Performance Analysis of Qualitative Evaluation Model for Software Reuse with AspectJ using AHP

Ravi Kumar[1]
Research Scholar, MMICT& BM
Maharishi Markandeshwar Deemed to be University
Mullana (Ambala)-133207, Haryana, India

Dalip[2]
Assistant Professor, MMICT& BM
Maharishi Markandeshwar Deemed to be University
Mullana (Ambala)-133207, Haryana, India

***Abstract*—Reusability is necessary for developing advance software. Aspect Oriented programming is an emerging approach which understand the problem of arrangement of scattered software modules and tangled code. The aim of this paper is to explore the AOP approach with implementation of real life projects in AspectJ language and its impact on software quality in form of reusability. In this paper, experimental results are evaluated of 11 projects (Java and AspectJ) using proposed Quality Evaluation Model for Software Reuse (QEMSR) and existing Aspect Oriented Software Quality Model (AOSQ). To evaluate AOP quality model QEMSR based on developers AOP projects by using Analytic Hierarchy Process (AHP) tools. Paper provides the evaluation of software reusability and positive impact on software quality. QEMSR model is used to assess Aspect Oriented reusability quality issues, which helps developers to adapt for software development. The overall quality of three models QEMSR, existing AOSQ and PAOSQMO are 0.62552223, 0.5283693, and 0.505815 calculated. According to this, QEMSR model is best in form of quality in same characteristics and sub-characteristics.**

Keyword—Reusability; AspectJ; software quality metrics; analytic hierarchy process

I. INTRODUCTION

Various software quality models described the assessment of software quality in software engineering. Quality assessment of software is an interesting research area in software engineering. Several AOSD seminars, workshops and research conferences had considered evaluation of quality of software model is emerging sector in traditional software engineering journals and conferences. According to IEEE/ACM "Software Engineering Curriculum Guidelines list software engineering education" in 2004 as one of the ten specific areas of software engineering education[5][20]. Various international network groups and research communities are working on software evolution. Software evolution concerned issues are very complex because it engages with various dimensions.

This paper focuses performance evaluation of proposed Qualitative Evaluation Model for Software Reuse (QEMSR) by experimentation method using characteristics and its sub-characteristics. We describe some metrics such as WMC, DIT, NOC, LCOM, and CBO for statistical value [10]. We also analyze the existing model such as Aspect Oriented Software Quality Model (AOSQ) and Proposed AO Software Quality Model (PAOSQMO) to examined performance evaluation. The negative impact on software quality is duplication of code. Crosscutting concerns reduced to have negative effect on understandability, maintainability, operability, modularity because understanding and changing crosscutting concerns requires touched various place in source code.

In existing system, firstly crosscutting concerns are derived after that distinguishes into aspects. Main traditional software reveals crosscutting concern that is called "tyranny of the dominant decomposition." In existing system, exploration helps to find out aspect. Aspects will help the software developers to examine where and how these tangling and scattering codes are implemented and its effect on quality of software [9]. This process is called aspect mining which is used to examine crosscutting concerns in existing model codes.

Contribution of the paper:

- To examine area of evolution of traditional programming (OOPs) different form evolution of Aspect Oriented Programming (AOP).
- To promote evolution of Object-oriented Programming (OOPs) be implemented to Aspect-oriented Programming (AOP).
- To improve performance evaluation of software quality models in software engineering.

This paper divides into eight sections. First section describe introduction about Aspect-oriented Programming. Related work has been done by the researcher explain in section two. Third section defines the framework or method to achieve research goal and motivation to do that work. Section four and five describe the platform used for practical work and design and result of experiment. Section six describes the analysis of experimental result and qualitative evaluation of 11 research case studies and its impact on quality. Examine performance evaluation of QEMSR model and existing model is described in section seven. In section eight, we discussed major finding of proposed quality model as conclusion and area for future research work for researcher point of view.

II. LITERATURE REVIEW

In late 1990s, Aspect–oriented Programming (AOP) is an emerging area in evolution of software and it declares the positive impact on software quality; simultaneously, various risks, challenges and paradoxes for AOP adoption for development of software. In 2006, Steimann stated the question:

"Does aspect orientation really have the substance necessary to found a new software development paradigm or is it just another term to feed the old buzzword permutation based research proposal and PhD thesis generator?"

In 1997, Kiczales explore the idea of AOP pattern to modularize the crosscutting concerns in existing system. Table I shows last ten years quality models which is described time to time by researchers. Kumar et. al. extends the ISO/IEC 9126[11] quality model by adding some extra characteristics and sub-characteristics in 2009, called Aspect Oriented Software Quality Model (AOSQUAMO). AOSQUAMO model is first purely based on Aspect Oriented Software Development (AOSD). In 2010 another quality model REASQ is derived by Castillo et.al. REASQ quality model is the combination of ISO/IEC 9126 and ISO/IEC 25030 define by UML.

Simultaneously, Kumar et. al. adds evolvability as an attribute in software quality model for AOP application in 2012 named Aspect-oriented Software Quality (AOSQ) model [1] [18]. This model described four sub-characteristics such as sustainability, design stability, extensibility and configurability [4] [16]. AOSQ model is based on AOSQUAMO and ISO/IEC9126 quality model [23].

G. Suryanarayana et.al. described MIDAS [13] model to analyze design quality assessment method for industrial software in 2013. T. Alrawashdeh and M.I. Muhairat were exploring the quantative evaluation of enterprise resource planning systems proposing ERPSQM model in 2014[3] [12] [17]. In 2016, Pardeep Kumar Singh and Yugal Kumar assess the empirical evaluation of Aspect-oriented software quality model using multi-criteria decision making approach using PAOSQMO model.

Pankaj Kumar and S.k. Singh also measure a comprehensive evaluation of Aspect-oriented software quality model (AOSQ) using Analytic Hierarchical Process (AHP) [26] [28]. In 2018, Petrus Mursanto and Dameria Christina Pasaribu define software quality rank using AHP and Object-oriented metrics which is used to perform evaluation of quality of QEMSR model[14][24][30].

