Framework for Maintainability Measurement of Web Application for Efficient Knowledge-Sharing on Campus Intranet

Sanjay Misra[1] and Fidel Egoeze[2]

[1] Covenant University, Ota, Nigeria
[2] Federal University of Technology, Minna, Nigeria
sanjay.misra@covenantuniversity.edu.ng,
egoeze@st.futminna.edu.ng

Abstract. Web Application is placed and accessed over a network, which could be an intranet, extranet or the internet. An intranet is identified as an ideal platform for knowledge- sharing and collaboration in organizations, or institutions. But it is at times hampered by maintainability issue which indeed is a key quality attribute of web applications. This paper presents an explicit description of a process for prediction of maintainability of web application based on design metrics and statistical approach. The work investigates whether a set of measures identified for UML class diagram structural properties (size, complexity, coupling, cohesion) could be good predictors of class diagram maintainability based on the sub-characteristics; understandability, analyzability, and modifiability. Results indicate that useful prediction models can be built from the measures and identified the strongest predictors from the proposed metrics. This framework can be applied to construct maintainability prediction models to control the maintenance tasks of the system and promote efficient collaboration in university campus.

Keywords: web application, maintainability, Intranet, metrics, correlation, regression, analysis.

1 Introduction

Web applications are software systems placed and accessed over a network. This could be an intranet providing private communication within an organization, an extranet for semi-private internet work communication or the internet providing world-wide interaction. Web application has grown from simple websites presenting hyperlink text documents to large – scale complex applications, supporting e-commerce and collaborative activities. Tramontana [14] describes web application as an extension of web site. Conallen [2] categorized web applications into two types as;

- Presentation-oriented: A simple document-sharing website consisting of hyper linked text document providing information to users.
- Service-oriented: A complicated web application that provides some services to users, for example, e-commerce applications.

B. Murgante et al. (Eds.): ICCSA 2014, Part V, LNCS 8583, pp. 649–662, 2014.

Web – based systems provide varied functionalities to diverse users, governments and organizations, and enterprises now recourse to the web for their activities. With this increased reliance on the web as a convenient platform for delivering complex applications, there is the need to ensure that web applications are reliable and of high quality.

A major attribute of quality software is maintainability which is about the ease with which to affect maintenance in a system. According to IEEE 1990 [6], maintainability is the ease with which a software system or component can be modified to correct faults, improve performance or other characteristics, or adapt to a changed environment. Saini et al [13] describes maintainability as the ease with which a product can be maintained in order to correct defects, meet new requirements, make future maintenance easier or cope with a changed environment. A maintainable system undergoes changes with a degree of ease.

ISO 9126 [7] identified maintainability with such sub-characteristics as; Analyzability, changeability, stability, and testability described as follows;

- Analyzability: How easy or difficult it is to diagnose the system for deficiencies or to identify the parts that need to be modified
- Changeability: How easy or difficult it is to make adaptations to the system.
- Stability: How easy or difficult it is to keep the system in a consistent state during modification.
- Testability: How easy or difficult it is to test the system after modification.

Maintainability can be quantified with the use of metrics which can be employed to develop models for predicting maintenance costs and effort. Metrics refers to analytical measurement applied to quantify the state of a system. It is a quantitative measure of the degree to which a system, component, or process possesses a given attribute [3].

There is no specific agreement on how to measure maintainability. Welker and Oman [15] suggested measuring maintainability in terms of Cyclomatic complexity, lines of code (LOC) and lines of comments. Polo et al [12] used number of modification requests, mean effort per modification request and type of correction to examine maintainability. Muthannaet[10] developed maintainability model using polynomial linear regressions. Kiewkanya et al [8] measured maintainability by measuring both modifiability and understandability. Ghosheh et al [5] used UML web design metrics to assess if the maintainability of the system can be improved by comparing and correlating the results with different measures of maintainability. Relationship between UML metrics and maintainability measures including Understandability time, and Modifiability time were studied. Mendes et al [9] propose an effort prediction for design and authoring effort usingsource code metrics. Size measured in terms of length, functionality, and complexity was the independent variable.

