International Journal of Computer Science and Network (IJCSN)
Volume 1, Issue 6, December 2012 www.ijcsn.org ISSN 2277-5420

A Methodology for Software Design Quality Assessment of Design Enhancements

[1]Sahar Reda, [2]Hany Ammar, [3]Osman Hegazy

[1] Faculty of Computer & Information, Cairo University, HP Company
Cairo, Egypt

[2] Computer Science and Electrical Engineering, West Virginia University
West Virginia, USA

[3] Faculty of Computer & Information, Cairo University
Cairo, Egypt

Abstract

The most important measure that must be considered in any software product is its design quality. Measuring of the design quality in the early stages of software development is the key to develop and enhance quality software. Research on object oriented design metrics has produced a large number of metrics that can be measured to identify design problems and assess design quality attributes. However the use of these design metrics is limited in practice due to the difficulty of measuring and using a large number of metrics. This paper presents a methodology for software design quality assessment. This methodology helps the designer to measure and assess the changes in design due to design enhancements. The goal of this paper is to illustrate the methodology using practical software design examples and analyze its utility in industrial projects. Finally, we present a case study to illustrate the methodology.

Keywords: Design quality, Independent variable, dependent variable.

1. Introduction

During the past years design quality assessment is essential for the success of the software products. The design is considered the spirit of any successful software system. Any error at this phase can be costly. The design quality must be addressed during the whole process of software development for two reasons [21]: (i) the design is the first phase in software system creation in which quality non-functional requirement can be addressed. (ii) The design has a significant impact on the quality of the final software product. The measuring of design quality is difficult because in many large systems there is no simple and fast procedure we can employ to measure quality factors. For measuring these factors, we have to express them in terms of metrics. Researchers have developed quality models and methodologies that attempt to measure design quality in terms of attributes, characteristics and metrics [27]. There are many metrics proposed for capturing the design quality of object-oriented designs. These metrics provides us ways to evaluate the design quality of software and their use in earlier the phases of software development. These metrics help organizations in assessing the quality of large software designs [6]. However, it is not straight-forward to determine which metrics are useful in capturing important quality attributes like fault-proneness and maintainability.
We adopt the notion of defining design metrics as independent variables that can be measured to assess their impact on design quality attributes as dependent variables. We have presented a set of important object oriented design metrics that can be assessed using a commercial software design measurement metrics tool [22]. The attributes of the design have two types described as follows:

- Quality Attributes: examples of these attributes are functionality, reliability, efficiency, usability, maintainability and portability. These are designated as the dependent variables [8].
- Measurable Attributes: refer to what we can measure. Examples of these are sizing, coupling, cohesion, complexity, and inheritance attributes. These attributes have impact on the quality attributes, and they are designated as the independent variables [4].

The aim of this paper is to assess the design quality enhancements. We propose a methodology for the software design quality assessment of design enhancements and develop customized user interface using Microsoft Excel to aid the designer in assessing design enhancements. The scope of design assessment is to measure and assess UML class diagrams.

The paper is organized as follows: Section 2 presents the approaches of software design quality assessment. Section 3 describes a methodology of design enhancements assessment. Finally, section 4 presents a case study to illustrate proposed methodology.

2. Software Design Quality Assessment Approaches

The software design quality assessment is considered one of important topics in the recent years since the number of the software products increases. There are different approaches in design quality assessment in different studies, some of them developed new object oriented design metrics, others are indentified the relationship between a set of object oriented design metrics[7,11,17,19] and the dependent variables such as re-usability or fault proneness[5, 6, 7,18, 20], and some has shown the validation of framework of a set of object oriented design metrics [20].The last and not the least, they develop new methodology [16, 21] or model [12, 13, 14, and 27] for software design quality assessment.
In this section, we divide the software design quality assessment approaches into two types: the first is to enhance and evaluate the independent variables since the quality of the independent variables is considered an important factor. The second shows the impact of the relationship between the independent variables and the dependent variables.

