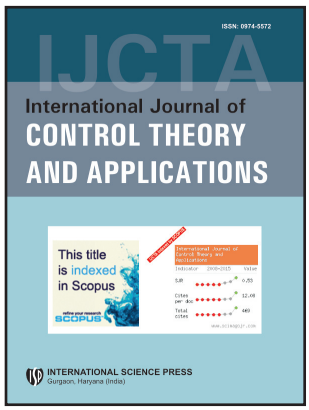

International Journal of Control Theory and Applications

ISSN : 0974-5572

Volume 10 • Number 16 • 2017

Measuring the Modifiability Metrics of UML Class Diagram using Software Quality Design Model

Sudha Rajesh[a] and A. Chandrasekar[b]

[a]*Research Scholar, Dept. of Computer Science & Engineering, Sathyabama University, OMR, Chennai, India. Email: sudharajesh2005@gmail.com*

[b]*Research Supervisor, Professor, Dept.of Computer Science & Engineering, St.Joseph's College of Engineering, OMR, Chennai, India. Email: drchandrucse@gmail.com*

Abstract: Software quality is defined as the degree for the software development process. The quality can be improved by maintaining the software design. The conflicts among stakeholders may affect the performance of the design. So it is necessary to maintain quality in early phases (design phase) rather than later phases of development. UML diagrams are broadly used to design the software, especially class diagram has the major part in object oriented software development life cycle. By measuring the metrics of class diagram leads to preeminent quality software. This paper proposed a model to measure the modifiability metrics of class diagram. This model will help the designers to evaluate a better software system. The main aim is to enhance the software design by improving the UML class diagram.

Keywords: Class diagram, Modifiability, Design Metrics, Software Quality Design Model.

1. INTRODUCTION

The first definition of quality history remembers is from Shewhart in the beginning of 20th century: There are two common aspects of quality: one of them has to do with the consideration of the quality of a thing as an objective reality independent of the existence of man. The other has to do with what we think, feel or sense as a result of the objective reality [1]. Due to continuously evolving requirements most software is modified many times after its first delivery. Current survey reports that the software costs are high for maintenance and evolution. Conflicts among stakeholders' quality attributes will leads to modification. Therefore, stakeholders anticipate a system to be designed so that it can be changes can be done easily and cost-effectively. This quality is referred as modifiability. Modifiability is defined as the changes occur in the system to increases the performance of the system. Whenever there is the need to change the properties of the system, the developers change the many features according to the demand of the developer [3].

Modifiability sometimes measured as maintenance effort, performance, in other word as maintainability, changeability, adaptability, etc, [2]. UML class diagrams is the strength of object oriented software design,

therefore the quality of a system is very important during the design phase. It consists different types of diagram to envision the state and to construct the software system. The proposed work measures some of the metrics of modifiability of a class diagram. By measuring these modifiability metrics in early stages (design phase) of software development can be enhanced to evaluate the object oriented system. This also helps the designer to make a decision, whether the software system should be altered.

2. RELATED WORKS

To estimate the quality of class diagrams the structural complexity measure is one of the most important measures. Chidamber and Kemerer are the primary examiners, proposed six metrics-Weighted Methods per Class, Response sets of Class, Lack of Consistency in Process , Combination of Classes , Intensity of Inheritance Tree , Quantity of sub classes for a class , with the help of various software excellence characteristics (e.g. effectiveness, complication, logic, recoverability, sustainability and testability) can be considered. Lulu He, Jeffrey Carver [4], if software maintenance is treated as an information-processing task, then the theory of tasks analysis can be applied. Wood proposed the task complexity model to address the lack of an adequate theoretical model for describing task variation in studies of human behavior. Three analytical dimensions describe task complexity: component, coordinate and dynamic. Total complexity is determined by all three dimensions.

