International Journal of Advance Research in Science and Engineering
Volume No.06, Special Issue No.(01), December 2017
www.ijarse.com
IJARSE
ISSN: 2319-8354

Analysis of operations and parameters involved in interface for CBSE

P.L. Powar[1] , Dr. R.K. Pandey[2], M.P. Singh[3], Bharat Solanki[4]

[1]Department of Mathematics and Computer Science , R. D. University, Jabalpur, India
[2]University Institute of Computer Science and Applications , R. D. University, Jabalpur, India
[3]Department of Computer Science , Dr. B. R. Ambedkar University, Agra
[4]Department of Mathematics and Computer Science, R. D. University, Jabalpur, India

ABSTRACT

Component Based Software Engineering is widely used due to its characteristics such as expandability, reusability, easier processing, convenience for developers, low cost, fast throughput etc. In view of these distinguished properties, many of the computer scientists are inclined to work in this area of research to provide secured, effective and low implementation cost for CBSE. Based on some theoretical informations, in this paper external and shallow based metrics are extracted using parser based tools. By extracting theses metrics, the parameters and operations have been defined at designing level of CBSE which in turns helps developer to write suitable codes for proposed software. An illustrative example has been considered in support of the assertions.

Keywords: Components, Interfaces, Operations, Parameters, Software Metrics, Software Engineering, Effort Estimation etc.

I INTRODUCTION

Considering the reusable software components, the design and development of computer based system which is the major part of component based software development (CBSD) has been studied widely. In the light of CBSD recently, the researcher have started focusing on software system composing instead of software programming. Component based development techniques involve methods for developing software systems by choosing ideal off-the-shelf software components and then assembling them.

The key goals of component based software engineering are as follows:

- Save time and money.
- Increase the quality of software.
- Find defects within the systems.

Dependencies in a component based software system which build the system (cf. [1]) consists of

- The individual dependencies of each component.
- The dependencies amongst the components.

For sufficiently large Component based software system (CBSS) [2] it is quite difficult to analyze and track dependencies relevant to the components. Initially, Li [3] has proposed the matrix model to manage dependencies between components for CBSS. Later, many researchers have used the same technique of representing the dependencies in terms of matrix.

The formal definition in terms of a component has been discussed widely [4]. In this paper, we shall be considering the definition due to Szyperski [5] which is the precise form of all earlier definition:

Definition [5]: A software component is a unit of composition with contractually specified interface and explicit context dependencies only. A software component can be deployed independently and is subject to composition by third parties. In order to use metrics for processing of CBSE, we have referred the work of Falcone and Atkinson [6]. The paper is organized as follows:

Section 1 describes the introduction of component based software development and their utilization. Section 2 covers the details of the various metrics according to their different views used in component based software. Existing work by various researchers have been discussed in Section 3. Research problem of the present paper has been described in Section 4. Section 5 provides the proposed methodology. Section 6 provides application of Java based parser tool with the help of case study. Section 7 provides the result of case study. The discussions on the results are given in section 8. Conclusion of the work has been discussed in Section 9.

II COMPONENT VIEWS AND METRICS

Software measurement is one of the important area of research in the field of software engineering and the researchers are exploring several new ideas in this direction since long. According to IEEE, "A software metric is a quantitative measure of the degree to which a system, component or process possess a given attribute".

For improving the quality of software components many researchers introduced various types of metrics. Considering the factors of complexity, customizability and reusability, Cho and Kim [7] have proposed a new class of metrics with further classification as time metrics and implementation metrics. Focusing on reusability analysis of components, Boxall and Arban [8] have defined a set of interface metrics. Chidamber and Kemerer [9] proposed a suite of six metrics namely number of children, depth of inheritance tree, weighted methods per class, coupling between objects, lack of cohesion in methods, response for class.

Falcone et. al. [6] described four fundamental views external view, shallow (internal) view, deep (internal) view, and complete view of components and used these to define a range of different metrics like TNP, AvgNP, AvgNOPI, AvgNOAI etc. which are based on the core structural and architectural properties of components. In the present paper, authors have considered external view and shallow view and computed several metrics on the basis of parameters and operations. (See Table 2.1.)

Metrics for External View(EV)	Types	Full form	Interpretation
Individual Metrics	TNP	Total Number of Parameters	The total number of parameters of an individual operation of the component.
Average Metrics	AvgNP	Average Number of Parameters	The average number of parameters for the operations of the component.
	AvgNOPI	Average Number of Operations per Provided Interface	The average number of operations per provided interface of the component.
	AvgNOAI	Average Number of operations per Acquired Interface	The average number of operations per acquired interface of the component.
Quantification Metrics	NPO	Number of Provided Operations	The total number of operations provided by the component.
	NAO	Number of Acquired Operations	The total number of operations provided by the acquired components.
Ratio Metrics	IOR= AvgNOPI / AvgNOAI		
	TOR= NPO/NAO		
Metrics for Shallow View(SV)	**Types**	**Full form**	**Interpretation**
Average Metrics	AvgNSO	Average Number of Subcomponent Operations	The average number of operations per subcomponent.

Table 2.1: Types of metrics based on operation and parameters

III EXISTING WORK

Powar et. al. [11], have computed the metrics based on external view and concluded that the NAC, NAI, NPI and DPIN play an important role to assess effort estimation and complexity of CBSE at early stage. Hence, by computing all these metrics, basically we would be in a position to predict the approximate cost of CBSE (see also [10], [15]). Poulin [12] presents a set of metrics used by IBM to estimate the efforts saved by reuse. Washizaki et. al. [13]

discuss the importance of reusability of software components in order to realize the reuse of components effectively and propose a model for black box components from the viewpoint of component users.