2.1 Evaluation of Independent Variables

While many of the object-oriented design metrics has been proposed, their validation is very important to measure design quality. Some researchers develop metrics or validate a set of object oriented design metrics. Chidamber and Kemmerer's Metrics Suite defined as the CK Metric Suite. Chidamber and Kemerer (CK) [1, 2] are the first researchers who are heavily cited in different studies [3,8, 9, 10, 15,27]. CK Metrics were defined for evaluating design complexity in relation to their effect on quality factors. These metrics help designers and testers to make better design decisions. They don't develop a tool for measuring the CK Metric Suite. In our work, we can measure and assess the design metrics using user interface and the commercial SD tool [23]. Devpriya Soni[20] has proposed to validate the hierarchical model of object-oriented design quality metrics to evaluate quality of object-oriented software. They could validate the model by proposing a framework that has been validated for both empirical and theoretical validation. The theoretical validation is to assess whether a metric actually measures what its purports to measure. The empirical validation determined the survey to validate empirically defined metrics. The results of the validation according to the observation of the questionnaire were done by the professionals and academic institutes. In our approach, we are able to determine a subset of independent variables [22] based on the clear impact on dependent variables such as maintainability, and fault-proneness.

2.2 The Relationship between Independent variables and Dependent Variables

The methodology presented in [21] assesses the design quality of object oriented designs. The assessment process is to obtain a quality indicator for each completive system then applies a stepwise aggregation mechanism using the logical scoring of preferences method to evaluate global quality attributes. The drawback of this methodology is that the designer has to specify the weights or indicators for each elementary attribute, for each partial attribute, and for each global attribute in each design element, namely classes and components. It is very difficult for designer to specify such weights or indicators. However, our concern in our study to develop methodology to show the effect of the independent variables on dependent variables, the methodology can relate the measured differences as positive or negative changes in the dependent variables. Amjan. Shaik el al[24] and Puja Saxena el al[25] have shown the effect of design metrics as independent variables on dependent variables such as fault proneness. They constructed a prediction model to identify the faulty class. They do not implement any tool to support their solution.

3. The Proposed Methodology

In this section, we present a methodology to enhance the process of the design quality assessment. It is based on measuring a subset of the most common object-oriented design metrics. The methodology of design assessment helps the designer to measure and assess the changes in design due to design enhancements. We develop a customized user interface using Microsoft Excel to support the methodology. Excel has a user-friendly environment where design metrics data can be easily imported and analyzed. Designers can determine the effective changes in design metrics due to design enhancements. The methodology can also be useful for large systems since we focus only on the changed or added part of the system, which is assumed to be a small localized part of the system. The developed user interface helps the designer in the design assessment process by facilitating focus on the

changed classes and changed design measures. The results of these changes are also captured in charts. The relationship between the metrics, which are the independent variables and specific quality attributes. We define here dependent variables, can be identified using a qualitative approach. The user interface recommends to the designer which design quality is better. In figure 1, we summarize proposed methodology architecture

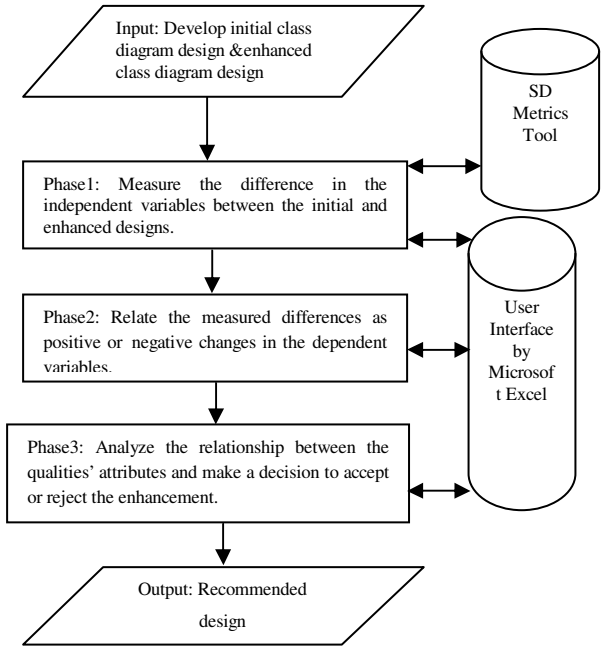

Fig1: Proposed Methodology Architecture

3.1. Measure the Difference in the Independent Variables between the Initial and Enhanced Designs.

The input of phase 1 is the initial and enhanced class diagrams as developed by the designer. The designer measures the difference in the independent variables between the initial and enhanced class diagram at the class or the package level by using the SD tool. In the following subsections, we describe each level.

3.1.1 Measurement of the Class Level

The first step is to measure the deltas values using the SD tool. The deltas values show the difference of measured values between the initial and enhanced designs. We can determine the changed classes and changed design metrics by the user interface. The following steps describe the input, execution and output of measuring at the class level.

Step1: Input: we import deltas' values from SD metrics tool [23] which is considered as input for the user interface.
Step2: Execution: In this step, the user interface identifies the changed classes and design metrics only.
Step3: Output: Changed classes and design metrics.

