Techniques for the Identification of Crosscutting Concerns: A Systematic Literature Review

Ingrid Marçal, Rogério Eduardo Garcia, Danilo Medeiros Eler,
Celso Olivete Junior, Ronaldo C. M. Correia

São Paulo State University – UNESP
College of Science and Technology
Campus at Presidente Prudente, São Paulo – Brazil
Email: in.marcal@gmail.com, {rogerio, daniloeler, olivete, ronaldo}@fct.unesp.br

Abstract. Modularization is a goal difficult to achieve in software development. Some requirements, named crosscutting concerns, cannot be clearly mapped into isolated source code units, and their implementations tend to cut across multiple units. Although several researches propose new approaches to identify crosscutting concerns, few works aim to provide analysis, synthesis and documentation of the aspect mining literature. To address this research gap, we conducted a systematic literature review to provide researchers with a state-of-the-art of the existing aspect mining techniques. We point out challenges and open issues on most of the techniques analyzed that could be improved in further researches.

Keywords: Crosscuttingness, review, aspects, crosscutting-concerns, concerns

1 Introduction

Crosscutting concerns are pieces of functionality that have their implementation tangled and/or scattered among various modules in a software system. They are difficult to identify and hinder program comprehension. The aspect mining research area is concerned with the development of tools and techniques to support the identification of crosscutting concerns in legacy software. Many techniques have been proposed from source code analysis [6] [5] [8] to the use of information available in software repositories [7] [3].

Whereas researchers demonstrate empirically [15][16] [17] [18] the negative impact of crosscutting concerns in software internal and external quality metrics, increases the number of new techniques and tools for identifying them. Few works, however, aim to provide the analysis, synthesis and documentation of aspect mining literature [30] [22]. To address this research gap, our systematic literature review (SLR) focuses on analyzing and synthesizing empirical evidence on aspect mining, aiming to provide researchers with a state-of-the-art of the existing techniques.

Our SLR helped to spot issues beyond weel known aspect mining problems as the subjectivity of most analysis, imprecise definitions and poor discussion of results. Our

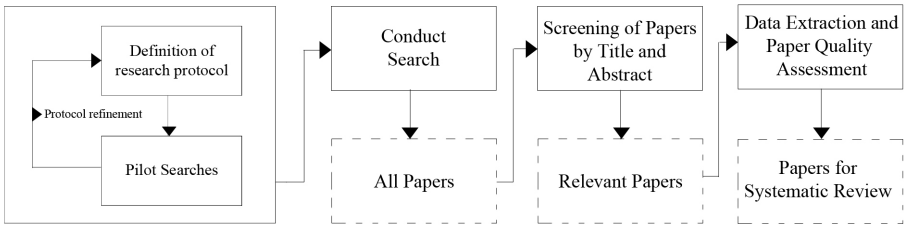

Fig. 1. Systematic Literature Review Process

in-depth analysis of aspect mining techniques can serve as a road map for researchers to improve their techniques and achieve better results. Based on our results and observations, we also propose new research directions to improve the existing aspect mining techniques.

To present our SLR and results obtained, the remainder of this paper is organized as follows. Section 2 describes the adopted research method. Section 3 gives a brief overview of the studies selected for this SLR and Section 4 describes the related work. Finally, Section 6 draws conclusions and presents the observed opportunities for future work in aspect mining.

2 Research Method

This research has been carried out as a Systematic Literature Review, which involves research methodologies developed to collect and assess documented evidence related to a specific topic [2]. SLRs are formal, systematic and follow a rigorous methodology predefined under a research protocol [2], which helps selecting relevant studies, improving rigor and making the reviewing a repeatable process. A protocol contains the definition of inclusion/exclusion criteria, questions for quality assessment, methods for data extraction and the research questions. The protocol for this SLR was elaborated based on rules, policies and procedures determined by the Campbell Collaboration[1].

We conducted a series of pilot searches to refine the definitions in the protocol. After the refinement, we carried out a primary search to obtain the set of papers. Then, we conducted a screening of papers by title and abstract to select only those relevant to our SLR. Finally, we evaluated each relevant paper according to our quality assessment criteria and extracted important information. Our systematic literature process is depicted in Figure 1. The following sections describe our research protocol in terms of databases and search strategy, studies selection criteria, quality assessment and data extraction. We, also, state our research question.

