Modeling Language Evolution for Model Family Support

Sanaa Alwidian
School of Electrical Engineering and Computer Science
University of Ottawa
Ottawa, Canada
salwidia@uottawa.ca

***Abstract*— In Model-Driven Engineering, models can evolve over time or vary along dimensions such as products. Such evolution results in a set of related models called *model family*. A model family can be captured with a "150% model" that merges the family members, while enabling the extraction of the individual models. In this context however, a 150% model may no longer conform to the original metamodel of the family members. This paper presents my Ph.D. research agenda on inferring the metamodel of a model family from the structure of the metamodel of its members. In particular, I aim to define a technique that minimally relaxes the original metamodel constraints related to multiplicities of attributes and association ends. Although a simpler problem is to infer minimal constraint relaxations from the current family members, the more interesting problem is to *predict* where such relaxations are needed in the metamodel, so that existing tools and analysis techniques can be adapted *once* and *minimally* for a given modeling language. This work is applicable to the regulatory domain, for example, as regulations evolve and have variations that need to be captured and analyzed using slightly different goal models. Such work would also indirectly help the community gain a better understanding of the nature of metamodels.**

Keywords*— Conformance; constraint relaxation; evolution; metamodel; Model-Driven Engineering; model family; variability; 150% model.

I. Introduction and Problem Formulation

In Model Driven Engineering (MDE), models and/or their metamodels evolve continuously and therefore need to be managed to ensure conformance. The metamodel evolution and model co-evolution problem [1],[2] is a well-addressed aspect of evolution in MDE. In such approaches, a metamodel evolves from MM to MM', and then a model co-evolution from M to M' is carried out *afterward* (see Fig. 1). However, to the best of our knowledge, there is a lack of approaches that attempt to evolve metamodels in response to model evolution. We initially refer to this context as the *model-triggered metamodel evolution* problem (Fig. 2), characterized as follows: If a model M (that conforms to a the original metamodel MM) evolves or varies, resulting in a new model M' that is no longer conform to MM, how should we extend MM (ideally with the least amount of changes) into MM', in order for M' to be conform to MM'?

A. *Problem Specification.*

A *model family* assembles a set of related models that vary along some dimension such as time or product in product/software lines. In a negative variability model (an approach that starts with a complete model of all variations and selectively remove deselected artefacts [3]), a model that captures the union of all members of a model family is often called "150% model" [3],[4]. A 150% model not only captures all the family members (for example, to enable analysis on all members at once), but also enables the extraction of individual members. In the context of a model family, we observed that even if each of the family members conforms to the same metamodel, the 150% model that captures this family *may not* conform to that metamodel. Another interesting observation is that the evolution/variation of models in a family does not require additions, deletions, or modifications of concepts to the original metamodel. Hence, in order to support model families (with large number of models), we only need to *relax* the metamodel internal constraints that are related to multiplicities of attributes and association ends and/or external well-formedness constraints. The general problem illustrated in Fig. 2 can therefore be simplified and characterized in this context (see Fig. 3): If models $M_0..M_n$ (that conform to a metamodel MM) are aggregated, resulting in a new model M_{150} that is no longer conforming to the original metamodel MM, how should we extend MM (ideally with the least amount of changes) into MM_{150}, in order for M_{150} as well as $M_0..M_n$ to conform to MM_{150}?

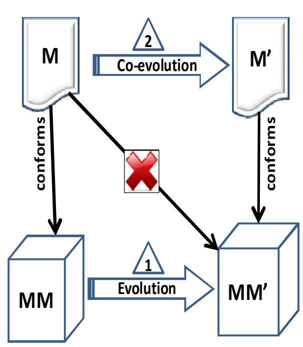

Fig. 1. Metamodel (MM) evolution and model (M) co-evolution problem

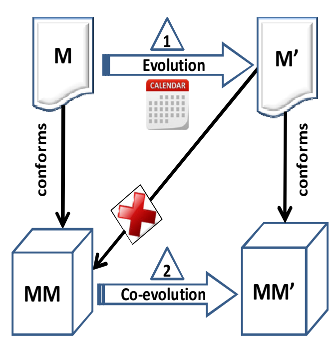

Fig. 2. Model-triggered metamodel (MM) evolution problem (general)

