Journal of Computer Sciences

Original Research Paper

Evaluating the Usability of Model Transformations Testing Approach (MTTA)

Lukman Ab. Rahim, Ziyaulhaq Aliyu and Emy E. Mustapha

Department of Computer and Information Sciences, Universiti Teknologi PETRONAS, 31750, Tronoh, Perak, Malaysia

Article history
Received: 30-12-2015
Revised: 01-03-2016
Accepted: 25-01-2017

Corresponding Author:
Lukman Ab. Rahim
Department of Computer and Information Sciences, Universiti Teknologi PETRONAS, 31750, Tronoh, Perak, Malaysia
Email: lukmanrahim@petronas.com.my

Abstract: Model Transformation (MT) is a key component in Model Driven Development (MDD). Model transformation is used to transform source model into a target model, improve the model quality and also introduce the design pattern and refactoring. Model transformation are not free from bugs similar to other software development artifacts and it needs to be verified. Code Generators (CG) are a type of model transformation that automatically generate code from software models. To verify a CG using Model Transformation Testing Approach (MTTA) effectively, MTTA requires the users to manually generate test model and develop assertions. Since both tasks are performed manually, a usability study is conducted to gauge the effects of the manual tasks towards the usability of MTTA from three perspective: Learnability, effectiveness and efficiency. The aim of this paper is to identify the usability problems of MTTA related to its learnability, efficiency and effectiveness. Usability test technique is used in this study and questionnaire was used to collect a quantitative data from the participants. A pilot test was conducted with two participants and later eight participants were recruited for the real test. The result shows that MTTA is learnable and effective but inefficient. We conclude with a discussion on the reason why MTTA is inefficient.

Keywords: Model-Driven Engineering, Model Transformation Testing, Verification, Usability

Introduction

It is familiar that the integral complex nature of software systems increases the challenges of software development (Lin, 2007). Abstraction and automation are among the prominent techniques for addressing software development complexity (Lin, 2007; Narayanan, 2008). The aim of model Driven Engineering (MDE) is to increase the level of abstraction in program specifications by the use of models at different levels of software development and improve automation in program development using executable model transformations. High level models are transformed into lower level until these models become executable by the use of either model interpretation (model-to-model transformation) or code generation (model-to code transformation) techniques (Frankel, 2003; Kleppe *et al*., 2003; Selic, 2003).

Model Transformations (MTs) are used to synthesize various types of artifacts from models or refine models to capture more system details. This shows that model transformations play a key role in MDE for converting models to other software artifacts. The main objective of MTs are to provide automation in MDE, to reduce the human effort that is associated with modeling and also to reduce the potential errors complexity (Lin, 2007; Narayanan, 2008).

The description of how source language constructs may be transformed into target language constructs are referred to as transformation rules. Various methods may be used for defining the transformation rules, such as Query View Transformation (QVT) and Atlas Transformation Language (Bézivin *et al*., 2003) (refer to (Czarnecki and Helsen, 2006) for more detail). Majority of these languages concentrate on the transformation implementations but usually not their verification (Guerra *et al*., 2013). However, as like other software artifact, models transformations are not free from bugs, therefore they need to be verified (Ab Rahim and Whittle, 2010). The main important principal of verification is to discover the defects in

the design and to assess whether the system is usable or not (Bézivin *et al*., 2006).

With regards to the above mention statement, the MDE group demands approaches that can be used for verifying model transformations (Guerra *et al*., 2013). Different Model Transformation Verification Approaches (MTVAs) are now currently in use for verifying model transformations such as, IMCAT (Ab Rahim and Whittle, 2010), VarroMC (Varró and Pataricza, 2003), StaatsMC (Staats and Heimdahl, 2008), FleureyMT (Fleurey *et al*., 2004), WangMT (Wang *et al*., 2006) and MTTA (Fleurey *et al*., 2007). The developers of those approaches focus on the approaches' features but ignore the usability part. Therefore this paper does not aim to cover all the MTVAs or to develop a new one. Instead, the usability study of one selected MTVAs was conducted (MTTA), which offer some solutions to alleviate the usability of MTTA related to its learnability, efficiency and effectiveness (these usability attributes are adapted from ISO 9241-11 (Abran *et al*., 2003)) and also to suggest improvements in the design flow which can accelerate the model transformation verification process.

The commonly used verification techniques are informal techniques such as testing. Using informal techniques for verification depends highly on the informal design and development activities. The reason why it is call informal is because the tools and approaches use for the verifications are highly depend on human reasoning without using mathematical reasoning (Miiller and Poetzsch-Heffter, 2000).

Informal verifications involves model verification by the use of human mind. This can be performed by an expert(s), which includes, a tester, an independent testing groups, a modeler and a modeling team. Informal techniques are not only used to verify, they can also give suggestions to possible errors that can later become a problem in the designed system. Since informal techniques involves human reasoning, they also give a wide range of coverage by considering different elements of the study simultaneously (Miiller and Poetzsch-Heffter, 2000). MTTA is a method for testing MT.

