ISSN:0975-9646

A Review of Model Based Slicing

Tamanna[1] , Dr. Sukhvir Singh[2]

Computer Science and Engineering, N.C. College of Engineering, Israna Panipat, India

***Abstract*- Software testing is important to reduce errors, maintenance and overall software costs. Testing the software is an activity whose aims to evaluate the feature or competency of system and determining that whether it meets required prospects. To comfort this one way is program slicing, this method is to break down the large programs into smaller ones and other is model based slicing that break down the large software architecture model into smaller models at the initial stage of SDLC (Software Development Life Cycle). It is completely a new approach to extract the sub model from a big model diagrams on the basis of slicing criteria. The planned procedure used the concept of model based slicing to slice the sequence diagram to extract the desired piece. This literature survey presents an overview of Model based slicing, including the various general methods and techniques used to compute slices.**

***Keywords*— Model Based Slicing, Feature Based Slicing, UML/OCL Model Verification, Model Transformation Verification Through Slicing, Dependency Graph, Model.**

I INTRODUCTION

For better visualization of architecture and due to the increase in size and complexity of software products the significance of architectural design has been increased [1]. The architecture of an object-oriented software system defines its high level design structure and allows an architect to reason about various properties of the system at higher level of abstraction. For this, Unified Modeling Language (UML) is best choice and extensively used to represent and construct the architecture of software system with the help of its various model diagrams. UML diagrams tell us about the structural and behavioral features of architecture [2]. Structural models (e.g., class diagrams, object diagrams, component diagrams) are used to describe various relations among objects, such as aggregation, association, composition and generalization/specialization etc. On the other hand, the behavioral models (e.g. communication and sequence diagrams, activity diagram, state diagrams) are used to describe a sequence of actions, states and their interaction, through which a use case is realized [3]. The task of analyzing UML Models is bit challenging since the information regarding system can be distributed across several model views.

Slicing may be referring as process or strategy to break body of information into smaller parts to examine it from different viewpoints that will yield more information so that researcher can understand it better. The term is also used to mean the presentation of information in a variety of different and useful ways. For this various ideas, approaches and slicing techniques has been proposed by various academicians' authors and researchers. Major focus of this literature is to provide review of slicing approaches and techniques present for UML models. Section 2 provides a brief review of Slicing of UML models and various techniques and approaches used by the researches hence it contains related work. Section 3 provides the list of tools used for model based slicing. Section 4 provides the conclusion of this literature analysis.

II UML MODELS SLICING

UML language is used to design the various model by using different parameter which support functional of system [11]. Unified modeling languages are standard languages for writing blueprint for design model. Autofocus components having a common global clock such that they all perform their computations simultaneously. Each clock cycle consists of two steps: firstly each component reads the values on its input ports and computes new values for local variables and output ports such that read input data and generates resultant output [7].

UML Models Slicing is a process of breakdown to extract and identify relevant model parts or related elements across model that corresponds to user defined slicing criterion. In Model based slicing several types of model relations, and dependency such as class-class, class-operation, operation- operation, class-object, object-object, guard condition in sequence diagram , conditional predicate, control flow , data flow etc., need to be taken into account. In this work, sequence diagram has been taken into account and various approaches present till date for slicing UML diagram have been listed.

A. Slicing Methodologies for UML Models

1) Using Dependency Relationships

Dependency Graph is an intermediate representation step while slicing UML Models that can define the various types of dependencies. Zhao [4] introduced the concept of architectural slicing which operates on architectural description of software system. According to the proposed architectural description there will be three types of dependencies. First is component-connector dependency where information flows from port (interface) of a component to role of a connector. Second type is connector component dependency in which information flow is from role of connector to port of component. Third type of dependency is additional dependency which can be used to represent a relation between two ports or roles within a component or connector.

Fangjun et.al [6] presented a method for slicing hierarchical automata. The given approach was based on representing the UML state chart by hierarchical automation for modeling dynamic aspects of software. The proposed method reduces the state space during model checking of UML state chart. The output slice proposed by technique is Extended Hierarchical automation instead of UML State chart models. The importance of Fangjun algorithm is its ability to remove the hierarchies and concurrent states, which are irrelevant to the properties of the hierarchical automation.

