Extended Abstract of Performance Analysis and Prediction of Model Transformation

Vijayshree Vijayshree
University of Stuttgart
vijayshree.vijayshree@iste.uni-stuttgart.de

Markus Frank
University of Stuttgart
markus.frank@iste.uni-stuttgart.de

Steffen Becker
University of Stuttgart
steffen.becker@iste.uni-stuttgart.de

1 INTRODUCTION

Nowadays model transformation techniques are a popular technique within Model-Driven Engineering (MDE). Model transformations enable the generation of new models, realizing changes on individual models, and the synchronization between models. In the model transformations there exist various transforming languages, our focus is on Query View Transformation Operational (QVTo).

However, a system being developed with model transformations can grow in size and can become complex. For example, in an automotive domain, an AUTOSAR model of a large electronic control unit (ECU) for modern cars has over 170.000 model elements. In such cases, the execution of badly performing model transformations can take up to hours. Hence, performance is an important quality attribute of model transformations, e.g., execution time, memory usage. The existing technique[2] to identify and visualize the root cause of low performing transformation rules are not addressed properly, meanwhile, performance optimization is limited only to the transformation engine. Hence, model transformation rules are considered to be fixed and unchangeable. Unfortunately, transformation engineers have no insights about how long the transformation takes place and also to predict the performance.

To the best of our knowledge, there exists no research which makes analyzes of the transformation rules and presents to the transformation engineers.

Therefore it is our vision to provide an approach towards performance engineering of model transformation. Hence, this approach will enable us to systematically *monitor and visualize causes for performance issues as well as predict the performance of model transformations*.

We will demonstrate and identify the root cause of low performing QVTo rules with the help of three different phases. Firstly, in the monitoring framework, we measure the execution time of each QVTo transformation rules. Secondly, the profiler will graphically visualize the results. Finally, we will demonstrate to predict the overall execution time of QVTo transformations with the data stored in the database.

2 RELATED WORK

We extensively focus on state of the art closely related to optimization of the performance of model transformation engine.

A class of approaches [2] analyzes model transformations and compute worst-case execution time upon optimal search order of the story pattern elements. Also, another approache [7] of the Henshin interpreter, considers models to execute the model transformations and thus address the bad smells which affects the performance of model transformation.

Amstel [8] investigates the factors such as size and complexity of the input models affecting the performance of model transformations. The author has compared different languages and systematically analyzed the influence of extracted metric value on performance of model transformations.

Piers [5] provides a way to detect performance issues on a ATL transformation by a detailed analysis of the execution. An execution profile stores information about the execution time, the memory used, etc of the model transformation.

Becker [1] made analysis performance and prediction, by generating prototypes from models, which in turn generate code skeleton or require detailed models for the prototype.

Groner [3] provides possible visualization and refactoring methods to improve the performance of model transformation in a declarative way.

Hence, these approaches show a significant performance improvement by refactoring the engine of the model transformations. On the other hand lacks the measurements and refactoring of transformation rules. Hence, in this paper, we provide an approach for supporting transformation engineers in identifying the root causes. Complementary helping engineers improve the transformation rules by themselves, which leads to the performance gain.

3 PROPOSED APPROACH

In the proposed approach we will contribute towards improving the performance of model transformation in an imperative way. Fig. 1 explains the three phases of our approach namely **Monitoring**, **Profiling**, and **Prediction**. To generate a large test set of input instance models we use VIATRA solver [6]. Thus, the generated instances are then transformed into the respected output model with the help of QVTo rules and run by the QVTo engine. During the execution we using the Kieker monitoring framework [9] to gather all the necessary operational profiles [8] and place them in the database. The data from the database are visualized to identify the performance bottlenecks. In turn, the data from the database is used to predict the performance before the actual transformations.

Monitoring WP1: To analyze the performance of transformations, engineers need to learn about the operational profile [8] of the
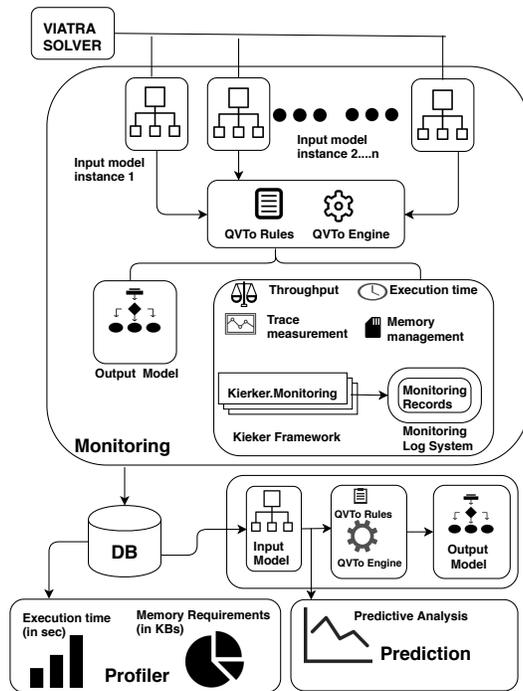
Figure 1: Proposed Approach

model transformations. The operational profile includes resource demands like execution times, rule evaluations, time spent in a model I/O. To measure the operational profile, we will extend the QVTo engine by injecting pointcuts of aspect-oriented programming [4] to the rules, whenever the engine executes these rules, Kieker monitoring API [9] is executed, to fetch operational profiles and measure them. Once the operational profile is measured they are stored in the database for further analysis and prediction.

