Software Readability Practices and the Importance of their Teaching

Isabel Braga Sampaio
Institute of Engineering of Porto
Polytechnic Institute of Porto
4200-072 Porto, Portugal
Email: ais@isep.ipp.pt

Luís Barbosa
School of Science and Technology
University of Trás-os-Montes and Alto Douro
Apartado 1013, 5001-801 Vila Real, Portugal
Email: lfb@utad.pt

***Abstract*—It is known that code readability has an impact on software quality. This paper introduces a preliminary list of 33 good practices for code readability that could be taught in object oriented programming courses of informatics programs. In this set are included practices with both positive and negative impact, the latter mainly from bad smells. A survey was made with OOP teachers in order to assess the significance of teaching a set of good practices. The results showed that teachers consider it important to teach the defined set. Such result was statistically significant. Still, the same level of significance was not awarded to all practices. Some additional results are presented.**

I. Introduction

Software readability is deemed as an important feature of software quality, due to its potential impact on software maintenance and evolution (e.g. [1]). Moreover, is the fact of software being commonly developed by teams whose members need to read code from each other, namely in code reviews. Reading other peoples code is a permanent need in open source projects, but it is no less true that a lot of these projects pay attention to code readability. It is during the higher education informatics programs that the writing of readable code should be initiated and reinforced, so that the future professionals are able to do so from right the very beginning of their careers. In other words, it could be said that the responsibility of teaching and helping the students develop the aforementioned skill lies with the higher education programs. Such skill is also relevant in students works, namely when they take part in group projects and whenever the teachers need to analyze the students work.

Nowadays, object orientation is the most commonly used programming paradigm. The main goal of this work is to develop a set of good practices for software readability to be used in object-oriented programming (OOP) to be used in teaching students of object-oriented programming (OOP) from informatics higher education programs. In order to achieve that goal, this work started with a literature review, which was intended to determine such set of good practices. From our point of view, such code readability improvement practices should be taught in informatics programs. Nonetheless, whether that is the general view among OOP teachers must be questioned. As such, the goal at a first stage was to ascertain the importance given to the teaching of good practices by the OOP teachers, using the practices already gathered during the literature review. A survey using that set of good practices was made among OOP teachers to achieve such goal. This set should be considered preliminary.

The initial literature review and the survey will be the main focus points of this paper. The literature review is presented in section 2, right after this introduction. The survey and its results are described in section 3. Section 4 presents the main conclusions and future research.

II. Related Work and Literature Review

This section describes an initial literature review. The review was performed with purpose of finding what could be called good practices for source code legibility. It followed a snowball approach, with some references leading to others. The review allowed us to find a set of good practices for source code legibility. Concerning the significance of their teaching, this is, to the best of our knowledge, the first study so far.

It is important to gather not only the good practices, but also those that might have a negative impact and should be avoided. In particular, we selected a list of bad smells [2] whose descriptions seem to make it clear that they have impact on readability. This set is mainly composed of bad smells. The list of compiled practices and their respective fonts are listed in Table I. Those practices cover different aspects of software, such as code structure (e.g. blank lines), code organization (e.g. identifiers scope), and code logic and its complexity (e.g. excess of decision ramifications). In the table, the numbers of the “negative” practices are suffixed with a hyphen. This is the final list used in the study.

As can be noted from the table, both groups views are not disjointed. An effort was made to reduce redundancy for the development of the list of good practices, however some could not be completely freed from that problem. Such is the case of practices 8, 20, and 24, which can be seen as different ways to achieve a limited scope, therefore it was decided to keep all three of them. The same occurs with practices 16 and -5, specifically about identifier names.

III. The Survey

This empirical study was conducted through a survey. The survey is considered an adequate research method when the researchers want to gather data from a large number of

978-1-4673-8614-2/16/$31.00 ©2016 IEEE

TABLE I
LIST OF COMPILED GOOD PRACTICES FROM THE LITERATURE REVIEW.