Kagdi [8] developed model slices from UML class models. His approach was to extracts parts of a class diagram in order to construct sub models from a given model of a system. Sutton et al. [8] presented the concept of model slicing to support maintenance of software through querying understanding, and analyzing large UML models. However, class models are lacking of explicit behavioural information and represent only structural behaviour. For the purpose of model slicing they define a model 'M' as directed by multi graph for finite set of elements, their set of relationships, and a function that maps element to element via a relationship.

Based upon these set of dependency relations dependence graph of UML class diagram was constructed. Maletic et al. [8] proposed an approach that contains different class relationship to define dependence relations corresponding to the relations among classes. Models that was proposed can be used in two important applications and they were slicing the architecture and measurements of coupling between component. As their graph representation has been derived from class diagrams alone, usefulness is limited to understanding static aspects of a modelled system.

For reducing the number of interference dependencies in state chart algorithm was proposed by Van [9] by using the concept of slicing with concurrent state. The proposed approach considers data dependency from the definition and use of variables that are common to parallel executing statements. He achieved this by exploiting the internal broadcasting mechanism and maintaining the state chart’s execution systematically. Chae et.al [10] proposed UML metamodel slicer to manage the complexity of UML metamodels which addresses to all UML diagram by modularizing metamodels into small metamodels.

An algorithm was proposed by Jaiprakash et al. [13][14] that generate the dynamic slices corresponding to any slicing criteria by traversing the model dependency graph which hold all the dependency of variables. Moha et al.[12] presented an approach for meta-model pruning algorithm. The proposed pruner takes input slicing criteria, i.e. operations, classes, etc of the meta- model to slice the architecture and extract all the mandatory dependencies between them. The pruner resulted into an output slice that satisfies all the structural constraints imposed by the input metamodel.

Mall et al. [15] presented a methodology to generate dynamic slices and test case with the help of UML sequence diagram. In this Message dependency graph (MDG) gets constructed which represent every message as node. To identify the provisional predicate associated with message in a sequence diagram, slicer can create dynamic slice according to the criteria. As an extension of previous work to generate automatic test case. At the designing part of SDLC according to the functionality of system, they proposed an approach [16] to use slicing technique on the UML sequence diagram. Sequence Diagram can capture time dependent sequence of interaction between different object and component. By analyzing these relation a proper functionality of the system can be visualize which can capture to generate test cases for better verification. This was the way to generate test data in their proposed approach to select conditional predicate from sequence diagram to make a slicing criteria in the slicer while keeping all other variable constant while traversing the every node of sequence diagram until the solution is found.

The slicer extract a slice according to user define slicer criteria from graph and Drawer converts the data sequence slice into sequence diagram with the help of Quick sequence diagram editor. Kobayashi et.al [17] proposed a sequence diagram slicing method to visualize the object oriented program behaviour. Nisansala et al. [18] focused on Model Checking as fully automated technique to reduce the size of model with the help of slicing.

2) Using Control and Data Flow

Many researchers dedicated their work to slice the models or architecture of the system into desirable small chunks. Control and Data Flow are the important part of system modeling or UML models that describe the nature of every component, their behavior, and working with other components and sequential pattern of interaction.

Author also represents the pre and post condition relationship of the state during path predicate coverage. Lano [19] defined that slicing can be carried out for UML state machines, using data and control flow analysis to remove elements of the machine that do not contribute to the value of a set of features in a selected state of the machine.

In this proposed method source model is taken as input with set of abstract variable then reduced by syntactic abstraction followed by semantically abstraction to generate abstract model from which symbolic tests are extracted according to selection criteria. They proposed three methods for identifying the relevant variable and generating abstract model. The first one is to consider data flow dependency only. Second one uses both data-flow and control-flow dependency. Third method is to use data flow and partial control flow dependencies to find as much as possible strong relevant variables. Julliand et.al [20] proposed an approach based on domain abstraction for generating test cases on the basis of syntactic abstraction and variable elimination with the help of model slicing.

3) Using UML/OCL Constraints

OCL allows the definition of expressions on UML models, an expression that evaluates the true or false of class invariant, or constraint. In another approach [21][22] author proposed a tool (UOST) to enable the efficient verification of UML/OCL Class diagram with the help of model slicing technique. The tool can verify the properties of the diagram with disjoint and non- disjoint sets of slicing.

