A Combine Usability Framework for IMCAT Evaluation

Ziyaulhaq Aliyu
Dept. of Computer and Info. Sci.
University Teknologi PETRONAS,
31750, Tronoh, Perak, Malaysia
ziyaulhaqaliyu@yahoo.com

Lukman Ab. Rahim
Dept. of Computer and Info. Sci.
University Teknologi PETRONAS,
31750, Tronoh, Perak, Malaysia
lukmanrahim@petronas.com.my

Emy Elyanee Mustapha
Dept. of Computer and Info. Sci.
University Teknologi PETRONAS,
31750, Tronoh, Perak, Malaysia
emy.elyanee@petronas.com.my

Abstract—Model transformation (MT) is a key component of model driven software development. It is used to transform source model into a target model, to increase model quality and introduce design pattern and refactoring. As with other software development artifacts, MTs are not free from bugs and thus they must be verified. Various researches have been carried out on model transformation verification approaches, but no one has evaluated their usability. Indirect model checking approach for transformation (IMCAT) is an approach that can verify code generators by model checking the generated code. In this paper a combine usability model to evaluate IMCAT usability related to its learnability, efficiency and reliability will be presented.

Index Terms—Model-driven Engineering, Model Transformation Verification, Model checking, Usability.

I. INTRODUCTION

Model Driven Engineering (MDE) is a process to develop software through which software is designed, specified, executed and deployed using a set of models. This process helps to minimize complexity by using an extensive engineering process. The theory was first put forward in 1987 [1]. Many MDE researchers proposed that almost all aspects of software development will be modeled hence everything is a model [2]. The aim of MDE is to improve the level of abstraction within a program specification as well as maximize automation within program development. The point made by MDE is to use models at various levels of abstraction for system development in order to raise the level of abstraction in program specification. Maximization of automation within a program development is achieved by having executable model transformation. Higher-level models can be transformed towards lower-level models until the model can become executable, either by using model interpretation or code generation [3].

With the increase in software development complexity, the Object Management Group (OMG) announces Model Driven Architecture (MDA) [4], a method of organizing and managing business enterprise architectures, supported by automated tools and facilities both for defining a model and assisting transformation between various types of model [5]. MDA separates the development process into different levels of abstraction, such as platform specific model (PSM) and platform independent model (PIM) [6]. For numerous incompatible modeling languages, a meta-modeling notations and Meta-Object Facility (MOF) had been created as a tool to unify the various modeling languages and established a framework to develop new languages [7]. Model driven development (MDD) is another MDE application in software development life cycle. MDD concentrates on the usage of models and automation of model generation by using model execution, model transformation and code generation tools. MDD approach is advanced by OMG with regard to software development through the initiatives of MDA in addition to its important standards, like UML, QVT and MOF [8, 9]. The term MDA and MDD are often used interchangeable, but according to [8], MDD means the activities that software developers performed while MDA is about building a formal framework for MDD to operate.

Model transformation (MT) is the transforming of one model to another model according to the transformation rules [4]. Various methods may be used for defining the transformation rules, such as QVT and ATL [10] (refer to [11] for more detail). Majority of these languages concentrate on the transformation implementations but usually not their verification [12]. Therefore, the MDE group needs to have techniques and methods that support transformations verification [12].

Verification is a way of checking a service, a system or a product, that meets its specification and designed purpose. Within all development scenarios, verification is used and basically performed by the developers [13]. According to [14], verification should generate an assurance that software is appropriate for a purpose and will establish the quality of confidence.

Indirect model checking approach for transformation (IMCAT) is an approach that verifies code generators by model checking the output of the code generators. A code generator (CG) is a tool that generates code from software models automatically. IMCAT also verify semantic conformance. Semantic conformance means to check the ability of the CG in generating a code according to the source model’s modeling language semantics. The generated code is annotated with assertions that are utilized to check semantic conformance. The assertions are added instantly by another model transformation produced by the user. When model checking the generated code, the model checker encounters and inspects the assertions for all possible execution. If all assertions are correct, the code generator adjusts towards the semantics from the source modeling language. If the assertion is violated, the model
checker creates a counter example to exhibit the way the breach happens. When the model checker is not able to accomplish the verification because of state space explosion, the source model is analyze to provide a suitable state space reduction technique to overcome the state space explosion problem. IMCAT provides four procedural steps in code generators verification. The first two steps are known as the preparation process and they are performed once for a code generator (CG) being verified. Step one known as the learning process in which a user studies the code generated by code generator in order to learn how it works. The next step requires developing a model transformation which is used to generate assertions. The third step happens when the CG is verified and the final step is only carried out if the model checker does not complete the verification because of state space explosion. IMCAT is the first approach that can verify code generators where the person in charge of the transformation does not have the transformation rules and this is essential especially when verifying commercial code generators since the source code is not available by the users. IMCAT use indirect technique to verify CG and using indirect technique enable it to verify code generators regardless of the language use in developing the CG. This study selected IMCAT among other approaches, because IMCAT was developed by one of the researchers at Universiti Teknologi PETRONAS (UTP) and it is a common practice to continue the research further.