IV RESEARCH PROBLEM

In the present paper, authors have extended the previous work (cf. [10], [11]) and a study has been carried out on Individual metrics, Average metrics, Quantification metrics and Ratio Metrics defined by Falcone et. al. [6] on the basis of different views described in Table 2.1for CBSE and extract the following nine metrics (i)TNP (ii) AvgNP (iii) AvgNOPI (iv) AvgNOAI (v) NPO (vi) NAO (vii) IOR (viii) TOR and (ix) AvgNSO. With the help of TNP, AvgNP, AvgNOPI, AvgNOAI, NPO, NAO, IOR, TOR and AvgNSO software developer can easily define the number of operations according to its interface, number of parameters, etc at design stage which can be help to predict the different complexities and architecture of CBSE at early stage i.e. at design level. In this paper, we have extract all the metrics defined in Table 2.1 and implemented practically on one live example with the help of Java based tool.

V PROPOSED METHODOLOGY

Java based parser tool developed in this paper has been implemented on the shopping cart. The proposed methodology is given as follows:

- Design the component diagram of shopping cart software using ArgoUML tool as per requirement of client or user.
- Create XMI file of given component diagram with the help of Export option XMI given in ArgoUML.
- Using Java based software and Netbeans tool the XMI file is then parsed using Java parser for extracting information related to various Individual metrics, Average metrics, Quantification metrics and Ratio Metrics like TNP, AvgNP, AvgNOPI, AvgNOAI, NPO, NAO, IOR, TOR and AvgNSO in CBSE.

VI IMLEMENTATION OF JAVA BASED TOOL FOR EXTRACTION OF DIFFERENT METRICS

In this section, we have considered the model of E-learning system which has been designed with the help of ArgoUML (UML Modelling tool). Our aim is to implement the Java based tool on the component diagram of the E-learning system. (cf. Figure 6.1). A Java based parser tool has been developed on component diagram to compute different metrics of external view, and shallow view described in Table 2.1. This tool works only with XMI files. For parsing the XMI file, SAX [14] – a Java API for XML, is used. The version implemented in the Java based tool is

International Journal of Advance Research in Science and Engineering
Volume No.06, Special Issue No.(01), December 2017
www.ijarse.com
IJARSE
ISSN: 2319-8354

SAX 2.0.1 as the SAX parser is an easy-to-use forward parser. The flow of process of how the Java based tool works is depicted in Figure 6.2.

