ISSN: 2348 - 2117
International Journal of Engineering Technology and Computer Research (IJETCR)
Available Online at www.ijetcr.org
Volume 3; Issue 3; May-June; Page No. 14-17

LI METRICS WHICH PREDICTS MAINTAINABILITY

Rashmi Gupta
Faculty of Computer Science Department, Amity University Haryana
goyal.rashmi18@gmail.com

Abstract:
Software metrics have been studied in the procedural paradigm as a quantitative means of assessing the software development process as well as the quality of the software products. Several studies have validated that various metrics are useful indicator of maintenance. However, software metrics have rarely been studied in object oriented paradigm. Very few metrics have been proposed to measure object oriented systems, and the proposed ones have not been validated.
This research concentrate on Li object oriented metrics and the analysis of these metrics with maintenance factor. In addition a set of new metrics are proposed that can find the impact on maintainability of a class by using the combination of one Li metric with another metric.

Keywords: Software metrics, Object-oriented metrics, Li Metrics, Maintainability.

1: INTRODUCTION

Software maintenance is one of the most difficult and costly task in the software development process. Among all the factors which have a potential influence on software maintainability, software metrics, especially which measures the inter-connectivity of system component, have been shown to have an impact of software system maintainability in the procedural paradigm [2].

Object oriented paradigm gives the way of effective maintenance of program components. The process of maintenance expedites the software development and thereby resulting in high quality work in minimum time. They are easy to understand, adapt and scale because of modular structure, relatively low coupling and high cohesion [1].

In this research paper, we are combining the one Li metric with one another to analyze the object oriented systems developed using maintenance factor.
Second section describe the prior work in the field of software metrics particularly Li Metrics. Since the third section of the research paper is proposing the new metrics Metric 1, Metric 2, Metric 3. The fourth section is result and analysis. Finally section five presents some conclusions and future work.

2: LITERATURE REVIEW

Understanding the object oriented paradigm is the first step toward the definition of metrics for that paradigm. The study of object oriented paradigm results in object oriented concepts such as: object, class, attributes inheritance, method, and message passing.

A significant number of object oriented metrics have been developed. For example, metrics proposed by Abreu [2], CK metrics [3], Li and Henry [4] metrics, MOOD metrics [5], Lorenz and Kidd [6] metrics etc.

Li discovered some metrics as he discovered problems with Chidamber and Kemerer metrics during the course of defining the unit definition model for the metrics. An alternative metrics suite was proposed by Li [LI98]. Six metrics, Number of Ancestor Classes (NAC), Number of Local Methods (NLM), Class Method Complexity (CMC), Number of Descendent Classes (NDC), Coupling through Abstract Data Type (CTA), and Coupling through Message Passing (CTM) were proposed in order to overcome some limitations found in Chidamber and Kemerer metrics.

2.1. Number of Ancestor Classes (NAC)

- NAC measures the total number of ancestor classes from which a class inherits in the class inheritance hierarchy.
- In this the unit for the NAC metric is class, because the attribute that the NAC metric capture is the number of other classes environments from which the class inherits.
- If a sub class accesses the inherited properties from the super class without using the method defined in the super class, the encapsulation of the super class is violated.
- The larger the NAC metric, the harder it is to maintain the class.

Corresponding author: Rashmi Gupta

2.2. Number of Local Methods (NLM)

- The Number of Local Methods (NLM) is defined as the number of local methods defined in a class which are accessible outside the class
- It measures the attributes of a class that WMC metric intends to capture. This attribute is for the usage of the class in an object-oriented design because it indicates the size of a class's local interface through which other classes can use the class.
- The larger the local interface of a class, the more effort is needed to design, implement, test, and maintain the class.

2.3. Class Method Complexity (CMC)

- The Class Method Complexity (CMC) metric is defined as the summation of the internal structure complexity of all local methods.
- This metric affect the effort required to design, implement, test and maintain the class.

2.4. Number of Descendent Classes (NDC)

- The definition of NDC is the total number of the descendent classes (subclass) of a class.
- NDC argued that the data attributes and methods that are inheritable from the class by its subclass. This might increase the complexity measure cause by the inheritance relations among the classes.
- It seems logical that the more direct children a class has, the more classes it may potentially affect due to inheritance.
- The larger the NDC metric, the harder it is to maintain the class.