Sarna et al. [23] proposed an algorithm for automatic generation of test cases from sequence diagrams. They first convert UML sequence diagram into graphical representation named as SDG (Sequence diagram graph). To retrieve the information for a specification of input/output, pre and post conditions for test cases generation they use the use case template, class diagram and data dictionary and expressed in OCL.

4) Using Feature Based Criteria

Archer et al. [24] projected a novel slicing technique on the feature model by taking cross-tree constraints into account with respect to set of features which are acting as slicing criteria. The core idea of proposed algorithm is to compute proposition formula representing the set of configuration and rules and to apply propositional logic reasoning techniques to construct an FM (representing its hierarchy, variability information, feature groups and cross-tree constraints).

By extended the previous author [25] also proposed the concept that how set of complementary set of operators like aggregate, merge and slice can provide practically and efficient support for separation of concerns from feature modeling. They defined that slicing process is both semantic and syntactic so they analyze the cross-cutting constraints to define the features that must be or cannot be sliced. In their proposed technique, the feature model and its cross-cutting constraints are first analyzed by transformation into predicates and then these predicates are transformed in a sliced feature model.

5) Using Model Languages

Kim [27][28] introduced the slicing technique called dynamic software architecture slicing (DSAS). Dynamic slicer takes slicing criterion as input, and reads the ADL source code of the architecture to identify the information of component and connector along with the event names used in the ADL and parameter names combined with those events.

In this approach software architecture is first designed by using ADL (Architecture description language) and later on mapped onto program statement as executable architecture. Dynamic slicer takes slicing criterion as input, and reads the ADL source code of the architecture to identify the information of component and connector along with the event names used in the ADL and parameter names combined with those events.

The proposed algorithm filters out the events that are not relevant and passes only those which are relevant to slicing criterion and generate resulting software architecture slice as shown in Fig 1.

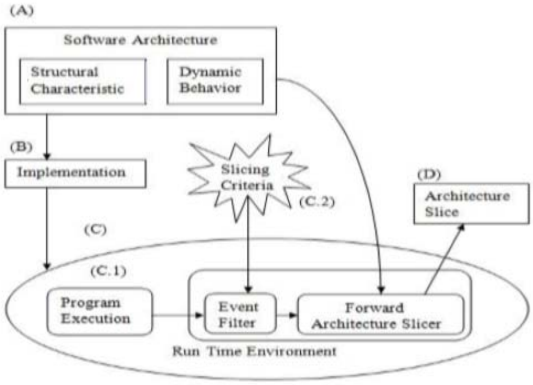


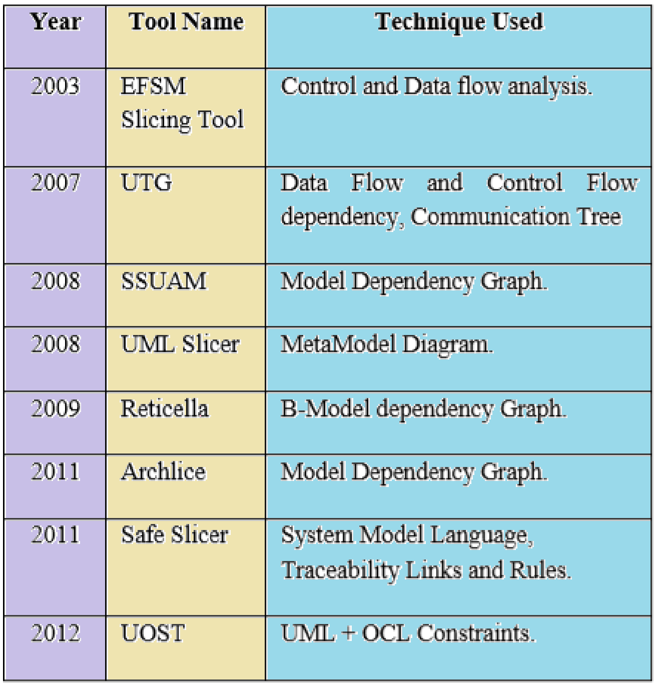
Fig 1 Dynamic Software Architecture Slicing Methodology Proposed by Kim [26]

Zoltán et.al [29][30] proposed dynamic backward slicing of model transformations technique with respect to program slicing. To slice the models they used model transformation language as a core of technique with the help of Dynamic Backward slicing by considering the Execution traces of program to generate final slice.

