Copy and Paste Behavior: A Systematic Mapping Study

Luqi Guan
Dep. Ing. Informática, Escuela Politécnica Superior
Universidad Autónoma de Madrid
Madrid, Spain
luqi.guan@estudiante.uam.es

*John W. Castro**
Dep. Ing. Informática y Ciencias de la Computación
Universidad de Atacama
Copiapó, Chile
john.castro@uda.cl

Xavier Ferre
Dep. Lenguajes y Sistemas Informáticos e Ingeniería de Software, E.T.S. de Ingenieros Informáticos
Universidad Politécnica de Madrid
Boadilla del Monte, Spain
xavier.ferre@upm.es

Silvia T. Acuña
Dep. Ing. Informática, Escuela Politécnica Superior
Universidad Autónoma de Madrid
Madrid, Spain
silvia.acunna@uam.es

***Abstract*—Both novice and experienced developers rely more and more in external sources of code to include into their programs by copying and pasting code snippets. This behavior differs from the traditional software design approach where cohesion was achieved via a conscious design effort. Due to this fact, it is essential to know how this copy and paste programming practices are actually carried out, so that IDEs and code recommenders can be designed to fit with developer expectations and habits. Our objective is to identify the role of copy and paste programming or code clone in current development practices. A Systematic Mapping Study (SMS) has been conducted, searching the main scientific databases. The search retrieved 1271 citations and 39 articles were retained as primary studies. The primary studies were categorized according to eight areas: General information of clone usage, developer behavior, techniques and tools for clone detection, techniques and tools for clone reuse, patterns of cloning, clone evolution, effects of code cloning in software maintenance and development, and tools for clone visualization. The areas, techniques and tools of clone detection and developer behavior are strongly represented in the sample. The areas that have been least studied in the literature found in the SMS are tools of clone visualization and patterns of cloning.**

Keywords— Copy and Paste; Systematic Mapping Study

I. INTRODUCTION

The huge amount of source code available online has changed coding practices. Both novice and experienced developers rely more and more in external sources of code to include into their programs by copying and pasting code snippets [1][2], which is basically a term used in system engineering. To copy the code and reuse the code, either by doing some modifications or without doing any modification in the existing code, are common activities in software development [3]. Copy and paste is often done by inexperienced or student programmers, who find the act of writing code from scratch difficult or irritating and prefer to search for a pre-written solution or partial solution they can use as a basis for their own problem solving [1]. Copy and paste is also done by experienced programmers, who often have their own libraries of well tested, ready-to-use code snippets and generic algorithms that are easily adapted to specific tasks [2]. This behavior differs from the traditional software design approach, where cohesion was achieved via a conscious design effort [4]. It also differs from the code reuse attained through the usage of re-use repositories built for such specific purpose. We need to know how this copy and paste programming practices are actually carried out, so that IDEs and code recommenders can be designed to fit with developer expectations and habits. The research work aims to identify the role of copy and paste programming or code clone in current development practices, by identifying through a Systematic Mapping Study [12] the current knowledge about this topic in the existing literature.

Paper organization. In Sec. 2, we present related work. In Sec. 3, we describe the research method of the SMS. Sec. 4 presents the results of the SMS. In Sec. 5, we discuss the results and threats to validity, and finally Sec. 6 concludes.

II. RELATED WORKS

We found six systematic reviews related to copy and paste [5]-[10]. The literature review by [5] presents various methods that researchers have used to study clone evolution and summarizes the advantages and disadvantages of relevant research on clone evolution. The literature review by [6] has studied code cloning and various techniques to detect code clones. The SMS by [7] focuses on metric-based clone detection techniques and various tools used in previous studies. The literature review by [8] puts a light on all the types of clones and various techniques for the detection of clones. The systematic review by [9] analyzes how code clones can be detected and which techniques and tools are used for this purpose. The literature review by [10] presented comparative review of various clone detection techniques. Most of these literature reviews are related to code clone detection and code clone

* Corresponding Author.

DOI reference number: 10.18293/SEKE2020-130

evolution, they do not refer to developer behavior, techniques and tools of clone reuse, patterns of cloning, tools for clone visualization and effects of code cloning in software maintenance and development. After analyzing papers that refer to those areas mentioned above, we can confirm that there is no SMS on these areas of code cloning, Therefore, we identify a lack of systematic approaches to identify the state of the art in these areas of code cloning.