We could handle the disadvantage of SD metrics tool by getting the changed classes and changed design metrics only using the user interface; The disadvantage of the SD metrics tool is the presentation of all system classes and all SD metrics, whether it is changed in the enhanced design or not.

3.1.2 Measurement of the Packaging Level

A package level, we measure the independent variables at the package level by using some of mathematical formulas. The system has class diagram including a set of packages, and each package has by in turn a set of classes. The following steps discuss the input, execution and output of the measurement at the package level:

Step1: Input: Get the measured design metrics values for initial and enhanced designs by SD metrics tool.

Step2: Execution: In this step, we measure design Quality (DQ) of each independent variable property (p) form the following set {coupling, cohesion, sizing, complexity} in class diagram cd_k using the values resulted from the SD metrics tool. The measurement obtained are normalized by dividing each measured metric value($_j$th) in each package (ith) by the maximum respective metric measured value in both designs in step 2.1 in Equation 1. Then, we calculate Design Quality (DQ) for each property {sizing, coupling, cohesion, complexity} by summing each measured value for each package ($_i$th) in metric ($_j$th) in design k then dividing by number of packages in clas diagram cdk shown in step 2.2 in Equation 2. The details are described in the following sub-steps.

Step 2.1: Evaluate the normalized design metrics in class diagram
In this step, we evaluate the normalized design metrics [NM] by dividing each quality measured value by maximum respective measured value in both designs. For example, for certain coupling metric (jth) we divide each measured value for each package (ith) in design k, by the maximum value for each metric j in all packages (ith) in class diagram (K) of class diagrams k= {1, 2..K}.

$$NM_{cd\,[k](i,j)} = \frac{M_{cd[k](i,j)}}{M_{max(i,k)\,\forall\, j}} \qquad (1)$$

Where

i= i th package of cd_k.
j= jth metrics of property {coupling, cohesion, sizing, complexity}, for each property has a set of design metrics, refer to the object oriented design survey results in [22].
$\mathbf{cd[k]}$ = the class diagram number (k).
$M_{cd[k](j,i)}$: Metric value, for each metric (j) in package (i) in class diagram (k).
$M_{max(j,k)\forall j}$: Max value for each metric (j) in all packages (i) in class diagrams (k).

Steps 2.2: Substitute the value of Equation1 in Equation2
We define Design Quality (DQ) for each property (p) {coupling, cohesion, sizing, complexity} in class diagram cd_k by substituting the value of $NM_{cd[k](j,i)}$ resulted from equation (1) . $NM_{cd[k](j,i)}$ is divided by the number of packages in cdk, denoted by number of the packages, which is $N_{cd[k]}$,and summing the resulted values for each package i for all packages in each metric, then summing over each metric j, and then dividing by the number of object oriented design metrics which is M. This process is repeated for each property (P).

$$DQ_{p,cd[k]} = [\sum_j^M [\sum_i^{N_{cd[k]}} (\frac{NM_{cd[k](j,i)}}{N_{cd[k]}})]/M] \quad (2)$$

Where
$\mathbf{cd[k]}$ = the class diagram number (k).
$DQ_{[p, cd[k]]}$: Design quality in class diagram k in property P.
M: Number of object oriented design metrics.
P: The property {coupling, cohesion, sizing, complexity}.
$N_{cd[k]}$: Number of packages in class diagram cd_k .
j : j th metrics of property {each property has the set of the metrics},for each property has a set of design metrics, refer to the object oriented design survey results in [22].
i : i th package of cd_k.

Step 2.3: Measure the percentage change rate
After we defined the values of $DQ_{[p, cd[k]]}$ for each property in initial and enhanced designs in step 2.2, we calculate the difference between them by measuring percentage change rate; it helps us to understand the change rate for each value. It is the ratio of the amount of difference to the original amount. The increased amount is really the percent of increase. If the amount decreases then the percent of the change is the percent of the decrease which will be a negative.

$$Percentage\ Rate\ \forall P = \frac{DQ[P, cd[k+1]] - DQ[P, cd[k]]}{[DQP, cd[k]] * 100}$$

Where
P: the property {coupling, cohesion, Sizing, complexity}
$DQ_{[P, cd[k]]}$: Design quality in class diagram k in property P.
$DQ_{[P, cd[k+1]]}$: Design quality in class diagram k+1 in property P.
Step 3: Output**:** The output of this phase is the values for $DQ[P, cd[k]]$ and the percentage change rate for each property {sizing, coupling, cohesion, complexity}, we ignore the inheritance since there are no inheritance in the package level.