MTTA is an informal technique for assuring the quality of model transformation. Testing gives partial assurances of model transformation verification correctness, because, it only shows the presents of error not it's absent. By executing a transformation on the source model, transformation testing validates that the expected output matched the real output.

MTTA is faced by some challenges as discussed by numerous studies (Ab Rahim and Whittle, 2010; Fleurey *et al*., 2007). The main challenges of model transformation testing are: (a) developing a test models, (b) developing a test oracle (assertions), (c) it does not entirely verify the model transformation correctness. However, transformation testing has become an area of interest for a number of reasons despite these challenges. The major advantages of MTTA are: (a) Ability to uncover bugs while preserving a low computational complexity (b) easy to perform testing activities (c) it is possible to analyze the transformation in its target environment. To use MTTA for verifying model transformations, verification engineer has to generate test model, generate test code and create oracles (assertions) (Fleurey *et al*., 2004).

Despite usability studies and MTVA's research are abundant, relatively no research work has been done to evaluate the usability of MTTA. According to Nielsen (2003; Kortum and Bangor, 2013) measuring usability has high utility because it quantifies how well verification engineers can interact with an approach. Verification engineers needs intuitive approaches and are not willing to offer large amount of time and effort to understand how to use these approaches (Ammar *et al*., 2015; Bevan, 2009). An approach fail if a specified users cannot get it to work, even if that approach performs its primary/technical function perfectly. The current study argue that usability is one of the most critical factors to the success of MTTA. The mere scale of complexity of some MTVAs makes it likely that the developer(s) may fail to anticipate some difficulties that verification engineers may undergo when using MTTA (Lin, 2007).

The aim of this research is to identify the usability problems of MTTA and provide some suggestions for improvement. Usability testing technique is used in this study to evaluate the usability of MTTA related to its learnability, efficiency and effectiveness. Four hypothesis were developed related to the above three usability attributes and the overall MTTA's usability. Pilot test was conducted with two participants and later eight participants were recruited for the real test. Participants are given training in two sessions and later are asked to complete questionnaire using 5-likert scale.

The usability evaluation of MTTA 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 enable verification engineers to identify key elements needed and usability issues that MTTA need to address so that MTTA can be used easily, efficiently and effectively. The methodology used in this study could also be used to evaluate the usability of different MTVAs.

Related Work

Usability Attributes

Usability evaluation is not a new idea and it has been used to evaluate tools for over 60 years (Johnson *et al*., 2007). Kortum and Bangor (2013) argues for the need of usability measurement:

"The measurement of usability has high utility because it quantifies how well users can interact with a

given product or service. Even if a product performs its primary technical function perfectly and a user cannot get the product to work, then that product has failed".

According to (Nielsen, 1994), usability is the question of either the system or product is better enough to satisfy the entire user's need and requirements and other potential stakeholders, which includes the users' clients and managers. Usability evaluation is most commonly used as a design tool in developing software. It may be used as a supporting tool for making decision when deciding whether to update the system or what to update. Johnson *et al*. (2007) sees usability as an end-cycle-affair-of the-production.

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. Based on the ISO (González *et al*., 2013) usability definition, three usability attributes are used. However, this research will adapt this definition. In other words, usability in this research will be treated as a measure that is defined by effectiveness and efficiency, but satisfaction is substituted for learnability. Each construct is defined as:

Effectiveness refers to the user's ability to finish a tasks using the approach and the quality of the output of that task (Brooke, 1996). This means that effectiveness evaluates approach usability by the quality of the output produced with the approach.

Efficiency refers to the high level of productivity of users in completing the task using the approach (Brooke, 1996). In other words refers to time taken to implement the approach.

Learnability refers to how the approach is easy to learn, enabling the user to accomplish their tasks rapidly. It is regarded as the degree whereby a user believes that the approach is ease of use (Nielsen, 1994).

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 (Booth, 1989). Shackel (1991) recognizes four criteria of usability evaluation; consumer attitude, effectiveness, versatility and learnability to find out how customers manage their tasks in using a system. Nielsen (1994), 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 (Green and Pearson, 2006). Other different usability model share related perspectives by adding some constructs. For example in (Brinck *et al*., 2002), usability definition includes efficiency, functionally correct, easy to learn and to remember, pleasing and error tolerant.

Verification Approaches

Fleurey *et al*. (2007) suggest that a reasonable test model sets have to satisfy these three coverage requirements: Association, attribute and class coverage. The survey by Rahim and Whittle (2013) shows some testing approaches that can be used for generating test models automatically and these approaches used to generate test models with assured coverage. Metamodel coverage is the appropriate notion of coverage for testing model transformation. In Metamodel coverage, each source meta-class would be instantiated once in each test model; furthermore, meta-classes properties (example, meta-attributes) must take numerous representative values. The Metamodel coverage definition was proposed by (Wang *et al*., 2006) based on MOF. The perception is that, since MDA modeling languages are defined in MOF, therefore, metamodel coverage may be described at the MOF level. Generally, metamodel coverage was described based on the structural concepts of MOF core (association, inheritance and feature). If the coverage of both three MOF core structural concepts is achieved, then the metamodel coverage is also achieved. Fleurey *et al*. (2007) made observation similar to this, where Essential Meta Object Facilities (EMOF) language is used.