2.1 Databases and Search Strategy

A search string was executed in four different databases (see Table 1) considering papers published from 2000 to 2014. We defined the following standard search string:

[1] www.campbellcollaboration.org

((crosscutting OR crosscut OR cross-cutting) AND (concern OR concerns)) AND (("aspect mining") OR ("separation of concerns") OR ("code mining") OR "coding mining")

However, different databases employ a particular search syntax. Thus, we used our search string to create an equivalent for each database. Our search resulted in 1289 papers, and 1274 were excluded for non-compliance with our selection criteria or do not achieve the minimum score defined in our quality assessment (See Sections 2.2 and 2.3). Our quality assessment resulted in 53 studies relevant to this SLR. Due to space issues this paper we only discuss details about 14 selected studies and draw conclusions based on all 53 selected studies. The number of papers retrieved from each database is shown in Table 1.

Table 1. Selected Databases

Database	Number of Studies
ACM Digital Library	214
IEEE Xplore	928
SpringLink	97
Willey Inter Science Journal Finder	50

2.2 Studies Selection Criteria

From all papers found by our search strategy, we selected to this SLR only those that showed empirical evidence of their results. Furthermore, we exclude all papers related to: Early aspects, editorials, tutorials, summaries and panels. We also exclude studies that proposed non-automatic or semi-automatic techniques.

2.3 Quality Assessment and Data Extraction

We elaborated 18 questions (Table 2) which corresponds to four analysis criteria considered important to ensure quality: (1) Initial Filter; (2) Rigor; (3) Credibility; (4) Relevance. Each question was answered according to the following. 0 point: Not acceptable; 0.5 point: Poorly Acceptable; 1 point: Acceptable. In Table 2, Questions 1-3 corresponds to the initial filter and helps to decide whether or not a paper must be considered for further analysis. Questions 4-12 assess the rigor with which the paper was elaborated. Questions 13-17 refer to the study credibility and question 18 evaluates the relevance of a paper. Final classification of a paper is obtained by adding the points obtained in each question (The classification is shown in Table 4).

Only papers classified with *Good Quality* and *Very Good Quality* were included in this SLR and read fully to collect relevant data . We used a set of questions (see Table 3) to help extracting all relevant information from each paper.

Table 2. Quality Assessment Questions

ID	Question
Q1	Is the study based on research or is it just an experience report?
Q2	Is the study main focus on crosscutting concerns identification/comprehension?
Q3	Does the study present experimental data or real application of some aspect mining technique?
Q4	Is there a satisfactory description of the context in which the study is inserted?
Q5	Is it clear how the results were obtained?
Q6	Were the results extraction methods well justified?
Q7	Were the results extraction methods well described?
Q8	Is there a clear description of the process analysis of the results?
Q9	Were the results well justified?
Q10	Were contradictory information considered when assessing the results?
Q11	Were appropriate metrics used to assess the results?
Q12	Did researchers critically examine their own influence on the results obtained during the collection and analysis of data?
Q13	Are the results clearly described?
Q14	Did the researchers discuss the credibility of the search results?
Q15	Were the approach limitations/disadvantages discussed?
Q16	Is there an adequate discussion of the results both negatives and positives?
Q17	Is the researchers conclusion justified by their results?
Q18	Is there a discussion about the research contributions to the Software Engineering area and for the identification of crosscutting concerns?

3 Selected Studies: a brief discussion

Analysing existing aspect mining techniques we notice different characteristics related to: approach, analysis type, granularity level, symptoms of crosscuttingness, manual effort involved and metrics for results evaluation. Also, each technique has its own limitations which impacts on results. This section presents an overview of 14 of the 53 selected studies according to their approach: Clone detection (Section 3.1), Fan-in Analysis (Section 3.2), Graph-based techniques (Section 3.3), Clustering Analysis (Section 3.4) and Development-history based techniques (Section 3.5). We show the selected studies in Table 5.