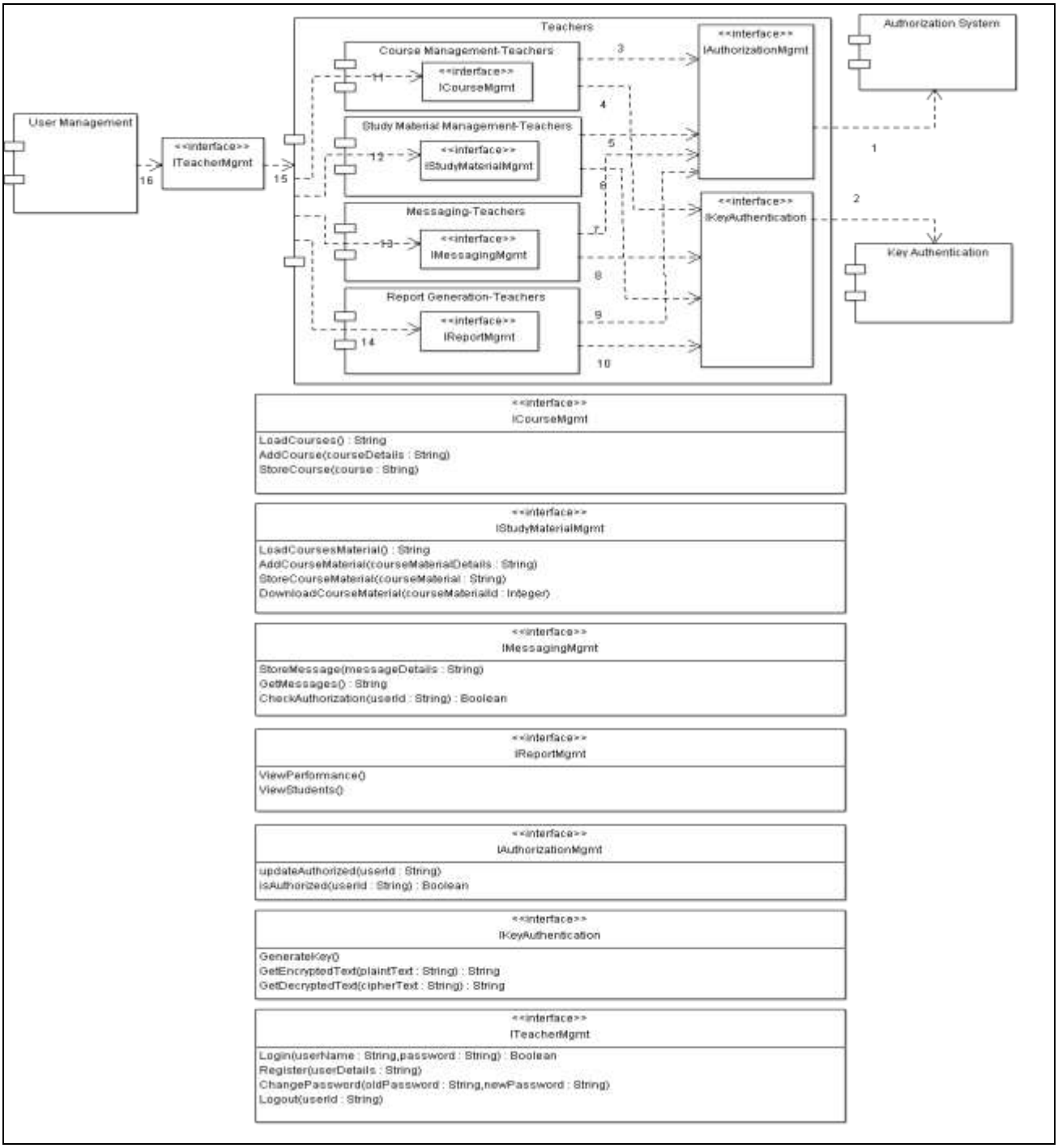

Figure 6.1: Component assembly diagram for E-learning system

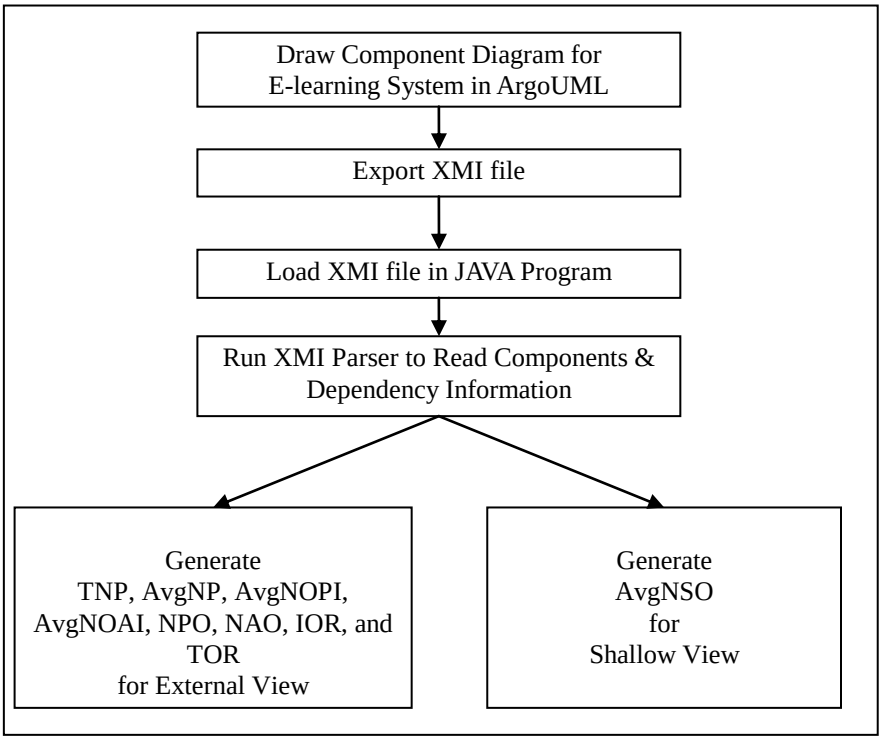

Figure 6.2: Working of Java based tool for E-learning System

The Individual metrics, Average metrics, Quantification metrics and Ratio Metrics implemented in the Java based tool are: TNP, AvgNP, AvgNOPI, AvgNOAI, NPO, NAO, IOR, TOR and AvgNSO all defined by Falcone and Atkinson [6] and defined in Table 2.1 shows some of the metrics currently obtained by using Java based tool, derived through XMI file.

VII RESULT

Figure 6.1 shows a component diagram of E-learning system with components, its sub-components, interfaces and operations. Considering the metrics describe in Table 2.1, the authors have extracted different metrics like TNP, AvgNP, AvgNOPI, AvgNOAI, NPO, NAO, IOR, TOR and AvgNSO which are displayed in Fig 7.1 to Fig 7.9. The significance of these metrics in evaluating the software at the early stage is given as follows:

7.1 Total number of parameters (TNP):

TNP is defined as the total number of parameters of an individual operation of the component. This is a metrics calculated based on the total number of parameters of the various operation i.e. it is a number of parameters defined in functions involved in the interfaces for the external view components. In this example (see figure 6.1) there are

three external components viz. teachers, authentication system, and key authentication. User management is interacting with external component teacher with the help of interface Iteacher management (named as IteacherMgmt), Teachers interacting with external component authentication system, and key authentication with the help of interface Iauthorization management (named as IAuthorizationMgmt) and IkeyAuthentication. Total number of operations are define in these interfaces are nine with ten parameters. **Hence, the value of TNP is equal to 10** (cf. Fig 7.1).

