ISSN No. 0976-5697

Volume 3, No. 6, Nov. 2012 (Special Issue)

International Journal of Advanced Research in Computer Science

RESEARCH PAPER

Available Online at www.ijarcs.info

Performance Computation Metrics for Object-Oriented Software Systems

Vipin Saxena*
Department of Computer Science, Babasaheb Bhimrao Ambedkar University (A Central University)
Vidya Vihar, Rae Bareli Road, Lucknow (U.P.) 226025, India
vsax1@rediffmail.com

Santosh Kumar
Department of Computer Science, Babasaheb Bhimrao Ambedkar University (A Central University)
Vidya Vihar, Rae Bareli Road, Lucknow (U.P.) 226025, India
sant7783@hotmail.com

***Abstract*—** Due to the evolution of Graphical User Interface (GUI) Technology, many of the software industries are shifting their old traditional software systems towards the object-oriented technology as in comparison; it is much faster than the old structured based technology. A study of the effective software design of object oriented technology is important as one can propose many of similar kinds of comparative designs for one small problem. The present work explores the possibilities of various kinds of metrics used for judging the performance of the object oriented software system. The different proposed metrics have been implemented on a real domain of Life Insurance Corporation (LIC) of India. For this purpose, a well known Unified Modelling Language is used with UML class and Sequence diagrams.

Keywords: UML, Object-Orientation, Computation Metric, Class Diagram and Sequence Diagram.

I. INTRODUCTION

In the current days, the Unified Modelling Language i.e. UML becomes very popular modelling language and widely accepted by the software professionals for modelling the huge and complex object-oriented systems. UML has a verity of modelling tools that helps in making blueprint, documenting and visualizing the architecture of any software system. Therefore, a UML is a standard modelling language for object-oriented systems and becomes one part of the software development also. The object oriented approach is commonly applied by the software developers and researchers in development field.

The performance computation metrics are very useful in computing the performance of any model; there are many methods available for computing the performance and validation of any object-oriented model. Through these metrics, the optimal solution of any object oriented problem is easily computed. Let us first describe some of the important references related to the said approach. Meghan et al. [1] have defined new features coupling metrics based on structural and textual source code information and extended the unified framework for coupling measurement to include these new metrics. Dubey and Rana [2] have presented a comparative analysis of various usability models and metrics. It then analysed the relationship between usability and object-oriented metrics. Turnu et al. [3] have presented the fractal dimension metric and its use to access object oriented software quality.

Johari and Kaur [4] have presented the results of empirical study which was conducted using open source software, JHotDraw 7.5.1 and computed the object oriented metrics, proposed by Chidamber and Kemmerer, and performed bug-class mapping for the software under study. Dallal and Briand [5, 6] have proposed a High-Level Design (HLD) class cohesion metric, which is based on realistic assumptions, complies with expected mathematical properties, and can be used to automatically assess design quality at early stages using UML diagrams and also proposed a formula that precisely measures the degree of interaction between each pair of methods, and we use it as a basis to introduce a low-level design class cohesion metric (LSCC). Dallal [7] has proposed a new class cohesion metric that has higher discriminative power than any of the existing cohesion metrics. Gandhi and Bhatia [8] have proposed Message received coupling (MRC) and Degree of Coupling (DC) metrics for the automatic detection of a set of design problems along with an algorithm to apply these metrics to redesign an object oriented source code. Chowdhury and Zulkernine [9] have presented a framework to automatically predict vulnerabilities based on CCC metrics. Oh et al. [10] have proposed novel metrics to measure ontology modularity and for evaluated the cohesion and coupling based on the theory of software metrics.

In the present work, the performance computation has been performed, a real case study of Life Insurance Corporation (LIC) of India is consider for computing the performance of a model for issuing a insurance policy.

II. UML MODEL FOR ISSUING POLICY

A. UML Class Diagram:

The structural behaviour of the system is shown by the UML class diagram; it consists of different properties like association, aggregation, generalisation and inheritance that are shown in the form of subclasses. The present work shows the complete process of issuing a policy and using the mobile system as an information exchange medium is explained. The UML class model consists of major classes like Agent Customer, Mobile_System, New_Busines, Main_Branch, Plan and Bank represented in Figure 1.

