Thesis Audit

Change-Based Model Persistence

Alfa Ryano Yohannis
ary506@york.ac.uk

Supervisors:
Dimitris Kolovos
Fiona Polack
Horacio Hoyos Rodriguez

Department of Computer Science
University of York
United Kingdom

November 26, 2018

Abstract

Most of the models in Model-Driven Engineering are persisted in state-based formats. While state-based persistence has certain advantages, it is problematic when it comes to detecting changes in large-scale models. As an alternative, this work proposes a change-based approach that involves persisting the full sequence of changes made to models. Persisting a model in a change-based format has the potential to deliver benefits over state-based persistence, such as the ability to detect changes, compare and merge models much faster and more precisely, which can then yield positive knock-on effects on helping developers compare and merge models in collaborative modelling environments. Nevertheless, change-based persistence also comes with downsides, such as ever-growing model file sizes and increased model loading time. So far, the initial implementation has been presented at the FlexMDE 2017 workshop, the proposed algorithm to optimise the loading time of change-based persistence has been presented at the ECMFA 2018 conference, and a paper discussing hybrid model persistence also has been presented at the Models and Evolution 2018 workshop. Currently, this research is still working on the change-based model comparison and merging. The thesis writing up is planned to start next year. An adjustment of the previous research plan is also presented in this report.

Contents

List of Tables

Chapter 1

Introduction

This Chapter briefly presents the background of this work as well as the research questions that will be addressed in this project. Several research objectives are then defined to answer the research questions. Lastly, research outputs and scoping are also presented.

1.1 Background

Most of the models in the context of Model-Driven Engineering are persisted in state-based formats. In such approaches, model files contain snapshots of the models' contents, and activities like version control and change detection are left to external systems such as file-based version-control systems and model differencing facilities. Activities such as change-detection (identifying parts that have changed in a model compared to a previous version) and model comparison (finding differences between models) are computationally consuming for state-based models [1]. Thus, a new approach is needed to make the computation more efficient.

As an alternative to state-based persistence, this work proposes that a model can also be persisted in a change-based format, which persists the full sequence of *changes* made to the model instead. The concept of change-based persistence is not new and has been used in persisting changes to software, object-oriented databases, and hierarchical documents [2, 3, 4]. The change-based approach can improve detecting differences more precisely at the semantic level – that is by providing finer-granularity information (e.g. types of changes, the order of the changes, elements that were changed, previous values, etc.) – and therefore provide support to resolve them [5]. The ordered nature of change-based persistence means that changes made to a model can be identified sequentially without having to explore and compare all elements of the model and its previous version. Based on these arguments, this work explores the advantages and shortcomings of change-based persistence as an alternative approach to state-based persistence for models conforming to 3-layer metamodelling architectures such as EMF and MOF. Persisting models in a change-based format can bring a number of envisioned benefits over state-based persistence, such as the ability to detect changes much faster and more precisely, which can then have positive knock-on effects on supporting (1) developers compare and merge models in collaborative modelling environments, and (2) incremental model management (e.g. incremental query [6] and model-to-text transformation [7]).

Nevertheless, change-based persistence also comes with downsides, such as ever-growing model files [2, 8] and increased model loading time [5] which increase storage and computation costs. A model that is frequently modified will increase considerably in file size since every change is added to the file. The increased file size (proportional to the number of persisted changes) will, in turn, increase the loading time of the model since all changes have to be replayed to reconstruct the model's eventual state. These downsides have to be mitigated to enable the practical adoption of change-based persistence. One approach to reducing the file size of change-based models is by removing changes that do not affect the eventual state of the model. For the increased loading time, it can be mitigated by ignoring – i.e. not replaying – changes that are cancelled out by later changes or employing change-based and state-based persistence side-by-side so that the benefits of state-based persistence on loading time can be obtained. Other downsides are change-based persistence requires integration with existing tools – since it is still a non-standard approach – for its adoption [9], and still has limited support for standard, text-based version controls for collaborative development [9]. These downsides can be addressed by developing a change-based persistence plugin for a specific development environment (e.g. Eclipse) and persisting changes in text-based format to support text-based version controls (e.g. Git, SVN).

