Using Trend Control Chart in Software Design Process Class Diagram Maintainability Monitoring

Yung-Hsiang Yang Kuo-Hsun Hsu
Department of Computer Science, National Taichung University of Education
Email: 23142162@pchome.com.tw glenn@mail.ntcu.edu.tw

Abstract

In object-oriented software design, the quality of a unified modeling language (UML) class diagram significantly affects the subsequent software design and final product quality. Therefore, if the software design issue can be detected and resolved early in the software development and design stage, maintenance costs in the subsequent software development stage can be reduced significantly.

The structural complexity and maintenance difficulty in software class diagram design stage usually increases with development time, which in turn reduces the maintainability of the software. In this study, the theory of Marcela Genero et al. is used to propose a quantified method that can measure the maintainability of a class diagram; a trend control chart is used to predict and monitor the maintainability in the class diagram design process.

From the experimental result, the maintenance difficulty in class diagram design process shows a certain trend with increase in time; the trend control chart can not only detect the cause of abnormity and delete it to enhance the quality of software design, but also be used to predict the trend of increase in maintenance difficulty.

Key words: Object-oriented design metrics, structural complexity, class diagram, control chart, maintainability.

1. Introduction

The quality and metrics of a software were important topics in software development; therefore, the enhancing software quality results in more valuable and competitive software. ISO/IEC 9126-1 has defined six quality features [1]: functionality, reliability, usability, efficiency, maintainability, and portability, wherein software maintainability refers to the capability of the software to be corrected in case of an error in the software or a change in the demand and functional specification. Today, software system functions are more diverse, and the complexity continually increases; therefore, if emphasis is only laid on the functional analysis, and the software maintainability quality is neglected, the subsequent maintenance will be considerably expensive.

Generally, the complexity of software increases with time; therefore, subsequent maintenance becomes more difficult. During software development, particularly in the design stage, a method could assess whether software maintenance difficulty grows normally. Thus, when abnormal growth or abnormal phenomenon occur, immediate diagnosis and root cause discovery can be conducted for taking the corrected action, which will reduce maintenance costs significantly. Therefore, this study considers class diagram maintainability in the software design stage as research target. Maintainability is an external quality attribute that can only be measured when the object oriented (OO) software product is (nearly) finished [2]. Therefore, creating early indicators of such quality is necessary. In literature, we found that the structural complexity of a class diagram generated in the design stage is highly related to software maintainability [3], and therefore, the structural complexity of the class diagram can be treated as an index of class diagram maintainability. Thus, through the quantification of structural complexity of the class diagram, its maintenance difficulty can be assessed.

In this study, a control chart is used as a quality control tool for the class diagram during software design; in other words, the maintenance difficulty data during development will be collected to be plotted into the control chart. Then, through a graphic display, the development personnel can clearly see any abnormal growth or abnormal phenomenon in the maintenance difficulty with increase in development time. This way, the development personnel can detect the problems earlier and correct them; therefore, massive loads in the subsequent maintenance and management can be greatly reduced.

This paper is divided into five sections: Section 2 introduces related background knowledge including software quality and software metrics, statistical process control and control chart, etc. Section 3 introduces the research method, including requirement elicitation and modeling and +plotting and quantification of class diagram, meanwhile, the quantified values were filled up into control chart to set up control chart. In section 4, a case of EZ class scheduling system example for elementary school is used, and the example is used to describe the operation flow. Section 5 concludes the paper by describing the research results of this study, its contribution, and future research direction.

2. Introduction of background knowledge

In this section, software quality model, software metrics, statistical process control, and control chart related knowledge is introduced.

2.1 Software quality and software metrics

2.1.1 Software quality

Every scholar has different definition of software quality. According to Schulmeer & McManus, software quality was defined as the capability of the entire function and characteristic of software product to satisfy the expected need [4] .

According to Deutach & Willis, software quality can be divided into the following two types [5]:

Process quality: This is the quality of a software product in the development process and is related to the general applications of important resources such as technology, tool, personnel, organization, and equipment.

Product quality: This refers to the characteristics of a software product such as integrity, correctness, and readability of document, traceability of design, reliability of code, and coverage of test.

2.1.2 Metrics of software

In software engineering research, software metrics are very important. According to DeMarco [6], "You cannot control what you cannot measure, and you cannot predict what you cannot measure." Therefore, the importance of software metrics is evident.

