International Journal of Advanced Research in Engineering Technology & Science
ISSN: 2394-2819
Email: editor@ijarets.or Volume-4, Issue-12 December- 2017 www.ijarets.org

OBJECT ORIENTED MODELS AND ONTOLOGIES IN OBJECT ORIENTED DESIGN

NITIN PANWAR[1], AMIT RANJAN[2]
[1,2]Assistant Professor,
IIMT College of Engineering
Greater Noida

ABSTRACT This paper is to describe the knowledge based ontologies with object oriented models in object oriented software engineering. Ontology itself known as the knowledge base foundation concept and closely related to the object oriented software Design because ontologies are closely related to modern object-oriented software design. There are more effective and efficient quality software for providing the maintainability and these maintainability software systems based on the status of some code, which shows high co-relation between automated model and evaluation.

This paper provides the review on the previous papers and described some maintainability models for better standards in maintainability. ontologies means to provide a knowledge base domain for resource data framework used in object oriented software system and also describe the use of ontologies, object modeling and ontology modeling.
KEYWORDS: object oriented, maintainability, models, ontology.

INTRODUCTION

There are various object-oriented paradigms in the frame work for the software. Maintainability is the way through which we can modification of an object-oriented software product to overcome their performance and attributes to adapt the product in a modified environment and maintainability is the strict engineering mostly used to bug fixing. The object oriented paradigm is the framework in software engineering, influencing all effort in information science. It is one of the main objectives of the software engineering discipline [18]. Object models are different from other modeling techniques because they have merged the concept of variables and abstract data types into an abstract variable type: an object. Objects have identity, state, and behavior and object models are built out of systems of these objects. To make object modeling easier, there are concepts of type, inheritance, association, and possibly class [1]. Ontology is well known as description of declaration and abstract way the domain information of the application, it involved with vocabulary and how to constrain the use of the data [6] And this maturing standards and tools can be used for ontology modeling [1].

DEFINITION

Object model is the mechanism of object-oriented paradigm, which is used for software engineering. In particular, the general software engineering principle of parting of concerns combined with object-oriented modeling characteristics has turned out to be very useful. The basic idea of object-orientation is the consequent application of the abstract data type concept, combining data and functionality. The abstract data type concept is applied in the context of the architecture of any object-oriented model. artificial intelligence (AI) arena has been focused on knowledge modeling. The term ontology is used to refer to “an explicit specification of a conceptualization [of a domain] is mentioned by Tom Gruber which we are already familiar with for quite sometimes. In other words, ontology refers to a formalization of the knowledge in the domain. Ontology is the concept which is separately identified by domain users, and used in a self contained way to communicate information. Combination of concept is the knowledge base or knowledge network. Some of the reasons why someone want to develop an ontology are to share common understanding of the structure of information among people or software agents, to enable reuse of domain knowledge, to make

domain assumptions explicit, to separate domain knowledge from the operational knowledge, to analyze domain knowledge [19]

MODELING

A. Object Modeling: The object-oriented model is based on a collection of objects. An object contains values stored in instance variables within the object. Thus objects contain objects to an arbitrarily deep level of nesting. Attributes/properties: objects will have at least one attribute. Possible slot types are primitive types (integer, boolean, string etc.), references to other objects (modeling relationships) and sets of values of these types. An object also contains bodies of code that operate on the object. These bodies of code are called methods. Method/Operations: They are attached to classes or slots and contain meta information, such as comments, constraints and default values. Relationship/relations: they represent the relation between objects/classes from object model (KB). Major classes of relations exist: relations combining labels (the name we tend to give to things) and concepts (the things themselves) and concepts and relations combining concepts (the part-whole relation)[13]. Objects that contain the same types of values and the same methods are grouped into classes. A class may be viewed as a type definition for objects. Analogy: the programming language concept of an abstract data type. The only way in which one object can access the data of another object is by invoking the method of that other object. This is called sending a message to the object. Internal parts of the object, the instance variables and method code, are not visible externally or some researchers called it as black box. The following Fig.1 shows a simple Banking System object Model, containing classes for Head-Office, Branch, Accounts held at that Branch, and the Customers who the Accounts belong to. Object/Class represent the tangible things. For example, an object representing a bank account. The object contains instance variables number and balance. The object contains a method pay-interest which adds interest to the balance. Under most data models, changing the interest rate entails changing code in application programs. In the object oriented model, this only entails a change within the pay interest method [10]. In commonly-known object-oriented data models attributes and associations are not defined with the class specification itself [14]. Instead, class properties are first-class primitive themselves [12]. One approach for implementing objects is to have a class, which defines the implementation for multiple objects. A class defines what types the objects will implement, how to perform the behavior required for the interface and how to remember state information. Each object will then only need to remember its individual state. Although using classes is by far the most common object approach, it is not the only approach (using prototypes is another approach) and is really peripheral to the core concepts of object-oriented modeling [19].

