Formal Verification of UML Profile

Arifa Bhutto, D.M. Akbar Hussain

Department of Energy and Technology, Esbjerg Niels Bohrsvej 8 6700 Esbjerg, Dk

Abstract: The Unified Modeling Language (UML) is based on the Model Driven Development (MDD) approach which capturing the system functionality using the platform-independent model (PMI) and appropriate domain-specific languages. In UML base system notations, structural view is model by the class, components and object diagrams and behavioral view model by the activity, use case, state, and sequence diagram. However, UML does not provide the formal syntax, therefore its semantics is not formally definable, so for assure of correctness, we need to incorporate semantic reasoning through verification, specification, refinement, and incorporate into the development process. Our motivation of research is to make an easy structural view and suggest formal technique/method which can be best applied or used for the UML based development system. We investigate the tools and methods, which broadly used for the formal verification of UML based systems.

Key words: Formal Methods, Components, Formal syntax, Domain specific, Semantic.

INTRODUCTION

The increasing complexity of software systems makes essential that performance aspects can be analyzed from the early phases along the entire software lifecycle. Traditionally the system development is divided into analysis, design and implementation.

If faults are discovered lately in the system development process, which lead you to either abandon the system entirely or go through redesign or re-develop the system, this process goes on until the system becomes acceptable, therefore, to avoid such faults, implementation phase is followed by the test phase and similarly analysis and design phase followed by the design verification phase.

As we know commercially development of the software design is based on the UML diagrams (Tsiolakis, 2001). However UML does not provide the assurance of correctness and verify the design models according to the corresponding system desired properties. Therefore, verification of design should be done by the formal methods which provide the suitable verification and validate the system according to the desired properties of the systems (Bondavalli, 2005).

The research focuses on to explore how formal techniques can be or should be used to verify the systems at different level of abstraction. We also investigate the various formal methods which is commonly used for the specification, verification and testing of the UML based design diagrams at different level of system development.

UML Overview:

In the last few years the Unified Modeling Language (UML) diagrams (W. Ahrendt, 2000) has been adopted as a standard design diagrams for software development, and it is positively helpful and successfully used from requirement specification to the implementation process. UML design diagrams are successful due to the following reasons:

- UML modeling is favorably applicable for the different types of domain or subject area.
- UML design diagram captures the structural, operational and behavioral aspect of the system.
- The same notations can be used for specification to the implementation of the system.
- UML is widely open language and most popular due to extensibility mechanisms.
- In UML we can developed or design the different systems domain, methods and process easily and more comfortably.
- UML can be extended or constructs in particular context like (e.g., performance requirement validation, refinements).
- UML has to support for the various tools developed by the different vendors.
- In UML diagram we can capture the static structure, behavioral structure and interaction of the systems components easily through available notations.

Corresponding Author: Arifa Bhutto, Department of Energy and Technology, Esbjerg Niels Bohrsvej 8 6700 Esbjerg, Dk
E-mail: Abh@et.aau.dk, Akh@et.aau.dk
The UML profile mainly has nine types of diagrams which help to design the systems of broad perspective (Gogolla, 2005). The name and purpose of the each diagram discuss in the Table 1.

