Joint Conference of 1996 International Computer Symposium
December 19~21, Kaohsiung. Taiwan, R.O.C.

A Validation of Coupling and Cohesion Metrics for Object-oriented programs

Shu-Fen Wu Bin-Shiang Liang Yen-Sung Lee* Feng-Jian Wang

Institute of Computer Science and Information Engineering
National Chiao-Tung University, Taiwan, R. O. C.

*Information System Department
Chung-Hwa Telecom *Cooperation LTD.*

Abstract

The empirical evaluation of software metrics is often questioned for lack of experimental rigor [8]. Most validations of previous software metrics were relied only on the analysis of correlation degree between metrics and software quality. The reliance, based solely on the correlation data, may be limited [4] [8]. By combining the rigorous experiment framework [1] and the validation methodology [12], in the paper, we propose a validation approach specifically for coupling and cohesion metrics of object-oriented software. With four phases of an experiment: definition, planning, operation and interpretation [1], our approach can help structure experiment process and provide a classification scheme for understanding and evaluating the experiment. In the planning phase, two kinds of direct reflectors for coupling and cohesion of object-oriented software respectively are proposed. A set of statistical models is provided for complete analysis between the direct reflectors and the metrics. With the analysis, the properties of quality assessment, quality control and quality prediction of a metric are evaluated properly.

Following the proposed validation approach, we conducted an experiment to evaluate the effectiveness of several existing coupling and cohesion metrics of object-oriented software empirically. The experimental results indicate that message passing is the major contribution of coupling of object-oriented software.

Keywords: software metrics, metrics validation, coupling and cohesion, object-oriented programs, experimental design.

1. Introduction

One major reason for lack of widespread acceptance of experimental studies for validating software metrics is the lack of rigorous study or design for the experiment. The study in an experiment includes pre-experiment design, operational definitions, experiment method, data collection, program sizes, program sample sizes, statistical validity, result interpretation [8]. Besides, most empirical evaluations of previous metrics were relied only on the analysis of correlation degree between software quality factors and metrics. The reliance, based solely on the correlation analysis, may be limited [4] [8]. Basili et al. [1] proposed a rigorous framework for the experiments in software engineering. Schneidewind [12] proposed a validation methodology providing complete theory bases of statistical analysis. Combining Basili's framework and Schneidewind's methodology, we propose a validation approach specifically for coupling and cohesion metrics of object-oriented software.

Based on Basili's framework, an experiment in our approach contains four phases: *definition*, *planning*, *operation* and *interpretation*. In the definition phase, six parts need to be specified: *motivation*, *object*, *purpose*, *perspective*, *domain* and *scope*. The planning phase consists of three parts: *design*, *criteria* and *measurement*. *Preparation*, *execution* and *data analysis* constitute the operation phase and the interpretation phase comprises of *interpretation*, *extrapolation* and *impact*. In this way, our approach helps structure the experiment process and provides a classification scheme for understanding and evaluating experimental studies.

In our approach, the validation for software metrics is achieved through the statistical analysis of the relationship between the quality factors used to reflect coupling and cohesion directly and the metrics. In the criteria part of the planning phase, we propose two kinds of direct reflectors for coupling and cohesion in a class. These reflectors are concerned with the modifications of a class in testing and maintenance phases. The direct reflectors for coupling and cohesion in a class hierarchy are defined based on those in classes within the hierarchy. In the design part of the planning phase, adopting four validity principles in the validation methodology [12], our approach provides a set of complete statistical models to analyze the relationship between these direct reflectors and the metrics. With the analysis, the properties of quality assessment, quality control and quality prediction of a metric are evaluated properly.

Following the proposed validation approach, we conducted an experiment to validate several existing coupling and cohesion metrics of object-oriented software. Twenty two subjects, including undergraduate and graduate, were asked to implement a simulation for financial management of a company. Totally, 174 classes, 23 class hierarchies, and 17 programs were collected. The coupling and cohesion metrics, analyzed through four validity principles in the experiment, include the metrics proposed in [6], [2], [7], and [3]. The experimental results indicate that message passings are the major source of coupling in object-oriented software. Information flow based metrics [6] is better than the others because the metrics consider the amount of arguments in a message passing in addition to the number of message passings. The other metrics satisfy the four principles also, however, they show weak validity.

In the following, we will describe the background of this research in section 2, and present our validation approach and the experiment in section 3 and 4 respectively. Section 5 concludes all the work.

2. Background

2.1 Experiment Framework

Basili et al. proposed an experiment framework [1], which contains four categories corresponding to phases of the experimentation process: *definition*, *planning*, *operation*, and *interpretation*.

The first phase *definition* of an experiment usually consists of six parts: 1) motivation, 2) object, 3) purpose, 4) perspective, 5) domain, and 6) scope. Three aspects, design, criteria and measurement, constitute *planning*. *Operation*, inclusive of preparation, execution and analysis, is the work in the third phase of an experiment. The last phase of an experimental study is *interpretation*, which comprises of interpretation context, extrapolation and impact.