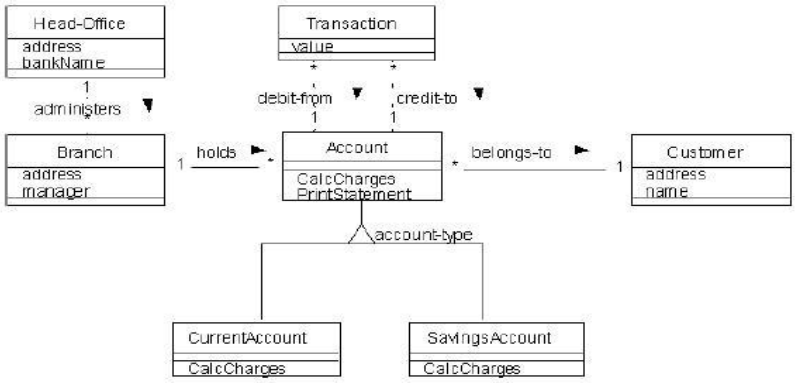

Fig. 1 Example of banking system object oriented model

Unified Modeling Languages (UML) is well known and widely used object modeling that consist of concepts /entitypes/classes in a specification hierarchy, the description of concepts by attributes which have range and relationship between concepts. UML defines several types of diagram that can be used to model the static and dynamic behaviors of a system. A UML object diagram does not define a standard set of primitive types for attributes and operation declarations; however, Object Constraint Languages (OCL) does and it is proposed that these be used for ontology modeling with UML Model an ontology as a static model consisting of a class diagram to depict the classes in the domain and their relationships, an object diagram to show particular named instances of those classes. Conceptual (or Ontology) modeling deals with the question on how to describe in a declarative and abstract way the domain information of an application, its relevant vocabulary, and how to constrain the use of the data. Modeling languages like UML and Object Data Management Group (ODMG) have been developed for object oriented models in software engineering. Common to all of these newer models is the arrangement of concepts/entity types/classes in a specialization hierarchy, the description of concepts by attributes which have ranges and relationships between concepts. Concepts, relationship types and attributes abstract from concrete objects or values and thus describe the schema (the ontology). On the other hand concrete objects populate the concepts, concrete values instantiate the attributes of these objects and concrete relations instantiate relationships [10]

ONTOLOGIES MODELING

Ontology is a formal explicit description of concepts in a domain of discourse (classes/concepts). Slots/properties/roles Properties of each concept describing various features and attributes of the concept. And a facts/role restriction means restrictions on slots. Ontology together with a set of individual instances of classes constitutes a knowledge base. In reality, there is a fine line where the ontology ends and the knowledge base begins. Classes are the focus of most ontologies. Classes describe concepts in the domain [19]. For example, a class of banking accounts represents all accounts. Specific accounts are instances of this class. The own account (e.g. Waralak account) is an instance of the class of accounts. A class can have subclasses that represent concepts that are more specific than the super class. For example, we can divide the class of all accounts into saving accounts, and checking accounts. Slots describe properties of classes and instances: Saving Account has a specific requirement; it is opened by the bank branch. We have slots describing the account in this example: the slot branch with the value New York Fifth Avenue branch. At the class level, we can say that instances of the class Account will have slots describing their account number, name, address, the branch of the account and so on. All instances of the class account, and its subclass Saving, have a slot branch the value of which is an instance of the class Branch. All instances of the class Branch have a slot open that refers to all the accounts (instances of the class Account and its subclasses) that the branch opens an account for a particular customer. Relation Of Ontologies With Maintainability In Object-Oriented System The software engineering contribution in object-oriented created a SWOOP ontology editor[29] whose debugging features were instrumental in locating logical inconsistencies and repairing them. At the time of maintainability there are some meta-data object oriented system is defined explicitly, then to remove all the data discrepancy from Resource development framework (RDF). We use approach ontologies, through which we can illuminated gaps in our knowledge of the way some object-oriented programming languages were structured. It has been identified as common reasons to use an ontological approach [23]. The term “object-oriented class” is used to disambiguate like a class from the OWL class, which can define the context to be clear and relationship captured object-oriented class may implement an interface, extend a super class, contain methods.

