International Journal of u- and e- Service, Science and Technology
Vol.8, No.2 (2015), pp.243-250
http://dx.doi.org/10.14257/ijunesst.2015.8.2.24

Research on Detection of Business Constraints based on AOP

Ning Chen

College of computer science, Xi'an Polytechnic University, Xi'an ShaanXi, China
chennvictor@gmail.com

Abstract

Aiming at the deficiency of magnitude of business constraints and high complexity, a constraints detection method is proposed. The paper begins with discussion on Aspect-Oriented programming (AOP), including its concept, thought, development, and advantage in handling crosscutting concerns, followed by description of the theory of constraints. Furthermore, combined AOP with first-order logic language, a schema on detection of the business constraints in requirement analysis is presented, which is simple and practicable, thus making optimal designs for improving software robustness. Finally, application of the method on management system of hotel is given, with satisfactory results.

Keywords: *Business Constraints, AOP, First-order logic language, Theory of Constraints*

1. Introduction

Nowadays, in software engineering, the constraint is inevitable. If there are no constraints, then the output of the system will be infinite, but in reality, any system output cannot be unlimited. Therefore, in any system, there are one or more constraints. In the software development process, the complex business constraints are formed by the complex relationships between services. In the discovery, description and modification of business constraints, the different business analysts may cause unusual problems such as redundancy and conflicts, these issues will make high costs for development and maintenance of software system, and even lead to software systems have some undesirable behavior. Small number of simple constraints can be achieved by manual testing, but due to the gradual increase in the number and complexity of constraints, manual detection method would be difficult to achieve. Therefore, the constraints detection allows of no delay, and we must propose a way to detect it, improve the reliability of the software radically.

Software engineering process is a chain of activities which can develop or maintain software and related products. In software engineering, conflict is inevitable. Conflict can exist in any portion of the software, while primary consideration of most researchers is the conflict within single software. Conflict within single software can make individual software not work. So finding conflict within software promptly and solving the problem promptly can improve software robustness.

The result of requirement analysis is an important basis for developing software systems. A large number of statistics indicate that 15% of software errors in the software systems arise from the error in requirement analysis. In order to improve software quality, ensure the success of software development, and reduce software development costs, a set of business constraints of the target system must be proposed, and requirement analysis must make sure that these business constraints should be strict correct.

In practice, if researcher does not optimize the requirement analysis, and does not detect the redundancy and conflict among components, a large amount of bug code and duplication code will be produced, therefore software reuse will fail. Nowadays, Aspect-

Oriented Programming (AOP) can be used to define crosscutting concerns, to reduce duplication of code in the system, effectively to reduce the coupling between modules. And AOP can detect constraints by using the first-order language, thus greatly improving the efficiency of software development. For the robustness, redundancy and conflict management is a necessary support facility [1].

By analyzing the idea and principles of AOP, the paper proposes a schema on detection of business constraints based on AOP. The schema can optimize requirement analysis in the software development process, and detect constraint redundancy and conflicts. Business constraints are extracted from requirement analysis and are defined as a crosscutting concern. Then the redundancy and conflict in the crosscutting concern are detected by use of first-order logic language, thus improving the overall efficiency of the software development.

2. AOP-based Detection of Business Constraints

2.1. Constraint Detection

Nowadays, with the rapid development of the computer, computer is widely used in all social business. Application software has been expanding and its complexity is increasing. Therefore, software reuse technology is paid more attention [2]. In the software development process, the complex relationship between the businesses can form complex business constraints. When different business analysts discover, describe and modify constraints, abnormal problems, such as redundancy and conflict, maybe are introduced. These problems will increase software development and maintenance costs, and even make the software system to produce unexpected behavior. Manual detection is easy to achieve small number of simple constraints. But with the number of constraints and the complexity increases, it has become powerless [3].