Metrics are indices of a type of quantification or symbol and are used to represent the characteristics of a certain object or abstract concept in the real world [7]. The real object attributes of software metrics can be divided into internal and external attributes. Internal attributes can be measured directly to obtain a measured value such as the number of classes of an object or coupling. External attributes such as software product quality cannot be measured directly; hence, we need to understand the relationship between internal and external attributes of software[8].

Can the measurement of internal software attributes be used to predict external software quality attributes? For measurements of internal characteristics to effectively predict external software characteristics, the following three conditions must be satisfied [8]:

1. Internal characteristics must be correctly measured.
2. The metrics that can be used to measure a certain relationship must exist in the external behavior characteristics.
3. Such a relationship must be understood and mathematically represented or modeled.

2.2 Statistical process control

Statistical process control includes all the tools used for quality improvement. It is a set of methods used to achieve process stability and enhance process capability through the reduction of variability [9, 10]. Statistical process control and its corresponding control charts were developed by Walter A. Shewhart in 1920 to effectively control product cost and quality.

2.3 Control Chart

This section introduces the control chart structure along with the control chart purpose, trend control chart introduction, reasonable sampling on the sample, and assessment method of the control chart.

1. Structure of control chart
   A traditional control chart has a central line (CL); at the upper and lower sides of the CL are control limits, and an upper control limit (UCL) was set at three standard deviations added to the mean value. At three standard deviations subtracted from the mean value was set the lower control limit (LCL).
2. Purpose of control chart
   The major purpose of control chart [11] was to check for an assignable cause in the process. Because the existence of an assignable cause will affect product quality, the cause must be detected, and an appropriate method must be selected to delete it; if there is no assignable cause, the process is stable and stays in a state of control.
3. Trend control chart
   Control charts can be divided into control charts for variables and control charts for attributes. A control chart for variables uses data measured using a measurement tool, and that for attributes uses data that almost appears in the form of counting. There are many types of control charts, and most of them are used when the mean values are under controlled state, among them, one control chart was used in the case when process mean values ascends or descends with time, which was called trend control chart. The central line of trend control chart was a straight line with slope not equal to zero, which can be represented by equation Xi = mt + b, and control limits were calculated as follows [12]:

$$m = \left[\sum tX - \left((\sum t \sum X)/k\right)\right] / \left[\sum t^2 - \left((\sum t)^2/k\right)\right]$$

$$b = \bar{X} - m\left(\sum t_i\right)/k$$

$$CL = X_i = mt + b$$

$$UCL_x = X_i + A_2 \overline{R}$$

$$LCL_x = X_i - A_2 \overline{R}$$

Description of formula symbol:
ti = time(i) = 1,2,3,4....k
x = each measured value
k = group number
n = sample number in the group
m = slope
b = t-intercept (X_i value at t=0)
A_2 = value checked from table = 0.577

4. Sampling method
   Rational subgroup must be considered during sampling; that is, the variance within the sample should be as small as possible. The variance between samples should be as large as possible. Such a sampling method was also called instant-time method, with sampling method as shown in Figure 1.

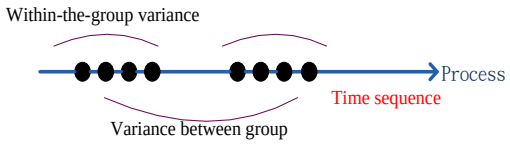


Figure 1 Instant-time method

According to a suggestion made by Leonard A. Doty[13], X chart was suggested to have a sample number of 1 and group number of 25.

5. The assessment method of control chart
   Many researchers used real examples to study the control chart, and they also studied the possible occurrence of assignable causes. There were several test rules to be used in assessing abnormity mode and assignable cause. In the Handbook of Western Electronic, four effective tests were proposed [14-16] :