III. Research Method

We aim to answer the following research questions: (**RQ1**) What is the state of the art of copy and paste? and (**RQ2**) How do developers use copy and paste? To answer both questions, we have carried out an SMS.

A. Define the Search Strategy

For the definition of the search string, we need to perform the following steps: Conformation of the control group (CG), identification and selection of the keywords, conformation of the search strings, and specification of the inclusion and exclusion criteria. To form the CG, we conducted a traditional search to identify papers directly related to our research. As a result of this search, we found a total of 10 papers: [3][13]-[21]. In the papers of the CG the words that appear most frequently must be identified. The keywords were obtained from a table with the frequency of all the words that appear in the articles of the CG. Once the keywords were identified, several options were built for the search string. Finally, we opted for the following search string: ("copy and paste code" OR "source code reuse" OR "code reuse" OR "code snippets reuse" OR "code clone" OR "code cloning" OR "software clones") AND (analysis OR design OR approach OR behavior OR habits OR intent OR research OR patterns OR "usage patterns" OR method OR techniques OR tools) AND ("software system" OR development OR developer OR system OR programming). The criteria used to retrieve the fundamental studies are summarized below. These criteria were applied by 3 of the authors of the paper.

a) Inclusion criteria: The paper is related to copy and paste behavior; OR the paper discusses aspects related to copy and paste patterns; OR the paper is related to code clones; OR the paper is about finding duplicated code.

b) Exclusion criteria: The paper is about traditional code reuse; OR the paper discusses about creating repository for future reuse; OR the paper is about programing for reuse; OR the paper is about managing duplicated code; OR the paper is a review; OR the paper is written in a language other than English.

B. Select the Studies

The search for studies was carried out in the following digital databases: Scopus, ACM Digital Library, and IEEE Xplorer. Once the list of *Retrieved Papers* is obtained (1271), it is necessary to eliminate duplicates between the databases and as a result of this first debug the *Non-Duplicate Candidate Papers* are obtained. Then, a first filter must be made applying the inclusion and exclusion criteria on the title, summary and keywords of each of the *Candidate Papers* (163). Studies obtained from the first filter were evaluated again in a second filter. In this second filter, each researcher applied the inclusion and exclusion criteria to the full text of each of the studies. As a result, the group of *Primary Studies* was obtained (39). The search was conducted in November 2019.

C. Extract the Data and Perform Data Synthesis

Once the *primary studies* are obtained, the relevant information is extracted to answer the research questions. Figure 1 provides an overview of the primary studies retrieved by the SMS. It is made of three categories, determined by the year of publication, type of paper and research areas.

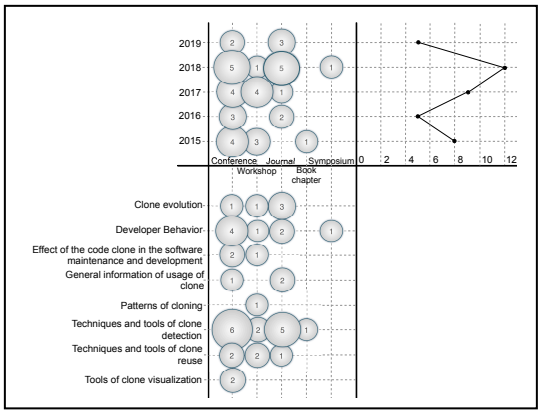

Figure 1. *Mapping showing the primary study distribution*

The left-hand side is composed of two scatter (XY) charts with bubbles at the intersections of each category. The size of each bubble is determined by the number of primary studies that have been classified as belonging to the respective category at the bubble coordinates. The right-hand side of the figure shows the number of primary studies by publication year. We can observe that publications started to grow from 2016 and many papers have been published since then, confirming the raising interest in this research area.

IV. Results

After analyzing the primary studies (see Figure 1) and papers belonging to the CG, we identified eight different research areas: General information of clone usage, developer behavior, techniques and tools for clone detection, techniques and tools for clone reuse, patterns of cloning, clone evolution, effects of code cloning in software maintenance and development, and tools for clone visualization. Next, we will describe each of these areas.