1.2 Research Questions

The hypothesis of this work is that **"Change-based persistence reduces the execution time of model change-detection, model comparison, and model merging for large models compared to their execution time in state-based persistence, with acceptable trade-offs on loading and persisting time, memory footprint, and storage space consumption"**. The execution time is the time required to complete the processes (e.g. change-detection, model comparison, model merging, or persisting changes). Model change-detection is identifying changed elements of a model compared to its previous version/ancestor while the model comparison is finding the differences between two models that come from the same ancestor. Model merging is reconciling two models that come from the same ancestor and combining them to produce a new model. Using the term "large models" we refer to models with more than 1M elements, consistently with [10, 11]. Model load time is the amount of time required to load a model into memory. Persisting changes is saving changes made to a model into a persistent representation (e.g. a file). Memory footprints are the sizes of memory used to execute the processes. Disk space consumption is the amount of storage consumed by the persistence.

To assess the validity of the hypothesis, this work aims to answer the following research questions:

1. **How to persist models in a change-based format? How does it perform compared to state-based persistence on saving changes?**

The concept of change-based persistence has to be translated into an implementation in a modelling framework context so that it can be applied for model persistence, and therefore its impact on model change-detection, model comparison, and model merging can be assessed.

2. **How to detect changes in change-based models – comparing them to their ancestors/previous versions? To what extent does change-detection in change-based models perform compared to change-detection in state-based models?**

   The purpose of using change-based persistence in this work is to improve change-detection. The change-based persistence will have change-detection time that is smaller than the change-detection time of state-based persistence.

3. **How to compare change-based models that come from the same ancestor? How does the comparison of change-based models perform compared to the state-based model comparison?**

   The knock-on effect of faster change-detection on model comparison will also be investigated. Due to the nature of change-based models, the mechanism to perform change-based model comparison will differ substantially from the current state-based model comparison. It is expected that comparison of change-based models will be significantly faster than the comparison of state-based models.

4. **How to merge different change-based models that come from the same ancestor? How does the merging perform compared to model merging in state-based persistence?**

   Another knock-on effect of faster change-detection of change-based persistence is faster model merging. Similar to the change-based model comparison, the mechanism to merge change-based models will differ substantially from merging state-based models. It is expected that the change-based model merging will be much faster than state-based model merging.

1.3 Research Objectives

This research aims to meet the following research objectives to answer the research questions.

1. Develop an implementation of change-based persistence so it can be applied to persist models in change-based format, and evaluate the correctness of change-based models that it produces and its performance on saving changes against state-based persistence.

2. Develop a solution to detect changes in change-based models, and compare its execution time and memory footprint against change-detection in state-based models.

3. Develop a solution to compare change-based models, and compare its execution time and memory footprint against model comparison in state-based models.

4. Develop a solution to merge different change-based models, and compare its execution time and memory footprint against model-merging in state-based models.

1.4 Research Outputs

By the end of this research, these following outputs will have been produced:

1. Prototypes for change-based persistence.

2. Solutions – including their implementation and evaluation – for file size and loading time reduction, change-detection (finding parts that already changed of a model compared to its previous version/ancestor), model comparison (finding differences between models that come from the same ancestor), and model merging of change-based persistence.

3. Publications and a thesis documenting the outcomes of this research.

1.5 Research Scope

The scope of this research will be restricted to models conforming to 3-level meta-modelling architectures. The Eclipse Modelling Framework will be used, as a representative example of such architectures, for the implementation of all algorithms and prototypes.

Chapter 2

Thesis Structure

This section provides the description of the planned thesis structure and its progress. Each subsection corresponds to a chapter in the thesis report and describes the purpose, contents, and progress of the chapter. Some of the contents have been discussed in this research's Progress Report [12], Thesis Outline [13], and written papers [14, 15, 16], and they will be transferred to the thesis report with proper adjustments. The planned time frame for each chapter is presented in Chapter 3.