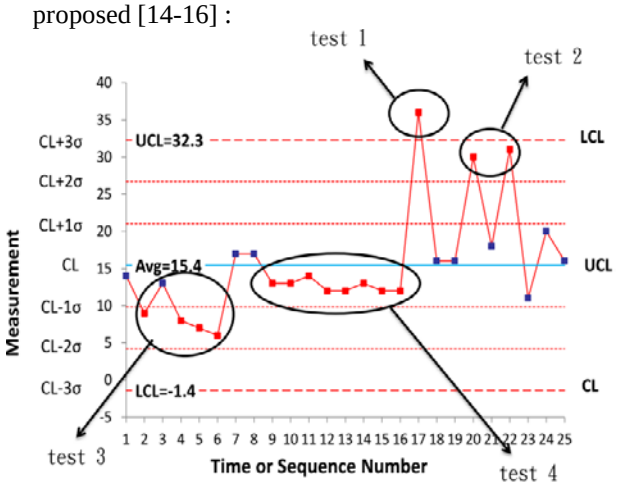


Figure 2 Four effective tests for unstable process

Test 1: One point falling out of the 3-sigma control limit.

Test 2: From three consecutive points, at least two points falling at the same side of the central line; their distances to the central line were greater than 2-sigma units.

Test 3: From five consecutive points, at least four points were falling at the same side of the central line, and their distances to the central line were greater than 1-sigma unit.

Test 3: At least eight consecutive points were falling at the same side of the central line.

The above Figure 2 represents the judgment method of these four tests. Tests 2, 3 and 4 were called run tests. [15, 16].

3. Research method

The purpose of this research was to apply the trend control chart in monitoring the maintainability of the class diagram in software design to assess if its maintenance difficulty can grow normally with time. This can help development personnel, in the development process, to detect any abnormal phenomenon and improve it to enhance software quality.

3.1 Flow chart of method

This research method started with software requirement elicitation and modeling to drawing a class diagram and quantifying it, and with the monitoring of the trend control chart, the maintainability trend of class diagram was assessed. The flow chart of this research method is shown in Figure 3.

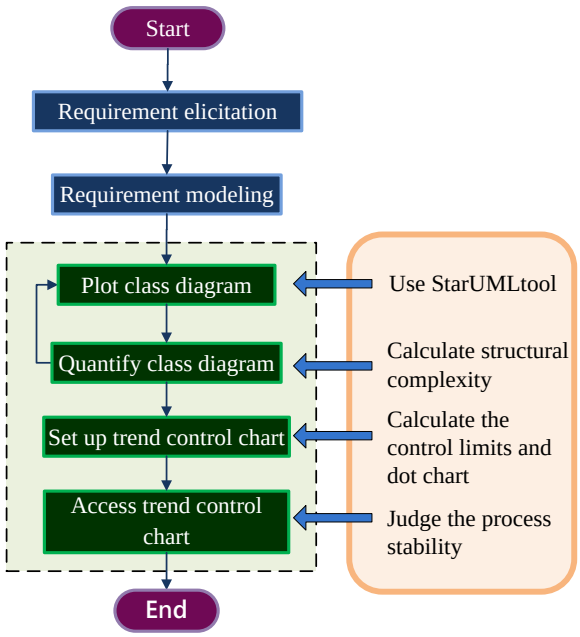


Figure 3 Flow chart of method

3.2 Requirement elicitation

In software development, requirement elicitation is the most important stage, which is also the most prone to errors [17]. The purpose of requirement elicitation was to obtain the user's need from the system, which was also called requirements gathering. The method for requirement elicitation includes interview, survey questionnaire, user observation, forum, brain storming, use case, and prototyping model [18]. All these methods could be used individually or in combination.

3.3 Requirement modeling

In this study, a use case diagram is used for

requirement modeling. At this stage, the use case document and use case diagram are generated. The steps to create a use case diagram are as follows: Identify the actor, identify out the user case, describe the use case, identify the relationship among use cases, and plot the use case diagram.

3.4 Plotting a class diagram

A class diagram describes a set of system classes, including class attributes and methods and the correlation among classes. The classification of a class includes the following: [19]
The class related to a problem field, called a field problem class, class used to deal with business logic, interface class related to user interface, data access class used for data access, and abstract concept. In this paper, the field problem class is investigated and plotted; the others are not within the scope of this research.