General Information of Clone Usage. This area deals with clone types and high-level uses of clone information, as well clone usage patterns [3][15][18]-[23].

Developer Behavior. This area is about how developers face the use of clones (how they search, how they embed them in their code, etc.) [13]-[16][19][20][24]-[30].

Techniques and Tools for Clone Detection. This area studies the techniques and tools for clone detection, analysis and management and the use of clone-aware tools [3][11][14][31]-[43].

Techniques and Tools for Clone Reuse. This area studies the techniques and tools for clone reuse. Such as the interactive approach for recommending where and how to modify the pasted code, the approach to merge similar pieces of code by creating suitable abstractions, etc. [44]-[48].

Patterns of Cloning. This area describes several patterns of cloning, such as forking, templating and customization; the pros and cons of cloning; and methods for managing code clones [17][49].

Clone Evolution. In this area the clone community focuses on how cloned code evolves over time [15][24][50]-[54]. As this code changes, it exhibits various patterns and characteristics.

Effects of Code Cloning in Software Maintenance and Development. This area studies the effects of code cloning. It deals with the maintenance problems that clone codes can cause, as well as the clone display tools and clone patterns and refactoring recommendations to solve such problems [18][55]-[57].

Tools for Clone Visualization. This area studies tools for code clone visualization. These code clone visualization tools are used for checking code and analyzing code clones [58][59].

V. Discussion and Validity Threats

The analysis reveals that clone detection areas, techniques and tools, and the related developer behavior are strongly represented in the sample. Whereas techniques and tools for clone detection are represented by 14 publications (35.9% of the total), developer behavior is the second largest group of primary studies, with a total of 8 publications, that is, 20.5% of all of the primary studies retrieved in the SMS (39). The areas that have been least studied in the literature found in the SMS are tools for clone visualization and patterns of cloning. Judging by the increase in the number of publications since 2016, the practice of copy and paste is of notable interest.

We identify as possible threats to validity: (i) coverage of research questions (RQs), (ii) bias towards certain publications, (iii) quality of the evaluation, and (iv) lack of knowledge of the area. It is probable that the proposed RQs could partially cover the study theme, which we try to mitigate by defining a work objective and raising several RQs in consensus, with the purpose of making the objective attainable. It is possible that in an SMS the process is directed towards a specific group of studies, which we avoid by forming a literature CG and by consensus building a search chain with explicit terms obtained from the CG. It is likely that the quality of the evaluation of the studies was not adequate due to lack of expertise in the research area, which we mitigate by including in the team an investigator with experience in the subject of code clone.

VI. Conclusions

This paper describes the SMS conducted to answer the following research questions. In this section, we have considered the 39 primary studies plus the 10 papers of the control group where one of them has been obtained in the set of primary studies, making a total of 48 papers analyzed.

RQ1. The research on copy and paste or code clone deals with eight areas: *General information of clone usage*, *developer behavior*, *techniques and tools for clone detection*, *techniques and tools for clone reuse*, *patterns of cloning*, *clone evolution*, *effects of code cloning in software maintenance and development*, and *tools for clone visualization*. Most primary studies and papers belonging to the CG (33.3%) focus on techniques and tools for clone detection, followed by the ones about developer behavior (27.1%) and the studies dealing with general information of clone usage (18.8%).

RQ2. Several patterns for using copy and paste have been defined: Elementary patterns (between, within, within and between, external paste) and complex patterns (repeat, distribution, relay, unknown). On the one hand, the elementary patterns are composed of a single copy and paste interaction involving one or more files. On the other hand, complex patterns are composed of two or more copy and paste incidents involving more than two files [13].

Acknowledgment

Work funded by FEDER/Spanish Ministry of Science and Innovation – Research State Agency: project MASSIVE, RTI2018-095255-B-I00, the R&D programme of Madrid (project FORTE, P2018/TCS-4314), and project PGC2018-097265-B-I00, also funded by: FEDER/Spanish Ministry of Science and Innovation – Research State Agency.

References

[1] G. Yarmish, and D. Kopec, "Revisiting novice programmer errors", ACM SIGCSE Bulletin, vol. 39(2), pp.131-137, 2007.

