, and future directions of this research.

II. SMS Planning

A. Objective and Research Questions

This work has two main objectives: (1) to provide a classification of the literature regarding the measurement of developers' cognitive load to pinpoint gaps in the literature; and (2) to identify emerging research topics for further investigation. To achieve these objectives, we defined nine research questions (RQs) to explore different aspects of available studies. Table I presents an overview of these research questions (RQs). In addition, this table also presents the description of their respective motivations, and the aspect that each research question aims to explore.

B. Search Strategy

Search String (SS). We identified search terms using the Populations, Interventions, and Outcomes (PIO) method [14], [16]. The Populations refer to the standards and technologies (such as EEG, and Brain Computer Interfaces). The Intervention terms are related to the mentioned technologies (measures of cognitive load in software development). The Outcomes are the contributions that practitioners and researchers expect related to software engineering (code, and diagram). The synonyms of each group were identified, and they were related using to the Boolean "AND" and "OR" operators as shown in II).

The Search String (SS) bellow is the combination of terms defined in Table II, thus used as default on search engines.

("brain computer interfaces" OR sensors OR devices) AND ("psychophysiological indicators" OR "brain synchronization" OR "cognitive load" OR emotions OR biometrics) AND ("software engineering" OR "software development" OR "software testing" OR "software maintenance" OR "computer programming" OR diagram OR code)

Search Engines. We chose the search engines listed in Table III because they are related to the area of computer science, cover relevant research venues, and are likely to return peer-reviewed studies written in English.

C. Exclusion Criteria

These criteria defined in this section were used to include and exclude studies retrieved from search engines during the selection process.

The Exclusion Criteria (EC) are specified bellow:

- **EC1:** the title, abstract or any other part of their content were closely related to the search keywords, however without any semantic interplay;
- **EC2:** the study was not written in English (the default language considered in our study), or a patent had been registered for it (grey literature, and studies in an initial stage were also removed);
- **EC3:** the title did not have any term specified in the search string, or even the meaning of the title is completely contrary to the purpose of the issues addressed in the research questions;
- **EC4:** the abstract did not address any aspect of the research questions;
- **EC5:** the study appears in duplicate; and
- **EC6:** the work did not address issues about measures of cognitive load on software developers or on software engineering tasks.

D. Defined Steps and Process Selection of Primary Studies

The filtering process aims at selecting representative studies based on well recognized procedures [14]. Fig. 1 shows the results of the selection process. In general, a total of 33 Primary Studies were selected from 2,612 papers. Seven steps of selection were applied to produce the final list of 33 primary studies (Listed on Appendix A).

- **Step 1: Initial search.** Retrieve the search results from the search engines using the defined the search string. In total, 2,612 studies were retrieved from digital libraries.
- **Step 2: Remove impurities.** Remove impurities obtained from the search results. For this, we applied the exclusion criteria EC1 and EC2. In this step we typically removed calls for conference paper, journal special issues, patents specifications, research reports, and materials not peer reviewed. In this step, 686 studies were removed, i.e.; about 26.3% were considered as impurities, and then 1,926 studies remained on this step.
- **Step 3: Filter by title and abstract.** Filter studies by reading title and abstract and applying the exclusion criteria EC3 and EC4. Therefore, we could filter out the studies with content not related to our research questions. In these step, 80.2% of studies were removed resulting on 165 articles.
- **Step 4: Combination.** The studies were filtered by title and abstract in the last step in different search engines directories. In this step, these studies were brought together in a unique directory.
- **Step 5: Removal of duplicates.** Usually a study can be found in two or more digital libraries. Thus, we applied EC5 for removing all duplicates, thereby assuring the uniqueness of each study. Then, 6.7% of studies (11 of 165) were duplicated and removed in this step. A total of 154 studies remained after this step.
- **Step 6: Filter by full text.** Filter studies by reading the full text and applying EC6. With we excluded studies that were not related to neither the research questions nor the main theme of this research (developers'cognitive load measurement). A total of 119 studies were removed in this step resulting on 35 studies.

TABLE I
RESEARCH QUESTIONS INVESTIGATED

Research Question	Description	Variable
RQ1: What are the types of sensors adopted to measure the cognitive load of developers?	List the sensors used to collect cognitive load from developers.	Sensors
RQ2: What metrics have been used to measure developers' cognitive load?	Discover and understand the metrics that were applied to measure developers' cognitive load.	Metrics
RQ3: What algorithms have been used to classify developer's cognitive load?	Understand the different algorithms considered for measuring cognitive load.	Algorithms
RQ4: For what purpose has developers' cognitive load been measured?	Identify for what purpose cognitive load has been measured.	Purpose
RQ5: Which tasks have been used to measure developers' cognitive load?	Classify software engineering tasks over the works published in the literature.	Software Development Tasks
RQ6: What were the artifacts used on cognitive tasks?	Determine which artifacts researchers commonly used on experimental tasks to measure developers' cognitive load.	Artifacts
RQ7: How many participants did the studies recruit to measure developers' cognitive load?	Find out the number of participants researchers used to conduct their experiments.	Number of participants
RQ8: Which research methods have been used to investigate cognitive load in software development tasks?	Understand the different types of research methods scientists on software engineering research field used to investigate developers' cognitive load.	Research methods
RQ9: Where have the studies been published?	Identify the venues in which research on developers' cognitive load have been published over the last years.	Research venues

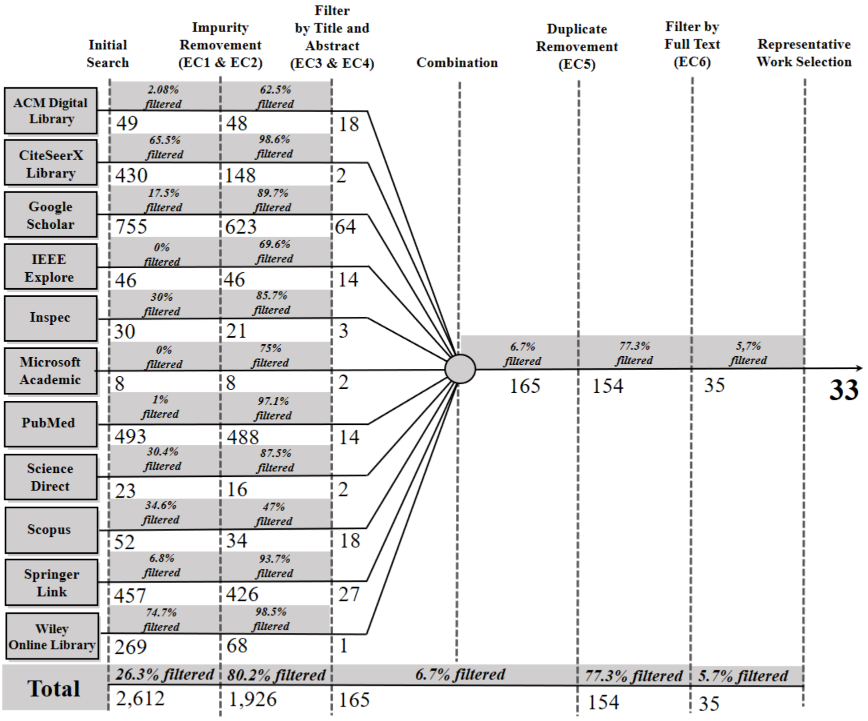


Fig. 1. Illustration of the execution of the study filtering process and the obtained results.