Sufia Nadeem Chishti explores the quality improvement in small scale projects using Aspect Oriented design in 2019[2] [19]. S. Dixit explores the performance of quality modeling using artificial neural network technique in Aspect Oriented Programming [7]. P. Kumar analyzes the metrics of Aspect Oriented and Object oriented using AspectJ and Java programming languages [8].

Hamed Fawareh proposed the software quality model for maintenance software purposes [6]. Bharti Bisht describes the metric approach to anticipate reusability of object oriented software systems [21].

K. Chitra measures the performance merits of software component using CK metrics [27]. We evaluate quality of QEMSR model using Analytic Hierarchical Process (AHP) that is based on AOS Quality Model (AOSQ) and PAOSQMO [25].

TABLE I. SOFTWARE QUALITY MODEL

Sr. No.	Quality Model	Year
1	Aspect-oriented Software Quality Model(AOSQUAMO)	2009
2	Quality Open Source Software (QualOSS) Model	2009
3	A software Component Quality Framework (Alvaro Model)	2010
4	REquairements, Aspects and Software Quality (REASQ) Model	2010
5	SCQM (Upadhyay Model)	2011
6	Software Quality Evaluation User's View (Al-Badareen Model)	2012
7	Quamoco Quality Meta–Model	2012
8	Aspect Oriented Software Quality Model (AOSQ)	2012
9	Method for Intensive Design Assessments (MIDAS) Model	2013
10	Aspect Oriented Software Reusability Measurement (AOSRM)	2014
11	ERPSQM	2014
12	Proposed Aspect Oriented Software Quality Model (PAOSQMO)	2016
13	Software Quality using AOP based Small Scale Projects	2019
14	AOSQ using Fuzzy Logic Model	2020
15	SQM for Maintenance Software Purposes	2020

III. MOTIVATION AND METHODOLOGY

Last few years, various researcher working on different software quality model in software engineering. All researcher derived own quality model using some characteristics and metrics. These researchers also evaluate only derived model and not compared other researcher model in respect of quality. Every researcher use different technique to evaluate own quality model like Analytic Hierarchy Process, fuzzy logic, Gang of Four design pattern, etc. No anyone researcher can perform quality evaluation with same parameter with different quality model which is identify best model. So, we decide or motivate that we perform or derive a quality model in respect of reusability and its characteristics and metrics and compare with other model with same parameter. We also extend the qualitative evaluation of a model in more informative form, which helps for software developers to take decision to implement software or applications.

We can assume research methodology for this paper is software reengineering which is comparison analysis technique. Firstly, we can divide our objective into two parts like goals and sub-goals as shown in Fig. 1. In goals part, we define performance evaluation as purpose and concept use reusability. In sub-goals, internal characteristics and metrics are defined which measure the statistical data to evaluate quality. We can re-engineer concept that involve forward and reverse engineering principles. For experimentation purpose, we use quasi-controlled experimentation.

According to QEMSR model, research manipulates one or more independent variables to examine their impact on one or more dependent variables, set of metrics and validation of metrics [15]. We also describe the experimental part using 11 real world projects. We implement these projects in AspectJ and Java language and assign weight of methods and calculate average mean value for qualitative evaluation. All the 11 projects implement to assess contemporary phenomena within its real world situation.

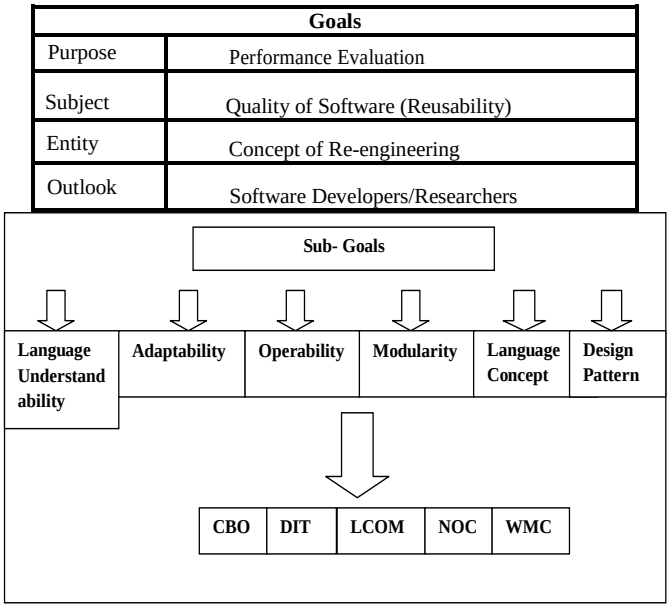


Fig. 1. Framework of QEMSR Model.

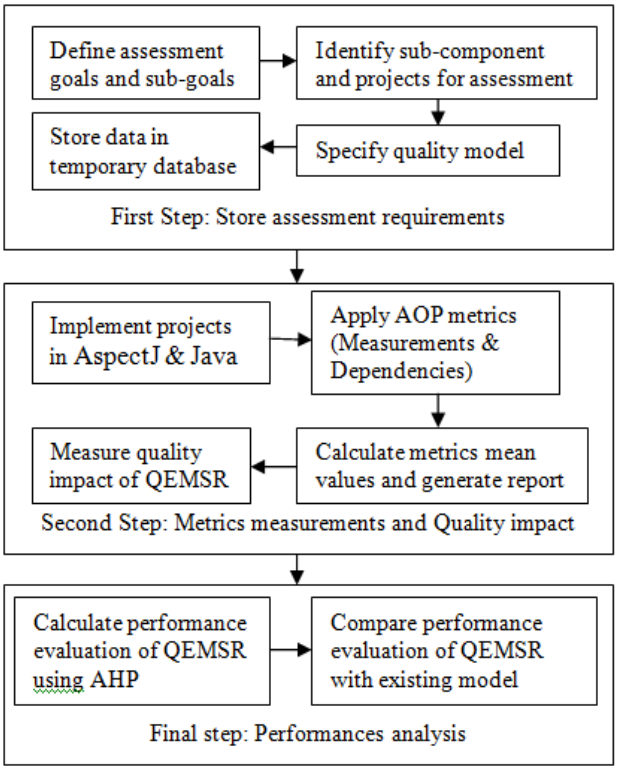


Fig. 2. Methodology for Performance Evaluation of QEMSR.

