Patterns and styles for incremental model transformations

Invited paper

Dániel Varró[1][2]

[1] Budapest University of Technology and Economics
Department of Measurement and Information Systems
Magyar tudósok krt. 2, Budapest, H-1117, Hungary
[2] MTA-BME Lendület Research Group on Cyber-Physical Systems
varro@mit.bme.hu

Abstract. The usability of code generators in industrial modeling frameworks is frequently hindered by long re-generation time imposed by complex dependencies between different modules and chains of code generation phases. Incremental model transformation techniques may improve both scalability and usability by regenerating only specific design artifacts dependent on a specific change. In this paper, I provide an overview on different styles and levels of incrementality in model transformations of the VIATRA open source framework.

Keywords: model transformation patterns, incremental evaluation, reactive programming, code generation

1 Incremental techniques in model-based design tools

As the role of model-driven engineering (MDE) steadily increases in the design of critical embedded and cyber-physical systems, more emphasis is put on the quality, scalability and usability of advanced design and verification tools [7,3]. Such tools aim to highlight conceptual flaws early in the design process, thus improve quality and decrease costs at the same time. Moreover, code generation and model transformation techniques deeply embedded in such tools improve productivity by automatically deriving various artifacts as necessitated by certification standards.

However, software and systems engineers need to face usability and scalability issues when using such tools due to complex workflows of interdependent design steps [4] or chains of automated model transformation [**?**] and code generation steps. On the one hand, such transformation chains are beneficial to reduce abstraction gaps by introducing intermediate models and unifying some optimization steps. But compiling a model into a full, standalone project may take hours in practice.

To tackle this challenge, incremental, event-driven model transformation techniques and tools [1,11,6] may be helpful to react to model changes. Incremental model validation efficiently re- evaluates constraints and design rules upon each model change [10,9,2]. Furthermore, in incremental code generators, only artifacts affected by a specific model change will be re-generated while the rest of the target output is left unaltered. Such techniques have been successfully used in various design and validation tools for embedded and cyber-physical systems as reported in [4,8,5][3].

2 Different styles of incrementality

The actual level of incrementality in these applications differs and it can be fine-tuned for a transformation problem. In our practice, different styles and levels of incrementality could be categorized as follows:

- **No incrementality** is a batch transformation re-executed from scratch for all source models and all changes of these source models.
- **Dirty incrementality** is a large-step incrementality approach which is very common in industrial practice. Here one marks a model to be dirty upon a source change, and then re-run transformations only on dirty models. This technique significantly reduces the number of regenerated artifacts, but cleanup after an error and chaining of transformation steps is non- trivial.
- **Incrementality by traceability** is a small-step incrementality approach which relies on the existence of traceability links between source and target model elements generated during a first transformation. Missing trace links are detected and incrementality is achieved by re-executing the transformations for untraceable elements only. This technique may further reduce the effects of a change, but it depends on a smart matcher of traceability links.
- **Reactive source incrementality** is also a small-step incrementality approach where the firing of transformation rules are triggered only by changes in the source model but not the traceability links between source and target models. Such approaches may restrict the expressiveness of the transformation language, but offer improved chaining of incremental transformations.

Sample model transformations illustrating these scenarios are available from `https://github.com/IncQueryLabs/incquery-examples-cps` with complete source code, documentation and performance evaluation using the VIATRA framework. More details on reactive incremental transformations are provided in [11].

Further challenges. In practice, we find the complex interactions and chaining of incremental transformation and code generation steps to be the most challenging. In this setup, a change in a (front-end) source model may trigger reactions, which results in changing one or more target models. However, as these model changes may trigger further reactions along the transformation chain which require incremental handling themselves - potentially using a different style. With inappropriate tool support, it is surprisingly easy to achieve circular dependencies between different steps where different incremental transformations are executed continuously (e.g. simultaneously running validations and quick fixes).

[3] And also in `https://github.com/IncQueryLabs/EMDW-Common/wiki`

Acknowledgments. The author is highly grateful to the entire VIATRA development team, and particularly, Gábor Bergmann, Ábel Hegedüs, Ákos Horváth, István Ráth and Zoltán Ujhelyi for different sort of inputs and feedback. This paper was partially supported by the MONDO Project (EU ICT-611125), the FP7 ARTEMIS CONCERTO (ART-2012-333053) project and the MTA-BME Lendület Research Group on Cyber-Physical Systems.

References

1. Bergmann, G., Dávid, I., Hegedüs, Á., Horváth, Á., Ráth, I., Ujhelyi, Z., Varró, D.: Viatra 3: A reactive model transformation platform. In: Theory and Practice of Model Transformations - 8th International Conference, ICMT 2015. LNCS, vol. 9152, pp. 101–110 (2015)
2. Cabot, J., Teniente, E.: Incremental integrity checking of UML/OCL conceptual schemas. J. Syst. Softw. 82(9), 1459–1478 (2009)
3. Etzlstorfer, J., Kusel, A., Kapsammer, E., Langer, P., Retschitzegger, W., Schoenboeck, J., Schwinger, W., Wimmer, M.: A survey on incremental model transformation approaches. In: Proceedings of the Workshop on Models and Evolution co-located with MoDELS 2013. CEUR Workshop Proc., vol. 1090, pp. 4–13 (2013)
4. Horváth, Á., Hegedüs, Á., Búr, M., Varró, D., Starr, R.R., Mirachi, S.: Hardware-software allocation specification of ima systems for early simulation. In: Digital Avionics Systems Conference (DASC). IEEE, Colorado Springs, US (2014)
5. Horváth, A., Ráth, I., Hegedüs, A., Balogh, A.: Decreasing your coffee consumption with incremental code regeneration. In: EclipseCon France (2015)
6. Jouault, F., Tisi, M.: Towards incremental execution of ATL transformations. In: Tratt, L., Gogolla, M. (eds.) Theory and Practice of Model Transformations, LNCS, vol. 6142, pp. 123–137. Springer Berlin Heidelberg (2010)
7. Kolovos, D.S., Rose, L.M., Matragkas, N., Paige, R.F., Guerra, E., Cuadrado, J.S., De Lara, J., Ráth, I., Varró, D., Tisi, M., Cabot, J.: A research roadmap towards achieving scalability in model driven engineering. In: Proceedings of the Workshop on Scalability in Model Driven Engineering. pp. 2:1–2:10. BigMDE '13, ACM (2013)
8. Micskei, Z., Konnerth, R., Horváth, B., Semeráth, O., Vörös, A., Varró, D.: On open source tools for behavioral modeling and analysis with fuml and alf. In: Proceedings of the 1st Workshop on Open Source Software for Model Driven Engineering (OSS4MDE@MoDELS 2014). CEUR Workshop Proceedings, vol. 1290, pp. 31–41 (2014)
9. Reder, A., Egyed, A.: Incremental consistency checking for complex design rules and larger model changes. In: Model Driven Engineering Languages and Systems, LNCS, vol. 7590, pp. 202–218. Springer Berlin Heidelberg (2012)
10. Ujhelyi, Z., Bergmann, G., Hegedüs, Á., Horváth, Á., Izsó, B., Ráth, I., Szatmári, Z., Varró, D.: EMF-IncQuery: An integrated development environment for live model queries. Sci. Comput. Program. 98, 80–99 (2015)
11. Varró, D., Bergmann, G., Hegedüs, A., Horváth, A., Ráth, I., Ujhelyi, Z.: Road to a reactive and incremental model transformation platform. Software and Systems Modeling (2016), in press