The LIC class has multiple associations with all the classes. The Customer class has multiple associations with Agent, Mobile_System, Plan and New_Business classes. The Agent class shows plan and convince the Customer class for open a policy in LIC, once the Customer class convince for opening a policy in LIC class then the Customer select the plan form the Plan class. After selecting the plan the customer sends a policy proposal to New_Business class where New_Business class validate the policy proposal and sends it for issuing the proposed policy to the LIC class and reject if Customer not fulfils the

CONFERENCE PAPER
II International Conference on Issues & Challenges in Networking, Intelligence & Computing Technologies
Organized by Krishna Institute of Engineering and Technology (KIET) Ghaziabad, India

eligibility criteria. The LIC class issues a proposed policy after depositing the premium amount by the Customer class in the LIC, then the LIC class deposit the premium amount into the Bank class and sends a confirmation of issuing policy with premium information to the Mobile_System class then the customer interacts with this information. The Customer's data is stored in the LIC's database and upload online by the same software provided by the LIC of India. The status of policy can viewed online or by fetching the database on the mobile system.

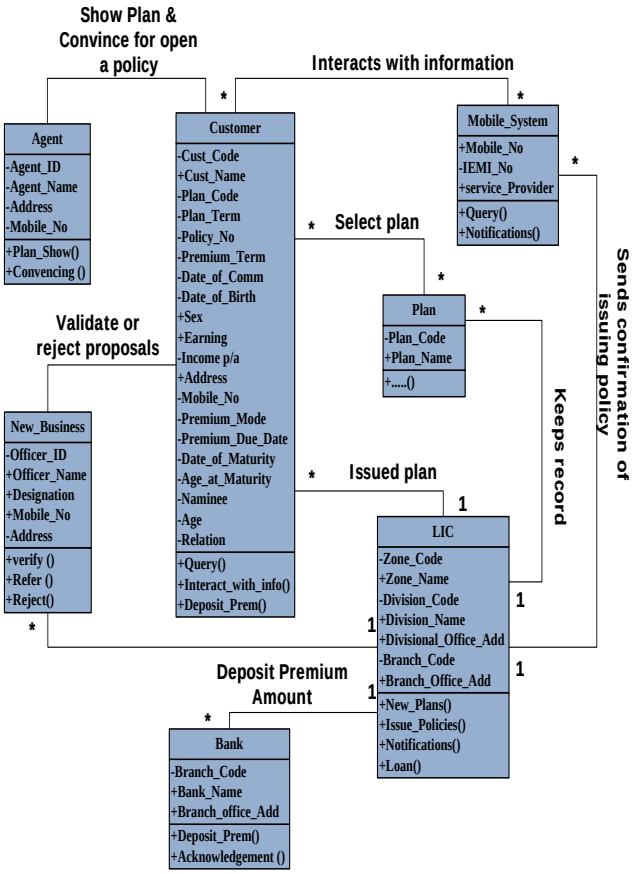


Figure. 1 UML Class model for Issuing Policy

B. UML Sequence Diagram:

The sequence diagram shows the dynamic behaviour of the system. For the above class model the sequence diagram is designed which is shown in the Figure 2. Therefore the sequence diagram shows the complete process of issuing a policy for the LIC of India. There are five major objects presented here like Agent, Customer, New_Business, LIC, Mobile_System and Bank. An arrow with message shows the communication between two objects and the vertical line shows the life line of an object. An Agent convince the Customer for opening a policy in LIC of India then Customer proposed a policy to the New_Business where New_Business Checks and validate the policy proposal; if the proposal is validated then LIC issues a policy after depositing the premium amount into the Bank and if the proposal is not validated then New_Business reject it. A confirmation information regarding policy is send by the LIC to the customer's Mobile_System. The purpose of this sequence diagram is to show that the system of issuing policy is working properly.

