A BASIS OF ONTOLOGY FOR ASPECT ORIENTED METRICS

Dr. Ananthi Sheshasaayee [1], Roby Jose [2]

[1]*Associate Professor And Head , Department Of Computer Science, QMGCW (India)*

[2] *Research Scholar, Department Of Computer Science, QMGCW (India)*

ABSTRACT

The feature modularization has paved the way for the growth of aspect oriented software development. Aspect Oriented paradigm supports the separation of concerns that are scattered over the system which helps in achieving modularity. Aspect Oriented Software Development encompass software engineering abstractions and complexity at new and different dimensions. As aspect orientation is a comparatively new paradigm, it lacks philosophical basis, as compared to object orientation. Since aspect orientation lacks a theoretical basis as enjoyed by object orientation, there is no clarity over what constitutes an aspect and the best approach to measure and assess the properties of an aspect oriented program. This paper looks at information necessitated to build ontology for aspect oriented quality measurement.

Keywords: Aspect Oriented Software Development, Concerns, Ontology, Separation Of Concerns, Software Measurement.

I. INTRODUCTION

These days' business and internet applications have to tackle features like logging, security et al that cut across multiple modules of a system. These features which are termed as concerns are identified in softwares developed of traditional approaches and isolated resulting in aspect oriented systems. Aspect oriented approach is not a replacement for object oriented approach, but complements object oriented approach. The work instigated by Chidamber and Kemerer [11] based on Wand and Weber [13] interpretation of the ontology for information systems had given elementary theoretical basis for object oriented measurement. The generalized concepts of Bunge [15] form the basis for the concept of objects. Even though Bunge's ontology did not provide an exact ontological definition for concepts in object oriented paradigm, Wand and Weber took up the concepts and changed to suit for object oriented domain. Many of the OO researchers applied the general concepts of Bunge to the object oriented domain. Bunge's ontology had significant followers in object oriented research as it had dealt with meaning and description of representations from real world. Consistent with this ontology, objects are defined independent of implementation considerations and include the concepts of encapsulation and inheritance. Wand and Weber's ontology for object orientation can be summarized as follows. The world is crafted of things: individuals and concepts. Individual possess properties and an onlooker can assign features to an individual but these are attributes and not properties. An individual and its properties constitute an object; therefore an object is a representation of the application domain that takes account of the methods and instance variables that a

designer assigns to that object. The Chidamber and Kemerer [11] suite of object oriented metrics was formed on the basis of Wand and Weber's interpretation of Bunge's ontology. Chidamber and Kemerer used ontological model for measuring code and analyzing properties than a graph theory or structural attribute approach. Aspect Oriented Software Development (AOSD) Europe EU network of excellence has produced an ontology for aspect oriented paradigm. This paper aims at shaping guidance for building ontology for aspect oriented maintenance metrics using the Public ontology for aspect orientation [2]. The rest of the paper is organized as follows. Section 2 presents some basic concepts of aspect orientation and the applicability of Dooyeweerd's theory of aspects to aspect oriented software development. Section 3 discusses a few of the related works. Section 4 provides an overview of public ontology for aspect orientation. The paper is concluded in Section 5.

II. ASPECT ORIENTATION

This section is divided into two subsections. The first subsection provides an overview of basic concepts of aspect oriented paradigm from the view of software architecture. The second subsection shows how Dooyeweerd's theory can be related to aspect oriented software development.

2.1 Basic Terminology

AOSD is a promising paradigm that allows separation of concerns. A concern is a part of the problem that is to be treated as only conceptual unit [9]. These concerns are termed crosscutting concerns as they cut across modularity of other concerns.

Aspect Orientation has been proposed as a method for improving separation of concerns [9, 10] in the structure of OO software. AOSD uses aspects as new abstraction and make available mechanisms for creating aspects and components at joinpoints.

Based on [3, 7] the concepts in Aspect Oriented programming are summarized below:

Separation of Concern: Separation of concerns simplifies system development by allowing the development of specialized expertise by producing, by and large more comprehensible arrangement of elements

Composition: Composition is bringing together separately created software elements. In other words composition integrates several modular artifacts into a logical whole.

