INTERNATIONAL JOURNAL OF PURE AND APPLIED RESEARCH IN ENGINEERING AND TECHNOLOGY

A PATH FOR HORIZING YOUR INNOVATIVE WORK

TRANSFORMATION OF UML SEQUENCE DIAGRAM TO JAVA CODE

HARSHAL D. GURAD[1], PROF. V. S. MAHALLE[2]

1. Post Graduate Student, Department of CSE, SSGMCE, Shegaon, Maharashtra, India.
2. Assistant Professor, Department of CSE, SSGMCE, Shegaon, Maharashtra, India.

Accepted Date: 15/02/2014 ; Published Date: 01/04/2014

Abstract: Unified Modeling Language (UML) is a visual modeling language. It has now become the de-facto industry standard for object-oriented (OO) software development which consists of designing and coding of the software. It has gained popularity among software practitioners because of its benefits, such as, cost reduction and accuracy. So, automatic code generation from UML diagrams is more beneficial because it eliminates the chances of introduction of human errors in the translation process and also reduces the time required for coding. This paper proposes an approach to code generation from UML diagrams. It mainly focuses on the UML sequence diagram as the model and the transformation of UML diagram to XMI is done with help of some pre-existing tool. The meta-elements are extracted from the XMI file with the help of the methods explained in the work. The output will be the java files having the consistent and redundancy free code equivalent to the given sequence diagram.

Keywords: UML, XMI, Java, Code Generation, Sequence Diagram.

PAPER-QR CODE

Corresponding Author: MR. HARSHAL D.GURAD

Access Online On:

www.ijpret.com

How to Cite This Article:

Harshal Gurad, IJPRET, 2014; Volume 2 (8): 703-710

INTRODUCTION

Nowadays software programs are more large and complex and also regular maintenance is required by these programs. So, to deal with such situations, many software developers have adopted the OO analysis and design paradigm[1]. Object Management Group (OMG)[2] introduced unified modeling language (UML) [3,4] to facilitate the use of standardized notations for object-oriented analysis and design. UML is the open, industry-standard visual modeling language and UML is mostly associated with modeling object oriented software systems.

UML is widely used in the software development, there consists of the designing and coding of the software. Designing phase is done with the help of the UML models, which consists of the statemachine diagrams, usecase diagrams, activity diagrams etc. Since the software development is time and human resource consuming, the reduction of consumption is done with the help of generation of code automatically without the coding in the development phase. [5,6]

So, the code generators are used. They are used to increase code quality and decrease envelopment time, since their goal is to generate repetitive source code while maintaining a high consistency level of the generated program code. The code generators provide more productivity; generate great volumes of code, which would take much longer if coded manually. Even though UML is being widely used, its full potential in helping software development activities is yet to be exploited. One potential use of sequence diagram is automatic code generation with which we can avoid the human errors during manual translation of the SDs to equivalent code, improve communication between design team and coding team, and reduce coding effort [6, 7].

In this work, we propose an approach to generate code from UML sequence diagram. We first transform UML diagram to XMI is done with help of some pre-existing tool. The meta-elements are extracted from the XMI file with the help of the methods explained in the work. The output will be the java files having the code equivalent to the given sequence diagram.

RELATED WORK

In this section, we review existing tools and approaches related to UML-based code generation. The tools such as Omondo EclipseUML[9],Altova Umodel[10],Magic Draw[11],Visual Paradigm[12] support code generation from class diagrams. To generate behavioural diagrams, UML behavioural diagrams such as statechart, collaboration and sequential diagrams are used.

Visual Paradigm [12] generates the behavioural code from statechart diagrams using templates for different operations and information associated with the transitions in the statechart diagrams.Niaz and Tanaka[13] propose an approach to generate Java code from UML class and statechart diagrams.Tanaka et al [13,14,15] reports a series of work on code generation from UML statechart diagrams.Engles et al.[16] propose a transformation process based on collaboration diagram to generate Java code.HiberObjects[17] generates code from UML 1.x SDs using templates for specific services supported by different types of object.

There are also some existing tools to automatically generate code from UML diagrams such as OCode, JCode, Rhapsody, dcode etc. OCode was developed by Ali and Tanaka [18] to generate Java code from UML state diagram. The tool takes state diagram represented in Design Schema List language (DSL) as an input. It consists of two components known as interpreter and code generator. The interpreter generates a transition table from the DSL while code generator generates Java code from the transition table. JCode was implemented by Niaz and Tanaka [13] to generate Java code from UML statechart. The tool takes statechart represented in DSL as an input. The tool consists of three components known as interpreter, transformer and code generator. The interpreter takes DSL as an input and generates intermediate transition table. The transformer generates transformed transition table from the intermediate transition table. The transformer focuses on resolving concepts like state hierarchy, state composition, compound transition, etc in the intermediate transition table. The transformed transition table is used as an input to the code generator to generate Java code. Rhapsody was developed by Harel and Gery [19] to generate C++ code from UML object and statechart diagrams. It also takes message sequence charts (MSC) as an additional input. The tool takes STATEMATE representation of statechart as an input and generates C++ code. dCode was created by Ali and Tanaka [18] to generate Java code from UML object, activity and statechart diagrams. The tool follows the same code generation process as described for OCode [12].