3.5 Quantified class diagram

This study has referred to research from Marcela Genero et al. [3] on the maintainability of a class diagram. For the effect of structural complexity of the class diagram on maintainability, a total of 11 metrics indexes of structural complexity for a class diagram were proposed, and they are mentioned in

Table 1. Marcela Genero et al. proposed 28 class diagrams and found 17 subjects. Each diagram had a test enclosed that included the description of three maintainability sub-characteristics: understandability, analyzability, and modifiability. Each subject had to rate each sub-characteristic using a scale consisting of seven linguistic labels (extremely difficult, very difficult, a bit difficult, neither difficult nor easy, quite easy, very easy, and extremely easy). They calculated the median of the subjects' ratings for each maintainability sub-characteristic. Using Spearman's rank correlation coefficient, each metric was correlated separately to the median of the subject's rating of understandability, analyzability, and modifiability. The analysis result is shown in Table 2.

Table 1 Metrics for UML class diagram structural complexity [3]

Metric name	**Metric definition**
NUMBER OF CLASSES (NC)	The total number of classes.
NUMBER OF ATTRIBUTES (NA)	The total number of attributes.
NUMBER OF METHODS (NM)	The total number of methods.
NUMBER OF ASSOCIATIONS (NAssoc)	The total number of associations.
NUMBER OF AGGREGATION (NAgg)	The total number of aggregation relationships within a class diagram.
NUMBER OF DEPENDENCIES (NDep)	The total number of dependency relationships.
NUMBER OF GENERALISATIONS (NGen)	The total number of generalisation relationships within a class diagram.
NUMBER OF AGGREGATIONS HIERARCHIES (NAggH)	The total number of aggregation hierarchies within a class diagram.
NUMBER OF GENERALISATIONS HIERARCHIES (NGenH)	The total number of generalisation hierarchies within a class diagram.
MAXIMUM HAGG (MaxHAgg)	It is the maximum of the HAgg values obtained for each class of the class diagram. The HAgg value for a class within an aggregation hierarchy is the longest path from the class to the leaves.
MAXIMUM DIT (MaxDIT)	It is the maximum of the DIT (Depth of Inheritance Tree) values obtained for each class of the class diagram. The DIT value for a class within a generalisation hierarchy is the longest path from the class to the root of the hierarchy.

Table 2 Spearman's correlation between the metrics and understandability, analyzability and modifiability [3]

	NC	NA	NM	NAssoc	NAgg	NDep	NGen	NAggH	NGenH	MaxHAgg	MaxDIT
Understandability	0.912	0.892	0.859	0.775	0.789	0.554	0.833	0.683	0.857	0.718	0.677
Analysability	0.926	0.896	0.883	0.724	0.812	0.529	0.848	0.693	0.863	0.684	0.759
Modifiability	0.943	0.907	0.909	0.730	0.788	0.525	0.881	0.676	0.891	0.673	0.805

Table 2 shows that not every metric index has the same level of effect on maintainability. Therefore, in this study, weighting was set up for the effect of each metric index on maintainability. The setup weighting was used to quantify the maintainability of the class diagram. The method of weighting setup was as follows:

1. The weights of the sub-characteristics of maintainability: Whether the importance of understandability, analyzability, or modifiability is the same as weighting, different designers have different views; therefore, researchers thought these three sub-characteristics were equally important; consequently, they have the same weighting.
2. There were correlation coefficients among the sub-characteristics of maintainability (understandability, analyzability, and modifiability), for each metric index, these three correlation coefficients were added, and then divided by three, to obtain a mean value, and this mean value was used as the effect level of the metrics index on maintainability.
3. After acquiring the effect level of individual metric

index on maintainability, these level data were adjusted, and the lowest value was used as the divisor. Then, each level data was divided by the divisor, and the answer was rounded off to two digits after the decimal point. After the adjustment, the proportion between values was maintained constant; therefore, the weightings of the effect of the metrics index on maintainability were then as shown in Table 3.

Table 3 Weighting for the influence of metrics index of class diagram on maintainability

Class diagram metrics index	NC	NA	NM	NAssoc	NAgg	NDep	NGen	NAggH	NGenH	Max HAgg	Max DIT
Weighting	1.73	1.68	1.65	1.39	1.49	1.00	1.59	1.28	1.62	1.29	1.39

4. After acquiring the effect weighting of each metric index on maintainability, the maintainability of the class diagram can then be quantified, and the quantification steps are as follows (For the example of calculating of quantification, please see Table 4):
   (1) Analyze the class diagram metrics index; then, calculate quantity on these 11 class diagram metrics indexes.
   (2) Multiply the quantified value by the weighting of each metric index.
   (3) Add the obtained 11 data items to get one value, which is defined as the maintainability value of the class diagram (The higher this value, the lower is the maintainability).

