RESEARCH ARTICLE OPEN ACCESS

Comparing Different Approaches for Identifying Crosscutting Concerns

Sonali R. Idate[1], Rishabh Gupta[2], Himanshu Gupta[3], Snehal D. Chaudhary[4], Priyanka S. Paygude[5]

1 (Associate Professor , Department of Information Technology, Bharati Vidyapeeth Deemed to be University College of Engineering, Pune ,India

2 (Student, Department of Information Technology, Bharati Vidyapeeth Deemed to be University College of Engineering, Pune

3 (Student, Department of Information Technology, Bharati Vidyapeeth Deemed to be University College of Engineering, Pune

4 (Assistant Professor , Department of Information Technology, Bharati Vidyapeeth Deemed to be University College of Engineering, Pune

5 (Assistant Professor , Department of Information Technology, Bharati Vidyapeeth Deemed to be University College of Engineering, Pune

Abstract:

Large-scale software applications are complex systems that involve myriad of different concerns .Ideally these concerns should be organized into different modules , but often some of these concerns crosscut each other .Identification of cross cutting concern in the beginning of each phase in software development lifecycle is very important . Crosscutting concerns identification has been neglected in software projects . This paper compares approaches such as Grounded Theory for Crosscutting Concerns Identification (GT4CCI), Discovering Early Aspects through Goals Interactions and Language Extended Lexicon (LEL) which can be a building block to identification of Crosscutting Concerns . Comparison is made on the basis of their type of requirement document which they accept and the end results .

Keywords ***: Aspect Oriented Programming , Crosscutting concerns , GT4CCI , GEA , LEL***

I. INTRODUCTION

Multiple programming problems have been encountered for which neither procedural nor object-oriented programming techniques are sufficient to clearly capture some of the important design decisions the program must implement. This forces the implementation of those design decisions to be scattered throughout the code, resulting in “tangled” code that is excessively difficult to develop and maintain. They have been hard to capture is that they cross-cut the system's basic functionality. Thus a new programming technique, called aspect-oriented programming[1] makes it possible to clearly express programs involving such aspects, including appropriate isolation, composition and reuse of the aspect code.

Different analysis strategies have been used in the earliest phase in software lifecycle when there is need to understand phenomena, and to think about problems, and construct mechanisms, and to describe solutions, and to communicate with each other. This paper compares three analysis techniques . The first one is GT4CCI [2] which treats each code separately, keeping in mind the end goal to encourage the comprehension and representation of connections built up with them , Goal-Driven Approach to Discovering Early Aspects[3] is a stretched out way to deal with perspective disclosure proposed by Jonathan Lee and LEL[4] which attempt to depict the importance of words and expressions particular to a given application area. Symbols are characterized through two properties which is examined in this paper . This paper presents the concept of crosscutting concern (section II) , briefly introduces various approaches such as GT4CCI (section III) , GDUC (section IV) , LEL(section V) , then compare these approaches (section VI , VII) and finally discusses the related work (section VIII) .

II. CROSSCUTTING CONCERN

CCC[5] is a type of concern that is entangled with other concerns. As the source code level, CCCs are often scattered over modules in a program. For instance, logging to carry out debugging is one CCC. As shown in the top of Figure 1, it is hard to

modularize CCCs by OOP. A logging code can be composed in numerous modules in a program in OOP. Such a circumstance compounds viability, in light of the fact that scattered logging codes make programs indistinguishable, and causing errors in modifying or deleting logging codes.

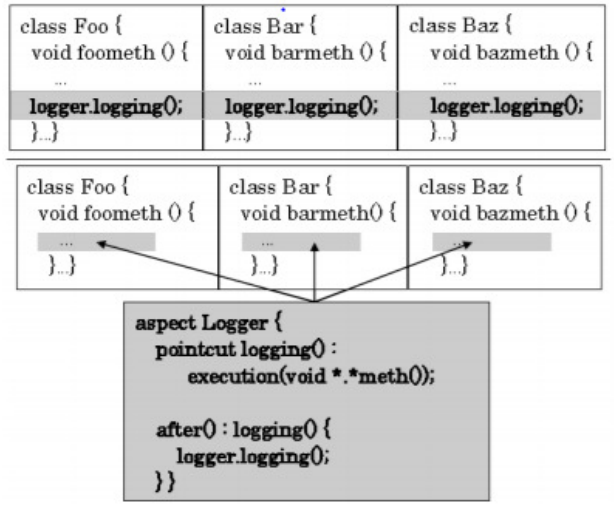

Fig. 1[5] Cross-cutting concern of logging code in OOP (top) and modularization of CCC of logging code in AOP (bottom)