3.2 Relate the Measured Differences as Positive or Negative Changes in the Dependent Variables.

We have finished measurement phase for the independent variables, in this phase; we specify the difference as positive or negative change in the dependent variable; by getting the relationship between independent and dependent variables. The user interface helps the designer to get the relationship between the dependent and independent variables; such as when sizing decreases, maintainability, understandability and reusability increase and the fault–proneness decreases. More details for the relationship between the independent variables and dependent variables have discussed in [22], the following steps describe the input, execution and output of this phase.

Step1: Input: the output of phase 1, which includes the deltas and DQ_{PK} values, is considered as Input of this step.

Step2: Execution: we determined the relationship by the user interface using independent variables results. We implement two algorithms to show how the user interface develops the relationship between independent variable and dependent variable in the class and package level

Step 2.1 Class level
For each property P [Sizing, Coupling, Cohesion]
For every metric (xi) in each class C for each property P
If (xi<0)
count_total_mins +=1
Else
count_ total_plus+=1
End For
End For
If (count_total_mins> count_ total_plus)
Positive impact on the dependent variables
Else
Negative impact on the dependent variables

Step 2.2 Package level
For every metric (x_i) in each Package P
If (DQ [sizing, coupling, complexity, $cd_{[k+1]}$] < DQ [sizing, coupling, complexity, $cd_{[k]}$])
Positive impact on the dependent variables
Else

Negative impact on the dependent variables
If (DQ [cohesion, $cd[_{k+1}]$] < DQ [cohesion, $cd[_k]$])
Negative impact on the dependent variables
Else
Positive impact on the dependent variables
End For

Step3: Output: The output of this step shows the difference as positive or negative change in the dependent variable by the user interface.

3.3 Analyze the relationship between the qualities' attributes and make a decision to accept or reject the enhancement.

In this phase, we analyze the results to make the decision of which design is applicable to implement. We can accept or reject the design enhancement and get the relationship between the independent variables and dependent variables.

Step 1: Determine the priorities of the quality attributes based on non functional requirements.
In this step, we determine the priorities of quality attributes based on non functional requirements; it is defined as the required attributes of the system, including, maintainability, understandability, test-ability, and understandability [5].The designer can determine priorities of quality attributes according to the customer requirements analysis document.
The aim of determining the priorities of the quality attributes is to analyze and compare the relationship between dependent and independent variables in the user interface. The user interface shows the Kivait chart to present the relationship between the changed classes and changed design metrics; A Kiviat chart is composed of axes as the changed classes extend from a central point as the changed design metrics in the design metrics, each axis represents a data category.

Step2: Determine the decision of enhancement

- The user interface compares between two designs and makes decision to check which design quality is better.
- Finally, the user interface recommends to designer which class diagram is better during the comparison process between two designs.

4. Case Study

In this section, we present a hospital case study to illustrate the methodology at the class level. There are two designs; the initial design as shown in figure 2, while the enhanced design was developed using the Strategy pattern in the original design as shown in figure 3.