2.1 Chapter 1: Introduction

This chapter is intended to present the motivation and purpose of this research. It will comprise the (1) background of the research as well as the (2) research hypothesis, (3) research questions, (4) research objectives, (5) research outputs, and (6) research scope. All of these contents have been defined in the Progress Report [12] and this Thesis Audit, and they will be transferred to the Thesis Report with proper adjustments.

2.2 Chapter 2: Literature Review

This chapter is dedicated to summarise work related to change-based persistence and comparison, critically assesses the advantages and disadvantages of current approaches, and seek the opportunities to contribute novel knowledge to the field. The chapter will comprise (1) change-based approaches in software engineering, (2) change-based model persistence, (3) state-based model persistence, (4) text-based comparison and merging, (5) state-based model comparison and merging, and (6) change-based model comparison and merging. Most of these topics have been discussed briefly in the Progress Report [12].

2.3 Chapter 3: Change-based Model Persistence

This chapter is planned to present the concept of change-based model persistence and its core implementation. The chapter will comprise the concept of change-based model persistence, and the design of the core implementation. The implementation is evaluated by comparing the eventual model produced by replaying a change-based representation to the same model but loaded from a state-based persistence (e.g. XMI). These contents have been published in a workshop [14].

2.4 Chapter 4: Optimised Loading of Change-based Model Persistence

Change-based persistence comes with a downside of ever-growing file sizes [2, 8] which causes increased loading time [5]. Reducing the loading time is essential to facilitate the practical adoption of change-based persistence. One way to reduce the increased loading time is by ignoring – not replaying – changes that are cancelled out by subsequent changes.

To evaluate the efficiency of the proposed approach, the optimised loading is compared to a naive loading of a change-based representation and loading the same model from a state-based representation. They are compared on time required to load models and the memory footprint after loading models. Evaluation is also performed on time required for persisting changes between change-based and state-based persistence to show the benefit of change-based persistence on saving changes.

This chapter is intended to present the optimisation approach along with its implementation and evaluation. The contents of this chapter will be largely based on an accepted conference paper [15].

2.5 Chapter 5: Hybrid Model Persistence

While optimised loading is faster than the naive loading, the benefits are moderate and that the optimised loading is still slower than loading from a state-based representation [15]. This finding has motivated the design and development of a hybrid persistence approach, that is augmenting a change-based representation with a state-based representation.

The hybrid model persistence approach is evaluated by comparing it to state-based persistence (e.g. XMI, NeoEMF [10]) on time, memory footprint, and storage space required for loading models and persisting changes. The evaluation is also performed on time required for detecting changes between hybrid and state-based persistence to show the benefit of change-based persistence over state-based model comparison.

This chapter is dedicated to present the hybrid model persistence approach together with its implementation and evaluation. The contents of this chapter will be largely based on a workshop paper [16] that is currently under review.

2.6 Chapter 6: Change-based Model Comparison

This chapter will present a change-based model comparison algorithm with its implementation and evaluation. Change-based persistence is expected to speed-up model comparison because the information required to identify changes is already contained in the models' representation.

The proposed algorithm will be evaluated by comparing it to state-based model comparison (e.g. EMF Compare [17]) on the time and memory footprint required to find all differences between models. This research will derive synthetic models from branches of real-world software projects as datasets in the evaluation. The alternative strategy is to perform an experiment on modellers. They will be asked to develop models in parallel. The produced models will be upscaled and used in the evaluation.

The findings of this chapter will be published together with the findings of Chapter 7.

2.7 Chapter 7: Change-based Model Merging

This chapter is planned to present change-based model merging. After identifying the differences between two versions of a change-based model, there is a need to reconcile all the differences into a new version of the model. Thus, conflicts between the two versions have to be identified and resolved – using predefined rules or based on users' decisions, and by then merging steps can be determined to merge the two versions.

Similar to change-based model comparison in Section 2.6, the proposed merging approach will also be evaluated by comparing it to the merging of state-based persistence (e.g. EMF Compare) on the affected time and memory footprint. The evaluation will also use the same datasets in Section 2.6.