TABLE II
A DESCRIPTION OF THE MAJOR TERMS AND THEIR SYNONYMS

Major Terms	Synonym Terms
Sensors	"brain computer interfaces" OR sensors OR devices
Cognitive Load	"psychophysiological indicators" OR "brain synchronization" OR emotions OR biometrics
Software engineering	"software development" OR "software testing" OR "software maintenance" OR "computer programming" OR diagram OR code

- **Step 7: Selection of Representative work.** Define the final list of the primary studies. Examining these 35 studies, it was observed that some of them are extensions of the same study. Finally, 5.7% (2/35) of studies were discarded resulting on 33 works selected as the *primary studies*.

III. RESULTS

A. *RQ1: Sensors*

Table IV shows the results of RQ1. It's important to identify which sensors were related to cognitive load, and why they were applied. The main finding is that software engineering researchers prefer the electroencephalogram (EEG) sensor to collect data related to cognitive load. In particular, 55% (18

TABLE III
LIST OF THE SEARCH ENGINES

Search Engines	Link
ACM Digital Library	http://dl.acm.org/
CiteSeerX Library	http://citeseerx.ist.psu.edu/
Google Scholar	https://scholar.google.com.br/
IEEE Explore	http://ieeexplore.ieee.org/
Inspec	http://digital-library.theiet.org/
Microsoft Academic	https://academic.microsoft.com/
Pubmed	https://www.ncbi.nlm.nih.gov/pubmed/
Scopus	http://www.scopus.com/
Science Direct	http://www.sciencedirect.com/
Springer Link	http://link.springer.com/
Wiley Online Library	http://onlinelibrary.wiley.com/

of 33) of the selected works used EEG. According to Kosti et al. [7] the application of EEG in software engineering empirical studies paves the way for validation of theories about cognitive processes related to software engineering tasks, such as, programming tasks, using the electrical signals from the brain (a.k.a. brain waves). In addition, a series of work concerned with monitoring and analyzing brain waves to trace human cognitive facets during development tasks, such as, code comprehension [1], [2], [17], their relation with developers' emotions [18]–[20], and for evaluating mental effort [7], [21], was produced.

TABLE IV
CLASSIFICATION OF TECHNOLOGIES FOR MEASURING THE COGNITIVE LOAD OF DEVELOPERS

Technologies	#Studies	Percentage	Studies
EEG	18	55%	[S01][S02][S04][S05][S07][S08][S09][S10] [S12][S13][S16][S23][S27][S28][S29][S30] [S32][S33]
Combination of Sensors	12	36%	[S03][S06][S11][S15][S17][S19][S20][S21] [S22][S24][S25][S26]
Eye-Tracking	2	6%	[S14][S31]
fMRI	1	3%	[S18]
Total	33	100	

Next, only 6% (2/33) of studies applied eye-tracking, and 3% (1/33) used Functional Magnetic Ressonance Imaging (fMRI) technologies. Furthermore, we did not find any study employing only one of the following technologies – Electrocardiogram (ECG), Blood Volume Pulse (BVP), and Electrodermal Activity (EDA) – to measure cognitive load from developers. Instead, these technologies are usually combined together with the EEG, acting as a set of combined sensors, as summarized in Table IV. Table IV shows that a significant number of the selected works (36%, 12/33) combined sensors to measure developers' cognitive load. We found out that the combinations of the data from these multiple sensors were used to improve the accuracy of the results from applying machine learning techniques [22]–[24].

B. RQ2: Metrics

Table VI presents the results for RQ2. The main finding is that a majority of studies focus on indicators related to brain activity, e.g., 42% of studies (14/33) focus on measures that are related to EEGs. Measures such as Event-Related Potential (ERP) (6%, 2/33), Evented Related Desynchronization (ERD) (6%, 2/33), Fractal Design (FD) (6%, 2/33), SSVEP (6%, 1/33), and ERSP (3%, 1/33), were obtained from brain waves and correlated to cognitive load.

TABLE V
CLASSIFICATION OF PRIMARY STUDIES THAT COMBINED SENSORS

List of Primary Studies	Set of Combined Sensors
Muller 2015 [S03]	EEG, Eye Tracking, Galvanic Skin Response
Randall Minas 2017 [S06]	EEG, Skin Conductance
Lauri Ahonen [S11]	EDA, ECG
Thomas Fritz [S15]	EEG, EDA, Eye Tracking
Nargess Nourbakhsh [S17]	Eye-blink, GSR
Norman Peitek [S19]	fMRI, Eye Tracking
Manuela Zuger [S20]	EEG, Eye Blink, EDA
Thomas Fritz [S21]	EEG, Eye Tracking, EDA
Sarah Fakhoury [S22]	fNIRS, Eye Tracking
Sebastian C. Muller [S24]	Heart Rate, RR
Seolhwa Lee [S25]	EEG, Eye Tracking
Sebastian C. Muller [S26]	EEG, Eye Tracking, EDA

TABLE VI
CLASSIFICATION OF PRIMARY STUDIES BASED ON METRICS RELATED TO COGNITIVE LOAD

Metrics	#Studies	Percentage	List of Primary Studies
Frequency Bands	4	12%	[S23][S28][S29][S30]
Power Spectrum of Each Band	3	9%	[S05][S08][S27]
Event Related Desynchronization	2	6%	[S02][S06]
Event Related Potential	2	6%	[S07][S12]
Fractal Dimension	2	6%	[S04][S13]
Volume of Interest	2	6%	[S18][S19]
Eye Fixation	1	3%	[S14]
Event Related Spectral Perturbation	1	3%	[S10]
Individual Alpha Frequency	1	3%	[S01]
Index of Cognitive Activity	1	3%	[S31]
Steady State Visual Evoked Potential	1	3%	[S32]
Set of Metrics	13	40%	[S03][S09][S11][S15][S16] [S17][S20][S21][S22][S24] [S25][S26][S33]
Total	33	100%	

The Event-Related Potential (ERP) was used to verify the effect of the use of the software system on the developers's mental load [21], and to classify emotions [25]. The Event-Related Disyncrhonization (ERD) was measured to calculate the mental load during source code comprehension [1], [2]. Steady State Visual Evoked Potential (SSVEP) was used for analyzing visual stimuli [26], Event Related Spectral Perturbation (ERSP) was utilized to analyze the influence of stress during activity switching [27], and Fractal Dimension (FD) was used to analyze and classify emotions from the brain waves [20]. 21% (7/33) of the studies were concerned with using the frequency bands to analyze the cognitive load of developers. In a software engineering context, few conclusions could be derived from the visual analysis of raw EEG signals regarding cognitive effort. However, some studies have already evidenced that using brain waves in combination with machine learning techniques can produce valuable outcome measures.

Rather than relying on a single metric, a relevant portion of works (40%) applied a set of metrics to measure developers' cognitive load. Table VII summarizes the selected studies grouped by set of combined metrics. It is worth noting that cognitive load analysis is not limited to brain-centric data.