A well understanding methods for achieving coverage have been introduced, such as equivalence partitioning which is used in model transformations context. As an example, Fleurey *et al*. (2004) used category partition testing technique by decomposing a source metamodel into equivalence classes and from each equivalence class, chooses a representative test models. Category partitioning technique was used to identify equivalence classes manually of all possible meta-attribute values. Then a tool was used to generate a test case for each equivalence and a representative value is selected from the equivalence class as the meta-attribute value.

Test oracle (assertions) is another significant challenge in model transformations testing. Whittle and Gajanovic (2005) checked outputs of the transformations against pre-existing knowledge sources given as constraint, for example, post-conditions on the target language. To check transformations post-condition is self-explanatory because it depend on the capability of the tester. Whittle and Gajanovic (2005) use this approach to generate code from NASA-relevant domain-specific-models context. In this case, the expected structure of the generated code properties was well known. The properties are captured as OCL output invariants and enable the code generator outputs to be checked easily for compliance to an expectable structure. Kolovos *et al*. (2006) take a similar approach, but in this study the authors consider a

dedicated language; Epsilon Comparison Language (ECL) which has been use for specifying constraints between source model and target model.

ECL rule can also be used to check if the source model State instance corresponds to a Java class in the generated code with the same name. Using ECL will allow the constraints specification across two different metamodel. Cariou *et al*. (2004) supports this work and a similar approach is taken by (Lano, 2009), where a set of constraints was proposed to verify model transformations syntactic and semantic correctness.

Materials and Methods

To evaluate the usability of MTTA for this research, usability testing technique is used. According to different researchers (Dix, 2009; Rubin and Chisnell, 2008; Kushniruk and Patel, 2004), usability testing is the most fundamental technique of evaluating the usability of a system or product, because it gives direct information on how people use a system or product and what are the exact problems associated with the product that is being tested. It measures the way a software artefact meets its specified usability goals and how well it relies on quantitative metrics of effectiveness, efficiency and satisfaction (Dix, 2009; Rubin and Chisnell, 2008; Kushniruk and Patel, 2004). It can also be used to rank the usability of different product (this study intends to compare MTTA with another approach).

To obtain a numeric results, a quantitative method is used where participants are given training on how to verify a code generator using MTTA and then asked to complete a questionnaire. The questionnaire is developed according to several hypotheses that have been adopted from previous works (Tsakonas and Papatheodorou, 2008; Lewis, 1995) and modified to fit to this research. The hypotheses are:

- H1: Learnability of MTTA influences its usability
- H2: Efficiency of MTTA influences its usability
- H3: Effectiveness of MTTA influences its usability
- H4: MTTA in overall is influenced by its usability

Initially, usability test materials were presented to two experts working in two different software companies for validation. A pilot test was later conducted with 2 participants and finally 8 participants were recruited for the actual test. Generally, the usability test is performed according to the following procedures. Firstly, participants are given a training for two sessions on how to use MTTA. In each session, participants are asked to perform a given task corresponding to what they learn during training. These tasks are activities required in testing a code generator. The code generator used in the training is IBM Rhapsody. A questionnaire is later given for the participants to complete to rate the users experience (learnability, effectiveness and efficiency) in using MTTA. Readers are referred to (Dix, 2009; Rubin and Chisnell, 2008; Kushniruk and Patel, 2004) for different methods of performing a usability test.

The authors for this study perform the following activities for MTTA's usability test:

- Selecting participants for pilot and actual study
- Preparing test materials
- Performing a pilot study and modifications to questionnaire and designed tasks based on pilot study results
- Conducting actual study and data collection
- Analyze collected data

The following subsections discuss each activity.

Participants

The best way of recruiting participants is by developing their profile (Barnum, 2010). This enable the testers to know who among them deserves to be part of the test. The participants recruited for this research have basic knowledge on software engineering, software testing and UML (experience in testing model transformation is not a requirement and this decision is made because we want to assess the learnability of MTTA). Those participants are considered as novice users (they don't have prior knowledge on testing model transformation). The work of (Gerardo, 2007) shows that novice users identify more usability problem than expert. An expert is a user that usually has previous experience with the tested system while novice user does not have previous experience with the system. In this context, both users (expert and novice) have neither tried the approach.

The 8 selected participants are postgraduate students from Computer and Information Science Department, Universiti Teknologi PETRONAS. There is a great motivation to recruit participants that are researchers within the field of computer. Because, participants that have high motivation in new technology will be excited to participate in a usability test and eager to perform the test with new technology. Moreover, this will give them a chance to follow the test step by step and watch for some application deficiencies during the test. On the contrary, participants that have less motivation may provide a critical judgment and opinion about the application (Lazar *et al*., 2010). Result and discussion section shows the variability of the respondent based on MTTA's pilot study.