3.1 Clone Detection Techniques

Clones are generally defined as a code fragment identical or similar to another code fragment. They occur most due to copying existing code and modifying it. Bruntink et al. [5] [6] use three different clone detection tools to find the maximal possible number of code fragments implementing crosscutting concerns. Generally, clone detection tools return tuples of code clones. Bruntink et al. however, studied *clone classes* instead of clone tuples. A clone class is a set of code fragments with an equivalence relation between any pair of fragments in the set [21]. The approach aims to evaluate the number of clone classes needed to cover all the extent of a crosscutting concern (i.e., all fragments of code implementing a crosscutting concern). To validate their results, Bruntink

Table 3. Data Extraction Questions (Adapted from [14])

Info.1 Study Identifier (ID)
Info.2 Citation
Info.3 Abstract
Info.4 Study type (Qualitative, quantitative)
Info.5 Study goals
Info.6 Study Hypothesis
Info.7 Study scenario (Which software system was used to mine for crosscutting concerns)
Info.8 Results assessment method (Manual/automatic)
Info.9 Technique application (Combined/isolated)
Inf.10 Results evaluation
Inf.11 Case study applied (if applicable)
Inf.12 Analysis method (static, dynamic, structural, behavioral)
Inf.13 Results granularity (methods, classes, code fragments, etc)
Inf.13 Hypotheses for the existence of crosscutting concerns (clone detection, high number of methods calls, etc)
Inf.14 Observed crosscutting concerns symptoms (Scattering, tangling)
Inf.15 User involvement in the identification process of crosscutting concerns (Input data analysis, manual validation of final results)
Inf.16 Findings and conclusions
Inf.17 Approach limitations, vantages and disadvantages
Inf.18 Study relevance
Inf.19 Other important observations

Table 4. Studies Final Classification Based on Punctuation

Punctuation	Meaning
0-10	Very Poor Quality
10.5-13	Poor Quality
13.5-15.5	Good Quality
16-18	Very Good Quality

Table 5. Selected Studies

Approach	Selected Studies
Fan-in Analysis	[26] [28]
Graphs	[23] [34] [24] [33]
Clone detection	[6] [5]
Clustering	[35] [12] [25] [29]
Software Development History	[7] [3]

et al. depend upon a given set of confirmed crosscutting concerns annotated manually by an expert on the system that the clone detection tools were executed. Although the use of well established coding conventions and idioms required, Bruntink et al. show that clone detection techniques can be used to find most code fragments implementing crosscutting concerns.

3.2 Fan-in Analysis

Marin et al. [26] [28] manually analyze methods with high fan-in value to classify them as crosscutting concerns or not. The fan-in value is computed as the number of calls to a method. Methods with high fan-in value have higher changes to be classified as crosscutting concerns.

Fan-in analysis is a generative approach based on the scattering symptom of crosscuttingness. According to Marin et al. if the same functionality is scattered throughout the code it is reasonable to assume that this functionality is implemented in a helper method that is called by several other methods. Each call contributes to the fan-in value of the helper method. Fan-in analysis generates crosscutting concerns candidates automatically (*seeds*) which diminish the manual effort made to analyze the concerns and crosscutting concerns in a system.

Fan-in analysis restricts the definition of crosscuttingness to methods frequently invoked from other methods. Some crosscutting concerns, however, do not exhibit such behaviour and consequently are pruned by the technique. Also, fan-in analysis potentially returns a high number of false-positives which are filtered manually or by setting the right configurations of automatic tools.

3.3 Graph-Based Techniques

Graphs allow to represent elements of code as a set of nodes and edges which show the relationship between nodes. Several studies selected for our SLR use graphs to represent code elements and the relationships among them [23] [34] [24] [33]. Zhang and Jacobsen [34] [33] use graphs to propose a model to automate the manual process followed to identify crosscutting concerns. Zhang and Jacobsen consider each node in a graph a different element in a software system (e.g., components, packages, classes and methods) and each edge the relationship among them. Krinke [23] [24], however, generates a different graph for each method. Each node on the graph represents a line of code that implements the method and edges show the execution flow of them.

We notice that, regardless the common use of graphs and the similarity in the approaches to identify crosscutting concerns, each author uses graphs to model different types of data (methods, fragments of code, etc.).

3.4 Clustering Analysis

Clustering Analysis groups data with similar characteristics. From the aspect mining point of view, clustering techniques aim to generate groups, called clusters, each containing all fragments of code implementing a specific crosscutting concern in the system and one extra cluster containing all elements of code that do not implement any crosscutting concern [29].