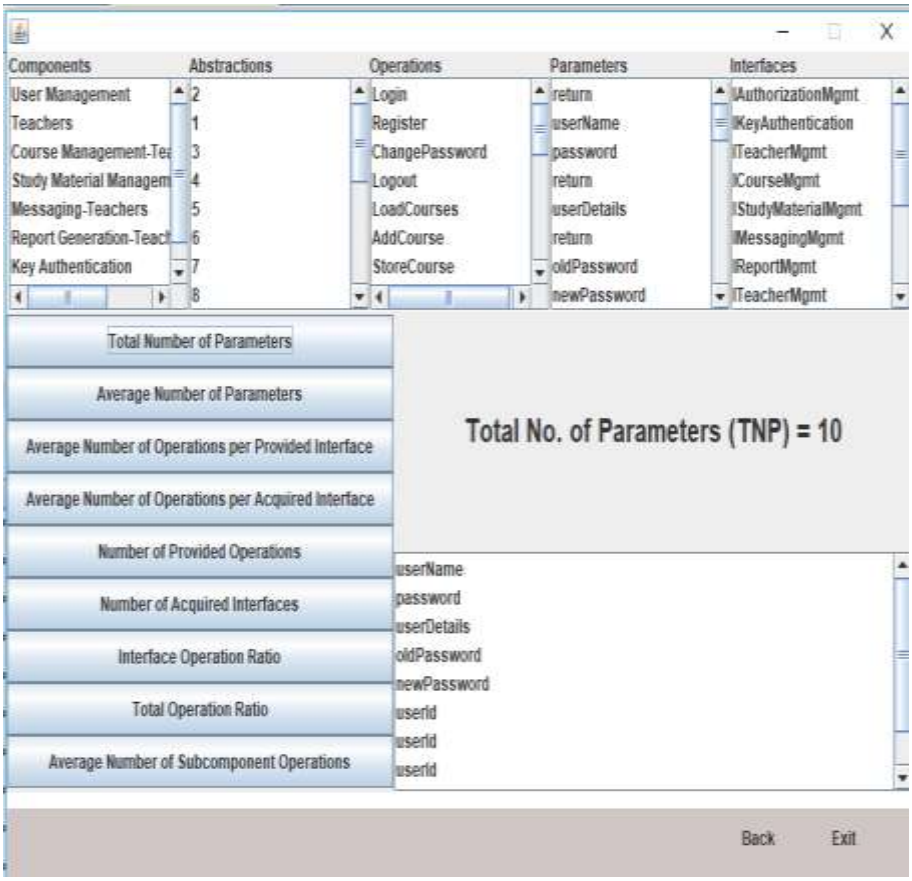

Figure 7.1: GUI for displaying Total number of parameters

7.2 Average Number of Parameters (AvgNP):

AvgNP is defined as the average number of parameters for the operations of the component. This is a measure of Average Number of Parameters required to be processed for the functionality of each operation. In the case study, we have total ten parameters with nine operations resulting in 10/9 is equal to 1.11. **Hence, the value of AvgNP is equal to 1.11** (See Fig. 7.2).

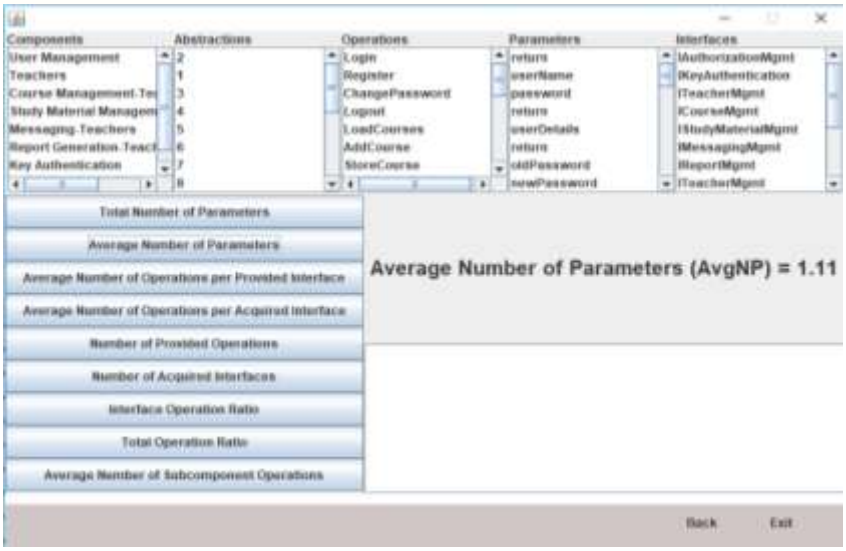

Figure 7.2: GUI for displaying Average number of Parameters

7.3 Average Number of Operations per Provided Interface (AvgNOPI)

AvgNOPI is defined as the average number of operations per provided interface of the components. In the present case study, the external components teachers and authentication system providing two interfaces to teacher and teacher provides one interface to user operations defining in these interfaces are nine resulting in average number of operation provided per interface is equal to 9/3=3 (three). **Hence, the value of AvgNOPI is equal to 3** (see Fig. 7.3).