Weaving: Weaving is the process of composing base modules of the system with aspects, thereby yielding a working system.

Crosscutting Concern: A crosscutting concern is a concern for which the implementation is scattered throughout the rest of an implementation.

Scattering: Scattering is the occurrence of elements that belong to one concern in modules encapsulating other concerns.

Tangling: The intermixing of the code for the implementation of concerns give rise to tangling.

Aspect: A modular unit designed to implement a concern is referred as aspect.

Join Point: A join point is a point of interest in some object of the software lifecycle through which two or more concerns may be composed.

Join Point Model: A Join Point Model provides the common frame of reference to enable the definition of the structure of aspects.

Pointcut: A Pointcut designator describes a set of join points

Advice: Advice is an aspect element, which augments or constrains other concerns at join points matched by a pointcut expression.

The above given terms and their definitions are as per the book by Filman et al[3] and Kiczales[7] works. The Kiczales[7] definition are based on AspectJ language, the aspect oriented extension for Java.

2.2 Dooyeweerd's Theory And Aspect Orientation

Herman Dooyeweerd was a Christian philosopher whose work [8] Dooyeweerd theory of aspects, tells that the behavior of parts is determined by the behavior of whole. These parts or aspects enable human beings and every other thing to do their jobs and each aspect has its very own interrelated laws. Dooyeweerd's aspectual suit for human beings introduces fifteen aspects. No aspect of the suit can be neglected in real life because life itself is multi-aspectual in nature like a system having multiple concerns. The day today life of human beings show that all aspects function because of the harmonious integration of the aspects. An aspect oriented system takes the disharmony that is created when aspectual properties are replicated as objects and uses the concept of aspects to harmonize them. In analogous to Dooyeweerd theory where areas of interest where splitted to fifteen aspects, in aspect oriented programming this can be seen as system requirements splitted into: logging, persistence, caching and security et al. where each one is represented as aspect. In an aspect oriented program, each of these aspects coexist but one cannot be derived from other.

III. RELATED WORK

The primary ontology for AOP is by van den Berg et al [2]. This consists of fifteen ontology views, of which two are of particular interest in identifying ontology-enabled development from an AOSD perspective. These are the Software System View and the Weaving View, which between them encompass the high level view of the system and the core engineering perspective of managing aspects. In addition, Black and Harman [1] propose some extensions to this model, in particular the integration of spatial, lingual and social aspects. These three aspects can be defined as follows. Spatial- If only the system is viewed from spatial aspect, the user will be conscious that the system may be located in geographically dissimilar locations. Lingual- A system may be viewed as language independent if the concern that it is written in a particular language can be separated out.

Social-This is related to the computer programmers. Separating out the social aspects of the programmers helps to understand the cognitive thinking of the programmers. These views and extensions have been integrated together to provide a single ontology of aspect oriented software development.

IV. ONTOLOGY FOR AO METRICS

Since AOSD is very young and almost all terminology are adapted from AspectJ point of view, it is high time that aspect orientation getting a foundation that is not constrained by a programming language. An ontology enables the sharing of common terminology and associated conceptual models to communicate effectively within stakeholders. An ontology is essential to integrate the results from various activities that may occur in parallel. An ontology is expected to define the meaning of set of concepts for a particular domain. The development of ontology is the process of developing textual definitions of concepts with clear-cut formal specifications of their interdependencies. As the software development life cycle is divided into different phases

the construction of ontology also involves the same phases. The activities in the each phase of ontological development is discussed below

4.1 Requirements

When building ontology for the software measurement in Aspect Oriented Paradigm, it should be able to satisfy certain requirements. The first being the ability to signify the terminology and concepts of AOSD [6]. In other words it should be able to give definition of the facts. The developed ontology must be generally accepted by AO metrics research community. This means that knowledge derived are applicable everywhere and the value and usefulness of the ontology must be acknowledged. The third requirement while building the ontology is that it should be accurate, complete, conflict free and non-redundant [4]. The third requirement specifies the properties the model should possess when it represents the semantic of AO metrics. The fourth requirement [14] while building the ontology for AO metrics requires that it should be verifiable (ability to check for correctness), traceable (origin of the definition could be determined) and unambiguous (single interpretation of the definition). The final requirement is that the model should be maintainable and usable [12]. Maintainability is the capability of the model to accept modification whereas Usability is the extend to which the product is understood, used and liked by a user under specified conditions