The generated clusters are analyzed to determine which are part of a crosscutting concern. Cojocar and Czibula [12] mine for crosscutting concerns searching for symptoms of scattering in methods. The authors apply several clustering techniques (k-means [19], Fuzzy Clustering [20], Genetic Clustering [1], Hierarchical Agglomerative Clustering [20], *Kam* [32] and *HAM* [31]) to validate their efficiency in grouping all methods

implementing crosscutting concerns in a unique cluster. Furthermore, they investigate the number of crosscutting concerns found by each clustering technique. It was observed that most crosscutting concerns are also found by Fan-in Analysis in [26] [28].

3.5 Development-history Based Techniques

Canfora and Cerulo [7] and Breu and Zimmermann [3] observed that crosscutting concerns are inserted in a software system and extended by a series of changes made in later commits. Software version control systems provide a wealth of historical data related to changes in software artifacts, which can be used to identify crosscutting concerns.

In [7] each transaction is a set of files committed by the same author, with the same change history, comments and with time span between commits less than 200 seconds. From these transactions Canfora and Cerulo study the lines of code that were added, changed or deleted to observe the evolution of crosscutting concerns.

Breu and Zimmermann [3] propose a technique that consider only changes that add code related to method calls. The technique is based on dynamic approaches as in [4], although Breu and Zummermann use information from the development history instead of execution traces.

4 Related Works

Kellens et al. [22] provide an in-breadth survey and comparison of techniques and tools for aspect mining. Their work focus on automated techniques for static or dynamic aspect mining. Ceccato et al. [9] provide a qualitative comparison of three aspect mining techniques (Fan-in Analysis, Identifier Analysis and Dynamic Analysis) applying them to JHotDraw. In [10], Ceccato et al. combine Fan-in Analysis, Identifier Analysis and Dynamic Analysis.

Cojocar et al. [11] noticed the importance of considering all the characteristics of aspect mining techniques for comparing and evaluating them. The authors propose a set of criteria to compare existing aspect mining techniques and a set of new evaluation measures to compare mining results. Marin et al. [27] proposes a common framework based on crosscutting concern sorts which allows easier comparison and combination of aspect mining techniques.

Most authors focus on comparing or combining a set of techniques or proposing methods and frameworks to support the comparison and combination of them. Regarding all cited related work, only Durelli et al. [13] provides a systematic review closely related to ours: the systematic review conducted in [13] aims to find out which techniques perform better on identifying crosscutting concerns and how to combine them for improving precision and recall. This work focus on the analysis and synthesis of challenges and issues regarding existing aspect mining techniques to support improvements.

5 Threats to Validity

Our search for primary studies was conducted in four different search engines, however it is possible we have missed relevant studies. The questions to select studies were

defined focusing on our search goal and are broad enough to cover different issues. Some questions defined in our data extraction form were not obvious to answer, and most answer required interpretation while reading the papers. To ensure the validity of our interpretations we analyzed several auxiliary sources as technical reports and papers used as reference by the selected studies. Also, we are aware of selecting only classified papers might be restrictive, but they were enough to draw important conclusions.

6 Concluding Remarks

This systematic review aims to expose some existing aspect mining techniques and detect the issues that impact on their efficiency. We analyzed several studies on aspect mining techniques (total of 53) and observed differences related to the type of analysis they execute (static, dynamic, etc.), their granularity level, hypothesis, symptoms of crosscuttingness, manual interference and metrics used to evaluate results.

Most techniques provides a poor evaluation of their results. Those that in fact discuss them show unsatisfactory results. Low precision is the worst problem encountered on the majority of the analyzed techniques which indicates a high number of false-positives. Furthermore, the results analysis is often done manually, which makes the results susceptible to subjectivity.

Poor evaluation of results is accompanied by lack of documentation. most researchers do not affer a detailed documentation of their results but fragments of code as examples of crosscutting concerns and, in general, a weak discussion of the motifs to classify them as crosscutting. Furthermore, some studies perform experiments in proprietary software which difficult repetition by other researchers.

Lack of documentation and experiments in proprietary software prevents establishing common benchmarks for aspect mining. Without benchmarks it is hard to compare or evaluate results forming a vicious cycle that hampers the evolution of the aspect mining research.

The lack of a common definition for what characterizes crosscuttingness is a well known problem in aspect mining. Most researches, however, is concerned in proposing new tools, approaches and techniques. Only a few studies offer valuable information about the main characteristics and evolution of crosscutting concerns in software systems.