2.5. Coupling through Abstract Data Type (CTA)

- The Coupling through Abstract data Type (CTA) is defined as the total number of classes that are used as abstract data types in the data-attribute declaration of a class.
- Two classes are coupled when one class uses the other class as an abstract data type [16].
- This metric gives the scope of how many other classes' a class needs in order to provide its own service to others.
- The number of variables having ADT (Abstract Data Type) may indicate the number of data structures dependent upon the definitions of other classes.
- The more ADTs a class has, the more complex the coupling is of that class with other classes.

2.6. Coupling through Message Passing (CTM)

- Coupling through Message Passing (CTM) is defined as the number of different messages sent out from a class to other classes excluding the messages sent to the objects created as local objects in the local methods of the class.
- Two classes can be coupled because one class sends a message to an object of another class, without involving the two classes through inheritance or abstract data type [LI98].
- The metric gives an indication of how many methods of other classes are needed to fulfill the class' own functionality.
- The number of messages send out from a class may indicate how dependent the implementations of the local methods are upon the methods in other classes.

3. PROPOSED METRICS

In this section, a set of new metrics are proposed to measure the reusability of an object oriented codes.

3.1. Metric1 (NAC + NDC)

- If a sub class accesses the inherited properties from the super class without using the method defined in the super class, the encapsulation of the super class is violated.
- It seems logical that the more direct children a class has, the more classes it may potentially affect due to inheritance.
- Therefore the maintainability of a class increases with the increase in combination of NAC and NDC. So NAC + NDC have positive impact on maintainability of a class

3.2. Metric2 (CTA + CTM)

- The number of variables having ADT (Abstract Data Type) may indicate the number of data structures dependent upon the definitions of other classes.
- The number of messages send out from a class may indicate how dependent the implementations of the local methods are upon the methods in other classes.

1. Therefore the maintainability of a class increases with the increase of coupling between the classes. So CTA + CTM have negative impact on maintainability