N.	Description	Source
1	maximum line length	[3]–[6]
2	lines indentation	[5]–[7]
3	Use of brackets in expressions	[6]
4	Uniformity in the notation, terminology and symbols used	[3]
5	Breaking the line after semicolon	[8]
6	blank lines	[5], [6], [8]
7	Related code lines should appear together	[9]
8	Short distance between definition and reference of a variable	[9]
9	The function that calls should be above the function that is called	[9]
10	The assignment operator include blank space before and after	[9]
11	Blank space between the arguments / parameters of functions	[9]
12	Blank space to emphasize the precedence of operators in an expression	[9]
13	Use of symbolic constants instead of numeric constants	[9]
14	Use of enumerations instead variables	[10]
15	Use { and } to enclose the statements in a loop	[10]
16	Naming identifiers appropriate	[5], [6], [11], [12]
17	Maximum identifier length (prox. 9 a 16 characters).	[3], [5], [6], [10], [13]
18	Comments: consistent with the code and controlled	[5], [6], [14]
19	Avoid over decisions branches	[5], [6]
20	Reorganization of instructions in a module so as to reduce the scope of variable	[15]
21	Avoid recursive functions	[5], [6]
22	Use overrriding when suitable	[5], [6]
23	Prefer containers to vectors	[5]
24	Narrow scope of the variables	[5], [15]
25	Variable consistency according to all its uses	[5]
1-	Long method	[2]
2-	Long parameter list	[2]
3-	Identical or very similar code existing in more than one locationl	[2]
4-	Identifiers names in method names	[2]
5-	Method names that do not describe methods making	[2]
6-	Incorrect definition / use of inheritance	[5]
7-	Long conditional logic blocks	[2]
8-	Local declaration without criterion	[2]

respondents [16]. In our case, the intention was to make it available to the largest possible number of respondents in a relatively short time. For these reasons it was also decided to use an online survey.

A. Research Goal and Questions

The main goal of this research was to gauge about the importance of teaching a set of source code readability “good practices” by the academic community of OOP teachers. The focus is on OO programming, but several practices are transversal to languages of other programming paradigms. As an initial study, it was decided to limit it to the universe of Portuguese OOP teachers.

From the research goal several research questions were defined:

*RQ*1 What level of importance do OOP teachers give to the teaching of good practices for software readability? *RQ*2 Do teachers account for readability when evaluating students works? *RQ*3 Do students already produce high readability code? In other words, what is the average level of readability of the code produced by students? *RQ*4 Does the teachers' experience affect the importance they give to readability?

RQ1 is the main research question. Although it is not possible to depict all variables that could intervene in answering the survey, experience in teaching OOP could have some importance. So a question, RQ4, was added in order to help depict that variable. As for RQ3, if students produce high readability code then this research project should be rethought. The aim of RQ2 is to know the current status of readability assessment.

B. Survey Design

The survey purpose can be classified as mainly descriptive in the sense that the main goal is to obtain information about the significance of a set of practices. But it is also exploratory, as almost nothing was already known about how teachers judge those practices. Asking the participants about the importance they give to the practices according to their past experience, makes the study design, following [17], cross-sectional (at a moment) and case control (past experience).

C. The Instrument

A questionnaire was developed for the survey to serve as the instrument to gather the required data. It was developed according to the study goal, based on the practices, and oriented to OOP teachers. The questionnaire was constructed after a literature review on software readability. Its construction followed the main recommendations found in the survey literature (e.g. [16], [18]).

The instrument consists of a total of 40 questions grouped according to their goals, as explained next, and it was written in Portuguese. The main group is composed of 33 non-optional questions about potential software readability factors acquired from the literature, as explained in section II. Each question corresponds to a practice from Table I so these can be evaluated. The table was indeed the result of the instrument evaluation described bellow. Following the practices, the questions were also divided into two sets of factors; the ones with positive impact on readability and the ones with negative impact, with 25 and 8 questions respectively.