Blouin et.al [31] [32] proposed a DSML (Domain Specific Model Language) 'Kompren' to model the model slicers for particular domain. Kompren refers to the selection of the set of classes and relations from the input metamodel expressed using an object-oriented meta-language.

III Tools For Model Based Slicing

TABLE 1
LIST OF TOOLS



Year	Tool Name	Technique Used
2003	EFSM Slicing Tool	Control and Data flow analysis.
2007	UTG	Data Flow and Control Flow dependency, Communication Tree
2008	SSUAM	Model Dependency Graph.
2008	UML Slicer	MetaModel Diagram.
2009	Reticella	B-Model dependency Graph.
2011	Archlice	Model Dependency Graph.
2011	Safe Slicer	System Model Language, Traceability Links and Rules.
2012	UOST	UML + OCL Constraints.

IV Conclusion

From this literature survey this has been listed out that for model based slicing techniques we need to use dependency relation, control and data flow, uml/ocl constraints, model language are present in literature with great importance on dependency relation. Hence there is need for such technique that can reduce the effort of generation of dependency graph as intermediate state. Slicing UML architectural models is a difficult problem since the model information is distributed across several diagrams with implicit dependencies among them. We had to first construct an intermediate representation called MDG by synthesizing information present in various architectural model elements. Such slices can be used for studying the impact of design changes, reliability prediction, understanding large architectures, etc. We are now trying to enhance our intermediate model by integrating the state and activity models into MDG to compute more accurate slices.

REFERENCES

[1]. Blouin, B. Combemale, B. Baudry, O. Beaudoux, “Kompren Modeling and Generating Model Slicers,” Journal of Software and System Modeling, Springer, 2012.

[2]. Grady Booch, James Rumbaugh, Ivar Jacobson, “The Unified Modeling Language User Guide," 2nd Edition, May 2005, Publisher. Addison Wesley.

[3]. M. Weiser, Program slicing, IEEE Transactions on Soft. Eng., 10, July 1984, pp.352–357.

[4]. Jianjun Zhao, "Slicing Software Architecture," Technical Report 97-SE-117, pp.85-92, Information Processing Society of Japan, Nov 1997.

[5]. Jianjun Zhao, “Applying slicing technique to software architectures,” In Fourth IEEE International Conference on Engineering of Complex Computer Systems, ICECCS’98, pp 87 –98, 1998.

[6]. W. Fangjun and Y. Tong, “Dependence Analysis for UML Class Diagrams,” J. Electronics (China), vol. 21, no. 3, pp. 249-254, May 2004, doi 10.1007/BF02687879.

[7]. M. Harman, A. Mansouri, and Y. Zhang, “Search Based Software Engineering: A Comprehensive Analysis and Review of Trends Techniques and Applications,” Technical Report TR-09-03, Dept.of Computer Science, King’s College London, Apr. 2009.

[8]. H. Kagdi, J.I. Maletic, and A. Sutton, “Context- Free Slicing of UML Class Models,” Proc. 21st IEEE Int’l Conf. Software Maintenance, pp. 635-638, 2005.

[9]. S. Van Langehove, “Internal Broadcasting to Slice UML State Charts: As Rich as Needed,” Proc. Abstracts of the FNRS Contact Day: The Theory and Practice of Software Verification, Oct.2005.

[10]. J.H. Bae, K. Lee, and H.S. Chae, “Modularization of the UML Metamodel Using Model Slicing,” Proc. Fifth Int’l Conf. Information Technology: New Generations, pp. 1253-1254, 2008.

[11]. ANSI/IEEE Standard 1008-1987, “IEEE Standard for Software Unit Testing”, pp.1-23, IEEE Computer Society, 1997.

[12]. Sagar Sen, Naouel Moha, Benoit Baudry, and Jean Marc Jézéquel, "Meta-model Pruning," In 12th International Conference on Model Driven Engineering Languages and Systems (MODELS’09), 2009.

[13]. Jung Ho Bae and Heung Seok Chae, “UMLSlicer: A tool for modularizing the UML metamodel using slicing,” In 8th IEEE International Conference on Computer and Information Technology (CIT), pp.772–777, 2008.

[14]. Jaiprakash T. Lallchandani, R. Mall, "Static Slicing of UML Architectural Models," Journal of Object Technology, vol. 8, no. 1, pp. 159- 188, January-February 2009.