Usability test materials were initially presented to two experts working in an industry with 15-20 years working experience for validation. A pilot test was conducted with 2 participants that were excluded from the actual test. Finally, eight participants were recruited for the actual test. Different researchers believe that a

small sample usability test works fine when the goal is to identify the system's usability problems (refer to (Nielsen, 2013; UDOHHS, 2011) for number of participants). The participants recruited for this test are postgraduate students in Computer and Information Sciences (CIS) Department, Universiti Teknologi PETRONAS in different fields. According to (Lazar *et al*., 2010), there is a great motivation to recruit participants that are researchers within the field of computer because they will follow the test step by step and watch for some application deficiencies during the test. Cost, is also another constraint to consider when recruiting participants (Lazar *et al*., 2010). The participants agree to participate in this study for free, but still a certificate of appreciation and refreshment were provided.

To learn about the participant's backgrounds and to confirm that they are all from similar backgrounds, we used a self-assessment and general skill questionnaire, which was collected at the beginning of the training session. For the self-assessment (generic characteristics), we collected information about their current study, level, age and confirmed that they are all postgraduate students. Concerning the general skills questionnaire, participants were ask to fill a pre-test questionnaire as shown in Table 1. The demographic data result shows that all participants share the same characteristics, therefore, this study classify them in the same group (as novice).

Task Design

The designed tasks must be appropriate for the evaluation method, the target users and the system under study. Cost and time limits are other constraints that have to be considered when designing tasks for the usability testing sessions (Nielsen, 1994). Most important parts of MTTA are considered (developing test model, generating test code and creating an assertions) when designing the tasks since according to Nielsen (1994), it is not necessary for the task to cover all parts of the product.

According to Rubin and Chisnell (2008) forming a group of people from the company to design the tasks (meaning that, in designing the tasks there is a need to have someone or group of people that have knowledge on the product that is going to be tested) since they know the critical parts of the product. Therefore one expert on model transformation testing participates in MTTA tasks design.

Clear instructions was provided to the participants on how to follow the sequence of the tasks and how to complete the given tasks (the time given for the tasks related to efficiency was used after conducting pilot test and expert validation). A scenario describes the goal that a participant will attempt to obtain and provide motivation for the tasks. Scenario and tasks have to be related in order to avoid some isolated and irrelevant tasks (Nielsen, 1994). Therefore, the tasks in this study are designed related to each scenarios. The test has been divided into two sessions and each participant was asked to solve five tasks. Since the main purpose of all sessions is to evaluate the overall usability of MTTA, the tasks are separated into three main scenarios.

Scenario 1

For scenario one, the participants will generate test models. Therefore, the task developed under this scenario (task 1A) is to measure to what degree participants understand how to generate test models from the metamodel. A copy of UML state machine metamodel was given to each participants and in the test, participants were asked to develop three test models that can be used to verify the code generator.

Scenario 2

The detail of this scenario is for the participant to identify where to add the assertions into the test code and to create the assertions. Two tasks were developed under this scenario, the test for the first task (task 1B) read "Study the test code and identify where to add the assertions" (a copy of generated code was issued to each participants). The second task (task 2B) reads "write the assertion that can be used to verify the code generator".

Scenario 3

On this scenario, the participants will run the test code using the Eclipse environment and document the results. However, this task contain two tasks. The first task (task 1C) read "using the Eclipse environment, run the test code" (each participant was ask to use Eclipse software for this task) and the second task (task 2C) reads "record the result of the run test code".

To evaluate these tasks (task 1A, task 1B, task 2B, task 1C and task 2C), 5-Likert scale questions were developed to rate participants understanding. Since efficiency is one of this study's goal, a time has been issued on each task. The allocated time was determine during the pilot study where the participants of the pilot study will go through the training and attempt the tasks themselves. We captured the time they take to perform the tasks and we also ask their opinion about the suitable times should be given to the actual study participants in completing each task.

Data Collections

Different methods can be used to collect the usability test data, which includes, interview, observations and

questionnaire. A questionnaire is used to collect the data from the participants' responses in this research. This activity is made because it is difficult to observe how the participants are using the approach for verification. Furthermore, questionnaire give greater consistency (promote reliability) compared to other methods since all the questions are the same (Lewis, 1995). In addition, quantitative data enable the tester to perform statistical analysis on the usability data and quantified data can be used to compare and contrast other research and may be used to measure change (Lewis, 1995). This study intends to compare MTTA with other existing research. However, the tester do carefully observed the tasks performed by the participants (using paper and pen) in order to support the usability test result.

Data Analysis

Firstly, pilot analysis will be conducted to identify the reliability of the measurement items using Excel software. Secondly, the Likert scale responses received from the respondents will be analyze to investigate the mean values for each single item. According to Nielsen (2012), recruiting test users is the best way for measuring usability since they complete their answers using different grading scales (e.g., Likert scale). Likert scale was developed in 1932 (Allen and Seaman, 2007) and named after the researcher. Statements are presented in Likert scale rather than questions and the respondents were requested to rate their agreement level with the statements using five alternatives. Many researchers suggest using Likert-type scale because of its easiness in statistical analysis. Therefore, most psychologists see it as an ordinal scale. The answers will be quantifiable by using the grading scales. Using the result, the user can take the value (mean value) of each attribute and combine these into a single usability factor. The factors investigated are Learnability, Efficiency and Effectiveness. T-test will be used for statistical analysis to test the developed hypothesis. Generally, T-test is used to compare the difference between two means for statistical differences and also could be used when the data was small (<30) and the variance is unknown (Singh, 2006).