III. GT4CCI

GT4CCI treats each code exclusively, remembering the ultimate objective to empower the comprehension and portrayal of relationships set up with them. GT4CCI approach takes requirement document as contribution for the investigation of crosscutting concern .The procedure presented by GT4CCI , contains five phases:

A. Open coding
B. Axial coding
C. Selective coding
D. Graph Analysis 4CCI
E. Results Table Creation

A. Open coding

It takes requirements document as input. In Open Coding every one of the prerequisites and different informations are examined and codes are made for each of these . Open coding is accomplished by segmenting data into meaningful expressions and portraying them in single words or short arrangement of words. Further, significant annotations and concepts are then appended to these expressions .

Basically , data is read several times and then tentative labels are created for chunks of data that summarize what we see happening .

B. Axial coding

It takes coded requirements document as input . Axial coding comprises of recognizing connections among the open codes and the connections among the codes . The relations are set up through connectors . Every connector distinguishes the sort of relations among each code. In GT4CCI basically two connectors are utilized :
1. is part of : demonstrates that the code is tangled inside the other code .
2. is in : demonstrates that the code is scattered .

C. Selective coding

It characterizes the connections among the codes .In like manner the entire coding process is refined . This refinement contains in analyzing the whole documents and codes and after that set the core category . Through this graphs are generated.
The diagrams produced in this progression are penniless down in detail in the accompanying stage of the GT4CCI: Graph Analysis 4CCI.

D. Graph Analysis 4CCI

In Graph Anlaysis 4CCI , as the name suggests , graph is analyzed which is generated to the core categories . Here we determine whether the core category may or may not be said a crosscutting concern by analysing relationships between the codes .Deterrmining Scattering and Tangling can be helpful in identifying crosscutting concern of a requirement document . A concern is considered scattered when its specification is fundamentally scattered between numerous others concerns

(regardless of requirements, functionalities,use cases, etc.) of the similar document. This scattering is represented by no less than two relations 'is in' between the core category and other codes .When the specification of a concern is interleaved with the specification of other concern then it is considered tangled. This tangling is represented by no less than two relations 'is part of" between the core category and different codes .

E. Results Table Creation

It is the final step proposed by GT4CCI . This table holds four attributes, that is, concern , scattered , tangled and crosscutting concern . The explanation behind this Results Table is to document, objectively, all information coming to fruition in view of usage of the approach in a requirements document .

Advantages

• Through GT4CCI approach it became possible through qualitative analysis of data to identify crosscutting concern .
• By analyzing all the data in the document , it beacame possible to identify crosscutting concern using GT4CCI.
• As GT4CCI is based on data present in the document analyzed , therefore the results obtained can be more easily justified and traced.

IV. DISCOVERING EARLY ASPECTS THROUGH GOALS INTERACTIONS

Jonathan Lee [6] introduced goal-driven use cases (GDUC) model, in which use cases are derived based on the analysis of goals interactions.The proposed approach [4] is an extension where early aspects were discovered to address crosscutting properties in the early stage of software development Analyzing early aspects improves early stage decision-making, and helps trace stakeholder interests throughout the software development life cycle. Lee constructed a Goal-driven use cases (GDUC) model of a Meeting Scheduler System proceeding with the application of three step process as summarized below:

1. Evaluate the relationships among goals and use cases.
2. Obtain goals relationships, including similarity and interaction relationships.
3. Establish goal clusters with the bidding process.

A. Step 1

The relationship between goals and use cases is represented by relational database where domain experts analyze the model to provide score rated from -5 to +5 based on how a particular goal can be achieved or affected by the given use case. Score 5 means the goal can be fully satisfied by the use case, -5 means the goal is fully denied by the use case, and 0 means the use case does not have any impact on the goal. Therefore degree of achievement for each goal is determined.

B. Step 2

The goal relationship can be determined by the help of two factors, similarity degree and the interaction degree. The Similarity degree is used for grouping goals into goal cluster whereas interaction degree evaluates the validity of goals in a goal cluster.