<table>
<thead>
<tr>
<th>UML Diagram</th>
<th>Use of the diagram in system design</th>
</tr>
</thead>
<tbody>
<tr>
<td>Class Diagrams</td>
<td>Describe the static structure of a system. Classes are an abstraction of entities with common characteristics and associations represent the relationships between classes.</td>
</tr>
<tr>
<td>Object Diagram</td>
<td>Define the static structure of a system at a particular time. Object diagrams are also closely linked to class diagrams. Just as an object is an instance of a class.</td>
</tr>
<tr>
<td>Package Diagram</td>
<td>Represent the elements of a system into related groups. To minimize dependencies between packages.</td>
</tr>
<tr>
<td>Sequence Diagram</td>
<td>Describe interactions among the stockholders in terms of an exchange of messages over time.</td>
</tr>
<tr>
<td>Use Case Diagrams</td>
<td>Design model that represent the usage of the system by different actors and use cases.</td>
</tr>
<tr>
<td>Collaboration Diagrams</td>
<td>Define the interactions between objects as a series of sequenced messages. It is describe the static structure and the dynamic behavior of a system.</td>
</tr>
<tr>
<td>State chart Diagrams</td>
<td>Describe the dynamic behavior of a system in response to external interaction. State charts are especially useful in modeling reactive objects whose states are triggered by specific events.</td>
</tr>
<tr>
<td>Activity Diagrams</td>
<td>It is design the dynamic nature of a system by modeling the flow of control from activity to activity and represents an operation on some class in the system that results in a change in the state of the system. Typically, activity diagrams are used to model workflow or business processes and internal operation of the system.</td>
</tr>
<tr>
<td>Component diagrams</td>
<td>Describe the collection of physical software components, including source code, run-time (binary) code, and executable.</td>
</tr>
<tr>
<td>Deployment diagrams</td>
<td>Define the physical resources in a system, including nodes, components, and connections.</td>
</tr>
</tbody>
</table>

**Formal Verification of UML Profile:**

Unified Modeling Language (UML) provides nine main diagrams which offering powerful abstractions for modeling systems, as we already described in the Table 1. However in UML currently has the lacks of the semantics specification which somehow creates many problems in implementation phase (M. Richters, 2002).

Like how we check the design of models at different views of the system for the different levels of design system.

How we can analysis and verify the UML based design models to check and validate the properties according to the system design and demands.

How we give precise definitions of correctness preserving model transformations, and provide effective rules and tool support for these transformations.

It is therefore, difficult to design the system software according the key software qualities, such as early simulation and validation or test case generation starting from standard UML based models (M. Richters, 2002).

Keep in view of the above questions formal verification and mathematically modeling give us appropriate and satisfied mechanism for the UML based design systems.

Further we investigate the several formal methods/tools such as SPIN which is well-known model checker for verification and translation of the models. rCOS is developed on the Unified Theories of Programming which based on the refinement rules for object-oriented design patterns and verification involves using FDR model checker.

Concurrent Sequential Process CSP is a specification notation for specifying communicating components. The use of Z,OZ, LTL, PVS for the verification of the consistency, model checking, refinement checking, analysis of the design patterns, of the UML based design diagrams.

<table>
<thead>
<tr>
<th>Levels</th>
<th>Languages</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td>Modeling</td>
<td>UML profile</td>
<td>Model M is Use for the Systems' S</td>
</tr>
<tr>
<td>Specification</td>
<td>CSP, OZ, B,DC,SPIN,PROMELA,ROCS,UPPAL,,etc</td>
<td>Specification S of M</td>
</tr>
<tr>
<td>Verification</td>
<td>LTL, Automata Theory, Z language, Duration calculus, PVS</td>
<td>Verification of V of S specification as Operational semantics O</td>
</tr>
</tbody>
</table>

**Comparison and Categorization Criteria:**

We present an overview of the UML diagrams, and use of such diagrams in the system design, however as we mentioned that due to the lack of the formal semantic the UML diagrams must be verified through formal techniques.

Many tools have been developed that support system development using UML. However, when specification and formal verification of UML based design system is an issue, there are much fewer approaches are available.
In this respect we reviewed different modeling methods and techniques which are most commonly used in the field of formal specification, verification and testing of UML based system design in various domains. Our motivation of research to explore the use of formal methods at different level of development process in this respect we describe our review.