All 33 questions started with “Level of importance you give to its teaching?” followed by a statement of the respective practice. This question was formulated to reflect the main goal of the questionnaire, as expressed by RQ1, as much as possible. For each readability question, the questionnaire uses an ordinal scale with five points. A five-point scale was used because it is considered to increase the reliability of the estimates [19]. The scale is of the Likert-type, with values ranging from 1 (the lowest value) to 5 (the highest value). The meaning of the values is: 1 - “Very low ”; 2 - “Low”; 3 - “Average”; 4 - “High”; 5 - “Very high”. It was intentional to avoid the “dont know” point in the classification scale, because the target participants, being OOP teachers, should have an opinion about each practice.

There is an additional question asking the respondents for any other practice they use. After that, the respondent can find a text-box for comments and suggestions. These two items are optional. In order to make it possible to answer the remaining research questions RQ2, RQ3 and RQ4, the following three non-optional questions were added to the beginning of the questionnaire: When evaluating students works do you take into consideration the code readability; In average how do you evaluate the readability of students source code; Number of OOP editions you teach (not necessarily of the same course). The first question uses a Boolean scale of true/false, and the second uses the usual 5 points scale. The third is numeric and accepts values from 1 to 10 or more. The learning of OOP in informatics programs is expected to happen at the undergraduate level. To verify that, a question about the level of the program (undergraduate or master) was added to the questionnaire.

In this case, there was only one demographic question. It was an optional question, asking the respondents to supply their e-mail addresses if they wish to receive survey results. To assure the respondents anonymity, the questionnaire does not require any identification data about them, except for the optional field for the e-mail addresses. A brief introduction to the questionnaire, including its goal, was inserted at the beginning of the questionnaire. The time to complete it was 7 minutes.

D. Instrument Evaluation

The instrument was developed and initially reviewed by the first author for the format of the questions, understandability, number of questions, and time required to complete it. Next, the questionnaire was sent to two informatics teachers for review. One was the second author of this paper and the other was a colleague of the first authors. With the information from the two reviewers, the instrument was improved, mainly regarding wording and redundancy. As a consequence, some questions were removed and others were reworded.

As a last evaluation step, the first authors colleague answered the reviewed questionnaire. As a result, one more question was removed, some others were improved, and the time to complete was analyzed. The conclusion was that 7 minutes should be enough.

E. Participants

According to the study goal, the target population is composed of all OOP teachers from Portuguese higher education informatics programs. However, to the best of our knowledge, such a list does not exist. Still, the ministry of education possesses in its site the list of all programs. Therefore, it was decided to select and access directly the informatics programs and ask the responsible for each program to make the questionnaire accessible to their OOP teachers. The result was a list of 44 informatics programs.

F. Hypothesis

Some hypotheses from the research questions were developed for statistical analysis.

RQ1: The level of importance teachers give to the teaching of a set of good practices for source code readability is the construct. This construct is operationalized through a set of questions about the practices. In order to respect the scale type, the analysis of the results can be done in terms of medians and percentages and statistical tests based on ranks [20], [21], that is, non-parametric tests.

For RQ1, the goal is ultimately to know if OOP teachers consider the teaching of this set of practices significant. In the 5-points classification scale of importance, the point 3 is the average value. So, to perceive the set of practices as important, the median of the answers should be above 3. From here the hypotheses are:

$H1_0$: The set of good practices is not important for the OOP teachers. That is, the median is 3 or less, $\theta <= 3$

$H1_a$: The set of good practices is considered important for the OOP teachers. That is, the median is greater than 3, $\theta > 3$

Because there are two subsets, identical hypotheses apply to each subset. For the tests, the median will be used with the Wilcoxon signed-rank test for a single sample for a significance level of 5% for one-tail. The result will be compared with the critical value.