TABLE VII
CLASSIFICATION OF PRIMARY STUDIES BASED ON SET OF COMBINED METRICS

List of Primary Studies	Set of Combined Metrics
Sebastian Muller 2015 [S03]	Alpha, Beta/Theta, Electrodermal activity, Pupil Size, MeanSCL
Ahamad Subani 2017 [S09]	Amplitude Assimetry, Absolute Power, Relative Power, Coherence, Phase Lag
Lauri Ahonen [S11]	Heart Rate, Standart Deviation of Normal to Normal, Skin Conductance Response, and Skin Conductance Level
Thomas Fritz 2014 [S15]	Power Spectrum of each band, EDA, and Eye-Blink Measures
Daniela Cernea 2012 [S16]	Eye tracking, Emotions
Nargess Nourbakhsh 2013 [S17]	Accumulative GSR, power spectrum of GSR, blink number and blink rate
Manuela Zuger 2015 [S20]	Variations of brain waves, Heart hate, skin temperature
Thomaz Fritz 2016 [S21]	Frequency bands, EDA, HR, HRV, Blood Volume Pulse, Respiratory Rate, eye blinks, fixations, pupil size
Sarah Fakhoury 2018 [S22]	HbT, HbO, HbR ,Oxy
Sebastian Muller 2016 [S24]	HRV, HR, Respiratory Rate, Skin Temperature, EDA
Seowlwa Lee 2017 [S25]	Frequency bands, eye blink, pupil size
Sebastian Muller 2015 [S26]	Frequency bands, EDA, HR, HRV, Blood Volume Pulse, Respiratory Rate, eye blinks, fixations, pupil size
Yueran Yuan [S33]	Frequency bands, mean, variance, min, max, skew, first-order-polynomial-fit, and second-order-polynomial-fit

Overall, this classification evidences that primary studies applicated a different combination of metrics related to cognitive load. Furtermore, the term cognitive load is commonly know to be misused in software engineering. Some primary studies use cognitive load as an absctract term for "mental effort" [7], [28], while others base on the context of cognitive load theory, in particular using concepts such as extraneous and intrinsic cognitive load [1], [2].

C. RQ3:Algorithms

TABLE VIII
ALGORITHMS AND APPROACHES FOR MEASURING THE COGNITIVE LOAD OF DEVELOPERS

Category	Machine Learning Algorithms	#Studies	Percentage	List of Primary Studies
Classication	Support Vector Machines (SVM)	5	15%	[S05][S07][S23][S25][S29]
	Naive Bayes	5	15%	[S15][S20][S21][S26][S33]
	Multi-algorithms of Classification	3	9%	[S09][S17][S32]
	K-means	1	3%	[S06]
	Decision Tree Classifier	1	3%	[S03]
	Logistic Regression	1	3%	[S01]
	Neural Network	1	3%	[S28]
	Random Forest Learners	1	3%	[S24]
	Relevance Vector Machines	1	3%	[S23]
Regression	Linear regression	1	3%	[S13]
	Does not use	13	40%	[S02][S04][S10][S11][S12][S14][S16][S18][S19][S22][S27][S30][S31]
	Total	33	100%	

The results in Table VIII show that studies tend to use more Classification Techniques than Regression techniques. Only 1% of the primary studies (1/33) applied Regression techniques to their approaches. This implies that studies in software engineering which measure the cognitive load of developers are focused on classification problems. Thus, the literature is focusing on prediction problems such as level of understanding, task difficulty, and emotion recognition. Subsection III-D describes the prediction problems which the software engineering researchers were focusing on classifying. Table VIII shows that for classifying these groups and subsets, they prefer to use Support Vector Machines (SVM) and Naive Bayes (NB): about 30% of primary studies (10/33) implemented these machine learning algorithms. The literature shows that these algorithms both achieved higher precision and accuracy in predicting situations where developers face high task difficulty [5], [23], and lower productivity [6]. A small portion of works focused on other classification strategies such as the Decision Tree Classifier (3%, 1/33), K-means (3%, 1/33), and Random Forest Learners (3%, 1/33).

Moreover, some works used more than one machine learning algorithm. Table VIII shows that about 9% (3/33) of works applied a set of multiple machine learning algorithms. Table IX summarizes the three articles with the machine learning techniques applied. This also reinforces the preference of SVM and Naive Bayes. Fritz and Muller [6] concluded that Naive Bayes has higher precision on data prediction than other alternative machine learning techniques. In addition, they highlighted that classification techniques are less precise during field experiments in realistic scenarios than those controlled experiments in labs. Finally, classification of multi-classes tends to decrease the precision and accuracy of classifiers. This means that it would be difficult to identify precisely more than two classes of expertise of developers.

TABLE IX
LIST OF PRIMARY STUDIES THAT APPLIED MORE THAN ONE MACHINE LEARNING TECHNIQUE

List of Primary Studies	Classification Algorithms
Ahmad Subhani 2017 [S09]	Decision Tree, Naive Bayes and K-Nearest Neighbor
Nargess Nourbakhsh 2013 [S17]	Support Vector Machines, and Naive Bayes
Zafer 2018 [S32]	Support Vector Machines, Naive Bayes, Linear Regression

D. RQ4: Purpose

Table X presents the classification of studies according to purposes for which they evaluated cognitive load.

The goal most explored by the primary studies is to understand the correlation between cognitive load and code comprehension. 25% (8/33) of primary studies aims investigating how the developers brain waves signals can be used to understand the process of comprehension of source code. The literature points that measuring code comprehension through cognitive load may be a more reliable alternative to classifying developer expertise rather than time of completion tasks, or correctness of produced software artifacts [1], [2]. Next, 18% (6/33) of the primary studies aimed to understand the difficulty of tasks based on cognitive load measurements of developers , specifically. Detecting the difficulty of an activity while the

TABLE X
CLASSIFICATION OF PRIMARY STUDIES BASED ON PURPOSE

Purpose	#Studies	Percentage	List of Primary Studies
Code Comprehension	8	25%	[S01][S02][S14][S18][S19][S22][S31][S33]
Emotion Recognition	6	18%	[S04][S05][S06][S07][S08][S13]
Task Difficulty	6	18%	[S15][S23][S25][S26][S27][S29]
Cognitive demand	3	9%	[S12][S17][S30]
Productivity	2	6%	[S03][S21]
Stress level	2	6%	[S09][S10]
Authentication	1	3%	[S28]
Code Quality	1	3%	[S24]
Interruptiblity	1	3%	[S20]
Pair-Dynamic level	1	3%	[S11]
Performance	1	3%	[S32]
Satisfaction	1	3%	[S16]
Total	33	100%	

developer performs it is information that has the potential to be used for various goals. According to Lee [12] the difficulty can be used to update the time estimate for completing a task and predict propensity for errors.

Furthermore, 18% (6/33) of primary studies used the cognitive load of developers with the objective of classifying and recognizing emotions. Some studies are dedicated to developing methods to detect emotions more precisely [25] [29]. Next, a small portion of primary studies focused on using the cognitive load data to assess stress levels (3%, 1/33), interruptibility (3%, 1/33), performance (3%, 1/33), productivity levels (3%, 1/33), and for predicting productivity (3%, 1/33).

E. RQ5: Software Engineering Tasks

Table XI presents the classification of studies according to the tasks for which they evaluated cognitive load.

TABLE XI
CLASSIFICATION OF PRIMARY STUDIES BASED ON TASKS

Tasks	#Studies	Percentage	List of Primary Studies
Programming	16	48%	[S01][S02][S03][S06][S11][S14][S15][S18][S19][S20] [S21][S22][S24][S25][S26][S30]
Observation	6	18%	[S05][S07][S13][S16][S27][S28]
Arithmetic Equations	3	9%	[S17][S23][S29]
Multitasks	3	9%	[S08][S12][S32]
Listen	2	6%	[S04][S09]
Reading	2	6%	[S31][S33]
Choose	1	3%	[S10]
Total	33	100%	

The main finding is that studies concentrated on investigating programming tasks. A near majority of studies (48%, 16/33) focused on investigating the cognitive load of practitioners during programming tasks. Next, 9% of studies (3/33) utilized tasks focusing on solving arithmetic equations. The Listening and Reading tasks were each considered in 6% (2/33) of the primary studies. In addition, 9% (3/33) of studies used two different tasks in their experiments.