Maintainability can be measured at the later or early stage of the development process. At the later stage maintainability measurement can be used to evaluate the system for improvements. The measurement at the early stage is important for early identification of possible risks, and efficient allocation of resources. For early evaluation of maintainability, indicators based on properties of early artifacts [1] such as the structural properties of class diagrams are necessary. Class diagram is a product of unified modeling language (UML). Genero*et al*[4] noted that UML class diagrams are the key outcome of those early phases and the foundation for all later design and implementation. By considering the importance of maintainability by using class diagrams this paper presents a framework for maintainability measurement of the web application based on design metrics applied to UML class diagrams.

The paper is organized as follows; Section 2 presents scope and motivation of the work. Selection of metrics and the data selection and analysis are in section 3 and section 4 respectively. The concluding remarks are in section 5.

2 Scope of the Work

Maintainability can be evaluated by measuring some of its sub- characteristics. A number of sub characteristics can be found but this work limits to evaluate maintainability by measuring the following sub-characteristics; understandability, analyzability, and modifiability of class diagrams, defined as;

- Understandability: Ease with which to understand the system. Everyone involved in the development should be able to understand what the code does.
- Analyzability: Ease with which to diagnose the system for deficiencies or to identify the parts that need to be modified
- Modifiability: Ease with which to make adaptations to the system

The motivation of this work is to establish that the web application maintainability depends on measurable (design) metrics. It is to investigate through statistical approach whether a set of measures identified for UML class diagram structural properties (size, complexity, coupling, cohesion) could be good predictors of class diagram maintainability based on the sub- characteristics mentioned above. To determine this, the following steps are taken: - Metrics selection, Data collection, Data analysis and model development. Data analysis involves Correlation, and Regression analysis. All these steps are explained in coming sections.

3 Metrics Selection

In this section, we identify the metrics that affect the dependent variables (maintainability). Several design metrics have been defined for software maintainability. For this work, design metrics based on UML extension for web application as proposed by Conallen [2] are used. Design metrics provide early identification of potential risks at the development stage. There are the four major structural properties commonly used to represent the quality of any software design irrespective of the development paradigm in use [11]. The design attributes being measured are size, complexity, coupling and cohesion. These attributes have been identified as the most popular software design predictors. These design attributes are summarized as follows:-

- Size: -Size quantifies the structural elements of a system. This can be measured by counting lines of code. The number of lines of code (LOC) is the common way of measuring size. For this work, size of the UML diagram is determined by counting the number of components in the class diagram.
- Complexity: - Measures the complexity of program control flow by calculating the number of loops and branches in a component. It indicates the degree of difficulty in understanding the internal and external structure of class and its relationship. Complexity is measured in this work by the number of associations and relations in the class diagram
- Coupling: - An indication of the degree of interdependency between modules. It is the degree of interaction between two components. Low coupling results in high maintainability. A class is coupled to another, if methods of one class use methods or attributes of another class. For this work, coupling is measured from the relationship and components in the UML class diagram.
- Cohesion: - An indication of the degree to which the methods and attributes of a class bind together. It defines the internal consistency within a class.A class is cohesive when its parts are highly correlated. This work measures cohesion using the relationship and components in the UML class diagram.

The metrics proposed for the study are defined in Gosheh et al [5], though some of them were first defined in Genero et al [3]. The metrics are summarized in Table 1.

4 Data Collection and Analysis

This stage describes the collection of data for the study. This study seeks to establish if any correlation exists between the metric values and the subjective ratings on maintainability sub characteristics. To this end, values of both variables need to be collected. First, the metric values are collected automatically using metric tool. For the subjective ratings on the maintainability sub-characteristics, data were collected from web developers and professionals through controlled experiment. This is for empirical validation of the proposed metrics. Key specifications of the experiment include;

- Goal definition: The experiment is to:-

(i) Determine if any relationship exists between the metrics and the maintainability sub characteristics: understandability, analyzability, and modifiability

(ii) Determine the metrics that are strong candidates to establish prediction models for maintainability of the web application.