CONCLUSIONS:

Object technology supports the ability to build applications by selecting and assembling objects from libraries. If a developer must create a missing object to meet the application's requirements, that new object may be placed in a library for reuse in future applications. For many simple systems, the developer may use the available objects to form the entire application instead of writing code. More complex development

efforts require the developer to modify the objects to meet specific requirements. Ontologies are promised to bright future. In this paper we propose that as ontologies are closely related to modern object-oriented software engineering, it is natural to adapt existing object oriented software development methodologies for the task of ontology development. This is some part of similarity between descriptive ontologies and database schemas, conceptual data models in object oriented are good applicant for ontology modeling, however; the difference between constructs in object models and in current ontology proposals which are object structure, object identity, generalization hierarchy, defined constructs, views, and derivations. We can view ontology design as an extension of logical database design, which mean that the training object data modelers could be a promising approach. An ontology use the equivalent of database schema But ontology represent a much more richer information model than normal database schema, and also a richer information model compared to UML class/object model.

Ontology is not meant to replace various software technologies in the procedural computation category, such as Java, SQL, data mining, statistics, etc. Instead, ontology brings most value when it is used in combination with such procedural technologies. For example, ontology cannot replace data-mining algorithms based on pattern matching or statistical techniques. However, it can help make data-mining procedures more efficient, adaptive, and smart by externalizing and organizing domain knowledge the data mining algorithms use in ontological models. For another example, ontology cannot replace software-engineering technology using object-oriented analysis and modeling. However, it can help software-engineering tools validate the generated models by externalizing and organizing metadata of the models in ontological models. For yet another example, ontology is not meant to replace database technology for storing large-scale data sets. However, it can be used with databases to provide a conceptual view of various data sources scattered in a number of databases with an ontological model, and virtually integrate (federate) the data sources without replicating data instances

REFERENCES

1. W. Vongdoiwang, .D. N. Batanov. (2004). Similarities and Differences between Ontologies and Object Model. CCCT'05 proceeding 2004. Austin, Texas.
2. S. Cranefield, M. Purvis. (1999). UML as an Ontology Modeling Language. Proceeding of the IJCAI-99 Workshop on Intelligent Information Integration, Department of Information Science, University of Otago, New Zealand.
3. O. R. Zaï ane (1995). The Object-Oriented Model. [Online]. Available: http://www.cs.sfu.ca/CC/354/zaiane/material/notes/C hapter1/ node8.html
4. Object Oriented Database. [Online]. Available: http://www.profc.udec.cl/~gabriel/tutoriales/giswb/vo l1/cp4/cp4-6.htm
5. Available: http://www.cs.vu.nl/~mcaklein/papers/oil-xmls.pdf [6] J. Angele, S.Staab, H. Schurr, Object Oriented Logics for Ontologies. Draft Whitepaper Series, Karlsruhe, Germany, 2003.
6. N. Cullot, C. Parent, S. Spaccapietra and et. Ontologies : A contribution to the DL/DB debat. to appear.
7. R. Volz, D. Oberle, R. Studer. (1999). Views for light-weight web ontologies. Proceeding of SAC 2003, Melbourne, Florida, USA.
8. B. Wouters, D. Deridder, E. V. Paesschen. (2000). The use of Ontologies as a backbone for use case management ", This research was partially supported by Wang Global and the Brusseles Capital Region (CCOOS Project), Belgium, to appear.
9. D. E. Jenz. (2003). It is High Time for Pursuing the Ontology-Centric Approach
10. J. Heflin, M. N. Huhns. (2003). The Zen of the Web. Guest Editors' Introduction, IEEE Internet Computing, pp. 30-33, September-October 2003.
11. S. Strom. (2000). Building a Large-Scale Generic Object Model:Applying the CYC Upper Ontology to Object Database Development in Java. [Online]. Available: www.techtrader.com.
12. Pandey A.,Bansal K.K.(2014): "Performance Evaluation of TORA Protocol Using Random Waypoint Mobility Model" *International Journal of Education and Science Research Review* Vol.1(2)
13. Tiwari S.P.,Kumar S.,Bansal K.K.(2014): "A Survey of Metaheuristic Algorithms for Travelling Salesman Problem " *International Journal Of Engineering Research & Management Technology* Vol.1(5)
14. Yadav R.K.,Bansal K.K.(2012) : "Analysis of Sliding Window Protocol for Connected Node" *International Journal Of Soft Computing and Engineering(IJSCE)* Vol.2(5) pp.292-294
15. P. Mohan, C. Brooks. (2004). Learning Objects on the Semantic Web. ChiMu Corporation, " Object Modeling ". Foundations of O-R Mapping, [Online]. Available: http://www.chimu.com/publications/-objectRelational/part0003.html.

