A Case of Plagiarism and Intellectual Property Theft in Software Engineering

Authors Accused of Plagiarism:

M. Aruna, M. P. Suguna Devi, and M. Deepa
V.L.B. Janakiammal College of Engineering and Technology
Coimbatore, India

Publication in Which the Plagiarized Material Appears:


IEEE makes this publication available at:

http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=4580073&isnumber=4579840

The same publication at the following link shows visually the portions that were plagiarized from the papers by Sarkar et al.:

http://engineering.purdue.edu/theft/ArunaSugunaDeviDeepa.pdf

Bottom Line on How Much Material Was Stolen:

Roughly 90% of the material in the accused publication was lifted verbatim from the two papers by Sarkar, Rama, and Kak. These are listed below.
Publications from Which the Material was Stolen:


In the rest of this document, we will refer to the first publication above as the **TSE paper** and the second publication as the **APSEC paper**.

Specifics of the Material Lifted Verbatim:

1. The following text appears in the Introduction section of the TSE publication by Sarkar et al.:

   “Our ongoing effort, from which we draw the work reported here, is focussed on the case of reorganization of legacy software, consisting of millions of lines of non-object-oriented code that was never modularized or poorly modularized to begin with. We can think of the problem as reorganization of millions of lines of code residing in thousands of files in hundreds of directories into modules, where each module is formed by grouping a set of entities such as files, functions, data structures, and variables into a logically cohesive unit. Furthermore, each module makes itself available to the other modules (and to the rest of the world) through a published API.”

   Except for one word, the above material was lifted wholesale by the accused authors; it appears in their Introduction section. The only word that is different is their deletion of “non” in “non-object-oriented” in the first sentence.
2. The following text appears in the Introduction section of the TSE publication by Sarkar et al.:

“..... the quality of modularization has more to do with partitioning software into more maintainable (and more easily extendible) modules on the basis of the cohesiveness of the service provided by each module.”

This material was lifted verbatim by the accused authors. It appears in their Introduction section.

3. The first paragraph of Section 2 of Sarkar et al.’s TSE paper contains:

“Some of the earliest contributions to software metrics deal with the measurement of code complexity [1], [2] and maintainability [3] based on the complexity measures proposed in [1], [2]. From the standpoint of code modularization, some of the earliest software metrics are based on the notions of coupling and cohesion [4], [5]. Low intermodule coupling, high intramodule cohesion, and low complexity have always been deemed to be important attributes of any modularized software”

Except for the deletion of “Some of,” the accused authors have reproduced word-for-word the rest of the paragraph in their Section 2, paragraph 1.

4. The third paragraph of Section 2 of Sarkar et al.’s TSE paper starts with the following sentence:

“With regard to modularity, Briand et al. [8] have given us a generic formalization of such fundamental notions as module and system, and such metrical notions as coupling, cohesion, and complexity.”

The accused authors have reproduced this sentence verbatim in Section 2 of their publication.

5. The 5th paragraph of Section 2 Sarkar et al.’s TSE paper contains:

“Schwanke also characterizes modules on the basis of function-call dependencies. If a function A calls function B, then, in the approach used by Schwanke, both A and B presumably belong to the same module.”
These two sentences have been reproduced verbatim in the third paragraph of the accused publication.

6. The 6th paragraph of Section 2 of Sarkar et al.’s TSE paper contains the following material:

“.... the lines of formulating metrics in the context of developing code modularization algorithms, Mancoridis et al. [18], [19] have used a quantitative measure called Modularization Quality (MQ) that is a combination of coupling and cohesion. Cohesion is measured as the ratio of the number of internal function-call dependencies that actually exist to the maximum possible internal dependencies, and coupling is measured as the ratio of the number of actual external function-call dependencies between the two subsystems to the maximum possible number of such external dependencies.”

The accused authors have stolen this material verbatim. It constitutes the next to the last paragraph of their Section 2.

The above instance is an example of the fact that, probably because of high anxiety, thieves often make mistakes during the act of committing thievery. While lifting the above material, the accused authors forgot to also lift the introductory clause of the first sentence. Without the introductory clause, the first sentence of how this paragraph appears in the accused publication makes no sense.

7. The following text appears toward the end of Section 2 of Sarkar’s TSE paper:

“The contribution by Wen and Tzerpos [29] adds a new consideration to the calculation of cohesion coupling metrics for the purpose of software clustering. These authors first try to isolate what they refer to as omnipresent objects, these being heavily used objects and functions in a software system, before the calculation of the more traditional coupling-cohesion metrics.”

This paragraph was copied verbatim and it constitutes the last paragraph of Section 2 in the accused publication.

