2017 IEEE International Conference on Software Quality, Reliability and Security (Companion Volume)

Using Eye Tracking Technology to Analyze the Impact of Stylistic Inconsistency on Code Readability

Qing Mi, Jacky Keung, Jianglin Huang, Yan Xiao
Department of Computer Science, City University of Hong Kong, Kowloon, Hong Kong
Qing.Mi@my.cityu.edu.hk, Jacky.Keung@cityu.edu.hk, jianhuang7@cityu.edu.hk, yanxiao6-c@my.cityu.edu.hk

***Abstract*—A number of research efforts have focused in the area of programming style. However, to the best of our knowledge, there is little sound and solid evidence of how and to what extent can stylistic inconsistency impact the readability and maintainability of the source code. To bridge the research gap, we design an empirical experiment in which eye tracking technology is introduced to quantitatively reflect developers' cognitive efforts and mental processes when encountering the inconsistency issue.**

***Index Terms*—programming style, stylistic inconsistency, eye tracking technology, code readability, program comprehension**

I. Context and Problem Statement

Programming style can be viewed as a kind of personal preferences when writing the source code, which commonly relates to the program's visual appearance (e.g., naming of variables, how and when to use comments, indentation and alignment). Due to different experiences and personalities, developers usually have their very own programming styles, yet these individually preferred programming styles may well conflict with each other. When collaborating, it is highly likely to degrade the readability and maintainability of the source code being produced. The problem is referred to as *stylistic inconsistency* in our previous research [2]. However, to the best of our knowledge, there is little valid and reliable evidence of how and to what extent stylistic inconsistency affects the comprehensibility of the source code, which is the fundamental problem that should be resolved first in this context. Thus, we carefully plan an empirical experiment with which to bridge the research gap. The expected contribution of our work is two-fold: 1) Provide solid evidence of the impact of stylistic inconsistency on source code comprehension. 2) Present a useful and reusable research framework for analyzing the influences different stylistic factors have upon code readability.

To accurately reflect developers' cognitive efforts required to comprehend a given program, we introduce eye tracking technology to capture ocular indices (e.g., gaze position, pupil size) as objective measurements. Basically, eye movements can be characterized as a sequence of fixations and saccades.

- **Fixation**: a spatially stable gaze on an object of interest that lasts for a certain period of time. The information acquisition and processing mainly occurs during fixations, which is the most relevant index to our experiment.
- **Saccade**: a quick and simultaneous movement of both eyes from one location to another.

Eye tracking technology is an affordable and easy-to-use approach enabling us to identify what a developer is indeed viewing, for how long, and in what order, which already has broad applications in the domain of usability evaluation, cognitive interface design, and so forth. However, there are only a few studies introduce eye tracking techniques and tools into the context of software engineering research. Rodeghero et al. [3] presented a novel tool for source code summarization based on the findings of eye tracking study. Fritz et al. [1] investigated how biometric sensors (i.e., eye tracking, EDA, EEG) can be used to measure developers' perceived difficulty while working on a change task. Considering their success, we employ eye tracking technology in this study. The fine-grained measurements can establish effective relations between cognitive processes and program comprehension, which provide more credible evidence for our analysis as compared to the tradition techniques such as survey and think-aloud strategy.

II. Proposed Methodology

In this section, we present a detailed description of the controlled experiment, which is designed to empirically investigate the impact of stylistic inconsistency on code readability.

A. Independent Variables

The primary independent variable is the presence or absence of stylistic inconsistency. Note that programming style is essentially a multifaceted concept [2]. In this study, we focus solely on typographic characteristics that represent the physical layout of the source code (e.g., indentation, alignment).

B. Dependent Variables

As is shown in Table I, a group of subjective and objective measurements is proposed to represent code readability, which is the primary dependent variable in this study. In addition to the traditional measurements (e.g., NASA-TLX) that have been widely used in cognitive research, we introduce a novel variable, namely visual effort. The general consensus is that ocular indices could effectively reflect individuals' cognitive responses and mental processes when solving a given task.

C. Secondary Variables

Secondary variables are a special type of independent variables that may affect the relationship between the primary independent variables and the dependent variables. In this study, we consider the following secondary variables:

- **Development Experience**: the level of professionalism. The two possible values are expert and novice according to the number of years involved in programming activity.