To sum up, the main research gap concerning these listed tasks is the lack of other software development cycle phases, such as analysis, design, and deployment phases. The primary studies which evaluated cognitive load on programming tasks usually required developers to complete a missing part of the source code. In others, they solved an algorithm mentally. None of the primary studies reported tasks related to software development phases, such as analysis, design, and deployment phases.

F. RQ6: Artifacts

Table XII show the answers of RQ6. The results shows that the most adopted artifacts by primary studies are source code (48%, 16/33), images (12%, 4/33), and math equations (9%, 3/33). A minority of primary studies used artifacts such as texts (9%, 3/33), reports (3%, 1/33), videos (3%, 1/33), images (12%, 4/33), sounds (3%, 1/33), or a combination of sounds and images (3%, 1/33), in experiments.

TABLE XII
CLASSIFICATION OF PRIMARY STUDIES BASED ON ARTIFACTS

Artifact	#Studies	Percentage	List of Primary Studies
Code	16	48%	[S01][S02][S03][S06][S11][S14][S15][S18][S19] [S20][S21][S22][S24][S25][S26][S30]
Images	4	12%	[S05][S07][S10][S27]
Equation	3	9%	[S17][S23][S29]
Texts	3	9%	[S28][S31][S33]
Localization	1	3%	[S12]
Product	1	3%	[S16]
Reports	1	3%	[S09]
Sounds	1	3%	[S04]
Sounds and Images	2	6%	[S08]
Video	1	3%	[S13]
Total	33	100%	

The primary studies were concerned mainly with using source code artifacts in experiments to evaluate the cognitive load of collaborators. In particular, 48% of studies (16/33) adopted source code artifacts. Furthermore, the primary studies chiefly used source code written in Java [2], [4], [23], [28], [30]–[33], with one study using code written in C# [5]. Thus, studies tend to choose a standard programming language for their experiments. This suggests that the experiments have not investigated the impact of the syntactic difference between the languages on the cognitive load. This would be helpful in highlighting which programming languages tend to be easier, or which require less cognitive load from the developers. Only one study used both C# and Java languages [34], but they where evaluated sepparetly, i.e., aiming to adapt the experiment to the developers' background. The difference in code structures has the potential to impact the cognitive load. Minas et. al. [4] found out that methods concentrating a majority of functionalities of a system demands less cognitive load than code-oriented design.

Moreover, an evident research gap is that several artifacts related to software engineering have not yet been used in experiments, such as the wide range of UML software diagrams [35]. Future studies have potential in exploring the impact that artifacts of different abstraction levels on the cognitive load. Recent studies measured it in terms of time effort, not in terms of indicators obtained from biological features [1], [2].

G. RQ7: Participants

Table XIII shows the number of participants that the primary studies used in their experiments. To sum up, a majority of works collected the cognitive load from less than 20 participants. 48% (16/33) of studies recruited on a range of between eleven and twenty participants. Furthermore, 12% (4/33) of studies recruited ten or fewer participants. Next, 18% (6/33) recruited between twenty-one and thirty participants. A small portion of works recruited more than thirty participants. Specifically, 12% (4/33) of primary studies fell into a range of 31 to 40 participants, and only 9% (3/33) were in a range of 41 to 50 participants. We did not find any study that had more than 50 participants.

TABLE XIII
CLASSIFICATION OF PRIMARY STUDIES BASED ON NUMBER OF PARTICIPANTS

#Participants	#Studies	Percentage	List of Primary Studies
0-10	4	12%	[S07][S20][S24][S30]
11-20	16	48%	[S03][S05][S08][S10][S12][S13][S15][S16][S17][S18] [S21][S22][S26][S27][S29][S33]
21-30	6	18%	[S04][S14][S19][S23][S31][S32]
31-40	4	12%	[S01][S02][S25][S28]
41-50	3	9%	[S06][S09][S11]
Total	33	100%	

The literature does not indicate an ideal number of participants to perform the experimental activities with EEGs. However, a minimum number of participants is required to generate consistent sampling. From consistent sampling it is possible to generalize more reliable conclusions. In an experimental task the participants must generate at least 25 to 30 samples. Achieving this number of samples is a challenging task mainly because of the difficulties encountered in recruiting participants, such as the limits which must be placed on participants' time, and the struggle of choosing the ideal participants' profiles for the experiment. For example, the classification of the data in different participants' profiles of the users fragments the sample data. Igor Crk et. al. [2] classified the participants into 4 different classes (ranked from 0 up to 4, i.e., respectively from the least to the most experienced). Although there were a large number of participants in general, this classification resulted in around 10 to 12 participants in each class. This fragmentation causes a reduced number of samples. Finally, an existing database with biometric data related to software activities would facilitate the acquisition of a good sample count.

H. RQ8: Research Methods

This research question classifies the research methods used by primary studies. The research methods were classified as controlled experiment, proposal of solution, philosophical paper, validation, and personal experience. These categories were based on [36]. Table XIV shows the answers of RQ8 according these categories.

A Majority of primary studies (67%, 22/33) were controlled experiments. Second, some studies (25%, 8/33) were proposals of solutions, i.e., they focused on proposing new solutions and resolving problems inherent to research gaps, e.g., Peitek et al. [33] propose the utilization of eye tracking and fMRI to measure program comprehension. Moreover, numerous papers propose methods of detecting emotions through the use of EEGs [4], [19], [20], [25], [29]. Emotions are important factors that impact many software development tasks, and conversely, software development tasks can have an impact on developers' emotions. 1% (1/33) were opinion papers, i.e., when authors give their opinion about future problems on their research field.

Specifically, numerous papers propose methods of detecting emotions through the use of EEGs [4], [19], [20], [25], [29]. Emotions are important factors that impact on many software developments tasks, and conversely, software development tasks can have an impact on developers' emotions.

TABLE XIV
CLASSIFICATION OF PRIMARY STUDIES BASED ON RESEARCH METHOD

Research Methods	Number of Studies	Percentage	Studies
Controlled Experiment	24	73%	[S01][S02][S03][S10][S11][S12][S14][S15] [S16][S17][S18][S20][S21][S22][S23][S24] [S25][S27][S28][S29][S30][S31][S32][S33]
Proposal of Solution	8	25%	[S04][S05][S06][S07][S08][S09][S13][S26]
Opinion Paper	1	3%	[S19]
Total	33	100%	

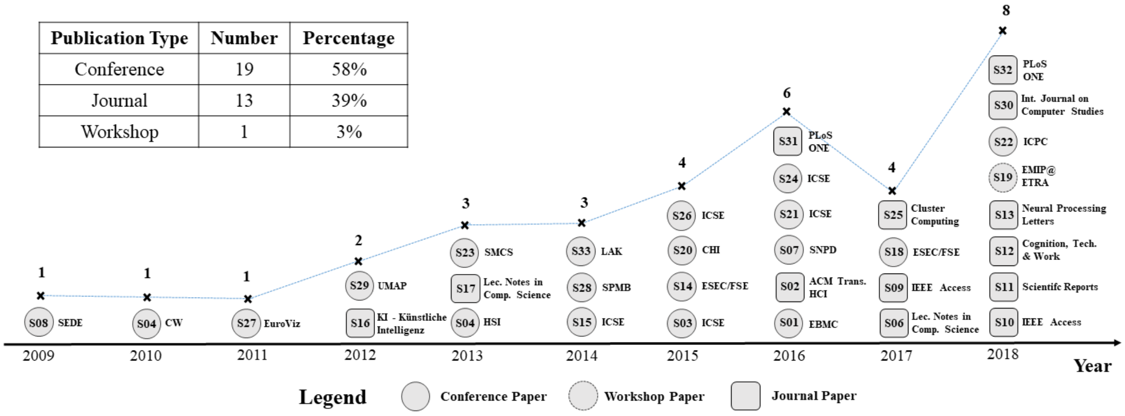
Moreover, no validation study was identified among the selected papers. This implies that there is a gap of replicating the proposed solutions on realistic scenarios (industry).