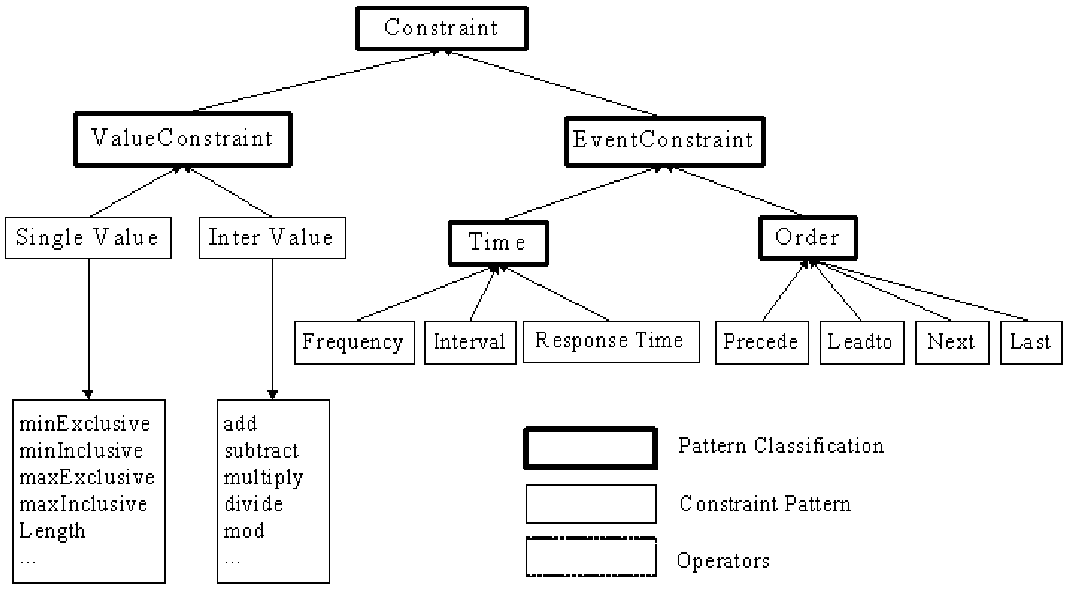

Figure 1. Example of Constraints

Theory of Constraints (TOC), TOC was put forward by the Israeli physicist Dr Eli Goldratt in the late 20th century. It is a management theory on how to better improve on the implementation of these business improvements. Its object is to enable enterprises to improve performance significantly, achieve greater gains in the short term. TOC is to help companies identify what are constraints in the process for achieving the objectives, and further pointed out that how to implement the necessary improvements to eliminate these constraints one by one, to more effectively achieve business goals. This paper draws on the ideas that use theory of constraints to discover business constraints, and applys it in the software development process, as shown in Figure 1.

Current constraints detection methods mainly consist of formal methods and software testing. The most representative are model checking, theorem proving, and software testing techniques. Model checking and theorem proving are based on formal mathematical methods, that is, using mathematical logic operator to determine the final behavior of the program meets business constraints of software system, can verify if software system is really no error from one point of view, and if business constraints of software system is ambiguity, inconsistency and incompleteness. As users demand more and more, the programming language has become increasingly complex, so, program design has become increasingly complex and complicated, growth of complexity at an exponential rate with the growth of the program. In order to verify the accuracy of the program, we need to cover all possible situations, which are a gigantic project, so the model checking and theorem proving are called heavyweight formal methods. For now, theorem proving and model checking only be used to prove the validity of some of the key core module, which has not been more widely used. Theorem proving and model checking can indeed protect the quality of the software, but they require personnel of experience and competence, and take a lot of time, therefore it is difficult to improve software productivity. While software testing is the process after a part of program is complete, in order to detect whether there is an error in the program to execute the program. Nowadays, the vast majority of software testing is manual testing. Although it improve the quality of the software system to a certain extent, with definition of a variety of test adequacy, but compared with the formal model checking method, it is still not sure software system will certainly be no errors after software testing, and we ultimately cannot completely guarantee program is free of error, finally there is no guarantee reliability of business constraints fundamentally.

Thus, these methods are either difficult to achieve in practice, or cannot guarantee the reliability of business constraints in software systems fundamentally. Therefore, we need to combine the advantages of validation and testing, to propose a new method to improve software reliability, namely integration of formal model with clarity and software testing with ease of use. Nowadays, AOP is a new approach for software development, by means of which, we can combine formal methods and software testing to monitor operation of software system, so that the operation of the system conform to business constraints. Business constraints is generated from the user's demand, thus ensuring the states and behaviors of the software system will conform to its business constraints while the operation of the software, ultimately improving the reliability of the software system.