The findings of this chapter will be published together with the findings of Chapter 6.

Chapter 3

Previous Plan

This Chapter presents previous plan's outstanding tasks for each chapter in the thesis report and the estimated time to complete the tasks. The timetable for the previous plan's remaining tasks is in Table 3.1.

1. **Chapter 1: Introduction**
   1.1. *Writing Up*
      1.1.1. Transfer this chapter to the thesis report. (estimated time: 5 days)
2. **Chapter 2: Literature Review**
   2.1. *Review Work*
      2.1.1. Review related work on the text-based comparison and merging, state-based model comparison and merging, and change-based model comparison and merging. (10 days)
   2.2. *Writing Up*
      2.2.1. Transfer the literature review to the thesis report. (5 days)
3. **Chapter 3: Change-based Model Persistence**
   3.1. *Writing Up*
      3.1.1. Transfer the related written paper to the thesis report. (5 days)
4. **Chapter 4: Optimised Loading of Change-based Model Persistence**
   4.1. *Writing Up*
      4.1.1. Transfer the related written paper to the thesis report. (5 days)
5. **Chapter 5: Hybrid Model Persistence**
   5.1. *Writing Up*
      5.1.1. Transfer the related paper to the thesis report. (5 days)
6. **Chapter 6: Change-based Model Comparison**

6.1. *Research*

6.1.1. Develop a solution for change-based model comparison. (15 days)
6.1.2. Develop an implementation of the solution. (15 days)
6.1.3. Evaluate the solution. (10 days)

7. **Chapter 7: Change-based Model Merging**

7.1. *Research*

7.1.1. Develop a solution for change-based model merging. (15 days)
7.1.2. Develop an implementation of the solution. (15 days)
7.1.3. Evaluate the solution. (10 days)

7.2. *Publication*

7.2.1. Write a paper for chapters 6 and 7. (20 days)

7.3. *Writing Up*

7.3.1. Transfer the related paper to the thesis report. (5 days)

8. **Chapter 8: Conclusions and Future Work**

8.1. *Writing Up*

8.1.1. Write conclusions and future works. (5 days)

Table 3.1: The time table for the remaining tasks of this research. One week consists of 5 working days.

Year	2018																											
Month	June				July				August				September				October				November				December			
Week	01	02	03	04	05	06	07	08	09	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28
Review Work																												
- Chapter 2																												
Research																												
- Chapter 6																												
- Chapter 7																												
Publication																												
- Chapter 6 & 7																												
Other																												
- Backup Time																												
Year	2019																											
Month	January				February				March				April				May				June				July			
Week	01	02	03	04	05	06	07	08	09	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28
Writing Up																												
- Chapter 1																												
- Chapter 2																												
- Chapter 3																												
- Chapter 4																												
- Chapter 5																												
- Chapter 6																												
- Chapter 7																												
- Chapter 8																												
- Other parts																												
- Wrap up																												
- Proofreading																												

3.1 Chapter 8: Conclusions and Future Work

This chapter is intended to present the conclusions of this research as well as the future work.

Chapter 4

Thesis Progress

This section reports the progress on the writing up of this research's thesis according to the planned thesis structure. So far, no writing up has been performed since it was planned to start on January 2019 (see Chapter 3). Fortunately, the contents of most chapters – excluding Chapters 6 and 7 – are already available in this research's previous reports [12], [13], and written papers [14, 15, 16]. The contents will be transferred to the thesis with proper adjustment.

The research activity on Change-based Model Comparison and Merging (Chapters 6 and 7) is already overdue, referring to the previous planned time frame, and now in the backup time (see Table 3.1). The previous proposed time frame to complete the research activities of both chapters is judged underestimated the technical challenges in implementing the proposed solutions. Thus, a new plan is proposed to adjust the time frame (see Chapter 5). The backup time now has been allocated to complete Chapter 6 – which is 26 weeks in total.