I. RQ9: Research Venue

Fig. 2 shows a ten-year chronological plot of the primary studies. Each article was attributed a value of one point. Based on this, the contribution index was calculated by adding up the points of each article in their corresponding years.

This **Contribution index** (dashed blue line) suggests that research about cognitive load raised significantly between 2015 and 2018. The contribution rate was higher in 2018 (8 points), followed in 2016 (6 points). Moreover, by coincidence the research focusing cognitive load increased in the software engineering research field, just after the release of technologies, such as, Emotiv, and NeuroSky. These technologies are widely cited in the published articles after the 2014.

Venue of publication. The results in Fig. 2 show that 58% (19/33) of primary studies were obtained from conferences. Four primary studies (S03, S15, S21, S24) were published in ICSE, two studies (S14, S18) were published in ESEC/FSEC, and one study (S22) was published in ICPC. Next, 39% (13/33) were obtained from journals. Among these, two of them (S31 and S32) were issued in PLOS One, another two papers (S09, S10) in IEEE Access , and one more (S11) in Scientific Reports. Finally, 3% (1/33) were published in workshops, i.e., the primary study (S19) was published at the EMIP workshop co-located with the ETRA conference.



Publication Type	Number	Percentage
Conference	19	58%
Journal	13	39%
Workshop	1	3%

Fig. 2. The research venues that primary studies were published over the last years.

J. Taxonomy

Fig. 3 presents the proposed taxonomy. This taxonomy was derived from the results presented. They are grouped into eight main categories: (i) which sensors were used to collect data related to cognitive load; (ii) which metrics was extracted from raw data of the sensors; (iii) the algorithms to classify the metrics obtained; (iv) purposes to measure cognitive load by primary studies; (v) the tasks used to evaluate cognitive load of participants; (vi) the artifacts which participants interacted during the tasks; (vii) range of participants who attended on the experiments/tasks; and (viii) research methods of the selected primary studies. The black color represents the root of the taxonomy, whereas the blue color represents the subcategories of the cognitive load on software engineering.

Note that this taxonomy is also a general view of this research field and through this overview there is the distribution of answers. For example, there is a concentration of classification techniques rather than regression ones (iii), and that although using other sensors than the EEG, the types of metrics (ii) analyzed are related to the brainwave data, while the metrics related to other sensors was concentrated in a set of multiple metrics.

IV. Discussion and Challenges for Future Research

This section presents a discussions and further challenges we identified in selected primary studies.

(1) A quality model for measuring the developers' cognitive load. A quality model is important for defining a set of attributes that defines a cognitive load measure. These attributes are taken into account in empirical studies that measure the cognitive load in software engineering. Some quality models have been proposed in the last several decades [37]. However, these quality models aim at software modeling in general rather than measuring the developers' cognitive load specifically. Further studies might extend these quality models for the purpose of measuring developers' cognitive load. Therefore, some future research should answer these challenging questions: (1) What attributes should the quality model aggregate to measure the cognitive load of the developers? (2) Does only one quality model serve as a guide for all different kinds of experiments in software engineering? (3) Or should specific models be proposed?

(2) Detection of Quality Concerns Based on Cognitive Indicators. The literature of software engineering research has reported that cognitive indicators can be used to detect quality concerns in source code such as bugs, and inappropriate comments can also be detected using cognitive indicators [23]. In practice, the predictions of quality concerns have been made by machine learning algorithms. Muller [23] applied a machine learning algorithm called Random Forest Learners. This algorithm successfully predicted quality concerns based on cognitive indicators. Despite this, the precision of the algorithm was very low, around 40% to 50%. Therefore, improving the precision of the detection of quality concerns is a potential challenge.

Muller [23] detected six types of quality concerns such as coding style violations, bugs, missing tests, insufficient exception handling, and inadequate comments. Future research could also be directed to detecting quality concerns related to bad smells [38]. Some examples of bad smells are Long Method, Large Class, Large Parameter List, and Primitive Obsessions.

(3) Sentiment and emotion detection of developers from biometrics. The data gathered from biometrics sensors, such as heart rates, eye blinks, and EEGs, can be used to detect emotions. Positive and negative emotions are part of a software developer's daily life. Thus, it is important to detect and classify the emotions of developers. The range of emotions a software developer can experience can affect performance factors during a software task, such as the correctness of artifacts, or the effort applied to resolve change tasks.

In addition, some works pointed out that emotions have impacts on software productivity. Future research could focus on a prediction model to classify emotions using machine learning or deep learning techniques. Finally, another research gap is to build a system which recommends tasks according to the developer's emotional state.

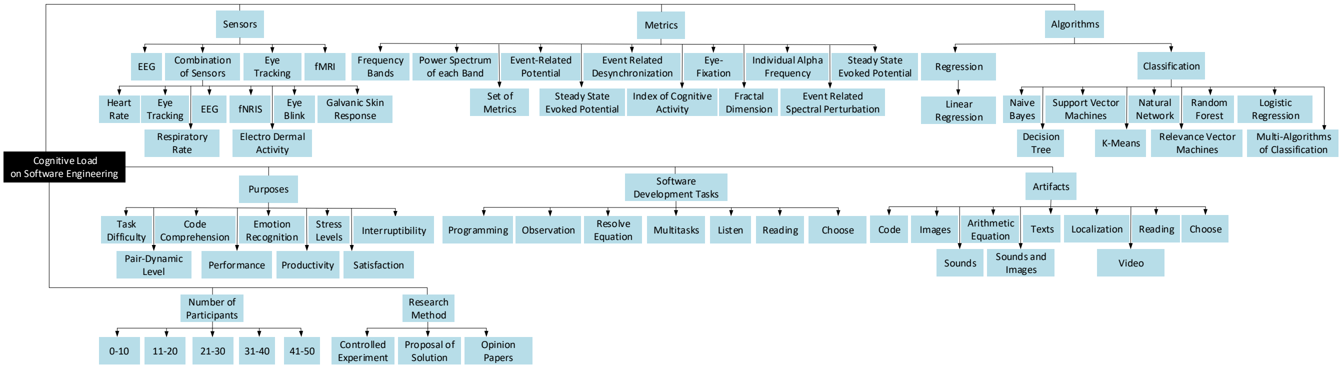


Fig. 3. Taxonomy of cognitive load in software engineering.

