08-Apr-2010 Dear Mr. Sen: Manuscript ID SOSYM-09-00000624 entitled "Automatic Model Synthesis to Test Transformations" which you submitted to the Journal of Software and Systems Modeling, has been reviewed. The comments of the reviewer(s) are included at the bottom of this letter. In view of the criticisms of the reviewers, a major revision of your manuscript is required for it to be reconsidered for publication in Software and Systems Modeling. This major revision must incorporate changes that address the reviewers' concerns. When preparing your revised manuscript, please carefully consider the reviewers' input, and create a detailed list of changes to the manuscript and responses to their comments and concerns. Your list of responses should be submitted and uploaded as a separate file in addition to your revised manuscript. Please note that resubmitting your manuscript does not guarantee eventual acceptance, and that your resubmission will be subject to re-review before a decision is rendered. Once you have revised your manuscript, go to [https://mc.manuscriptcentral.com/sosym](https://mc.manuscriptcentral.com/sosym" \t "_blank) and login to your Author Center. Click on "Manuscripts with Decisions," and then click on "Create a Resubmission" located next to the manuscript number. Then, follow the steps for resubmitting your manuscript (please see also "How to proceed" instructions below). Please submit both the revised manuscript and the file describing the detailed changes and responses to the reviewers' comments and concerns. Because we are trying to facilitate timely publication of manuscripts submitted to Software and Systems Modeling, your revised manuscript should be uploaded as soon as possible, and certainly before 06-Jul-2010. If it is not possible for you to submit your revision within this time, please notify me by e-mail and we will try to accommodate your constraints. However, if we cannot extend the resubmission date, the paper will have to be treated as a completely new submission. I look forward to your resubmission. Sincerely, Prof. Bernhard Rumpe Editor in Chief, Software and Systems Modeling [Bernhard.Rumpe@sosym.org](https://mail.irisa.fr/squirrelmail/src/compose.php?send_to=Bernhard.Rumpe%40sosym.org) Login information: User ID: (Person not available) E-Mail: (Person not available) Password: (Person not available) How to proceed: 1) Login to [https://mc.manuscriptcentral.com/sosym](https://mc.manuscriptcentral.com/sosym" \t "_blank) 2) Go to your "Author Center" 3) Click on "Manuscripts with Decisions" 4) Select the "create a resubmission" link of the corresponding manuscript. Note: Directly below this link you'll find the number of remaining days for the resubmission. 5) Follow the instructions on the screen. Note: Page 6 of the resubmission process shows a table "My Files" with all files of the original submission. If you have updated a file for your resubmission, please delete the original version and upload the revised file. Editor' Comments to Author: Editor Comments to the Author: Thank you for your submission. We have now received three reviews. The reviewers are all broadly in agreement that the paper presents an interesting idea. However, there are several suggestions for improvement made by the reviewers that need to be addressed thoroughly before the paper can be accepted for publication. In general, the reviewers comment on: - several reviewers commented on the requirement that several manual steps are still needed, such as manual conversion to Alloy. The paper should discuss any potential to obviate the negative impact of such manual conversions and provide some discussion on the issue. Such manual conversion seems to require that the model engineer needs to understand the formalism of Alloy. - more than one reviewer also commented on the structure of the paper and how perhaps there is too much of a focus on background material, and that some of the foundation material and the contribution of this paper are mixed too much. - two reviewers also posed questions about how pre/post conditions could be alternatively expressed and whether an incorrect target model really requires an increase in the pre-condition (or just an indication that the transformation itself is incorrect). Numerous minor improvements were also suggested by the reviewers. Please make any necessary revisions to your paper based on the attached reviews. We look forward to receiving your revision soon. Best wishes, Richard and Jeff (Theme Issue Editors) Reviewer(s)' Comments to Author: Please note: Reviewers might have uploaded their review as a separate files which unfortunately are not attached to this message. To get access to these files please go to your author center in Manuscript Central and click on "Manuscripts with Decisions". Find the entry with Manuscript ID SOSYM-09-00000624 and click on the