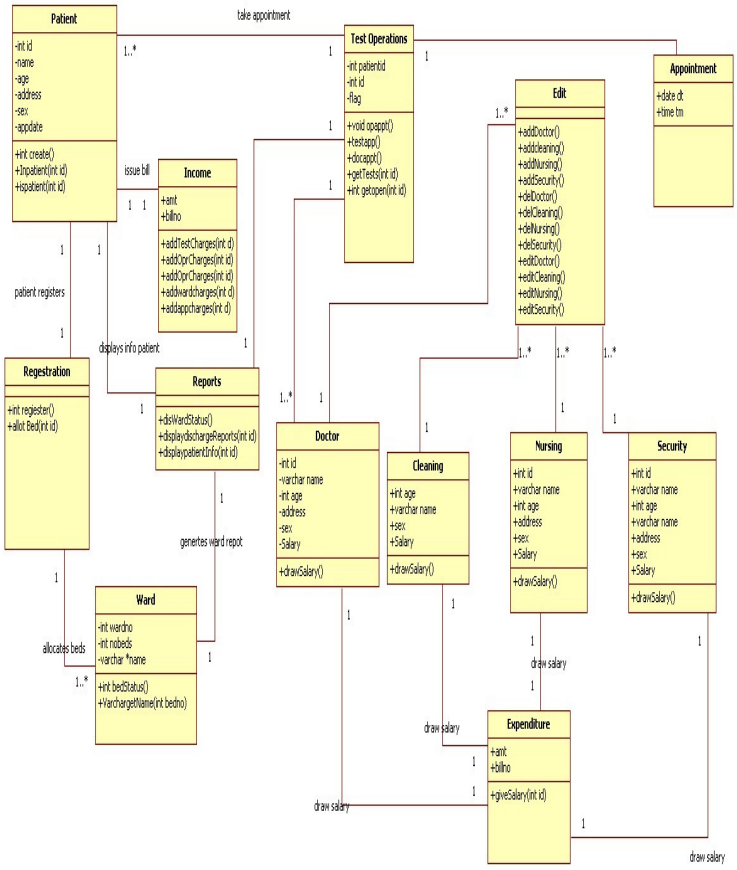

Fig. 2: Class Diagram [CD1] of Hospital Case Study

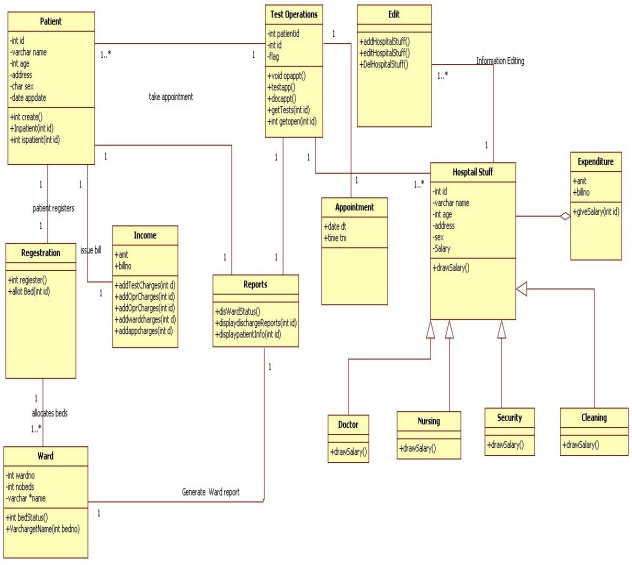

Fig. 3 Enhanced Class Diagram [CD2] of Hospital Case Study by Using Strategy Pattern

4.1Measure the Initial and Enhanced Class Diagrams

In this section, we measure the class diagrams [CD1,CD2] by SD metrics tool and then exporting the results to the user interface. In the following subsections, describe the steps of the design measurement.

4.1.1 Measure the Class Diagrams by getting the deltas from SD metrics tool

By SD metrics tool, we can measure the deltas between two designs. The deltas, as defined, show the difference of measured values from the first design [CD1] to second design [CD2] (values of design metrics in the second design minus values of design metrics in the first design) as shown in figure 4. For example, for "Doctor" Class, in figure 4, the number of attributes [NumAtt] =6, and in enhanced class diagram =0, the delta values is considered as 0-6= -6, it means the value of the Num of attributes is decrease by -6.

Name	NumAttr	NumOps	NumPubOps	Setters	Getters	Nesting	IFImpl	NOC	NumDesc	NumAnc	DIT	CLD	OpsInh	AttrInh
.Design Model.Patient	0	0	0	0	-1	0	0	0	0	0	0	0	0	0
.Design Model.Hospital Staff	5	Demo	1	0	0	0	0	4	4	0	0	1	0	0
.Design Model.Appointment	0	0	0	0	0	0	0	0	0	0	0	0	0	0
.Design Model.Edit	0	0	0	0	0	0	0	0	0	Demo	0	0	0	0
.Design Model.Regestration	0	0	0	0	0	0	0	0	0	0	0	0	0	0
.Design Model.Expenditure	0	0	Demo	0	0	0	0	0	0	0	0	0	0	0
.Design Model.Test Operations	0	0	0	0	0	0	0	0	0	0	0	0	0	0
.Design Model.Ward	0	0	0	0	0	0	0	0	0	0	0	0	0	0
.Design Model.Income	0	0	0	0	0	0	0	0	0	0	0	0	0	0
.Design Model.Reports	0	0	0	0	0	0	0	0	0	0	0	0	0	0
.Design Model.Nursing	-6	0	0	0	0	0	0	0	0	1	1	0	1	6
.Design Model.Security	-7	0	0	0	0	0	0	0	0	1	1	0	1	6
.Design Model.Doctor	-6	0	0	0	0	0	0	0	0	1	1	0	1	6
.Design Model.Cleaning	-4	0	0	0	0	Demo	0	0	0	1	1	0	1	6

Fig. 4: The deltas Design Comparison results between CD1 and CD2 in SD Tool

4.1.2 Click the button "Changed Classes & Design Metrics"

In the below figure 5,in the user interface, we can find changed classes design metrics by clicking button "Changed Classes & Design Metrics", It presents the changed classes in the rows and changed metrics in the columns with same values of SD tool. Therefore, the designer could determine the changed classes and metrics easily through a large of the classes.