2.2 The Validation Methodology

Schneidewind proposed a methodology for validating software metrics [12]. In the methodology, the quality factors are first defined as a direct indicator for the software quality which the validated metrics may predict. Secondly, the values of the metrics and the data of the quality factors for each measured entity (maybe a class or a class hierarchy) are collected phase by phase. At last, the validity principles, the relationships between the quality factors and the validated metrics, provide the rationale for validating metrics.

A metric should perform quality functions to achieve project quality goals. The quality functions concerned with this methodology are assessment, control and prediction. *Quality Assessment* is the evaluation of the relative quality of software components. The purpose of *assessment* is to provide software managers with a rational basis for assigning priorities to improve quality or allocate resources. *Quality Control* is the evaluation of software components against predetermined critical values of metrics and identification of components that fall outside quality limits. The purpose of *control* is to allow software managers to identify software which has unacceptable quality sufficiently early in the development process to take a corrective action. *Quality Prediction* is a forecast of the values of quality factors based on the values of the metrics in the earlier phase.

The validity principles are defined to support assessment, control and prediction functions for the validity of a metric.

Correlation: The variation in F explained by the variation in M, which is given by R^2 (coefficient of determination), where R is the linear correlation coefficient, must exceed a specified threshold, or $R^2 > \beta_a$, with specified α. In other words, R^2 measures the degree in which the following holds.

Magnitude $[M_i < M_j] \Leftrightarrow$ Magnitude $[F_i < F_j]$

This criterion supports the quality assessment property.

Consistency: The rank correlation coefficient r between F and M must exceed a specified threshold, or $r > \beta_c$, with specified α. In other words, r measures the degree in which the following holds.

Rank $[M_i < M_j] \Leftrightarrow$ Rank $[F_i < F_j]$

This criterion also supports the quality assessment property.

Discriminative Power: The critical value of a metric M_c must be able to discriminate for a specified F_c between elements (components 1, 2,..., i,...n) of vector F in the following way:

$M_i > M_c \Leftrightarrow F_i > F_c$ and $M_i \le M_c \Leftrightarrow F_i \le F_c$

This criterion supports the quality control property.

Predictability: There exists a function of M, $f(M)$, where M is measured at time T_1, which can predict F, measured at time T_2. This criterion supports the quality prediction property.

3. Our Proposed Validation Approach

In our approach, an experiment is constructed with four phases [1]: *definition*, *planning*, *operation* and *interpretation*. In the definition phase, six parts need to be specified: *motivation*, *object*, *purpose*, *perspective*, *domain* and *scope*. The planning phase consists of three parts: *design*, *criteria* and *measurement*. *Preparation*, *execution* and *data analysis* constitute the operation phase and the interpretation phase comprises of *interpretation*, *extrapolation* and *impact*. Table 3.1 shows our validation approach.

In our approach, the validation for software metrics is achieved through the statistical analysis of the relationship between the quality factors for coupling and cohesion and the metrics. In the criteria part of the planning phase, we propose two kinds of direct reflectors for coupling and cohesion respectively for a class and for a class hierarchy, which will be presented in section 3.1. In the design part of the planning phase, adopting four validity principles in the validation methodology [12], our approach provides a set of complete statistical models to analyze the relationship between the direct reflectors for coupling and cohesion and the metrics. We will present these statistical models in section 3.2.

3.1 The direct reflectors

In Basili's framework, there need criteria that directly reflect coupling and cohesion of software which the metrics intend to capture. Our validation approach is emphasied specifically for validating coupling and cohesion metrics of object-oriented software so the direct reflectors in this approach are defined for the quality of coupling and cohesion of object-oriented software. Change is considered an attribute which can reflect coupling most [15]. The direct reflectors defined in the approach are mainly concerned with the modification of classes. They include the changes made in other classes (or the class itself) when 1) cleaning the bugs in one class in the testing phase or 2) modifying the requirements of one class in the maintenance phase. The direct reflectors of coupling are defined as "*ctcp*" (changes made in the testing phase for coupling) and "*cmcp*" (changes made in the maintenance phase for coupling). The followings list the way how we collected the data about *ctcp* for a class, *cmcp* for a class, *ctcp* for a class hierarchy, and *cmcp* for a class hierarchy.

ctcp for a class named C: When there are errors in class *C* or the functions performed by class *C* do not meet its requirements in the testing phase, ***the number of total changes*** made in other classes of the system for cleaning the "bugs" in class *C* is *ctcp* for class *C*. (The modifications made in a class for the same small goal are counted as **one** change on collecting data. This rule is also applied to the following situations.)

cmcp** for a class named C*: The original requirements for some classes in the system are changed in the maintenance phase. ***The number of total changes accompanying the modification of class *C* due to the changes of its requirements made in other classes in the system are *cmcp* for class *C*.

ctcp** for a class hierarchy named H*: When there are errors in the classes of class hierarchy *H* or the functions performed by the classes of class hierarchy *H* do not meet their requirements in the testing phase, ***the number of total changes made in other classes outside the class hierarchy *H* for cleaning the "bugs" in the classes of class hierarchy *H* is *ctcp* for class hierarchy *H*.

cmcp** for a class hierarchy named H*: The original requirements for some classes in the system are changed in the maintenance phase. ***The number of total changes accompanying the modifications of the classes of class hierarchy *H* due to the changes of their requirements made in other classes outside class hierarchy *H* is *cmcp* for class hierarchy *H*.