RQ2 and RQ3: These two questions were considered as mainly descriptive and no hypothesis were formulated for their test.

RQ4: For RQ4, the experience of the teachers in teaching OOP can be measured by the number of editions a teacher has taught OOP. Hypothesis for RQ4 are:

$H4_0$: The number of editions is independent of the level of importance attributed to the teaching of the set of practices. That is, the variables are mutually independent, $\rho = 0$

TABLE II
DESCRIPTIVE STATISTICS.

Q#	Median	Range	IqR
1	3	4	1
2	5	3	1
3	3	4	1
4	5	2	1
5	4	3	1.5
6	4	3	1
7	4	4	0.5
8	3	4	2
9	3	4	1.5
10	3	4	2
11	2	4	2
12	2	3	2
13	4	4	1.5
14	3	4	1
15	4	4	1
16	5	4	1
17	4	4	1
18	4	4	2
19	4	4	0
20	4	3	1
21	4	4	1
22	4	4	2
23	3	4	1
24	4	4	1.5
25	4	2	1

Q#	Median	Range	IqR
1-	4	3	2
2-	4	4	1
3-	4	2	1
4-	4	4	2
5-	5	2	1
6-	5	3	1
7-	4	4	1
8-	4	4	2

$H4_a$: There is a correlation between the number of editions and the level of importance attributed to the teaching of the set of practices. That is, $\rho <> 0$

The non-parametric Spearman correlation coefficient will be used with a p-value for an alpha of 5% for two-tails.

G. Data Collection

After the evaluation, the instrument was published via the same Web site used for its development. A short text about the survey was prepared. The link and the text were sent by e-mail to the program directors or department heads, asking them to make the link available to their OOP teachers. The questionnaire was made available for approximately a month. There was no follow-up step.

H. Results

The number of teachers that answered the questionnaire was thirty-three (33), and eleven (11) among them supplied their e-mail addresses. All 33 filled questionnaires were about OOP teaching at undergraduate level.

RQ1: Table II presents the median, range and inter-quartile range (IqR) for each question. In the table, the practices to avoid are still numbered with a hyphen at the end. The median for all the classifications was 4. Table III contains the median for each teacher. The Wilcoxon signed-rank test provided the result $T = 24.5$ for a $median = 3$ knowing that 7 rank differences were equal to 0. Given that the critical value of T for $n = 26$ and one-tail is 110 the null hypothesis was rejected.

The two subsets of teachers medians for the positive and negative questions, respectively, are listed at Table IV. The same statistical test in the same conditions was applied. For the first set of medians (positive) the test result was $T = 21.5$ for a $C_c = 75$, and for the second set it was $T = 16.5$ for a $t_c = 151$. The result was significant for both.

RQ2 and RQ3: Thirty-two (32) teachers account for readability when evaluating students works and one (1) does not. The median for the question about the level of the readability of students code was 3.

RQ4: The totals for experience with OOP teaching, counted in number of editions, are showed in table V. The column entry 10+ means 10 or more editions. For better accuracy not only the correlation between the number of editions taught and the teachers total medians were computed, but also between the number of editions with the teachers medians for positive factors and with the teachers medians for negative factors using the values from Table IV.

All correlations were calculated using the Spearman correlation coefficient. The p-value for an alpha of 5% and two-tails was also determined. Table VI shows the results for the 3 correlations. From the values the null hypothesis could not be rejected for none of them.

I. Discussion

As stated, the questionnaires were not sent directly to the potential respondents and, consequently, there is no way to know how many teachers received the questionnaire, nor the population size. As such, there is no concrete value for the response rate.

Starting with research question **RQ1**, there was a value of 4 for the total median of answers to the readability practices questions. The medians of the classifications attributed by each teacher to all questions (Table III) were high. For the three null hypotheses, the obtained values for T for alpha 5% and one-tail allow to reject the null hypothesis in the 3 cases. Also, for the 3 tests the $\Sigma(positive\_ranks) > \Sigma(negative\_ranks)$. Concluding, teachers consider the global set of practices and each of the subsets important.