Felix Bachmann, Len Bass, Robert Nord [5], Modifiability is a quality attribute of the software architecture that relates to "the cost of change and refers to the ease with which a software system can accommodate changes". It brings up four concerns: (1) Who makes the change? (2) When is the change made? (3) What can change? and (4) How is the cost of change measured? Maintainability is a subset of what we are calling modifiability. Modifiability, as a concept, is not sufficiently specific to aid designers. They must know what specific modifications are expected in order to apply the appropriate tactics that will allow for those modifications. Geeta Laxmi, Mrs. Kavita Agrawal, Dr. Rizwan Beg [6], the significance of maintainability is a key factor to software quality for producing high class software within time and budget. It developed by four models to measure analyzability, understandability, modifiability and maintainability of the class diagrams. Maintainability model measures the sustainability of the design in terms of their investigations of the classes, understandability of classes, and flexibility of the classes. These models may assist software designer to assess the design and take appropriate decision to improve the design, early in the development life cycle.

Kiranjit Kaur, Sami Anand [7], Sustainability of any software in the planning phase, assists the designer to get betterment the maintainability of software, prior to the delivery to a customer. The maintainability is a way to enhance the software system or a module that can be customized to clear the mistakes and to improve the performance. The author discuss about the multivariate linear model. These metrics help a software designer for the purpose of improving the maintainability of a class diagram in the design phase, which are helpful in feature to reduce the increasing high cost of software maintenance phase. Marcela Genero, Luis Jiménez, Mario Piattini [8] & Matinee Kiewkanya and Pornsiri Muenchaisr [9], OO conceptual models, like class diagrams are the key artifact of the early development, so focusing in their quality should contribute to the quality of the OOIS which is ultimately implemented. A set of measures for evaluating UML class diagram gives the structural complexity. From metrics values we have built prediction models for the understandability, analyzability and modifiability of class diagrams based on fuzzy classification and regression trees. The data used to build those prediction models was collected through a controlled experiment.

O. M. Alshareet [10], to calculate the relation between quality measurements and Agile Development, significantly assess the planned price of integrate class dimensions with Agile Development and to enlarge a resolution sustain scheme for measuring the intentional value of combine eminence dimensions with Agile

Development. Santonu Sarkar, Avinash C. Kak, Girish Maskeri Rama [11], the ambition is to afford metrics that distinguish huge object-oriented systems with considerations of dependency. Metrics are distinguished with the quality of modularization. N.Sankar Ram & Dr.Paul Rodrigues [12], they planned to develop a system which uses Architectural Tradeoff Analysis Method to forecast the hazards, with additional attributes and they have used artificial intelligence to expect the hazards of software architecture which supports the Stakeholders' Knowledge base. Mary Shaw, Paul Clements, D. Colquitt and J. Leaney [13, 14], these peoples says that nonfunctional features of a module is described by set of quality attributes. Software is decided with the help of combination of attributes. from this we can conclude that the following attributes are very essential for the development of software, functionality, reliability, usability, efficiency, maintainability, and portability , also they are separated into associate quality attributes. Asad Raza, Haider Abbas, Louise Yngstrom and Ahmed Hemani [15], Software architecture is the rear end of a software system by means of heavy outline and progress about it. Software architecture study is the major action in software enlargement which is used to recognize hazards, harms and to discover whether the design will meet up its superiority with the quality attributes.

3. PROPOSED MODEL

The analysis of stakeholders' views fallout some changes in the overall system. The changes will result in modification of the design. Since the class diagram acquire an important position in design phase, it is necessary to measure the metrics of modifiability of a class diagram. A model is proposed to measure the metrics (Figure 1).

