ISSN (Online) 2278-1021
ISSN (Print) 2319-5940

IJARCCE

International Journal of Advanced Research in Computer and Communication Engineering

SITES

Smart And Innovative Technologies In Engineering And Sciences

Gyan Ganga College of Technology

Vol. 5, Special Issue 3, November 2016

A Review of Software Architecture Metrics Extracted using UML

Zafar Shareef[1], Rishabh Ojha[2]

B.E. Vth Semester student, Computer Science, Gyan Ganga College of Technology, Jabalpur, India[1, 2]

Abstract: Unified Modeling Language was introduced in late 90's, since than software industries and practitioners have been adopting this language for detecting design deficiencies at the early stage in the design phase. UML has now become the de facto design document being used in the early design phase. It has gained wide popularity being able to systematically represent artifacts for software architecture. For Component Based Software Development (CBSD), metrics have been proposed by authors, for understanding complexity so that deficiencies in the design phase can be eliminated, which can cause problems in the later phases of the SDLC. A variety of tools has been proposed for extracting metrics, CAME tool has been proposed for extracting metrics for Software Component assembly from UML design documents. Components being black box in nature; interacts only through their interfaces. The complexity of these interactions can be studied through metrics. The XMI (XML Meta Data Interchange) standard is now part of the UML tools. Using XMI file, the component metrics can be extracted. This paper presents a systematic literature review of metrics extraction for CBSD using UML tool. This paper addresses the software architecture metrics which includes object oriented metrics as well as component based metrics extracted using design document UML.

Keywords: Component Based Software Development, Component diagram, Component Assembly Metrics Extractor, Software metrics, Software Architecture, XMI.

I. INTRODUCTION

Software architecture includes object oriented concepts as well as component based software engineering. Software architecture specifications and models proposed [1] provides a better, reliable and a blue print to build complex software systems for later software engineering activities. Component Based Systems (CBS) means assembly of components, where individual component is assembled resulting in the form of assembly, for interoperating amongst them. Research in component testing suggests that mostly errors or faults have been located in few software components [2]. In a component assembly, if these components are identified, then the possibility of their failure can be avoided, resulting in a successful software system [3]. A software system comprises a number of different components, built by different companies; these components exhibit different qualities, where a set of metrics for software system proposed by [4] will help not only in gathering information but also help in assessing complexity in order to locate software components which might create trouble at later stages [4]. A number of metrics have been proposed by different authors from time to time, but these metrics are inadequate for component based systems, as the components being black box in nature, interact only through interfaces. A number of reasons have been given by Gill and Grover [5] why traditional software metrics are not suitable for component based systems. Metrics have been proposed for understanding complexity of interfaces, constraints and interaction Mahmood and Lai [3]. This information in the form of static and dynamic metrics was proposed theoretically by [4], but [4] lack empirical validation. The static metrics proposed by [3] were extracted at the early stage of software development in the design phase with the help of UML tool [6].Software metrics proposed by [7] were extracted using a tool UXSOM (UML Generated XML to Software Metrics). This paper is concentrated towards the software architecture metrics which includes object oriented metrics as well as component based metrics extracted using design document UML. Section 2 describes the related work. Section 3 describes significance of software metrics extracted using UXSOM tool [7].Section 4 describes the significance of metrics extracted using CAME tool (Component Assembly Metrics Extraction using UML). Section 5 provides conclusions derived out of this review.

II. RELATED WORK

The application of software architecture metrics will be useful only if we are able to automate the metrics extraction procedure implemented through UML tool. Numerous efforts in this direction have been made by the research community working. Several commercial as well as open-source metric tools exist today. Open source metrics tools are: UXSOM [7], OOMeter [8], UMP [9], Fujaba [10], JMetric [11], are among them. The

DOI 10.17148/IJARCCE

automation of software metrics extraction using XMI file is efficiently handled by these tools. A number of commercially available software metric tools include SDMetrics [12], Borland Together Control Center (TCC) [13] and Jhawks [14]. The tool computes a number of metrics on XMI files.