To achieve goals and sub-goals, we also use R. Marti, Henry and Li, Garcia et. al. and C & K metrics definition and these metrics associated for quality measurement in AOP [29]. QEMSR model proposed to validate metrics and analysis of qualitative evaluation and its impact on quality for AOP. To validate metrics we use experimental results of 11 projects implementations (Java & AspectJ). Experimental result gives intuitive information for the analysis of evolutionary aspects during Aspect-oriented software evolution. Fig. 2 describes the methodology for performance evaluation of QEMSR.

IV. Experimental Set-Up

Set-up for experimentation is for 11 projects (AspectJ and Java) to collect descriptive value (metrics) for the analysis of quality of software using AOP metric tools; a common AOP metric tool for both Aspect-oriented and Object-oriented metrics, such as R. Martin, Henry and Li and C & K. For doing experiment operating system required MS Windows XP/7/8, AspectJ 1.6, Java JDK 1.6v and AOP metrics 0.3 binary[20]. Ms-excel sheet generated for manipulation of descriptive data after successful execution of set of list files in a command line for a given source running compile.bat,(.1st)(projects) and metrics.bat files. All these descriptive data used for analysis for several AOP characteristics by impact tests and statistical tests.

V. Experimental Design and Results

We can design procedure for 11 projects (AspectJ and Java) implementation for analysis of quality of AOP software consist five steps:

- Description of 11 projects which is used for experimentation or implementation (Java and AspectJ) as shows in Table II.
- Collection of data for experimental results and descriptive data used for AOP metric tools shown in Table IV.
- QEMSR framework which shown in Fig. 1.
- Methodology for performance evaluation of QEMSR shows in Fig. 2.

Ms-excel sheet generated for manipulation of descriptive data after successful execution of set of list files in a command line for a given source running compile.bat, (.1st)(projects) and metrics.bat files. All these descriptive data used for analysis for several AOP characteristics by impact tests and statistical tests.

The main goal to provide qualitative evaluation using 11 real world projects implementation (AspectJ and Java) using metric and statistical data with regard to reusability characteristics and sub-characteristics from the software developers view point. Only interesting metrics for this evaluation is DIT, NOC, CBO, LCOM, WMC of reusability characteristics and sub-characteristics. In this paper 11 projects real world system from different size and domain is shown in Table II. Table III shows the description of metrics adapted for QEMSR. Table IV shows the absolute mean values of 11 projects (AspectJ and Java).Using measurement of metrics we evaluate the experimental results on 11 projects and correlation among reusability characteristics and sub-characteristics. Table V shows the difference of average mean value of all 11

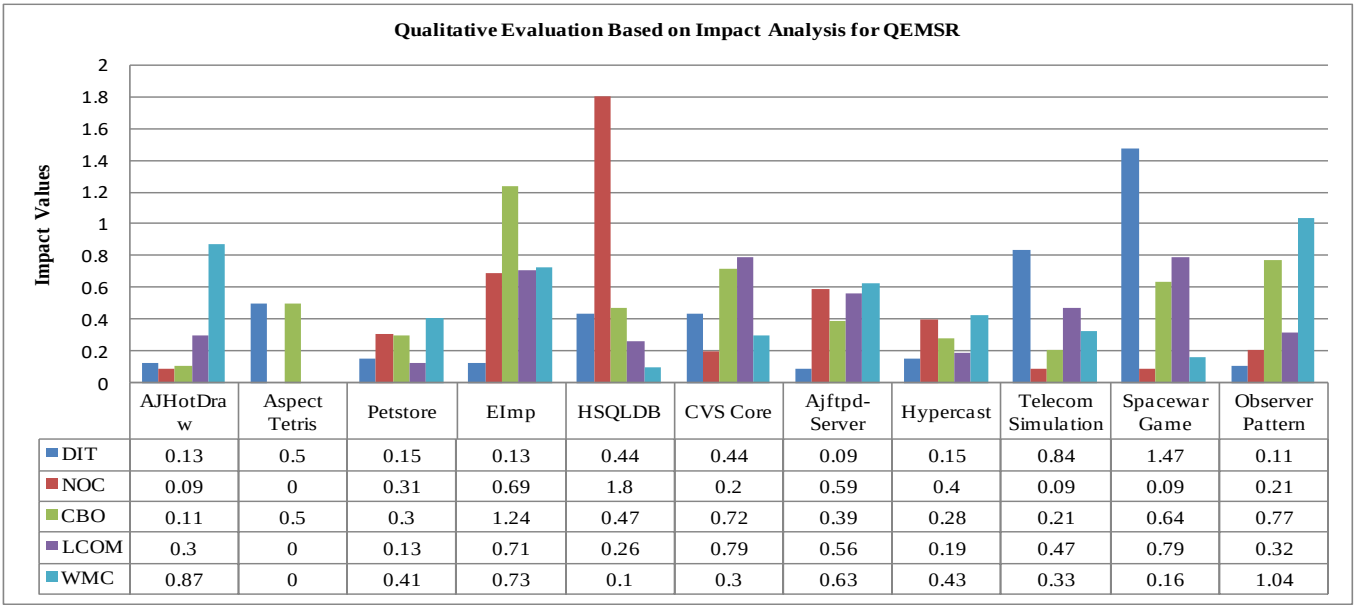
projects metrics and also calculate the impact of every metrics as graphically shown in Fig. 3. Table V contains the average mean value of metrics calculated as sum of different module divide by number of module taken for analysis. AHP is applied on these mean values to get corresponding weights of characteristics and sub-characteristics in which total quality weight has been taken as 1.000. These weights used for comparing for Aspect –oriented projects. Aspect-oriented version of 11 projects shows an improvement in all structured complexity metrics. So for performance evaluation we compare existing AOSQ model and PAOSQMO model to select best suitable model for implementation in Aspect-oriented technology based projects.