8. Section 3 of Sarkar et al.’s APSEC paper starts with the following large paragraph:
“We believe that many of these previous approaches suffer from shortcomings with regard to the goals we have in mind. The approaches that carry out software partitioning purely on the basis of function call dependencies (or file-dependencies that are derived from function-call dependencies) are obviously not suitable for meeting our goals. Function call dependencies are semantically orthogonal to the groupings on the basis of cohesiveness of service. To elaborate, in code partitioning on the basis of function call dependencies, if a function A calls a function B, then both A and B must belong together in the same module. But using function call dependencies as the sole basis for modularization runs counter to the very spirit of what is meant by modules in modern code writing. Modules pull together functions not because they call one another, but because they serve similar purposes with respect to the rest of the software. For example, the number of intra-module function calls in the java.util module (referred to as a package in the Java parlance) is minimal. The main reason for why the functions in the java.util module belong together is because they provide very similar services to the rest of the software.”

The accused authors have stolen this entire paragraph and placed in their Section 3. In fact, half of their Section 3 consists of this stolen material.

9. Section 4 of Sarkar’s APSEC paper containing the following material:

Modern software engineering dictates that large software be organized along the following lines:

(a) The software system should consist of a set of modules where each module is a collection of data structures and functions that together offer a well-defined service. In other words, the structures used for representing knowledge and any associated functions in the same module should cohere on the basis of similarity-of-service as opposed to on the basis of function call dependencies.

(b) The modules should interact with one another only through the exposed API functions. With regard to code maintenance, this is desirable for isolating faults and rectifying them quickly.

(c) Whenever feasible, the modules should be organized in a hierarchical manner in a set of layers. A layer should only be aware of the layers below it (that is, function calls are only made to the lower layers) and should not be aware of the layers above it. A layer can be thought of as horizontal partitioning of the system.
(d) Modules should be independently testable and releasable. The impact of a single change should typically stay confined to a module and should minimally impact other modules.

Except for the last sentence of the third item and the last sentence of the fourth item, the above material was reproduced verbatim by the accused authors in their Section 4.

10. The last paragraph of Section 4 of Sarkar et al.’s APSEC paper consists of the following text:

“These considerations have led us to formulate the following metrics for measuring the quality of module interactions. Of the four considerations listed above, the testability related consideration is more complex and depends on what tools are brought to bear and what protocols are used for testing code. Therefore, for now, we will ignore this consideration. Given the importance of this issue, we certainly plan to take up testability related issues in a future research contribution. Additionally, as was stated in the introduction, metrics that focus solely on the interactions between the modules cannot exist in isolation from the metrics that measure other qualities of code modularization. Therefore, the set of metrics shown below includes those that are needed to simultaneously report these other attributes.”

Except for the second and the third sentences, this entire paragraph appears verbatim at the end of Section 3 of the accused publication.

11. The following material appears in Section 3 of Sarkar et al.’s TSE paper:

“The structures used for representing knowledge and any associated functions in the same module should cohere on the basis of similarity-of-service as opposed to, say, on the basis of function call dependencies. Obviously, every service is related to a specific purpose. We present the following principles as coming under the “Similarity of Purpose” rubric:

• Maximization of Module Coherence on the Basis of Commonality of Goals.”

The above mentioned material has been lifted verbatim and appears as Section 4.1 in the accused publication.
12. All of Section 4.2 of the accused publication has been lifted verbatim from Sarkar et al.’s TSE publication (Principle P2 in Section 3).

13. Table 1 in the accused publication was constructed by lifting verbatim the rows 2, 3, and 4 from Table 2 in Section 8 of Sarkar et al.’s TSE paper.

14. The first paragraph of Section 4 of Sarkar et al.’s TSE paper contains:

“...coupling-based structural metrics that provide various measures of the function-call traffic through the API’s of the modules in relation to the overall function-call traffic.”

This text constitutes the first paragraph of Section 5 of the accused publication.

15. The entire Section 5.1 of the accused publication was lifted from Section 4.1 of Sarkar et al.’s TSE paper. The name of the metric and its formulation are verbatim copies of the material in Section 4.1 of Sarkar et al.’s paper. What comes after the formulation is cut-and-paste in bits and pieces from Section 4.1 of Sarkar et al.’s paper.

16. Here is another example of the stupidity of the accused authors: The last paragraph of their Section 5.1 was lifted from Section 4.1 of Sarkar et al.’s TSE paper, but without the introductory clause of the first sentence. The introductory clause used by the accused authors does not go with the rest of that sentence and the paragraph.

17. The entire Section 5.2 of the accused publication was lifted verbatim from Section 4.2 of Sarkar et al.’s paper.

18. The entire Section 5.3 of the accused publication was lifted verbatim from Section 4.3 of Sarkar et al.’s TSE paper.
19. The entire Section 5.4 of the accused publication was lifted from Section 4.4 of Sarkar et al.’s TSE paper.

20. The entire first paragraph of Section 6 of the accused publication was lifted from Section 9 of Sarkar et al.’s TSE paper. This stolen paragraph consists of the first sentence of the first paragraph of Section 9 and the second sentence of the second paragraph of the same section of the Sarkar et al. paper.

21. The first half of the second paragraph of the accused publication was lifted wholesale from Section 9 of Sarkar et al.’s TSE paper.

This PDF file is a part of the documentation posted at

http://engineering.purdue.edu/theft/

with regard to this case of blatant plagiarism. This website also points to a marked-up copy of the accused publication in which the stolen material has been highlighted.