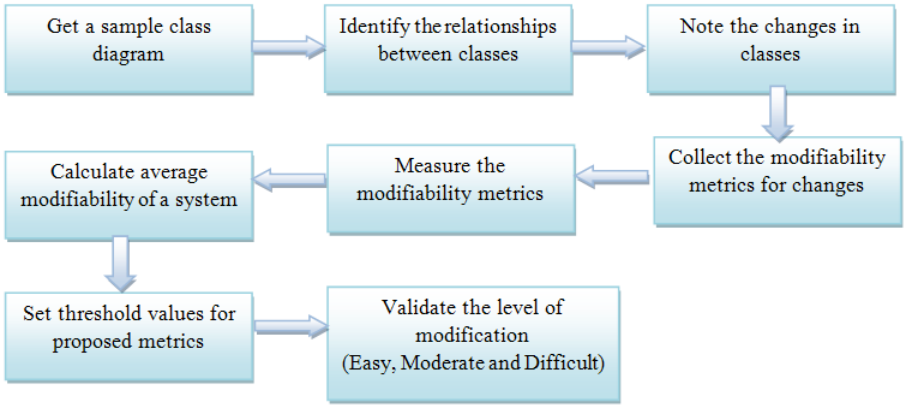

Figure 1: Software Quality Design Model

4. SELECTION OF MODIFIABILITY METRICS FOR HOSPITAL MANAGEMENT

A sample design of hospital management (Figure 2) was taken as a system to measure the modifiability metrics. If a class c is modified, then all classes dependent to it also get modified. Before measuring, it must be selected. The selection of metrics is in Table 1. The system consists of 15 classes; each class consists of attribute and methods, which termed as properties and operations respectively. The properties are very important, because the associations between the classes are defined in terms of the same properties. In the following system, some classes are having attributes and methods, some classes are not having the same. So the modifiability for those classes will be zero. Others will have their own values with modifiability.

The average modifiability of a system is calculated by the ratio of summation of all modifiability classes' to the number of classes in a system n.

