Aspect-Oriented Software Development based Solution for Intervention Concerns Problems:Case Study

Farhad Soleimanian Gharehchopogh
Department of Computer Engineering, Science and Research Branch, Islamic Azad University, West Azerbaijan, Iran

Esmail Amini
Department of Computer Engineering, Science and Research Branch, Islamic Azad University, West Azerbaijan, Iran

Behnam Zebardast
Department of Computer Engineering, Science and Research Branch, Islamic Azad University, West Azerbaijan, Iran

ABSTRACT
All existing methods for Developing Software Systems, most insist on a separate system to keep the components together till they have been had the least overlapping. But these methods in the management system those have some parts and are using use case, and involved in the other parts of the systems, are inefficient. With arriving the Aspect-Oriented Programming, programmers were able to Implement the overcome some of these requirements and Implement them in a separate unit, but there are still some of the analysis requirements and design, because of the wrong analysis and design they cannot be implemented as a measure. In this article we want to prominent the phase of analysis and design of this work using the Aspect-Oriented Software, in order to implement them in the Implementation phase as a cup-on.

Keywords
Intervention Concerns, Software Development, Aspect-Oriented Software Development, Modularization Requirements.

1. INTRODUCTION
in modern programming techniques system developers have little attention to Modularization of the use case that are involved with the different parts in the system that we called intervention concerns in the Implementation phase are the use case which make the Complexity and Fragmentation in program, the system fails to develop. In the Implementation phase are some guidelines that can help programmers to solve some of these problems but there are some of the Concerns Intervention that should be diagnosed very early then Implementation phase and will provide a mechanism for them till can Modularization in the Implementation phase. With Aspect-Oriented Software Development, systems analysis and design phase we can by using the correct diagnosis achieve to use case Intervention Concerns a system and achieve them in the form of aspect model and System level to prevent them from being scattered and lost by a system and achieve to a Modularization System [1, 2].

In the second part of this issue we expressed concern concepts and Interventions Concerns more directly. In the third section examines and expresses the benefits and problems of the Aspect-oriented Programing and it. In the fourth section, we introduce the Development of Aspect-oriented Programing, in Section five using Aspect- oriented Software Development and The proposed new structure is based on aspects of Library Management Systems as a case study we will gain a method that is well-respected. In the sixth section presents the Conclusions and results as well as will provide the works that can be done in this area in the future.

2. Aspect-Oriented Software Development Challenges Ahead
With considering ways to develop the software that took advantage of them the methods of breaking the problem into smaller cases were used to overcome the Complexity So that these modules have to feel the same. But these methods do not have time to Implement an Embedded Software System so that problems may arise in the separation of the modules in this phase Analysis and Design of the system is wrong and you may not measure correctly separated the many problems encountered with the system development process. If these modules are attached to each other the separation them is not possible by a team of Software Development and also changing in the structure of a measure affects the whole system, that it raises the cost of System Maintenance. The Factor which Make Complexity in these systems is the Intervention Concerns.so separation the Intervention Concerns is the main problem. So that in all phases of System Development is Independent. To achieve this Goal-oriented Software Development Aspect was introduced so that the power Modularization Software used in the production cycle. This approach at first helps us diagnose use cases of the system and then based on their, we can Implement System Concerns in the case of modeling Aspects [1, 2, 7].

The purpose of the Software Engineering is to produce a system expected to realize the tasks that these tasks are well known System Requirements [7]. Within a given structure may be several requirements have the ability to aggregate and it is also depended on the users' expectations, in this case it is called Concerns. While some of these Concerns are essential to achieve system goals and System Performance is disturbed without them [4]. When systems are complex, developers need to overcome the Complexity by Putting Forwarding

some solutions, one of which is in the System Development in a component and the Requirements Encapsulates in the form object and service, but there are some requirements that have not inherently capable of being encapsulated in a component, and always are involved with various systems and Are scattered in the whole system, which ultimately led to the scattering problem and the Complexity of the system [5]. The Requirements which have such a feature called Interference Concerns. By Using Object-oriented Programming, Intervention Concerns are scattered throughout the code and prevent the achievement of this method by producing software is useless because we need to enter codes related to Concern Intervention in various parts of the System Components to reach to related Requirements of this Concerns and communicated between them [6].

When the System Developer needs to Implement a component in the system May require in one part of it to an additional requirement, which it is Implemented before. If there are codes of Implementing in several components the Requirement for Intervention is a Concern and this causes System Complexity and because the codes of this concern has spread in the Distribution System Component, say there is scattering in the system and the system will not respond to changes in it [3, 7].

3. Aspect-Oriented Programming

One of the main reasons for the Complexity and Distribution of software systems are Intervention Concerns that despite the modern techniques such as Object-oriented Programming can be seen in the code [7]. With the Aspect-oriented Programming dramatic change make in the solving problem Intervention, So that programmers use Aspect-oriented techniques, which could interfere the Concerns locally and Modularization inside aspects. With using The Aspect-oriented Programming can a piece of code Implementation Concerns related to Interference separate from other System Components and Encapsulates them in the form of one or more Aspects of the systems that reduce System Complexity and reducing code scattering is due to Interference Concerns and code rate is very much more Understandable [8].