### Table 3: Summary of Methods Categorized by modeling level.

<table>
<thead>
<tr>
<th>UML Notations</th>
<th>Formal Methods</th>
<th>Level of Modeling</th>
<th>Line Supported Literature</th>
<th>Tool Support</th>
<th>Affordable</th>
<th>Semantics</th>
<th>Analysis of the Tool</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object, Class, Diagram Components Diagram, Statecharts Diagram</td>
<td>OCL (Object Constraint Language) Specification language, heavily use of the Mathematically specification</td>
<td>Available</td>
<td>Yes supported by the IBM, OMG</td>
<td>The Object Constraint Language (OCL) is the established language for the specification of properties of objects and object structures in UML models, based on Model Driven Development MDD</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Statecharts Diagram, Class Diagrams, Sequence Diagram, Object Diagrams</td>
<td>CSP (Concurrent Sequential Process) Specification notation for specifying communicating components</td>
<td>Available</td>
<td>Research Groups</td>
<td>Describing patterns of interaction in concurrent systems, safety critical system design</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Object Diagram Class Diagram Use Case Diagram</td>
<td>PVS (Prototype Verification Systems) Specification and Verification language for UML object, class diagram and OCL, constrains</td>
<td>Available</td>
<td>Research groups</td>
<td>PVS is the research prototype, based on the theorem proving and model checking</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>B-Methods Diagram Class Diagram Components Diagram Statecharts Diagram</td>
<td>Formal Specification language, Verification of Model based approaches</td>
<td>Available</td>
<td>Research groups</td>
<td>B-Methods is the research developed tool used for analysis tool for property checking and animation of the class diagram. It is one of few formal methods which has robust, commercially available support tools for the entire development life-cycle from specification through to code generation</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Z, OZ, language Diagram Class Diagram Components Diagram Statecharts Diagram</td>
<td>Formal specification language, modeling specification</td>
<td>Available</td>
<td>Research Groups</td>
<td>Z is based on the standard mathematical notation used in axiomatic set theory, lambda calculus, and first-order predicate logic</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Object, Class, Diagram Components Diagram, Use Case Diagram</td>
<td>PROMELA Process or Protocol Meta Language Verification Modeling Language</td>
<td>Available</td>
<td>Research Groups</td>
<td>Verify the Modeling behavior of the models</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Object, Class, Diagram Components Diagram Statecharts Diagram</td>
<td>SPIN (Simple Promela Interpreter) Model Checker, Simulator</td>
<td>Available</td>
<td>Research Groups</td>
<td>In addition to model-checking, SPIN can also operate as a simulator, following one possible execution path through the system and presenting the resulting execution trace to the user. Spin has been used to trace logical design errors in distributed systems design</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Class Diagram Object, Components Diagram</td>
<td>rCOS (refinement calculus for object Systems) Unifying Theories of Programming (UTP)</td>
<td>Available</td>
<td>UNSUB research group</td>
<td>rCOS supports modelling at different views and their relationships of the system and the separation of concerns in the development</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Object-oriented design Architecture, UPFAL Model checker</td>
<td>Model Checker based on the timed Automata theory</td>
<td>Available</td>
<td>Research group from Uppals Sweden and the Aalborg University in Denmark</td>
<td>Object composition (process algebra) Synchronization actions, Time intervals</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

The detail of the review is based on the following criteria:

1. **UML Notations:** We note that UML notations are used to capture the system design in conceptual, specification and implementations of the system.
2. **Formal methods used for the verification:** We investigated the various methods, tools, languages which are used for the UML base design system for verification at different levels.
3. **Level of Modeling:** We list out the formal methods which are used for the different level of the system design like specification, verification, testing, simulator and model checker.
4. **Line supported Literature:** We investigate the possible availability of the literature.
5. Tool Support: We list if the method being described is supported either with a proposed tool, or with preexisting tool.
6. Affordable: We investigate the available tool is affordable in term of usage and commonly available for working.
7. Semantics: The semantics of methods are available or not.
8. The Methodologies: The usage of the methods with respect to investigation of the problem at different level of UML based system design.

Conclusion:
The focus of the study, to give a comprehensive review of UML based design notations and integration of UML notations with the formal verification and specification methods. We investigate the various formal methods and describe the use of the formal methods in different level of system design.

We found that up to know much have been done in the field of verification and validation of UML based notations but no one complete model support the all of the desirable properties In the table 2 we, provides a unique approach to investigate the desired formal methods/tools for the various level of development cycle. In table 3 we categorized and analyze the various formal methods/tools are used for verification and validation of UML notations.

We also investigate that an integration of different formal methods may be required to generate a new complete model which support the model checking at early stage of design system, which is also our future work.

REFERENCES