[15]. J. Lallchandani and R. Mall, “A Dynamic Slicing Technique for UML Architectural Models,” IEEE Transaction on Software Engineering, Vol. 37, No. 6, NOV/DEC 2011.

[16]. Philip Samuel, Rajib Mall, “Slicing-Based Test Case Generation from UML Activity Diagrams,” ACM SIGSOFT Software Engineering Notes, Vol. 34 No. 6, November 2009.

[17]. Philip Samuel, Rajib Mall, “A Novel Test Case Design Technique Using Dynamic Slicing of UML Sequence Diagrams,” e-Informatics Software Engineering Journal, Vol. 2, Issue 1, 2008.

[18]. Nisansala Yatapanage, KirstenWinter, and Saad Zafar, “Slicing behavior tree models for verification,” In IFIP Advances in Information and Communication Technology, Vol. 323, pp. 125–139, 2010.

[19].Philip Samuel , Rajib Mall, Pratyush Kanth, “Automatic test case generation from UML communication diagrams,” Information and Software Technology (ELSEVIER), 2007.

[20]. Asadullah Shaikh, Robert Clarisó, Uffe Kock Wiil, and Nasrullah Memon, “Verification-driven slicing of UML/OCL models,” In Proceedings of the IEEE/ACM international conference on Automated software engineering, pp. 185–194, ACM, 2010.

[21].Asadullah Shaikh, Uffe Kock Wiil, and Nasrullah Memon, "Evaluation of tools and slicing techniques for efficient verification of UML/OCL class diagrams," Advances in Software Engineering, vol.18, pp 173-192, 2011.

[22].Monalisa Sarma, Debasish Kundu, Rajib Mall, “Automatic Test Case Generation from UML Sequence Diagrams,” 15th IEEE International Conference on Advanced Computing and Communications, 2007.

[23]. Mathieu Acher, Philippe Collet, Philippe Lahire, and Robert France, “Slicing feature models,” In 26th IEEE/ACM International Conference On Automated Software Engineering (ASE’11), IEEE/ACM, 2011.

[24]. Mathieu Acher, Philippe Collet, Philippe Lahire, and Robert France, “Separation of Concerns in Feature Modeling: Support and Applications,” In Aspect- Oriented Software Development(AOSD’12), ACM Press, 2012.

[25]. Arnaud Hubaux, Patrick Heymans, Pierre-Yves Schobbens, Ebrahim Khalil Abbasi, and Dirk Deridder, “Supporting multiple perspectives in feature-based configuration,” Software and Systems Modeling, 2012.

[26]. T. Kim, Y.-T. Song, L. Chung, and D.T. Huynh, “Dynamic Software Architecture Slicing,” Proc. 23rd Int’l Computer Software and Applications Conf., pp. 61- 66, 1999.

[27].T. Kim, Y.-T. Song, L. Chung, and D.T. Huynh, “Software Architecture Analysis: A Dynamic Slicing Approach,” J. Computer and Information Science, vol. 1, no. 2, pp. 91-103, 2000.

[28]. Davide Falessi, Shiva Nejati, Mehrdad Sabetzadeh, Lionel Briand, and Antonio Messina, “SafeSlice: a model slicing and design safety inspection tool for SysML,” In SIGSOFT/FSE’11 19th ACM SIGSOFT Symposium on the Foundations of Software Engineering (FSE-19) and ESEC’11: 13rd European Software Engineering Conference (ESEC-13), ACM, 2011.

[29]. Zoltán Ujhelyi, Ákos Horváth, and Dániel Varró, “Towards dynamic backward slicing of model transformations,” In 26th IEEE/ACM International Conference on Automated Software Engineering (ASE 2011), pp.404–407, IEEE Computer Society, 2011.

[30]. Zoltán Ujhelyi, Ákos Horváth, and Dániel Varró, “Dynamic Backward Slicing of Model Transformations,” IEEE Fifth International Conference on Software Testing, Verification and Validation, 2012.

[31]. A. Blouin, B. Combemale, B. Baudry, O. Beaudoux, “Modeling model slicers,” Proceedings of the 14th international conference on Model driven engineering languages and systems, 2011.

[32]. A. Blouin, B. Combemale, B. Baudry, O. Beaudoux, “Kompren Modeling and Generating Model Slicers,” Journal of Software and System Modeling, Springer, 2012.