Designing and implementing a tool for measuring cohesion and coupling of Object-Oriented Systems

Atica M. Altaie

Department of Software, College of Computer Science and Mathematics, University of Mosul, Iraqatica_altaie@uomosul.edu.iq

Article History: Received: 14 February 2022; Revised: 3 March 2022; Accepted: 6 March 2022; Published online: 10 March 2022

Abstract

Cohesion and coupling are considered essential elements of an object-oriented system's internal quality. The class diagram represents an important part of the design phase of the development life cycle of the software. Measurement of metrics early in the design phase will lead to enhancing the quality of design and source code and minimize the efforts required to build software. This tool calculates coupling and cohesion for class diagrams using design metrics. The metrics of the tool gather the data and information after analyzing the XMI file created by the UML tools. Then, we specify the cohesion of the class and the relationships of coupling via matching tokens of a specific class with tokens of other classes. The proposed methodology extracts tokens based on metrics for cohesion and coupling. Finally, the results lead to a suitableassessment of the software quality in terms of its cohesion and coupling measures. The values of DCH and DC metrics range from 0 to 1, the high cohesion and low coupling are better for the measure of maintainability, testability, readability, reliability, and reuse.

1. Introduction

The aims of the software engineering phases are to create products which have all of the functions that a user requires while keeping costs and timelines within the originally specified parameters. Furthermore, the software of these products is simpler to comprehend and maintain. It is required to travel through the many stages of the software life cycle in order to obtain this high-quality software[1][2]. Object-oriented programming is a method for creating reusable, modular software systems. Although discussions about object-oriented technology are frequently embroiled in debates over which language is better, the object-oriented approach is really a modeling technique. Increased knowledge, ease of maintenance, and ease are the goals of object-oriented programming [3][4].

Cohesion and coupling are considered essential elements of an object-oriented system's internal quality (OOS). Class cohesion refers to the degree of class members belong to each other (attributes and methods). On the other side, class coupling refers to the degree to which a class in software is coupled to other classes. For faster and easier systems development, smart software engineersshould optimize cohesion (cohesion supports encapsulation) and decrease coupling (coupling slows encapsulation) [5][6]. As a result, metrics for controlling Software coupling and cohesion have become essential. A Software's

proclivity for coupling results in a high level of complexity. Low cohesion and strong coupling can be used to design software that is reliable, maintainable, and extendible. Many scholars have contributed to the design of a range of metrics for managing Software coupling and cohesion in order to compute Software cohesion and coupling[7][8].Calculating the cohesion and coupling effect on testability, reliability, maintainability, and reusability [9][10].

The UML(Unified Modeling Language) is a graphical representation of a system model that partially represents the phases of design and implementation of software that will be built by the software development team. The word "class diagram" refers to a diagram in which the contents are classes with all characteristics and methods of a class diagram that shows the use of classes [11][12].

The design of the program code has an impact on its quality. The designers' goal is to have the best design possible at all times. During the design process, it is vital to avoid errors that may result in poor source code quality in the future [13][14].It is for this reason that we propose a tool aims to present a model that helps software engineers to calculate coupling and cohesion for class diagram using object oriented design metrics and give an indicator to software engineers to enhance their design and code.The tool is quite useful for recognizing class cohesion and coupling to reduce the cost of maintainability, testability, and reusability.

This study is organized in five sections: Section 2 analyses the previous studies correlated. Section 3 covers the background on metrics used and the tool architecture of proposed model. Section 4covers testing the model and results throughcase studies. Section 5 outlines conclusion and possible future work.

2. Previous studies

Cohesion and coupling arefeatures that affect the softwarequality .These metrics can be computed by design andsource code of software. In [15] the metric proposed of Class Cohesion measured the amount of similarity between approaches is used which represented as a root to compute cohesion of a class. The similarity between couples of approaches isrepresented as the ratio of the total sum number of common attributes to the number of special attributes mentioned by both approaches. Cohesion is represented as the ratio of the total sum of all similarities between couples of approaches to the total number of probablecouples of approaches.