Chapter 7 and publication for Chapters 6 and 7 are postponed to next year. The time frame for Chapter 7 (Change-based Model Merging) is only allocated 20 days since it is estimated that the foundation for Change-based Model Merging will have been addressed in Chapter 6 thus it will require less time to complete it.

Some of the technical challenges – which I am still working on – that made the Chapter 2.6 overdue are:

1. Defining a complete list of rules (conditions) that determines the differences between two change-based models and evaluate their correctness against the differences identified in state-based model comparison (e.g. using EMF Compare). Some differences cannot be identified using current rules which means new rules should be formulated to detect the unidentified differences. For example, when deleting an element from a reference, EMF automatically also remove the element from the opposite reference and also detected in the state-based comparison using EMF Compare. In the current implementation of CBP, no event is generated from this kind of operation thus differences on opposite references cannot be detected.

2. Correctness evaluation. Since the differences identified using change-based comparison approach can be different from the differences identified using

state-based comparison. For example, two models $m_l = [a,\ b]$ and $m_r = [b,\ a]$ are compared. From the change history, element a was moved to the last position (thus a in m_r should be moved back to the first position to make both models equal), but state-based comparison might detect it as b should be moved to the last position to make the m_r equals to m_l. Thus, the correctness evaluation should not be on equality of the differences but on the equality of the states of the models after all differences are merged unidirectionally to one of the models.

3. The merging of differences mentioned above also has to be performed in correct order. An algorithm should be defined to determine a correct order to merge differences. Different order of merging differences might lead to a different final state.

Chapter 5

Updated Plan

This Chapter presents the new plan for the outstanding tasks for each chapter in the thesis report and the estimated time to complete the tasks. The new timetable is presented in in Table 5.1. Compared to the previous time table (Table 3.1), the backup time now has been allocated to Chapter 2.6 which means 26 weeks are allocated complete it. Chapter 2.7 and publication for Chapters 2.6 and 2.7 are postponed until next year.

1. **Chapter 1: Introduction**
    1.1. *Writing Up*
        1.1.1. Transfer this chapter to the thesis report. (estimated time: 5 days)
2. **Chapter 2: Literature Review**
    2.1. *Review Work*
        2.1.1. Review related work on the text-based comparison and merging, state-based model comparison and merging, and change-based model comparison and merging. (10 days)
    2.2. *Writing Up*
        2.2.1. Transfer the literature review to the thesis report. (5 days)
3. **Chapter 3: Change-based Model Persistence**
    3.1. *Writing Up*
        3.1.1. Transfer the related written paper to the thesis report. (5 days)
4. **Chapter 4: Optimised Loading of Change-based Model Persistence**
    4.1. *Writing Up*
        4.1.1. Transfer the related written paper to the thesis report. (5 days)
5. **Chapter 5: Hybrid Model Persistence**
    5.1. *Writing Up*

5.1.1. Transfer the related paper to the thesis report. (5 days)

6. **Chapter 6: Change-based Model Comparison**

   6.1. *Research*

      6.1.1. Develop a solution for change-based model comparison. (55 days)

      6.1.2. Develop an implementation of the solution. (55 days)

      6.1.3. Evaluate the solution. (20 days)

7. **Chapter 7: Change-based Model Merging**

   7.1. *Research*

      7.1.1. Develop a solution for change-based model merging. (15 days)

      7.1.2. Develop an implementation of the solution. (15 days)

      7.1.3. Evaluate the solution. (10 days)

   7.2. *Publication*

      7.2.1. Write a paper for chapters 6 and 7. (20 days)

   7.3. *Writing Up*

      7.3.1. Transfer the related paper to the thesis report. (5 days)

8. **Chapter 8: Conclusions and Future Work**

   8.1. *Writing Up*

      8.1.1. Write conclusions and future works. (5 days)

Table 5.1: The new time table for the remaining tasks of this research. One week consists of 5 working days.