4. Aspect-Oriented Software Development

As mentioned in Section 3, the Aspect-oriented programming would measure the Concerns of the aspects in to the aspects, But there were still some Concerns in systems that had not the feature of measuring, The reason is that these Concerns as the main part of the use case are modeling and separating them from the inside of their use case and Modularization is not possible. A use case may also act as a concern in the system so that it should be involved with several other components. So we can conclude that Intervention Concern can be considered as a use case member that to come into the system at component level is involved with several classes that cause system complications and misunderstandings of that Concern Intervention. By using Aspect-oriented software development and with a new attitude towards this issue by using a new method of Aspect-oriented Modularization Concerns for Intervention and also reach to improve the modeling of use case [6, 8]. One can use the Aspect-oriented software development identify as the Concerns Intervention before detecting the Implementation phase and wrap them in measured action That it needs-based development as a foundation member of based on our work, we get the desired result [8]. Because the use case of as a tool for software development companies is very accessible and the various concepts of it is accessible, for Aspect-oriented. Method that can be used by Using Aspect-oriented development based on use case, at first concern intervention by the use case of the system is modeling Then the use case are designed their cover based on class and Finally by using cutting aspects of the system and use case that their concerns have are modeled system [9, 10].

4.1 Modularization of Concern within Aspect

Aspect-oriented programming uses an experimental concept that called for modular aspect concerns in the code. The modular nature of the code is to will prevent from scattering Implementing codes of the system components and encapsulates them inside itself. In The Object-oriented approach to cope with complex systems, a new approach called "object" is proposed to break large systems into small systems. Which it is used in Aspect-oriented programming concept and the Aspect-oriented programming of Aspects and class structure uses and the main operation and powered can be implemented by classes so that the Intervention can be implemented by Aspect [11, 12].

5. Library Management System: A Case Study

The main objective of software development is to achieve maximum stakeholder who wants to achieve this goal, we developed a system based on use cases because these use cases come from use case needs and Concerns of the system can be distinguished by using them [10]. By using of modeling of the first stage can be achieved complexity and distribution in a system [12]. In this paper, we investigate of Library Management system as a case study and the model of system which we're using is Enterprise Architect Version 7.0. Fig 1. we can see the use case of the Library Management System with their relations.

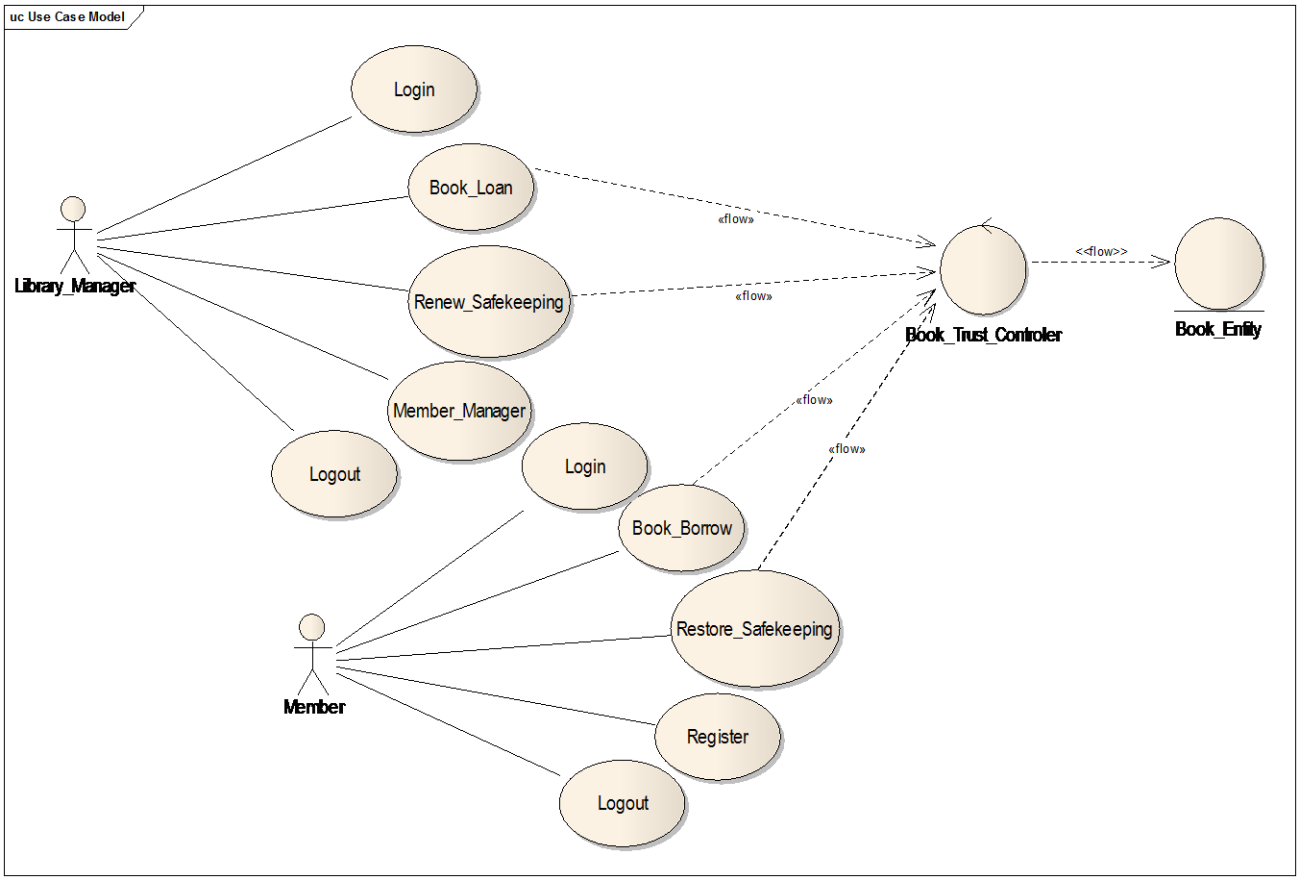

Fig 1: Diagram of the Relationship between the use cases of the Library Management System

