Critical Analysis of current AORE and AOD Methods

Prerna Tyagi¹, Namrata Sharma² Mr. Vaibhav Yyas³

¹M.Tech (IT) student, AIM & ACT Banasthali University, Rajasthan, India
²M.Tech (CS) student, AIM & ACT Banasthali University, Rajasthan, India
³Asst. Prof., AIM & ACT Banasthali University, Rajasthan, India

Abstract— To build software structure extra configurable and maintainable system development is very difficult in parallel or distributed system because here several crosscutting concerns are available to hold. In aspect oriented Software Engineering is a changeable area Software Engineering takes new things to enhance, reliability, extensibility and flexiness of the technique. In aspect-oriented software development (AOSD) two big problems of existing object-oriented programming standard that help to decrease the crisis of code scattering and code tangling. Aspect-oriented requirement engineering is the first technique that is introduced the aspect-oriented concept in requirement level. Aspect-oriented requirement engineering is the first techniques that are introduced the aspect-oriented concept in requirement level.


INTRODUCTION-

AO(ASPECT-ORIENTATION)-

Different examples of aspects are distribution, sharing, resources and synchronization. Generally aspects are wrap together in the system main components generate difficulty of code tangling. Programming constructs are provided by AspectJ with the of programming constructs developers communicate aspect and the code at that place aspects must be applied. Requirements Engineering methods that openly recognise the consequence of clearly addressing both non-functional and functional crosscutting concerns, in adding together to non- crosscutting ones, are call Aspect-Oriented Requirements Engineering Approaches. The materialization of these approaches is encouraged by three major factors: first one is – 1. Composition 2. Traceability

3. Development of new technology

Figure1 - This figure shows the Aspect oriented Programing development stages of Aspectual decomposition and recomposition.

Above figure demonstrate the concern implementation of different stages. Following steps of concern implementation such as the aspectual recomposition.

AORE (ASPECT-ORIENTATED REQUIREMENT ENGINEERING): Aspect-oriented requirement engineering is the first techniques that are introduced aspect-oriented concept in requirement level. Necessity are forever starting from business difficulty e.g. improving automotive safety, processing passport applications and adding features to cell phones [3]. Assignment may build up fresh product or apprehension with growing presented or heritage system. The technique balances the requirement to make official the, viewpoints, requirements and composition operators and rules to effortlessness
representation and testimony generation among the requirement to remain the artefacts simple adequate to be understandable for the end users. [7]

AOSD (Aspect-oriented Software Development): AOSD approach offers an organized in favour of composition, representation, identification and modularization of crosscutting concerns. Crosscutting concerns for example synchronization as well as reliability consigns to value issues or else working of software so to facilitate cannot be presenting efficiently modularized by live software development approach such as Object-oriented technique. [8]

Figure 2- figure shows software requirements design models and program naming standards.

AOSD (Aspect-oriented software development) be a rising software development technique. AOSD (Aspect-oriented software development) permits various concerns to be show apart from moreover automatically joined in functioning system. [12] Established software growth has centre on top of decayed schemes hooked on part of key functionality.[8] Aspect-oriented software development (AOSD) concentrates on recognize, presentation and specification of crosscutting concerns also that modularisation divide functionality element while their automated composition in a working system.

AOP(Aspect-oriented Programming)-
AOP assembles on to peak of available methodologies e.g. Procedural programming and OOP, enhance them through concepts and build in command to modularize crosscutting concerns. With the help of AOP, by selecting base methodology execute core .Object oriented programming is an example of base methodology then core concerns is implementing as class implementing. In this method aspect summarizes crosscutting concern that is specifying the requirement of other component. Aspect orientation programming is a method that provides disconnection of crosscutting concerns. In numerous traditions, developing a scheme using Aspect oriented Programing is related to just beginning a method

By the use of different techniques implement them, or identifying those concerns, and explains shape of concluding system through joining. [1] The AOP research centre of population describes following steps of subsequent method:

1. Concern implementation
2. Aspectual decomposition
3. Aspectual decomposition

Modules with the help of AOP put into practice crosscutting concerns in aspects instead of fusing them in the core modules.[1] Aspect-oriented software development is ways that are basically designed to find improved activities of concerns, crosscutting concerns are aspect of a program that affect other concern. AOP language execution perform by two steps, first step is applying weaving rule for combining individual concerns after that it exchange the information into executable code. This process is called weaving. Code scattering be origin when a solitary issues are execute in different modules, while crosscutting concerns, through description, are extended more than numerous modules,associated implementations are furthermore scattered greater than different modules.