Manyapproaches have been proposed to recognize coupling of source codelike [16],[17].The authors in [16] proposed a framework for measuring coupling in softwareby means of XML. First, the program is analyzed and saved in XML format. Then this file used to computethe metrics of coupling. A relatedmodelwhichaccompanied in 2016 [17] that implements Java source code, parses and generates the XML file. Thedifferent measures of coupling between classes areimplementedby means of the code andthe XMLtokens.

In [18], the authorspresented the coupling metrics to assess the refactoring impact, relating to the metrics before refactoring processes and after. The studyevaluated only a C++ source code that accomplished by a single programmer for some refactoring forms and theselimitationsforms threats to the authority of their results.In another study, Chávez et al. [19] examinein what way refactoring processesimpact on many internal quality attributes including cohesion, coupling. The authors observed that programmers apply 94% of the refactoring processes to source code components with one serious attribute of internal quality,

65% of the refactoring processesincrease the associated attributes and the lasting 35% processessave the attributes unaffected. Root-canal refactoring processes are applied (also called as pure refactoring) on source code to either repeatedlyincrease internal quality or at least not get worse.

Pantiuchina et al. [20] purpose a model to investigate quality metrics to detect a planto improve the quality at design and code phase by programmers empirically. While earliermodels like [21], [22]surveyed programmers to examine whether metrics support their opiniononthe quality of source code, The programmersobviouslymake a plan for enhancingthe attributes of quality likecode complexity, cohesion,readability and coupling.

3. Tool Architecture of Proposed Model

3.1 Metrics Used

The proportion of messages received by a class over the number of messages sent by the class is used to calculate degree of coupling.

$$\text{Degree of coupling DC} = \text{MRC/MPC}$$

Where MRC is the number of messages a class has received from other classes, and MPC denotes the number of messages a given class has passed on to other classes. As a result, it is calculated at the class level rather than the object level [23].

The ratio of the number of attributes used in a class to the total number of attributes for that class is used to calculate the degree of cohesiveness. The functional power of the qualities is examined in the Degree of cohesion. It shows how strongly a class's methods or a service's operations are influenced by the characteristics in the class [24][25].

$$\text{Degree of cohesion (DCH)} = \text{NAU/TNA}$$

a. Tool Architecture

The following steps are used to programming the proposed tool architecture and described in figure 1 in details:

1. Class Diagram: Initially we create a Class Diagram of software design by Enterprise architect tool.
2. XMI Document: Generate XMI document from class diagram by Enterprise architect tool to have the ability to use the information that has been generated in our tool.
3. XMI Parser: XMI-Parsers that constructing a tree ofDOM (Document Object Model) of the XMI file by the Java programming language. DOM provides a standard way for accessing and manipulating XMI documents and extracts information via tokens needed to compute the metrics of cohesion and coupling.

4. Cohesion & Coupling metrics Calculation:Calculate DHA and DC metrics for each Class in the class diagram using information that extracted from XMI parser.
5. Show Results: The results are displayed based on metrics.