5.1 Analysis of Use Case

After identifying the relationship between use cases should be identified concrete classes needed to realize them. Depending on your circumstances, your items are measured in one or more classes. If your case involves one or more Intervention Concerns the Implementation of the code will be broadcast in several classes [13]. With analyzing of use cases items can be configured easier and the development of parallel systems can be incredible [12]. In Figure (2) you can see use case and classes, that includes the Library Management System, and Concerns Intervention.

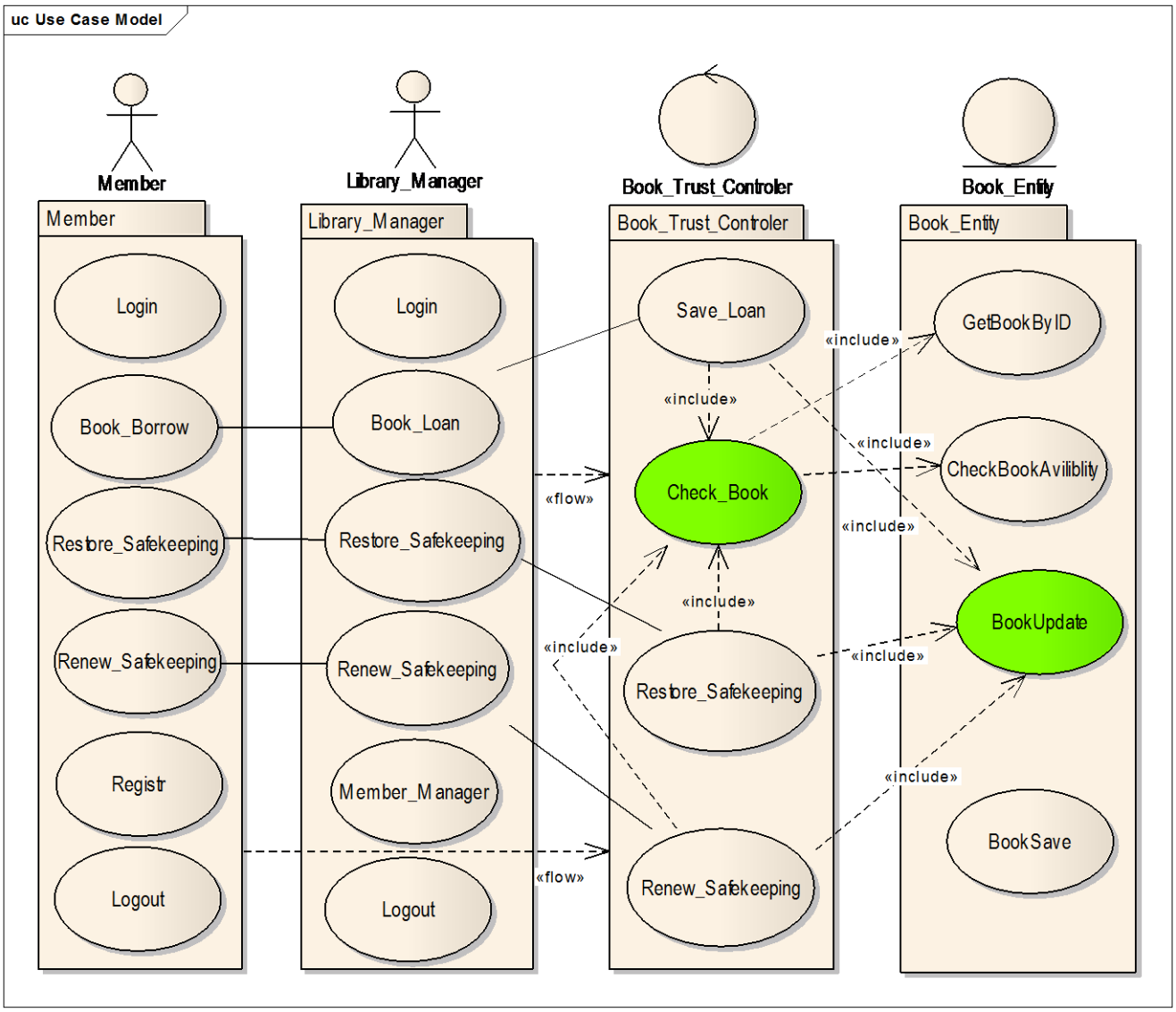

Fig 2: The Concerns Intervention of Library Management System

After identifying the classes and the use case must enter the details and internal structure of each class according to the internal structure of each class, some points that should be analyzed [12, 13, and 14]. It should be noted that at the realization of the different use cases of the class makes complexity in a class and also each section of use case does not constitute a complete class but also includes some parts of the class that will be required to fulfill use case requirements [15]. In Figure 3 you can see the cutting of the parts of the use case which are scattered in the class system.

As in Fig3. each member of use case, includes some parts of class that is required for its realization. In practice that we use a case we need to have complete characters of it. Therefore all scattering parts of a class should be combined with each other till a use case properly work [13, 14].