Also, it was observed that techniques which strongly depends upon syntactic information hardly identify symptoms of entanglement. An alternative is using semantic information, however, it is difficult to capture and relate the meaning of syntactically different fragments of code.

The directions for aspect mining research that emerged from our SLR are the need for exploratory studies about the nature of crosscuttingness. Knowing why, how, where and when crosscutting concerns emerge in software systems helps to formulate better definitions for crosscuttingness and consequently improve current techniques. Capturing design intent for example, helps to explore how developers make design decisions in the first place. This can be used to evaluate in what circumstances crosscutting concerns are incorporated in code.

The use of semantic information and combined techniques is still faulty. Indeed, our observations revel that every approach relies on different types of analysis to search for crosscutting concerns. Most, however, relies on syntactic/dynamic information and do not try to combine different approaches to suppress possible deficiencies.

7 Acknowledgement

The authors acknowledge the financial support of the Brazilian financial agency São Paulo Research Foundation (FAPESP) – grant (2013/03452-0).

References

1. Babu, G.P., Murty, M.N.: A near-optimal initial seed value selection in k-means algorithm using a genetic algorithm. Pattern Recogn. Lett. 14(10), 763–769 (Oct 1993)
2. Biolchini, J., et al.: Systematic review in software engineering. System Eng. and Comp. Sci. Dept. COPPE/UFRJ, Tech. Rep. ES 679(05), 45 (2005)
3. Breu, S., Zimmermann, T.: Mining aspects from version history. In: Proc. 21st IEEE/ACM Int. Conf. on Automated Softw. Eng. pp. 221–230 (Sept 2006)
4. Breu, S., Krinke, J.: Aspect mining using event traces. In: Proc. 19th IEEE Int. Conf. on Automated Softw. Eng. pp. 310–315. ASE '04, IEEE Comput. Soc., Washington, DC, USA (2004), http://dx.doi.org/10.1109/ASE.2004.12
5. Bruntink, M., et al.: An evaluation of clone detection techniques for crosscutting concerns. In: Proc. 20th IEEE Int. Conf. on Soft. Maint. pp. 200–209 (Sept 2004)
6. Bruntink, M., et al.: On the use of clone detection for identifying crosscutting concern code. IEEE Trans. on Softw. Eng. 31(10), 804–818 (Oct 2005)
7. Canfora, G., Cerulo, L.: How crosscutting concerns evolve in jhotdraw. In: 13th IEEE Int. Workshop on Softw. Tech. and Eng. Practice. pp. 65–73 (2005)
8. Canfora, G., et al.: On the use of line co-change for identifying crosscutting concern code. In: 22nd IEEE Int. Conf. on Soft. Maint. pp. 213–222 (Sept 2006)
9. Ceccato, M., et al.: A qualitative comparison of three aspect mining techniques. In: Proc. 13th Int. Workshop on Program Comprehension. pp. 13–22 (May 2005)
10. Ceccato, M., et al.: Applying and combining three different aspect mining techniques. Softw. Quality J. 14 (2006)
11. Cojocar, G.S., Şerban, G.: On some criteria for comparing aspect mining techniques. In: Proc. 3rd Workshop on Linking Aspect Tech. and Evol.
12. Cojocar, G., Czibula, G.: On clustering based aspect mining. In: Proc. 4th Int. Conf. on Intelligent Comp. Commun. and Processing. pp. 129–136 (Aug 2008)
13. Durelli, R.S., et al.: A systematic review on mining techniques for crosscutting concerns. In: Proc. 28th Annu. ACM Symp. on Appl. Comp. ACM, New York, NY, USA (2013)
14. Dyb, T., Dingsyr, T.: Empirical studies of agile software development: A systematic review. Inf. and Softw. Tech. 50, 833 – 859 (2008)
15. Eaddy, M., Zimmermann, T., Sherwood, K., Garg, V., Murphy, G., Nagappan, N., Aho, A.: Do crosscutting concerns cause defects? Software Engineering, IEEE Transactions on 34(4), 497–515 (July 2008)
16. Filho, F.C., Cacho, N., Figueiredo, E., Maranhão, R., Garcia, A., Rubira, C.M.F.: Exceptions and aspects: The devil is in the details. In: Proceedings of the 14th ACM SIGSOFT International Symposium on Foundations of Software Engineering. pp. 152–162. SIGSOFT '06/FSE-14, ACM, New York, NY, USA (2006)