Profiling WP2: The **WP1** will provide the raw and too detailed data about the operational profile of the model transformations. The raw data contains the execution time of each transformation rule, overall execution time. This detailed data will not directly help the transformation engineer in understanding where exactly lies the performance issues. Hence, this raw-data, in turn, needs to be visualized to support the engineer in identifying the root cause of badly performing transformation rules. Therefore, we are designing the profiler which presents the analyzed raw-data to the transformation engineer. A performance decline can be the result of changes in the model transformation or by changes in the meta-model or the operational profile. With the help of the developed profiler, we can easily identify and rank a list of possible causes for the performance decline by using monitoring data.

Prediction WP3: To support the engineer in predicting the performance, we need to develop a prediction framework. The developed framework will help to predict the performance change of model transformations. To predict the performance of a model without having a prior reference model or historic operational profile data of previously transformed model is always a difficult job and performance prediction of a model may not be accurate. To

generate the different instance reference model we need to scale the input model either automatically or manually. However, scaling manually is always error-prone and tedious job while we need to be very specific about the dependencies of the scaling elements. Hence, to overcome such a problem we are reusing the existing VIATRA [6] tool to automatically generate the instances of the input model, each instance is different. Then each instance model is transformed to obtain an output model. Subsequently, the operational profile (e.g., execution time and memory usage) of each instance model is obtained and thus, data are stored in the database. Eventually, the complete setup of generating instances, transforming the model and measurements of the operational profile is run in a continuous integration environment at a defined interval of time, which in turn serves as the reference data and thus, helps in performance prediction of transformations.

4 CONCLUSION

In this paper, we demonstrated an approach to identify the root cause of low performing QVTo rules. We presented the three phases of our approach namely monitoring, profiling, and prediction. In the monitoring phase, we will systematically monitor all the operational profile with the help of aspect- oriented pointcuts and kieker framework. In the profiling phase, we visualize the monitored operational profile of the monitoring phase and support the transformation engineer to identify root cause of badly performing transformation rules. With the use of VIATRA solver, we will automatically generate instances of input model and perform model transformations to measure the operational profiles and store them in database. This particular monitored data will be used for the prediction purposes.

5 ACKNOWLEDGEMENT

REFERENCES

[1] Steffen Becker, Tobias Dencker, and Jens Happe. 2008. Model-driven generation of performance prototypes. In *SPEC International Performance Evaluation Workshop*. Springer, 79–98.

[2] Sven Burmester, Holger Giese, Andreas Seibel, and Matthias Tichy. 2005. Worst-case execution time optimization of story patterns for hard real-time systems. In *Proc. of the 3rd International Fujaba Days*. 71–78.

[3] Raffaela Groner, Matthias Tichy, and Steffen Becker. 2018. Towards Performance Engineering of Model Transformation. In *Companion of the 2018 ACM/SPEC International Conference on Performance Engineering*. ACM, 33–36.

[4] Gregor Kiczales, John Lamping, Anurag Mendhekar, Chris Maeda, Cristina Lopes, Jean-Marc Loingtier, and John Irwin. 1997. Aspect-oriented programming. In *European conference on object-oriented programming*. Springer, 220–242.

[5] William Piers. 2010. ATL 3.1–Industrialization improvements. In *Proceedings of the 2nd International Workshop on Model Transformation with ATL*. Citeseer.

[6] Oszkár Semeráth, András Szabolcs Nagy, and Dániel Varró. 2018. A graph solver for the automated generation of consistent domain-specific models. In *Proceedings of the 40th International Conference on Software Engineering*. ACM, 969–980.

[7] Matthias Tichy, Christian Krause, and Grischa Liebel. 2013. Detecting Performance Bad Smells for Henshin Model Transformations. *Amt@ models* 1077 (2013).

[8] Marcel F van Amstel, Mark GJ van den Brand, and Phu H Nguyen. 2010. Metrics for model transformations. In *Proceedings of the Ninth Belgian-Netherlands Software Evolution Workshop (BENEVOL 2010), Lille, France (December 2010)*.

[9] André Van Hoorn, Jan Waller, and Wilhelm Hasselbring. 2012. Kieker: A framework for application performance monitoring and dynamic software analysis. In *Proceedings of the 3rd ACM/SPEC International Conference on Performance Engineering*. ACM, 247–248.