Table 4 Example for quantifying maintainability of class diagram

Internal structure of class diagram	NC	NA	NM	NAssoc	NAgg	NDep	NGen	NAggH	NGenH	MaxHAgg	MaxDIT
Internal structure calculation (1)	9	13	18	5	4	1	2	1	1	0	0
Weighting (2)	1.73	1.68	1.65	1.39	1.49	1.00	1.59	1.28	1.62	1.29	1.39
(1)*(2)	15.57	21.84	29.7	6.95	5.96	1.00	3.18	1.28	1.62	0	0
Quantified data = ∑ (1)*(2)	87.1										

3.6 Set up trend control chart

After quantifying the quality at each stage of the class diagram, the trend control chart setup was then started, and the setup steps were as shown in Figure 4.

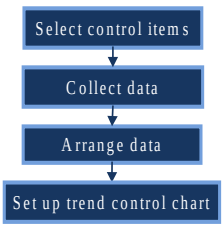


Figure 4 Setup steps for trend control chart

1. Select control items: This study is intended at investigating the trend change in the maintainability of a class diagram with development time; therefore, the maintainability of the class diagram was selected as the control item.
2. Collect data: Generally, more than 25 sets of quantified data are collected, and the sequence points are obtained according to the data to be plotted on the control chart.
3. Arrange data: Organize the collected data according to sampling time.
4. Set up trend control chart: Record the sample data to the data record table; next, calculate and plot the control limit, and then, plot the values of sampling statistical quantities on the control chart.

3.7 Assessment and cause analysis on control chart

The assessment method of the control chart was as shown above; please refer to four effective tests in the Handbook of Western Electronic to find out assignable causes and avoid them.

4. Application case: example from the class scheduling system of EZ elementary school

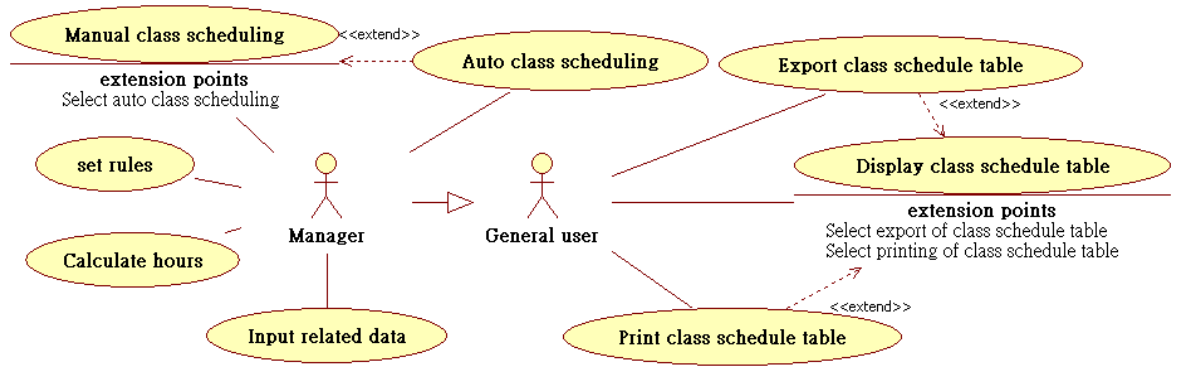
The purpose of this software was to design a class scheduling system exclusively for an elementary school to help the teaching affairs group leader to use a faster and more accurate way of completing class scheduling. A concise interface was designed to enable easy usage to reduce effort and time consumption.