Only two practices (11 and 12) about separation with blanks had a median 2 - mode was 2 and 3, respectively. This means that teachers in general attribute low significance to their teaching. Other 7 practices scored average. One, the 11 was also about blanks. According to the range in Table II, for many questions there is a large variability among the teachers answers. In other words, although teachers agree that practices are important, the importance they give to each practice varies.

No practice to be avoided (Table II right) attained a median lower than 4, but for the positive, medians 2 and 3 were observed. However, the same statistical test in the same conditions was applied to the two subsets of medians of answers by each teacher and the results were significant. This shows that, despite some differences, each kind of practices, taken individually, is considered important.

Regarding **RQ2**, thirty-two out of thirty-three teachers answered affirmatively. This means that teachers account for readability when evaluating the students works, and surely they should give importance to readability, otherwise they wouldnt

TABLE III
RESPONDENTS MEDIANS.

Teacher	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	
Median	5	3	3	4	4	4	4	4	4	3	4	4	4	4	4	5	
Teacher	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33
Median	4	4	3	4	4	4	3	4	3	1	4	4	4	5	4	5	3

TABLE IV
TEACHERS MEDIANS FOR POSITIVE AND NEGATIVE PRACTICES. 33 TEACHERS.

Positive median	5	4	3	3	4	4	4	3	4	4	4	3	4	4	4	4	
cont.	4	4	3	4	4	3	3	4	4	4	3	4	3	1	4	3	3
Negative median	4	5	3	3	5	4.5	4.5	4	5	5	4	4	4	3.5	4	5	
cont.	4	5	4.5	4	5	4	4	5	5	5	3	5	3.5	1.5	3.5	5	4

TABLE V
NUMBER OF EDITIONS.

N. of Editions	1	2	3	4	5	6	7	8	9	10+
Responses	5	5	2	2	4	0	3	1	0	11
Percent(%)	15	15	6	6	12	0	9	3	0	33

TABLE VI
CORRELATIONS BETWEEN EXPERIENCE AND IMPORTANCE MEDIANS.

	global medians	medians of pos.	medians of neg.
ρ	-0,085817811	0,026635753	-0,064897985
p-value	0,634890513	0,883024042	0,719734081

evaluate it. This is important, because although the results for RQ1 refer to the specific set of practices in study, the importance of readability becomes more generalizable with this result.

About **RQ3**, the median was 3. This value means that in general students code readability is considered average, not good, but not bad. In this question there was only one teacher who classified with 5, but 3 classified with 1. It is reasonable to say that there is room for the improvement of students code readability.

For **RQ4**, the 3 correlations between the experience, measured by the number of editions taught, and the global teachers medians, partial teachers medians for positive factors, and partial teachers medians for negative factors, were all very low and without statistical significance (the 3 p-values were above the alpha of 0.05). So, the null hypothesis could not be rejected and can be concluded that teachers experience appears to have no effect on the way they perceive the importance of teaching readability good practices and, even more, independently of being practices with positive or negative impact.

Although the correlations to answer RQ4 for the 2 subsets of practices with positive and negative on readability were approximately 0, one was positive and the other was negative. Also, looking to Table II for the medians of the questions of the subsets can be seen that no median is less than 4. What is more, from RQ1 there is an apparent difference between the importance teachers give to both subsets. So the question arises to know if these subsets of - that were shown to be important when considering each subset individually and together - are seen equally important by the teachers or not. In other words, it is necessary to compare the medians of teachers ratings to each group of questions.

For this situation the same teachers are answering two sets of questions, meaning that the analysis should be for a paired sample. Using Wilcox for paired samples gives a significant result of $T = 22$, for 2-tail and alpha=5%, which is less than the critical $t_c = 89$. The null hypothesis can be rejected and the importance of positive and negative practices could not be considered as identical (coming form the same distribution) by the respondents. In other words, the negative are considered more important. However, care is needed due the large difference between the 2 sets size.