link "view decision letter" next to it. At the end of the appearing pop up window you'll find links for every attached file if exist. Please make sure that your browser does not suppress pop up windows. Reviewer: 1 Public Comments (these will be made available to the author) The paper presents an automatic approach to the generation of models for testing model-to-model transformations. The approach receives the meta-model of the source language and a set of pre-conditions of the transformation as input. As output, it generates input models according to a selected search strategy. The approach is presented through the classical class2rdbms transformation. Points in favour ---------------- + interesting, useful + model transformation testing is a hot topic + title, abstract and introduction are appropriate + the paper is well organized and easy to follow (except section 5) Points against -------------- - the extension is not really big, I would consider it just in the limit to be acceptable. Moreover, two of the contributions are debatable and not technically sound. In detail: -> Contribution 1 (integration of meta-model pruning) makes use of techniques that were presented in [10] by the authors. -> Contribution 2 (demonstration that the method consistently generates good test models) applies the method just to the running example, therefore it only demonstrates that the method works well for THIS example, but not in general. -> Contribution 3 (usefulness of the approach) is based on a wrong premise: that obtaining an incorrect target model as a result of a transformation implies that the developer has to increase the transformation pre-condition, that is to say, he has to reduce the set of valid input models. On the contrary, it is more likely that this indicates that the transformation is wrong. For instance, a proper implementation of the class2rdbms should be able to transform models where children classes override parent's attributes, instead of forbidding the transformation of such models. - the related work section is incomplete. Comments -------- - In the introduction, when presenting contribution 2, it is said that "these pre-conditions were not initially envisaged by the panel of world experts...". I do not think there are missing pre-condition constraints, but the authors are using an incorrect implementation of the transformation. - Fig. 2(a) does not show the Ecore meta-model, it is more similar to the OMG's UML meta-model. For example, the Ecore meta-model does not define "associations" but "EReferences", and there is no role "general" for inheritance but "eSupertypes". - The first invariant in Fig. 2(b) uses property "allGenerals" (I suppose it should be "general" according to the meta-model). - In the last sentence of section 2 it is said "...about 10% of the time...". Which time? - Page 7, column 2: "CTwo.element and C3.elem are both of type String" -> Int? - Page 8, column 2: "the algorithm terminates for a finite meta-model because the rules do not remove" -> do not add? - Listings are difficult to read, please use a bigger font size. - In Listing 1, I'm not sure whether the transitive closure should be done for role "general" (instead of "parent"). - Does the MMCC allow defining partitions manually? In the example, the partitions consider that the classifier name can be equal or different from the empty string. However, one may have a transformation where classifiers are transformed differently depending on whether their name start by "\_" or not, and in that case, the partitions should consider that fact. - In Listing 4, the model fragment should be named "mfAllRanges7a" to be consistent with table 1. On the other hand, I would put a more realistic example because the shown predicate is not generated, is it? - In the last paragraph of section 3.5, it should be indicated the number of objects for which the model fragments are generated. - Fig. 5 does not show the meta-model pruning step. - The model in Fig. 6 includes an attribute, but the meta-model in Fig. 2 defines properties and not attributes. - Section 4.1 illustrates how incorrect target models can help use to improve the transformation pre-condition. However, it is not considered that obtaining an incorrect target model can also be due to an incorrect implementation of the transformation. The method is not able to distinguish between these two reasons of failure. In the presented example, it is clear that the problem is not the transformation pre-condition but the transformation itself, as it should have detected the existence of a previous column and not creating it twice. Thus, (at least) facts 5, 7 and 8 in appendix C are not fair pre-conditions. - Section 5 is difficult to follow, it has many typos. - Results in table 4 come from a particular transformation, since mutation analysis depends on the transformation where it is