TABLE XV
MAIN FEATURES OF RELATED WORK THAT WAS EXPLORED

ID	Title & Reference	Research Method	Number of Studies	Search Protocol	Domain	Goal(s)	Range Period	Questions/Dimensions Addressed
RW01	A Survey of Wearable Biometric Recognition Systems [10]	Survey	Does not Specify	No	Biometric Recognition	Explore specific issues that resides on wearable biometric recognition systems	Until 2016	Categorization of wearable sensors, technique for processing raw signals, machine learning techniques, and discussion of issues such as the biosignal quality, and lack of public datasets.
RW02	Predictive biometrics: a review and analysis of predicting personal characteristics from biometric data [9]	Literature Review	∼100	No	Prediction of Personal Characteristics	Classify and analyze the more general predictive capabilities of biometric data	Until 2016	Categorization of studies that predicts low (such as gender and age estimation) and high level individual characteristics, machine learning techniques to classify users, discussion and analysis about the studies, and Future development of predictive biometrics.
RW03	Categorization of Mobile EEG: A Researcher's Perspective [8]	Literature Review	29	Yes	EEG data capture	Categorize Mobile or ambulatory EEG devices	Until 2017	A development of a scheme to classify EEG Devices, Discussions of results and shortcomings on the scheme for classification
RW04	Eye tracking in library and information science: a literature review [39]	Systematic Literature Review	59	Yes	Eye Tracking metrics	Classify studies that apply eye tracking technology on the field of information science and library	2005 to 2015	Eye-tracker devices, Eye-tracking measures, method for data-analysis, and methods for collecting supplementary data of eye track devices.
RW05	Psychophysiological measures of human cognitive states applied in human computer interaction [40]	Literature Review	∼20	No	Human Computer Interaction	Review and classify the measures of human cognitive states applied in the HCI.	Until 2010	Measures of cognitive states, and the possible practical applications.

V. RELATED WORKS

This section summarizes some studies that are closely associated with this research. This summarization is presented in Table XV. Each study is discussed in detail below.

RW1 [10] carried out a review about wearable biometric recognition systems. It covered articles published up until 2016 and did not follow a systematic research protocol. The authors focused on wearable biometrics signals for recognition systems. They investigated the types of wearable sensors and the similarities of measures across different signals, and they also listed the machine learning techniques used to classify signals.

RW2 [9] introduced a literature review about the potential capabilities of biometrics for predicting personal characteristics of users, such as age and gender, i.e, low-level characteristics. The authors analyzed about 100 articles to conduct the literature review. They described the biometric sources, databases, and the number of participants which the studies used to collect information.

RW3 [8] reviewed the literature and created a scheme to classify "wireless EEG" devices. This is because there is a wide range types of EEGs. The authors proposed classifying the EEG devices based on dimensions such as the number of channels, the sample rate, battery life, and electrode type. Equipment such as Emotiv [41] and NeuroSky [42], widely used in software engineering experiments, were classified as equipment with low portability, precision and sampling rate.

RW4 [39] conducted a systematic literature review about eye tracking within the research field of library and information science. It covered a total of 59 primary studies published between 2005 and 2015. The authors limited the search to primary studies from two databases. In this study they classified the selected studies based on metrics, research venues, research methods, and number of participants.

RW5 [40] made a literature review of the psycho-physiological measures applied to human-computer interaction, i.e., to evaluate the experience of the user in relation to computers. They analyzed about 20 studies published up until 2010. In this study the authors classified articles based on the measures they found in the literature of HCI, and the possible applications of psycho-physiological measures.

To sum up, our work is the first to conduct a systematic mapping about the measures of cognitive load in the software engineering domain. For this, we adopted a well-defined search protocol, and found 33 primary studies published through 2018.

VI. THREATS TO VALIDITY

Threats to *construct validity* concern the measures taken to avoid bias in the selection of studies. First, the keywords that formed the search string were extracted from articles related to the topic. Second, the search string was based on well-defined practices in the literature [13], [15]. Moreover, this search string was used in the main search engines related to the computer science and software engineering research field. The authors also manually reviewed each step of the selection process.

Threats to *internal validity* concern how the internal aspects were established to avoid affecting the findings, i.e., that the findings were derived from an adequate analysis of the results. For this, we classified terms based on the literature of software engineering, as well as based on concepts of cognitive load present in already-published papers related to this research field [5], [6].

Threats to *conclusion validity* concern the guidelines followed to avoid hasty and premature conclusions of this research. The conclusions were formulated right after all the results were collected, thereby eliminating the fishing problem [43], i.e., we did not expect any specific conclusion before analyzing the results.

VII. Conclusion

We conducted a Systematic Mapping Study about measures of cognitive load in software engineering. For this, we first carried out an in-depth selection process on 2,612 potentially relevant studies, and 33 were selected as primary studies. The results of classification of these primary studies were presented followed by a list of outstanding future challenges.

In essence, the findings of this article point that the application of cognitive load in the software engineering research field is in an initial stage and has potential to future research. Overall, the primary studies selected focused on investigating programming tasks, and applied machine learning classification techniques to identify the programmer's difficulty level, and their code-level comprehensibility.

Thus, this area has potential for future research. Besides corroborating the findings already achieved in the literature, these include: investigating ways to integrate the cognitive load measurement on industry environments effectively; and investigating the impact of software artifacts of different abstraction levels such as software models versus code-level, on the comprehension of software developers.

Acknowledgement

This study was financed in part by the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) - Finance Code 001.

Appendix

Selected Primary Studies

S01 I. Crk and T. Kluthe, "Assessing the contribution of the individual alpha frequency (IAF) in an EEG-based study of program comprehension," In Int. Conf. of the IEEE Eng. in Medicine and Biology Society (EMBC), 2016, 4601-4604.

S02 I. Crk, T. Kluthe, and A. Stefik. "Understanding Programming Expertise: An Empirical Study of Phasic Brain Wave Changes". ACM Trans. Comput.-Hum. Interact. 23, 1, 2015.

S03 S. C. Müller and T. Fritz, "Stuck and Frustrated or in Flow and Happy: Sensing Developers' Emotions and Progress," Int. Conf. on Soft. Eng., Florence, 2015, 688-699.

S04 Y. Liu, O. Sourina and M. K. Nguyen, "Real-Time EEG-Based Human Emotion Recognition and Visualization," 2010 Int. Conf. on Cyberworlds, 2010, 262-269.

S05 N. Jatupaiboon, S. Pan-ngum and P. Israsena, "Emotion classification using minimal EEG channels and frequency bands," Int. Joint Conf. on Comp. Science and Soft. Eng. (JCSSE), Maha Sarakham, 2013, 21-24.

S06 R. K. Minas, R. Kazman, & E. Tempero. "Neurophysiological Impact of Software design processes on software developers." In Int. Conf. on Augmented Cognition, 2017, 56-64.

S07 J. Jiang, Y. Zeng, L. Tong, C. Zhang and B. Yan, "Single-trial ERP detecting for emotion recognition." IEEE/ACIS Int. Conf. on SNPD, 2016, pp. 105-108.

S08 Wahab, A., Kamaruddin, N., Palaniappan, L. K., Li, M., & Khosrowabadi, R. "EEG signals for emotion recognition." Jour. of Comp. Methods in Sciences and Eng., 2010.

S09 A. R. Subhani, W. Mumtaz, M. N. B. M. Saad, N. Kamel and A. S. Malik, "Machine Learning Framework for the Detection of Mental Stress at Multiple Levels," in IEEE Access, vol. 5, pp. 13545-13556, 2017.