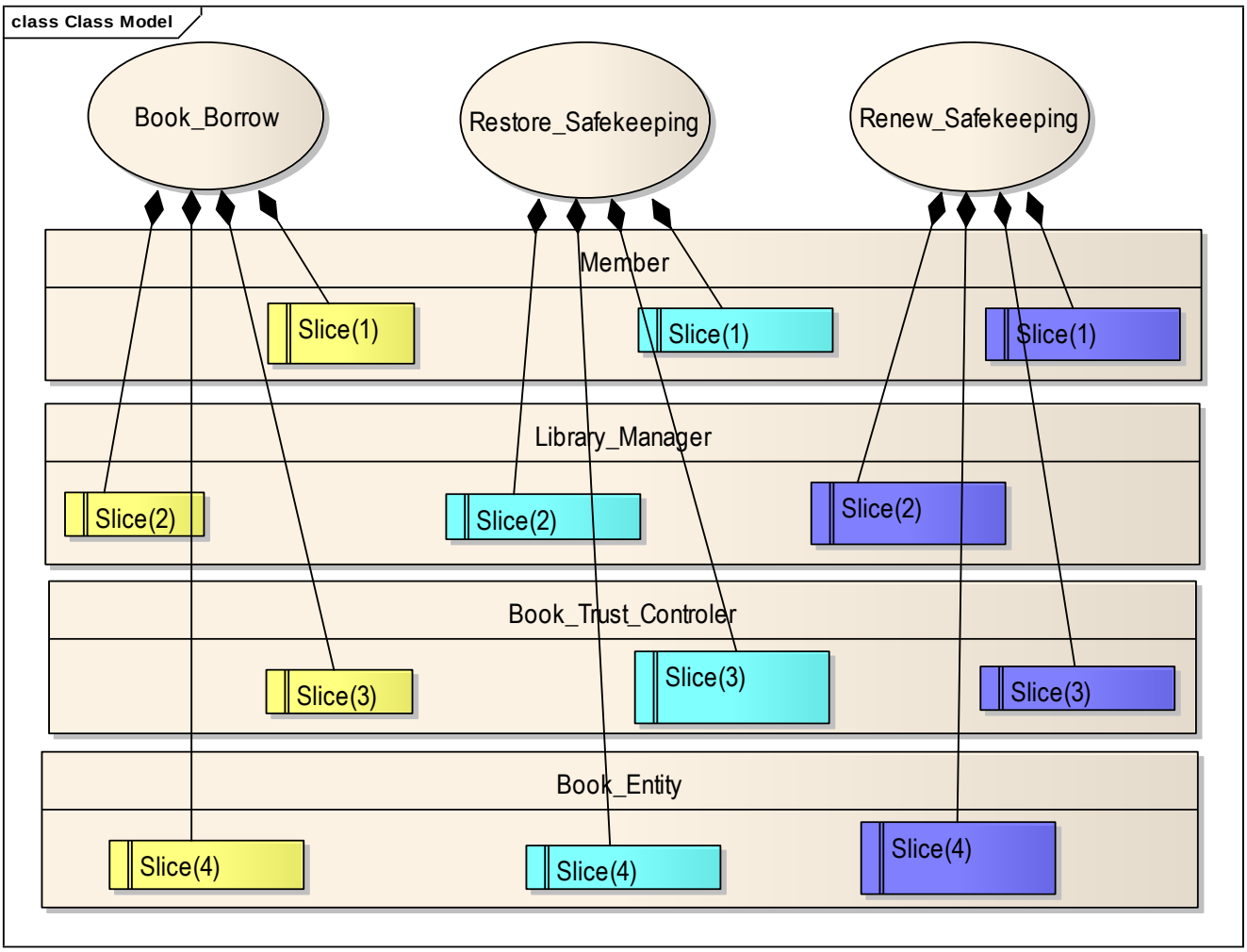

Fig 3: Distribution of use case cutting in component-level of Library Management System

5.2 The Realization of Use Cases using Aspects

The use case in the modeling work of stakeholder work effective. It should be noted that at the design stage and Implementation stage concerns showed be separated, that it a use case is possible by cutting your cases [12]. After cutting should be identified a use case, to achieve the required classes that recognize and to determine their roles [14]. The Book_Trust_Controler class library management system plays a controller interface for other classes of systems in order to achieve Book_Entity class. The Book_Trust_Controler as a coordinator and also a class that have required controls to realize the main use case of the Library Management System. The Book_Trust_Controler class role as a controller class can be seen in Fig 4.

Fig. 4. shows the sequence that the other classes of system use to achieve class Book_Entity, Book_Trust_Controler. In fact the Book_Trust_Controler class receives and performs the necessary controls for realizing the use case and then puts them through interfaces that provide Book_Entity class, will perform the update operation.