Cohesion was also captured by the changes in the testing phase and the changes in the maintenance phase so the direct reflectors of cohesion are defined as "*ctch*" (changes made in the testing phase for cohesion), and "*cmch*" (changes made in the maintenance phase for cohesion). The way to collect the data about *ctch* and *cmch*, described as follows, is different from that for coupling.

ctch** of a class named C*: When there are errors in class *C* or the functions performed by class *C* do not meet its requirements in the testing phase, ***the number of total changes made in class *C* itself for cleaning the "bugs" in class *C* is *ctch* for class *C*.

cmch** of a class named C*: The original requirements for some classes in the system are changed in the maintenance phase. ***The number of total changes made in class *C* itself due to the changes of its requirements is *cmch* for class *C*.

ctch** of a class hierarchy named H*: When there are errors in the classes of class hierarchy *H* or the functions performed by the classes of class hierarchy *H* do not meet their requirements in the testing phase, ***the number of total changes made in the classes of class hierarchy *H* for cleaning the "bugs" in these classes of *H* is *ctch* for class hierarchy *H*.

cmch** of a class hierarchy named H*: The original requirements for some classes in the system are changed in the maintenance phase. ***The number of total changes made in the classes of class hierarchy *H* due to the changes of requirements of these classes is *cmch* for class hierarchy *H*.

3.2 The statistical models

In section 2.2, we introduced four validity principles to validate the metrics in Schneidewind's methodology. Here list the statistical models on which these validity principles are based.

(1) ***Correlation***:

The statistical model for correlation analysis is called **simple linear regression** [10]. The assumption behind the model is that the distribution of the random error, ε, is NID(0, σ^2). The strength of linear relationship between the independent variable and the dependent variable is measured by *correlation coefficient* γ, whose value ranges from -1, for perfect negative correlation, to +1, for perfect positive correlation. The null hypothesis H_0: $\gamma=0$ represents there is no correlation between the independent variable and the dependent variable, and significance level α equals to 0.05. Besides, *Pearson's coefficient of determination*, γ^2, which also acts as an index of the degree of correlation, represents the variation in the dependent variable explained by the variation in the independent variable.

(2) ***Consistency***:

The statistical model for consistency analysis is **Spearman's rank correlation** [10]. It is classified into nonparametric regression. Tests of model adequacy do not require any assumptions about the probability of distribution of ε. *Spearman's correlation coefficient* γ_s, an alternative to *Pearson's correlation coefficient*, is based on ranks. The larger the absolute value of γ_s, the stronger the relationship between the ranks of the independent variable and the dependent variable. The null hypothesis H_0: $\gamma_s=0$ represents there is no consistency between the independent variable and the dependent variable, and significance level α equals to 0.05.

(3) ***Discriminative power***:

We used the **contingency table** to analyze discriminative power of the metrics [9]. The values of each variable are classified into two categories according to their critical values M_c and F_c (M_c is the critical value for the metric and F_c is the critical value for the direct reflector). The value in a cell is the count of the category for its corresponding row and its corresponding column and the percentage of the count for the total observations. The objective of such a classification is to determine whether the two directions of classification are independent.

χ^2-statistic is used to test the null hypothesis that the two classifications are independent against the alternative hypothesis that the two classifications are dependent, and significance level α equals to 0.05. Large values of χ^2 imply that the observed counts do not closely agree and therefore the two classifications are not independent. The success rate, which is a summation of the count percentages for the total observations in the correct classifications, also acts as an index of the independence of two classification. The larger the success rate, the more dependent two classifications.

(4) ***Predictability***:

The statistical model of predictability analysis is an extension of that of correlation analysis: **multi-**

linear regression. If a regression equation can be established, we can assert the metric possesses predictability about the corresponding direct reflector. First, the basic linear model of regression for predictability is:

$$DV = \beta_0 + \beta_1 IDV_1 + \beta_2 IDV_2 + \ldots + \beta_n IDV_n$$

where β_i is coefficient of IDV_i. We define the null hypothesis $H_0 : \beta_0 = \beta_1 = \ldots = \beta_n$ (that is, there does not exist a linear relationship between dependent variables and independent variables), and significance level α=0.05. The level of strength can be observed from γ or γ^2 value.