- Participants (subjects): The participants were experienced industrial practitioners in the field. A total of fifteen participants were involved. All the participants had good background knowledge of the tasks.

Table 1. Considered Metrics

Type	Metric Name	Description
Size	Server Page (NServP)	Number of server Page
	Client Page (NClienP)	Number of client pages
	Webpage (NWebP)	Number of web pages = server Pages + client pages
	Form Pages (NFrmP)	Number of Form Pages
	Form Elements (NFrmE)	Number of Form Elements
	Client Scripts Components (NCsrC)	Number of Client scripts components
	ServerScriptComponents (NSscrC)	Number of server scripts components
	Class (NC)	Number of classes
	Attributes (NA)	Number of attributes
	Methods	Number of methods
Complexity	Associations (NAss)	Number of associations
	Aggregations (NAgg)	Number of aggregation relationships
	Link (NLnk)	Number of link relationship
	Submit (NSbmt)	Number of Submit relationship × Form Elements
	Build (NBlds)	Number of builds relationship × (Server Script Component + Client Scripts Components)
	Forward (NFwrd)	Number of forward relationships
	Include (NIncld)	Number of include relationships
Coupling	WebDataCoupling (Wdata)	Number of data exchangedover number of server pages (Form Elements/Server Page)
	WebControlCoupling (WContr)	Number of relationships over number of web Pages (Link + submit + build + Forward + include) / web Pages
	EntropyCoupling (EntCoup)	1/n × (- log1/(1 + m)) Where n = number of elements m = number of relationships
Cohesion	EntropyCohesion (EntCoh)	Total entropy coupling/ entropy coupling of one class diagram.

- Experiment materials and tasks: This work is on intranet web application for promotion of knowledge- sharing in academic institutions. The materials used for the study consist of 21 class diagrams of an Academic Management Information System. As indicated earlier, the subjects are to rate each maintainability sub-characteristics on a five- point scale as depicted below;

Very Easy	Easy	Medium	Difficult	Very Difficult
5	4	3	2	1

The metric values of the UML class diagrams are shown in Appendix (Table 6), and the subjects' ratings on the maintainability sub characteristics are in Appendix (Table 7). The subjects' ratings presented are the mean values of the rating.

-Variable definition: As indicated above, values of two variables are involved in the maintainability measurement. The variables are defined as follows;

- The independent variables are the metric value of the UML class diagram size, complexity, coupling, and cohesion.
- The dependent variables are the maintainability sub-characteristics; understandability, analyzability, and modifiability measured using the subjects' ratings

4.1 Data Analysis

As noted earlier, this work is to determine the correlation between the metric values and the subjective ratings on maintainability. It is also to identify from the correlation, the strongest predictors for model development. These tasks are accomplished using statistical analysis including correlation, and regression analysis.

4.1.1 Correlation Analysis

This is used to establish the correlation between the dependent variables and the metrics proposed (independent variables) and identify the metrics that are the strongest predictors. This can be accomplished by examining the correlation between each metric and the subjective ratings provided by the software developers [10]. The higher the value of the coefficient of determination, the stronger the relationship between maintainability and the metric considered. Thus, metrics with high correlation coefficients are candidates in the maintainability prediction. The metrics are ranked in order of the correlation coefficients. Spearman rank correlation was used to determine the correlation of the data collected in the study. Each of the metrics was correlated separately to the subjective ratings. To calculate Spearman correlation coefficient (R), data are arranged in ranks and the difference in rank (d) computed for each pair. Spearman correlation equation is expressed as follows;

$$R = 1 - \frac{6\,(\Sigma d^2)}{n\,(n^2-1)} \qquad (1)$$

Where

d = difference in ranks for each pair of items
n = number of observations.