4.1 Requirement elicitation and modeling

In this study, the interview method was used to acquire requirement. The researcher of this study (studied at the Department of Computer Science at National Taichung University) interviewed the teaching affairs group leader (with four years of experience being the teaching affairs group leader in an elementary school) to follow his recognition on the class scheduling of the elementary school and

actual operation situation to propose the system requirement. Modeling and description were accompanied with creating the use case diagram. The related requirements were summarized as shown in Figure 5:

Use Case Diagram of WEB version of the requirement of EZ elementary school class scheduling system



Name of use case (UC)	Brief description
1. Input related data	The user can input class scheduling related data (including the assignment of manager), meanwhile, the input data was also inspected to see if there is any conflict. Meanwhile, the input data can be changed or removed.
2. Setup of rules	The user can set up the class scheduling rule.
3. Manual class scheduling	For more detailed rule or special rule, self-arrangement of class was possible. After automatic class scheduling, for some unsatisfied class schedule, manual class scheduling was possible.
4. Automatic class scheduling	Subtracting the manual scheduled class or without any manual scheduling, the rest of classes can be automatically scheduled according to the setup rules, and for the unsatisfied class schedule, automatic class scheduling can be conducted again.
5. Calculation of hours	Calculation can be conducted on specific teacher for total hours, class count and hourly pay taught by the specific teacher in a semester.
6. Display of class schedule table	It can display the schedule table of the selected teacher or class.
7. Printing of class schedule table	It can print the schedule table of the selected teacher or class.
8. Export of class schedule table	It can export the class schedule table of the selected teacher or class, and the export format can be word, excel and PDF, etc. program files.
Remark	Manager inherits the general user (role generalization). In addition to his own use case, the manager can also participate in the use case of the general user.

Figure 5 Use Case Diagram of the WEB version of the requirement of EZ elementary school class scheduling system

4.2 Plotting and quantification of class diagram

A class diagram was usually used to express the internal structure of the system, and its main purpose was to set up the static view model of the software system. The static structure of the system affects its capability to adapt to change in future. The design of a class diagram of the WEB version of EZ elementary school class scheduling system was as shown in Figure 6.

As the sample count should at least be 25, the sampling time should not be large. According to the class diagram plotting experience of researcher, an interval of 20 minutes in the design stage of the class diagram was quantified and used to plot class diagrams, consisting of 30 stages. The value of maintainability of the class diagram in each stage was as shown in Table 5. (The higher this value, the lower is the maintainability, and consequently, the higher is the maintenance difficulty)

Table 5 Value of maintainability of class diagram in each stage

Stage	1	2	3	4	5	6	7	8	9	10
Value	24.43	25.11	38.31	46.56	54.55	67.75	79.1	80.87	90.74	85.76

Stage	11	12	13	14	15	16	17	18	19	20
Value	97.74	97.74	107.7	117.6	140.79	143.8	142.21	136.76	138.49	156.16

Stage	21	22	23	24	25	26	27	28	29	30
Value	150.34	160.97	206.64	223.02	227.92	219.92	234.18	204.03	204.93	204.93