$$\mathrm{AM(S)} = \frac{\sum_{i=1}^{n} \mathrm{M}(c_i)}{n} \tag{1}$$

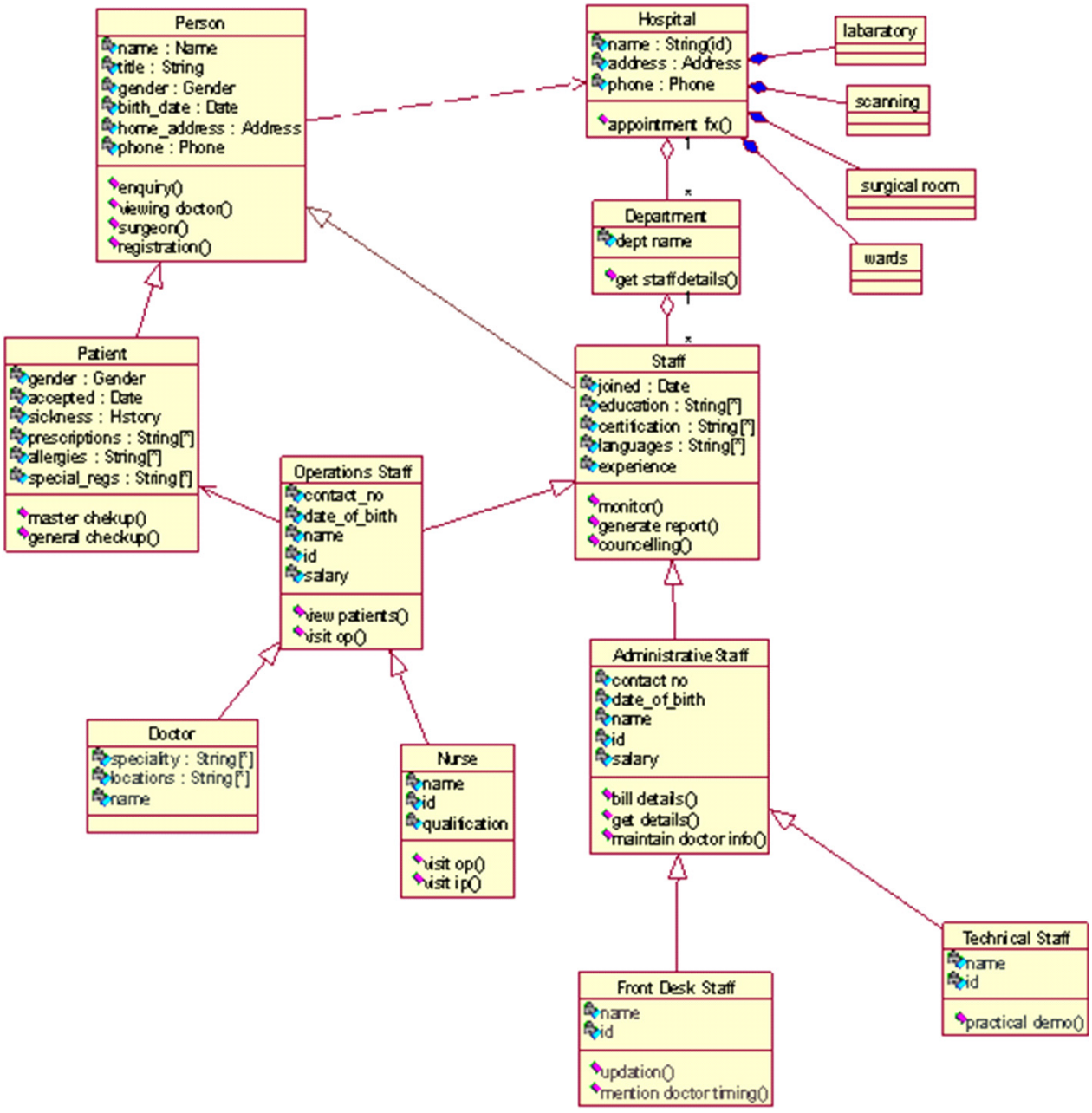

Figure 2: Hospital Management System

Table 1
Modifiability Metrics

Metrics	*Weight values for metrics*	*Definition*
C(c)	–	**C**omplexity of a class c
M(c)	–	**M**odifiability of a class c
AM(S)	–	**A**verage **M**odifiability of a system
MG(c)	6	**M**odifiability **G**eneralization of a class c
MA(c)	4	**M**odifiability **A**ggregation of a class c

MC(c)	5	**M**odifiability **C**omposition of a class c
MD(c)	1	**M**odifiability **D**ependency of a class c
MCAss(c)	2	**M**odifiability related to **C**ommon **Ass**ociation
MAssC(c)	3	**M**odifiability related to **Ass**ociation **C**lass
ASup(c)	–	Total(All) No. of super classes for a given class c
ASub (c)	–	Total(All) No. of sub classes for a given class c
ISup(c)	–	Immediate super class for a given class c
ISub(c)	–	Immediate sub class for a given class c
n	–	Total No. of classes

Modifiability of a class *c*:

The modifiability of a class C is calculated by adding all the medications of classes' associated with **G**eneralizations, **A**ggregation, **C**omposition, **D**ependency, **C**ommon **Ass**ociations, **Ass**ociation **C**lasses is as follows,

$$M(c) = C(c) + MG(c) + MA(c) + MC(c) + MD(c) + MCAss(c) + MAssC(c) \quad (2)$$

Complexity of a class c:

$$C(c) = \text{Total No. of attributes} + \text{Total No. of attributes operations in a given class } c \quad (3)$$

Modifiability **G**eneralization of a class c:

The generalization of a class in hospital management system is defined as the summation of complexity of all sub classes and the product of generality weight value as follows,

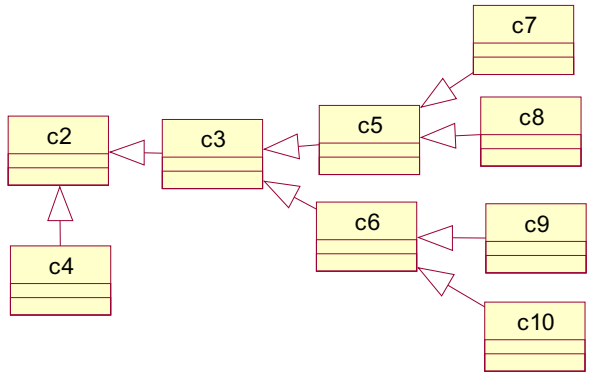

Figure 3: Generalization of a class c2

$$MG(c) = \frac{W_G C\left[ASub_G(c)\right]}{2} \quad (4)$$

Modifiability **A**ggregation of a class c MA(c):

The aggregation of a class in hospital management system is defined as the product of aggregation weight value and sum of complication of direct super class of aggregation with the complexity of all sub classes of generality in direct super class of aggregation of a class is as follows,

$$MA(c) = \frac{W_A\left[C(ASub_G(ISup_A(c)) + C(Isup_A(c)))\right]}{2} \quad (5)$$

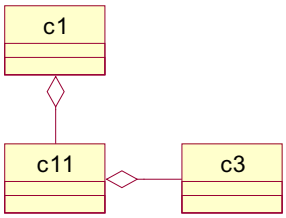

Figure 4: Aggregation of a class c11

Modifiability **C**omposition of a class c MC(c):