Constraint detection should be defined while setting sufficient constraints before the software deployment. So, in order to implement effective constraint detection, we must rely on the basis of the implementation of the business logic of the system, thus achieving the management of the redundancy of resources and conflict. But these constraints as well as business logic, scattering, permeating are throughout the system [4]. The emergence of AOP is an effective solution to solve the assembly constraint detection and rational business logic problems. In the 1997 European Conference Object-Oriented Programming (ECOOP97), Gregor Kiezales who is the chief scientist in Xerox Palo Alto Research Center and the professor of University of British Columbia first proposed the concept of AOP. AOP separate the main concern and crosscutting concern, weakening the coupling factor between the two concerns. AOP technology can encapsulate related crosscutting concerns, and ensure crosscutting concerns reusing. According the requirement, adding, removing, or changing the crosscutting concerns without affecting the main concerns, can improve system scalability and flexibility [5]. It can be used to solve the concerns of the code tangling problem which can provide an effective way to describe the crosscutting concerns for developers. Each year thereafter on ECOOP it always has the AOP-related workshops. Major corporations, universities, research institutions have invested in its research.

2.2. Investigation

Since the theory of constraints has been proposed, with the development of 20 years, TOC has formed a complete set of management ideas and methods, which are widely used in modern business management, including Operations Management, Finance, Distribution, Project Management, Marketing, Sales and so on. In the short term, TOC has make business performance improved significantly, and brought enormous benefits for the enterprise. TOC research and application in China has gradually been paid attention. Academic research on the theory of constraints mainly focuses on product combination, production planning, project management, maintenance of equipment, management [6].

Software requirement is divided into functional requirement in software systems and binding requirement in software systems, which shows the importance of constraint. At runtime, we extract constraints from software requirement and set monitor code relating to constraints in interceptor from the different procedures. Thus we use one or more independent interceptor to detect potential conflicts. An interceptor can intercept request message from the client to the server, and intercept response message from the server to the client, and assess that constraint is the conflicting or not. These interceptors form function modules of the software as a flexible interface, similar to the bones of a joint. However, in the past few years, problems relating to constraints are not received enough attention. Functional requirements are modeled and are divided into different design and implementation modules, but constraints are often hidden in the implementation of functional modules.

As for AOP, the last decade, it in most cases is silent. However, in recent years, with the rapid development and the application of computer technology, the overall design and centralized control of the traditional software development methods has increasingly shown their inherent limitations. Many people began to talk about the AOP, then more and more the technical staff came to realize the great advantages of AOP and its potential strength [7]. The AOP can be effectively developed, and help software developers to write the easily understood code and reduce the complexity of software development. With determining the scope of the revisable code, the AOP can improve the quality of the code. Currently, many foreign universities and research institutions have invested to study AOP and achieved certain results. Many large companies have applied this new technology. In addition to software, AOP is applied in the database, operating system, middleware, networks, compilers and other new advances direction [8]. But the domestic aspect-oriented programming technology is still in the initial stage. It can only be used in very few languages, and have the source code to be woven into. But Ivar Jacobson, the RUP (Rational Unified Process) father, UML founder as the representatives of the scientists, appreciate the AOP highly. Dr. Ivar Jacobson is also engaged in the AOP research, and he believes AOP will ultimately change the software development methods [9].

2.3. The Related Theory of Constraint Detection

2.3.1. The Concept and Advantages of AOP

AOP is more popular in recent years, and is one of hot topics in computer technology. AOP, based on object-oriented programming method, is developed as an innovative approach for software development. Object-oriented Programming uses class to encapsulate data and behavior, to package system function, each packaged function as an object [10]. The main concept of object-oriented is vertical structure. Its purpose is to obtain a more clear and efficient logic unit. But AOP is used to extract section from business process. Its main concept is to divide application into business logic processing part and universal service support for business logic processing, which is “crosscutting concern”. The "crosscutting concern" throughout the program needs multiple vertical modules [11]. So, ones can achieve crosscutting module, then join the crosscutting module and the main module together by weaving them by the mechanism from AOP. Finally, ones can construct the final actual system. AOP makes crosscutting concerns as modules with a clear border, so as to produce the system architecture which is easy to design, implement and maintain [12].

