A Brief Survey on Aspect-Oriented Modeling Methodologies

Abstract-- The methodology of Aspect oriented software development is the new technique to improve the modularity in the area of software development. It eliminates the limitation of traditional classical approaches. It gives more emphasis on the crosscutting concern with the help of Separation. Over the last few years, several techniques of crosscutting have been introduced. In this article several existing approaches aspect oriented development have been surveyed.

Keywords-- Software engineering, Aspect oriented methodologies, Aspect-oriented Software Development Crosscutting

I. INTRODUCTION

Now a day’s the Aspect Oriented Software Development (AOSD) has become an emerging development in the era of computer science it provides various methods of modularity to user. It improves the present data used in software. There were few drawbacks in the classical languages because of their un-modularized data that results the scattering of code. Aspect oriented programming allows the modularization of data. The initial phase of Aspect Oriented Software Development is the Aspect oriented architecture; it improves crosscutting by separation of codes. It refers to the ability of encapsulation and manipulation of software that are critical in nature.

The prime motivation behind this article is to define and implement an Aspect Oriented Software Framework for the software development with minimum scattering of code. Crosscutting includes the basic faults such as exception handling. Such functionalities directly affect the software design and development. By using the aspect-oriented architecture, not only the efficiency of design can be enhanced but also the coding becomes so easier to reuse. The traditional procedural methodology or object oriented methodology unable to expand the software scale and software evolution process very well. But this can be done through Aspect Oriented Methodology as it deals with the crosscutting concerns.

Software architecture using Aspects-Oriented methodology has become a solution for the complex software encountered in a large system at the time designing and development. By using this methodology, reuse of a software application at different level become easier. Reusing software improves the time and cost management in the development and maintenance process.

II. LITERATURE SURVEY

During last few decades a large amount of work has been performed in the field of aspect-oriented software techniques. A research paper by Heba [2] aims to solve crosscutting concerns by offering better modularization codes. This paper provides a brief overview of the state-of-art in AOP. Another technique is introduced by Navasa [1] to propose the integrated concepts from aspect oriented methodology in the field of architectural design of software system. This particular work is focused on the aspect in which separation can be done by means of architectural style and Architectural Description Languages. Process modeling approach represents aspect from initialization of software to implement it in a better technique as proposed by Iqbal and Allen [5]. He suggests the identification of aspects in the Case Model and Sequence Diagram of the system. Kersten and Murphy [14] described the Object-Oriented development practices as well as some rules and policies that were employed to achieve maintainability. Walker [15] provided usefulness and usability of Aspect-Oriented programming. Hamza and Darwish [4] proposed a new approach to identify the model of candidate aspects from functional and non-functional requirements of the system and propagate them to design phase. Zakaria [13], proposed a quantitative assessment of Aspect Oriented solutions.

Mohmmad and Jorg[25] in his paper has analyzed the suitability of UML for modeling aspect-oriented software. They have taken a bottom-up approach starting from a small piece of code of program found in the aspect-oriented architectural and programming language. Through this paper they have discovered the similarities between the aspect and software connectors.

III. RESEARCH METHODOLOGIES

Research methodologies that are used in this paper in order to fulfill the objective is generally “Exploratory research studies”, whose main purpose is to formulating a problem for more precise investigation where several techniques has been tested. Another one is “feasibility research strategy”. This methodology has been depart from the hypothesis testing. Hypothesis testing determines the validity of the assumptions. This paper has a view which describes and implements software architecture in terms of Aspect-Oriented Software Development. The contribution of this paper has been taken into the account of initial hypothesis testing and its main goals. These contributions describe the aspect-oriented architecture design in a formal way. The software architecture solutions derived from this contribution should be satisfy the
feasibility of hypothesis that makes the easy implementation of the aspect-oriented software architecture [26]

IV. SIMULATION RESULTS & DISCUSSIONS

The impact of aspect-orientation techniques on the object-oriented methods seems to be a better result in the whole development life cycle of a system[9]. The implementation of software architecture using aspect-oriented techniques shows the positive result as compare to the previous techniques. These improved techniques enhance the software qualities like reusability, reduce complexities, maintainability etc which leads to the better development of whole life cycle. Discussion includes the crosscutting concerns, aspects and the inherent limitation of the class abstraction [11]. The architecture design activity should be integrated into the complete development of the system life cycle. It must be clear that how the elements of the requirement models are integrated in the architectural model [26]. The two main oppositions in aspect-oriented techniques is that in implementation of crosscutting modules, a hard core of understanding of module implementations is necessary. Another one is that it makes the debugging process much harder [2]. ASOD approach allows the ‘Separation Of Concerns’ by modularizing crosscutting concerns. The main emphasis in this approach has been made on implementation level.

Our discussion also considers the crosscutting concerns that inherit the limitations of class abstraction. The architecture design in the aspect-oriented must be integrated into the whole development of system life cycle. Also the validation and the evaluation under the architectural level is quite possible. The input which is to be installed in the system design must be appropriate under this phase. The aspect also describe that at the architectural level, functionalities also get affected. Aspect oriented concepts also assist the architectural phase of existing modeling language. It is also stated that the aspect-oriented approach makes the system easy to update, easy to extend and easy to reuse and also provide the interface to modularize the system and separate the software process. Once the crosscutting concern enters in this phase, it goes into the new aspect-oriented version. After that system enters into the new phase and get involved in the new environment and meet the requirements very easily. Using AOSD techniques in the development of the whole system, engineer can prove the beneficial result of crosscutting concern as the usage of modules at the all level of system is become possible. Synchronization aspect at all the modules of a system provides the benefit of reusable in codes. This paper will test the hypothesis that no reduction of coding is required. Aspect orientation is relatively a new paradigm that enhances the programming model under the software development. In this framework, there would be some instruction for engineer to follow so that a high quality of software can be developed.

CONCLUSIONS & FUTURE WORK

In this article, the crosscutting concerns are surveyed that are responsible for scattering of codes and cannot be rectified easily by using traditional procedural and object-oriented methods. Hence our paper used Aspect Oriented Software Development (AOSD) process in order to identify and rectify the crosscutting difficulties and nonfunctionalities. In this process, concerns are separated into the modules known as ‘Aspects’. This paper defines an Aspect-Oriented Software Architecture for software development with minimum complexities produced by the software. With the help of this architecture not only the design efficiency can be improved but also the reusability can be improved.

References