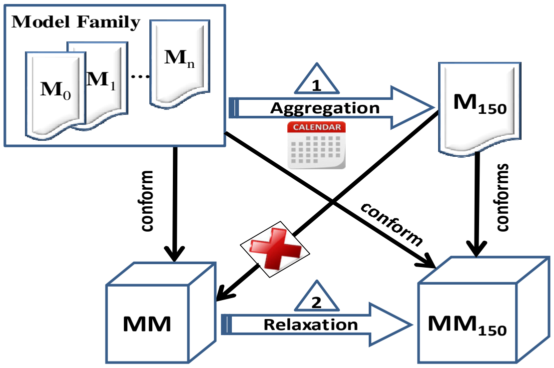

Fig. 3. Model family-specific metamodel evolution problem

B. *Motivation*

This work is inspired by issues faced with regulation modeling, in collaboration with Transport Canada, where there are regulations for different types of parties (e.g., airports and airlines of different sizes) requiring slightly different goal models [4]-[6]. If we try to capture all model variants with one goal model (e.g., using the Goal-oriented Requirement Language – GRL [7]) to minimize maintenance problems, we would face conformance issues because the language does not permit to capture the family with one model. For example, GRL limits the number of links between a pair of goals to 1, whereas the family model may need many. This is an issue along the product dimension, which is not limited to GRL but also common to most modeling languages. Similar problems occur along the time dimension when a model evolves. If a product has many versions over time, and if we want to analyze all versions (e.g., before releasing a patch that would affect them all), a 150% model would allow reasoning about all members at once, instead of reasoning about each member individually. The concern here is not in constructing the 150% model itself. Rather, the challenge emerges when 150% models violate conformance with the original metamodel, MM. In this case, we need to relax MM into MM150 to ensure that this 150% model is representable. A minimal relaxation is desirable in this context to minimize potential modifications to existing tools and analysis approaches.

C. *Scope of the Work.*

This work focuses on inferring the metamodel of a model family from the structure of the metamodel of its members through a minimal set of constraint relaxations. We are not dealing with the more general problem of inferring a metamodel from a collection of models that conform to different metamodels; our member models all use the same language. Also, we are not dealing with dynamic metamodel co-evolution upon changes; generating a new metamodel each time there is a new member added to the family is not practical, as this would imply developing new tools (for producing, analyzing, and transforming 150% models) each time. We are also not looking at language-specific solutions, e.g., through using metadata or user links in GRL. The ongoing challenge of this research is hence to predict the locations where metamodel relaxations are needed, without relaxing too much. Our long-term goal is hence to develop tools for the relaxed language only once. This research questions hence are:

- **RQ1**: How can we minimally relax a metamodel to support a model family (through a 150% model) in a way that enables the generation of all (and only) individual members?
- **RQ2**: How can we predict where relaxation is needed (i.e., relaxation points) in the original metamodel, for all potential 150% models of a specific language?
- **RQ3**: To what extent can the current tools and analysis techniques be adapted to model families (with minimum adaptation effort) to reason on all members of a 150% model?

II. RELATED WORK

Several approaches manage evolution of MDE models and metamodels in two directions: metamodel evolution and model co-evolution [1],[2],[8],[9],[10] (Fig. 1) and operation co-evolution [11]-[14]. In both directions, the goal is to update models and/or operations so that they conform to their evolved metamodel. In addition, metamodels can be extended with profiles such as in the UML.2x profile mechanism [15] to further restrict the metamodel's constructs and enforce the well-formedness of models of the domain-specific language. Furthermore, approaches for (meta)model decoration/annotation, such as the one from Kolovos et al. [16] have been used in an extension context. Model versioning approaches [17]-[20] have also been proposed to handle model evolution and track it through versioning, where differences between versions of the same model are detected. Run-time oriented approaches, such as EMF Facets [21], allow metamodel extension by adding classes, attributes or containment references. Aprajita et al. [22], [23] explicitly extended the metamodel of GRL to document explicit changes of model elements to specific versions of a metamodel. In [24], I have proposed the theoretical foundation of this work. The paper discussed the problem of metamodel relaxation to support evolution of models in the context of model families, and proposed a solution of predicting the locations of relaxations in a metamodel through tracking versions of members in a model family.