III. UXSOM: AS A TOOL SUPPORT

A number of researchers had been working for finding new ways to automate the code exported using UML tools. UXSOM is a tool based on Java platform which analyzes class diagrams represented in XMI format. Software architecture metrics that are extracted using UXSOM tool helps in assessing the details of UML class diagrams, deriving attributes, operations, classifiers and packages and their relationships. UXSOM [7] works with variety of UML tools and calculate software metrics. UXSOM [7] can extract metrics from UML tools like ArgoUML, UMLet, ESS-Model, MagicDraw, Sparx Systems Enterprise Architect. This tool shows the significance and necessity of the extraction of a standard software metrics calculation system. The tool parses five tools only and works with XMI files containing class diagram. The tool provides an understanding of the UML tools selected for extracting nine software metrics from class diagrams and packages.

A. Implementation of **UXSOM** tool

The tool is based on Java platform, and supports cross-platform, which analyzes artefact like UML class diagram modelled in ArgoUML and represented in XMI based formats. The XMI file is generated with the help of Export XMI option, which is then parsed for extracting information related to various metrics of class diagram.

B. Advantages of **UXSOM** tool

The tool derives attributes, operations, classifiers, and packages and their relationships quantitatively. The tool is mainly focused on class diagram thus assessing the details. These details obtained from the tool will be of help to software engineers who program their project from UML class diagrams modelled at the early stage of software life cycle.

IV. CAME: AS A TOOL SUPPORT

A Component assembly includes different components developed by different companies for different platforms, each having different qualities. Integration of many components into a larger application raises the issue of software quality. Another problem is re-assembling company's legacy code to a new component based standard. These components on integration in an assembly; changes the complexity of component assembly as well as change in complexity of an individual component. These problems create several types of risks, which have to be identified by a developer. To tackle those risks, the developer should attain more information from their software artefacts, where a set of metrics will be useful in gathering such information [4].

The component based software metrics proposed by [4] are extracted using CAME tool. The tool is based on Java platform, implemented for component based metrics in a parser based tool. The metrics are calculated from UML design documents. A model of UCRS (University Case Registration System) [15]; is represented in ArgoUML, for which an XMI file is generated. A Model of Component assembly is designed, where different components and their interfaces are represented through artefacts. Parsing of this XMI file is done to extract static metrics proposed by [4]. The tool extract metrics limited to component diagrams and interfaces only. It works only with XMI files.

A component when deployed and executed may yield on its own the expected results, but its behaviour and functionality when integrated with other components to make a complete application may be different to the expected [16], the static metrics extracted through UXSOM helps in assessing the functionality of each component when integrated with other components and functionality of the application on the whole. The results of the static metrics obtained through UXSOM are useful indicators for gathering required information.

A. Implementation of **CAME** tool

The tool is developed in Java using Netbeans 6.8. It is SAX parser based tool. Using ArgoUML a model is designed creating component artefacts through Deployment diagram option; the XMI 1.2 file is generated with the help of Export XMI option, which is then parsed for extracting information related to various metrics in a component assembly for component-based systems. The parser parses the XMI file which contains information about all the components integrated into the system.

B. Advantages of **CAME** tool

The tool displays information like number of components, their interfaces and the operations available in an interface of a component assembly. Components names are displayed and after selecting a particular component, it's provided and required interface can be known. The proposed static metrics by [4] are: CPD (Component Packing Density), CID (Component Interaction Density), CIID (Component Incoming Interaction Density), COID (Component Outgoing Interaction Density), CAID (Component Average Interaction Density). $CRIT_{link}$ (Link Criticality Metric), $CRIT_{bridge}$ (Bridge Criticality Metric), $CRIT_{inheritance}$ (Inheritance Criticality Metric), $CRIT_{size}$ (Size Criticality Metric). The metrics measures complexity, Interactions, Incoming and Outgoing and Average interactions. Each metric have its own significance in relation to component assembly which indicates the developer has to spend more effort on analysing the module and locating the risks.