16. B. Morgan. Java and the Component Object Model. [Online]. Available: http://docs.rinet.ru/ZhPP/ch16.htm#TypeLibrariesand ObjectDescriptionLanguage
17. J. Greenfield. (2004). The Case for Software Factories. JOURNAL3: Microsoft Architects Journal, Issue 3, July 2004. [Online]. Available: http://msdn.microsoft.com/library/default.asp?url=/li brary/enus/dnmaj/h tml/nexgen.asp
18. Netmation Inc. (2005). Object Oriented developments [Online] Available:http://netmation.com/exp0028.htm
19. G. Engels., L. Groenewegen. (2000). Object-Oriented Modeling: A Roadmap [Online]. Available: www.cs.ucl.ac.uk/staff/ A.Finkelstein-/fose/finalengels.pdf
20. N. F. Noy., D. L. McGuinness. (2001). Ontology Development 101: A Guide to Creating Your First
21. Ontology [Online]. Available: protege.stanford.edu/publications/ontology_develop ment/ontology101.p df
22. J. Rumbaugh, M. Blaha, W. Premerlani, F. Eddy, and W. Lorensen. (1991). Object-oriented modeling and design. Englewood Cliffs, New Jersey: Prentice Hall.
23. DAML Ontologies by Keyword (account register). [Online]. Available: http://www.daml.org/cgibin/hyperdaml?http://www.d aml.org/2001/06/e xpenses/check-ont
24. Object Oriented Analysis and Design Using UML Mark Collins-Cope Objective view software development Mangazine Issue 9; Ryby, Rails, Ajax, AspectJ [Online]. Available: http://ratio.co.uk/W1.html
25. N. F. Noy, and D. L. McGuinness. mentioned in that ontologies can build on the experience using Protégé-2000 (Protege 2000), Ontolingua (Ontolingua 1997), and Chimaera (Chimaera 2000) as ontology-editing environments.
26. G. Wilkie. (2001). Object-Oriented Software Engineering - The professional Developer's Guide(on OMG's OOA/OOD proposal) [Online]. Available: www.idi.ntnu.no/grupper/su/courses/dif8901-/presentations2001/a01-wilkie.ppt
27. D. L. McGuinness, F. V. Harmelen. (2004). OWL Web Ontology Language Overview. [Online]. Available: http://www.w3.org/TR/owl-features/
28. M. Denny. Language Suitability: Ontology Tools Survey, Revisited. [Online]. Available: http://www.xml.com/pub/a/2004/07/14/onto.html?-page=2
29. R.G.G. Cattell. (1994). The Object Database Standard: ODMG-93, Release 1.1, Morgan Kaufmann Publishers, San Francisco [Online]. Available:http://www.objs.com/x3h7/odmg.htm.
30. Alphawork. What is ontology? Frequently asked questions. [Online]. Available: http://www.alphaworks.ibm.com/contentnr/semantics faqs
31. Kalyanpur A., Parsia, B., Sirin, E., Cuenca-Grau, B., Hendler, J.: Swoop: A 'Web' Ontology Editing Browser, Journal of Web Semantics Vol 4(2).