Table 3.1 Our validation approach

I. Definition (I)		
Motivation	***Object***	***Purpose***
Validate the coupling and cohesion metrics	Object-oriented programs	Evaluate the relationship between the coupling and cohesion metrics and the direct reflectors
I. Definition (II)		
Perspective	***Domain***	***Scope***
Developer	Programmer Program / Project	Single project Multi-project Replicated project Blocked subject-project
II. Planning		
Design	***Criteria***	***Measurement***
Experimental designs Validity principles 1. Correlation ana. 2. Consistency ana. 3. Discriminative power analysis 4. Predictability analysis Statistical ana. models 1. Scatter plots 2. Histograms 3. Simple linear regression 4. Spearman's rank correlation 5. Chi-square contingency table 6. Multi-linear regression	Direct reflectors of quality: Coupling 1. Changes made in the testing phase. 2. Changes made in the maintenance phase. Cohesion 1. Changes made in the testing phase. 2. Changes made in the maintenance phase. Indirect reflectors of quality: Coupling and cohesion metrics	Metric definition The direct reflectors of coupling: *ctcp, cmcp* The direct reflectors of cohesion: *ctch, cmch* The validated metrics: Coupling and cohesion metrics Data collection methodology Objective vs. subjective
III. Operation		
Preparation	***Execution***	***Analysis***
Pilot study Participant training	Data collection Data validation	Preliminary analysis Primary analysis
IV. Interpretation		
Interpretation Context	***Extrapolation***	***Impact***
Statistical framework Study purpose Field of research	Sample representative	Visibility Replication Application

4. The Experiment

Following the proposed validation approach, we constructed an experiment to validate several existing coupling and cohesion metrics of object-oriented software. The coupling and cohesion metrics validated in the experiment are introduced in section 4.1, section 4.2 presents the experiment design by following our proposed approach and data analysis is shown in section 4.3.

4.1 A Survey of Coupling And Cohesion Metrics for Object-Oriented Software

In this paper, we validate three categories of metrics including cohesion metrics, coupling metrics and inheritance coupling metrics.

(1) Cohesion Metrics

- Information Flow Based Cohesion for a Class (*ICH(C)*): sum of the amount of information flow caused by the messages passed within the class, where the amount of the information flow caused by a message passing is one plus the number of arguments in the message passing [6].
- Information Flow Based Cohesion for a Class Hierarchy (*ICH(H)*): sum of the amount of information flow caused by the messages passed within the class hierarchy, where the amount of the information flow caused by a message passing is one plus the number of arguments in the message passing [6].

(2) Coupling Metrics

- Coupling between Objects (*CBO*): Coupling can exist between classes that are not related through inheritance. One class is coupled to another if the methods of the former use the methods or instance variables of the latter. *CBO* for a class is the number of classes which has coupling with the class [2].
- Response for a Class (*RFC*): The *RFC* metric value for a class is the sum of the number of its instance methods and the number of all other methods that the class directly invokes [2].
- Coupling through Message Passing (*MPC*) for a Class: When an object needs some service provided by other objects, messages are sent from the former to the latter. Message-passing coupling (*MPC*) is used to measure the complexity of message passing among classes. Since the message is defined in a class and used by objects of the class, the number of message passings is calculated at the class level instead of the object level [7].

MPC = number of ***message passings*** defined in a class.

- Coupling through Abstract Data Types (*DAC*) for a Class: The concept of an abstract data type (ADT) is discussed in [13]. A class can be viewed as an implementation of an ADT [5]. A variable defined within a class X may have a type of ADT which is declared in another class Y. This phenomenon causes a particular type of coupling between class X and class Y. The metric proposed by [7] is data abstraction coupling (*DAC*) for a class as follows:

DAC = number of ADTs defined in a class.

- Information Flow Based Coupling for a Class (*ICP(C)*): sum of the amount of information flow caused by the messages passed from the class to the other classes, where the amount of the information flow caused by a message passing is one plus the number of arguments in the message passing [6].

• Information Flow Based Coupling for a Class Hierarchy (*ICP(H)*): sum of the amount of information flow caused by the messages passed from the class hierarchy to the other class hierarchies, where the amount of the information flow caused by a message passing is one plus the number of arguments in the message passing [6].

(3) Inheritance Coupling Metrics

• Depth of Inheritance Tree (*DIT*): The *DIT* metric for a class is a measure of how many ancestor classes which can potentially affect this class. The calculation of the *DIT* metric is defined as the depth level for a class in the inheritance hierarchy [7]. The root class's DIT is zero.

DIT = *inheritance level* number in the inheritance hierarchy.

• Class Inheritance-related Coupling (*CIC*): For a class, there are usually two kinds of clients: objects that invoke operations upon instances of the class, and subclasses that inherit from the class. With inheritance, coupling will occur when a class accesses a variable or uses a member function defined in a proper ancestor class. For a class, we define a count of such accesses and uses as *Class Inheritance-related Coupling (CIC)* [3].

CIC = number of *inheritance accesses* to the variables or methods in its superclasses.

• Information Flow Based Inheritance Coupling for a Class (*IH_ICP(C)*): sum of the amount of information flow caused by the messages passed from the class to its superclasses, where the amount of the information flow caused by a message passing is one plus the number of arguments in the message passing [6].