OVERVIEW

Software development process consists of designing and coding of the software. In the software development process, complexity and time required for designing and coding depends on the complexity of the software program. Coding of the software may create overhead in the development process as compared to that of design phase because of the complexity involved in the coding phase which in terms increases the time for development phase. Consider an example of software development, consisting of the design and coding of the software. Here the human resource will get divided into design and coding team, in which the coding will definitely consume the larger amount of it. So to overcome such situation, the automatic code generation from design model diagrams is introduced which also can avoid the human errors during manual translation of the SDs to equivalent code, improve communication between design team and coding team, and reduce coding effort. For the designing of the software concept and the flow of the data and control in the software UML diagrams are use. UML has mainly two versions, they are, UML 1.x and UML 2.x. While UML 2.x is mostly used, as it supports the superstructure specification [19].There are thirteen different types of UML diagrams. We can usefully divide this set of diagrams into static model and dynamic model. The static models captures the things and the structural relationships between the things and are considered as the Structural Diagrams; the dynamic model captures how things interact to generate the required behavior of the software system and are considered as the Behavior Diagrams An UML behavior diagram includes a set of interaction diagrams such as sequence, communication, interactionoverview, timing diagrams in which interactions are involved. Sequence diagram describes the sequence of messages that are exchanged, along with their corresponding occurrence specifications on the lifelines.

In this paper the code generation happens based on the sequence diagram, which focuses on the message interchange between a numbers of lifelines. Each lifeline represent the participant in the interaction, here it is the object of the class. The message passed between the objects is the sequence flow of the program. Message can be defined as a named element, one specific kind of communication between lifelines of an interaction. Each message reflects either an operation call or sending and reception of a signal, so that message could be of actions like: synchronous call, asynchronous call and asynchronous signal.[8]

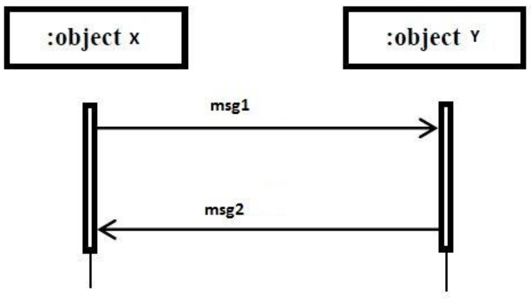

Fig. 1 Sequence Diagram

Also each message specification have the sender and the receiver in which the sender will be calling or passing a message to the receiver and receiver may call or send message back to the sender as per requirement. In the above sequence diagram, each message can be considered as the sequence flow from each of the class object in the timeline. The Fig.1 shows a sample sequence diagram which consists of two objects X and Y. Messages msg1 and msg2 are passed between the objects X and Y. Here object X is the sender of msg1 and object Y is the receiver of msg1. While that for msg2 object Y is the sender and object X is the receiver. The flow of the message shows the order of execution of the code. As in sample figure, the message msg1 is happening before m2, so that the sequence flow is message msg1 after that message msg2

CODE GENERATION APPROACH

In this work, we first take UML sequence diagram as a input and convert it to the corresponding XML Metadata Interchange (XMI). XMI is a metadata representation of the model, which is used for the conversion of the model to text and wise versa. XMI format is the combination of the XML tags and the UML elements. Figure 2 shows the code generation process which shows the flow of data, that is the conversion of the sequence diagram to the corresponding XMI format and this XMI file is then transformed to respective code. Code obtained from the XMI Representation may not be consistent. So, the obtained code is optimized first so as to obtain more consistent, redundancy free code. In this way, we obtain the final generated code from the optimization of generated output from XMI representation of sequence diagram.

IMPLEMENTATION METHODOLOGY

First we take UML sequence diagram as anis the input to the system and convert it to the XMI format which is having the metadata information of the design model. The XMI representation of the UML sequence diagram is generated with the help of some freely available pre-existing tool.

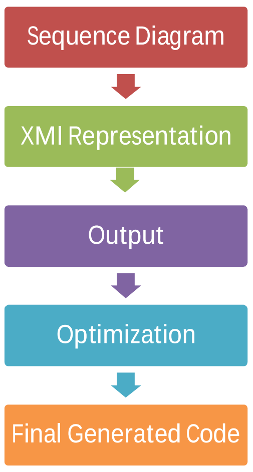

Fig.2 Code Generation System.