Results and Discussion

This section presented the findings and analysis of this study. The analysis was discussed in four subsections: In the first subsection, the reliability analysis was performed. Demographic profile of the participants' analysis was conducted in the second subsection. Factor mean analysis was performed in subsection three and finally in the fourth subsection, statistical analysis was perform to test the developed hypothesis. Eight developed hypotheses have been discussed and were all supported for this study as shown in Table 4.

Reliability Analysis

Pilot test is a small test designed for testing logistics and collect information prior to the actual test, for improving the quality of the test materials (Lancaster *et al*., 2004). The objective of conducting pilot study is to make sure that the research materials can be used properly and that the information is consistent. A suggestion by Baker (Denise *et al*., 2001) related to the reasonable number of participants for conducting pilot test is 10-20% of the actual study sample size. When conducting usability testing, Nielsen (1994) suggest that 1 or 2 participants are sufficient for pilot study.

A reliability test was conducted on all the 30 closed ended Likert scale questions based on the data collected during the pilot study using two participants which are excluded from the actual test. The reliability confidence of Cronbach Alpha for this set of pilot study was 0.787 as shown in Table 2. In theory, a variable that achieves a coefficient value of Cronbach's Alpha of 0.6-0.7 is regarded as achieving high internal consistency and reliability (Landis and Koch, 1977). Thus, due to high coefficient values of Cronbach's Alpha, it can be concluded that the respective respondents were able to understand all questionnaires and they admitted the necessity for asking the questions.

Demographic Analysis

The generic demographic characteristics include age, educational level and gender. Other demographic information include experience on software engineering, UML, MT, MTV and MTVAs. The results and discussion of the demographic characteristics follows.

All participants are postgraduate students (100%) and share the same gender (male, 100%). 87.5% of the participants are in the age of 31-40, only 12.5% is between 41-50 years of age.

For the general skill questionnaire, 75% of the participants Strongly Agree while 25% Agree that they have experience of software engineering. Most of the participants, 62.5% Strongly Agree that they have experience on UML, while 37.5% Agree. However, in terms of the experience with MT, 75% of the participants Strongly Disagree while 25% Disagree. The demographic result of experience of MTV shows that majority of the participants (62.5%) Strongly Disagree while 37.5% Disagree. The last demographic question indicates that 87% of the participants

Strongly Disagree while 12.5% Disagree with the experience of MTVAs (Table 1 shows the summary of this demographic analysis).

Factor Mean Analysis

After conducting the reliability test, this study continue to analyze the actual test data. For the factors investigated in this study, the researcher was able to analyze the mean value for each single items investigated based on the responses received from the respondents participating in this study. The mean value obtained from the real analysis shows Learnability got a mean value of 3.98 as the highest among the three. Efficiency got an average of 2.33 as the lowest and Effectiveness got a mean value of 3.96 as shown in Fig. 1.

Strong and weak items have been identified in factors mean analysis as shown in Table 2 and 3 respectively:

All the strong items of MTTA come from its learnability. These might be because all the participants have basic knowledge on software engineering and they have knowledge of software testing which is relevant to MTTA. According to Nielsen, prior knowledge has influenced on learnability.

Effort and time consuming are among the MTTA's challenges because most of its steps are performed manually. Therefore, this might probably be the reason why all the weak items come from MTTA's efficiency, because the time given to the participants is not enough for them to finish a given tasks. Furthermore, MTTA requires the participants to perform analysis manually. For example, the participants have to analyze the metamodel to create test models and the participants have to analyze the generated code to identify where to add the assertions. Another explanation for the low efficiency score is the number of test models that the participants have to generate. If we consider that the goal of testing is to identify as much faults as possible then having many test models that covers many aspects of the metamodel would be beneficial. Using equivalence partitioning or boundary value analysis will definitely help in creating many relevant test models but it will also mean more oracles to prepare and test code to create and execute.

From the MTTA's weak item only one item is from learnability, which is "I easily learn how to generate test model from the metamodel". This result is again may be due to the difficulty in the participants in learning how to analyze the metamodel. Furthermore, the concept of metamodel and the complexity of the metamodel itself may contribute in the problem of learning how to create test models. The survey performed in (Rahim and Whittle, 2013) shows that current research in creating test models are performing this task based on metamodel. Apart from metamodel, other sources used to create metamodel are the model transformation rules and the contract/specification of the model transformations. Both these sources are also complex and difficult to analyze, thus difficult to learn.

Effectiveness is neither the strong point nor the weak point of MTTA. This shows that the participants still consider testing as a viable technique to be used in testing model transformation. But, its effectiveness is questionable as it depends on how many faults it can detect. The capability of MTTA to detect faults depends on the correctness of the test models and the oracles and the coverage of the test models.