Table 4.1 A summary for the metrics and the attributes which they measure.

Metrics/ Attributes	*Cohesion*	*Coupling*	*Inheritance Coupling*
ICH(C)	*		
ICP(C)		*	
IH_ICP(C)			*
ICH(H)	*		
ICP(H)		*	
CBO		*	
RFC		*	
MPC		*	
DAC		*	
DIT			*
CIC			*

4.2 Experiment Definition

The experiment was done on several existing coupling and cohesion metrics of object-oriented software with a given project from the perspective of developers. The subjects of the experiment are individual programmers, who repeatedly worked on the same project and the objects are the software products. The purpose is to evaluate the effectiveness how the metrics reflect coupling and cohesion of an object-oriented program.

4.3 Experiment Planning

Twenty two subjects with different programming capabilities including undergraduates and graduates developed a small project "a simulation for financial management in a company" in Turbo C++ or Borland C++ integrated environment on PC/DOS system. The experiences of the subjects are nonsensitive to the project and the implementation environment. The complexity of the project can exhibit enough coupling and cohesion to strengthen the validity of the experiment. Each subject accomplishes the project by his own design architecture.

4.4 Experiment Operation

Data forms were designed for subjects to collect the data of the direct reflectors phase by phase: *ctcp*, *cmcp*, *ctch* and *cmch*. At the end of every phase in life-cycle, subjects were asked a few questions to ensure that they did what they should do.

4.5 Data analysis

Data analysis as follows was made to observe the characteristics of the metrics.

4.5.1 Correlation Analysis

The statistical techniques usually require some assumptions. The assumptions behind simple linear regression are that the distribution of the residuals is NID(0, σ^2), i.e., the *normality with mean zero assumption*, the *constant variance assumption* and the *independence assumption* [11]. The assumptions of correlation analysis for the validated metrics and the direct reflectors hold [14].

Null Hypothesis H_0: $\gamma=0$ (Two variables are not correlated.)

Alternative Hypothesis H_a: $\gamma\neq 0$ (Two variables are correlated.)

Significance level $\alpha=0.05$

The correlation coefficient γ and the coefficient of determination γ^2 of simple linear regression for the cohesion metrics, *ICH(C)* and *ICH(H)*, are shown in Table 4.2. All the p-values are far smaller than α so H_0 is rejected; that is, the direct reflectors and the cohesion metrics are correlated. The results indicate that the correlation criterion is verified and the cohesion metrics have the quality assessment property.

Table 4.2 Results of correlation analysis for the cohesion metrics. IDV: independent variable; DV: dependent variable; p: p-value

DV\IDV	*ICH(C)*	*ICH(H)*
ctch	$\gamma = 0.48270$ $\gamma^2 = 0.23300$ $p = 0.0000$	$\gamma = 0.56105$ $\gamma^2 = 0.31477$ $p = 0.0066$
cmch	$\gamma = 0.43989$ $\gamma^2 = 0.19351$ $p = 0.0000$	$\gamma = 0.52642$ $\gamma^2 = 0.27712$ $p = 0.0171$

Table 4.3 shows the results of correlation analysis for the coupling metrics. There are five metrics (*ICP(C)*, *CBO*, *DAC*, *MPC*, *RFC*) which fully satisfy correlation criterion; that is, the quality assessment function of the five metrics is validated. The results also indicate that the metrics

defined based on message passings, *ICP(C)* and *MPC*, are correlated with *ctcp* and *cmcp* more than the metrics defined from other viewpoints.

Table 4.3 Results of correlation analysis for the coupling metrics.

DV\IDV	ICP(C)	CBO	DAC	MPC	RFC	ICP(H)
ctcp	γ = 0.68219 γ^2 = 0.46538 p = 0.0000	γ = 0.30998 γ^2 = 0.09609 p = 0.0000	γ = 0.33391 γ^2 = 0.11150 p = 0.0000	γ = 0.54917 γ^2 = 0.30159 p = 0.0000	γ = 0.43592 γ^2 = 0.19003 p = 0.0000	γ = 0.34986 γ^2 = 0.12240 p = 0.1105
cmcp	γ = 0.66545 γ^2 = 0.44283 p = 0.0000	γ = 0.38756 γ^2 = 0.15020 p = 0.0000	γ = 0.40558 γ^2 = 0.16449 p = 0.0000	γ = 0.49331 γ^2 = 0.24335 p = 0.0000	γ = 0.37245 γ^2 = 0.13872 p = 0.0000	γ = 0.20929 γ^2 = 0.04380 p = 0.3759

Table 4.4 shows the results of correlation analysis for the inheritance coupling metrics. Only metric *IH_ICP(C)* satisfies correlation criterion; that is, *IH_ICP(C)* has the quality assessment property. The other two metrics are not correlated with factors *ctcp* and *cmcp*.

Table 4.4 Results of correlation analysis for the inheritance coupling metrics.