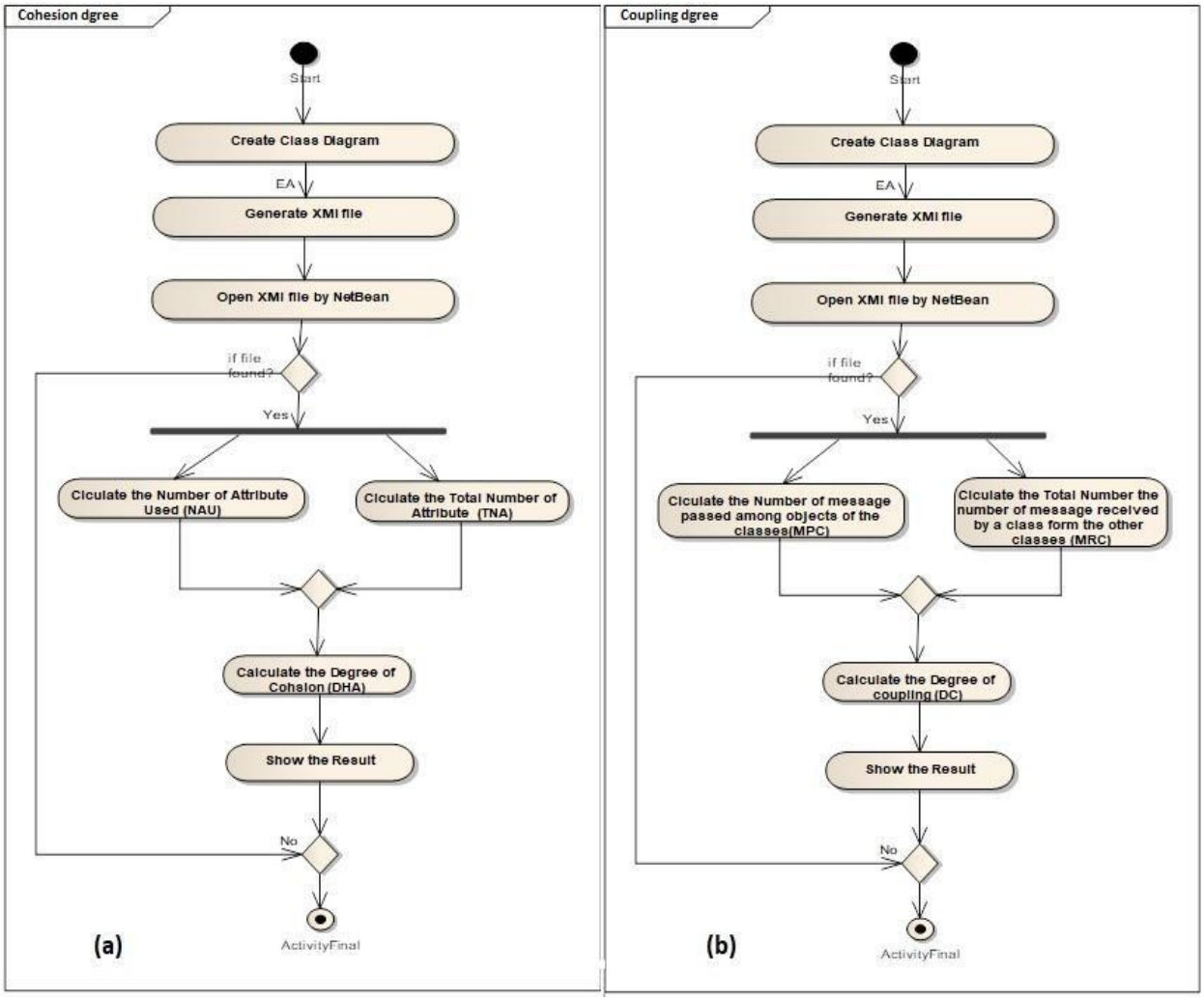

Figure 1: Tool Architecture of the Proposed Model (a) represents the calculation steps of cohesion degree and (b) represents the calculation steps of coupling degree

4. Testing the model and Results: Case Studies

This section explains the testing of the proposed model and results that are obtained after executing it.A case study of the Airplane system is taken as case study 1, Hospital system as case study 2, and ATM system as case study 3. The metrics were calculated for each system to obtain cohesion and

coupling values. The classes of three systems are building using enterprise architect tool and these diagrams are described in figure 2:

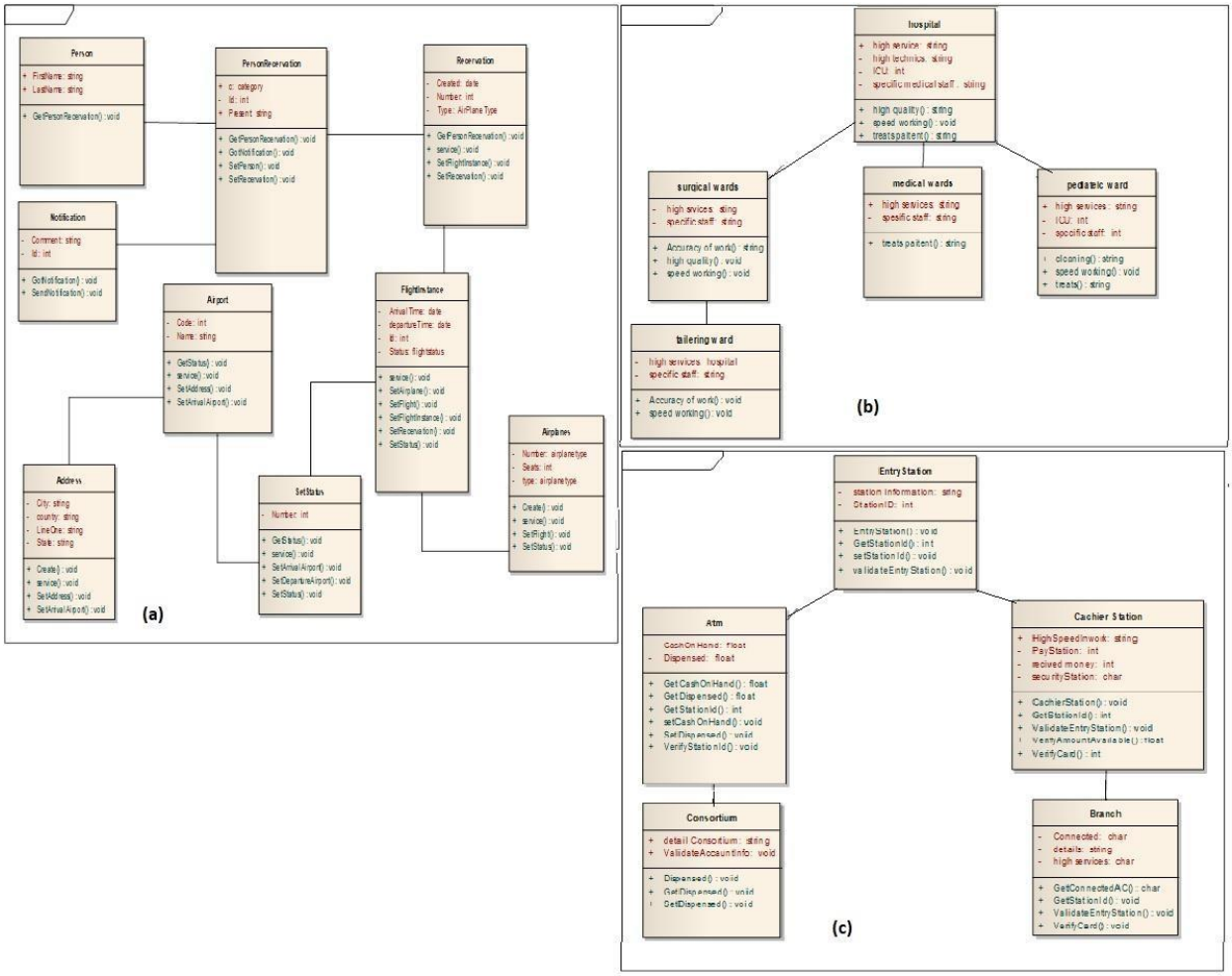

Figure 2 Class diagram of three case studies:(a) Airplane class diagram (b) Hospital class diagram (c) ATM class diagram

In these diagrams, we calculatethe degree of cohesion (DHA) and coupling (DC) for each class. Table 1 lists the results for each system. The value of the DC metric ranges from 0 to 1 and when the value is close to zero, the better for the measure of maintainability, testability, readability, reliability, and reuse. For example class person in airplane system, the value of DC metric is 0 this means high degrees of maintainability, testability, readability, reliability, reuse. Class airport in the same system the value of DC metric is 1 this means low degree of maintainability, testability, and reuse.

The value of the DCH metric ranges from 0 to 1 and when the value is close to 1, the better for the measure of maintainability, testability, readability, reliability, reuse. For example in the airplane system,

The classes Airport and SetStatus are equal to 1, this means high degrees of maintainability, testability, readability, reliability, reuse and class PersonReservation in the same system is equal to 0.33,this meansthe low degree of maintainability, testability, readability, reliability, and reuse.