Table 1. Respondent's demographic summary

	Likert Scale						
Relevant characteristics	SD	D	N	A	SA	Freq	%
Experience of SE				25%	75%	8	100
Experience of UML				37.5%	62.5%	8	100
Experience of ST	75%	25%				8	100
Experience of MT	62.5%	37.5%				8	100
Experience of MTT	87.6%	12.5%				8	100

Noted that: SD = Strongly Disagree, D = strongly agree, N = Neutral, SA = Strongly Agree and A = Agree. However, SE = Software Engineering, UML = Unified Modeling Language, ST = Software Testing, MT = Model Transformation and MTT = Model Transformation Testing

Table 2. Strong items

Factors	Items	Items mean
Learn	I easily learn how to generate test code from the test model	4.5
Learn	I easily learn how to write down the result of each test code	4.5
Learn	I easily remember how to use MTTA	4.5
Learn	I learned to use MTTA quickly	4.5
Learn	I easily learn how to insert assertions in to the test code	3.4
Learn	I easily learn how to import test code in to the eclipse	3.4

Table 3. Weak items

Factors	Items	Items mean
Learn	I easily learn how to generate test model from the metamodel	2.6
Effic	Less time taken to insert assertions in to each test code	2.6
Effic	It is easy to use MTTA for model transformation verifications	2.6
Effic	Less time taken to run each test code with an assertion into it	2.1
Effic	Less time taken to write down the result of each test code	2.1
Effic	Not much effort needed in using MTTA	2.1

Note: Effic = efficiency and Learn = learnability

Table 4. Statistical decision to accept or reject Ho

Group	T-Val>T-Cri	Decision	Positive/negative influence
Learnability	9.59 > 1.99	Supported	Positive
Efficiency	6.94 > 1.99	Supported	Positive
Effectiveness	6.62 > 2.01	Supported	Positive
MTTA in Overall	3.25 > 1.97	Supported	Positive

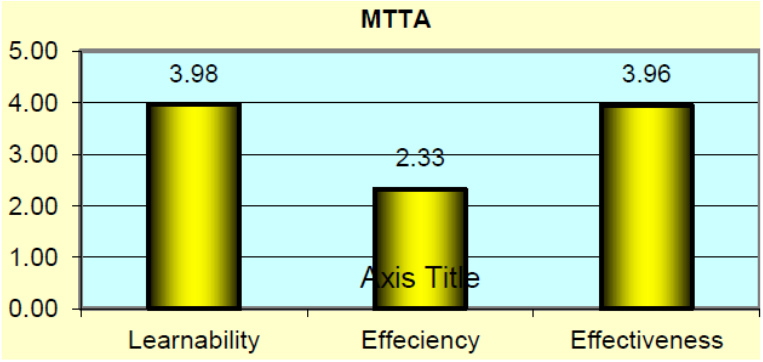

Fig. 1. MTTA Factors mean analysis

Hypothesis Testing

For this study, 5% level of significance is used since it is often accepted as a standard for rejection and two tail test has been used because based on our develop hypotheses, the influence can be positive or negative. From the results shown in Table 2 and 3, the four null hypotheses (Ho) are rejected, which enable the researcher to accept the alternate hypotheses (Ha). The four developed hypotheses were all supported as shown in Table 4.

Conclusion

This study brought some insights into MTTA usability evaluation. Usability testing techniques is used for this study and questionnaire was used to collect a quantitative data from the participants that perform the actual test. Pilot study was conducted earlier with two participants and later eight participants were involved in the actual test. From the result of MTTA's usability test, participants agree that MTTA is learnable and effective but inefficient. Participants did not probably find MTTA difficult to learn because they have basic knowledge on software engineering and also might have knowledge on software testing which is relevant to MTTA. This finding highlights the influence of prior knowledge on usability.

Effort and time consuming are among the MTTA's challenges, this might probably be the reason why participants consider MTTA inefficient. Furthermore, MTTA requires the participants to perform analysis manually. For example, the participants have to analyze the metamodel, to create test models and the participants have to analyze the generated code to identify where to add the assertions. Another explanation for the low efficiency score is the number of test models that the participants have to generate. If we consider that the goal of testing is to identify as much faults as possible then having many test models that covers many aspects of the metamodel would be beneficial. Automating the test model generation will be a good idea to improve the efficiency of MTTA.

The general contribution of this research is to support and facilitate the systematic development of MTTA in software development. The study also highlighted the important of usability, therefore the developers of model transformation verification approaches may consider usability as an important

factor when developing those approaches. The recommendations given could also be used by the developers to improve MTTA's usability so that verification engineers could use it easily, efficiently and effectively. The methodology used within this study could also be used to measure usability of other model transformation verification approaches.

Various usability attributes have been discussed in the literature review but this study addresses only three. There is a need in further study to consider additional usability attributes for example flexibility, memorability, helpfulness and adaptability. Different studies believe that more participants detect more usability problems. Therefore future research will focus on recruiting more number of participants for usability test.

Acknowledgment

The authors would like to acknowledge the contributions of pilot study and usability test participants for their effort and time.

Author's Contributions

Lukman Ab. Rahim is a researcher specializing in Model Driven Engineering and contributes in the study of model transformation testing methods. Emy E. Mustapha specializes in usability testing and contributes in the study preparation for the usability test. Ziyaulhaq Aliyu conducted the usability test under the supervision of Lukman Ab. Rahim and Emy E. Mustapha.