4.2 Design

The design or ontology engineering can be done using Common Warehouse Metamodel (CWM) [5]. The metamodel is described in the Unified Modelling Language (UML).In the CWM Business Nomenclature two levels are distinguished viz a *taxonomy* with *concepts* at semantic level (conceptual model or domain model) and a *glossary* with *terms* at representation level (terminology). The design should be able to capture correct properties, avoid ambiguous language and be focusing only on relevant properties.

4.3 Implementation

In this phase the model can be implemented using modelling languages and tools. Various tools are available for supporting modification requests and modification tracking.

So the disciplined process of ontology engineering falls on the line of production of software development with requirements, design and Implementation.

V. CONCLUSION

Ontology is about specification of a concept and provides a common vocabulary to communicate concerning a subject area. The paper describes Dooyeweerd theory of aspects and its relativity to aspect oriented programming. The paper also provides guidance to a theoretical basis upon which an ontology that gives a better and simple understanding of the aspect oriented programs can be constructed. Also the ontology can be used to give a strong philosophical basis for aspect oriented metrics as it is the case with object oriented metrics contributed by Chidamber and Kemerer.

REFERENCES

[1] S. Black, and M. Harman, “Aspect Oriented Software Development: Towards A Philosophical Basis,” Technical Report TR-06-01, Department of Computer Science, King’s College London, 2006

[2] K. van den Berg, J. M. Conejero, and R. Chitchyan, "AOSD Ontology 1.0 - Public Ontology of Aspect-Orientation," AOSD Europe, 2005.

[3] Filman, R., et al., Aspect-Oriented Software Development. 2004: Addison-Wesley.

[4] Shanks, G., Tansley, E. & Weber, R. (2003). Using Ontology to Validate Conceptual Models. Communication of the ACM, October 2003, Vol 46, No 10, p. 85 - 98.

[5] CWM (2003). Common Warehouse Metamodel (CWM) Specification, March 2003,Version 1.1, Volume 1, formal/03-03-02

[6] Skulmoski, G. (2002). Shifting Gears: the De Facto Global Standard for Project Management.

[7] Kiczales, Gregor, et al. "An overview of AspectJ." ECOOP 2001—Object-Oriented Programming. Springer Berlin Heidelberg, 2001. 327-354.

[8] Basden, A. "The Dooyeweerd pages." (2000).

[9] Tarr, Peri, et al. "N degrees of separation: multi-dimensional separation of concerns." Proceedings of the 21st international conference on Software engineering. ACM, 1999

[10] Kiczales, Gregor, et al. Aspect-oriented programming. 11th European Conference on Object Oriented Programming in:LNCS,vol.1241,Springer Verlag,1997,pp 220-242 1997

[11] S.R.Chidamber and C.F. Kemerer. A metrics suite for object oriented design. IEEE transactions on Software Engineering,20(6):476-493,June 1994

[12] ISO/IEC9126 (1991). International Standard ISO/IEC 9126. Information technology Software product evaluation,Quality characteristics and guidelines for their use,International Organization for Standardization, International ElectrotechnicalCommission, Geneva

[13] Wand, Yair, and Ron Weber. "An ontological model of an information system."Software Engineering, IEEE Transactions on 16.11 (1990): 1282-1292.

[14] ANSI/IEEE 830 (1984). Guide to Software Requirements Specification

[15] Bunge, Mario. Treatise on basic philosophy: Ontology I: the furniture of the world. Vol. 1. Springer, 1977.

Biographical Notes

Dr.(Mrs.). Ananthi Sheshasaayee is working as Associate Professor in PG and Research Department of Computer Science, Quaid E Millath Govt College for Women, Chennai, India.

Mrs. Roby Jose is currently pursuing Ph.D from PG and Research Department of Computer science from Quaid E Millath Govt College for Women, Chennai, India.