applied. Different transformation implementations can give different analysis results. Therefore, this analysis should be applied to different kinds of transformations before stating that the framework works well in general. - In table 3, ClassModel should be Model, Attribute should be Property, and I do not know what a Bit-width Integer is. Data in the last column of the two last rows is missing. Also, I wonder why inheritance is not included in this table, since it is one of the main features that guides the transformation of uml models into rdbms models. If the generated test models do not contain inheritance then it is not a good test model set for this particular transformation. This is more or less discussed at the end of section 5, where it is said that most unguided models do not contain inheritance as it is not imposed by the meta-model. This makes me think that perhaps the unguided strategy could be improved, and that the results obtained in the experiments may be influenced by this "flaw". - There are inconsistencies between table 4 and its explanatory text. For instance, it is said that there is an increase in mutation score for unguided and model fragments based strategies. However, in the first row of the table we obtain the highest mutation score for the biggest but also for the lowest number of classes and operations (70.1). The 80.41% mutation score corresponds to set 6, which is not the one with less classes and associations. Finally, the mutation scores 91.24 and 92.27 correspond to sets 2, 5 and 7. - The related work is incomplete, it mainly contains autoreferences. Other approaches to the analysis and verification of model transformations should be discussed. For example: \* Ethan Jackson, Janos Sztipanovits. "Formalizing the Structural Semantics of Domain-Specific Modeling Languages". Journal of Software and Systems Modeling, 2008. \* László Lengyel, Tihamér Levendovszky, Hassan Charaf: "Constraint Validation in Model Compilers”, in Journal of Object Technology, vol. 5, no. 4, pp. 107-127 \* "Verification and validation of declarative model-to-model transformations through invariants". Journal of Systems and Software, Volume 83, Issue 2. Pages: 283-302. 2010. \* "A Testing Framework for Model Transformations". Yuehua Lin, Jing Zhang and Jeff Gray. Model-Driven Software Development, 2005. Pages 219-236. \* ... - Remove from appendix C those facts that are listed in appendix B. In appendix B, include a comment explaining each fact (as done in the other appendix). Remove headers "initial model..." and "discovered model..." from listings in appendix B and C. - The presented framework allows one to automatically detect target models which are not correct instances of the target meta-model. Apart from that, it doesn't seem to be able to detect whether a target model is a valid transformation of the source one, which restricts its practical applicability for model-to-model transformation testing. However, I feel it should be easy to consider transformation invariants that could be automatically checked on the generated target models (e.g. persistent classes are always transformed in tables). Are you working in this direction? - Also, I would like to see the method returning exceptions of the transformation itself, instead of exceptions of its pre-condition (which is non-executable). Typos ----- - page 2, column 2: can generate model -> can generate models - page 3, column 1: generation fail to -> generation fails to - page 3, column 2: relevant the -> relevant to the - page 4, column 1: MM\_I (For -> MM\_I (for - page 5, column 1: of an Property -> of a Property (several times in the paper) - page 5, column 2: this not really limitation -> this is not a limitation - page 6, column 1: of a 15 class -> of 15 class - page 8, column 1: and and -> and - page 9, column 1: and a its -> and its - page 10, column 1: these trivial solution -> these trivial solutions - page 14, column 2: that are raise -> that raise - page 15, column 1: can navigation -> can navigate - page 15, column 2: of a generated ( -> of a generated input model ( - page 16, column 1: Respecting the -> According to - page 16, column 1: transformations produces -> transformation produces - page 17, column 1: of number -> of the number - page 17, column 1: The large of number -> A large number of - page 17, column 1: are the use different -> are those specific to different - page 17, column 2: models satisfies -> models satisfy - page 17, column 2: Total number -> The total number - page 19, column 1: we shown 4 -> we show 4 - page 20, column 1: not be been -> not been Reviewer: 2 Public Comments (these will be made available to the author) The paper presents an approach (and a tool) to generate test models for any given model transformation. The effective input domain of the transformation (the input domain that the transformation