www.ijecs.in
International Journal Of Engineering And Computer Science ISSN:2319-7242
Volume 4 Issue 4 April 2015, Page No. 11332-11344

Transformation Of Analysis Model To Business Process Model

[1].Kulshan Pattanaik
M.E.(CSE) Student
Dept.of Computer science and Engineering,NITTTR
Kulshan_pattanaik@yahoo.com
Laser Ignition System For Internal Combustion Engine

Abstract: Software projects have become larger over the time and therefore the software requirements specifications have grown as well. One way to document requirements is the use of Use Cases, which describe the interaction of an actor with the desired system. Commonly used UML diagrams are - Use Case, Activity, Sequence, collaboration, statechart and Class diagrams. Use case and activity diagrams model the behavioral aspect of the system, whereas, Class diagrams represent the static design of a system. UML,being visual in nature, is easy to understand and communicate. However, as the number of Use Cases increases with the described functionality, the overview, the dependencies and possibly the execution order of Use Cases is lost more and more. Therefore, the task of locating specific Use Cases in large documents and avoiding contradictions becomes a hassle. Design and development of software has become much more complex in the last decade, resulting in evolution of design and development paradigms. Object oriented systems have thus become an integral part of more complex Service Oriented Architecture (SOA) to address complex issues. Automatic translation of UML use case and activity models to BPMN design elements would ensure consistent evolution of Object oriented systems to Service oriented paradigm.

1. Introduction

A modeling language is an artificial language which is used to express information or knowledge or systems in a structure that is defined by a consistent set of rules which are used to interpret the meaning of the components in the structure. A modeling language can be graphical or textual. Graphical modeling languages use a diagram technique with named symbols that represent concepts and lines that connect the symbols and represent relationships and various other graphical notation to represent constraints whereas Textual modeling languages typically use standardized keywords accompanied by parameters to make computer interpretable expressions[1][2].

1.1 Type of modeling languages

There are different types of modeling languages such as graphical, algebraic, behavioral, domain-specific, and object oriented.Graphical modeling languages are primarily used to represent the natural language requirements that are typically used to express the stakeholder's needs for a large-scale software- system. It is primarily used in the field of software engineering. Business Process Modeling Notation (BPMN) is an example of a Process Modeling language (PML).Algebraic Modeling Languages (AML) are high- level programming languages that is used to describe and solve problems of higher complexity like large scale mathematical computation or large scale optimization problems. The advantage of using AML is the similarity of its syntax to the mathematical notation of optimization problems. This allows a very concise and readable definition of problems in the domain of optimization like sets, indices, algebraic expressions, powerful sparse index and data handling variables, constraints with arbitrary names. The algebraic formulation of a model does not contain any hints how to process it. Behavioral languages are designed to describe the observable behavior of complex systems consisting of components that execute concurrently. These languages focus on the description of key concepts suchas:concurrency,nondeterminism, synchronization, and communication. The semantic foundations of Behavioral languages are process calculus or process algebra. Java modeling language is an example.Domain-specific modeling (DSM) is a software engineering methodology used to design and develop IT systems such as computer software. It involves systematic use of a graphical domain- specific language (DSL) to represent the various facets of a system. DSM languages tend to support higher-level abstractions than General-purpose modeling languages, so they require less effort and fewer low-level details to specify a given system.Object modeling language are modeling languages based on a standardized set of symbols and ways of arranging them to model an object oriented software design or system design. It is commonly used in combination with a software development methodology to progress from initial specification to an implementation plan and to communicate that plan to an entire team of developers and stakeholders. Because a modeling language is visual and at a higher-level of abstraction than code, using models encourages the generation of a shared vision that may prevent problems of differing interpretation later in development. Often software modeling tools are used to construct these models, which may then be capable of automatic translation to code.

2. UML Overview

UML is a standard language used to specify, visualize, construct, and document the artifacts of software systems [2][5][6][7][8].UML was created by Object Management Group (OMG) and UML 1.0 specification draft was proposed to the OMG in January 1997. UML is different from the other common programming languages like C++, Java, COBOL since it deals with the use of pictorial languages which makes it simple to understand and implement it to make software blue
prints.