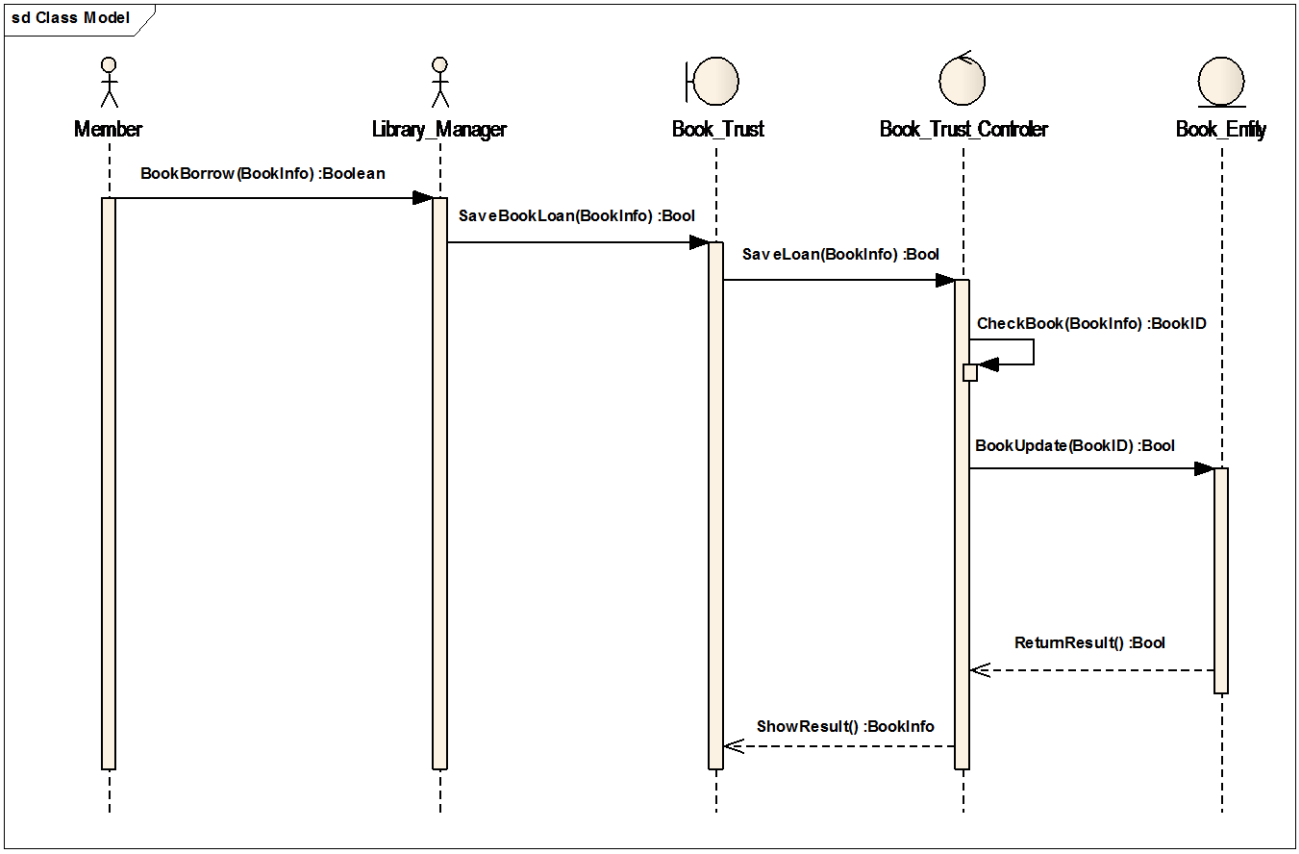

Fig 4: Sequence Diagram of Registered the Book Loan in Library Management System

5.2.1 Identify Aspects of the System

After characterizing the interaction of components in the system, Aspects that causes reducing dispersion in system must be specified, which ways of identifying them are in the form of correspondence between the classes? If there a use case in a class that is associated with several other classes, actually we face with the Concerns that their realization requires the use of other system components and at this situation to reduce the complexity of it; we need to define aspects [15, 16]. In The Library Management System according to the correspondence courses between Book_Trust_Controler and Book_Entity, we need to define Aspects to reduce the complexity of our system. Figure (5) shows Aspects that are needed to reduce the complexity and reality of them.