As the resulting artifact of this PhD thesis will be an evolved (e.g., relaxed) meta-model to accommodate model families, the above approaches are considered as related work. However, there are three major conceptual differences between my proposed work and existing approaches. The first difference is the driving factor of evolution. While existing approaches deal with models and/or operations co-evolution triggered by metamodel evolution, our work targets the evolution/relaxation of metamodels triggered by model evolution or variation, in the context of model families. The second difference is that unlike the existing approaches that conduct transformation/migration of models each time a metamodel changes, our approach aims to infer a single relaxed metamodel that accommodates all potential 150% models of a language, so as to develop tools for this relaxed metamodel only once. Finally, some of the related approaches either add new concepts to the original metamodel [21], or modify the language's validity constraints by further constraining their restrictions [15], while our approach relaxes some constraints instead. The approach of [22], [23] is currently specific to one language and supports limited kinds of changes to versions.

III. PROPOSED SOLUTION

I propose a Metamodel Relaxation approach, MeRe, to support the representation of model families by means of minimally relaxing particular constraints related to multiplicities and/or association ends. MeRe works in four main phases to enable metamodel relaxation: first, the union of all model elements in all valid members of a model family is captured by one single model, M150, as in [25]. Second, changes among the different versions of models are detected through the use of M150, whose elements can now be extended with a delta (Δ) annotation. The Δ denotes a change of elements and/or links from M1 to M2 (for example, a GRL decomposition link in M1 becomes a contribution link in M2). This delta could be inferred by calculating the difference between M1 and M2, Diff (M1, M2), using, for example, the approach proposed by Rivera et al. [26]. The purpose of this phase is to detect and extract pairs of elements that have changed, denoted as Ei and EiΔ. Third, conformance between the original metamodel MM and the M150 model is verified. This is done by checking if the co-existence of change pairs (obtained in phase 2) in the same model could cause a violation of association/attribute multiplicities or other external (OCL) constraints of MM. For instance, two different links between the same pair of GRL goals will cause a violation. If non-conformance is detected, the fourth phase takes place, where the modeler decides on the relaxation points. At this level, it is still challenging to predict the exact locations of metamodel multiplicities that need to be relaxed, independently of the models in a family. Note that we do not have to follow a naïve brute-force approach and relax all multiplicity constraints and external constraints in the metamodel. Instead, we need to identify a technique to predict automatically where relaxation is needed in the metamodel, based on its structure or, empirically, on patterns of usages of the language.

IV. PLAN FOR EVALUATION AND VALIDATION

I am planning to validate and evaluate my PhD work following some of the methods described in [27],[28]. In particular, I will demonstrate the applicability of MeRe empirically, based on a large collection of models. As a first step, I will consider several models of products and several of their versions, and construct M150 for each family. In addition, I will consider the use of one model repository [29] to generate a set of virtual model versions or families, and also construct M150 for them. For each of these M150, if it violates MM, I will relax MM to MM150, with the minimum amount of changes such that MM150 would be small enough to only accommodate the members of a particular M150. To predict where relaxation is needed in MM (and to decide when to stop relaxing MM) for all potential M150 (RQ2), I will capture all M150 (from first step) in one single M150 called BigM150, and infer a relaxed metamodel called BigMM150. Then I will conduct a comparison based on differences between BigMM150 and each of the MM150 that I got before. The purpose of this step is to identify and predict relaxation points in BigMM150. Prediction quality will be measured with common metrics (e.g., precision and recall). These steps will be done for at least 3 languages (GRL, UML, and another one to be decided), so the results are not language dependent. I am then planning to conduct case studies from domains with highly evolving models, such as regulation modeling (section 1) to examine the applicability of the proposed approach in practice.