TABLE II. DESCRIPTION OF 11 PROJECTS (ASPECTJ & JAVA)

Name of projects	Description
AJHotDraw	Framework for structured and technical 2D graphics. http://ajhotdraw.sourceforge.net
AspectTetris	Implementation of Tetris game in AspectJ. http://www.guzzt.com/coding/aspecttetris.shtml
PetStore	Demo for the J2EE platform which represent existing applications of E-commerce. http://java.sun.com/developer/releases/petstore/
Eimp	Eclipse plug-in which support collaborative software developments for distributed teams. http://eimp.sourceforge.net
HSQLDB	Used for a relational database management system implementation. http://vrwxv.hsqldb.org
Hypercast	Software for developing application programs and protocols for overlay network, application layer.
CVS Core	Eclipse plug-in which implements the basic functionalities of a CVS client such as check out and check in system stored in a remote repository. http://www.eclipse.org/eclipse/plateform-cvs/
AJFTPd-Server	Crosscutting concern implementation for security. Application level Server for BLP access control. http://homepages.wmich.edu/plbijjam/cs555 Projects/
Telecom AspectJ	Examples of AspectJ http://www.eclipse.org/aspectJ/
Spacewar Game AspectJ	Examples of AspectJ http://www.eclipse.org/aspectJ/
Observer Pattern AspectJ	Examples of AspectJ http://www.eclipse.org/aspectJ/

TABLE III. DESCRIPTION OF METRICS ADAPTED FOR QEMSR

Name of Metrics	Description
WMC	Total number of weighted operation in a class
CBO	Total number of interfaces declaring class or number of class or fields which can be called by a given class
LCOM	Total pairs of operation working on common fields minus total number of pairs of operation working on different
DIT	Longest path length From aspect/ class to the given class hierarchy root
NOC	It measures the total number of class, immediate descendants.



	AJHotDraw	Aspect Tetris	Petstore	EImp	HSQLDB	CVS Core	Ajftpd-Server	Hypercast	Telecom Simulation	Spacewar Game	Observer Pattern
DIT	0.13	0.5	0.15	0.13	0.44	0.44	0.09	0.15	0.84	1.47	0.11
NOC	0.09	0	0.31	0.69	1.8	0.2	0.59	0.4	0.09	0.09	0.21
CBO	0.11	0.5	0.3	1.24	0.47	0.72	0.39	0.28	0.21	0.64	0.77
LCOM	0.3	0	0.13	0.71	0.26	0.79	0.56	0.19	0.47	0.79	0.32
WMC	0.87	0	0.41	0.73	0.1	0.3	0.63	0.43	0.33	0.16	1.04

Fig. 3. Qualitative Evaluation based on Impact Analysis for QEMSR.

TABLE IV. ARITHMETIC MEAN VALUES OF QEMSR METRICS OF 11 PROJECTS

Projects /Metrices	Reusability and its Sub-characteristics									
	Modularity				Operability		Adaptability		Understandability	
	DIT		NOC		CBO		LCOM		WMC	
	AO	OO	AO	OO	AO	OO	AO	OO	AO	OO
AJHotDraw	1.233	1.418	0.4794	0.5288	0.3390	0.3064	0.4567	0.6569	0.4976	0.2663
Aspect Tetris	0.333	0.667	0.667	0	1.00	0.667	0	0	0	0
Petstore	1.021	1.208	0.2784	0.4038	l.3904	1.068	1.367	1.5696	0.976	1.663
EImp	1.233	1.418	0.4794	1.5288	2.3904	1.068	0.4567	1.5696	0.4976	1.863
HSQLDB	0.233	0.418	1.4794	0.5288	0.967	0.6569	0.1976	0.2663	0.3390	0.3068
CVS Core	0.233	0.418	0.4294	0.538	0.1567	0.5696	0.2976	0.1663	1.3904	1.068
Ajftpd-Server	2.1233	1.9418	1.4794	0.9288	1.567	2.5696	0.276	0.63	0.3904	1.068
Hypercast	2.1233	1.8418	0.8794	0.6288	0.2567	0.3569	0.1976	0.1663	0.3904	0.680
Telecom-simulation	0.233	1.418	0.4794	0.5288	0.567	0.4696	0.976	0.663	1.3904	2.068
Spacewar Game	1.033	0.418	0.4794	0.5288	2.567	1.5696	0.2976	0.1663	2.3904	2.068
Observer Pattern	1.133	1.018	0.4094	0.5188	0.1567	0.6696	0.2976	0.2263	1.3904	0.680

TABLE V. QUALITATIVE EVALUATION BASED ON IMPACT ANALYSIS FOR QEMSR

Project / Metrics	Reusability and its Sub-characteristics														
	Modularity						Operability			Adaptability			Understandability		
	DIT			NOC			CBO			LCOM			WMC		
	Diff	Impact	Qualitative Evaluation	Diff	Impact	Qualitative Evaluation	Diff	Impact	Qualitative Evaluation	Diff	Impact	Qualitative Evaluation	Diff	Impact	Qualitative Evaluation
AJHotDraw	0.19	0.13	Extremely Helpful	0.05	0.09	Extremely Helpful	0.03	0.11	Extremely Helpful	0.20	0.30	Very Helpful	0.23	0.87	Not so Helpful
Aspect Tetris	0.33	0.50	Helpful	0.67	0.00	Extremely Helpful	0.33	0.50	Helpful	0.00	0.00	Extremely Helpful	0.00	0.00	Extremely Helpful
Petstore	0.19	0.15	Extremely Helpful	0.13	0.31	Very Helpful	0.32	0.30	Very Helpful	0.20	0.13	Extremely Helpful	0.69	0.41	Helpful
EImp	0.19	0.13	Extremely Helpful	1.05	0.69	somewhat Helpful	1.32	1.24	Not at all Helpful	1.11	0.71	somewhat Helpful	1.37	0.73	somewhat Helpful
HSQLDB	0.19	0.44	Helpful	0.95	1.80	Not at all Helpful	0.31	0.47	Helpful	0.07	0.26	Very Helpful	0.03	0.10	Extremely Helpful
CVS Core	0.19	0.44	Helpful	0.11	0.20	Extremely Helpful	0.41	0.72	somewhat Helpful	0.13	0.79	somewhat Helpful	0.32	0.30	Very Helpful
Ajftpd-Server	0.18	0.09	Extremely Helpful	0.55	0.59	Helpful	1.00	0.39	Helpful	0.35	0.56	Helpful	0.68	0.63	somewhat Helpful
Hypercast	0.28	0.15	Extremely Helpful	0.25	0.40	Very Helpful	0.10	0.28	Very Helpful	0.03	0.19	Extremely Helpful	0.29	0.43	Helpful
Telecom simulation	1.19	0.84	Not so Helpful	0.05	0.09	Extremely Helpful	0.10	0.21	Very Helpful	0.31	0.47	Helpful	0.68	0.33	Very Helpful
Spacewar Game	0.62	1.47	Not at all Helpful	0.05	0.09	Extremely Helpful	1.00	0.64	somewhat Helpful	0.13	0.79	somewhat Helpful	0.32	0.16	Extremely Helpful
Observer Pattern	0.12	0.11	Extremely Helpful	0.11	0.21	Very Helpful	0.51	0.77	somewhat Helpful	0.07	0.32	Very Helpful	0.71	1.04	Not at all Helpful