Ethics

Participation during pilot and usability tests are voluntary and participants are made known that their feedbacks will be contributing to a research project.

References

Ab Rahim, L. and J. Whittle, 2010. Verifying semantic conformance of state machine-to-java code generators. Proceedings of the 13th International Conference on Model Driven Engineering Languages and Systems, Oct. 3-8, Springer, Norway, pp: 166-180. DOI: 10.1007/978-3-642-16145-2_12

Abran, A., A. Khelifi, W. Suryn and A. Seffah, 2003. Usability meanings and interpretations in ISO standards. Software Quality J., 11: 325-338. DOI: 10.1023/A:1025869312943

Allen, I.E. and C.A. Seaman, 2007. Likert scales and data analyses. Quality Progress, 40: 64-65.

Ammar, L.B., A. Trabelsi and A. Mahfoudhi, 2015. A model-driven approach for usability engineering of interactive systems. Software Quality J. DOI: 10.1007/s11219-014-9266-y

Barnum, C.M., 2010. Usability Testing Essentials: Ready, Set...Test! 1st Edn., Elsevier, Boston, ISBN-10: 0123785537, pp: 408.

Bevan, N., 2009. Usability. Proceedings of the Encyclopedia of Database Systems, (EDS' 09), Springer, pp: 3247-3251.

Bézivin, J., F. Büttner, M. Gogolla, F. Jouault and I. Kurtev *et al*., 2006. Model transformations? Transformation models!. Proceedings of the 9th International Conference on Model Driven Engineering Languages and Systems, Oct. 1-6, Springer, Italy, pp: 440-453. DOI: 10.1007/11880240_31

Bézivin, J., N. Farcet, J.M. Jézéquel, B. Langlois and D. Pollet, 2003. Reflective model driven engineering. 6th International Conference on The Unified Modeling Language. Modeling Languages and Applications, Oct. 20-24, Springer, San Francisco, pp: 175-189. DOI: 10.1007/978-3-540-45221-8_17

Booth, P.A., 1989. An Introduction to Human-Computer Interaction. 1st Edn., Erlbaum, Hove, ISBN-10: 0863771238, pp: 268.

Brinck, T., D. Gergle and S.D. Wood, 2002. Designing Web Sites that Work: Usability for the Web. 1st Edn., Morgan Kaufmann, San Francisco, ISBN-10: 1558606580, pp: 481.

Brooke, J., 1996. SUS-A Quick and Dirty Usability Scale. In: Usability Evaluation in Industry, Jordan, P.W., B. Thomas, I.L. McClelland and B. Weerdmeester (Eds.), CRC Press, ISBN-10: 0748404600, pp: 189-194.

Cariou, E., R. Marvie, L. Seinturier and L. Duchien, 2004. OCL for the specification of model transformation contracts. Conference Workshop on OCL and Model Driven Engineering, (MDE' 04), pp: 69-83.

Czarnecki, K. and S. Helsen, 2006. Feature-based survey of model transformation approaches. IBM Syst. J., 45: 621-645. DOI: 10.1147/sj.453.0621

Denise, F., C.T. Beck and B.P. Hungler, 2001. Essentials of Nursing Research: Methods, Appraisal and Utilization. 1st Edn., Lippincott.

Dix, A., 2009. Human-computer interaction. Springer.

Fleurey, F., B. Baudry, P.A. Muller and Y. Le Traon, 2007. Towards dependable model transformations: Qualifying input test data. J. Software Syst. Model., 8: 185-203.

Fleurey, F., J. Steel and B. Baudry, 2004. Validation in model-driven engineering: Testing model transformations. Proceedings of the 1st International Workshop on Model, Design and Validation, Nov. 02-02, IEEE Xplore Press, pp: 29-40. DOI: 10.1109/MODEVA.2004.1425846

Frankel, D.S., 2003. Model Driven Architecture: Applying MDA to Enterprise Computing. 1st Edn., John Wiley and Sons, Indianapolis, ISBN-10: 0471462276, pp: 352.

Gerardo, J.L.S., 2007. The effectiveness of novice users in usability testing. University of Oslo Library.

González, M., L. Moreno and P. Martínez, 2013. Approach design of an accessible media player. Univ. Access Inform. Society, 14: 1-11. DOI: 10.1007/s10209-013-0342-z

Green, D. and J.M. Pearson, 2006. Development of a web site usability instrument based on ISO 9241-11. J. Comput. Inform. Syst., 47: 66-72.

Guerra, E., J. de Lara, M. Wimmer, G. Kappel and A. Kusel *et al*., 2013. Automated verification of model transformations based on visual contracts. Automated Software Eng., 20: 5-46. DOI: 10.1007/s10515-012-0102-y

Johnson, R.R., M.J. Salvo and M.W. Zoetewey, 2007. User-centered technology in participatory culture: Two decades "Beyond a narrow conception of usability testing". IEEE Trans. Profess. Commun., 50: 320-332. DOI: 10.1109/TPC.2007.908730

Kleppe, A.G., J.B. Warmer and W. Bast, 2003. MDA explained: The Model Driven Architecture: Practice and Promise. 1st Edn., Addison-Wesley, Boston, ISBN-10: 032119442X, pp: 170.