[2] R. Pittenger, "Building ASP.NET web pages dynamically in the code-behind", 2019, https://www.codeproject.com/Articles/25573/Building-ASP-NET-Web-Pages-Dynamically-in-the-Code.

[3] A. Vashisht, A. Sukhija, A. Verma, and P. Jain, "A detailed study of software code cloning", IIOAB J.-Special Issue: Comp. Science, vol. 9(2), pp. 20-32, 2018.

[4] R.N. Taylor, N. Medvidovic, and E. Dashofy, "Software architecture: Foundations, theory, and practice", John Wiley & Sons, First Ed., 2009.

[5] K. Wang, L. Zhang, and S. Yann, "A study on code clone evolution Analysis", in Proc. ICSESS'17. Beijing, China, pp. 340-345, 2017.

[6] K. Solanki, and S. Kumari, "Comparative study of software clone detection techniques", in Proc. MITicon'16. Bang-San, Thailand, pp. 152-156, 2016.

[7] D. Rattan, and J. Kaur, "Systematic mapping study of metrics based clone detection techniques", in Proc. AICTC'16. XBikaner, India, art. 76, pp. 1-7, 2016.

[8] G. Chatley, S. Kaur, and B. Sohal, "Software clone detection: A review", Int. J. Cont Theory and Applic., vol. 9(41), pp. 555-563, 2016.

[9] Q.U. Ain, W.H. Butt, M.W. Anwar, F. Azam, and B. Maqbool, "A systematic review on code clone detection", IEEE Access, vol. 7, pp. 86121-86144, 2019.

[10] N. Saini, S. Singh, and Suman, "Code clones: Detection and management", Procedia Computer Science, vol. 132, pp. 718-727, 2018.

[11] V. Saini, H. Sajnani, J. Kim, and C. Lopes, "SourcererCC and SourcererCC-I: Tools to detect clones in batch mode and during software development", in Proc. ICSE-C'16. Austin, TX, USA, pp. 597-600, 2016.

[12] B. Kitchenham, and S. Charters, "Guidelines for performing systematic literature reviews in software engineering", Tech. rep., Keele University and Department of Computer Science University of Durham, 2007.

[13] T.M. Ahmed, W. Shang, and A. E. Hassan, "An empirical study of the copy and paste behavior during development", in Proc. 12th Working Conf. on Mining Soft. Repositories. Florence, Italy, 2015, pp. 99-110.

[14] M. Balint, R. Marinescu, and T. Girba, “How developers copy”, in Proc. ICPC'06. Athens, Greece, pp. 1-10, 2006.

[15] D. Chatterji, J. C. Carver, and N.A. Kraft, “Claims and beliefs about code clones: Do we agree as a community? A survey”, in Proc. IWSC'12. Zurich, Switzerland, pp. 15-21, 2012.

[16] D. Chatterji, J.C. Carver, and N.A. Kraf, “Cloning: The need to understand developer intent”, in Proc. IWSC'13. San Francisco, CA, USA, pp. 14-15, 2013.

[17] C. Kapser, and M.W. Godfrey, “Cloning considered harmful considered harmful”, in Proc. WCRE'06. Benevento, Italy, pp. 645-692, 2006.

[18] M. Kim, L. Berman, T. Lau, and D. Notkin, “An ethnographic study of copy and paste programming practices in OOPL”, in Proc. ISESE’04. Redondo, Beach, USA, pp. 83-92, 2004.

[19] T.D. LaToza, G. Venolia, and R. DeLine, “Maintaining mental models: A study of developer work habits”, in Proc. ICSE'06. Shanghai, China, pp. 492-501, 2006.

[20] K.T. Stolee, S. Elbaum, and G. Rothermel, “Revealing the copy and paste habits of end users”, in Proc. VL/HCC'09. Corvallis, OR, USA, pp. 59-66, 2009.

[21] G. Zhang, X. Peng, Z. Xing, and W. Zhao, “Cloning practices: Why developers clone and what can be changed”, in Proc. ICSM'12. Trento, Italy, pp. 285-294, 2012.