Now, as a way of illustration computation of the Spearman Rank correlation for the first predictor, NServP, with the sub characteristics; Understandability, Analysability, and Modifiability are demonstrated in Table 2, Table 3 and Table 4. Spearman Correlation Summary for the metrics and the dependent variables are summarized in Table 5.

Table 2. Metrics Correlation with Understandability: NServpVs Understandability

NServP	NservP Rank	Underst.	Underst. Rank	d	d^2
4	4.5	3	8	-3.5	25
4	4.5	4	15	-10.5	10.25
3	1.5	4	15	-13.5	182.25
5	8.5	2	2.5	6	36
3	1.5	4	15	-13.5	182.25
6	11.5	4	15	-3.5	12.25
4	4.5	5	20	-15.5	240.25
5	8.5	2	2.5	6	36
6	11.5	3	8	3.5	12.25
7	13	3	8	5	25
9	14.5	3	8	6.5	42.25
5	8.5	4	15	-6.5	42.25
11	17	3	8	9	81
14	19.5	5	20	-0.5	0.25
10	16	4	15	1	1
4	4.5	5	20	-15.5	240.25
12	18	3	8	10	100
15	21	2	2.5	18.5	342.25
9	14.5	3	8	6.5	42.25
5	8.5	4	15	-6.5	42.25
14	19.5	2	2.5	17	289
					$\sum d^2 = \underline{2071.5}$

$$R = 1 - \frac{6x2071.5}{21x440}$$
$$= 1 - 1.345 \quad = 0.345$$

Table 3. Metrics Correlation with Analyzability: NservPVs Analyzability

NservP	NservPRank	Analyz ab.	Analyza. Rank	d D	d²
4	4.5	2	4	0.5	0.25
4	4.5	2	4	0.5	0.25
3	1.5	3	11	-9.5	90.25
5	8.5	3	11	-2.5	6.25
3	1.5	3	11	-9.5	90.25
6	11.5	2	4	7.5	56.25
4	4.5	2	4	0.5	0.25
5	8.5	4	16.5	-8	64
6	11.5	2	4	7.5	56.25
7	13	2	4	9	81
9	14.5	3	11	3.5	12.25
5	8.5	2	4	4.5	20.25
11	17	3	11	6	36
14	19.5	3	11	8.5	72.25
10	16	3	11	5	25
4	4.5	5	20	-15.5	240.25
12	18	4	16.5	2.5	6.25
15	21	4	16.5	4.5	20.25
9	14.5	5	20	-5.5	30.25
5	8.5	5	20	-11.5	132.25
14	19.5	4	16.5	3	9
					1,040

$$R = 1 - \frac{6 \times 1040}{21 \times 440}$$
$$= 1 - 0.675 \quad = 0.325$$

Table 4. Metrics Correlation with Modifiability:NservpVs Modifiability

NServP	NServP Rank	Modifiability	Modifiability Rank	D	d²
4	4.5	1	1.5	3	9
4	4.5	2	5.5	1	1
3	1.5	2	5.5	4	16
5	8.5	2	5.5	-3	9
3	1.5	1	1.5	0	0
6	11.5	3	12	-0.5	0.25
4	4.5	2	5.5	1	1
5	8.5	2	5.5	-3	9

Table 4. (*continued*)

6	11.5	3	12	-0.5	0.25
7	13	3	12	-1	1
9	14.5	3	12	-2.5	6.25
5	8.5	3	12	3.5	12.25
11	17	5	20	3	9
14	19.5	4	17	-2.5	6.25
10	16	4	17	1	1
4	4.5	2	5.5	1	1
12	18	5	20	2	4
15	21	5	20	-1	1
9	14.5	3	12	-2.5	6.25
5	8.5	3	12	3.5	12.25
14	19.5	4	17	2.5	6.25
					112

$$R = 1 - \frac{6x112}{21x440}$$
$$= 1 - 0.073 \quad = 0.927$$

The entire correlation results are summarized as follows:

Table 5. Spearman Correlation Summary for the metrics and the dependent variables

Metrics	Understandability	Analyzability	Modifiability
NServP	-0.345	0.325	**0.927**
NClienP	**0.942**	-0.131	-0.083
NWbP	0.196	0.332	**0.855**
NFrmP	-0.124	0.162	0.415
NFrmE	0.065	0.527	0.319
NCSrC	-0.018	0.411	0.480
NAssR	0.045	**0.904**	0.259
NAggR	**0.882**	0.066	0.098
NLnkR	0.166	0.195	0.518
NSbmtR	0.236	-0.202	0.499
NBldsR	0.016	0.113	-0.239
FwdR	0.302	0.266	0.265
NIncdR	0.017	0.50	0.475
Wdata	0.407	-0.191	-0.739
Wcontr	0.017	-0.178	-0.798
Entcoh	-0.139	0.566	0.430

From the Table, it is observed that the strongest predictors for the Maintainability sub-characteristics are:

(i) Understandability – NclienP (0.942), and NAggR (0.882)
(ii) Analyzability – NAss (0.904)
(iii) Modifiability – NservP (0.927), and NwebP (0.855)

So, these are the variables to be used to build the prediction models.

4.1.2 Regression Analysis

Regression analysis is a statistical method for predicting the value of a dependent variable from one or more predictors (independent) variables. Researchers usually rely on (multiple) regression to predict dependent (criterion) variable from independent (predictor) variables. A model of relationship is hypothesized and estimates of the parameter values are used to develop an estimation regression equation [16]. Various tests are then used to evaluate the model. The general linear regression equation applied to develop the model is given as:

$$Y = a + b_1X_1 + b_2X_2 + \text{----} + b_kX_k$$

where

Y = dependent variable
$X_1, X_2, \text{---} X_k$ = independent variables
$b_1, b_2, \text{---} b_k$ = coefficients of regression
a =Intercept.

For multiple regression involving two independent variables,

$$b_1 = \frac{SDy}{SDx_1} \times \beta_1 \quad \text{(SD = standard deviation)} \tag{2}$$

$$b_2 = \frac{SDy}{SDx_2} \times \beta_2 \tag{3}$$

$$a = \overline{Y} - b_1\overline{X} - b_2\overline{X_2} \tag{4}$$

where

$$\beta_1 = \frac{r_{y1} - r_{12}\, r_{y2}}{1 - r^2_{12}} \tag{5}$$

$$\beta_2 = \frac{r_{y2} - r_{12}\, r_{y1}}{1 - r^2_{12}} \quad \text{(r = Pearson correlation coefficient)} \tag{6}$$

$$r_{y1} = \frac{Sx_1y}{\sqrt{Sx^2_1 Sy^2}} \tag{7}$$

$$r_{12} = \frac{Sx_1x_2}{\sqrt{Sx^2_1Sx^2_2}} \quad (8)$$

$$ry_2 = \frac{Sx_2y}{\sqrt{Sx^2_2Sy^2}} \quad (9)$$

Where

$$Sx_1^2 = \Sigma X_1^2 - \frac{(\Sigma X_1)^2}{n} \quad \text{(n = number of observations)} \quad (10)$$

$$Sx_2^2 = \Sigma X_2^2 - \frac{(\Sigma X_2)^2}{n} \quad (11)$$

$$Sy^2 = \Sigma\, \frac{y^2 - (\Sigma y)^2}{n} \quad (12)$$

$$Sx_1\, x_2 = \Sigma X_1X_2 - \frac{(\Sigma X_1)(\Sigma X_2)}{n} \quad (13)$$

$$Sx_1y = \Sigma X_1Y - \frac{(\Sigma X_1)(\Sigma Y)}{n} \quad (14)$$

$$Sx_2y = \Sigma X_2y - \frac{(\Sigma X_2)(\Sigma Y)}{n} \quad (15)$$

It should be noted that for a case of one independent variable, simple regression equation is applied. This is expressed as;