S10 C. T. Lin, J. T. King, J. W. Fan, A. Appaji and M. Prasad, "The Influence of Acute Stress on Brain Dynamics During Task Switching Activities," in IEEE Access, vol. 6, pp. 3249-3255, 2018.

S11 L. Ahonen, B. U. Cowley, A. Hellas, & K. Puolamäki "Biosignals reflect pair-dynamics in collaborative work: EDA and ECG study of pair-programming in a classroom environment." Scientific reports, 2018, n. 8, v. 1.

S12 I. Solís-Marcos, K. & Kircher, "Event-related potentials as indices of mental workload while using an in-vehicle information system." Cognition, Technology & Work, 2018, 1-13.

S13 Li, Y., Zheng, W., Cui, Z., Zong, Y., & Ge, S. "EEG Emotion Recognition Based on Graph Regularized Sparse Linear Regression." Neural Processing Letters, 1-17.

S14 K. Kevic, B. M. Walters, T. R. Shaffer, B. Sharif, D. C. Shepherd, and T. Fritz. "Tracing software developers' eyes and interactions for change tasks." In Proc. of the 2015 10th ESEC/FSE, 2015, 202-213.

S15 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 Int. Conf. on Software Engineering (ICSE). 2014, 402-413.

S16 D. Cernea, P. S. Olech, A. Ebert, & A. Kerren "Measuring subjectivity." KI-Künstliche Intelligenz, 2012, n. 26, v. 2, 177-182.

S17 N. Nourbakhsh, Y. Wang, & F. Chen. "GSR and blink features for cognitive load classification," In IFIP Conf. on HCI, 2013, 159-166.

S18 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 Proc. of the 2017 11th ESEC/FSE, 2017, 140-150.

S19 N. Peitek, J. Siegmund, C. Parnin, S. Apel, and A. Brechmann. "Toward conjoint analysis of simultaneous eye-tracking and fMRI data for program-comprehension studies." In Workshop on Eye Mov. in Progr. (EMIP), 2018.

S20 M. Züger and T. Fritz. "Interruptibility of Software Developers and its Prediction Using Psycho-Physiological Sensors." In ACM Conf. on CHI. 2015, 2981-2990.

S21 T. Fritz and S. C. Müller, "Leveraging Biometric Data to Boost Software Developer Productivity," Int. Conf. on Software Analysis, Evolution, and Reengineering (SANER), 2016, 66-77.

S22 S. Fakhoury, Y. Ma, V. Arnaoudova, & O. Adesope "The Effect of Poor Source Code Lexicon and Readability on Developers' Cognitive Load." In Proc. Int. Conf. Program Comprehension (ICPC), 2018.

S23 A. Sinharay, D. Chatterjee and A. Sinha, "Evaluation of Different Onscreen Keyboard Layouts Using EEG Signals," Int. Conf. on Systems, Man, and Cybernetics, 2013, 480-486.

S24 S. C. Müller and T. Fritz. "Using (bio)metrics to predict code quality online." Int. Conf. on Software Engineering (ICSE). 2016, 452-463.

S25 S. Lee, D. Hooshyar, H. Ji, K. Nam, & H. Lim. "Mining biometric data to predict programmer expertise and task difficulty.", Cluster Computing, 2017, 1-11.

S26 S. C. Müller "Measuring software developers' perceived difficulty with biometric sensors." Int. Conf. on Software Engineering, pp. 887-890, 2015, IEEE Press.

S27 E. W. Anderson, K. C. Potter, L. E. Matzen, J. F. Shepherd, G. A. Preston, & C. T. Silva. "A user study of visualization effectiveness using EEG and cognitive load." In Comp. Graphics Forum, v. 30, n. 3, 2011, 791-800.

S28 Q. Gui, Z. Jin and W. Xu, "Exploring EEG-based biometrics for user identification and authentication," IEEE Signal Processing in Medicine and Biology Symposium (SPMB), 2014, 1-6.

S29 F. C. Galán and C. R. Beal. "EEG estimates of engagement and cognitive workload predict math problem solving outcomes." In Int. Conf. on User Modeling, Adaptation, and Personalization (UMAP). 51-62.

S30 M. V. Kosti, K. Georgiadis, D. A. Adamos, N. Laskaris, D. Spinellis, & L. Angelis. "Towards an affordable brain computer interface for the assessment of programmers' mental workload." Int. Journal of HCI, 2018, 115, 52-66.

S31 V. Demberg, & A. Sayeed. "The frequency of rapid pupil dilations as a measure of linguistic processing difficulty." PLOS one, v. 11, n.1, 2016.

S32 Z. Iscan, & V. Nikulin. "Steady state visual evoked potential (SSVEP) based brain-computer interface (BCI) performance under different perturbations." PLOS one, v. 13, n 1, 2018.

S33 Y. Yuan, K. Chang, J. N. Taylor, and J. Mostow. "Toward unobtrusive measurement of reading comprehension using low-cost EEG." In Int. Conf. on Learning Analytics And Knowledge (LAK). 2014, New York, 54-58.

References

[1] I. Crk and T. Kluthe, "Assessing the contribution of the individual alpha frequency (IAF) in an EEG-based study of program comprehension," in

2016 38th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC), Aug 2016, pp. 4601–4604.

[2] I. Crk, T. Kluthe, and A. Stefik, "Understanding programming expertise: An empirical study of phasic brain wave changes," *ACM Trans. on Computing-Human Interaction*, vol. 23, no. 1, pp. 2:1–2:29, dec 2015.

[3] A. Sinharay, D. Chatterjee, and A. Sinha, "Evaluation of different on-screen keyboard layouts using EEG signals," in *2013 IEEE International Conference on Systems, Man, and Cybernetics*, Oct 2013, pp. 480–486.

[4] R. K. Minas, R. Kazman, and E. Tempero, "Neurophysiological impact of software design processes on software developers," in *Augmented Cognition. Enhancing Cognition and Behavior in Complex Human Environments*. Springer International Publishing, 2017, pp. 56–64.

[5] 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 *Proc. of the 36th Int. Conference on Software Engineering*, ser. ICSE 2014, 2014, pp. 402–413.

[6] T. Fritz and S. C. Müller, "Leveraging biometric data to boost software developer productivity," in *2016 IEEE 23rd International Conference on Software Analysis, Evolution, and Reengineering (SANER)*, vol. 5, March 2016, pp. 66–77.

[7] M. V. Kosti, K. Georgiadis, D. A. Adamos, N. Laskaris, D. Spinellis, and L. Angelis, "Towards an affordable brain computer interface for the assessment of programmers' mental workload," *International Journal of Human-Computer Studies*, vol. 115, pp. 52–66, 2018.

[8] A. D. Bateson, H. A. Baseler, K. S. Paulson, F. Ahmed, and A. U. Asghar, "Categorisation of mobile EEG: A researcher's perspective," *BioMed research international*, vol. 2017, 2017.

[9] M. Fairhurst, C. Li, and M. D. Costa-Abreu, "Predictive biometrics: a review and analysis of predicting personal characteristics from biometric data," *IET Biometrics*, vol. 6, no. 6, pp. 369–378, 2017.

[10] J. Blasco, T. M. Chen, J. Tapiador, and P. Peris-Lopez, "A survey of wearable biometric recognition systems," *ACM Computing Surveys*, vol. 49, no. 3, pp. 43:1–43:35, Sep. 2016.

[11] S. Fakhoury, Y. Ma, V. Arnaoudova, and O. Adesope, "The effect of poor source code lexicon and readability on developers' cognitive load," in *Proc. Int'l Conf. Program Comprehension (ICPC)*, 2018.