Implementation Steps

1. UML sequence diagram is modeled in the tool such as UMLGraph

2. Export the XMI representation of the sequence diagram from the tool for java language.

3. Extract the metadata from the XMI file.

Implementation of step 1 and 2 can be done with the help of some pre-existing tool such as UMLGraph , which is available on the internet freely. But for the step 3, we have to follow the following procedure.

Extraction of the metadata information from the XMI file is the most important step in this work. The class information, its visibility and the methods and attributes are contained within the XMI file. Each of the class has its own messages passed between the objects.

First we extract the class name, methods and its visibility.

Then we extract the operations and parameters related with the operation contained within the class. By doing this we get the operation name and visibility.

Now we extract the operation name, its visbility, its return type, parameter and its data type of all the classes contained within the XMI file.

Now we start to extract the sequence flow of project.

And now finally we complete the sequence flow of the project.

By applying this methodology we can generate the code for the XMI representation of the given sequence diagram. If we don't take both conditional and the looping statements into consideration then the code obtained will be simplex one and we don't need to perform any optimization for that code. But, if we take both conditional and the looping statements into the consideration, the code obtained will be redundant and inconsistent. So, to avoid redundancy and inconsistency within the code we need to perform optimization of the generated code with the help of optimizer. And then finally we get our final consistent and redundancy free final generated code.

CONCLUSION

This paper proposes an approach to code generation from UML sequence diagram. This paper briefly explain ways for generation of code from the XMI representation of the given diagram

For this we are making use of xml metadata interchange (XMI) of given sequence diagram. We can efficiently extract simple sequence flow from XMI representation of corresponding sequence diagram. While we have to use some complex programming methodology for extracting sequence flow containing conditional and looping statements.

REFERENCES

1. C. Larman: Applying UML and patterns: an introduction to object-oriented analysis and design and the unified process (Prentice-Hall, PTR, 2001, 2nd edn.)

2. http://www.omg.org/spec/uml/2.2/superstructure

3. G. Booch, J. Rumbaugh, I. Jacobson,: 'Object-oriented analysis and design' (Addison-Wesley, 2002)

4. Pilone, D., Pitman, N.: 'UML 2.0 in a nutshell' (O'Reilly, 2005)]

5. Usman, M., Nadeem, A.: _Automatic generation of Java code from UML diagrams using UJECTOR', Int. J. Softw. Eng. Appl., 2009, 3, (2), pp. 21–38. R. E. Sorace, V. S. Reinhardt, and S. A. Vaughn, "High-speed digital-to-RF converter," U.S. Patent 5 668 842, Sept. 16, 1997.

6. Jakimi, A., El Koutbi, M.: _An object-oriented approach to UML scenarios engineering and code generation', Int. J. Comput. Theory Eng., 2009, 1, (1), pp. 35–41.

7. Usman, M., Nadeem, A.: 'Automatic generation of Java code from UML diagrams using UJEC TOR', Int. J. Softw. Eng. Appl., 2009, 3, (2), pp.21–38

8. http://www.uml-diagrams.org/sequence-diagrams.html.

9. 'http://www.ejb3.org/', 2 November 2010

10. http://www.altova.com/umodel/uml-code-generation.html', 2 November 2011

11. http://www.magicdraw.com/', 2 November 2010

12. http://www.visual- paradigm.com/product/vpuml/', 2 November 2010

13. Niaz, I.A., Tanaka, J.: 'An object-oriented approach to generate Java code from UML statecharts', Int. J. Comput. Inf. Sci., 2005, 6, (2)

14. Ali, J., Tanaka, J.: 'Converting statecharts into Java code'. Proc. Of Fourth World Conf. on Integrated Design and Process Technology (IDPT99), Dallas, Texas, USA, 2000

15. [Niaz, I.A., Tanaka, J.: 'Mapping UML statecharts to Java code'. Proc. Of IASTED Int. Conf. on Software Engineering (SE 2004), Innsbruck, Austria, 2004, pp. 111–116

16. Engels, G., Hucking, R., Sauer, S., Wagner, A.: 'UML collaboration diagrams and their transformations to Java'. Proc. of the Second Int. Conf. on the UML, 1999, (LNCS 1723) pp. 473–488

17. http://objectgeneration.com/eclipse/04-sequencediagrams.html', 2 November 2010

18. J. Ali, and J. Tanaka, "An Object Oriented Approach to Generate Executable Code from the OMT-based Dynamic Model", Journal of Integrated Design and Process Science (SDPT), vol. 2, no. 4, 1998, pp.65-77.

19. D. Harel, and E. Gery, "Executable Object Modeling with Statecharts", 18th International Conference on Software Engineering (SE'96), IEEE Computer Society Press, Berlin, Germany, March 25-29, 1996, pp.246-257.