Less than 0.20 = “Extremely Helpful” 0.20-0.40 = “Very Helpful” 0.40-0.60 = “Helpful”
0. 60-0.80 =”somewhat Helpful” 0.80-1.00=”Not so Helpful” Greater than 1.00 =” Not at all Helpful”

VI. Evaluation of Results

The collection of data for every module (interface, class, aspect) of every system use the extended version of Aspect-oriented metric tools. For every real life project experimental result are represented independent. Crosscutting concerns investigated intensively for all 11 projects which show in Table II. For all project system represent common software problems and solution of those problems. Table IV define the average mean value of Aspect-oriented and Object-oriented implementations of 11 projects. The measurements of metrics have been computed but experimental results of 11 projects. The evaluation of quality of QEMSR model using characteristics and sub-characteristics and metrics adopted from C & K metric suite such as NOC, DIT, LCOM, WMC, and CBO. A smaller average value of lack of cohesion and coupling is between object taken for AOP AspectJ projects. Remaining metrics take same trends variation between values.

We can compare calculated percentage of all 11 project using matrices and determine difference of both AspectJ and Java implementation. 07 (64%) DIT metrics have higher value through Java implementation. 04 (36%) DIT metrics have higher value through AspectJ implementation. 04 (36%) LCO metrics have higher value through Java implementation. 06 (54%) LCO metrics have higher value through AspectJ implementation. 01 (10%) LCO have the same value. 07 (64%) NOC metrics have higher value through Java implementation. 04 (36%) NOC metrics have higher value through AspectJ implementation. 03 (27%) CBO metrics have higher value through Java implementation. 08 (73%) CBO metrics have higher value through AspectJ implementation. 03 (27%) WMC metrics have higher value through Java implementation. 07 (63%) WMC metrics have higher value through AspectJ implementation. 01 (10%) WMC have the same value. CBO and WMC have higher value as compared to NOC and DIT using AspectJ implementation. According to this, coupling is high in AspectJ implementation due to high value of WMC and CBO than the Java implementation. Limited numbers of projects are implemented in this paper, so we can't generalize the experimental results. Experimental results improve the validation of metrics for Aspect Oriented Programming and impact on quality of metrics. QEMSR model supports to take decision or choose the best quality for the applications software.

VII. Performance Analysis of Qemsr Model using AHP

In this paper, we used two approaches to appraise the AOP and its impact on quality.

1) Qualitative evaluation of Aspect-oriented programming using QEMSR model and Analytic Hierarchy Process technique, similar approach used by Kumar A adapted in this paper [18]. Developer's projects used to determine impact of quality using Aspect-oriented programming (AspectJ) and Object-oriented programming (Java).

2) Describe performance evaluation of QEMSR model using Analytic Hierarchy Process (AHP) with existing model Aspect-oriented Software Quality (AOSQ) model and Proposed Aspect-oriented Software Quality Model.

Saaty proposed Analytic Hierarchy Process technique uses the pair wise matrix to analyze ambiguity in multi-criterion decision-making problems. In this paper, n elements have main characteristics such as mC_1, mC_2,mC_3……mC_n considered, which have compared related weight of mC_i with respect to mC_i denoted as a_{ij}. A square matrix A= $[a_{ij}]$ of order n as given in equation (1).

$$A=[a_{ij}]=\begin{array}{c} \\ mC_1 \\ .. \\ .. \\ .. \\ mC_n \end{array}\begin{array}{c} \begin{array}{ccccccc} mC_1 & mC_2 & .. & .. & .. & mC_n \end{array} \\ \begin{pmatrix} 1 & a & .. & .. & .. & a_n \\ 1/a_{12} & 1 & .. & .. & \dots & a_{2n} \\ .. & .. & .. & .. & .. & .. \\ .. & .. & .. & .. & .. & .. \\ .. & .. & .. & .. & .. & .. \\ 1/a_{1n} & 1/a_{2n} & .. & .. & .. & n \end{pmatrix} \end{array} \quad (1)$$

Where $a_{ij} = 1/a_{ij}$, for i is not equal to j and $a_{ij} = 1$ for all i.

Matrix is said to be reciprocal metric.

$$A.\boldsymbol{\omega} = \lambda_{max}.\boldsymbol{\omega}\ ,\ \lambda_{max} \geq n \quad (2)$$

Matrix involving human decision making, decision are inconsistent to a lesser or greater degree, in such a case find vector $\boldsymbol{\omega}$ satisfy the equation (2).

Here $\boldsymbol{\omega}$ is Eigen Vector and λ_{max} define Eigen value. The dissimilarity between λ_{max} and n if any is an indicator of inconsistency of decision. Saaty (1980) describe a consistency Index (CI) and Consistency Ratio (CR) to validate the consistency of the comparison matrix. Following equation is defined for validation:-

$$\text{Consistency Index (CI)} = (\lambda_{max} - l) / (n-1) \quad (5)$$

$$\text{Consistency Ratio (CR)} = CI / RI \quad (4)$$

Here RI is the average consistency Index over several random entries of same order reciprocal matrix. Saaty (1980) suggested that if the Consistency Ratio exceeds 0.1, set of decision or judgment may be too inconsistent to be reliable. In that condition, a new comparison matrix is required to prepare until Consistency Ratio (CR) is less than equal to 0.1.