Year	2018																											
Month	June				July				August				September				October				November				December			
Week	01	02	03	04	05	06	07	08	09	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28
Review Work																												
- Chapter 2																												
Research																												
- Chapter 6																												
Year	2019																											
Month	January				February				March				April				May				June				July			
Week	01	02	03	04	05	06	07	08	09	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28
Research																												
- Chapter 7																												
Publication																												
- Chapter 6 & 7																												
Writing Up																												
- Chapter 1																												
- Chapter 2																												
- Chapter 3																												
- Chapter 4																												
- Chapter 5																												
- Chapter 6																												
- Chapter 7																												
- Chapter 8																												
- Other parts																												
- Wrap up																												
- Proofreading																												

Chapter 6

Publications

Three papers have been written. The first paper [14] has been presented at the FlexMDE 2017 workshop, the second paper [15] has been presented at the ECMFA 2018 conference, and the third paper [16] has been presented at the Models and Evolution 2018 workshop. These papers can be found in Appendices.

1. A. Yohannis, F. Polack, and D. Kolovos, "Turning Models Inside Out," in Proceedings of the 3rd Workshop on Flexible Model Driven Engineering co-located with ACM IEEE 20th International Conference on Model Driven Engineering Languages and Systems (MoDELS 2017), 2017.

2. A. Yohannis, H. Hoyos Rodriguez, F. Polack, and D. Kolovos, "Towards Efficient Loading of Change-based Models," in Proceedings of the 14th European Conference on Modelling Foundations and Applications (ECMFA 2018) co-located with Software Technologies: Applications and Foundations (STAF 2018), 2018.

3. A. Yohannis, H. Hoyos Rodriguez, F. Polack, and D. Kolovos, "Towards Hybrid Model Persistence," in Proceedings of the Models and Evolution 2018 (ME 2018) co-located with ACM IEEE 21th International Conference on Model Driven Engineering Languages and Systems (MoDELS 2018), 2018.

Bibliography

[1] D. S. Kolovos, D. Di Ruscio, A. Pierantonio, and R. F. Paige, "Different models for model matching: An analysis of approaches to support model differencing," in *Proceedings of the 2009 ICSE Workshop on Comparison and Versioning of Software Models*, ser. CVSM '09. Washington, DC, USA: IEEE Computer Society, 2009, pp. 1–6. [Online]. Available: http://dx.doi.org/10.1109/CVSM.2009.5071714

[2] R. Robbes and M. Lanza, "A change-based approach to software evolution," *Electr. Notes Theor. Comput. Sci.*, vol. 166, pp. 93–109, 2007. [Online]. Available: https://doi.org/10.1016/j.entcs.2006.06.015

[3] E. Lippe and N. van Oosterom, "Operation-based merging," in *SDE 5: 5th ACM SIGSOFT Symposium on Software Development Environments, Washington, DC, USA, December 9-11, 1992*, 1992, pp. 78–87. [Online]. Available: http://doi.acm.org/10.1145/142868.143753

[4] C. Ignat and M. C. Norrie, "Operation-based merging of hierarchical documents," in *The 17th Conference on Advanced Information Systems Engineering (CAiSE '05), Porto, Portugal, 13-17 June, 2005, CAiSE Forum, Short Paper Proceedings*, 2005. [Online]. Available: http://ceur-ws.org/Vol-161/FORUM_17.pdf

[5] T. Mens, "A state-of-the-art survey on software merging," *IEEE Trans. Software Eng.*, vol. 28, no. 5, pp. 449–462, 2002. [Online]. Available: https://doi.org/10.1109/TSE.2002.1000449

[6] I. Ráth, Á. Hegedüs, and D. Varró, "Derived features for EMF by integrating advanced model queries," in *Modelling Foundations and Applications - 8th European Conference, ECMFA 2012, Kgs. Lyngby, Denmark, July 2-5, 2012. Proceedings*, 2012, pp. 102–117. [Online]. Available: https://doi.org/10.1007/978-3-642-31491-9_10