Table 1: Theresults ofthe proposed tool : values of cohesion and coupling.

Airplane class diagram						
Class name	MRC	MCP	DC	NAU	TAN	DCH
PersonReservation	2.0	2.0	1.0	1.0	3.0	0.33
FlightInstance	3.0	5.0	0.6	2.0	4.0	0.5
Reservation	2.0	3.0	0.66	2.0	3.0	0.66
Airport	3.0	3.0	1.0	2.0	2.0	1.0
Person	0.0	1.0	0.0	1.0	2.0	0.5
Airplanes	3.0	0.0	0.0	1.0	3.0	0.33
Address	3.0	0.0	0.0	3.0	4.0	0.75
SetStatus	2.0	3.0	0.66	1.0	1.0	1.0
Notification	0.0	1.0	0.0	NAU	TAN	DCH
Hospital class diagram						
medical wards	1.0	0.0	0.0	1.0	2.0	0.5
tailering ward	2.0	0.0	0.0	0.0	2.0	0.0
Hospital	0.0	4.0	0.0	3.0	4.0	0.75
surgical wards	2.0	2.0	1.0	2.0	2.0	1.0
pediateic ward	1.0	0.0	0.0	2.0	3.0	0.66
ATM class diagram						
Branch	3.0	0.0	0.0	1.0	3.0	0.33
Consortium	2.0	0.0	0.0	1.0	2.0	0.5
EntryStation	0.0	2.0	0	1.0	2.0	0.5
Atm	1.0	2.0	0.5	2.0	2.0	1.0
Cachier Station	1.0	3.0	0.33	1.0	4.0	0.25

5. Conclusion and Future Work

In the proposedtool, we suggest a method to calculate cohesion and coupling exploiting UML diagrams. So, through the building and testing of the proposed model, conclusions are:Give an indicator to software engineers on cohesion and coupling values to may change the way that information is

presented and to make early improvements and the class with high cohesion and low coupling can be reused in other systems design and reduced effort needed for maintainability, testability, readability, reliability, reuse and These values decide if the code needs to be refactoring for enhancing the quality of design and source code. The proposed tool can be expanded to include other software engineering concepts metrics and acceptingnot only XMI documents, but also XML documents.

References

[1] Choudhary B.,(2018), "Comparing Service Orientation and Object Orientation: A Case Study on Structural Benefits and Maintainability", Institute of Software Technology University of Stuttgart.

[2] Hutchinson B., Smart A., Hanna A., Denton E., Greer C., Kjartansson O., Barnes P., & Mitchell M., (2020), "Towards Accountability for Machine Learning Datasets: Practices from Software Engineering and Infrastructure", arXiv preprint arXiv:2010.13561.

[3] Roy A. &Karforma S., (2013), "Object oriented metrics analysis for implementation of authentication in smart card based E-Governance mechanism", Researchers World – Journal of Arts, Science and Commerce, 4(2), pp: 103 – 109.

[4] Sun Q., Wu J., and Liu K., (2020), „„Toward understanding Students" learning performance in an object-oriented programming course: The perspective of program quality,"" IEEE Access, vol. 8, pp. 37505–37517.

[5] Kaur N., Negi A., & Singh H, (2018), "Object Oriented Dynamic Coupling and Cohesion Metrics: A Review", Lecture Notes in Networks and Systems, pp.861–869.

[6] Miquirice, S. A., &Wazlawick, R. S. (2018), "Relationship Between Cohesion and Coupling Metrics for Object-Oriented Systems", Information and Software Technologies, pp.424–436.

[7] Kalantari, S., Alizadeh, M. &Motameni, H., (2015), "Evaluation of reliability of object-oriented systems based on Cohesion and Coupling Fuzzy computing", Journal of Advance in Computer Research 6(1), pp. 85–99.