In this sequence to determine the sub-characteristics and characteristic for software in Aspect-oriented, we manage a survey from programmer's expert or software developers working in industry and academic experts who have completed their projects and worked in AOP domain. We can identify the weight value of characteristics and sub-characteristics. A table is used to fill the pair wise relative weight value of eight characteristics from mC_1 to mC_6. The mean of all gathered samples of pair wise relative weight are given in square matrix A = $[a_{ij}]$ of order eight in equation, which is derived using equation(1) to apply Analytic Hierarchy Process. We have calculated Eigen vector and Eigen value to find the corresponding weight of mC_1, mC_2, mC_3, mC_4, mC_5, mC_6 and CR. We also create a reciprocal matrix after that to calculate Eigen value and Eigen vector for CR and CI.

We assign value it to a square matrix taken from survey. We also assign pair wise relative weight value to all six characteristics using equation (1). Further step to calculate Eigen value and Eigen vector of get corresponding weights and CR. We calculate Eigen vector to multiply all the entries in every row of matrix A and take n^{th} root (i.e. 6^{th} root) of the product helps in getting Eigen vector. Sum of the n^{th} root and used to normalize the Eigen vector element.

$A=[a_{ij}]=$

$$\begin{bmatrix} 1 & 2.00 & 4.00 & 2.00 & 3.00 & 7.00 \\ 0.50 & 1 & 0.25 & 1.00 & 1.00 & 4.00 \\ 0.25 & 0.33 & 3 & 2.00 & 2.00 & 4.00 \\ 0.50 & 0.33 & 0.5 & 0.33 & 0.05 & 1.00 \\ 0.33 & 2.00 & 1.0 & 2.00 & 1 & 3.00 \\ 0.14 & 0.25 & 0.25 & 0.33 & 0.3 & 1 \end{bmatrix} \quad (5)$$

Table VI shows all calculations and clearly show that A_n. We calculate A. ω and multiply the matrix (A_1 to A_6) from Eigen vector. Calculation of first row in Table V shown below:

(1* 0.3499) + (2 * 0.1069) + (4 * 0.2189) + (2 * 0.0695) + (3 * 0.144) + (7 * 0.0342) = 2.2497.

The values for remaining five rows are calculated similarly. As per equation (2), $\lambda_{max} \geq 6$, to determine product of A.**ω** Eigen value also determined by using $\lambda_{max=}$ (A. ω/ ω). All values are greater than six which satisfy the condition $\lambda_{max} \geq n$ we calculate Consistency Index using equation (3):

CI = (6.46792- 6) / (6-1) = 0.093584

After that we calculated CR for set of judgment using CI for considered samples. RI value can be taken from Saaty a scale that is 1.24[22].

CR = (0.093584 / 1.24) = 0.07547

The calculated value of Consistency Ratio (CR) is 0.1 which indicates estimate is acceptable. The assessment of overall quality of any AOP projects evaluated using below mentioned formula:-

AO Project Quality =

$$\sum_{i=0}^{n} \text{Comparative value of Sub characteristics } (SCi) * \text{weight value of SCi}$$

Where n is the number of sub-characteristics, SCi is sub-characteristic *i.* We are determining quality of our QEMSR model and existing Aspect-oriented Software Quality (AOSQ) model and existing Proposed Aspect-oriented Software Quality Model (PAOSQMO) as shown in Table VII. The overall quality of three models QEMSR, AOSQ and PAOSQMO are 0.62552223, 0.5283693, 0.505815. According to this, QEMSR model is best in form of quality in same characteristics and sub-characteristics. This calculation shows that overall quality of QEMSR is defined positive impact on software quality. This paper also extends the methodology adapted by Kumar A and based on random choice and decision of experts on AOP technology. Fig. 4 shows the analysis of quality values of all internal characteristics of QEMSR, AOSQ and PAOSQMO model graphically.

TABLE VI. EIGEN VALUES AND EIGEN VECTORS FOR MAIN CHARACTERISTICS

	mC_1	mC_2	mC_3	mC_4	mC_5	mC_6	Eigen Vector (ω)	A. ω	$\lambda_{max=}$ A. ω/ ω
mC_1	1	2	4	2	3	7	0.3499	2.2497	6.4295513
mC_2	0.5	1	0.25	1	1	4	0.1069	0.686875	6.425397568
mC_3	0.25	0.33	3	2	2	4	0.2189	1.343252	6.136372773
mC_4	0.5	0.33	0.5	0.33	0.5	1	0.0695	0.448812	6.457726619
mC_5	0.33	2	1	2	1	3	0.144	0.933767	6.484493056
mC_6	0.14	0.25	0.25	0.33	0.3	1	0.0342	0.235091	6.874005848
							1.00	Mean = 6.467924527	

TABLE VII. PERFORMANCE EVALUATION OF QUALITY OF QEMSR, AOSQ, PAOSQMO

Eigen vector for Main-characteristics	Eigen vector for Sub-characteristics	Weight for Sub-characteristics of QEMSR	Weight for Sub-characteristics of AOSQ	Weight for Sub-characteristics of PAOSQMO	Quality value of QEMSR	Quality value of AOSQ	Quality value of PAOSQMO
0.3499	0.2185	0.321	0.179	0.0137	0.0701385	0.0391115	0.001375
	0.2444	0.112	0.088	0.0053	0.0273728	0.009856	0.0048
	0.3464	0.05	0.05	0.0021	0.01732	0.0025	0.00016
	0.1442	0.132	0.148	0.0304	0.0190344	0.019536	0.00274
	0.0465	0.131	0.169	0.0046	0.0060915	0.022139	0.00046
0.1069	0.2071	0.164	0.236	0.0084	0.0339644	0.038704	0.00075
	0.2929	0.15	0.15	0.0147	0.043935	0.0225	0.00147
	0.2929	0.132	0.168	0.1279	0.0386628	0.022176	0.01023

