International Journal on Recent and Innovation Trends in Computing and Communication ISSN: 2321-8169
Volume: 4 Issue: 7 209 - 210

Generalization Index: Defining a metric for the detection of smells in UML Class Diagrams in Eclipse Modeling Framework in Eclipse

Er.Parul, M.tech Research Scholar
Department of Computer Engineering
Punjabi University
Patiala, India
e-mail: parulgulati91@gmail.com

Er. Brahmaleen K. Sidhu
Department of Computer Engineering
Punjabi University
Patiala, India
e-mail: brahmaleen_sidhu@yahoo,co.in

***Abstract*—** In the field of Software Engineering, while designing the software design and maintaining the source code quality, a lot of good and bad practices come into being. With the continuous evolutions in the field of modeling in software development processes, Model Driven Software Development (MDSD), focuses towards the quality of software models. Unified Modeling Language (UML) is a graphical notation for expressing object-oriented designs. With its emergence as a modeling standard and being widely accepted by most software development organizations, in this research paper we focus on UML Class Diagrams. Metrics are mathematical models used for measuring. In software engineering, metrics are utilized for measuring quality aspects of software models. A manual model review is very time consuming and prone to errors, so it becomes essential to automate the tasks as effectively as possible. The Eclipse plug-in EMF Metrics supports specification and calculation of metrics wrt. specific EMF based models. A new definition technique for EMF quality assurance can be defined using Java, an OCL query or Henshin Pattern. In this paper we propose an algorithm for the calculation a new metric named Generalization Index Metric (GIX) for Java Code.

Keywords- *Software quality, MDSD, UML Class Diagrams, Metrics, EMF, Generalization Index Metric (GIX).*

*****

I. INTRODUCTION TO METRICS

In the field of Software Engineering, while designing the software design and maintaining the source code quality, a lot of good and bad practices come into being. With the continuous evolutions in the field of modeling in software development processes, Model Driven Software Development (MDSD), focuses towards the quality of software models. In Model Driven Software Development, models have become of chief importance where quality assurance of the overall software product relies considerably on the quality assurance of involved software models. For this reason, software developers need objective and valid measures for use in the evaluation and improvement of product quality from the initial stages of development.

Quality assurance techniques for models originate from corresponding techniques for software code by lifting them to models. Especially class models are closely related to programmed class structures in object-oriented programming languages Class diagrams are a key artifact in the development of object-oriented (OO) software because they lay the foundation for all later design and implementation work. Quality in software products is characterized by the presence of different external attributes such as functionality, reliability, usability, efficiency, maintainability and portability.

Unified Modeling Language (UML) is a graphical notation for expressing object-oriented designs. With its emergence as a modeling standard and being widely accepted by most software development organizations, in this research paper we focus on UML Class Diagrams. In the initial stages of development UML Class Diagram is to be composed of following UML Constructs :

- Classes
- Packages
- Each class has its attributes and operations.
- Attributes have their name.
- Various relationships exist among classes: Association, Aggregation, Generalization, Dependencies.

II. EMF METRICS IN ECLIPSE

Wherever A manual model review is very time consuming and prone to errors, so it becomes essential to automate the tasks as effectively as possible. We implemented tools supporting the included techniques metrics, smells, and refactoringsfor models based on the Eclipse Modeling Framework (EMF) , a common open source technology in model-basedsoftware development. The Eclipse plug-in EMF Metrics supports specification andcalculation of metrics wrt. specific EMF based models. While performing quality assurance tasks in EMF in Eclipse we get two options: either to use existing techniques or define new techniques, shown in the figure below :

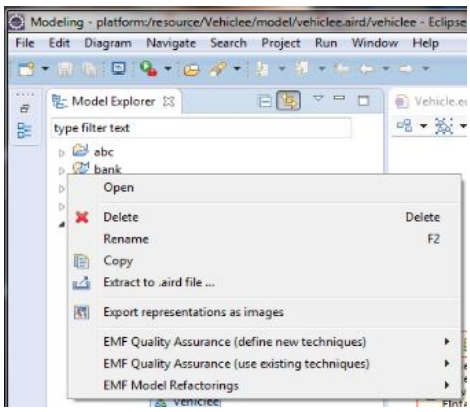

Figure 1: EMF Quality Assurance

On selecting EMF Quality Assurance (use existing techniques), we get a list of pre-defined metrics database. The metrics preferences allow the display order of the metrics to be changed and the metrics database to be cleared (to force recalculation of all metrics). The metrics can now trigger warnings that show up in the task view as well as the editors, indicating methods and types for which metrics safe ranges are being violated. The minimum and maximum for each metric can be set in the preferences. A screenshot of the list of metricsis shown in the next figure:

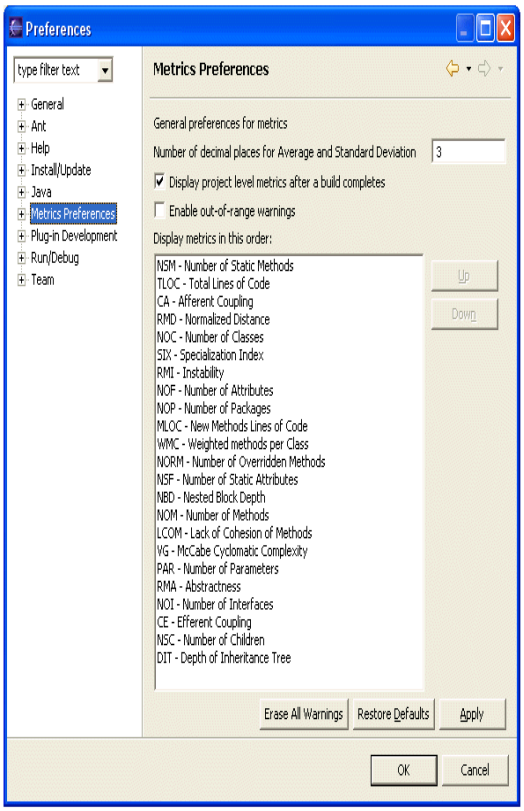

Figure 2: List of Metrics

III. GENERALIZATION INDEX METRIC

A new definition technique for EMF quality assurance can be defined using Java, an OCL query or Henshin Pattern. In this paper we propose an algorithm for the calculation a new metric named Generalization Index Metric (GIX) for Java Code.

GENERALIZATION INDEX METRIC (GIX)

1. Create dictionary for mapping attribute names from system model. (eg., attribute 1 = a_1)
2. Calculate total number of attributes for superclass and subclasses for actual system model. (eg.,*TotalCount* = count + $count_1$ + …$count_n$, count being number of attributes in superclass. $count_1$….$count_n$ for n number of subclasses).
3. *For* each overridden attribute
   Iterate through superclass and subclasses to find the overridden attribute count.
   If attribute superclass is equal to attribute subclass
   Then put the count in the list common count
   If attribute superclass is not equal to attribute subclass
   Then remove it from the list common count
   Add all the count in the list *CommonCount.*
4. Use formula to calculate metric Generalization Index(GIX)

   GIX =
5. *If* GIX < 0
   Then metric does not specify a smell in the system model
   If GIX >= 0
   Then metric specify a smell in the system model.

.

IV. CONCLUSIONS AND FUTURE SCOPE

This metric has been proposed keeping in consideration the association - 'generalization' existing in a class diagram. UML diagram's generalization association is also known as inheritance. Subclass is a particular case of the superclass and inherits all attributes and operations of a superclass, but can have its own additional attributes and operations. In some cases overlapping takes place where some instances of subclasses can belong to two or more subclasses at the same time. The default value has been set to zero due to several empirical studies considered by various authors on metrics. The value of the metric specifies the presence of a smell.
The presence of a smell makes it considerable for detection, which can be done either by combining other metrics with the proposed metric or looking for a new smell which has still not been investigated at all.
Further work is also necessary towards measuring OO models which cover dynamic aspects of OO software, such as, sequence diagrams, statechart diagrams, etc.

ACKNOWLEDGMENT

With the writing of this thesis, I would like to extend my thanks to all of the people who have helped me to be able to write it.

My first thanks go to my supervisor Er. Brahmaleen Kaur Sidhu, Assistant Professor, Department of Computer Engineering, Punjabi University, Patiala who helped and guided me during the duration of this work. Apart from her guidance in this duration, her methods of the distribution of the knowledge of the subject have also benefitted me a lot. I am very grateful to her for providing me with important resources for the research and analysis.

I am very much grateful to my parents for their cooperation in this project.

REFERENCES

[1] T. Arendt, M. Burhenne and G. Taentzer, "Defining and Checking Model Smells: A Quality Assurance Task for Models based on the eclipse modeling framework," Philipps-Universit¨at Marburg, FB12 - Mathematics and Computer Science, Hans-Meerwein-Strasse, D-35032 Marburg, Germany, December 3,2010.

[2] C. Bouhours, H. Leblanc and C. Percebois, "Bad smells in design and design patterns," Journal of object technology, vol. 8, no. 3, 2009.

[3] K. Ch. and P. L., "Design Recovery by Automated Search for Structural Design Patterns in Object-Oriented Software," in 3rd Working Conference on Reverse Engineering, 1996.

[4] M. Fowler, "Refactoring : improving the design of existing code," Addison-Wesley, 1999.

[5] M. Genero, M. Piattini and C. Calero, "A survey of metrics for UML Class Diagrams," Journal of Object Technology, vol. 4, no. 9, 2005.

[6] https://en.wikipedia.org/wiki/UML_tool.

[7] I. Polasek, "Anti-Pattern Detection as a Knowledge Utilisation," FIIT STU, Bratislava, Slovakia.

[8] R. Wieman, "Anti-Pattern Scanner: An Approach to detect Anti-Patterns and Design Violations," 2011.