	NumAtt	NumPubOps	NumAssEI_ssc	NumAnc	DIT	opsInh	attrInh
Expenditure	0	0	-3	0	0	0	0
Ward	0	0	2	0	0	0	0
Income	0	9	2	0	0	0	0
Reports	0	9	0	0	0	0	0
Nursing	-6	0	-1	1	1	1	6
Security	-7	0	-1	1	1	1	6
Doctor	-6	0	-3	1	1	1	6
Cleaning	-4	0	-1	1	1	1	6

Clear Changed Classess | Changed Classess & Desgin Metrics | Show Kiviat Diagram

Fig. 5: Changed classes and Object oriented design metrics in User Interface by using Microsoft Excel

4.1.3 Generating the Kiviat Chart

In the user interface, we can find the relationship between the classes and the design metrics by clicking button "Show Kiviat Diagram" as shown in figure 6.Kivait chart presents the changed classes [cleaning, doctor, security, nursing, ward, income, reports and Expenditure] and the changed metrics [NumAtt, NumPubOps, NummAssEI_ssc, NumAnc, DIT, OpsInh and attrInh].

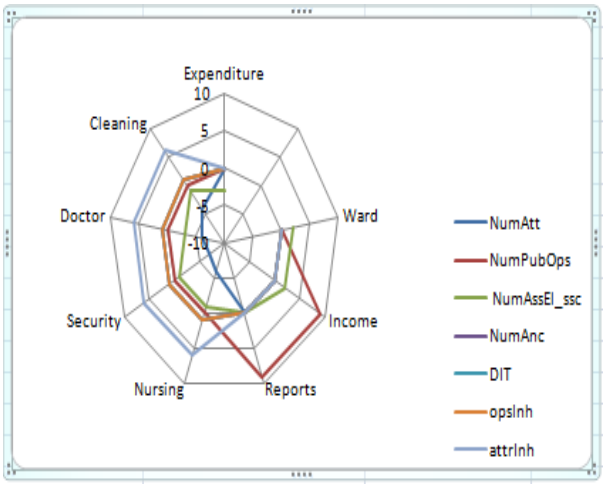

Fig. 6: the Hospital Kiviat Diagram in User Interface by using Microsoft Excel

4.2 Assess the Relationship between the Dependent variable and Independent variables

Another way for assessment in the user interface; by representing the relationship between the independent and dependent variables. In figure 7, there are two buttons, one button for clearing the relation and another to create the relation between the dependent variable and independent

variables, more details for the relationship between the independent variables and dependent variables have discussed in [22]. The rows represent independent variables of sizing, coupling, inheritance, cohesion and complexity, while the columns show maintainability, understandability, reusability and fault-proneness as dependent variables. The relationship between the sizing and dependent variables in the user interface presents the positive impact for each maintainability, understandability and reusability with"++" and also for the fault -Proneness with"--".We have low coupling in CD2, Since the relationship between the coupling and dependent variables in the user interface presents the positive impact for each maintainability, understandability and reusability with"++" and "--" for the fault –Proneness to decrease the number of the faults in hospital system. Also we have high inheritance in CD2, the relationship between the inheritance in the user interface presents the positive impact for each maintainability, understandability and reusability with"++" and also for the fault -Proneness with"--".

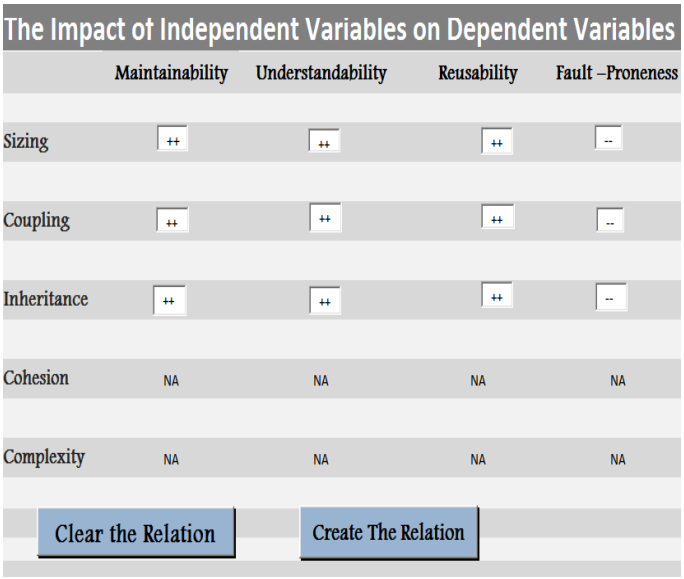

Fig. 7: the relationship between the Dependent variable and Independent variables by the User Interface