	0.2071	0.164	0.136	0.0254	0.0339644	0.022304	0.0254
0.2189	0.0849	0.166	0.134	0.0818	0.0140934	0.022244	0.00736
	0.1399	0.154	0.146	0.0703	0.0215446	0.022484	0.0703
	0.1607	0.05	0.05	0.0009	0.008035	0.0025	0.007
	0.1742	0.145	0.155	0.0135	0.025259	0.022475	0.0135
	0.1607	0.054	0.16	0.017	0.0086778	0.00864	0.0017
	0.1399	0.118	0.182	0.0028	0.0165082	0.021476	0.0003
	0.1399	0.151	0.149	0.0131	0.0211249	0.022499	0.0092
0.0695	0.3333	0.154	0.146	0.01	0.0513282	0.022484	0.0008
	0.3333	0.165	0.135	0.01475	0.0549945	0.022275	0.01475
	0.3333	0.034	0.16	0.0249	0.0113322	0.00544	0.0224
0.144	0.0633	0.0567	0.033	0.0253	0.00358911	0.0018711	0.0228
	0.1371	0.0762	0.138	0.1197	0.01044702	0.0105156	0.0083
	0.1514	0.0345	0.155	0.0846	0.0052233	0.0053475	0.0084
	0.1604	0.0651	0.149	0.0356	0.01044204	0.0096999	0.0032
	0.1604	0.0234	0.166	0.0089	0.00375336	0.038844	0.081
	0.1671	0.0765	0.135	0.0039	0.01278315	0.0103275	0.0028
	0.1604	0.0321	0.1679	0.0545	0.00514884	0.0538959	0.0491
0.0342	0.1778	0.0612	0.1388	0.0094	0.01088136	0.00849456	0.0113
	0.2346	0.05	0.05	0.0095	0.01173	0.0025	0.086
	0.2789	0.0532	0.1648	0.0477	0.01483748	0.00876736	0.0334
	0.3087	0.0431	0.1569	0.0546	0.01330497	0.00676239	0.00482
				Total	0.62552223	0.5283693	0.505815

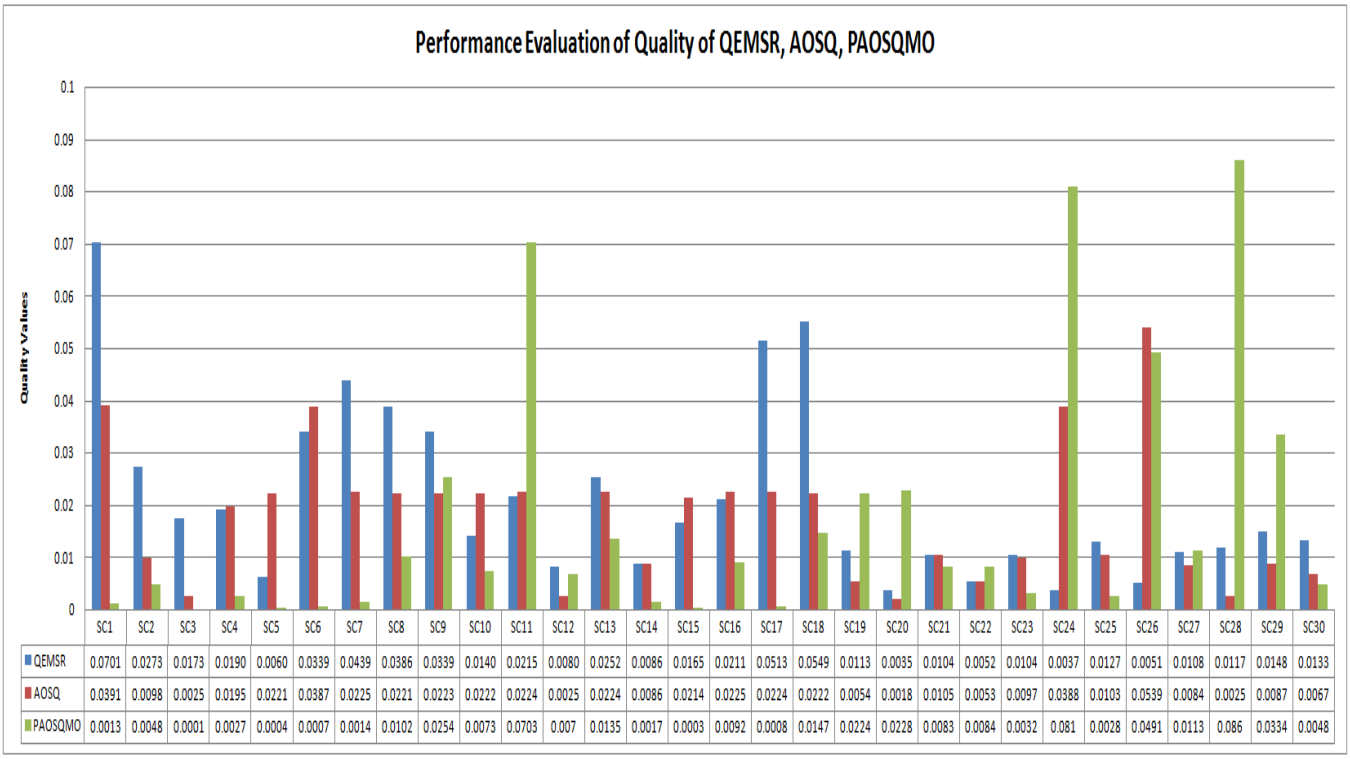


Fig. 4. Performance Evaluation of Quality of QEMSR, AOSQ, PAOSQMO.

VIII. Conclusion and Future Research

In AOP, AspectJ is a popular language which provides a support to the software developers to achieve improved quality. AOP is a standard that is trusted for quality improvement. AOP quality measurement has been trusted by evaluation of experimental results using a new QEMSR method and set of metrics for reusability and its sub characteristics. The set of AOP metrics (Coupling, Cohesion, size metrics such as DIT, NOC, CBO, LCOM, WMC, RFC) have authorized to support AspectJ and Java and an authentication of these existing metrics for quality assessment instead of new metrics proposed for AOP. Comparisons of projects are not industrial projects. Nevertheless, this paper provides the evaluation of quality and methodology of comparison as a single unit.