C. Step 3

Bidding process is used to form goal clusters. The goals are bid into clusters and each bid is therefore validated by evaluating total interaction degree followed by checking of scattering and tangling degrees. Total interaction degrees, scattering degrees and tangling degrees combines together to form a stability function. Stability function validates a goal cluster having largest bid to a goal.
Total interaction degrees defined as a summation of interaction degrees. Scattering degree defined as the number of goals in a goal cluster and tangling degree defined as the number of goal clusters that a goal participate. The bid is granted to a goal cluster if the inclusion of a goal to the goal cluster increases the value of the total interaction degree of all goal clusters. If the value of total interaction degree of all goal clusters remains unchanged after the bidding,

the stability is maintained by evaluating scattering degree and tangling degree.

GEA

Goal-Driven Approach to Discovering Early Aspects [7] is an extended approach to aspect discovery proposed by Jonathan Lee. Formulation, construction, classification, and identification are four important phases in GEA process. It consists of two additional features:

- An extended goal structure is used to represent goals that are traceable to user requirements such as functional or non-functional, rigid or soft, and actor-specific or system-specific. The relationship between goals and use-cases can be evaluated numerically.

- To improve clustering process, an ArgoUML-based tool integrated with MapReduce and HBase was developed. MapReduce allows parallelism in the grouping procedures whereas intermediate data and final results are stored and managed by HBase.

Advantages

- The relationships among goals can be more easily processed computationally.

- The numerical representation relationships among goals are more informative and can be used as a basis for discovering early aspects by exploring the existence of common properties shared by goals.

V. LANGUAGE EXTENDED LEXICON

LEL takes a simple idea as a starting point, describing the language of the application domain before describing the application. Although LEL can be built from requirements or source code, we do recommend building it directly from the application language, so as to apply the approach early on and make good use of the available knowledge, avoiding reworking of items .

Basically LEL try to describe the meaning of words and phrases specific to a given application domain. Symbols are defined through two attributes.

Notion and behavioral responses. Notion describes the symbol denotation. Behavioral responses describe connotation.

Every symbol of LEL has a place with one of four classes :

- Subject
- Object
- Verb
- State

***Example*:** banking application

Identification of crosscutting concerns using LEL

This approach depends on the LEL in order to identify crosscutting concerns. The LEL must be constructed normally; once it is built, the requirement engineer must make groups of symbols according to the state identified in LEL. Then, the references from the behavioral responses must be counted. After that, the strategy ranks groups ordering by probability, from the most probable to the least probable

Crosscutting concerns candidates. This strategy only ranks the concerns according to their possibility to be considered crosscutting and the strategy does not determine a limit above which all concerns are crosscutting.

The detailed steps to the approach are as follows

- Construction of LEL organized into groups.
- Reference counting.
- Ranking of groups.
- Final analysis

Advantages Of LEL:

- Identification of crosscutting concerns is easy through Language Extended Lexicon.
- LEL requires less time to identify crosscutting concerns than other techniques.

VI. COMPARISON OF RESULTS

• In case of GT4CCI , identification of result is done by contextual analysis by taking any of the requirement document . Recall and Precision[8] are two metrics that are used to formulate the correctness of the obtained results. By implementing GT4CCI, the results can be traced and easily justified by analysing data present in the document. By implementing GT4CCI, the results can be traced and easily justified by analysing data present in the document.

• In the case of Goal-driven Early Aspect (GEA), early aspects are found based on the clustering of goals, that is, certain goals are affected by a same early aspect, which is denoted as:
Aspect:(UseCase(i),[Goal(a),Goal(b)……])

Where goals a and b are affected by use casei.
• In case of LEL , the results will be abstract, that is, coarse-grained. Late steps in software development such as codification make conceivable to apply techniques which identify crosscutting concerns more precisely, but early stages have high-level abstraction artifacts, so the analysis cannot produce a result different from a coarse-grained one.
.

VII. COMPARISON OF ARTIFACTS ANALYZED

TABLE I

SR NO.	Approach	Artifacts Analyzed
1	GT4CCI	any requirement document
2	GEA	GDUC Model
3	LEL	any requirement document or source code

In the above table comparison is made on the basis of what artifacts is analyzed to proceed the identification . As it can be seen that GT4CCI approach requires any requirement document , whereas , GEA requires GDUC Model and LEL requires any requirement document or source code .