The composition of a class in hospital management system is defined as the product of composition weight value and sum of complexity of direct super class of composition with the complexity of all sub classes of generalization in direct super class of composition of a class is as follows,

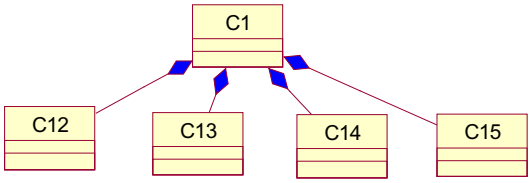

Figure 5: Composition of a class c1

$$MC(c) = \frac{W_C\left[C(ASub_G(ISup_C(c)) + C(Isup_C(c)))\right]}{2} \tag{6}$$

Modifiability Dependency of a class c MD(c):

The dependency of a class in hospital management system is defined as the product of dependence weight value and sum of complexity of immediate sub class of dependency with the complexity of all sub classes of generalization in immediate sub class of dependency of a class is as follows

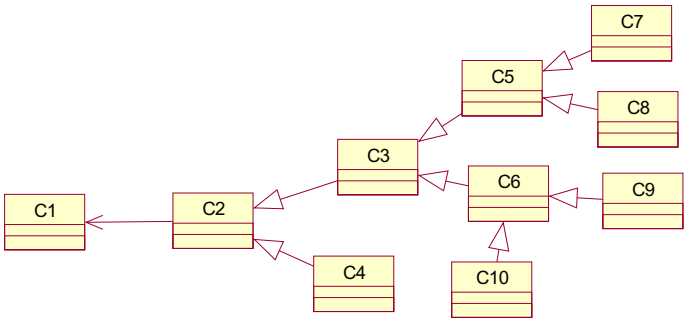

Figure 6: Dependency of a class c1

$$MD(c) = \frac{W_D\left[C(ASub_G(ISub_D(c)) + C(Isub_D(c)))\right]}{2} \tag{7}$$

Modifiability related to **C**ommon **Ass**ociation MCAss(c):

The common association of a class in hospital management system is defined as the product of common association weight value and sum of complexity of immediate sub class of common association with the complexity of all sub classes of generalization in immediate sub class of common association of a class is as follows,

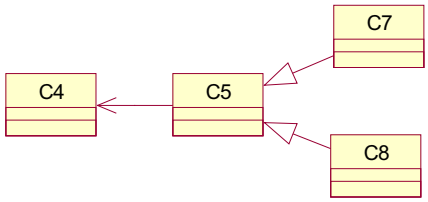

Figure 7: Common Association of a class c4

$$MCAss(c) = \frac{W_{CAss}\left[C(ASub_G(ISub_{CAss}(c)) + C(Isub_{CAss}(c)))\right]}{2} \tag{8}$$

Modifiability Related to **Ass**ociation **C**lass MAssC(c):

The association of a class in hospital management system is defined as the product of association weight value and sum of complexity of immediate sub class of association with the complexity of all sub classes of generalization in immediate sub class of association of a class is as follows,

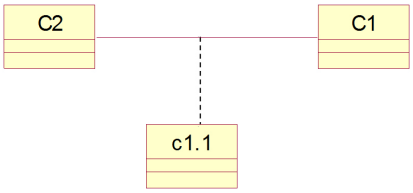

Figure 8: Association Class of a class c1.1

$$MAssC(c) = \frac{W_{AssC}\left[C(ASub_G(ISub_{AssC}(c)) + C(Isub_{AssC}(c)))\right]}{2} \tag{9}$$

5. RESULTS AND DISCUSSION

The super and sub classes of every class was determined for hospital management system is shown in [Table 2], then the average modifiability of a system is measured by using all relationships of a class is shown in [Table 3]. Even though the super and sub classes are defined for all classes, if changes are made, then the class properties will affect in many other classes. So designers and stakeholders' should take care while selecting the modifiability metrics.