AOP introduces an abstract concern. It has some advantages that do not exist in OOP. First, some advantages are the code concentrated, easily to understand, resolving the code tangling and code scattering caused by the OOP cross-module. Second, modular crosscutting concerns ultimately carry out the system code, therefore redundancy is small, easy to understand and maintain. Third, the system is easy to expand. Due to aspects of the module does not know the existence of crosscutting concerns, it is easy to add new features by creating a new dimension to make the system easy to expand. Fourth, the code is reusable. AOP achieves every aspect as independent module. Between the modules are loosely coupled, making the code better reusability.

2.3.2 First-order Language

First-order logic language is used to detect the conflict. First-order language has more quick processing speed than other languages. First-order language is used in the form of first order logic language. Here, what is the first-order language is simply introduced. Under the general sense, the first-order language is composed by the eight categories of symbols [13].

- Individual symbols: expressed by the standardized form with a lower case Latin letters, such as a b c;
- Relation symbol: such as F G H; among them, there is a special binary relation symbol, called the equal sign, recorded as ≡;
- Function symbols: such as f g h;
- Sign free variables: such as u v w;
- Constraint argument symbols: such as x y z;
- Linked symbolic:¬ ∧∨→↔;
- Quantifier symbols: ∀ (universal quantifier symbol), ∃ (existential quantifier symbol);
- Punctuation(simply called punctuation): （ ） ， namely, left parenthesis, right parenthesis and commas.

2.4. Schema on Detection of Business Constraints based on AOP

Constraints conflict detection should be defined while setting sufficient constraints before the software deployment. It is essential that detect of these conflicts in the implementation process. If the software can automatically do detect of sensitive conflict as early as possible, we can take appropriate measures, thus avoiding many potential disasters, and improving performance. As we all know, any system must meet the requirement of certain business functions, the so-called business logic. To achieve this business logic, the system must also provide the necessary support facilities, such as access control, log management.

Now we have a basic understanding of the principle and method based on the detection of business constraints on AOP. To implement this algorithm, we can follow the next three steps:

1) First of all, it demands requirement analysis. Business logic is modeled by TOC, and then business constraints are achieved by UML denotation.

2) Separation of concerns: Using the relevant technology of AOP, the business logic is defined as the core concerns, and resource management will be defined as a constraint

crosscutting concerns, thus isolating from the system cross-cutting concerns, forming a separate functional module, a separate redundancy and conflict detection.

3) Achieving concerns: The UML model of the system is the corresponding XML language. Then it converted into first-order language. So we can find the redundancy and conflicts, and then modify it, design an appropriate template library. When receiving service information, it matches with the business constraints, finds information which breached the business constraints, handles the error.

For example: first-order language assess whether there is redundancy, mainly to see whether there is inclusion relationship.

Supposed t1, t2 are the same object constraint, $\forall t1 \in D$, $t2 \in D$, , if $\forall t1 \in D \Rightarrow \forall t2 \in D$, called t1 and t2 are tolerant, redundancy, and then it need to modify.

Assess whether there is conflict constraints, mainly to see whether it is consistent. $\forall t1 \in D$, $t2 \in D$, $r \in R$, if $t1 \in D(r) \Rightarrow t2 \notin D(r)$, t1 and t2 are mutually exclusive, thus conflict existing. D(r) is constraint in the r condition.

3. Case

For example, we will design a computer-based hotel management system.

We know that, as a successful application of the computer in the hotel, software can significantly improve economic efficiency, quality of service and work efficiency of the hotel, make the hotel managements fully understand the situation to refine and improve the internal management system.

At this point, we should develop a computer-based hotel management system. However, programmers often analyze the business constraints from their own fields, thus a considerable part of the hotel management system does not achieve the desired effect. An important reason: hotel software is management software, which needs alteration along with change management of national, organization and hotel. Therefore, we should allow the secondary development of hotel, as well as other necessary software maintenance. Therefore, we should analyze business constraints from the perspective of the management, which can be easily understood and modified.