VIII. RELATED WORK

Non-functional properties such as security and availability have been addressed by various requirement engineering approaches.Various aspect-oriented requirements engineering (AORE) approaches have been proposed to counter both functional and non-functional crosscutting concerns.These methodologies are compared with AOREby RuzannaChitchyan, Awais Rashid, Peter Sawyer [9] , looking at how they address both functional and non-functional concerns for engineering requirements . Two groups of approaches are considered for the comparison : The first group includes PREview[10], Non-Functional Requirements Framework [11], and Problem Frames [12]. The approaches in this group have been chosen to represent contemporary RE work which recognises presence of non-functional concerns. The secondgroup includes Arcade [13] and Theme/Doc [14]. This group presents examples of work on Aspect-Oriented (AO) RE. This comparison is used to derive a set of key challenges pertaining to the handling of crosscutting requirements to be addressed by AORE.

IX. CONCLUSION

This paper briefly presented the comparison of three approaches for identifying crosscutting concern . These approaches provide their own views of the requirements document, representing vulnerabilities and issues that should be better understood and analyzed. This paper compares correctness of results and the artifactsanalyzed by each approach. Using the approach GT4CCI one can identify crosscutting concern through qualitative analysis based on context where the concern is provided in the requirement document. This paper also discussed

goal-driven approach to the discovery of early aspects through goal clustering by means of a bidding process as an attempt towards the analysis of software system . LEL is used to synthesize the knowledge of application language . It organizes the knowledge into symbols which have connections to other symbols , and these connections build a hypertext .This paper compares the good correctness of results and the artifactsanalyzed by each approaches .

X. REFERENCES

[1] Aspect-Oriented Programming .Gregor Kiczales, John Lamping, Anurag Mendhekar, Chris Maeda, Cristina Lopes, Jean-Marc Loingtier and John Irwin.

[2] GT4CCI: An Approach Based on Grounded Theory for Crosscutting Concerns Identification in Requirements Documents . Larissa de Alencar Sobral, Lyrene Fernandes da Silva.

[3] Deriving requirements specifications from the application domain language captured by Language Extended Lexicon .Leandro Antonelli, Gustavo Rossi,Julio Cesar Sampaio do Prado Leite, Alejandro Oliveros.

[4] Jonathan Lee, Discovering Early Aspects through Goals Interactions. 19th Asia-Pacific Software Engineering Conference, 2012.

[5] Comparative Evaluation of Programming Paradigms: Separation of Concerns with Object-, Aspect-, and Context-Oriented Programming .Fumiya Kato, Kazunori Sakamoto, Hironori Washizaki, and Yoshiaki Fukazawa.

[6] W. Lee, W. Deng, J. Lee, and S. Lee, “Change impact analysis with a goal-driven traceability-based approach,” International Journal of Intelligent Systems, vol. 25, pp. 878–908, August 2010.

[7] Jonathan Lee, GEA: A Goal-Driven Approach to Discovering Early Aspects.IEEE transactions on software engineering, vol. 40, no. 6, june 2014.

[8] Anacleto, A.C.S. Aplicação de Técnicas de Data Mining em Extracção de Elementos de Documentos Comerciais. Tese de Mestrado. Universidade do Porto. Porto, Portugal. 2009.

[9] Comparing Requirements Engineering Approaches for Handling Crosscutting Concerns. Ruzanna Chitchyan, Awais Rashid, Peter Sawyer.

[10] I. Sommerville and P. Sawyer, "PREview Viewpoints for Process and Requirements Analysis," Lancaster University, Lancaster REAIMS/WP5.1/LU060, 29 May 1996.

[11] L. Chung, B. A. Nixon, E. Yu, and J. Mylopoulos, Non-Funcitonal Requirements in Software Engineering: Kluwer Academic Publishers, 2000.

[12] M. Jackson, Problem Frames: Analyzing and Structuring Software Development Problems: ACM Press, 2001.

[13] A. Rashid, A. Moreira, and J. Araujo, "Modularisation and Composition of Aspectual Requirements," presented at 2nd International Conference on Aspect Oriented Software Development (AOSD), Boston, USA, 2003.

[14] E. Baniassad and S. Clarke, "Theme: An Approach for Aspect-Oriented Analysis and Design," presented at International Conference on Software Engineering, 2004.