[22] A. Khan, H.A. Basit, S.M. Sarwar, and M.M. Yousaf, “Cloning in popular server side technologies using agile development: An empirical study”, Pakistan J. Eng. and Applied Sciences, Vol. 22, pp. 1-13, 2018.

[23] J.F. Islam, M. Mondal, and C.K. Roy, “Bug replication in code clones: An empirical study”, in Proc. SANER'16. Suita, Japan, pp. 68-78, 2016.

[24] S. Bharti, and H. Singh, “An industrial study on developers’ prevalent copy and paste activities”, in Proc. ICNGCIS'17. Jammu, India, pp. 147-152, 2017.

[25] D. Chatterji, J.C. Carver, and N.A. Kraft, “Code clones and developer behavior: Results of two surveys of the clone research community”, Emp. Soft. Eng., vol. 21(4), pp. 1476-1508, 2016.

[26] A. Ciborowska, N.A. Kraft, and K. Damevski, “Detecting and characterizing developer behavior following opportunistic reuse of code snippets from the web”, in Proc. MSR'18. Gothenburg, Sweden, pp. 94-97, 2018.

[27] L. Müller, M.S. Silveira, and C.S. de Souza, “Do I know what my code is saying?: A study on novice programmers’ perceptions of what reused source code may mean”, in Proc. IHC'18. Belém, Brazil, pp. 1-10, 2018.

[28] T. Ohta, H. Murakami, H. Igaki, Y. Higo, and S. Kusumoto, “Source code reuse evaluation by using real/potential copy and paste”, in Proc. IWSC'15. Montreal, pp. 33-39, 2015.

[29] B. Van Bladel, A. Murgia, and S. Demeyer, “An empirical study of clone density evolution and developer cloning tendency”, in Proc. SANER'17. Klagenfurt, Austria, pp. 551-552, 2017.

[30] B. Xu, L. An, F. Thung, F. Khomh, and D. Lo,“Why reinventing the wheels? An empirical study on library reuse and re-implementation”, Empirical Software Engineering, pp. 1-35, 2019.

[31] M.S. Aktas, and M. Kapdan, “Structural code clone detection methodology using software metrics”, IJSEKE, vol. 26(2), pp. 307-332, 2016.

[32] M. Gharehyazie, B. Ray, M. Keshani, M.S. Zavosht, A. Heydarnoori, and V. Filkov, “Cross-project code clones in gitHub”, Empirical Software Engineering, vol. 24, pp. 1538-1573, 2019.

[33] T.A.D. Henderson, and A. Podgurski, “Rethinking dependence clones”, in Proc. IWSC'17. Klagenfurt, Austria, pp. 66-74, 2017.

[34] B. Joshi, P. Budhathoki, W.L. Woon, and D. Svetinovic, “Software clone detection using clustering approach”, in: Arik S., Huang T., Lai W., Liu Q. (eds). Neural Information Processing. ICONIP 2015 (pp. 520-527). Lecture Notes in Computer Science, vol 9490. Springer, 2015.

[35] T. Kamiya, “An execution-semantic and content-and-context-based code-clone detection and analysis”, in Proc. IWSC'15. Montreal, Canada, pp. 1-7, 2015.

[36] K. Kim, D. Kim, T.F. Bissyandé, E. Choi, L. Li, J. Klein, and Traon, “FaCoY: A code-to-code search engine”, in Proc. ICSE'18. Gothenburg, Sweden, pp. 1-12, 2018.

[37] M. Mondal, C.K. Roy, and K.A. Schneider, “SPCP-Miner: A tool for mining code clones that are important for refactoring or tracking”, in Proc. SANER'15. Montreal, Canada, pp. 484-488, 2015.

[38] A.-F. Mubarak-Ali, S. Sulaiman, S.M. Syed-Mohamad, and Z. Xing, “Code clone detection and analysis in open source applications”, Comp. Syst. Softw. Eng.: Conc., Meth., Tools, and Appl., pp. 1112-1127, 2018.

[39] B. Priyambadha, and S. Rochimah, “Behavioral analysis for detecting code clones”, Telkomnika, vol. 16(3), pp. 1264-1275, 2018.