[12] S. Lee, D. Hooshyar, H. Ji, K. Nam, and H. Lim, "Mining biometric data to predict programmer expertise and task difficulty," *Cluster Computing*, Jan 2017.

[13] B. A. Kitchenham, D. Budgen, and O. Pearl Brereton, "Using mapping studies as the basis for further research - a participant-observer case study," *Information Software Technology*, vol. 53, no. 6, pp. 638–651, Jun. 2011.

[14] K. Petersen, S. Vakkalanka, and L. Kuzniarz, "Guidelines for conducting systematic mapping studies in software engineering: An update," *Information and Software Technology*, vol. 64, pp. 1–18, 2015.

[15] B. Kitchenham and S. Charters, "Guidelines for performing systematic literature reviews in software engineering," 2007.

[16] B. A. Kitchenham, E. Mendes, and G. H. Travassos, "Cross versus within-company cost estimation studies: A systematic review," *IEEE Trans. on Software Engineering*, vol. 33, no. 5, pp. 316–329, May 2007.

[17] Y. Yuan, K.-m. Chang, J. N. Taylor, and J. Mostow, "Toward unobtrusive measurement of reading comprehension using low-cost EEG," in *Proc. of the Fourth International Conference on Learning Analytics And Knowledge*, ser. LAK '14, 2014, pp. 54–58.

[18] Y. Li, W. Zheng, Z. Cui, Y. Zong, and S. Ge, "EEG emotion recognition based on graph regularized sparse linear regression," *Neural Processing Letters*, pp. 1–17, 2018.

[19] N. Jatupaiboon, S. Pan-ngum, and P. Israsena, "Emotion classification using minimal EEG channels and frequency bands," in *The 2013 10th International Joint Conference on Computer Science and Software Engineering (JCSSE)*, May 2013, pp. 21–24.

[20] Y. Liu, O. Sourina, and M. K. Nguyen, "Real-time EEG-based human emotion recognition and visualization," in *2010 International Conference on Cyberworlds*, Oct 2010, pp. 262–269.

[21] I. Solís-Marcos and K. Kircher, "Event-related potentials as indices of mental workload while using an in-vehicle information system," *Cognition, Technology & Work*, Apr 2018.

[22] M. Züger, S. C. Müller, A. N. Meyer, and T. Fritz, "Sensing interruptibility in the office: A field study on the use of biometric and computer interaction sensors," in *Proc. of the 2018 CHI Conf. on Human Factors in Computing Systems*. ACM, 2018, p. 591.

[23] S. C. Muller and T. Fritz, "Using (bio)metrics to predict code quality online," in *Proc. of the 38th Int. Conf. on Software Engineering*, ser. ICSE '16. New York, NY, USA: ACM, 2016, pp. 452–463.

[24] S. C. Müller, "Measuring software developers' perceived difficulty with biometric sensors," in *Proceedings of the 37th International Conference on Software Engineering - Volume 2*, ser. ICSE '15. Piscataway, NJ, USA: IEEE Press, 2015, pp. 887–890.

[25] J. Jiang, Y. Zeng, L. Tong, C. Zhang, and B. Yan, "Single-trial ERP detecting for emotion recognition," in *2016 17th IEEE/ACIS International Conference on Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing (SNPD)*, May 2016, pp. 105–108.

[26] Z. İşcan and V. V. Nikulin, "Steady state visual evoked potential (SSVEP) based brain-computer interface (BCI) performance under different perturbations," *PLOS ONE*, vol. 13, no. 1, p. e0191673, 2018.

[27] C. T. Lin, J. T. King, J. W. Fan, A. Appaji, and M. Prasad, "The influence of acute stress on brain dynamics during task switching activities," *IEEE Access*, vol. 6, pp. 3249–3255, 2018.

[28] 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*, ser. ESEC/FSE 2017. ACM, 2017, pp. 140–150.

[29] A. Wahab, N. Kamaruddin, L. Palaniappan, M. Li, and R. Khosrowabadi, "EEG signals for emotion recognition," *Journal of Computational Methods in Sciences and Engineering*, vol. 10, no. 1-2 SUPPL. 1, 2010.

[30] S. C. Müller and T. Fritz, "Stuck and frustrated or in flow and happy: Sensing developers' emotions and progress," in *Proc. of the 37th Int. Conf. on Software Engineering*, ser. ICSE '15, 2015, pp. 688–699.

[31] K. Kevic, B. M. Walters, T. R. Shaffer, B. Sharif, D. C. Shepherd, and T. Fritz, "Tracing software developers' eyes and interactions for change tasks," in *Proceedings of the 2015 10th Joint Meeting on Foundations of Software Engineering*, ser. ESEC/FSE 2015, 2015, pp. 202–213.

[32] L. Ahonen, B. U. Cowley, A. Hellas, and K. Puolamäki, "Biosignals reflect pair-dynamics in collaborative work: EDA and ECG study of pair-programming in a classroom environment," *Scientific reports*, vol. 8, no. 1, p. 3138, 2018.

[33] N. Peitek, J. Siegmund, C. Parnin, S. Apel, and A. Brechmann, "Toward conjoint analysis of simultaneous eye-tracking and fMRI data for program-comprehension studies," in *Proceedings of the Workshop on Eye Movements in Programming, EMIP '18*. New York, NY, USA: ACM, 2018, pp. 1:1–1:5.

[34] M. Züger and T. Fritz, "Interruptibility of software developers and its prediction using psycho-physiological sensors," in *Proc. of the 33rd Annual ACM Conf. on Human Factors in Computing Systems*, ser. CHI '15, 2015, pp. 2981–2990.

[35] OMG, "Unified modeling language: infrastructure, version 2.4.1," document formal/2011-08-06. Technical report, OMG, Tech. Rep., 2011.

[36] R. Wieringa, N. Maiden, N. Mead, and C. Rolland, "Requirements engineering paper classification and evaluation criteria: A proposal and a discussion," *Requirements Engineering*, vol. 11, no. 1, pp. 102–107, december 2005.

[37] C. Lange, "Assessing and improving the quality of modeling," *Technische, Universiteit Eindhoven*, 2007.

[38] M. Fowler, K. Beck, J. Brant, W. Opdyke, and D. Roberts, *Refactoring: improving the design of existing code*. Addison-Wesley Professional, 1999.

[39] H. Lund, "Eye tracking in library and information science: a literature review," *Library Hi Tech*, vol. 34, no. 4, pp. 585–614, 2016.

[40] A. C. Dirican and M. Göktürk, "Psychophysiological measures of human cognitive states applied in human computer interaction," *Procedia Computer Science*, vol. 3, pp. 1361–1367, 2011.

[41] Emotiv, "Testbench™ specifications, emotiv, 2014." [Online]. Available: https://www.emotiv.com/files/Emotiv-EPOC-Product-Sheet-2014.pdf

[42] J. Katona, I. Farkas, T. Ujbanyi, P. Dukan, and A. Kovari, "Evaluation of the neurosky mindflex eeg headset brain waves data," in *2014 IEEE 12th International Symposium on Applied Machine Intelligence and Informatics (SAMI)*. IEEE, 2014, pp. 91–94.

[43] C. Wohlin, P. Runeson, M. Höst, M. C. Ohlsson, B. Regnell, and A. Wesslén, *Experimentation in software engineering*. Springer Science & Business Media, 2012.