Various researches have been conducted in model transformation verification approaches, but no one evaluate the usability of those approaches. Usability is the degree where by software artifact may be used by specified users to attain specified goals with efficiency, effectiveness and satisfaction in a specified context of use [15]. In this paper a combine usability model is developed that evaluates IMCAT’s usability related to its learnability, efficiency and reliability. The usability evaluation of IMCAT approach in this research specifically aims to support and facilitate the systematic development of model transformation verification techniques in software development. The contribution of this research enables the model transformation users to identify key elements needed and usability issues that IMCAT approach need to address.

II. RELATED WORK

A. Usability Attributes

Usability studies have received significant attention in various fields, especially in software engineering; it consists of multiple constructs from various perspectives, such as reliability, efficiency, learnability, memorability, and others that focused largely on interface design [16]. Table 1, Shows the summary of IMCAT usability attributes. This study describes usability attributes related to IMCAT approach according to the literature review. Because most of the researchers consider this three attributes as the fundamental for usability evaluation. Table 1, point out the description and the measurement criteria used by these three usability attributes. Like for learnability we use measurement criteria; the time it takes the user to reach a certain level of performance, for efficiency; the time taken to implement IMCAT approach and for the reliability we use measurement criteria; task of user performance and the number of errors by the users. Majority of usability research focus on either producing principles of system design or improving the design of the existing system.

### TABLE I.  SUMMARY OF IMCAT USABILITY ATTRIBUTES

<table>
<thead>
<tr>
<th>Attributes</th>
<th>Description</th>
<th>Measures</th>
</tr>
</thead>
<tbody>
<tr>
<td>Learnability</td>
<td>The approach is easy to learn, enabling the user to accomplish their tasks quickly. Is the degree whereby a user believes that using IMCAT approach will be free of effort (ease of use).</td>
<td>Time it takes the user to reach a certain level of performance and user’s self-report</td>
</tr>
<tr>
<td>Efficiency</td>
<td>Achievement of a high level of productivity by user using IMCAT</td>
<td>Time taken to implement IMCAT</td>
</tr>
<tr>
<td>Reliability</td>
<td>The completeness and accuracy whereby a user can achieve a specific goals in using IMCAT</td>
<td>Task of user performance and number of errors by the user</td>
</tr>
</tbody>
</table>

Usability attributes have been identified by different researchers from various disciplines. For example four usability aspects; effectiveness, attitude, learnability and usefulness has been suggested by [17], [18], recognizes four criteria of usability evaluation; consumer attitude, effectiveness, versatility, and learnability to find out how customers manage their tasks in using a system. [19], suggested five attributes; learnability, low error rate, efficiency, memorability and satisfaction. International Organization for Standardization (ISO) suggested another usability model based on some main construct, such as efficiency, effectiveness and satisfaction. This three constructs has been established by ISO as an international standard known as ISO9241-11 [20]. Other different usability model share related perspectives by adding some constructs. For example in [21], usability definition includes efficiency, functionally correct, easy to learn and to remember, pleasing and error tolerant. While [22] proposed efficiency, adaptability and helpfulness as attributes of usability. Usability test study by [23] adopted many criteria for usability such as, satisfaction, effectiveness, supportiveness, intuitiveness and usefulness.

B. Verification Approaches

This study considers approaches that can verify code generators (model-to-code transformation) and semantic conformance in the literature review. This criterion is selected because IMCAT, which is the focus of our study, is used to verify code generators for semantic conformance.

Indirect technique is used by [25] to transform source model and target model in to equivalent transition systems. Preserved transformation properties are translated from a source-specific representation to a target-specific representation and then model checked both on the target and source transition system. The technique has limitations towards
behavioral models that have a suitable transition system translation and also does not address model checking scalability problem. A similar approach is used by [26], who have shown a reasonable scalability close to 1200 lines of code when using model checking for verifying preservation of Simulink-to-C code generator properties. The above approaches are similar to IMCAT, as they used model checking techniques in verifying the properties.

Partitioning testing approach is used by [27] and [28]. [27], provides tool that support testing transformation rules written in Tefkat by the use of category partitioning testing technique. Even though [28] follows a similar approach, [28] considers another partitioning testing technique known as the classification-tree method.

The work of [29] and [30] resembles one another but the domains they use differ. In the case of [29], they consider domains geometric state estimation while [30] considers problems of data analysis. Both approaches are domain specific model based code generators and use automated theorem prove to check the properties of the generated code. The basic idea of both approaches is to insert annotations automatically into the generated code. These works have resemblance to IMCAT in automatic insertion of assertions, but IMCAT uses model checking technique instead of theorem proving.