DV\IDV	IH_ICP(C)	CIC	DIT
ctcp	**γ = 0.52146 γ^2 = 0.27192 p = 0.0000**	γ = 0.15860 γ^2 = 02515 p = 0.1132	γ = 0.09039 γ^2 = 0.00817 p = 0.3687
cmcp	**γ = 0.50466 γ^2 = 0.25468 p = 0.0000**	γ = 0.11174 γ^2 = 0.01249 p = 0.2759	γ = 0.01671 γ^2 = 0.00028 p = 0.8710

4.5.2 Consistency Analysis

Spearman's correlation is classified into nonparametric regression. It does not require the normal distribution of data.

Null Hypothesis H_0: γ_s=0 (Two variables are not correlated in ranks.)

Alternative Hypothesis H_a: $\gamma_s \neq 0$ (Two variables are correlated in ranks.)

Significance level α=0.05

Table 4.5 shows correlation coefficient γ_S of Spearman's correlation analysis for the cohesion metrics. Metric *ICH(C)* fully satisfies the consistency criterion and metric *ICH(H)* is rank-consistent with factor *cmch*. The results represent there is an 100γ_s% ranking consistency between the direct reflectors and the metrics.

Table 4.5 Results of correlation analysis for the cohesion metrics.

DV\IDV	ICH(C)	ICH(H)
ctch	**γ_s= 0.4263 p = 0.000**	γ_s = 0.3567 p = 0.103
cmch	**γ_s = 0.4990 p = 0.000**	γ_s = 0.5015 **p = 0.024**

The results of Spearman's correlation for the coupling metrics are illustrated in Table 4.6. There are five metrics (*ICP(C), CBO, DAC, MPC, RFC*) which fully satisfy the consistency criterion.

Table 4.6 Results of correlation analysis for the coupling metrics.

DV\IDV	ICP(C)	CBO	DAC	MPC	RFC	ICP(H)
ctcp	γ_s = **0.6467** p = **0.000**	γ_s = 0.2827 **p = 0.000**	γ_s = 0.2694 p = **0.000**	γ_s = 0.5524 p = **0.000**	γ_s = 0.5755 p = **0.000**	γ_s = 0.3646 p = 0.095
cmcp	γ_s = 0.5853 p = **0.000**	γ_s = 0.3669 **p = 0.000**	γ_s = 0.3908 p = **0.000**	γ_s = 0.4834 p = **0.000**	γ_s = 0.4809 p = **0.000**	γ_s = 0.2367 p = 0.315

Table 4.7 shows the results of Spearman's correlation for the inheritance coupling metrics. Only metric *IH_ICP(C)* fully satisfies the consistency criterion. Metric *CIC* is rank consistent with factor *ctcp* and *DIT* is rank consistent with factor *cmcp*.

Table 4.7 Results of correlation analysis for the inheritance coupling metrics.

DV\IDV	IH_ICP(C)	CIC	DIT
ctcp	**γ_s = 0.4977 p = 0.000**	γ_s = 0.2716 **p = 0.006**	γ_s = 0.1750 p = 0.080
cmcp	**γ_s = 0.4455 p = 0.000**	γ_s = 0.0572 p = 0.578	γ_s = 0.2668 **p = 0.008**

4.5.3 Discriminative Power Analysis

Null Hypothesis H_0: The two classifications are independent.

Alternative Hypothesis H_a: The two classifications are dependent.

Significance level α=0.05

The following shows the results of discriminative power analysis for the critical values of metric *ICH(C)*. Let the critical value of factor *ctch* be as close as its mean (0.809), F_c=1. The value of metric *ICH(C)* equal to 4 discriminates the values of factor *ctch* best because it has the largest success rate of discrimination which is the total percentage of the counts of the correct classifications. The p_value of chi-square test for $ICH_c(C)$=4 and $ctch_c$ =1 is smaller than α=0.05 so H_0 is rejected; that is, the two classifications are dependent. Similarly, let the critical value of factor *cmch* be set as close as its mean (2.903), F_c=3. A metric may have more than one critical value because the distribution of data is discrete. The critical values of metric *ICH(C)* for factor *cmch* are 2 and 3. Therefore *ICH(C)* satisfies the discriminative power criterion and it has the quality control property. A summary table of the discriminative power analysis for the metric *ICH(C)* is shown in Table 4.8. The analysis results of discriminative power for the other metrics are shown in [14].

Table 4.8 Summary of discriminative power analysis for metric *ICH(C)*

	Chi-square χ^2	p-value	Success rate
$ctch_c$=1 $ICH_c(C)$=4	14.05847	0.00018	80.9%
$cmch_c$=3	23.69906	0.00000	79.6%

$ICH_c(C)$=2, 3			

4.5.4 Predictability Analysis

Null Hypothesis H_0: $\beta_0=\beta_1= ... =\beta_n$ (There exists no linear relationship.)

Alternative Hypothesis H_a: There exists a linear relationship between ***DV*** and ***IDV***.