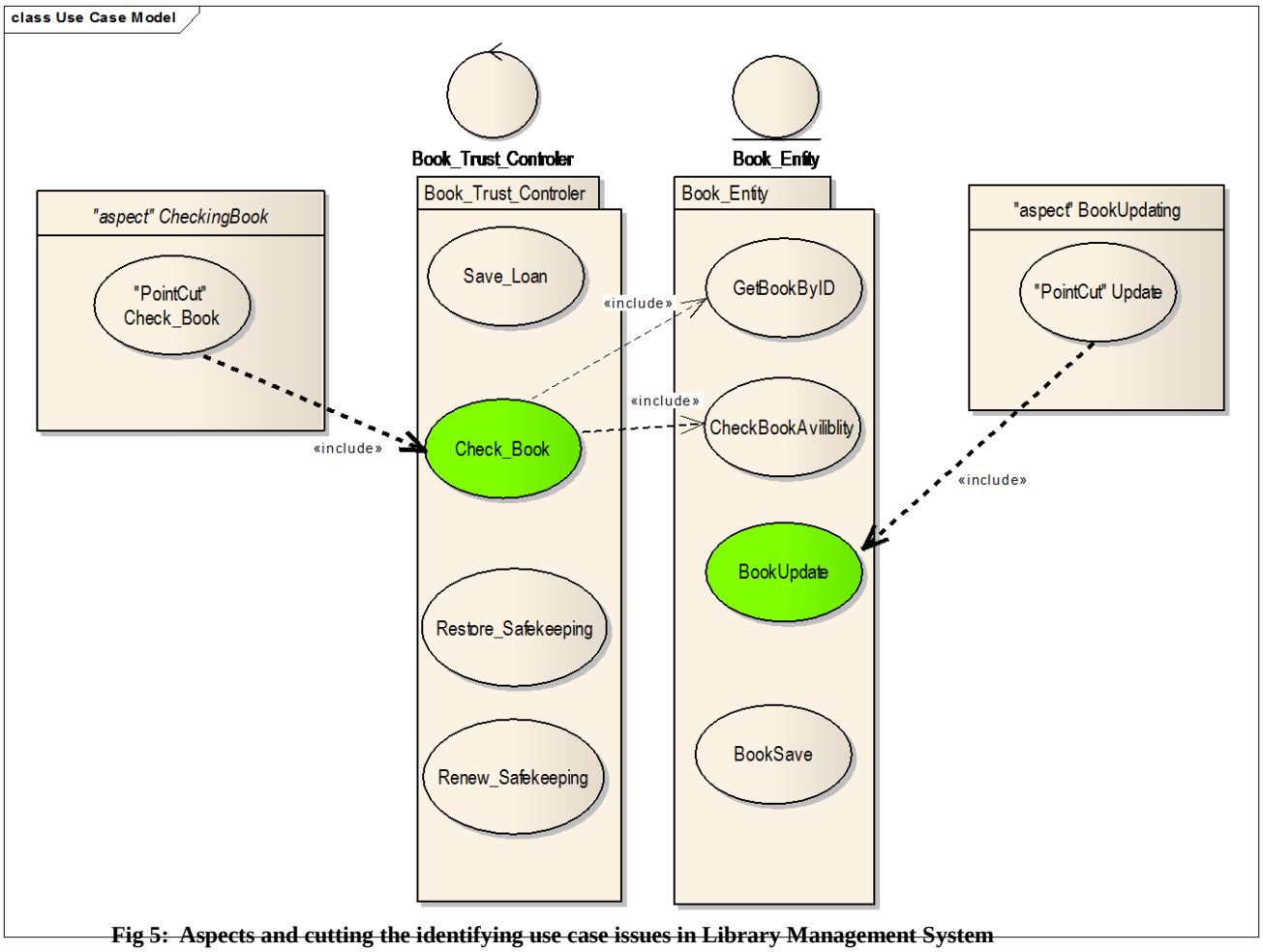

Fig 5: Aspects and cutting the identifying use case issues in Library Management System

As you can see in Fig 5. every correspondence aspect cover a separate use case in the system in such a way that minimized the scattering code. Of Course, this post should also be noted that the Concern with one Aspect of the system is also possible that this model is used less in systems with much less Concern [16].

5.2.2 Modularization Concerns Intervention using Aspect-Oriented

As we see in the previous section, aspect is an entity that can be cut from use case and Concerns, in the form of Intervention [13]. That will support modularity. There is a cut for any Concerns about the use of Intervention used in the model system. In fact, as an element of the stereotypical "aspect" will appear in your system and includes some Parts of the class as a Concern that the Intervention in other components of the system has appeared [17]. Aspect makes the piece of code that plays as an aspect role and is the main reason for the complexity and fragmentation in the system Removed and measured inside the desired component [14, 17]. Fig 6. the BookUpdate use cases that are dispersed in the different components have measured inside the BookUpdatin.