In the face of a simple system, we can easily identify the core issues. But when faced with a complex system, it is often difficult to detect a lot of problems, and in many cases, which are the interaction of many factors, the combined effect. In this case, we need to use the theory of constraints with UML denotation to achieve critical information on business constraints.

From flow chart shown in Figure 2, we can be conscious of that, every client carries out the same set of check-out process. Now some of the constraints, drawing from some repeating actions, such as registration of information (Whether ID number is true, is greater than 18 years of age, *etc.*,), are defined as crosscutting concerns of AOP, separating from the business of specific customers for constraints detection.

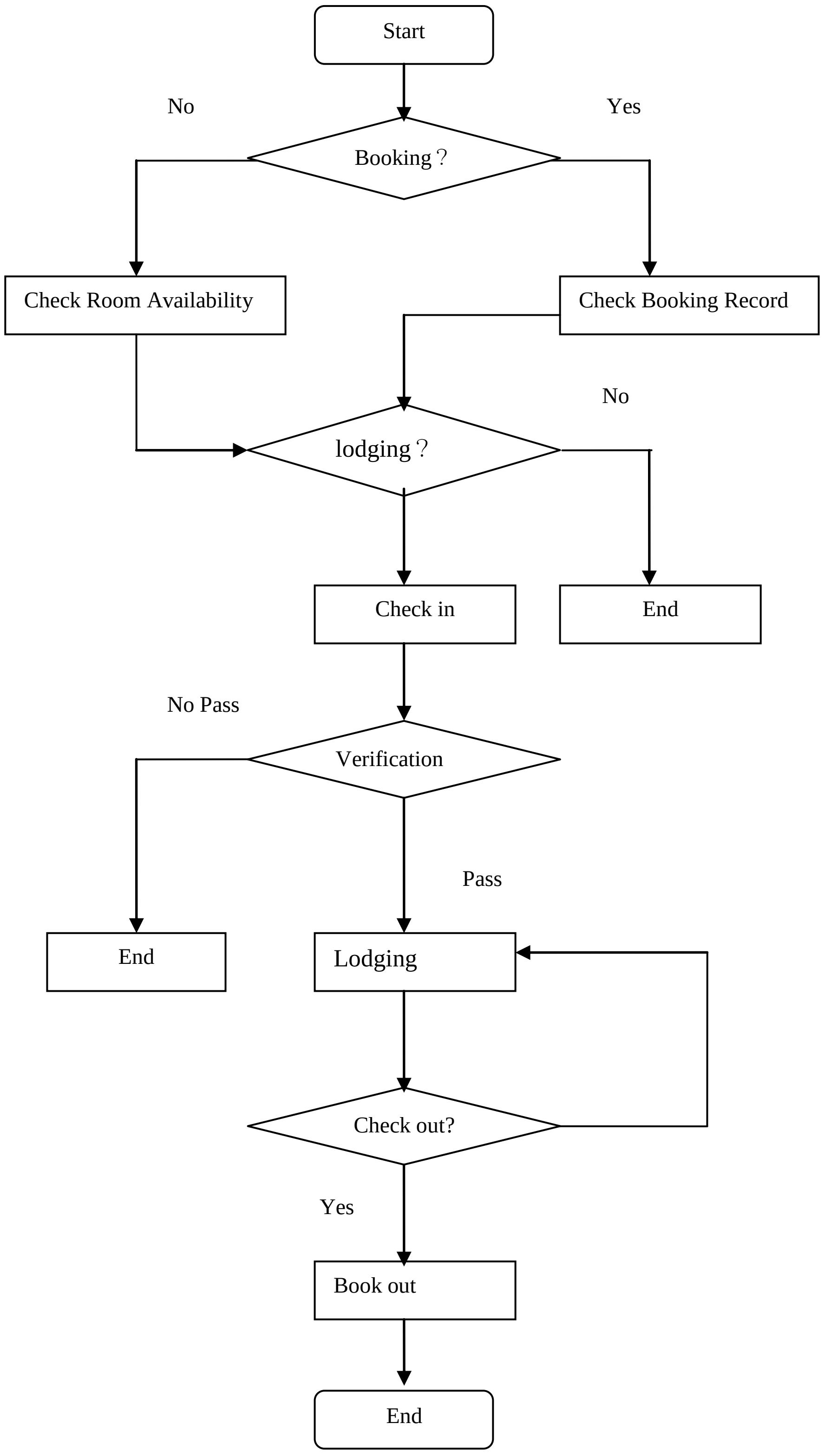

Figure 2. Flow chart

4. Conclusion

The AOP and theory of constraints, although both made earlier, but its development in China are still in its initial research stage. In fact, 'aspects' come from the concept of programming', and 'constraint' come from the requirement. Even if researchers find the relationship between the two, however, the accuracy relationship between constraints and aspects is still unclear. This paper is to explore the AOP-based detection of business constraints. It uses the combination AOP framework with first-order language to achieve business constraints redundancy and conflict detection. So as to achieve optimization in the software development process, to improve the overall efficiency of the development software. This deserves further research.

Acknowledgements

This work is supported by "Quality Project" of Degree and postgraduate education in Xi'an Polytechnic University (13YZL05). This paper is a revised and expanded version of a paper entitled "Schema design on detection of business constraints based on AOP" presented at The 6th International Conference on Multimedia, Computer Graphics and Broadcasting (MulGraB 2014), Haikou, Hainan China , December 20-23, 2014.

References

[1]. J. L. Fiadeiro and A. Lopes, "A model for dynamic reconfiguration in service-oriented architectures", Software & Systems Modeling, vol. 2, no. 12, **(2013)**.