2. Process Modeling

Process Modeling is the art of capturing of an ordered sequence of business activities and supporting information. There are different levels of process modeling like Process Maps that deals with simple flow charts of the activities, Process Descriptions which includes flow charts extended with additional information, but not enough to fully define actual performance and Process Models dealing with flow charts extended with enough information so that the process can be analyzed, simulated, and/or executed. Business Process Modeling Notation (BPMN) is an example of process modeling[9][10].Two new concepts were introduced in BPMNwhich created a big difference between UML and BPMN and they are Orchestration which defines processes internal to a specific participant, organization. It is a private process within pool (and between lanes).Sequence Flows and Message Flows are a part of these and Choreography that defines interactions between two or more participants, organizations, B2B, global processes. It deals with public exchange of messages between pools in a collaboration.

4. Transformation Rules:-

We used a set of rules to transform Formalize Analysis model into BPMN Notation.The rules are the defined as follow:

Rule1

The use case activity whose node is marked as start' will be assigned as Start Event of the BPMN node. The BPMN node will labeled as Activity ID (act_ID) of the activity node.

Rule 2

The use case activity whose node is marked as 'end' will be assigned as End Event of the BPMN node. The BPMN node will labeled as Activity ID (act_ID) of the activity node.

Rule3

The use case activity whose node is marked as 'action / decision' will be assigned as Intermediate Event of the BPMN node. The BPMN node will labeled as Activity ID (act_ID) of the activity node.

Rule 4

The use case activity whose node is marked as 'fork' will be assigned as Parallel Gateway of the BPMN node if both the post Element of the activity node are of the type 'basic'. The BPMN node will labeled as Activity ID (act_ID) of the activity node.

Rule 5

The use case activity whose node is marked as 'fork' will be assigned as Exclusive-OR Gateway of the BPMN node if one the postElement of the activity node are of the type 'basic' and the other is of the type 'alternate'. The BPMN node will labeled as Activity ID(act_ID) of the activity node.

5.Algorithm for automated transformation:

These rules cited in the above section are realized using two algorithms namely NodeGeneration and FlowGeneration. The flow of the algorithm is as follow: The elements of Formalized Analysis Model (FAM) in the form of different table schema are used as input to the first algorithm named NodeGeneration. The outputs of this algorithm are different BPMN nodes. This output along with the Array FAM_Flow is fed as input to the second algorithm named FlowGeneration. The Array FAM_Flow is formal method of storing the flow information of events of FAM. The Flow Generation algorithm will generate the BPMN design elements

6. Algorithm for Conversion of scenarios to BPMN:-

The algorithm Node Generation as proposed below will generate the BPMN nodes. We define the algorithm using the tuple relational calculus. The algorithm is proposed as follow.# The following query is the realization of rule 1 of .It generates the Start of the #BPMN node. It selects the Graphical_notation from BPMN_node and map that #with the start event of the activity_node {t. Graphical_notation | BPMN_node (t) ^ t. ID = '1' ^ □ d (d.act_ID | ActivityState (d) ^ d.activity_node = t.activity_node ^ t.label = d.act_ID ^ d.activity_node = 'start') # The following query is the realization of rule 2.It generates the End #of the BPMN node. It selects them Graphical_notation from BPMN_node and map #that with the end event of the activity_node {t. Graphical_notation | BPMN_node(t) ^ t.ID = '2' ^ □d(d.act_ID | ActivityState(d) ^ d.activity_node = t.activity_node ^ t.label = d.act_ID ^ d.activity_node = 'end')}

7. Conclusion

BPMN supports detection of wrong conditions, missing use cases, unknown parallelism. Within this project we have presented an approach to visualizing dependencies and ordering for use case sets. By using this approach, it is possible to create BPMN models that make the dependencies between use cases explicit by using preconditions, post conditions and triggers. Stake holders and requirements engineers are therefore able to discuss the ordering and the conditions of the use cases, which can greatly improve the quality of the overall use case model. In this project ,we have proposed an approach for automated translation of Formalized A nalysis Models, that consists of a formal grammar based description of UML models used in Analysis phase, to Business Processes Design and development of software has become much more complex in the last decade resulting in evolution of design and development paradigms. Object oriented Systems have thus become an integral part of more complex Service Oriented Architecture(SOA).Evolution of software design and development from object oriented to SOA domain has become the necessity in this evolving scenario. Thi approach would help us in seamless evolution of object oriented systems to service oriented domain. As this model is based on a formal grammar, this model can be automated resulting in correct and consistent transformations. The goal of this project is to propose a frame work for automatic transformation of UMLanalysis models to BPMN design elements.