V. EXPECTED CONTRIBUTIONS

By addressing RQ1-RQ3, this research will provide these scientific contributions:

- Characterization of the requirements for minimally relaxing modeling languages to support all potential 150% models of a language (RQ1).
- Prediction heuristics for the locations where metamodel relaxations are needed, so that existing tools and analysis techniques be adapted once for families (RQ2).
- Examples of tools and analysis techniques evolved to support MM150, for two languages. This will enable reasoning about all family members at once (RQ3).
- Proof-of-concept tool support for MeRe.

VI. CURRENT STATUS

As of July 2017, I did a literature review of the model/metamodel evolution problem over the last 10 years and derived a characterization of the model-triggered metamodel evolution problem in the context of model families. A paper accepted in the ME'2017 workshop covers the theoretical foundations of this work. I am working towards formalizing models and metamodels based on ontologies to infer patterns of relaxation needed in the metamodel. A publication on this topic is the next step. The evaluation and validation of MeRe is currently in progress, with a paper planned for submission in the fall 2017. I intend to carry out integration of MeRe with a modeling tool as a prototype. I aim for a journal paper in 2018 that summarizes my findings and combines them with results of a case study. I aim for the publication of the PhD thesis by the end of 2018.

VII. CONCLUSION

This paper presents my research motivation, problem statement, proposed solution, evaluation plan and the current progress on the MeRe approach. The ongoing challenge of this research is to predict metamodel relaxations points such that existing tools and analysis techniques would be adapted once for all potential model families.

REFERENCES

[1] J. Sprinkle and G. Karsai, "A domain-specific visual language for domain model evolution," Journal of Visual Languages and Computing, 15(3-4), pp. 291–307, 2004

[2] A. Cicchetti, D. Ruscio, R. Eramo, and A. Pierantonio, "Automating co-evolution in model-driven engineering," in 12th International Enterprise Distributed Object Computing Conference (EDOC), pp. 222–231. IEEE Computer Society, 2008.

[3] A. Polzer, D. Merschen, G. Botterweck, A. Pleuss, J. Thomas, B. Hedenetz, and S. Kowalewski, "Managing complexity and variability of a model-based embedded software product line," Innovations in Systems and Software Engineering, 8(1), pp. 35–49, 2012.

[4] A. Palmieri, P. Collet, and D. Amyot, “Handling regulatory goal model families as software product lines,” in Advanced Information Systems Engineering-27th International Conference, LNCS, vol. 9097, pp. 181–196. Springer, 2015.

[5] A. Shamsaei, D. Amyot, A. Pourshahid, E. Yu, G. Mussbacher, R. Tawhid, E. Braun, and N. Cartwright, “An approach to specify and analyze goal model families,” in 7th System Analysis and Modelling (SAM) Workshop, LNCS, vol. 7744, pp. 347–52. Springer, 2012.

[6] R. Tawhid, M. Alhaj, G. Mussbacher, E. Braun, N. Cartwright, A. Shamsaei, D. Amyot, S.A. Behnam, and G. Richards, “Towards Outcome-Based Regulatory Compliance in Aviation Security,” in 20th IEEE Int. Requirements Engineering Conference (RE’12), pp. 267–272. IEEE CS, 2012.

[7] D. Amyot and G. Mussbacher, “User Requirements Notation: The first ten years, the next ten years,” Journal of Software, 6(5), pp. 747–768, 2011.

[8] A. Cicchetti, D. Davide, R. Eramo, and A. Pierantonio, “Meta-model differences for supporting model co-evolution,” in Proceedings of the 2nd Workshop on Model-Driven Software Evolution (MODSE), pp. 1–10. Université de Nantes, 2008.

[9] S. Becker, B. Gruschko, T. Goldschmidt, and H. Koziolek, “A process model and classification scheme for semi-automatic meta-model evolution,” in 1st Workshop MDD, SOA und IT-Management (MSI), GI, GiTO-Verlag, pp. 35–46, 2007.

[10] B. Gruschko, D. Kolovos, and R. F. Paige, “Towards synchronizing models with evolving metamodels,” in International Workshop on Model-Driven Software Evolution, paper 3, 2007.