4.3 Analyze the Results and Make a Decision to accept or reject the Enhancement.

The designer determines the priorities of the quality attributes based on the non function requirements in the hospital requirements analysis document. We assumed that the hospital system is planned to decrease the number of the faults, to be easy to be understood, less of reusable and maintainable. As shown in figure 8, the user interface is recommended Second design [CD2] rather than initial design [CD1] since there are the positive impact of sizing, coupling and inherence with "++" in dependent variables. We are able to assess the design by simple way using proposed methodology and user interface.

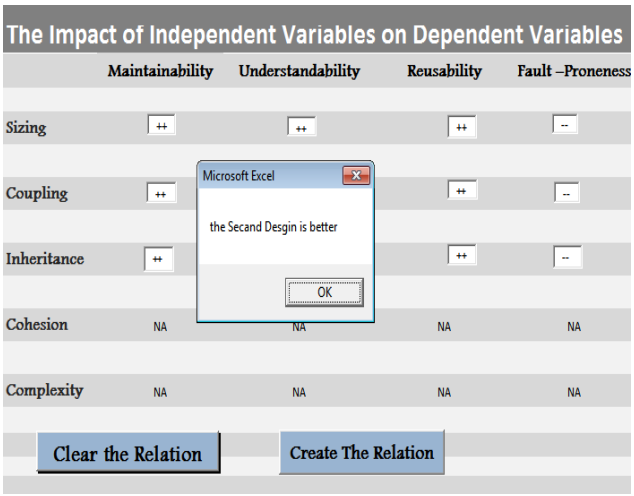

Fig. 8: the recommended enhancement design in hospital system by the user interface

5. Conclusion

In an ideal software design, the relationship between modules shows the relation between qualities attributes as dependent variables and measurable design metrics as independent variables such as loose coupling and tight cohesion. In order to achieve that, we are able to provide the designer with the methodology to assess the design using a set of object oriented design metrics that are supported by the SD tool. In this paper, we have presented a methodology for assessment of design enhancements. We developed a user-interface to support the designer to measure and assess the design. We presented a case study to illustrate the methodology and show the impact of the software design quality assessment methodology on the design process. The next step in our future work is to enhance the user interface to be connected it directly with the SD tool, adding the additional features and functions to the user interface such as additional graphs and charts.

Acknowledgments

This research work was funded in part by Qatar National Research Fund (QNRF) under the National Priorities Research Program (NPRP) Grant No.: 09-1205-2-470.

References

[1] Chidamber, S., Kemerer, C.,"Towards a Metrics Suite for Object Oriented design". Proc. Conference on Object-Oriented Programming:Systems,Languages and Applications (OOPSLA'91). Published in SIGPLAN Notices, 26 (11), 197-211-1991.

[2] Chidamber, S., & Kemerer, C. "A metrics suite for object oriented design". IEEE Transactions on Software Engineering, 1994, 20(6), pp.476–493.

[3] Lake, A., Cook, C, "Use of factor analysis to develop OOP software complexity metrics", Proc. 6th Annual Oregon Workshop on Software Metrics, Silver Falls, Oregon, 1994.

[4] Lanza, Michele, Marinescu, Radu, "Object-Oriented Metrics in Practice book", ISBN 978-3-540-24429-5, 2006.

[5] Briand, Devanbu, Melo, "An Investigation into coupling measures for object-oriented designs", Proceedings of the 19th International Conference on Software Engineering, ICSE '97, Boston, 412-421, 1997.

[6] Lionel C. Briand, Jürgen Wüst, John W. Daly, and D. Victor Porter "Exploring the relationship between design measures and software quality in object-oriented systems" Journal of Systems and Software Volume 51 , Issue 3,Pages: 245 – 273, (May 2000).