References

1. A. Cockburn "Writing Effective Use Cases," Addison-Wesley, 2005.
2. R. Dijkman, M. Dumas and C. Ouyang "Semantics and Analysis of Business Process Models in BPMN," Information and SoftwarE Technology (IST), 2008.
3. D.L¨ubke"Transformation of Use Cases to EPC Models,"In Proceedings of the EPK 2006 Workshop.
4. ObjectManagementGroup"UnifiedModelingLanguage Superstructure,"2004
5. Object Management Group. Business Process Modeling Notation (BPMN) 1.1," 2008.
6. B. Oestereich, C. Weiss, C. Schr¨oder, T. Weilkiens, and A. Lenhard "UML,"2003.
7. S. Som´e. "An approach for the synthesis of State transition graphs from Use Cases,". In Proceedings of the InternationaL Conference on Software Engineering Research and Practice, 2003
8. S. Som´e. "Supporting Use Cases Based Requirements Simulation,"In Proceedings of the International Conference on Software Engineering and Practice (SERP'04) 2004
9. J. Whittle, "A Formal Semantics of Use Case Charts," Technical Report ISE TR-06-02
10. .Daniel L¨ubke , Kurt Schneider and Matthias Weidlich, "Visualizing Use Case Sets as BPMN Processes", Requirements Engineering Visualization 2008
11. Jayeeta Chanda, Ananya Kanjilal, Sabnam Sengupta, Swapan Bhattacharya"Traceability of Requirements and Consistency Verification of UML Use case, Activity and Class Diagram A Formal Approach," Proceedings of International Conference on Methods and Models in Computer Science (IEEE ICM2CS), 2009.
12. Jayeeta Chanda, Ananya Kanjilal, Sabnam Sengupta, Swapan Bhattacharya "FAM2BP: Transformation Framework of UML Behavioral Elements into BPMN Design Element,".
13. Xiao He, "A Metamodel for the Notation of Graphical Modeling Languages," 31stAnnual International Conference on Computer Software and Applications 2007.
14. Bock and D. Ryan, "Accuracy in Modeling with Extended Entity Relationship and Object - Oriented Data Models".
15. S.R. Chidamber and C.F. Kemerer, "A Metrics Suite for Object Oriented Design," IEEE Transactions on Software Engineering, *1994.*
16. Long, Q., Liu, Z., Li, X. and Jifeng, H, "Consistent Code Generation from UML Models," In Proceedings of Australian Software Engineering Conference, IEEE Computer Society, 2005.
17. Ouyang, C. Dumas, M. ter Hofstede and A.H.M. van der Aalst, "W.M.P. Pattern-based Translation of BPMN Process Models to BPEL Web Services," International Journal

   of Web Services Research 5, 2008
18. Wahl, T. Sindre, G "An Analytical Evaluation of BPMN Using a SemiotICnQuality Framework," In: Siau, K. (ed.) Advanced Topics in Database Research, Idea Group, Hershey, Pennsylvania 2006.
19. S. White, "Using BPMN to Model a BPEL Process, " Technical report, BP Trends 2005.
20. J. Ye, S. Sun, L.Wen, and S.Wen, "Transformation of BPMN to YAWL," In H.Zhou, editor, International Conference on Computer Science and Software Engineering, 2008.
21. J. Camara, C. Canal, J. Cubo, and A. Vallecillo, "Formalizing WSBPEL Business Process using ProcessAlgebra,'' InCONCUR'2005
22. M. Dumas, A. Grosskopf, T. Hettel, and M. Wynn, "Semantics of Standard Process Models with OR-Joins,'' Technical report, Queensland University of Technology, BPM Center *Technical report* 2007.
23. O. M. Group, "Business Process Modeling Notation (BPMN) Version 1.0. OMG Fina Adopted Specification," Object Management Group, 2006.
24. Carbone, M. and Santucci, and G, "Fast & Serious: a UML Based Metric for Effort Estimation," In 6th ECOOP
25. Workshop on Quantitative Approaches in Object-Oriented Software Engineering 2011.
26. Bock, D., Ryan and T, "Accuracy in modeling with extended entity relationship and object oriented data models,"Journal of Database Management 1993.
27. Shoval, P. and Shira, "Entity-relationship and object-oriented data modeling:an experimental comparison of design quality," Data Knowledge

Engineering of the 10th WSEAS international conference on evolutionary computing, 1997.