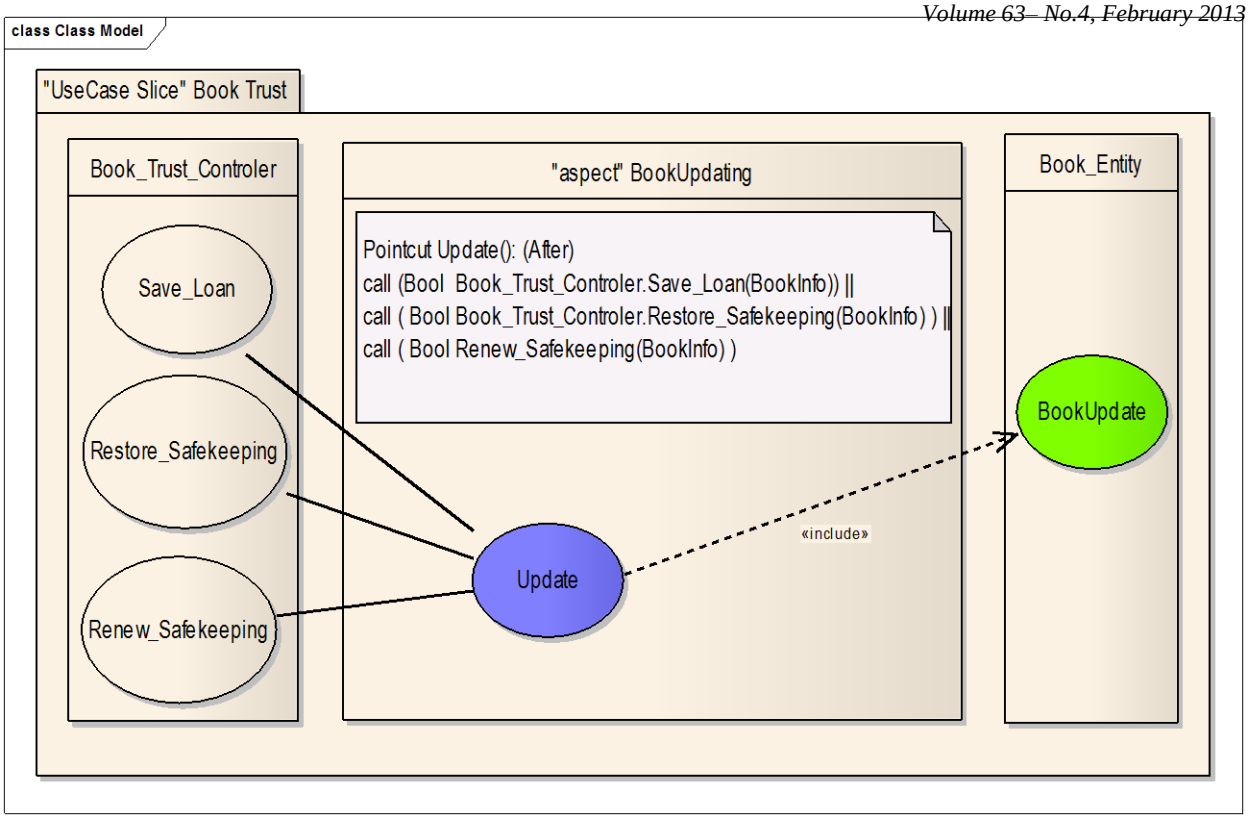

Fig 6: The Modularization of Book Update Concern within the Book Updating

As in Fig 6. see the section specifying the use case Book Updating and measure them in reducing their dependence on components and the dispersion is reduced, This in itself will reduce the system complexity. Checking Book aspects can also be expanded so as to cover cutting the existing use case in Book_Trust_Controler component. After entering into the system aspects we have taken on a big step on scattered Modularization concerns CheckingBook, BookUpdating in the system modeling phase as the complexity of the code are taken from foreign components [16]. Fig 7. shows the effect of Modularization concerns into aspects reducing on complexity and dispersion library management systems.

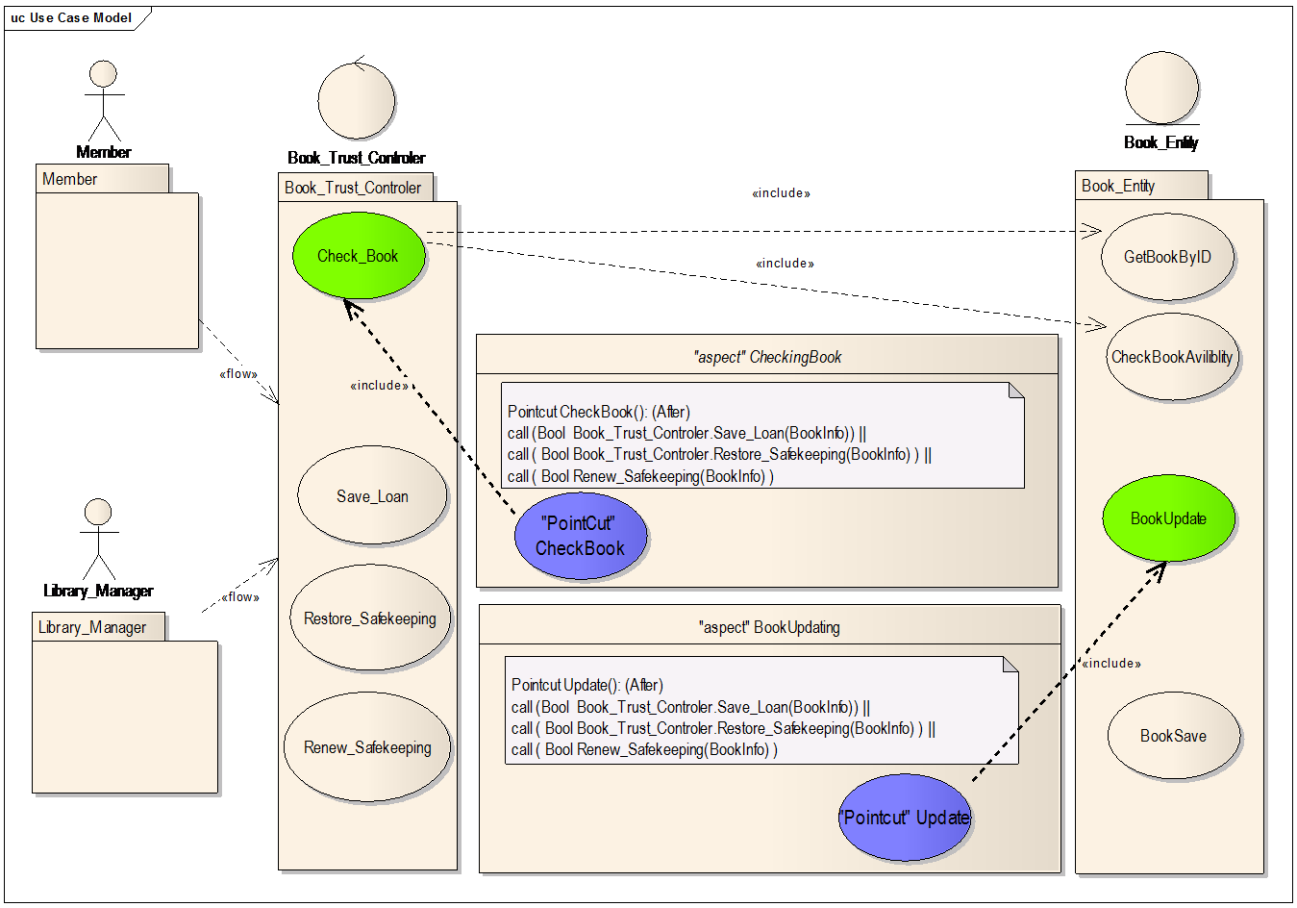

Fig 7: Entering the Aspects into the system components and their influence on the relations between use cases

As in Figure (7) you can see many ties in the system that existed before entering the Interfere with the concerns identified and classified them into modular aspects of the and the more cause independence in the system components [16, 17].