[7] M. Xenos, D. Stavrinoudis, K. Zikouli, D. Christodoulakis, "Object-Oriented Metrics-A Survey" Proceedings of the FESMA, Federation of European Software Measurement Associations, Madrid, Spain, 2000.

[8] Ralf Reiing "Towards a Model for Object-Oriented Design Measurement" 5th International ECOOP Workshop on Quantitative Approaches in Object-Oriented Software Engineering, 2001.

[9] Stojanovic, M. and El Emam, K."ES2: A Tool for Collecting Object-Oriented Design Metrics from C++ and Java Source Code", National Research Council Canada, June 2001.

[10] By J. Bansiya , C.G. Davis "A Hierarchical Model for Object-Oriented Design Quality Assessment " Software Engineering, IEEE Transactions, Volume: 28 , Issue: 1 Page(s): 4 – 17, Jan 2002 .

[11] R. Martin, "Agile Software Development: Principles, Patterns, and Practices BooK", Prentice Hall, 2003.

[12] Rajendra K. Bandi, Vijay K. Vaishnavi, Fellow, IEEE, and Daniel E.Turk "Predicting Maintenance Performance Using Object-Oriented Design Complexity Metrics", IEEE VOL.29, NO. 1, pages 77-87, JANUARY 2003.

[13] M. El Wakil, A. El Bastawissi, M. Boshra, and A. Fahmy "Object-Oriented Design Quality Models – A Survey and Comparison", 2nd International Conference on Informatics and Systems (INFOS04), March 2004.

[14] J. Bansiya , C.G. Davis "A Hierarchical Model for Object-Oriented Design Quality Assessment " IEEE Volume 28 No.1 January 2002.

[15] K.K.Aggarwal, Yogesh Singh, Arvinder Kaur and Ruchika Malhotra "Software Design Metrics for Object- Oriented Software" Journal of Object Technology, Vol. 6, No. 1, January-February 2006.

[16] Walid M. Abdelmoez, Katerina Goseva-Popstojanova, Hany H. Ammar "Methodology for Maintainability-Based Risk Assessment" pp 337-342 28 August 2006, IEEE Computer Society.

[17] Seyyed Mohsen Jamali, "Object Oriented Metrics (A Survey Approach) ", Tehran Iran, January 2006.

[18] K.K Aggarwal, Yogesh Singh, Arvinder Kaur,RuchiMalhotra,"Investigating effect of Design Metrics on Fault Proneness in Object-Oriented Systems", Journal of Object Technology, vol. 6, no. 10, pp. 127-141,November-December 2007.

[19] C. Neelamegam, M. Punithavalli, Sri Nehru Maha, Coimbatore, and Sri Ramakrishna " A Survey - Object Oriented Quality Metrics ", Global Journal of Computer Science and Technology , Vol. 9, No.4, 2009.

[20] Devpriya Soni, Ritu Shrivastava and M. Kumar, "A Framework for Validation of Object-Oriented Design Metrics", Maulana Azad National Institute of Technology, Department of Computer Applications, International Journal of Computer Science and Information Security, Vol. 6, No.3, 2009.

[21] Devpriya Soni, Dr. Namita Shrivastava, Dr. M. Kumar "A Methodology for Empirical Quality Assessment of Object-Oriented Design" International Journal of Computer Science and Information Security, Vol. 7, No.2, 2010.

[22] Sahar Reda Ragab, Hany Ammar ,"Object Oriented Design Metrics and Tools: A Survey" , Informatics and Systems (INFOS), The 7th International Conference on Computing & Processing (Hardware/Software), 2010 .

[23]SD metrics tool, "A tool for measuring object-oriented design metrics from UML models" Available at: http://www.sdmetrics.com, visited in September 2011.

[24]Amjan. Shaik,, Dr.C.R.K.Reddy, Bala Manda,M.Tech ,"Empirically Investigating the Effect Of Design Metrics On Fault Proneness in Object Oriented System", International Journal of Computer Science & Engineering Technology, Vol. 2 No. 4, ISSN : 2229-3345,April 2011.

[25]Puja Saxena, Monika, Saini Empirical, "Studies to Predict Fault Proneness: A Review", International Journal of Computer Applications (0975 – 8887, Volume 22– No.8, May 2011.

[26]Microsoft Excel, Available at:http://office.microsoft.com/en-us/excel-help visited in August 2012.

[27]Amjan Shaik,C.R.K. Reddy,A.Damodaram "Object Oriented Software Metrics and Quality Assessment: Current State of the Art"International Journal of Computer Applications (0975 – 8887) Volume 37– No.11, January 2012.