really uses) is obtained via metamodel pruning and then transformed into Alloy to be able to generate the test models and to perform mutation analysis. The methodology is described by presenting a common transformation example. The main disadvantage of the approach is that most of the constraints related to the automatic generation of test models (OCL constraints, precondition and postcondition of a transformation) must be manually encoded into Alloy, and therefore users need to know the Alloy formalism and the corresponding model and metamodel representation that the authors propose. In this sense, Fig. 1 shows a model transformation scheme where preconditions and postconditions of a model transformation are presented. Are they really usually considered/defined in MDE? How are they usually described in MDE environments? In Page 6 is said that you use Ecore and OCL to describe the metamodel and its constraints, but nothing is said about how the preconditions are specified. In fact, you encode them directly in Alloy. Could be them defined for instance with the Kermeta language? Can you provide me an example of a postcondition? In the example I can only find constraints over the output metamodel. 1. Are the title, abstract, and keywords appropriate? They are appropriate, although I would remove the analysis results from the abstract. 2. Does the introduction state the objectives of the submission in terms that encourage the reader to read on? Yes. 3. How relevant is this submission to the readers of this journal? The target audience of the journal are practitioners and researchers from industry and academia with a vested interest in high quality modeling practices and research. Indicate the extent that the paper will be relevant to this target audience. This paper addresses topics which fall within the scope of the journal, and their treatment is rigorous, sound and practical. 4. How does this submission advance the field of software and system modeling research and practice? Comment on any novel contributions or significant insights gained. The journal aims to publish papers that deepen understanding of modeling practices and techniques or contribute significant new ideas that revolutionize or incrementally advance the field. It provides a way to generate test models for any given model transformation 5. Is the submission technically sound? For example, comment on (1) adherence to standards if standard notations/techniques/methods are used, (2) soundness of mathematical expressions, and (3) soundness of conclusions drawn from objective premises. Yes. It seems so. However, it is not clear whether the proposal has been validated with other examples different from the one presented in the paper. 6. Does the submission contain sufficient and appropriate references? Indicate important missing references, if any. Also indicate if references are excessive. Yes. 7. Comment on the organization of the submission. Is it focused? Is the length appropriate for the topic? The length is appropriate. However, the paper is not well-structured: The proposed methodology is presented in Section 4, although some of its parts are previously introduced as “foundations” in Section 3, e.g., the way they qualify models, or the criteria they use to combine partitions in their tool. I think that foundations and the methodology itself are really mixed. I would recommend the authors to introduce the methodology first, then present the example and explain the methodology using the example, and finally show the results of the experiments/analysis. 8. Please comment on the readability of this submission. Please comment on the degree of effort required to read and understand this paper. The paper is readable in general, although it would significantly benefit from a reorganization according to the suggestions given above. Furthermore, the model typing section should be improved in some aspects: - It is said that the notion of model type conformance is based on the work presented in [19]. However, model subtyping is based on the work of [18]. Are [18] and [19] model type notions compatible? - It is not clear if the matching relation presented in this section covers or not all metamodel elements and properties. The algorithm and the two relaxation made should be better explained. - The provided example (Figs. 3 and 4) could be easily understood with a “real” example (by using two metamodels with concrete names). - It is said that “all operations written for MMe are valid for the large meta-model MM” which comes from the model subtyping definition of [18]. But then is said that “All instances of UMLCD are instances of UML”. Does it also comes from the same definition (of [18])?Or has been checked only for the UML Class diagram example? In summary, I recommend that the authors: a) re-organize the paper in order to improve its readability and understandability b) clarify the issues mentioned above (and below) MINOR ISSUES: 1. English grammar should be thoroughly revised throughout the document. Among other things, please check punctuation marks: comma is used several times where a conjunction “and” is needed; in page 6, the constraints in the pre-condition for class2rdbms are introduced by using “(a)”, “(b)”, “(c)” and “(d)”. The separation between every two changes (the author uses a full stop or no punctuation mark); etc. 2. The “pre-condition imrpovement” keyword should be replaced by “pre-condition improvement”. 