$$Y = a+bx \quad (16)$$

Where

$$b = \frac{Sxy}{Sx^2} \quad (17)$$

$$a = \overline{Y} - b\overline{X} \quad (18)$$

$$Sxy = \Sigma XY - \frac{\Sigma X \Sigma Y}{n} \quad (19)$$

and

$$Sx^2 = \Sigma X^2 - \frac{(\Sigma X)^2}{n} \quad (20)$$

These are the underlying computations in the development of regression models involving two independent variables, and one independent variable respectively. However, computations using statistical software packages prove to be easier and more dependable, especially for cases involving more than two independent variables.

5 Conclusion

This paper has presented a maintainability quantification framework that could be employed to develop prediction models to control the maintenance tasks of the intranet web application based on design metrics applied to UML class diagram, and statistical approach. The process involved metrics identification, data collection, correlation analysis and regression analysis and model development. Design attributes measured were size, complexity, coupling and cohesion and subjective ratings were obtained on maintainability sub-characteristics through controlled experiment. Data collected were analyzed using Spearman rank correlation analysis and from the analysis the strongest predictors that could be applied to build maintainability prediction model were identified. Computations underlying the regression models were presented. The framework presented in this paper has provided detailed steps to reliably measure maintainability. This can be used to effectively control the maintenance of intranet application for efficient knowledge- sharing on a university campus.

Future Work: In this paper we have presented regression model which explain that how to compute the regression coefficients (for developing the model). The application of the model is the task of the future work. We are collecting the separate data for the examining maintainability sub- characteristics.

References

1. Briand, L., Arisholm, S., Counsell, F., Houdek, F., Thevenod-Fosse, P.: Empiricalstudies of object- oriented artefacts, method and processes; State of the art and future directives. Empirical software engineering 4(4), 357–404 (2000)
2. Conallen, J.: Building web applications with UML. Addison-Wesley Publishing Company, Reading (2003)
3. Genero, M., Piattini, M., Calero, C.: Empirical validation of class diagram metrics. In: Proceedings of the International Symposium in Empirical Software Engineering, pp. 195–203. IEEE Computer Society Press (2002)
4. Genero, M., Manso, E., Cantone, G.: Building UML class diagram maintainability prediction models based on early metrics. In: Proceedings of the 9th International Symposium on Software Metrics 2003, pp. 263–275 (2003)
5. Ghosheh, E., Black, S., Qaddour, J.: Design Metrics for Web Application maintainability Measurement. In: Proceeding of 6th IEEE/ACS International Conference Computer Systems and applications, pp. 778–784. IEEE, Doha (2008) ISBN 97814244196
6. IEE: Institute of Electronics Engineers. IEE standard computer Dictionary. A Compilation of IEE Standard Computer Dictionary. New York (1990)