V. CONCLUSION

The present work reviews Software architecture metrics based on the concepts of Object oriented and Component based technology. The metrics extracted from XMI file will surely benefit software engineers, software project managers and system analysts who program their project from UML artefacts. This will help them to identify risks involved and uncover problem areas.

ACKNOWLEDGMENT

We thank the reviewers for their excellent constructive comments, which led to considerable improvement in the quality of the paper.

REFERENCES

[1] International-Standard-Organization, "ISO/IEC Standard for Systems and Software Engineering – Recommended Practice for Architectural Description of Software-Intensive Systems," ISO/IEC 42010 IEEE Std 1471-2000 First edition 2007-07-15, pp. c1-24, 6 2007.

[2] N.E. Fenton and N. Ohlsson, Quantitative analysis of faults and failures in a complex software system. IEEE Transactions on Software Engineering 2000; 26(8): 797-814.

[3] S. Mahmood and R. Lai, A complexity measure for UML component-based system specification. Software Practice and Experience, 2008, 38, 117-134.

[4] V.L. Narasimhan and B. Hendradjaya, Some theoretical considerations for a suite of metrics for the integration of software components. Information Sciences, 2007, 844-864.

[5] N.S. Gill and P.S. Grover, Component-based measurement: few useful guidelines, ACM SIGSOFT Software Engineering Notes, 23(6) (2003), 1-4.

[6] J.W. Shareef and R.K. Pandey, CAME: Component Assembly Metrics Extraction using UML, ACM SIGSOFT Software Enginereing Notes, July, 38(4), (2013), 1-12.

[7] M.K. Nuthakki, M. Mete, C. Varol and S.C. Suh, UXSOM: UML Generated XML to Software Metrics, ACM SIGSOFT Software Enginereing Notes, May, 36(3), (2011), 1-6.

[8] J.S. Alghamdi, R.A. Rufai and S.M. Khan, OOMeter: A Software Quality Assurance Tool. Proceedings of the Ninth European Conference on Software Maintenance and Reengineering (CSMR'05), 1534-5351, pp: 1-2.

[9] H. Kim and C. Boldyreff, Developing Software Metrics Applicable to UML Models. In 6th ECOOP Workshop on Quantitative Approaches in Object-oriented engineering, Malaga, Spain, 2002.

[10] Universitat Paderborn, last access: 10-01-13, www.fujaba.de/

[11] Sourceforge.net, last access: 20-12-2012. http://jmetric.sourceforge.net.

[12] SDMetrics Website. www.sdmetrics.com, last access: 20-01-2012.

[13] R.C. Gronback, "Using Audits, Metrics and Refactoring in Borland Together Control Center^{TM}, Borland Together White Papers, www.borland.com/resources/en//together_software remo- -deling. pdf, last access: 16-11-2011.

[14] Virtual Machinery, "Object-Oriented Software Metrics – A short guide". Virtual Machinery Website. http://www.virtualmachinery.com/jhawkmetrics.htm, last access: 12-01-2012.

[15] Y. Liu, H.C. Cunningham, Mapping component specifications to Enterprise JavaBeans implementations. Proceedings of the 42nd Annual Southeast Regional Conference, 2004. ACM Press: New York, pp: 177-182.

[16] P. Parthasarthy, Component Integration Metrics and their Evaluation. Master Theses & Specialist Projects. Paper 414, http://digitalcomons.wku.edu/theses/414, last access:04-11-2011.

BIOGRAPHIES

Zafar Shareef is pursuing Bachelor of Engineering in Computer Science from Gyan Ganga College of Technology, Jabalpur. His research interests include modeling, component based systems and metrics that can be obtained at the early stage of software life cycle.

Rishabh Ojha is pursuing Bachelor of Engineering in Computer Science from Gyan Ganga College of Technology, Jabalpur. His research interests include Big-data, Component based systems.