Table 1: Value of proposed metrics

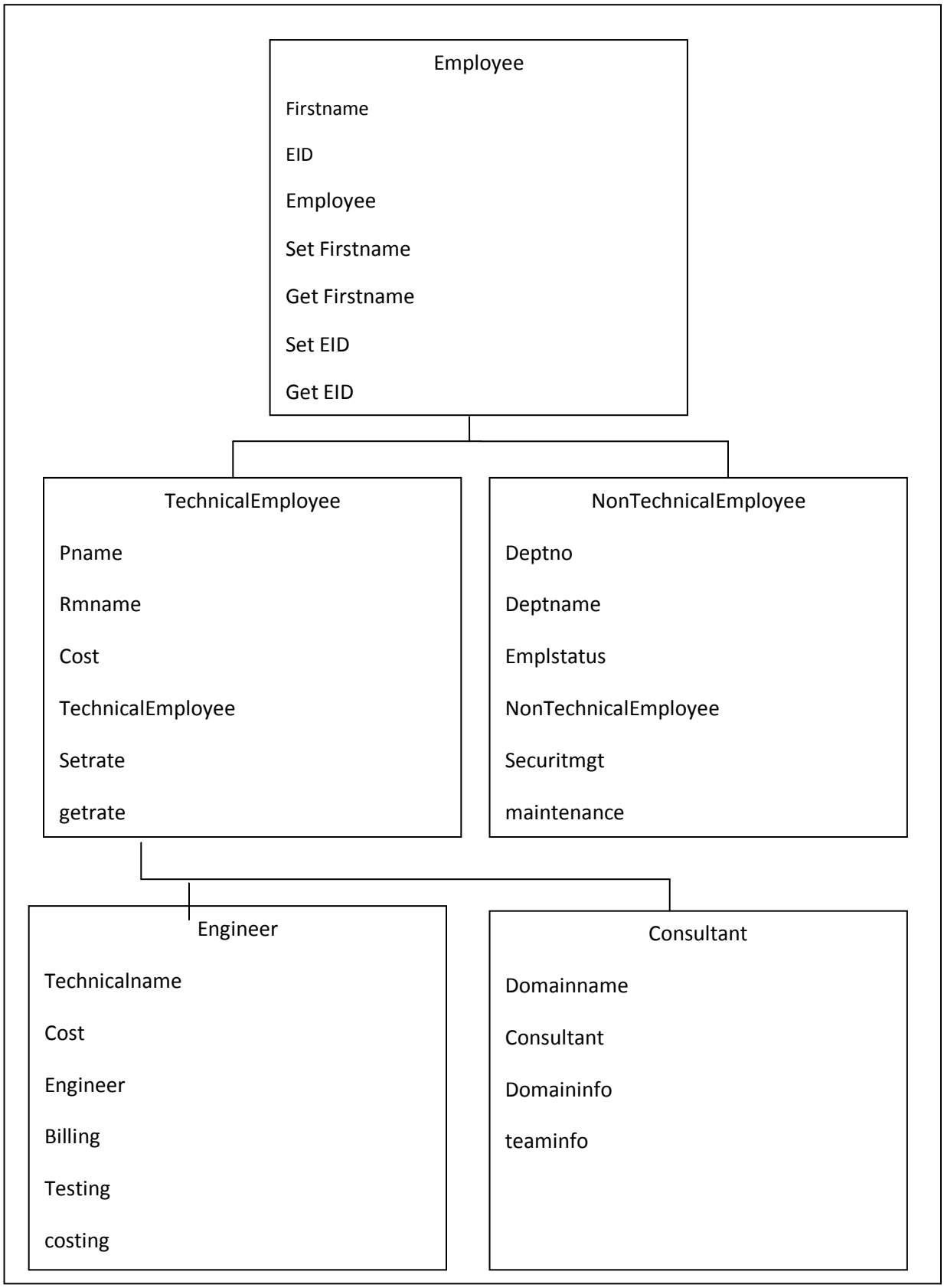

Table 2: Value of metrics

Class			Metric1			Metric2
	NAC	NDC	(NAC + NDC)	CTA	CTM	(CTA + CTM)
Employee	0	2	2	4	2	6
TechnicalEmployee	1	2	3	0	1	1
NonTechnicalEmployee	1	0	1	0	2	2
Engineer	1	0	1	0	3	3
Consultant	1	0	1	0	2	2

5. CONCLUSION AND FUTURE WORK:

The aim of this research was to implement the Li metrics for the object oriented paradigm. Propose and implement additional metrics for the object oriented paradigm. Investigate these metrics and their relationship for maintainability.

The result of the analysis of the Li metrics shows that There is a strong relationship between metrics and maintainability factor. The prediction is successfully cross validated. Another future prospect would be to have the data set as projects with identical requirements done in different object oriented languages.

6. REFERENCES:

1. Li,W.,Henry,S.,Kafura,D.,andSchulman,R."MeasuringObject-OrientedDesign,*"JournalofObjectOriented Programming*,July-August1995.
2. Lorenz,M.,and Kidd,J.*Object–Oriented Software Metrics*, Prentice Hall, Englewood Cliffs,N. J., 1994.
3. Mayer,T.,andHall,T."CriticalAnalysisofCurrentOODesignMetrics", *SoftwareQualityJournal*,8,1999.
4. Pressman, R. "APractitioner's Approach to Software Engineering," Mc-grawhillPublications,2001.
5. Sheldon,F.T.,Jerath,K.,andChung,H."Metricsfor Maintainability of ClassInheritance Hierarchies ",*Journalof SoftwareMaintenance*,issue3,May 2002.
6. Basili,V.,Briand,L.,and Melo,W."AValidationofObjectOrientedDesignMetrics as Quality Indicators",*IEEE Trans. Software Engineering*. vol.22,1996.
7. Chidamber,S.R.,andKemerer,C.F."AMetricSuiteforObjectOrientedDesign,"*IEEE Trans.SoftwareEngineering*,Vol. 20, 1994.
8. Diev S., "Software estimation in the maintenance context," ACM Software Engineering Notes, Vol. 31, No. 2, 2006