Following figure demonstrate working of banking scheme that are execute login with the help conventional approaches. Still after using a designed login module that suggests an API such as hides the formatting and stream the login message, every client account the module, ATM module, such as database module still needs the code to invoke the login API. [1]

Figure 3:- figure depicts the logging module that is dividing into following modules such as accounting module.
Each module working is correlated to logging module. For illustration, in a scheme using a database, performance concerns might have an effect on all the modules right to use the database. [1] Code tangling is origin at what time a component is put into practice with the purpose of manages numerous concerns concurrently. Developer frequently thinks about concerns such as performance, synchronization, logging, security, business logic, and so forth at the same time as implementing a module. This guide to the instantaneous occurrence of essentials from each concern’s functioning and marks in code tangling.

Aspect-Oriented Programming provides a further methodology of thinking concerning program Structure therefore on permits developers to divide and modularize issues as crosscutting issues. These issues square measure managed in a side therefore on permits to simply manage each the crosscutting and core issues. Our demonstration grasp up the suggests with the aim of the applying of side headed program can continuing an exceedingly high-quality detachment of issues and so in sensible movableness and maintainability of such systems. Aspect oriented Programming provides a further technique of thinking relating to program structure therefore on permits developers to divide and modularize issues as crosscutting issues. These issues are managed in a facet therefore on permits to simply manage each the crosscutting and core issues. Our demonstration grasp up the suggests with the aim of the applying of facet oriented program can continued in a very high quality detachment of issues and thus in smart movability and maintainability of such system.

AOM (Aspect Orientation Modeling) - A new approach of modularisation is generated by aspect orientation that separating crosscutting concerns. AspectJ has been originally emerging at the programming stage by aspect orientation of prominent protagonists. In aspect orientation approach prior to implement software development Life cycle following stages of (SDLC) software development Life cycle are requirement engineering, first one is analysis second one is design and coding . Development with Model Driven Engineering (MDE) is driven simultaneously.

In software development life cycle have compared to aspect orientation approach. Evaluation of AOM (Aspect orientation Modelling) techniques is necessary to initial create such an ordinary kind via signifies of an alleged CRM (Conceptual Reference Model) for Aspect Oriented Modeling. Basic ingredients of Aspect Oriented Modelling (AOM) explain by Conceptual Reference Model (CRM) and relationship between AOM and CRM represented by graphical Unified Modelling Language (UML) class diagram. Concern composition deals with different conceptual mechanism.

![Diagram of Business logic, security, Logging and persistence.](image)

**Figure 4:** Following Figure depicts the Business logic, security, Logging and persistence.

![Diagram of Implementation modules.](image)

**Figure-5:** Following Figure Manage the Implementation modules such as concern of following modules as Business logic, Persistence, Logging. [4]

Figure demonstrates how the accomplishment modules in a method all deal among both first one is system level and second one is business concerns. Different views are defining classification as a
composition of following concerns with the purpose of tangled mutually through the present implementation techniques. So that the autonomy of concerns cannot be managed.

I. LITERATURE REVIEW

AODM (ASPECT-ORIENTED DESIGN MODELING)-

Aspect-Oriented Design Modeling (AODM) technique enlarges ordinary UML among aspect-oriented impressions. Aspect Oriented design modeling Unified Modeling Language (UML) extension is originally described toward hold the aspect oriented impressions of AspectJ accomplishment language. Though, AODM has developed toward make generic along with till maintain asymmetric Aspect-Oriented Programming (AOP) technique. However, Symmetric Aspect-oriented implementation method is not sustained by AODM. AspectJ AOP is designed by AODM, design elements of AODM communicate to AspectJ. Behavioural and structural crosscutting both techniques supported by AODM. Crosscutting formation is shown in class diagram inside collaboration diagram. In aspect concept of composite advice is not available.

As well as Aspect- oriented Design Model is not concept of collaboration template. Because of that Collaboration template introduce join point designation diagrams. IPDD essentially be UML diagrams as activity diagrams, sequence, class and object diagrams and state charts e.g. wildcards signature patterns and name. Joint point destination diagrams shows an self governing pointcut method that be able to applied some UML based Aspect oriented modelling language, permit to choose all type of joint points as behavioural-static, structural static and structural-dynamic.

Theme/UML Method-

The theme is a study and proposed method that maintains the partition of concerns used for design and analysis stages of software lifecycle. Theme technique gives a Unified Modeling Language (UML) supported Aspect Oriented Design (AOD) language that identifies Theme/UML that expands the Unified Modeling Language (UML) meta-model. Theme technique shows worries in theoretical with proposes builds term theme. Themes be very simple to compare to aspects and include worries among related to the symmetric separation. Other anxiety, if it is crosscutting or it is not, might be summarized into a theme. Aspect and base theme are representing with the help of theme/UML. Aspect and base are defines in packages. Theme maintains the requirement of two kind of integration first is Override and second one is merging. Because of symmetric model to facilitate theme sustains, integration relationships be able to precise between dissimilar themes kinds. Integration requirement is signified through a curve moreover be able to identify next two levels, Override integration is a condition for structural crosscutting. Override crosscutting be damaged to identify that, lie on top between theme essentials is to determined by restore the essentials in accessible themes through the theme essentials so as to overriding. It is connecting theme elements as well as specifies the substitute of essentials across theme. By itself prevail integration be represented through one direction arc. Second merge integration is a requirement for structural crosscutting. Merge integration take hold to identify that partly cover among theme essentials is used to determine through calculating the parts for partly cover elements simultaneously. The Result for merge is the addition of the fraction of those essentials particular to exist merged. A merge requirement is capable to merge elements among a lot of themes. So it be capable for measured to exist requirement intended for structural crosscutting. With the help of two ways arc merge integration is represented.