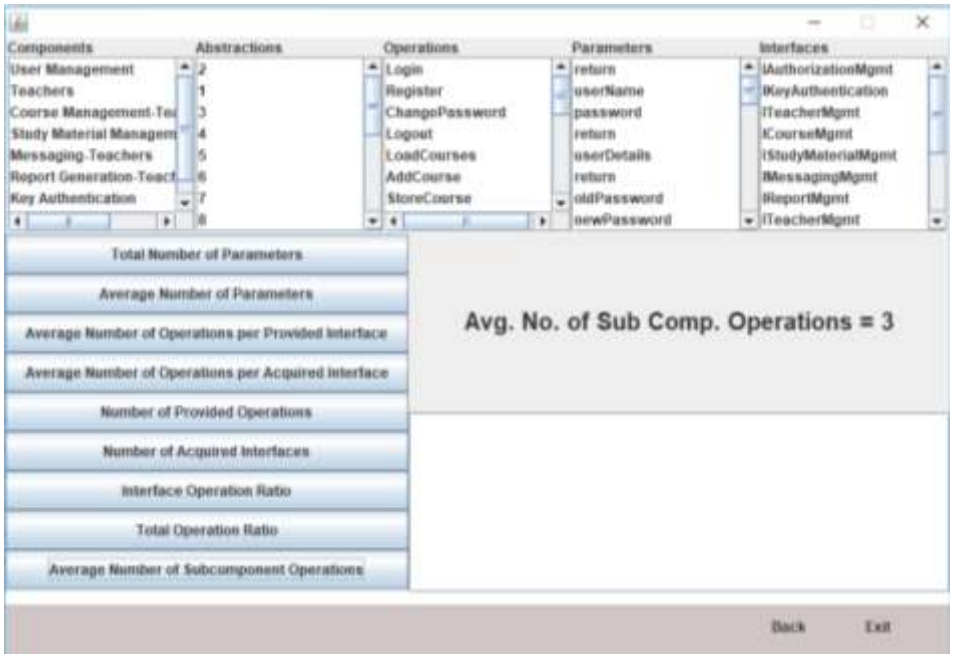

Figure 7.3: GUI for displaying Average Number of Operations per Provided Interface

7.4 Average Number of Operations per Acquired Interface (AvgNOAI):

AvgNOAI is defined as the average number of operations per acquired interface of the component. In the present case study, user management acquiring teacher component through one interface ITeacherMgmt and subcomponents of teacher acquiring component authorization system and key authentication through two interfaces IAuthorizationMgmt and IKeyAuthentication. The total number of acquired interfaces are three and the total number of operations available in the interfaces are nine. **Hence, the value of AvgNOAI is 9/3=3 (three)** (see Fig. 7.4).

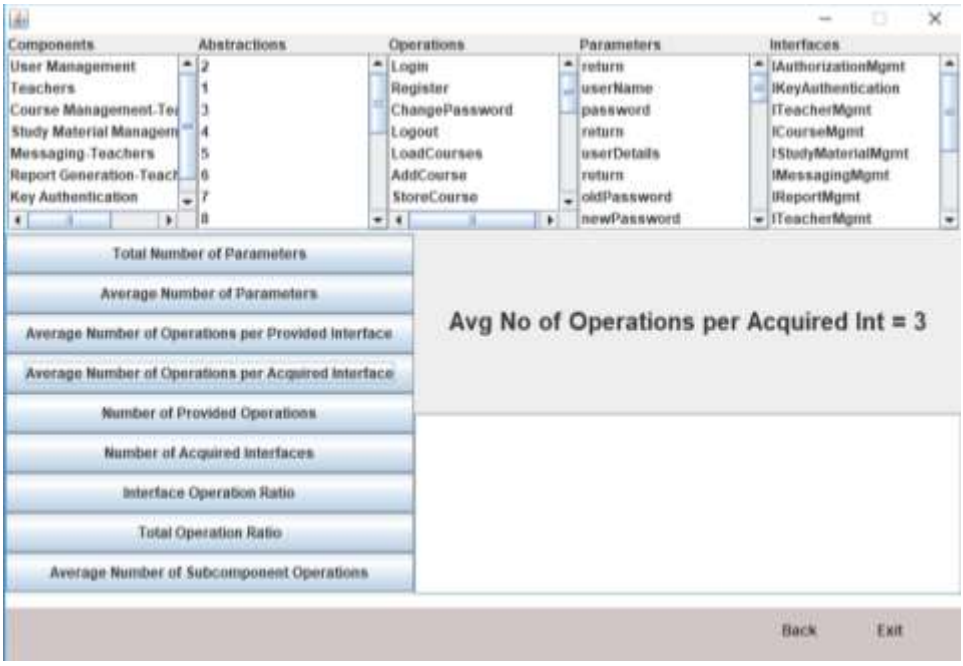

Figure 7.4: GUI for displaying Average Number of Operations per Acquired Interface

7.5 Number of Provided Operations (NPO):

NPO is defined as the total number of operations provided by the component. This is the total number of operations available in all provided interfaces of the external view components. In the present case study, the numbers of provided interfaces are three with total number of nine operations. **Hence, the value of NPO is equal to 9** (see Fig. 7.5)**.**

International Journal of Advance Research in Science and Engineering
Volume No.06, Special Issue No.(01), December 2017
www.ijarse.com
IJARSE
ISSN: 2319-8354