Table 2
All and Immediate - Super and Sub Classes

CLASSES	*CLASS NAME*	*ASup(c)*	*ASub (c)*	*ISup(c)*	*ISub(c)*
C1	Hospital	0	0	0	0
C2	Person	0	8	0	C2, C4
C3	Staff	C2	6	C2	C5,C6
C4	Patient	C2	0	C2	0
C5	Operation staff	C2,C3	2	C3	C7, C8
C6	Administrative Staff	C2,C3	2	C3	C9, C10
C7	Doctor	C2, C3, C5	0	C5	0
C8	Nurse	C2, C3,C5	0	C5	0
C9	Front desk staff	C2, C3, C6	0	C6	0
C10	Technical staff	C2,C3, C6	0	C6	0
C11	Department	0	0	0	0
C12	Lab	0	0	0	0

CLASSES	CLASS NAME	ASup(c)	ASub (c)	ISup(c)	ISub(c)
C13	Scanning	0	0	0	0
C14	Surgery room	0	0	0	0
C15	Wards	0	0	0	0

If the modifications are more, then we cannot get a quality system. Immediate super classes are having more impact on the changeable property of a class. Similarly immediate sub classes are also having more collision. If the impacts get enlarged during the development life cycles, then it makes the entire system to be in chaos. The results are given based on the associations between the classes with the help of **G**eneralizations, **A**ggregation, **C**omposition, **D**ependency, **C**ommon **Ass**ociations, and **A**ssociation **C**lasses.

Table 3
Average Modifiability of Hospital Management System

CLASSES	C(c)	MG(c)	MA(c)	MC(c)	MD(c)	MCAss(c)	MAssC(c)	M(c)
C1	4	0	0	10	27	30	66	137
C2	10	132	0	0	22	0	0	164
C3	5	93	0	0	15.5	0	0	113.5
C4	8	0	0	0	0	15	0	23
C5	7	24	0	0	4	0	0	35
C6	8	24	0	0	4	0	0	36
C7	3	0	0	0	0	0	0	3
C8	5	0	0	0	0	0	0	5
C9	5	0	0	0	0	0	0	5
C10	3	0	0	0	0	0	0	3
C11	2	0	4	0	0	0	0	6
c12	0	0	0	0	0	0	0	0
c13	0	0	0	0	0	0	0	0
c14	0	0	0	0	0	0	0	0
c15	0	0	0	0	0	0	0	0
		Average Modifiability of a System (AM(S))						35.37

The calculation for the average modifiability of a system is exposed in Table 3 and the values for the same are revealed with the help of C(c), M(c), AM(S), MG(c), MA(c), MC(c), MD(c), MCAss(c), MAssC(c), ASup(c), ASub (c), ISup(c), ISub(c) and shown in Figure 9. The threshold value for AM(S) is set as Easy, Moderate & Difficult to modify, and the range for AM(S) is made known in Table 4.

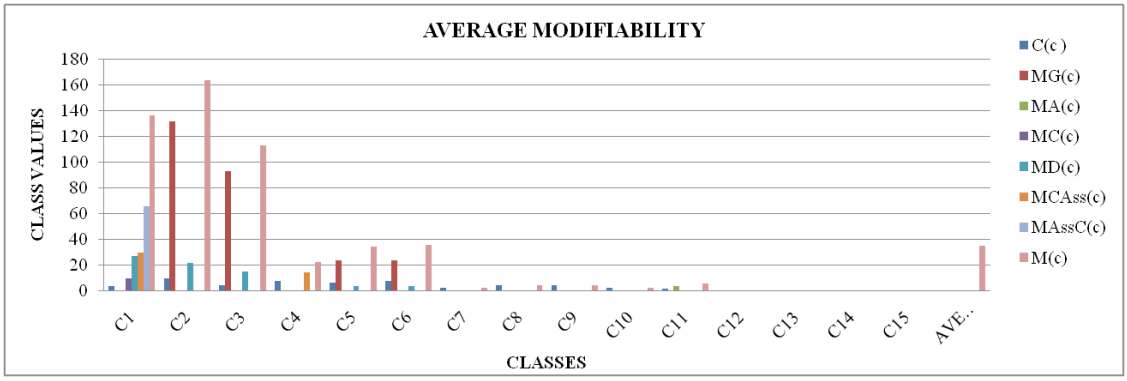

Figure 9: Average Modifiability of the System for All Classes

Table 4
Modifiability Level

Range of AM(S)	*Modifiability level*
AM(S) < 50	Easy to modify
> 50 AM(S) < 75	Moderate to modify
AM(S) > 75	Difficult to modify