For future research perspective, to validate the quality metrics for large and more complex (commercial) system empirical study require in AOP research. Experimentation on large industrial projects for this domain is very difficult. This paper assessment provides some intuition about AOP and its quality which can't be generalized and it needs supplementary study. The focus of future research is on native programming languages, which is extension of AOP.

References

[1] Pankaj Kumar, "Aspect oriented software quality model: The AOSQ model", Advanced Computing in International Journal, Vol. 3, No.2, 2012.

[2] S. N. Chishti and S. K. Singh, "Exploring the quality improvement in small scale project using aspect oriented design", International Journal of Recent Technology and Engineering, Vol.8, issue 2, 2019.

[3] Vinobha A. and Senthil Valan S, "Evaluation of reusability in aspect oriented software using inheritance metrics", IEEE International Conference on Advanced Communication Control and Computing Technologies (ICACCCT), 2014.

[4] Djamel Meslati and Soumeya Debboub, "Quantitative and qualitative evaluation of aspectJ, Jboss AOP and CaesarJ using Gang-of-Four design patterns", International Journal of Software Engineering and its Applications,Vol.7,No. 6, 2013.

[5] Ghareb M.I. and Gary Allen," Identifying similar pattern of potential aspect oriented functionalities in software development life cycle", Journal of Theoretical and Applied Information Technology, Vol. 80, No.3, 2015.

[6] Hamed Fawareh, " Software quality model for maintainance software purposes", International Journal of Engineering Research and Technology, Vol. 13, No.1, 158-162, 2020.

[7] S. Dixit, S. K. Singh,"Performance of aspect oriented software quality modeling using artificial neural network technique", International Journal of Computer Applications, Vol. 182, Issue 36, 6-10, 2019.

[8] S. K. Singh and P. Kumar, "An innovative approach to analyze object-oriented and aspect oriented applications metrics using the Java and AspectJ programming language", International Journal of Advance Research Engineering and Technology, Vol. 11, Issue 11, 149-158, 2020.

[9] K. Sirbi and P.J. Kulkarni " Design pattern Vs aspect oriented programming- A qualitative and a quantitative assessment", International Journal of Computer Science & Communication, Vol.1, No. 2, pp: 233-237, 2010.

[10] G. Kiczales, J. Irwin, J. Lamping, "Aspect oriented programming", European Conference on Object Oriented Programming, pp: 220-242, ECOOP'1997.

[11] AL-Badareen, "Software quality evaluation: user's view", International Journal of Applied Mathematics and Informatics, Issue 3, Volume 5, 2011.

[12] Al-Rawashdeh and Feras M. Al'azzeh, "Evaluation of ERP systems quality model using AHP technique", Journal of Software Engineering and Applications, 7, 225-232, 2014.

[13] Samarthyam G Ganesh and T. Sharma, "MIDAS: A design quality assessment method for industrial software", IEEE International Conference on Software Engineering, San Francisco, USA, 2013.

[14] A. Przybylek, "An empirical study on the impact of AspectJ on software evolvability", Empir Software Eng, 23, 2018-2050, 2018.

[15] R. Kumar and Dalip, "Implementation of qualitative evaluation model with real life problem using AspectJ", International Conference on Globel EnterpreneurshipTrends and Empowerment through Innovation, Accepted Springer Proceeding, 2021, in press.

[16] Ram Chatterjee and Ritika Choudhary, "Predilection of reusability over maintainablity in aspect oriented system", International Journal of Computers and Technology, Vol. 6(3), 2013.

[17] O. P. Sangwan, P. K. Singh, A. Singh and A. Pratap,"A quantative evaluation of reusability for aspect oriented software using multi-criteria decision making approach", World Applied Sciences Jouranal30(12):1966-1976, 2014.

[18] D. Gotseva and M. Pavlov,"Aspect oriented programming with AspectJ", International Journal of Computer Science Issue, Vol.9, Issue 5, No 1, 2012.

[19] R. Kumar, Dalip and M. Rai," A comparative study of AOP approaches: AspectJ, Spring AOP, Jboss AOP", Proceeding of the World Congress on Engineering and Computer Science, San Francisco, USA, 2019.

[20] Heba A. Kurdi "Review on aspect oriented programming", International Journal of Advanced Computer Science and Applications, Vol. 4, No. 9, 2013.

[21] Bharti Bisht and Parul Gandhi, "Metric approach to anticipate reusability of object oriented software systems", Turkish Journal of Computer and Mathematic Education, Vol.12, No. 6, 2021.

[22] Saaty, T.L., "The analytic hierarchy process", McGraw-Hill, New York, 1980.

[23] S. K. Singh and P. Kumar, "An extensive analysis of the characteristics of an AOSQ model using fuzzy logic model", International Journal of Advance Research and Technology", Vol.11, Issue 12, 1351-1360, 2020.

[24] Farhan M. Al Obisat, Zaid T. Alhalhouli, "Review of literature on software quality", World of Computer Science and Information Technology Journal, Vol. 8, No. 5, 32-42, 2018.

[25] Shashank Joshi and GeetaBagade, "Exploring AspectJ refactoring", International Journal of Computer Applications, 2016.

[26] H. Bindu, Sk. RiazurRaheman and Amiya Kumar Rath, "Dynamic slice of aspect- oriented program: A comparative study"IJRITCC, Vol.2, Issue 2, pp: 249-259, 2014.

[27] K. Chitra and G. Maheswari, "Enhancing reusability and measuring performance merits of software component using CK metrics", International Journal of Innovative Technology and Exploring Engineering, Vol.8, 2019.

[28] Pankaj Kumar and S.K. Singh, "A comprehensive evaluation of aspect-oriented software quality (AOSQ) model using AHP technique", IEEE, 2nd International Conference on Advances in Computing, Communication & Automation (ICACCA), 2016.

[29] Petrus Mursanto, Dameria Christina Pasaribu, "Defining software quality rank using analytic hierarchy process and object-oriented metrics", IEEE, International Conference on Advanced Computer Science and Information System (ICACSIS), 2018.

[30] Pankaj Kumar and S.K. Singh, "A comprehensive investigation of quality of AOP based small scale projects using aspect-oriented software quality (AOSQ) model", IEEE, International Conference on Advances in Computing, Communication Control and Networking (ICACCCN), 2018.