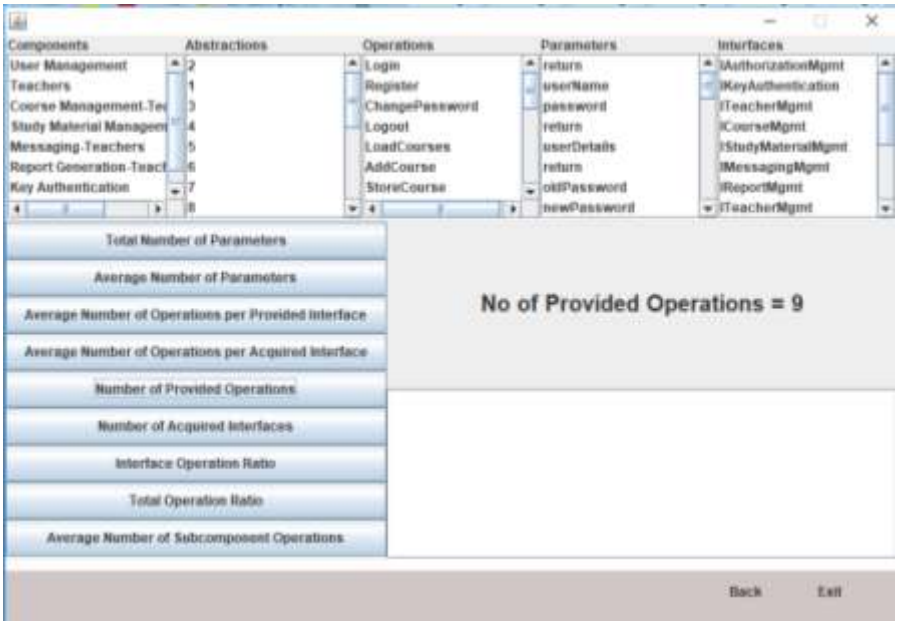

Figure 7.5: GUI for displaying Number of Provided Operations

7.6 Number of Acquired Operations (NAO):

NAO is defined as the total number of operations provided by the acquired components. This is the total number of operations available in all acquired interfaces of the external view components. In the case study, the numbers of acquired interfaces are three with nine operations. **Hence, the value of NAO is equal 9** (see Fig. 7.6)**.**

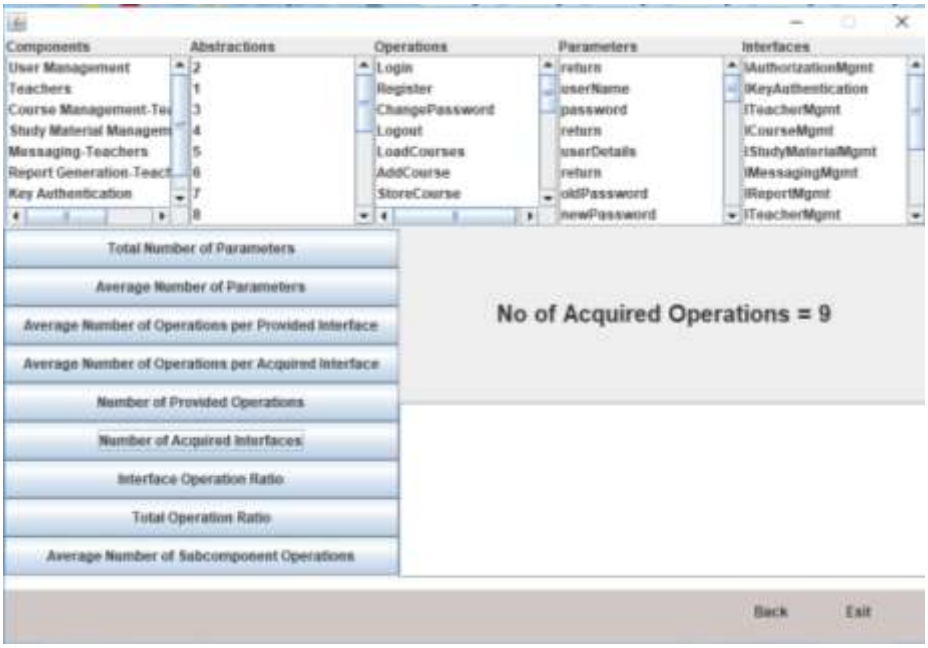

Figure 7.6: GUI for displaying Number of Acquired Operations

International Journal of Advance Research in Science and Engineering
Volume No.06, Special Issue No.(01), December 2017
www.ijarse.com
IJARSE
ISSN: 2319-8354

7.7 Ratio Metrics (IOR):

IOR is defined as the ratio of Average number of operations of provided interfaces and average number of operations of the acquired interfaces. The IOR is defined as:

IOR= AvgNOPI / AvgNOAI

In this example, the value of AvgNOPI is three and the value of AvgNOAI is also equal to three. **Hence the value of IOR is 3/3=1(one)** (see Fig 7.7)**.**

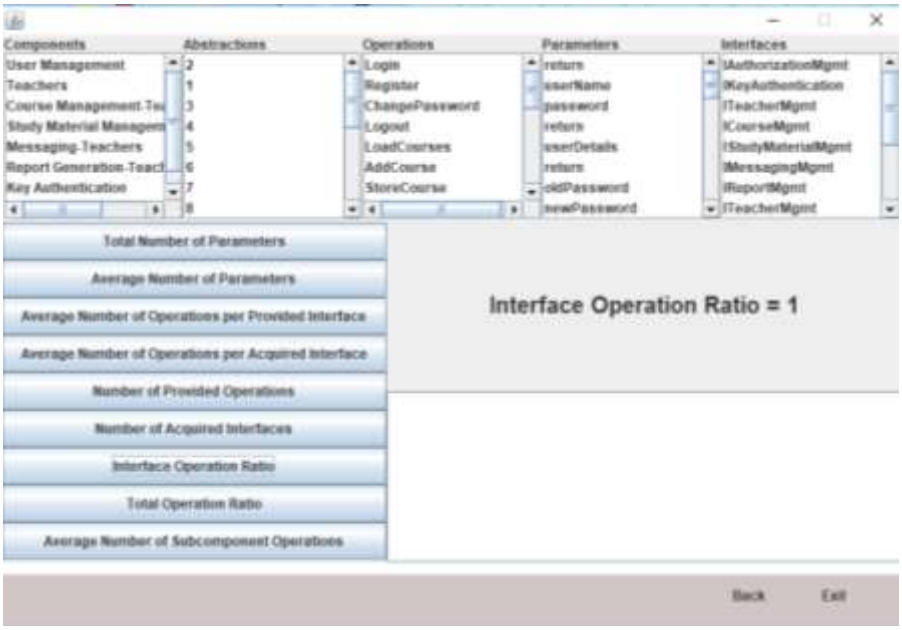

Figure 7.7: GUI for displaying IOR

7.8 Ratio Metrics (TOR)

This is defined as the ratio of number of operations of provided interfaces and number of operations of the acquired interfaces. The TOR is defined as :