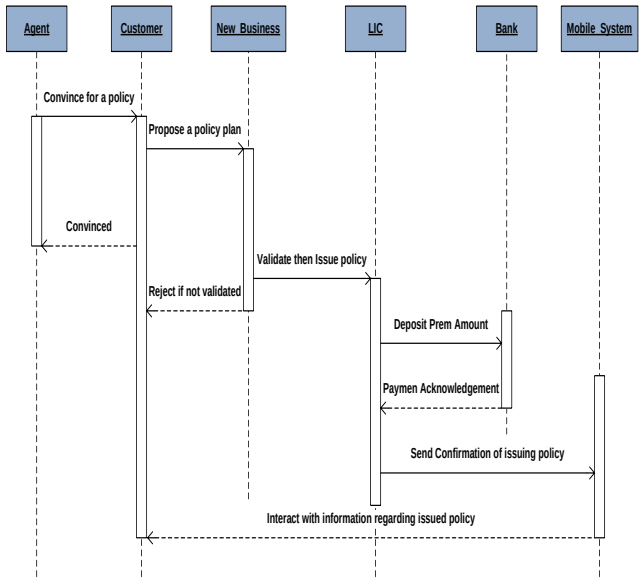


Figure. 2 UML Class model for Issuing Policy

III. EXPERIMENTAL STUDY

In this section let us first validate the UML model through the state transition diagram corresponding to the transition table, then check the dependency of classes by finding the degree of coupling and then compute the performance of the model by applying the makove chain.

A. Validation of the UML Model through FSM:

For validating the proposed model, author illustrates the state transition diagram of UML class model for issuing policy which is shown in Figure 3. The states are transformed from one state to another state, the states are shown in the rectangle and the arrow shows the corresponding action.

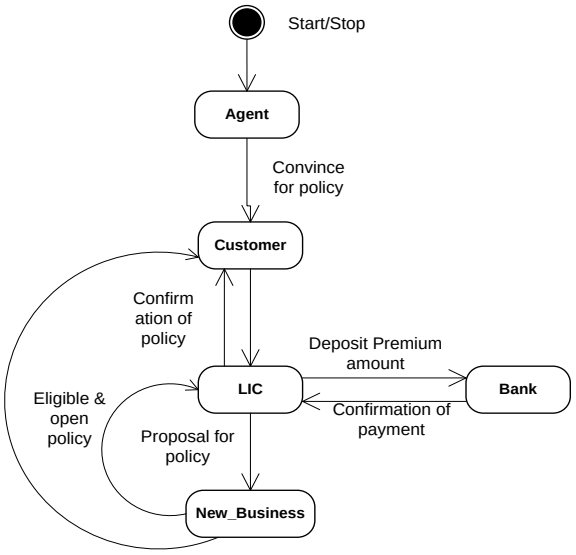


Figure. 3 State Diagram for Issuing Policy

Here, the author has assumed that an Agent is an initial state while the Customer is the finale state of opening policy, the initial state Agent is equivalent to 'q_0' and the Agent go to the Customer for convincing for opening the policy; after convincing the Customer the state is change say 'a' to the "Customer" state say 'q_1'. After that the Customer go to the LIC, the state is change say 'b' to "LIC" state say 'q_2', then

CONFERENCE PAPER
II International Conference on Issues & Challenges in Networking, Intelligence & Computing Technologies
Organized by Krishna Institute of Engineering and Technology (KIET) Ghaziabad, India

the Customer go to the New_Business and propose a policy the state is change 'c' to the "New_Business" and system enters into new state say 'q_3'. Here the New_Business validate the policy proposal and go back to the previous state 'q_2' for opening the policy with message eligible and open policy say 'c' and go back to the final state 'q_1' for rejection with message not eligible say 'c''. The LIC issues a policy after depositing the premium amount and system enters into the state say'd' to Bank say 'q_4'. After depositing the premium amount to the Bank, the Bank confirms the payment say 'd'' and system go to the state 'q_2'; as getting the payment confirmation from the Bank , the LIC sends confirmation regarding policy and send the policy bond with premium receipt to the Customer say 'e' and system go to the final state say 'q_1'. Therefore the Finite State Machine (FSM) for issuing policy with the help of these equivalent states is shown in the Figure 4.

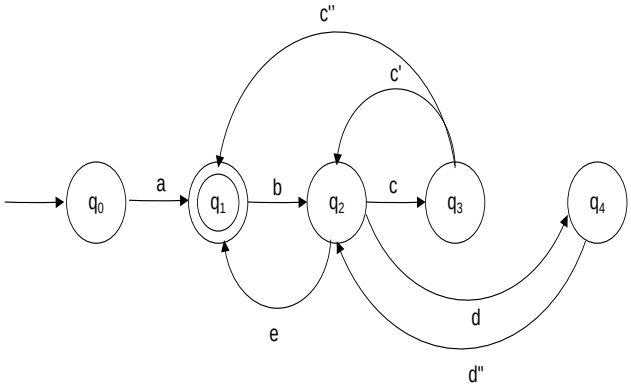


Figure. 4 Finite State machine for Issuing Policy

From the above figure the state transformation is assumed on the basis of {a, b, c,…… e} events. These inputs are considered as terminals for the finite state machine shown in the figure and the set of states are assumed to be non-terminals for the shown machine, where q_0 is the initial and q_1 is the finale state of the above finite state machine of issuing policy in LIC of India. The several productions can be introduced for the shown finite state machine and the corresponding transition table is shown in the table1:

$q_0 \rightarrow a\ *q_1$
$*q_1 \rightarrow bq_2$
$q_2 \rightarrow cq_3$
$q_2 \rightarrow dq_4$
$q_2 \rightarrow e*q_1$
$q_3 \rightarrow c'q_2$
$q_3 \rightarrow c''*q_1$
$q_4 \rightarrow eq_2$

Table 1: Transition Table

Input/State	a	b	c	c'	c''	d	d''	e
$\rightarrow q_0$	$*q_1$	-	-	-	-	-	-	-
$*q_1$	-	q_2	-	-	-	-	-	-
q_2	-	-	q_3	-	-	q_4	-	$*q_1$
q_3	-	-	-	q_2	$*q_1$	-	-	-
q_4	-	-	-	-	-	-	-	q_2

B. Dependency of classes:

A class is dependent to other class if the functionality of one class is affected when the changes are made in the other class. The dependency of classes is measured by finding the degree of coupling. The degree of coupling is measured by the ratio of message received coupling (MRC) to message pass coupling (MPC), where MRC is the set of number of message received by a class while MPC is the set of number of message passed by the class.

$$\text{Degree of Coupling (DC)} = \frac{MRC}{MPC}$$

The message calling graph (MCG) is design for computing the degree of coupling which is shown in Figure 5:

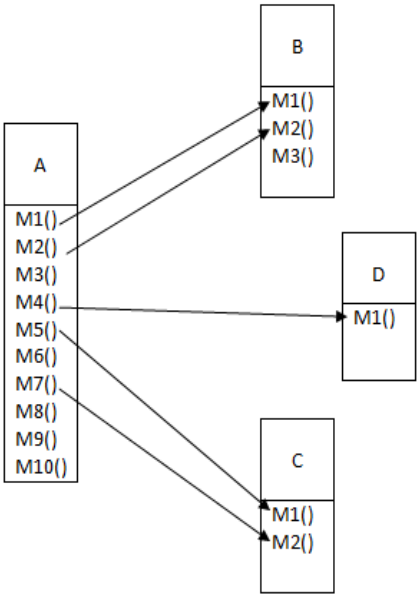


Figure. 5 Method Calling Graph

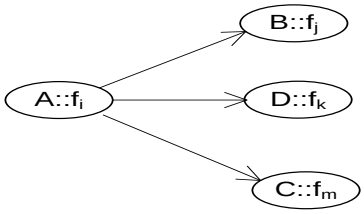


Figure: 6

The nodes of the message calling graph is in the form of $A::f_i$, $B::f_j$, $C::f_k$, and $D::f_l$ where f_i, f_j, f_k and f_l are the methods of the classes A, B, C and D respectively. The graph shows that the function of class A is called by all the three classes B, C and D.

For computation of degree of coupling, compute the values of MRC and MPC as shown in the figure 5.

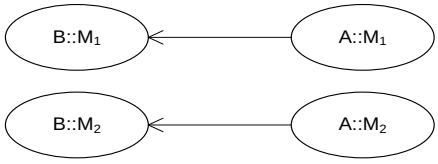
The class A has 10 methods but not call any methods from other classes, so the MRC is 0 for this class and MPC is 6. Therefore the degree of coupling is:

$$\text{Degree of Coupling (DC)} = \frac{MRC}{MPC}$$

CONFERENCE PAPER
II International Conference on Issues & Challenges in Networking, Intelligence & Computing Technologies
Organized by Krishna Institute of Engineering and Technology (KIET) Ghaziabad, India

DC = 0/6 = 0

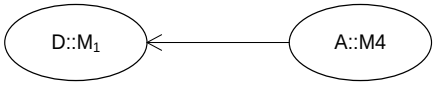
There are three methods in class B, it call the two methods of class A. the MCG for this class is shown as:



Thus the MRC for class B is 2 while MPC for it is 0, then the degree of coupling is

DC = 2/0 = ∞

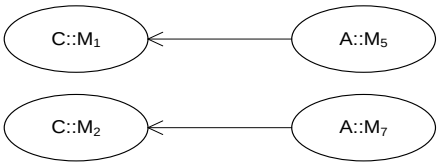
The class D has only 1 method and it calls the only one method, the MCG for this class is shown as:



Thus the MRC for class D is 1 and MPC is 0 then the degree of coupling is

DC = 1/0 = ∞

The class C has two methods, the MCG for class C is shown as:



Thus the MRC and MPC of this class is 2 and 0 respectively, therefore the degree of coupling is

DC = 2/0 = ∞

Therefore the degree of coupling of each class is shown in the table 2:

Table 2: Class level metrics

Class	Object-Oriented Matrices		
	MPC	MRC	DC
A	6	0	0/6
B	0	2	∞
C	0	2	∞
D	0	1	∞

IV. CONCLUSIONS

From the above experimental studies, it is observed that the proposed model of issuing policy in Life Insurance Corporation of India is validated through the finite state machine by producing all the necessary productions which are shown in the table 1 and observed that the designed model is reliable and having reusable, while the degree of coupling shows that the dependency of the classes of the proposed model. The class level metric shows that the classes B, C, D are strongly dependent on the class A and observed that these classes are need to redesign because the best object-oriented design is based on the low coupling and high cohesion.

V. ACKNOWLEDGEMENT

Thanks are due to University Grant Commission New Delhi, for financial support to carry out the above research work.

VI. REFERENCES

[1] Meghan Revelle, Malcom Gethers and Denys Poshyvanyk," Using Structual and textual information to capture feature coupling in object-oriented software", Empirical software Engineering, Vol. 16, issue, pp. 773-811.

[2] S. K. Dubey and A. Rana, "Usability Estimation of software system by using object-oriented metrics", ACM SIGSOFT software Engineering notes, Vol. 36, Issue 2, pp. 1-6, 2011.

[3] Ivana Turnu, Giulio Concas, Michele Marchesi and Roberto Tonelli, "The Fractal Dimension Metric and its use to access object-oriented software quality", 2nd International Workshop on Engineering Trends in Software Metrics, pp. 69-74, 2011

[4] K. Johari and A. Kaur, "Validation of Object oriented Metrics using open source software system: an empirical study", ACM SIGSOFT Software Engineering Notes, Vol. 37, Issue 1, pp. 1-1-4, 2012.

[5] Jehad al Dallal and Lionel C. Briand, "An Object-Oriented high-level design-based class cohesion metric", Information and software technology, vol.52, Issue12, pp. 1346-1361, 2010.

[6] Jehad al Dallal and Lionel C. Briand , " A Precise Method-Method Interaction-Based Cohesion Metric for object-oriented classes ", ACM Transaction for Software Engineering and Methodology (TOSEM), Vol. 21, Issue 2, Article No. 8, 2012.

[7] Jehad al Dallal, "Fault Prediction and the Discriminative powers of connectivity-based object-oriented class cohesion metrics ", Information and Software Technology, Vol. 54, Issue 4, pp. 396-416, April 2012.

[8] P. Gandhi and P. K. Bhatia, " Optimization of Object-Oriented design using Coupling Metrics", International Journal of Computer Applications, vol. 27, no. 10, pp. 41-44, August 2011.

[9] I. Chowdhury and M. Zulkernine, "Using complexity, coupling, and cohesion metrics as early indicators of vulnerabilities", Journal of Systems Architecture: the EUROMICRO Journal, Elsevier North-Holland, Inc. New York, NY, USA, Vol. 57 Issue 3, March, 2011.

[10] S. Oh, H. Y. Yeom and J. Ahn, "Cohesion and coupling metrics for ontology modules", Information Technology and Management, Kluwer Academic Publishers Hingham, MA, USA, Vol. 12 Issue 2, June 2011.

[11] Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specification, PhD thesis submitted to university of Malaya.

CONFERENCE PAPER
II International Conference on Issues & Challenges in Networking, Intelligence & Computing Technologies
Organized by Krishna Institute of Engineering and Technology (KIET) Ghaziabad, India