Significance level α=0.05

Table 4.9 shows the results of predictability analysis for the cohesion metrics. *R*-square, γ^2, is an indicator to show the accuracy degree in which the given values of the metrics can predict the factor values. In Table 4.9, all the *p*-values are smaller than α=0.05 so H_0 is rejected; that is, there are linear relationships between the factors and *ICH(C)* and between the factors and *ICH(H)*.

Table 4.9 Predictability analysis results for the cohesion metrics

D.V.	*Regression Equation*	γ	γ^2	*p_value*
ctch	=0.228704***ICH(C)*** +0.564082	0.48270	0.2330	**0.0000**
ctch	=0.59306***ICH(H)*** +0.858596	0.56105	0.31477	**0.0066**
cmch	=0.300258***ICH(C)*** +2.318853	0.43989	0.19351	**0.0000**
cmch	=0.143639***ICH(H)*** +5.554849	0.52642	0.27712	**0.0171**

Table 4.10 shows the results of predictability analysis for the coupling metrics. R-square, γ^2, is a regression quality indicator showing the capability of prediction; that is, how much variance in the dependent variable is explained by the independent variable. From Table 4.10, most metrics have predictabilities for factors *ctcp* and *cmcp*. More than 50% of the variance in factor *ctcp* is accounted for by the combination of all the coupling metrics and about 50% of the variance in factor *cmcp* is accounted for by the combination of all the coupling metrics.

Table 4.10 Predictability analysis results for the coupling metrics

D.V.	*Regression Equation*	γ	γ^2	*p_value*
ctcp	=0.057136***ICP(C)*** + 0.098730	**0.68219**	**0.46538**	0.0000
ctcp	=0.413824***CBO*** + 1.135865	0.30998	0.09609	0.0000
ctcp	=0.469927***DAC*** + 1.124611	0.33391	0.11150	0.0000
ctcp	=0.096027***MPC*** + 0.843825	0.54917	0.30159	0.0000
ctcp	=0.096913***RFC*** + 0.476119	0.43592	0.19003	0.0000
ctcp	=0.042852***ICP(C)*** - 0.061846***DAC*** - 0.190405***CBO*** + 0.007009***RFC*** + 0.081417***MPC*** + 0.053645	**0.75082**	**0.56373**	0.0000
ctcp	=0.055307***ICP(H)*** +1.859125	0.12240	0.34986	0.1105
cmcp	=0.227367***ICP(C)*** +7.616468	**0.66545**	**0.44283**	**0.0000**
cmcp	=2.133859***CBO*** +11.352922	0.38756	0.15020	**0.0000**
cmcp	=2.353763***DAC*** +11.347632	0.40558	0.16449	**0.0000**
cmcp	=0.350981***MPC*** +11.200671	0.49331	0.24335	**0.0000**
cmcp	=0.343639***RFC*** +10.074828	0.37245	0.13872	**0.0000**
cmcp	=0.173267***ICP(C)*** +0.014312***DAC*** +0.116866***CBO***- 0.009927***RFC*** +0.222151***MPC*** +7.300627	**0.70863**	**0.50215**	**0.0000**
cmcp	=0.075124***ICP(H)*** +10.386779	0.20929	0.04380	0.3759

Table 4.11 shows the results of predictability analysis for the inheritance coupling metrics. Metric *IH_ICP(C)* has a predictability for factor *ctcp* and *cmcp* than the other inheritance coupling metrics. More than 50% of the variance in factor *ctcp* is explained by the combination of all the inheritance coupling metrics and more than 60% of the variance in factor *cmcp* is explained by the combination of all the inheritance coupling metrics.

Table 4.11 Predictability analysis results for the inheritance coupling metrics

D.V	*Regression Equation*	γ	γ^2	*p_value*
ctcp	=0.026449*IH_ICP(C)* + 0.053056	**0.52**	**0.27**	**0.00**
ctcp	=0.021879*CIC* 0.176231	0.16	0.023	0.11
ctcp	=0.086179*DIT*+0.167480	0.09	0.01	0.37
ctcp	=0.034171*IH_ICP(C)* +0.140828*DIT* -0.038712*CIC*+0.885128	**0.56**	**0.32**	**0.00**
cmcp	=0.131999*IH_ICP(C)* +1.227035	**0.50**	**0.25**	**0.00**
cmcp	=-0.092459*CIC* +2.526429	0.11	0.01	0.28
cmcp	=0.098074*DIT*+2.260946	0.02	0.00	0.87
cmcp	=0.196993*IH_ICP(C)* +1.341494*DIT* - 0.449940*CIC* - 0.885128	**0.67**	**0.44**	**0.00**

5. Conclusions

Combining Basili's framework and Schneidewind's methodology, we propose a validation approach specifically to validating the coupling and cohesion metrics of object-oriented software empirically. This approach provides a rigorous experiment framework and a set of statistical models for validation. Two kinds of direct reflectors, *ctch* and *cmch* for cohesion and *ctcp* and *cmcp* for coupling, are define as criteria to reflect cohesion and coupling directly. A validation for coupling and cohesion metrics is achieved through the analysis of the relationship between the direct reflectors and the metrics by the provided statistical models in our approach.