III. IMCAT APPROACH

IMCAT [24] verifies code generators using model checking technique. More specifically it is used to check the code generators’ ability in generating a code that is in line with the source model’s modeling language semantics. The approach also considers semantic conformance verification. IMCAT approach verifies a code generator (CG) by model checking the generated code that provides a higher guarantee that the code has the required properties. The generated code is annotated with assertions that are used to check the preserved semantics of the source modeling language. The assertions are added automatically by another model transformation (MT1) developed by the user. When model checking the generated code, the model checker encounters and checks the assertions for all possible executions. If all assertions are correct, the code generator conforms to the semantics of the source modeling language. If an assertion is violated, the model checker produces a counter example to show how the violation occurs. If the model checker is unable to complete the verification due to state space explosion, the source model is analyzed to identify a suitable state space reduction technique that can be implemented when model checking the generated code. The summary of IMCAT approach is illustrated in Fig. 1.

IMCAT has four procedural steps for code generators verification. Step one and two are call preparation process and are perform for verifying code generators.

- In The first step, the user studies the code generators output in order to recognize how it works. This step is essential when verifying commercial code generators where by the source code is not available. The users need to identify where to insert the assertions required for code generators verification. A questionnaire based analysis is proposed by IMCAT in understanding the code generators, where by the CG is studied in order to find answers for the list of questions that had been prepared earlier.

![Figure 1. Summary of IMCAT approach (adapted from [24])](image)

- In the second step a model transformation called MT1 is developed by the user that can add assertions automatically. Basically, MT1 is made up of transformation rules that can be used to generate those assertions. The user can write the transformation using any appropriate model transformation language and standard techniques are used for writing the transformation rules. IMCAT used testing and mutation for checking the validity of model transformation developed by the user.

- In this step the model transformation is verified. MT1 takes the same input model as the code generator and use it to create the assertions. The generated code is annotated with an assertion and passes it to model checker; finally the assertions are used by a model checker to verify the generated code.

- The last step is perform when the model checking cannot be completed due to the state space explosion problem (state space explosion is a condition where by the model checker need to check many system states). Due to this, the model checker is unable to finish the verification. IMCAT provides a method that uses heuristics to select suitable state space reduction techniques to tackle the state space explosion problem.

IV. METHODOLOGY

The frame work of this research is to test the three hypotheses as shown in Fig. 2. The hypothesis is to ascertain whether;

- Hypothesis 1: Learnability of IMCAT influences its usability.
Hypothesis 2: Efficiency of IMCAT influences its usability.
Hypothesis 3: Reliability of IMCAT influences its usability.

![Usability diagram]

Fig. 2. Research framework

Subjective approach is going to be used for IMCAT learnability evaluation where participants will be brought into a lab to give them training for a fixed period of time. Later the participants will be asked to complete a set of given tasks and finally 30 learnability survey related questions will be answered (ease of learning). The laboratory session will be divided into five sessions. This will enable the participants to know more about IMCAT approach since it is new and also reduce the size of the questionnaire. The first and second lab sessions cover an introduction to model driven engineering and Unified Modeling Language State Machine Semantics (UMLSMS). The idea of these two sessions is to enable the participants to have some background knowledge on the field. The last three sessions cover the IMCAT approach. In the last three sessions, each session will cover 10 learnability questions that can be filled by the participant so that at the end of these sessions, 30 questionnaires will be filled.

The second part of this research is to evaluate the efficiency and reliability of IMCAT. In this part an experimental method is going to be used, Rhapsody and Visual Paradigm will be used as a case study. Fig. 3 shows the complete usability framework that contains the three usability attributes (learnability, efficiency and reliability) in the first row. Whereas the second row shows user activities and behavior factors such as time, speed and error then the final row shows the variables to be measured in IMCAT which includes: easy to learn, easy to implement, speed of performance and rate of errors.

V. CONCLUSION

This study brought some insights into IMCAT usability evaluation that covers two approaches: questionnaire and experiment. This is the first study conducted in usability evaluation related to model transformations verification approaches. The framework used within this study could also be used to measure usability of other model transformation verification approaches. On the other hand, this study has limitation. Various usability attributes have been discussed in the literature review but the framework suggested in this study addresses only three. This limitation shows a need for further studies that cover additional usability attributes for example flexibility, memorability, helpfulness and adaptability.

REFERENCES

[12] Esther Guerra, Juan de Lara, Manuel Wimmer, Gerti Kappel, Angelika Kiesel, Werner Reitschlag, Johannes Schönböck and Wieland Schwinger Automated verification of model transformations based on...