978-1-5386-2072-4/17 $31.00 © 2017 IEEE
DOI 10.1109/QRS-C.2017.102

TABLE I
DEPENDENT VARIABLES ADOPTED IN THIS STUDY AND THE CORRESPONDING MEASUREMENTS

Dimension	Dependent Variable	Measurement	Description
Subjective	Self-Report Effort	NASA-TLX[a]	Participant ratings for perceived workload.
Objective	Comprehension Test	Accuracy	Correct answer rate for comprehension test.
		Answer Time	Average response time for completing the given task.
	Visual Effort	Fixation	The total number of fixations on the given program. More fixations indicate less efficiency.
		Saccade	The total number of saccades. More saccades indicate more searching effort.
		Gaze Time	The amount of time spent viewing the given program. Longer duration indicates more difficulty.
		Pupil Dilation	The percentage change in pupil diameter. Larger pupil indicates more cognitive effort.
		Blink Rate	The relative change (from baseline) in blink rate. Lower blink rate indicates higher mental load.

[a]NASA Task Load Index: a widely used rating scale for assessing subjective mental workload.

- **Style Preference**: the preference of programming style, which is used to determine if familiarity with a particular programming style has an effect on experimental results.

D. Hypothesis

We then generate null hypothesis according to our research objective in a parameterized form. H_0: There is no significant difference in X between different Y when performing comprehension tasks, where X is one of the dependent variables (i.e., self-report effort, comprehension test, visual effort), and Y is one of the independent variables (i.e., stylistic inconsistency, development experience, style preference). By replacing the parameters, we can obtain the corresponding null hypothesis.

E. Stimuli and Task Design

The stimuli shown to the participants are a series of code snippets conforming to different programming styles. As we focus solely on the influences of typographic variations, all other respects are deliberately made equal. To avoid the carryover effects, we carefully plan a between-subjects design. All participants are evenly divided into two groups. In the *Control Group*, participants come to a set of code snippets with a consistent programming style, whereas in the *Treatment Group*, the code snippets alternate with different programming styles to simulate a scenario of stylistic inconsistency. The primary task for participants is to answer the questions accompanied by each code snippet. To alleviate the learning burden, each task is designed to take only a short time to complete.

F. Experimental Procedure

Referring to previous studies (including but not limited to [1], [3]), the procedure of our experiment is designed as follows: 1) Participants are expected to sign an informed consent form. 2) Calibrate eye tracker (i.e., the device used to track eye movement in an unobtrusive way) for each participant due to varying eye characteristics. 3) The experimental procedure and main tasks are introduced to participants, but not the true objective. Also, participants are told that they are free to withdraw from the experiment at any time they want. 4) A pilot experiment is performed to familiarize participants with the equipment and the corresponding workflow. The data obtained in this stage will be discarded without any analysis. 5) In order to guarantee a consistent performance, participants are required to rest 1-2 minutes at intervals for relaxation. 6) For each task, participants are instructed to view a certain code snippet, and then complete a post-hoc comprehension test that consists of 1-2 objective questions. Both the code snippet and the questions are displayed on one screen. Eye movement data is implicitly captured as the participants solving the given task. 7) After each task, participants are required to report their comprehension effort using NASA-TLX. 8) A post-experiment questionnaire is conducted to collect the participants' demographics (e.g., age, gender, ethnicity, education, development experience, style preference). During the entire experimental procedure, stage 5-7 may appear several times sequentially.

III. CONCLUSIONS

As no conclusive evidence in literature has shown that how and to what extent stylistic inconsistency degrades code readability and maintainability, we carefully plan a controlled experiment to bridge this research gap, in which eye tracking technology is introduced to capture participants' cognitive responses and mental processes. We believe that our results will be beneficial for both researchers and practitioners to better understand the stylistic inconsistency issue, and take further actions to improve code readability as the ultimate goal.

ACKNOWLEDGMENTS

This work is supported in part by the General Research Fund of the Research Grants Council of Hong Kong (No. 125113, 11200015 and 11214116), and the research funds of City University of Hong Kong (No. 7004683 and 7004474).

REFERENCES

[1] 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 - ICSE 2014*, pages 402–413, New York, New York, USA, 2014. ACM Press.

[2] Q. Mi, J. Keung, and Y. Yu. Measuring the stylistic inconsistency in software projects using hierarchical agglomerative clustering. In *Proceedings of the The 12th International Conference on Predictive Models and Data Analytics in Software Engineering*, PROMISE 2016, pages 5:1–5:10, New York, NY, USA, 2016. ACM.

[3] P. Rodeghero, C. Liu, P. W. McBurney, and C. McMillan. An Eye-Tracking Study of Java Programmers and Application to Source Code Summarization. *IEEE Transactions on Software Engineering*, 41(11):1038–1054, nov 2015.