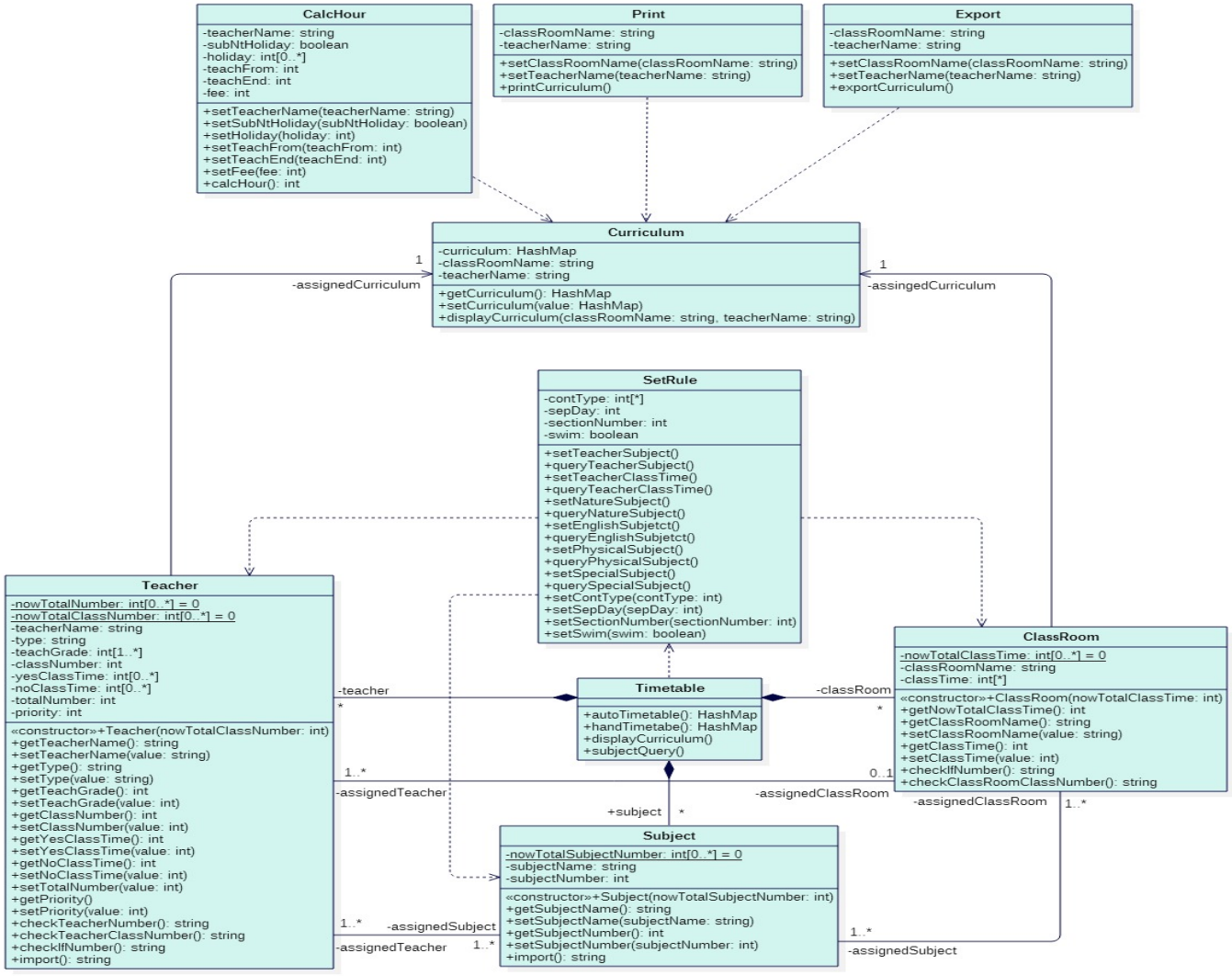


Figure 6 class diagram of the WEB version of EZ elementary school class scheduling system

4.3 Set up trend control chart

According to the collected and quantified data, the trend control chart plotted was as shown in Figure 7.

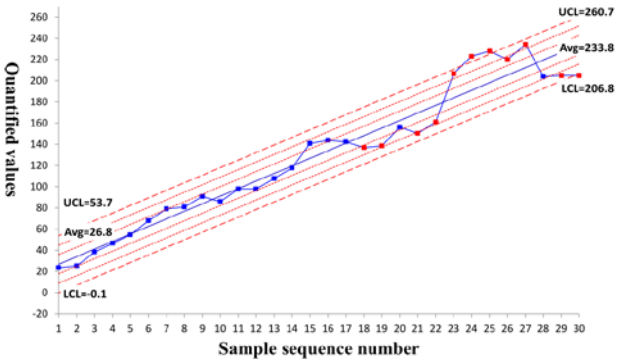


Figure 7 Trend control chart

4.4 Assessment and cause analysis of control chart

The assessment method of the control chart was as described above; please refer to four effective tests proposed in the Handbook of Western Electronic to analyze abnormity phenomenon and cause:

1. Test 3 situation appears in sample numbers 18, 19, 20, 21, and 22: From five continuous points, at least four points falling at the same side of the central line, and their distances to the central line greater than 1-sigma unit.
   Cause analysis: At this stage, researchers perform refactoring on the system; therefore, some classes, attributes, and operations are deleted. At a sample count of 20, the class was once increased; however, this seems unnecessary, because this will reduce the cohesion of the system.
2. Test 3 situation appears similarly in sample numbers 23, 24, 25, 26, and 27: From five continuous points, at least four points falling at the same side of the central line, and their distances to the central line greater than 1-sigma unit.
   Cause analysis: In the sample numbers between 22 and 23, the researcher started plotting the class that

was easier to set up (CalcHour, Print, Export); this has led to fast growth on the quantified value, and the sample numbers 24 and 25 even fell out of the control limit of 3-sigma.