7. ISO 9126-1 Software Engineering –Product quality- Part 1: Quality model (2001)
8. Kiewkanya, M., Jindasawat, N., Muenchaisri, P.: A methodology for constructing maintainability model of object- oriented design. In: Proceedings of 4th International Conference on Quality Software, pp. 206–213. IEEE Computer Society (2004)
9. Mendes, E., Mosley, S.: Council: Web Metrics – Estimating design and authoring effort. IEEE Multimedia 8(1), 50–57 (2001)
10. Muthanna, S., Kontogiannis, K., Ponnainbalam, K., Stacey, B.: A maintainability Model for Industrial Software Systems using Design Level metrics. In: Proceeding of the 7th Working Conference on Reverse Engineering, Brisbane, Australia, pp. 248–256 (2000) ISSN: 1095-1350
11. Perepletchikov, M.: Software design metrics for predicting maintainability of service – oriented software; Phd thesis, School of Computer Science and Information technology. RMIT University Melbourne, Austalia (2009)
12. Polo, M., Piattini, M., Ruiz, F.: Using code metrics to predict maintenance of legacy programs: A case study. In: Proceedings of the Internal Conference on Software Maintenance (ICSM), pp. 202–208. IEEE Computer Society, Florence (2001)
13. Saini, R., Dubey, S.K., Rana, A.: Analytical study of Maintainability models for quality evaluation. Indian Journal of Computer Science and Engineering 2(3) (2011) ISSN 0976-5106
14. Tramontana, P.: Reverse engineering of web Application: Universita Degli Studidi Napoli Federico II. PhD thesis (2005)
15. Welker, K.D., Oman, P.W.: Software maintainability metrics models in practice. Journal of Defense software engineering 8(6), 19–23 (1995), `http://Crosstalkonline.org`
16. Zhao, L., Hayes, J.: Maintainability Prediction: a regression analysis of measures of evolving systems. In: Proceeding of the 21st IEEE International Conference on Software Maintenance, September 25-29, pp. 601–604 (2005)

Appendix

Table 6. Measure Values of UML Class Diagrams

Diagram	ServP	NclP	NwbP	FrmP	FrmE	CSC	ASS	Aggr	Lnk	Sbmt	Blds	Fwd	Incd	Wdata	Wcontr	Entcoh
D1	4	3	7	1	4	0	1	0	2	4	3	1	2	1	1.7	1
D2	4	4	8	1	4	0	1	1	3	3	2	2	2	1	1.5	1
D3	3	4	9	2	3	1	2	2	2	3	3	1	1	1	1.1	1
D4	5	1	5	2	4	0	2	2	2	4	3	1	2	0.8	2.4	2
D5	3	3	6	2	4	2	2	1	2	4	2	2	2	1.3	2	2
D6	6	4	10	3	3	2	1	0	3	4	3	1	2	0.5	1.3	1
D7	4	6	10	2	4	3	1	1	2	4	3	1	2	1	1.2	3
D8	5	1	6	3	4	3	3	0	2	3	4	2	1	0.8	2	3
D9	6	3	9	1	3	2	1	2	3	3	3	2	2	0.5	1.6	2
D10	7	3	10	3	4	4	1	3	4	3	1	2	2	0.6	1.2	4
D11	9	3	12	5	4	4	2	0	5	4	2	1	3	0.4	1.3	4
D12	5	4	9	4	4	3	2	2	4	3	2	3	1	0.3	1.4	3
D13	11	2	13	1	2	1	1	0	5	3	2	2	3	0.2	1.2	1
D14	14	6	20	2	4	2	2	3	3	2	1	3	2	0.3	0.6	2
D15	10	4	14	3	4	4	2	3	1	3	2	1	2	0.4	0.6	4
D16	4	6	10	1	4	6	3	3	4	4	2	4	3	1	1.7	6
D17	12	3	15	4	6	6	3	2	3	1	2	4	3	0.5	0.9	6
D18	15	2	17	6	5	6	4	0	5	2	3	1	2	0.3	0.8	6
D19	9	3	12	3	4	2	10	6	4	3	2	1	3	0.4	0.9	2
D20	5	4	9	1	4	2	12	7	2	2	4	2	2	0.3	1.3	5
D21	14	1	15	2	6	4	2	7	4	3	1	2	3	0.4	0.9	4

Table 7. Subjective ratings on maintainability

Diagram	Understandability	Analyzability	Modifiability
D1	3	2	1
D2	4	2	2
D3	4	3	2
D4	2	3	2
D5	4	3	1
D6	4	2	3
D7	5	2	2
D8	2	4	2
D9	3	2	3
D10	3	2	3
D11	3	3	3
D12	4	2	3
D13	3	3	5
D14	5	3	4
D15	4	3	4
D16	5	5	2
D17	3	4	5
D18	2	4	5
D19	3	5	3
D20	4	5	3
D21	2	4	4