[40] S. Reddivari, and M.S. Khan, “A topic modeling approach for code clone detection”, in Proc. SEKE'18. San Francisco Bay, USA, pp. 486-491, 2018.

[41] M. Sudhamani, and L. Rangarajan, “Code similarity detection through control statement and program features”, Expert Systems with Applications, vol. 132, pp. 63-75, 2019.

[42] J. Svajlenko, and C.K. Roy, “Fast and flexible large-scale clone detection with cloneworks”, in Proc. ICSE-C'17. Buenos Aires, Argentina, pp. 27-30, 2017.

[43] C. Wijesiriwardana, and P. Wimalaratne, “Component-based experimental testbed to facilitate code clone detection research”, in Proc. ICSESS'17. Beijing, China, pp. 165-168, 2017.

[44] S. Abid, S. Javed, M. Naseem, S. Shahid, H.A. Basit, and Y. Higo, “CodeEase: Harnessing method clone structures for reuse”, in Proc. IWSC’17. Klagenfurt, Austria, pp. 24-30, 2017.

[45] Y. Lin, X. Peng, Z. Xing, D. Zheng, and W. Zhao, “Clone-based and interactive recommendation for modifying pasted code”, in Proc. ESEC/FSE'15. Bergamo, Italy, pp. 520-531, 2015.

[46] K. Narasimhan, C. Reichenbach, and J. Lawall, “Cleaning up copy–paste clones with interactive merging”, Automated Software Engineering, vol. 25, pp. 627-673, 2018.

[47] A. Ohtani, Y. Higo, T. Ishihara, and S. Kusumoto, “On the level of code suggestion for reuse”, in Proc. IWSC'15. Montreal, Canada, pp. 26-32, 2015.

[48] T. Zhang, and M. Kim, “Poster: Grafter: Transplantation and differential testing for clones”, in Proc. ICSE-Companion'18. Gothenburg, Sweden, pp. 422-423, 2018.

[49] J. Kanwal, K. Inoue, and O. Maqbool, “Refactoring patterns study in code clones during software evolution”, in Proc. IWSC'17. Klagenfurt, Austria, pp. 45-46, 2017.

[50] J. Kanwal, H.A. Basit, and Maqbool, “Structural clones: An evolution perspective”, in Proc. IWSC'18. Campobasso, Italy, pp. 9-15, 2018.

[51] M. Mondal, C.K. Roy, and K.A. Schneider, “Bug-proneness and late propagation tendency of code clones: A Comparative study on different clone types”, J. of Systems and Softw., vol. 144, pp. 41-59, 2018.

[52] T.L. Nguyen, A. Fish, and M. Song, “An empirical study on similar changes in evolving software”, in Proc. EIT'18. Rochester, USA, pp. 560-563, 2018.

[53] J.R. Pate, R. Tairas, and N.A. Kraft, “Clone evolution: A systematic review”, J. Softw.: Evol. Proc., vol. 25(3), pp. 261-283, 2013.

[54] F. Zhang, X. Su, W. Zhao, and T. Wang, “An empirical study of code clone clustering based on clone evolution”, J. of Harbin Institute of Technology (New Series), vol. 24(2), pp. 10-18, 2017.

[55] A. Lerina, and L. Nardi, “Investigating on the impact of software clones on technical debt”, in Proc. TechDebt'19. Montreal, Canada, pp. 108-112, 2019.

[56] M. Mondal, C.K. Roy, and K.A. Schneider, “Does cloned code increase maintenance effort?”, in Proc. IWSC'17. Klagenfurt, Austria, pp. 1-7, 2017.

[57] S. Wagner, A. Abdulkhaleq, K. Kaya, and A. Para, “On the relationship of inconsistent software clones and faults: An empirical study”, in Proc. SANER'16. Suita, Japan, pp. 79-89, 2016.

[58] D. Mondal, M. Mondal, C.K. Roy, K.A. Schneider, S. Wang, and Y. Li “Towards visualizing large scale evolving clones”, in Proc. ICSE-Companion'19. Montreal, Canada, pp. 302-303, 2019.

[59] H. Murakami, Y. Higo, and S. Kusumoto, “ClonePacker: A tool for clone set visualization”, in Proc. SANER'15. Montreal, Canada, pp. 33-39, 2015.