6. CONCLUSION

Software quality is the key element of software development life cycle. The quality can be maintained by the notification of the changes due to stakeholders' concerns. So a model is proposed to measure the metrics of changes in terms of modifiability. In this paper average modifiability of hospital management system is calculated and threshold value is defined as [AM(S) < 50] AM(s) = 35.37, so it is easy to do modification. If the modifications are less in the designing, it will help the designer to evaluate the preeminent software system. The result shows that the modifications can be done up to 50% to 75% of attributes. The hospital system is having only 15 classes, with these classes the AM(s) is around 35%. If the numbers of associations of classes are increases and stakeholders' opinion changes, then the modifications will be difficult. In future this average modification may be reduced by minimizing the classes and their metrics. This work also brings out that, if any modification is done in any class it will affect the classes related to it

REFERENCES

[1] Mrs. Sudha Rajesh, Dr.A. Chandrasekar, "Analyzing Stakeholders' Concerns and Estimation of Quality Attributes", National Conference on Frontiers in Applied Science and Computer Technology, NIT, March 2015.

[2] ISO/IEC.2000. Information technology -Software product quality Part1: Quality model. ISO/IEC FDIS 9126-1:2000(E).

[3] Hardeep Singh, Aseem Kumar, A novel approach to enhance the maintainability of object oriented software engineering during component based software engineering, International journal of computer science and mobile computing, pg.778 – 786, Vol. 3, issue. 3, March 2014.

[4] Lulu He, Jeffrey Carver, "Modifiability Measurement from a Task Complexity Perspective: A Feasibility Study *",* Third International Symposium on Empirical Software Engineering and Measurement, 978-1-4244-4841-8/09, 2009 IEEE.

[5] Felix Bachmann, Len Bass, Robert Nord, "Modifiability Tactics" Software Architecture Technology Initiative, CMU/ SEI-TR-002, September 2007.

[6] Geeta Laxmi, Mrs. Kavita Agrawal, Dr. Rizwan Beg, "Maintainability Measurement Model of Object Oriented Design", International Journal of Advanced Research in Computer Science and Software Engineering, ISSN: 2277 128X, Volume 4, Issue 11, November 2014.

[7] Kiranjit Kaur, Sami Anand, "A Maintainability Estimation Model and Metrics for Object-Oriented Design (MOOD)", and ISSN: 2278 – 1323: International Journal of Advanced Research in Computer Engineering & Technology, Volume 2, No 5, May 2013.

[8] Marcela Genero, Luis Jiménez, Mario Piattini, "A Prediction Model for OO Information System Quality Based on Early Indicators", University of Castilla-La Mancha.

[9] Matinee Kiewkanya and Pornsiri Muenchaisr, "Constructing Modifiability Metrics by Considering Different Relationships", Chiang Mai J. Sci. 2011; 82-98, www. Science. Cmu. ac. th / journal-science.

[10] O. M. Alshareet,"An Empirical Study to Develop a Decision Support System (DSS) for Measuring the Impact of Quality Measurements over Agile Software Development (ASD)", Indian Journal of Science and Technology, Vol 8(15), 70774, July 2015.

[11] Santonu Sarkar, Avinash C. Kak, Girish Maskeri Rama, "Metrics for Measuring the Quality of Modularization of Large-Scale Object-Oriented Software", IEEE Transactions on Software Engineering, Vol. 34, No. 5, September/October 2008.

[12] N.Sankar Ram, Dr.Paul Rodrigues,"Enhanced Intelligent Risk Divination Using Added Quality Attributes Injected ATAM and Design Patterns", International Journal of Computer Science and Security, Volume-2 Issue-2.

[13] Mary Shaw and Paul Clements, "The Golden Age of Software Architecture" IEEE Software March/April 2006.

[14] D.Colquitt and J. Leaney "Expanding the view on Complexity within Architecture Trade-off Analysis Method", Proceedings of the 14th Annual IEEE International Conference and Workshops on the Engineering of Computer-Based Systems.

[15] Asad Raza, Haider Abbas, Louise Yngstrom and Ahmed Hemani, "Security Characterization for Evaluation of Software Architectures using ATAM", 2009 IEEE.