[2]. M. L. Goc, C. Frydman and L. Torres, "Verification and validation of the SACHEM conceptual model", International Journal of Human-Computer Studies, vol. 2, no. 56, **(2002).**
[3]. A. Kumar, R. Kumar and P. S. Grover, "An Evaluation of Maintainability of Aspect-Oriented Systems: a Practical Approach", International Journal of Computer Science and Security, vol. 2, no. 1, **(2007).**
[4]. F. Munoz, B. Baudry, R. Delamare and Y. L. Traon, "Usage and testability of AOP: An empirical study of Aspect", Information and Software Technology, vol. 2, no. 55, **(2013).**
[5]. G. J. Friedman, "Constraint theory: an overview", International Journal of Systems Science, vol. 10, no. 7, **(1976).**
[6]. H. Rebêlo, R. Lima, G. T. Leavens, M. Cornélio, A. Mota and C. Oliveira, "Optimizing generated aspect-oriented assertion checking code for JML using program transformations", An empirical study, Science of Computer Programming, vol. 8, no. 78, **(2013).**
[7]. C. He and Z. Li, "Automatic Detection to the Behavioral Conflict in AOP Application Based on Design by Contract", Journal of Software, vol. 6, no. 11, **(2011).**
[8]. A. C. Neto, R. Bonifácio, M. Ribeiro, C. E. Pontual, P. Borba and F. Castor, "A design rule language for aspect-oriented programming", The Journal of Systems & Software, vol. 9, no. 86, **(2013).**
[9]. W.-Y. Liang and P. O'Grady, "Design with objects: an approach to object-oriented design", Computer-Aided Design, vol. 12, no. 30, **(1998).**
[10]. M. Marin, A. Deursen, L. Moonen and R. Rijst, "An integrated crosscutting concern migration strategy and its semi-automated application to JHotDraw", Automated Software Engineering, vol. 2, no. 16, **(2009).**
[11]. O. A. L. Lemos and P. C. Masiero, "A pointcut-based coverage analysis approach for aspect-oriented programs", Information Sciences, vol. 13, no. 181, **(2010).**
[12]. N. Kerdprasop, K. Intharachatorn and K. Kerdprasop, "Prototyping an Expert System Shell with the Logic-Based Approach", International Journal of Smart Home, vol. 4, no. 7, **(2013).**
[13]. D. K. Pradham, "Fault-tolerant computer system design", Prentice Hall, **(1996).**

Author

Ning Chen, is associated professor of college of computer science, Xi'an polytechnic university in Xi'an, China. His research interests include computer algorithms, parallel computing, intelligent information processing, *etc.*