6. CONCLUSION AND FUTURE WORKS

The purpose of this paper is to Solve Concerns Intervention problem with using Aspect-oriented software development. In the article first identified that the Intervention Concerns are emerging the main stimulus Aspect-oriented software development methods. The paper also examines the Aspect-oriented programming language of Aspect-oriented software development which makes up the core infrastructure. After all, the Aspect-oriented software development began and the development phases, were check and finally we describes a method of Aspect-oriented software development in the form of case study Library Management System, There were cases in which the core business and we realize that the Aspect-oriented software development helps us to modularize the Concerns Intervention in the phases of design, modeling and architecture. The main topic of Aspect-oriented software development system is to achieve aspects of concern because it is a phase that all functions are transferred to the next phases. In fact, if not achieve a suitable method for obtaining concerns (aspects) in the requirements phase; we will not have the other Aspect-oriented software development. Therefore, in the future work trying to be providing a more precise focus on early intervention system finding concerns So as to identify them before the Implementation phase and added to the power of Aspect-oriented software development in Modularization concerns.

7. REFERENCES

[1] IEEE 2000, "IEEE recommended practice for architectural description of software intensive systems". Available: http://www.win.tue.nl/~johanl/educ/2II45/Lit/software-architecture-std1471-2000.pdf. Last Availabel 02.08.2012.

[2] R.S.Pressman, *Software Engineering: apractitioner's approach*, Fifth edition, McGraw-Hill, page 541, 2001.

[3] S. Brinkkemper,"Method engineering: engineering of information systems development methods and tools", *Inf. Software Technol*. Vol. 38, N. 4, pp. 275-280, 1996.

[4] B.Morin, O.Barais, R.Ramos,"Towards a Generic Aspect-Oriented Modeling Framework", Author manuscript, published in "Models and Aspects workshop", at ECOOP 2007. Availabel: http://www.irisa.fr/triskell/publis/2007/morin07a.pdf , Last Availabel: 02.08.2012

[5] S. Apel, D. Batory,"An Analysis of Eleven AspectJ Programs", Technical Report, Number MIP-0801

Department of Informatics and Mathematics University of Passau, Germany, April 2008.

[6] B., Nuseibeh, J.Kramer, and A.Finkelstein, "Expressing the relationships between multiple views in requirements specification", Proceedings of 15th International Conference on Software Engineering, Baltimore, USA, pp: 1-10, May 1993.

[7] G. Booch, I. Jacobson, and J. Rumbaugh, *Object Oriented Analysis and Design with Applications*, 3rd Edition, Addison.Wesley, 2007.

[8] D.L. Parnas, "On the Criteria To Be Used in Decomposing Systems into Modules", Communications of the ACM, Vol. 15, No. 12, pp. 1053-1058, 1972.

[9] K.Gregor, J.Lamping, A.Mendhekar, C.Maeda, C.Lopes, J.Loingtier, and J. Irwin "Aspect-Oriented Programming", Proceedings of the European Conference on Object-Oriented Programming, vol.1241, pp.220–242, 1997.

[10] R.E.Filman, D.P.Friedman, Aspect-oriented programming is quantification and Obliviousness, RIACE Technical Report 01.12, pp.1-9, May 2001.

[11] P. Jayaraman, J. Whittle, A. Elkhodary, and H. Gomaa. "Model Composition in Product Lines and Feature Interaction Detection Using Critical Pair Analysis", In MoDELS'07: Proceedings of the 10th International Conference on "Model Driven Engineering Languages and Systems", LNCS, pages 151–165, Nashville TN USA, Vanderbilt University,Springer-Verlag, Oct. 2007.

[12] B.Morin, O.Barais, J.M. J. Equel, " Weaving Aspect Configurations for Managing System Variability", Availabel: http://www.irisa.fr/triskell/publis/2008/Morin08a.pdf. Last Availabel: 02.08.2012.

[13] R.T. Alexander, J.M. Bieman, and A.A. Andrews,"Towards the Systematic Testing of Aspect-Oriented Programs", Technical Report CS-4-105, ColoradoState University,2004.

[14] A. Rashid, R. Chitchyan, "Aspect-Oriented Requirements Engineering: A Roadmap", Proceeding EA '08 Proceedings of the 13th international workshop on Early Aspects Pages 35-41 ACM New York, NY, USA ,2008.

[15] H.Hosny, A.A.Zakaia, Metrics for Aspect-Orinted Software Design, Technical report The American University in Cairo, 2004.

[16] O. Aldawud, T. Elrad, A. Bader, "Uml profile for aspect-oriented software development", Lucent Technologies Naperville, IL, Illinois Institute of Technology Chicago, IL, 2006.

[17] J. Araujo, A.Moreira, I. Brito, A. Rashid, "Aspect-Oriented Requirements with UML", Universidade Nova Lisboa, Instituto Politecnico de Beja, Lancaster University Lancaster, October 2002.