TOR= NPO/NAO

In the present system, the value of NPO and NAO, both are equal to nine. **Therefore the value of TOR is 9/9=1(one)** (see Fig. 7.8)**.**

International Journal of Advance Research in Science and Engineering
Volume No.06, Special Issue No.(01), December 2017
www.ijarse.com
IJARSE
ISSN: 2319-8354

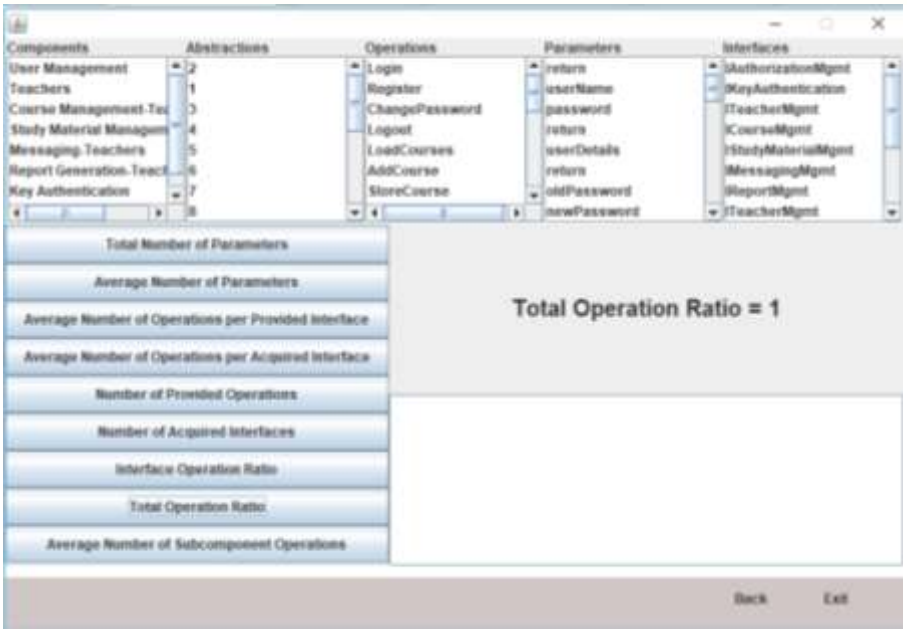

Figure 7.8: GUI for displaying TOR

7.9 Average Number of Subcomponent Operations (AvgNSO):

AvgNSO is defined as the average number of operations required per subcomponent. AvgNSO has been computed corresponding to shallow view. In the example under consideration, the total number of components are four with total number of twelve operations. **Hence, the value of AvgNSO is 12/4= 3 (three)**(see Fig. 7.9) **.**