3. The 5th paragraph of the introduction starts with a sentence already mentioned in the 4th paragraph: “…using different strategies and qualify them using mutation analysis” 4. Page 3: A reference to black-box (and white-box) testing is required. 5. Page 4: “The effective input metamodel contains only classes, properties, their dependencies relevant the transformation” -> “The effective input metamodel contains only classes, properties, and the dependencies relevant to the transformation” 6. The page margins are not respected in some cases (e.g., “pre-conditions” in page 4). 7. Page 5: “Our case study is the transformation from UML Class Diagram models to RDBMS models called class2rdbms.” Please expand both domains or none of them. 8. Page 6. “an Property” -> “a Property” 9. Page 8: A reference to the Scala language is required. 10. Page 15: It is said that “CARTIER transforms eMMI , its invariants, the transformation, pre-condition pre(MT) and test strategy to an ALLOY model “ and actually is the user who has to manually encode almost all constraints. Please clarify. 11. Page 15: ”models that are raise exceptions” -> ”models that raise exceptions” 12. Table 2 and Fig 5 seems to be an single figure (due to its distribution) 13. Page 16: “these loop structures can navigation through” -> “these loop structures can navigate through” 14. Page 16: “It represents an excerpt (bottom part) of an output model produced by the original transformation of a generated (excerpt on the top part)” Something is missing before the parenthesis. 15. Page 18: “For instance, we can ask questions such as whether a large number of Association objects have a correlation with the mutation score?” -> “?” should be replaced by “.” 16. Page 20: “In the box whisker diagram of Figure 7 we shown” -> In the box whisker diagram of Figure 7 we show” 17. Page 20: “In the median case both AllRanges and AllPartitions strategy” -> “In the median case both AllRanges and AllPartitions strategies” 18. Appendix B and C: There are two different facts with the same name (no4CyclicClassProperty) Reviewer: 3 Public Comments (these will be made available to the author) The title of the paper is Automatic Model Synthesis to Test Model Transformations. However, most of the interesting and hard aspects that would benefit from automation are still performed manually: as there's no static analysis of the transformation, the classes/properties comprising the effective metamodel need to be specified manually. Also, OCL constraints are hand-translated to Alloy. The coherency of the paper can be substantially improved by shortening non-core sections (e.g. 3.2, 3.3) which have been explained in detail elsewhere, and/or revising its scope. The paper currently touches on all the following aspects: model typing, metamodel pruning, transforming Ecore to Alloy, input domain partitioning and transformation mutation, and this makes it particularly challenging to follow. Additional comments follow: Page 6: "... can be of type Class up to a depth of 3 ...". This constraint seems fairly arbitrary. Having a cycle doesn't always imply an infinite loop. This can be avoided - for example - if caching is used. Page 6: "In [15] the authors present some limitations of OCL". I can't see how this is relevant to the discussion. Page 15: "CARTIER performs static analysis...". I don't see where this is explained in the paper. By contrast, in page 5, the authors mention that they treat transformations as black-box components. Page 16: "The Kermeta interpeter throws a StackOverflowException": This is not a feature; it is the default behaviour of any Java-based interpreter. Also, this does not happen in all cases of infinite loops (e.g. while (true) {}) but only in cases where infinite loops are triggered by infinitely recursive method calls. Page 7: I can't follow the calculation of 10% from 20secs vs. 3hrs Page 10: "UML". UMLCD appears to be a "simplification" of UML, not a subset of it (at least not of a version I'm familiar with) Some spelling/grammar errors follow: Page 4: "their dependencies relevant the transformation" -> "and dependencies relevant to the transformation" Page 6: "not really limitation" -> "not really a limitation" Page 18: "are the use different SAT" -> "are the use of different SAT"