More than one reason can help explain the difference. Firstly, bad smells are well known and studied, and the software community in general, and teachers in particular, recognize their importance. Secondly, this subset of bad smells resulted from a selection made during the review that allowed to obtain a more "reliable" bad smells subset regarding readability. Also, we can speculate that the bad smells practices are more directly related with code errors than the positive practices, which are indirectly related with code errors and maintenance via readability. Possibly this was also the case before with the practices related with the use of blanks. To finish, can be tough that more effort is needed to make the positive readability factors more well known.

To conclude, readability and the teaching of good practices for software readability were considered important by the respondents in general. We believe that more research is needed, and the development of a set of good practices is of interest. Surely, only a portion of OOP teachers was contacted and answered the survey, but according to the authors experience as teachers, the obtainment of similar results can be expected for the larger population of teachers.

J. Limitations

Response bias means that survey responses do not represent the sample responses, or the population, and it is usually associated with low response rates. This problem can limit external validity [22]. It is known that surveys usually have a low response rate [16]. Nonetheless, as explained before, the teachers response rate has no known value. Even if we did not have a large response rate but are sure that the respondents are representative of the larger population, the analysis can be done [17]. In the present case there is no reason to suspect that non-respondents would answer differently.

Validity in general and content validity, in particular, were in some extension strengthened by the evaluation done before the survey. Construct validity cannot be assured, but all the practices being collected from the literature gives some degree of validity to the questions. Additionally, the question formulated for all the practices was the same, except for a minor note concerning the direction of the impact. However, this survey is also exploratory. A tendency of a Likert scale, which is also true for a Likert-type scale, is the central tendency bias. This tendency was not measured, but the median for the majority of the questions was above the central point, showing no significant problems with central tendency bias.

To sum up, the major limitation for the validity/reliability of the survey was the sampling strategy used, which was not probabilistic but, nevertheless, tried to cover as many teachers as possible indirectly through the programs. As such, randomness was present in the respondents sampling. Additionally, the survey only consulted OOP teachers from one country. Finally, variations about the position of the OOP courses in each program curriculum and the existence of previous programming courses have not been ascertained.

IV. Conclusions

The main contributions of this work are a set of practices for source code readability and a survey to gauge the importance of its teaching within OOP courses of informatics higher education programs. An initial literature review was done about readability practices, resulting in an initial set of 33 practices. Next, a survey with the set of good practices to be filled by OOP teachers was performed.

For the main research question about the importance of teaching the practices the answer was positive, with a global median of 4 (in 1 to 5) and statistically significant. Furthermore, the importance of teaching each subset of positive and negative practices when considered individually was also statistically significant. In conclusion teachers recognize the importance of teaching readability practices, this in particular, and, indirectly, if teachers consider them important, then that gives a more general value to the practices. Also they account for readability when evaluating students works. However, there are 2 practices with negative medians and the range of scores for each practice is somewhat large. This issue leads to the belief that the set can be improved. The second subset, mainly composed of bad smells, suggests that it could be easier for teachers to perceive the importance of practices that are more directly linked to code errors as well as more well known.

The results are an encouragement to further research of good practices for software readability aiming to produce a set of practices to be taught in informatics OOP courses. So, plans for future research are to improve the set of good practices used in this study, which is already being done, by deepening and expanding the literature review about readability practices.

V. Acknowledgments

The authors would like to thank Dr Barbara Kitchenham for her help with the statistical analysis and priceless observations, and to Prof. Alberto Sampaio for his opinions.

References

[1] B. Boehm and V. R. Basili, "Software defect reduction top 10 list," *Foundations of empirical software engineering: the legacy of Victor R. Basili*, vol. 426, 2005.