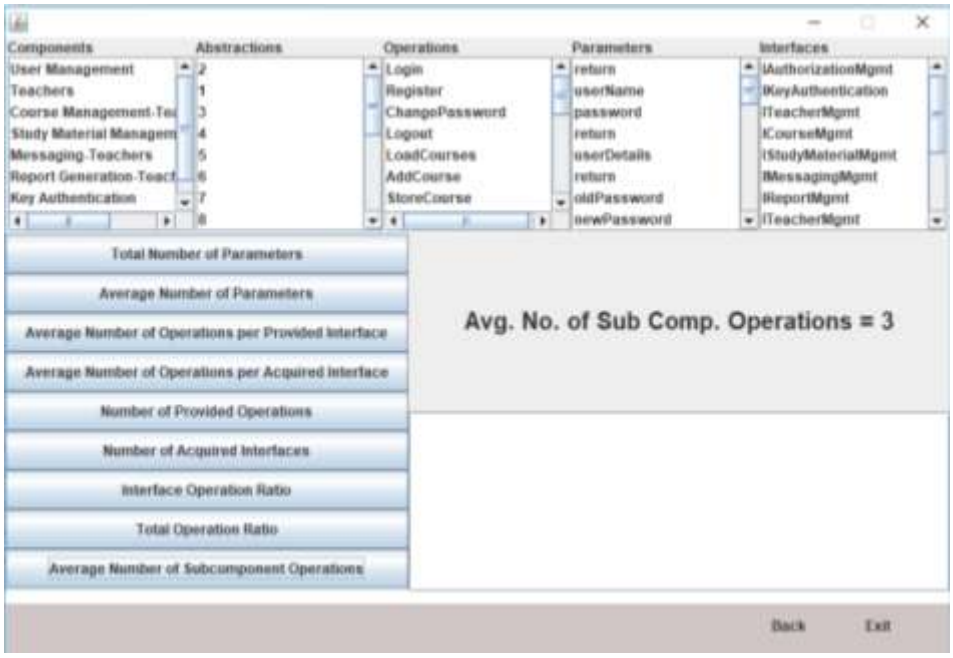

Figure 7.9: GUI for displaying Average Number of Subcomponent Operations

VIII DISCUSSION

In the present case study, the value of TNP is equal to 10. If the value of TNP will increase then line of coding in the expected outcome will also increase, which would result in higher cost. But through this Java based tool we can select best design by comparing the value of TNP amongst different proposed designs of expected software. The value of AvgNP is equal to 1.11. If value of AvgNP is increased, it will result in the increment of complexity of the system, which would arise due to additional coding of software. But in the present case study, this value turns out to be 1.11 which is quite less. Hence, it indicates the reduced complexity of the system. The value of AvgNOPI and AvgNOAI both are equal to 3. If these values increase, then utility of the interface will also increase and system will be more useful. Similarly, the value of NPO and NAO is equal to 9. If NPO and NAO is increased, then functionality of system will also increases, which would result in more utility of the system. If both the value of IOR and TOR will be greater than or equal to one, then it shows that the number of available operations per acquired interface is adequate. In case if the value is less than one, we need to develop additional interfaces, operations, parameters etc. which would be more tedious. The value AvgNSO shows the average number of operations required per subcomponents and these subcomponents exist at deeper level. Hence if the value of AvgNSO increases, affects the complexity of system.

IX CONCLUSION AND FUTURE WORK

In this section, we elaborate the role of each metric computed by the authors in the software development process for CBSE. The computation of TNP in Java based parser tool represents the total number of parameters. By using AvgNP, the system analyst can estimate average number of parameters. AvgNOPI and AvgNOAI calculates average number of operations per provided Interface and average number of operations per Acquired Interface. NPO and NAO is used to identify number of provided operations and number of acquired operations. AvgNSO is used to identify average number of subcomponent operations.

Finally, it may be concluded that the TNP, AvgNP, AvgNOPI, AvgNOAI, NPO, NAO, IOR, TOR and AvgNSO play an important role to assess effort estimation and complexity of CBSE at early stage. By computing these values of metrics numerically, user can easily find the number of parameters, average number of parameters, number of operations per provided interface and average number of operations per acquired interface etc. at design stage, which helps in coding for proposed model to generate the expected outcome of software at later stage and can modify it according to the requirement of the client at early stage. Also by computing these values, we can predict the complexities and architecture of CBSE at early stage. This tool can also be modified to extract other remaining metrics for component-based systems, which will be considered in a future version.

REFERENCES

[1] Szyperski C., Component Software - Beyond Object-Oriented Programming, Addison-Wesley, 1998.

[2] Vieira M., Richardson D. J., Issues in Describing and Analyzing Component Dependencies, Technical Report UCI-ICS-01-39, Department of Information and Computer Science, University of California, Irvine, CA, July 2001.

[3] Li., B. "Managing Dependences in Component Based Systems Based on Matrix Mode", *Net. Objectdays (NODE) 2003 conference*, Erfurt, Germany, 2003, pp.1-12.

[4] Szyiperski C. and Pfister C., Workshop on Component-Oriented Programming, Summary. In Muhlhauser M.(ed.)*Special Issues in Object-Oriented Programming — ECOOP96 Workshop Reader*, Springer 1997.

[5] Szyperski C., Component Software, Addison-Wesley Professional; 2002.

[6] Falcone G., Atkinson C., A Basis for a Metric Suite for Software Components, 12th ECOOP Workshop on Quantitative Approaches in Object-Oriented Software Engineering - Paphos, Cyprus, July 2008 pp. 53-63.

[7] Cho E.S., Kim M.S & Kim S.D., Component Metrics to Measure Component Quality. Proceedings of Eight Asia Pacific Software Engineering Conference, Macau, December 2001, pp. 419-426.

[8] Marcus, Boxall, and Araban S., Interface Metrics for Reusability Analysis of components. Proceedings of Australian Software Engineering Conference (ASWEC'04), Melbourne, Australia, April 2004, pp. 40-46.

[9] Chidamber, and Kemerer, A metrics suite for object oriented design. IEEE transactions on Software Engineering,Vol .20, no. 6, June 1994, pp 476-493.

[10] Powar P.L., Singh M.P., Solanki B. , Agarwal R., Computation of Software Metrics (View Based): XMI approach, International Journal of Engineering Technology and Computer Research (IJETCR), Volume 5, issue 5, sep.-oct. 2017, pp. 123-133.

[11] Powar P.L., Singh M.P, Solanki B., and Shareef J.W., Computation of external view based software metrics: Java based tool, International Journal of Computer Science and Engineering, Volume 5, Issue 8, Aug 2017, pp. 33-43.

[12] Poulin J., Caruso J. and Hancock D., "The Business Case for Software Reuse, IBM Systems Journal, volume 32, issue 4, 1993, pp. 567-594.

[13] Washizaki H., Yamamoto H. and Fukazawa Y., "Software Component Metrics and It's Experimental Evaluation," Proc. of the International Symposium on Empirical Software Engineering (ISESE 2002), October 2002.

[14] SAX, Retrieved on March 5, 2011 http://sax.sourceforge.net.

[15] Pandey R. K., Shareef J.W., Design of a Component Interface Complexity Measurement Tool for Component-Based Systems, ACM SIGSOFT Software Engineering Notes, July 2015, Volume 40 Number 1, pg. 1-12.