17. Garcia, A., Sant'Anna, C., Figueiredo, E., Kulesza, U., Lucena, C., von Staa, A.: Modularizing design patterns with aspects: A quantitative study. In: Proceedings of the 4th International Conference on Aspect-oriented Software Development. pp. 3–14. AOSD '05, ACM, New York, NY, USA (2005)
18. Greenwood, P., Bartolomei, T., Figueiredo, E., Dosea, M., Garcia, A., Cacho, N., Sant'Anna, C., Soares, S., Borba, P., Kulesza, U., Rashid, A.: On the impact of aspectual decompositions on design stability: An empirical study. In: Proceedings of the 21st European Conference on Object-Oriented Programming. pp. 176–200. ECOOP'07, Springer-Verlag, Berlin, Heidelberg (2007)
19. Jain, A.K., et al.: Data clustering: A review. ACM Comput. Surv. 31(3), 264–323 (Sep 1999)
20. Jain, A.K., Dubes, R.C.: Algorithms for Clustering Data. Prentice-Hall, Inc., Upper Saddle River, NJ, USA (1988)
21. Kamiya, T., et al.: Ccfinder: a multilinguistic token-based code clone detection system for large scale source code. IEEE Trans. on Softw. Eng. 28(7), 654–670 (Jul 2002)
22. Kellens, A., et al.: Trans. on aspect-oriented softw. develop. iv. chap. A Survey of Automated Code-level Aspect Mining Techniques, pp. 143–162. Springer-Verlag, Berlin, Heidelberg (2007)
23. Krinke, J.: Mining control flow graphs for crosscutting concerns. In: Proc. 13th Work. Conf. on Reverse Eng. pp. 334–342 (Oct 2006)
24. Krinke, J.: Mining execution relations for crosscutting concerns. Softw., IET 2(2), 65–78 (April 2008)
25. Maisikeli, S., Mitropoulos, F.: Aspect mining using self-organizing maps with method level dynamic software metrics as input vectors. In: Int. Conf. on Soft. Tech. and Eng. vol. 1, pp. V1–212–V1–217 (Oct 2010)
26. Marin, M., et al.: Identifying aspects using fan-in analysis. In: Proc. 11th Work. Conf. on Reverse Eng. pp. 132–141 (Nov 2004)
27. Marin, M., et al.: A common framework for aspect mining based on crosscutting concern sorts. In: Proc. 13th Work. Conf. on Reverse Eng.
28. Marin, M., et al.: Identifying crosscutting concerns using fan-in analysis. ACM Trans. Softw. Eng. Methodol. 17(1), 3:1–3:37 (Dec 2007)
29. McFadden, R., Mitropoulos, F.: Aspect mining using model-based clustering. In: Southeastcon, 2012 Proc. IEEE. pp. 1–8 (March 2012)
30. McFadden, R., Mitropoulos, F.: Survey and analysis of quality measures used in aspect mining. In: Southeastcon, 2013 Proc. IEEE. pp. 1–8 (April 2013)
31. Serban, G., Cojocar, G.S.: A new hierarchical agglomerative clustering algorithm in aspect mining. In: 3rd Balkan Conf. in Informatics. pp. 143–152 (Sept 2007)
32. Serban, G., Moldovan, G.: A new k-means based clustering algorithm in aspect mining. In: Proc. 8th Int. Symp. on Symbolic and Numeric Algorithms for Sci. Compu. pp. 69–74 (Sept 2006)
33. Zhang, C., Jacobsen, H.A.: Mining crosscutting concerns through random walks. Softw. Eng., IEEE Trans. on 38(5), 1123–1137 (Sept 2012)
34. Zhang, C., Jacobsen, H.A.: Efficiently mining crosscutting concerns through random walks. In: Proc. 6th Int. Conf. on Aspect-oriented Softw. Develop. pp. 226–238. AOSD '07, ACM, New York, NY, USA (2007)
35. Zhang, D., et al.: Automated aspect recommendation through clustering-based fan-in analysis. In: 23rd IEEE/ACM Int. Conf. on Automated Softw. Eng. pp. 278–287 (Sept 2008)