Following the proposed approach, we conducted an experiment to validate several existing coupling and cohesion metrics of object-oriented software. The analysis results show that the metrics defined from the viewpoint of message passings are better than the others. Besides, the results show that the metrics *ICH(C)*, *ICP(C)*, *CBO*, *MPC*, *RFC* and *IH_ICP(C)* are validated for four validity principles. These analyses between all the metrics and the direct reflectors are at a significance level α=0.05. Table 5.1 summarizes the validation results of all the cohesion and coupling metrics.

Table 5.1 Summary of validation for the cohesion and coupling metrics. • : a metric is valid

Validity principles \ Metrics		**Cohesion**	
		ICH(C)	*ICH(H)*
Correlation	*ctch*	•	•
	cmch	•	•
Consistency	*ctch*	•	
	cmch	•	•
Discriminative power	*ctch*	•	
	cmch	•	•
Predictability	*ctch*	•	•
	cmch	•	•

Validity principles \ Metrics		**Coupling**					
		ICP(C)	*CBO*	*DAC*	*MPC*	*RFC*	*ICP(H)*
Correlation	*ctcp*	•	•	•	•	•	
	cmcp	•	•	•	•	•	
Consistency	*ctcp*	•	•	•	•	•	
	cmcp	•	•	•	•	•	
Discriminative power	*ctcp*	•	•	•	•	•	•
	cmcp	•	•		•	•	•
Predictability	*ctcp*	•	•	•	•	•	
	cmcp	•	•	•	•	•	

Validity principles \ Metrics		**Inheritance Coupling**		
		IH_ICP(C)	*CIC*	*DIT*
Correlation	*ctcp*	•		
	cmcp	•		
Consistency	*ctcp*	•	•	
	cmcp	•		•
Discriminative power	*ctcp*	•	•	
	cmcp	•		•
Predictability	*ctcp*	•		
	cmcp	•		

Reference

[1] Basili, V. R., Selby, R. W. and Hutchens, D. H., "Experimentation in Software Engineering," IEEE Trans. on Software Engineering, Vol. SE-12, No. 7, July 1986, pp. 733-743.

[2] Chidamber, S. R. and Kemerer, C. F., " A Metrics Suits for Object- Oriented Design," *IEEE trans. on Software Eng.*, vol. 20, 1994, pp. 476-493.

[3] Chandrashekar, R. and Michael, R. L., "Reliability and Maintainability Related Software Coupling Metrics in C++ Programs," *Proc. of Symp. on Software Reliability Eng.*, 1992, pp.303-311.

[4] Kafura, D. and Canning, J., "A Validation of Software Metrics Using Many Metrics and Two Resources," *8th Int. Conf. on Software Eng.*, 1985, pp.378-385.

[5] Korson, T. and McGregor, J. D., "Understanding Object-Oriented: a Unifying Paradigm," *Communications of ACM*, Vol.33, No.9, 1990, pp.40-60.

[6] Lee, Y. S., Liang, B. S., Wu, S. F. and Wang, F. J., "Measuring the Coupling And Cohesion of an Object-Oriented Program Based on Information Flow," *Proceedings of the International Conference on Software Quality*, November 6-8, 1995, pp.81-90.

[7] Li, W. and Henry, S., "Object-Oriented Metrics that Predict Maintainability," *Journal of System and Software*, 23, 1993, pp.111-122.

[8] Macdonell, S. G., "Rigor in Software Complexity Measurement Experimentation," *J. Systems Software* 1991, pp.141-149.

[9] McClave, J. T. and Dietrich II, F. H., *A First Course in Statistics, Third Edition*, Maxwell Macmillian International, New York, 1989.

[10] Mendenhall, W. and Sincich, T., *Statistics for Engineering And the Sciences, Third Edition*, Maxwell Macmillan International, New York, 1991.

[11] Pfleeger, S. L., "Experimental Design and Analysis in Software Engineering Part 5: Analyzing the Data," *ACM SIGSOFT Software Engineering Notes*, Vol. 20, No. 5, 1995.

[12] Schneidewind, N. F., "Methodology for Validating Software Metrics," *IEEE Trans. on Software Engineering*, Vol. 18, No. 5, May 1992, pp. 410-421.

[13] Sebesta, R. W., *Concepts of Programming Languages*, 2nd, Benjamin/Cummings, ISBN 0-8053-7130-3, 1993.

[14] Wu, S. F. and Wang, F. J., "Measuring Coupling and Cohesion of an Object-oriented program," *Technique Report No. CSIE-96-1003*, Computer Science and Information Eng. Inst. Chiao-Tung University, Hsinchu, Taiwan, R.O.C., 1996.

[15] Yourdon, E. and Constantine, L.L., *Structured Design: Fundamentals of a Discipline of Computer Program and Systems Design*, Prentice Hall, Englewood Cliffs, New Jersey, 1979.