Aspect-Oriented Architecture Modeling (AAM)-

AAM (Aspect-oriented Architecture Modeling) is a technique so as to define middle to high propose levels. Aspect-oriented Architecture Modeling such as AOD language sustains two stages of aspects one is context-free and second is a context specific aspect. The context-free facet to state by the high level. And this facet is that the reusable facet. Aspect-oriented design modeling technique is support metamodelling. Context-specific aspects area unit aspects that area unit sample of context. Free aspects thus on the be precise for reprocess all above all style models. Aspect –oriented architecture modeling as well as AOD language holds two kinds of aspects fiets one is context free.

Figure-6: An aspect-oriented design model.
and second one is context specific aspects. Context free aspects are usable aspects with the aim of expressed next to high level. Context specific facets square measure those aspects that square measure example of context free aspect with the aim of nominal reprocess within elaborate style models. Aspect- oriented modeling not clearly converse in condition of AOD method. In study the first model be recognized and better level aspects therefore on capture a meticulous crosscutting concern, a middle level side be necessary to know to facilitate concern. During style part the first model be articulated within a middle level design. On now, style be able to charge the so much on top of the bottom level side styles accessible to look at if perspective necessary to become aware the so much on top of the bottom level aspects within a middle level design, be accessible within the first model.

Motorola WEAVR Approach- The Motorola WEAVER technique has been build up in a manufacturing setting, particularly in perspective of telecommunication industry. It makes a use of Specification and Description Language (SDL) to identify representation. The WEAVER technique is, therefore, supports UML 2.0. Model Behaviour is used by sequence diagrams, use class diagrams and action language of SDL. Entity aspect is characterized by <aspect> a pointcut-advice and stereotype method is used in support of composition of aspects as well as target models. Both code generation and target execution be also maintained. The Motorola Weaver is associate degree add-in to Teleological that performs four distinct functions. Motorola Weaver includes a summary. That permits the consequences of an aspect on a model to be pictured and valid. Third, it performs full side weaving at the modeling level. It provides a join point visual image engine that allows the implications of an aspect on a model to be pictured plus valid. Third, Motorola Weaver execute full surface weaving on the modelling level. Finally, it includes a simulation engine that permits aspect models be simulated, whereas not breaking the quality structure of Aspects. That permits the impact. The level of development of the Motorola Weaver is holding on initial stage. It’s difficult to develop the quantity ion the tool. Additional buildup the visualization engine testing capability and simulation for aspect orientated model. It is necessary to enhance the quantity of accepting and assurance in aspect-oriented technique along with development teams.

**Simplicity**- There should be appropriate logical clarification of all diagrams, artefacts and semantics used in the language. In this criteria language is tested for comprehensiveness of artefacts.

**Extensibility**: This measure evaluates whether the modeling language supports modeling of aspecual and non aspectual components as well. If more non aspecual components need to be modelled is it possible by the language to model those components. If components introduced other then mentioned in the specification of language then how it could be modelled by proposed language.

**Scalability**: In this criterion the modeling language is measured against capability of handling expansion and changes. AODM is evaluated against available literature over undersized and outsized systems.

**Traceability**: In this criterion evaluation is done on whether components (aspects) are traceable from requirements engineering stage to implementation stage of software life cycle. Software constructs should be traceable from initial stage to later stages and also traceability of component from abstraction level to advanced level.

**TABLE OF SIMPLICITY, EXTENSIBILITY**-

<table>
<thead>
<tr>
<th>Approach</th>
<th>Simplicity</th>
<th>Extensibility</th>
</tr>
</thead>
<tbody>
<tr>
<td>AODM</td>
<td>Mid</td>
<td>Provided Partially</td>
</tr>
<tr>
<td></td>
<td>Explanation: AODM provides template based approach it offers templates with parameter for depicting behavioural characteristics of aspect. The simplicity and comprehensibility of language is low.</td>
<td>Explanation: AODM supports incorporation of new aspecual components. It follows UML profiling approach.</td>
</tr>
<tr>
<td>Theme/UML</td>
<td>Mid</td>
<td>Provided Partially</td>
</tr>
<tr>
<td></td>
<td>Explanation: This approach provides patterns based on UML for composing individual elements. The nitrated model</td>
<td>Explanation: Theme / UML approach based on metamodelling. It allows extensions of non aspecual components.</td>
</tr>
</tbody>
</table>
The approach also uses UML notations and models for implementation of system elements. Because of crosscutting behaviour aspectual components like join points, pointcut and advices should have different representation.