Kolovos, D.S., R.F. Paige and F.A. Polack, 2006. Model comparison: A foundation for model composition and model transformation testing. Proceedings of the International Workshop on Global Integrated Model Management, May 20-28, ACM, Shanghai, China, pp: 13-20. DOI: 10.1145/1138304.1138308

Kortum, P.T. and A. Bangor, 2013. Usability ratings for everyday products measured with the System Usability Scale. Int. J. Human-Comput. Interact., 29: 67-76. DOI: 10.1080/10447318.2012.681221

Kushniruk, A.W. and V.L. Patel, 2004. Cognitive and usability engineering methods for the evaluation of clinical information systems. J. Biomed. Informat., 37: 56-76. DOI: 10.1016/j.jbi.2004.01.003

Lancaster, G.A., S. Dodd and P.R. Williamson, 2004. Design and analysis of pilot studies: Recommendations for good practice. J. Evaluat. Clin. Pract., 10: 307-312. DOI: 10.1111/j..2002.384.doc.x

Landis, J.R. and G.G. Koch, 1977. The measurement of observer agreement for categorical data. Biometrics, 33: 159-174. PMID: 843571

Lano, K., 2009. Model Transformation Specification and Verification. In: UML 2 Semantics and Applications, Lano, K. (Ed.), John Wiley and Sons, Hoboken, ISBN-10: 0470522615, pp: 349-395.

Lazar, J., J.H. Feng and H. Hochheiser, 2010. Research Methods in Human-Computer Interaction. 1st Edn., John Wiley and Sons, Chichester, ISBN-10: 0470723378, pp: 446.

Lewis, J.R., 1995. IBM computer usability satisfaction questionnaires: Psychometric evaluation and instructions for use. Int. J. Human-Comput. Interact., 7: 57-78. DOI: 10.1080/10447319509526110

Lin, Y., 2007. A model transformation approach to automated model evolution. The University of Alabama at Birmingham.

Miiller, P. and A. Poetzsch-Heffter, 2000. Modular specification and verification techniques for object-oriented software components. Foundations of Component-Based Systems.

Narayanan, A., 2008. Verification of model transformations. Vanderbilt University.

Nielsen, J., 1994. Usability inspection methods. Proceedings of the Conference Companion on Human Factors in Computing Systems, Apr. 24-28, ACM., Boston, pp: 413-414. DOI: 10.1145/259963.260531

Nielsen, J., 2003. Usability 101: Introduction to usability. Nielsen Norman Group.

Nielsen, J., 2012. How many test users in a usability study. Jakob Nielsen's Alertbox.

Nielsen, J., 2013. How many test users in a usability study. Nielsen Norman Group.

Rahim, L.A. and J. Whittle, 2013. A survey of approaches for verifying model transformations. Software Syst. Model., 14: 1003-1028. DOI: 10.1007/s10270-013-0358-0

Rubin, J. and D. Chisnell, 2008. Handbook of Usability Testing: How to Plan, Design and Conduct Effective Tests. 2nd Edn., John Wiley and Sons, Hoboken, ISBN-10: 0470386088, pp: 384.

Selic, B., 2003. The pragmatics of model-driven development. IEEE Software, 20: 19-25. DOI: 10.1109/MS.2003.1231146

Shackel, B., 1991. Usability-Context, Framework, Definition, Design and Evaluation. In: Human factors for Informatics Usability, Shackel, B. and S.J. Richardson (Eds.), Cambridge University Press, New York, ISBN-10: 0521365708, pp: 21-37.

Singh, Y.K., 2006. Fundamental of Research Methodology and Statistics. 1st Edn., New Age International, New Delhi, ISBN-10: 8122418864, pp: 322.

Staats, M. and M.P. Heimdahl, 2008. Partial translation verification for untrusted code-generators. Proceedings of the 10th International Conference on Formal Methods and Software Engineering, Oct. 27-31, Springer, Japan, pp: 226-237. DOI: 10.1007/978-3-540-88194-0_15

Tsakonas, G. and C. Papatheodorou, 2008. Exploring usefulness and usability in the evaluation of open access digital libraries. Inform. Process. Manage., 44: 1234-1250. DOI: 10.1016/j.ipm.2007.07.008

UDOHHS, 2011. Usability. Gov. Step-by-step usability guide, U. D. o. Health and H. Services.

Varró, D. and A. Pataricza, 2003. Automated formal verification of model transformations. Proceedings of hte Workshop on Critical Systems Development in UML, (UML' 03), San Francisco, USA, pp: 63-78.

Wang, J., S.K. Kim and D. Carrington, 2006. Verifying metamodel coverage of model transformations. Proceedings of the Australian Software Engineering Conference, Apr. 18-21, IEEE Xplore Press, pp: 10-10. DOI: 10.1109/ASWEC.2006.55

Whittle, J. and B. Gajanovic, 2005. Model transformations should be more than just model generators.