[2] M. Fowler, *Refactoring: Improving the Design of Existing Code*. Boston, MA, USA: Addison-Wesley, 2000.

[3] B. Shneiderman, *Software psychology: Human factors in computer and information systems (Winthrop computer systems series)*. Winthrop Publishers, 1980.

[4] B. A. Benander, N. Gorla, and A. C. Benander, "An empirical study of the use of the goto statement," *Journal of Systems and Software*, vol. 11, no. 3, pp. 217–223, 1990.

[5] Y. Tashtoush, Z. Odat, I. Alsmadi, and M. Yatim, "Impact of programming features on code readability.," *International Journal of Software Engineering & Its Applications*, vol. 7, no. 6, 2013.

[6] R. P. Buse and W. R. Weimer, "A metric for software readability," in *Proceedings of the 2008 international symposium on Software testing and analysis*, pp. 121–130, ACM, 2008.

[7] R. J. Miara, J. A. Musselman, J. A. Navarro, and B. Shneiderman, "Program indentation and comprehensibility," *Commun. ACM*, vol. 26, pp. 861–867, Nov. 1983.

[8] R. Namani and J. Kumar, "A new metric for code readability," *IOSR Journal of Computer Engineering (IOSRJCE)*, vol. 6, pp. 44–48, Nov. - Dec. 2012.

[9] R. C. Martin, *Clean Code: A handbook of agile software craftsmanship*. Prentice Hall, 2009.

[10] S. McConnell, *Code Complete*. Redmond, WA, USA: Microsoft Press, second edition ed., 2004.

[11] B. Caprile and P. Tonella, "Restructuring program identifier names," 2000.

[12] P. Tonella and S. L. Abebe, "Code quality from the programmer's perspective," *Proceedings of Science, XII Advanced Computing and Analysis Techniques in Physics Research, Erice, Italy*, vol. 5, p. 153, 2008.

[13] G. Scanniello and M. Risi, "Dealing with faults in source code: Abbreviated vs. full-word identifier names," in *Proceedings of the 29th IEEE International Conference on Software Maintenance*, ICSM, (Eindhoven, The Netherlands), pp. 190–199, IEEE, 2013.

[14] L. Tan, D. Yuan, G. Krishna, and Y. Zhou, "/*icomment: Bugs or bad comments?*/," *ACM SIGOPS Operating Systems Review*, vol. 41, no. 6, pp. 145–158, 2007.

[15] Y. Sasaki, Y. Higo, and S. Kusumoto, "Reordering program statements for improving readability," in *Proceedings of the 17th European Conference on Software Maintenance and Reengineering*, CSMR, pp. 361–364, IEEE, 2013.

[16] S. L. Pfleeger and B. A. Kitchenham, "Principles of survey research: Part 1: Turning lemons into lemonade," *SIGSOFT Softw. Eng. Notes*, vol. 26, pp. 16–18, Nov. 2001.

[17] B. A. Kitchenham and S. L. Pfleeger, "Principles of survey research part 2: Designing a survey," *SIGSOFT Softw. Eng. Notes*, vol. 27, pp. 18–20, Jan. 2002.

[18] E. Babbie, *The practice of social research*. Thomson Wadsworth, 11th ed., 2007.

[19] H.-W. Jung, S.-G. Kim, and C.-S. Chung, "Measuring software product quality: A survey of iso/iec 9126," *IEEE software*, no. 5, pp. 88–92, 2004.

[20] B. Kitchenham and S. L. Pfleeger, "Principles of survey research part 6: Data analysis," *SIGSOFT Softw. Eng. Notes*, vol. 28, pp. 24–27, Mar. 2003.

[21] D. J. Sheskin, *Handbook of parametric and nonparametric statistical procedures*. CRC Press, 3rd ed., 2004.

[22] P. J. Lavrakas, ed., *Encyclopedia of survey research methods*. Sage Publications, 2008.