[8] Roy A. &Karforma S.,(2013), "Coupling and cohesion analysis for implementation of authentication in EGovernance", ACEEE Conference Proceedings Series 02, Fourth International Joint Conference - Advances in Engineering and Technology (AET) 2013, Elsevier, Pp: 544-554.

[9] Sharma A. &Vishwakarma P. K., (2021), "Maintainability Evaluation for Object Oriented Software Metrics Using Tool Cohesion Inheritance (COIN)", Turkish Journal of Computer and Mathematics EducationVol.12 No.1S, pp.233-238.

[10] Singh D.,(2019), "An optimizating the software metrics for UML structural and behaviourl diagrams using metrics tool", INFOCOMP Journal of Computer Science, vol.18, no.1, pp.9-19.

[11] Alsarraj, R. G., Altaie, A. M., & Fadhil, A. A., (2021). "Designing And Implementing A Tool To Transform Source Code To Uml Diagrams", Periodicals Of Engineering And Natural Sciences, 9(2), pp. 430–440.

[12] Abu Hassan A &Alshayeb M.,(2019), "A metrics suite for UML model stability, Software & Systems Modeling", vol.18, no.1, pp.557-583.

[13] Rapatsalahy A. M., Razafimahatratra H., Mahatody T., Ilie M., Ilie S., & Raft R. N, (2020), "Automatic generation of software components of the Praxeme methodology from ReLEL", 2020 24th International Conference on System Theory, Control and Computing (ICSTCC), pp.843-849.

[14] H. Razafimahatratra, T. Mahatody, J.P. Razafimandimby and S.M. Simionescu, (2017),"Automatic detection of coupling type in the UML sequence diagram", 21st International Conference on System Theory, Control and Computing, Sinaia, Romania, pp. 635-640.

[15] Bonja C. &Kidanmariam E., (2006),"Metrics for class cohesion and similarity between methods", In: Proceedings of the 44th Annual Southeast Regional Conference, ACM, pp. 91–95.

[16] Kayarvizhy S., &Kanmani N., (2011),"An Automated Tool for Computing Object Oriented Metrics Using XML", in International Conference on Advances in Computing and Communications, pp. 69–79.

[17] Bidve V. S. andSarasu P., (2016), "Tool for measuring coupling in objectoriented java software," International Journalof Engineering and Technology, vol. 8, no. 2, pp. 812– 820.

[18] Kataoka Y., Imai T., Andou H., &Fukaya T.,(2002),"A quantitative evaluation of maintainability enhancement by refactoring". In 18th International Conference on So‡ware Maintenance (ICSM). Pp.576–585.

[19]Chávez A., Ferreira I., Fernandes E., Cedrim D., & Garcia A., (2017), "How does refactoring affect internal quality attributes?", Proceedings of the 31st Brazilian Symposium on SoftwareEngineering - SBES"17.

[20] Pantiuchina J., Lanza M., &Bavota G., (2018), "Improving code: The (mis) perception of quality metrics", In 2018 IEEE International Conference on Software Maintenance and Evolution (ICSME), IEEE, pp. 80–91.

[21] Bavota G., Dit B., Oliveto R., Di Penta M., Poshyvanyk D., & De Lucia A., (2013),"An empirical study on the developers" perception of software coupling", In Proceedings of the International Conference on Software Engineering (ICSE), pp. 692– 701.

[22] Steve Counsell, Stephen Swift, Allan Tucker, and Emilia Mendes. Object-oriented cohesion subjectivity amongst experienced and novice developers: an empirical study. ACM SIGSOFT Software Engineering Notes, 31(5):1–10, September 2006.

[23] Miquirice S. A.&Wazlawick R. S., (2018), "Relationship between cohesion and coupling metrics for object-oriented systems," ICISTInternational Conference on Information and Software Technologies, Springer International Publishing, pp. 424–436.

[24] Li, W., & Henry, S., Object-oriented metrics that predict maintainability. J. Syst. Softw. 23(2), 111–122 (1993)

[25] Joy Christy A.,&Umamakeswari A., (2020),"An Object-Oriented Software Complexity Metric for Cohesion", Intelligent Systems Reference Library, vol 185. Springer, Cham.