**Provided Explanation:**
The UML approach based on metamodeling. It allows implementations of non aspectual components.

**Motorola Weaver approach**

**Mid-High Explanation:**
There is an in-depth use of state machines to represent aspects and pointcuts that is not as apprehensible as a national illustration of those constructs may well be.

**Provided Partially Explanation:**
There is no support on the market for the heavy-weight extension nevertheless. The light-weight extension mechanism is supported by the approach.

---

**TABLE I**

This table explains the simplicity and extensibility of different approaches. This table define the evaluation criteria of approaches. Simplicity explains the comprehensiveness of artefacts. Extensibility supports modelling components non aspectual and no aspectual.

<table>
<thead>
<tr>
<th>Approach</th>
<th>Traceability</th>
<th>Scalability</th>
</tr>
</thead>
<tbody>
<tr>
<td>Theme/UML</td>
<td>Provided</td>
<td>High</td>
</tr>
<tr>
<td>AOSD/UC</td>
<td>Provided</td>
<td>Low</td>
</tr>
<tr>
<td>AAM</td>
<td>Provided</td>
<td>Low</td>
</tr>
<tr>
<td>JAC Design Notations</td>
<td>Provided</td>
<td>Low</td>
</tr>
</tbody>
</table>

---

**TABLE OF TRACEABILITY, SCALABILITY**

<table>
<thead>
<tr>
<th>Approach</th>
<th>Traceability</th>
<th>Scalability</th>
</tr>
</thead>
<tbody>
<tr>
<td>AODM</td>
<td>Partially</td>
<td>Low</td>
</tr>
<tr>
<td>AOSD/UC</td>
<td>Provided</td>
<td>High</td>
</tr>
</tbody>
</table>
uses the concept of use case slices. This approach provides tracing on internal components.

provides case study over complex hotel management. This approach is scalable for complex systems also.

<table>
<thead>
<tr>
<th>AAM</th>
<th>Partially</th>
<th>Explanation: AAM approach is based on representation at architecture level. Tracing of components from requirements engineering to later stages are not supported.</th>
<th>Mid</th>
<th>Explanation: Over the AAM approach availability of literature is insufficient. No example found for applicability of AAM over any complex system.</th>
</tr>
</thead>
<tbody>
<tr>
<td>JAC Design Notations</td>
<td>Not Provided</td>
<td>Explanation: No support for models to be traced from abstract to refined level. Only traceability of abstract diagrams between different stages of software development are possible.</td>
<td>Mid</td>
<td>Explanation: JAC design having limited applicability over Aspect modelling. No scalability example of the approach is available.</td>
</tr>
<tr>
<td>Motorola Weaver Approach</td>
<td>Not Provided</td>
<td>Explanation: External traceability is not supported. As way internal traceability is bothered.</td>
<td>Mid</td>
<td>Explanation: The approach is ascendible that is clear by the examples provided within the literature. A close telecommunication equipment has been enforced that shows the ascendible nature of the approach.</td>
</tr>
</tbody>
</table>

**TABLE II** - This table explain the traceability and extensibility of different approaches. This table defines the evaluation criteria of different approaches. This table shows efficiency and maturity of following approaches. Scalabilitygility gives criteria of modeling language is measured next to capability of management expansion and changes. Traceability criterion assessment is done scheduled whether components are traceable from requirements engineering stage to accomplishment phase of software life cycle.

**CONCLUSION** - In this paper, we explain six approaches with respect to aspect-oriented modeling. we have studied six approaches Theme/UML, AODM, AAM, JAC design notations, AOSD/UC. AODM approach provides partially traceability and low scalability. Aspects of the scheme summarize the crosscutting concerns, that are specify the following components inside the scheme requirement that is rush mutually to figure the absolute scheme. AspectJ makes available sustain for modular execution of a variety of crosscutting. Aspect-Oriented Technologies during this context. We tend to confer the Motorola Weaver, its joint point model for transition oriented state machines, and discussed its current of adoption.

The eventual objective of Aspect oriented Programming is to build it probable to work among crosscutting aspects of systems manners as separately as feasible. Aspect Oriented Programming is altering the manner for the growth of software in some regions of computing and software Engineering. Aspect Oriented Programming is a programming thing that lifts modularity through impassiveness of cross-cutting concerns along with therefore, results in enhanced maintainability and portability of distributed applications.

**References** -