[7] B. Ogunyomi, L. M. Rose, and D. S. Kolovos, "Incremental execution of model-to-text transformations using property access traces," *Software & Systems Modeling*, Mar 2018. [Online]. Available: https://doi.org/10.1007/s10270-018-0666-5

[8] M. Koegel, M. Herrmannsdoerfer, Y. Li, J. Helming, and J. David, "Comparing state- and operation-based change tracking on models," in *Proceedings of the*

14th IEEE International Enterprise Distributed Object Computing Conference, EDOC 2010, Vitória, Brazil, 25-29 October 2010, 2010, pp. 163–172. [Online]. Available: https://doi.org/10.1109/EDOC.2010.15

[9] M. Koegel and J. Helming, "Emfstore: a model repository for EMF models," in *Proceedings of the 32nd ACM/IEEE International Conference on Software Engineering - Volume 2, ICSE 2010, Cape Town, South Africa, 1-8 May 2010*, 2010, pp. 307–308. [Online]. Available: http://doi.acm.org/10.1145/1810295.1810364

[10] G. Daniel, G. Suny, A. Benelallam, M. Tisi, Y. Vernageau, A. Gmez, and J. Cabot, "Neoemf: A multi-database model persistence framework for very large models," *Science of Computer Programming*, vol. 149, pp. 9 – 14, 2017, special Issue on MODELS'16. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0167642317301600

[11] J. Espinazo-Pagán, J. S. Cuadrado, and J. G. Molina, "Morsa: A scalable approach for persisting and accessing large models," in *Model Driven Engineering Languages and Systems, 14th International Conference, MODELS 2011, Wellington, New Zealand, October 16-21, 2011. Proceedings*, 2011, pp. 77–92. [Online]. Available: https://doi.org/10.1007/978-3-642-24485-8_7

[12] A. Yohannis, "Progress report: Change-based model persistence," Department of Computer Science, University of York, York, United Kingdom, Tech. Rep., December 2017.

[13] ——, "Progress thesis outline: Change-based model persistence," Department of Computer Science, University of York, York, United Kingdom, Tech. Rep., May 2018.

[14] A. Yohannis, D. S. Kolovos, and F. Polack, "Turning models inside out," in *Proceedings of MODELS 2017 Satellite Event: Workshops (ModComp, ME, EXE, COMMitMDE, MRT, MULTI, GEMOC, MoDeVVa, MDETools, FlexMDE, MDEbug), Posters, Doctoral Symposium, Educator Symposium, ACM Student Research Competition, and Tools and Demonstrations co-located with ACM/IEEE 20th International Conference on Model Driven Engineering Languages and Systems (MODELS 2017), Austin, TX, USA, September, 17, 2017.*, 2017, pp. 430–434. [Online]. Available: http://ceur-ws.org/Vol-2019/flexmde_8.pdf

[15] A. Yohannis, H. H. Rodriguez, F. Polack, and D. Kolovos, "Towards efficient loading of change-based models," in *Modelling Foundations and Applications - 14th European Conference, ECMFA 2018, Held as Part of STAF 2018, Toulouse, France, June 25-29, 2018. Proceedings*, 2018 (to be presented http://eventmall.info/ecmfa2018/program), accessed: 2018-04-19. [Online]. Available: https://goo.gl/5H5oBN

[16] A. Yohannis, H. H. Rodriguez, F. Polack, and D. S. Kolovos, "Towards hybrid model persistence," in *Proceedings of MODELS 2018 Workshops:*

ModComp, MRT, OCL, FlexMDE, EXE, COMMitMDE, MDETools, GEMOC, MORSE, MDE4IoT, MDEbug, MoDeVVa, ME, MULTI, HuFaMo, AMMoRe, PAINS co-located with ACM/IEEE 21st International Conference on Model Driven Engineering Languages and Systems (MODELS 2018), Copenhagen, Denmark, October, 14, 2018., 2018, pp. 594–603. [Online]. Available: http://ceur-ws.org/Vol-2245/me_paper_3.pdf

[17] Eclipse, "EMF Compare," https://www.eclipse.org/emf/compare/, accessed: 2018-01-15.