[11] D. R. Davide, I. Ludovico, and A. Pierantonio, “A methodological approach for the coupled evolution of metamodels and ATL transformations,” in ICMT 2013: Theory and Practice of Model Transformations, LNCS, vol. 7909, pp. 60–75. Springer, Berlin, Heidelberg, 2013.

[12] D. Méndez, A. Etien, A. Muller, and R. Casallas, “Transformation migration after metamodel evolution,” in Proc. Model and Evolution Workshop (ME’10), <inria-00524145>, 2010.

[13] B. Gruschko, “Towards structured revisions of metamodels and semi-automatic model migration,” Position Paper for the Eclipse Modeling Symposium, 2006.

[14] J. M. Sprinkle, Metamodel driven model migration. Doctoral dissertation, Vanderbilt University, Nashville, USA, 2003.

[15] OMG, Unified Modeling Language (UML), Version 2.5, formal/2015-03-01, 2015.

[16] D. S. Kolovos, L. M. Rose, N. D. Matragkas, R. F. Paige, F. A. C. Polack, and K. J. Fernandes, “Constructing and navigating non-invasive model decorations,” in ICMT 2010, LNCS, vol. 6142, pp. 138–152. Springer, 2010.

[17] M. Alanen and I. Porres, “Version control of software models,” in Advances in UML and XML-Based Software Evolution, Chapter III, pp. 47–70. Idea Group Publishing, 2005.

[18] H. Oliveira, L. Murta, and C. Werner, “Odyssey-VCS: a flexible version control system for UML model elements,” in Proc. 12th Int. Workshop on Software Configuration Management, pp. 1–16. ACM, 2005.

[19] K. Altmanninger, G. Kappel, et al., “AMOR – towards adaptable model versioning,” in 1st International Workshop on Model Co-Evolution and Consistency Management (MCCM'08), Workshop at MODELS'08, Toulouse, France, 2008.

[20] M. Koegel, J. Helming, and S. Seyboth, “Operation-based conflict detection and resolution,” in CVSM'09: Proceedings of the 2009 ICSE Workshop on Comparison and Versioning of Software Models, pp. 43–48. IEEE Computer Society, 2009.

[21] EMF Facet, http://eclipse.org/facet [online; accessed: April 2017].

[22] Aprajita and G. Mussbacher, “TimedGRL: Specifying goal models over time,” in 6th Int. Model-Driven Requirements Engineering Workshop (MoDRE), pp. 125–134. IEEE CS, 2016. doi:10.1109/REW.2016.035.

[23] Aprajita, S. Luthra., and G. Mussbacher “Specifying Evolving Requirements
Models with TimedURN”. 9th Workshop on Modelling in Software Engineering (MiSE 2017), Buenos Aires, Argentina, pp. 26-32, May 2017.

[24] S. Alwidian and D. Amyot “Relaxing Metamodels for Model Family Support”. *11th Workshop on Models and Evolution (ME 2017)*, Austin, USA, Sept. 2017. IEEE.

[25] K. Czarnecki and M. Antkiewicz, “Mapping features to models: a template approach based on superimposed variants,” in GPCE 2005, LNCS, vol. 3676, pp. 422–437. Springer Berlin Heidelberg, 2005.

[26] J.E. Rivera and A. Vallecillo, “Representing and operating with model differences,” in International Conference on Objects, Components, Models and Patterns, LNBIP, vol. 11, pp. 141–160. Springer Berlin Heidelberg, 2008.

[27] B. Kitchenham, S. Linkman, and D. Law, “Desmet: a methodology for evaluating software engineering methods and tools”. Computing & Control Engineering Journal, 8(3):120–126, 1997.

[28] P. Runeson, M. Höst, A. Rainer, and B. Regnell, Case Study Research in Software Engineering - Guidelines and Examples. Wiley, 2012.

[29] J. Di Rocco, D. Di Ruscio, L. Iovino, and A. Pierantonio, “Collaborative Repositories in Model-Driven Engineering”, IEEE Software, 32(3), pp. 28–34, May 2015.