3. Test 2 situation appears in sample numbers 29 and 30: From three continuous points, at least two points falling at the same side of the central line and their distances to the central line greater than 2-sigma units.
   Cause analysis: From sample numbers 27 to 28, it has entered the final stage; final inspection on the system was conducted, and unnecessary attributes and methods were deleted. Therefore, at sample numbers from 27 to 28, a large descent can be seen, and at sample numbers 29 to 30 was the final inspection stage; therefore, there was no change in the quantified value.

5. Conclusion

This study showed that designing a class diagram is different from manufacturing a product in a factory, in which maintaining control items stay within control status is easy. The change in maintainability of a class diagram, however, varies considerably with time; the increase is not stable, and its causes include, for example, the designer being tired, system refactoring, insufficient skill (during designing, the designer still had to refer to data), or sometimes a lazy designer. From the trend control chart of this study, the trend change in maintainability can be seen, and the assignable causes can be analyzed so as to avoid these in the future.

In this research, it was found that it was less representative if an individual case was used to plot the trend control chart, meanwhile, other insufficient parts found can all be used as research topics to be studied in the future; therefore, suggested future research can in the following direction.

1. After system refactoring, it will have large effect on maintainability; therefore, the effect and change generated by trend control chart after refactoring and the time appropriate for refactoring can be investigated.
2. Together with the move range control chart (Rm Chart), the relationship and effect with maintainability of class diagram can be investigated.
3. If quantification can be made on models in other design stages, for example, activity diagram and sequence diagram, and if generalized assessment can be made on the data obtained through control chart analysis, comprehensive assessment on the software system can be known.

References

[1] ISO/IEC9126-1. (2001). *Software engineering - product quality - part1: Quality Model*.

[2] L. Briand, E. Arisholm, S. Counsell, F. Houdek, and P. Thévenod–Fosse, "Empirical studies of object-oriented artifacts, methods, and processes: state of the art and future directions," *Empirical Software Engineering*, vol. 4, pp. 387-404, 1999.

[3] M. Genero, M. Piattini, and C. Calero, "Empirical validation of class diagram metrics," in *Empirical Software Engineering, 2002. Proceedings. 2002 International Symposium n*, 2002, pp. 195-203.

[4] G. G. Schulmeyer and J. I. McManus, *Total Quality Management for Softwart Quality*. Southern Africa.: Internaltional Thomson Pub., 1996.

[5] M. S. Deutsch and R. R. Willis, *Software quality engineering: a total technical and management approach*: Prentice-Hall, Inc., 1988.

[6] T. DeMarco, *Controlling Software Projects: Management, Measurement, and Estimates*: Prentice Hall PTR, 1986.

[7] 林信惠, 黃明祥, and 王文良, *軟體專案管理*. 臺北: 智勝文化出版社, 2002.

[8] I. Sommerville, "Software Engineering. International computer science series," *ed: Addison Wesley*, 2004.

[9] D. C. Montgomery, G. C. Runger, and N. F. Hubele, *Engineering statistics*: John Wiley & Sons, 2009.

[10] D. C. Montgomery, *Introduction to statistical quality control*: John Wiley & Sons, 2007.

[11] 房克成 and 林清風, *管制圖與製程管制*. 臺北市: 中華民國品質協會, 2010.

[12] I. KnowWare International. (2016, 4 月 27 日). *SPC Software for Excel | Six Sigma Software | QI Macros*. Available: https://www.qimacros.com/control-chart-formulas/xmr-trend-chart-formula/

[13] L. A. Doty, *Statistical process control*: Industrial Press Inc., 1996.

[14] H. Bajaria, "AT&T Statistical Quality Control Handbook," ed: Western Electric, 1985.

[15] D. J. Wheeler and D. S. Chambers, *Understanding statistical process control*: SPC press, 1992.

[16] D. J. Wheeler, *Advanced topics in statistical process control* vol. 470: SPC press Knoxville, TN, 1995.

[17] 吳仁和 and 林信惠, "系統分析與設計—理論與實務應用 (四版), 台北市: 智勝文化," 2010.

[18] 李允中, "軟體工程," *臺北市, 臺灣: McGraw-Hill*, 2009.

[19] 游峰碩 and 資訊管理, *UML 物件導向系統分析與設計第二版*: 博碩文化, 2011.