Incremental Queries and Transformations: From Concepts to Industrial Applications

Dániel Varró(✉)

Department of Measurement and Information Systems,
Budapest University of Technology and Economics,
Magyar tudósok krt. 2, Budapest 1117, Hungary
varro@mit.bme.hu

Abstract. Model-driven engineering (MDE) is widely used nowadays in the design of embedded systems, especially in the automotive, avionics or telecommunication domain. Behind the scenes, design and verification tools in these domains frequently exploit advanced model query and transformation techniques to support various rich tool features. The rapid increase in the size and complexity of system models has drawn significant attention to incremental model query and transformation approaches, which enable fast and incremental reactions to model changes caused by systems engineers or automated design steps. In this paper, I overview two open source Eclipse projects, EMF-IncQuery and Viatra, which have been actively used as a basis for developing various academic and industrial tools for critical systems.

Keywords: Model queries · Model transformations · Incremental evaluation · Reactive programming · Software tool qualification

1 Software Tools in Model-Based Systems Engineering

Model-driven engineering plays an increasingly important role in the design of critical embedded and cyber-physical systems in various application domains including automotive, avionics or telecommunication. Advanced design and verification tools aim to *simultaneously improve quality and decrease costs by early validation* to highlight conceptual design flaws well before traditional testing phases in accordance with the correct-by-construction principle. Furthermore, they improve productivity of engineers by automatically synthesizing different design artifacts (source code, configuration tables, test cases, fault trees, etc.) necessitated by certification standards (like DO-178C or ISO 26262).

There are two main trends nowadays in the software tool market of systems engineering. On the one hand, certain market shares are dominated by very few industrial tools (e.g. Matlab Simulink, Dymola, MagicDraw, DOORS) each of which typically supports a specific development stage (requirements engineering, simulation, allocation, test generation, etc.). In order to protect the important intellectual property rights, these tools are of closed nature, which implies such

R.M. Freivalds et al. (Eds.): SOFSEM 2016, LNCS 9587, pp. 51–59, 2016.
DOI: 10.1007/978-3-662-49192-8_5

huge tool integration costs for system integrators (like airframers or car manufacturers) that can easily exceed the total licensing costs of individual tools. On the other hand, recent initiatives (like PolarSys) have started to promote the development of open language standards and the systematic use of open source software components in tools for critical systems to reduce licensing costs and the risks of vendor lock-in.

When software tools are used for developing a critical system, the tools themselves need to be validated with the same scrutiny as the system under design by *software tool qualification*, especially, when no further human checking is carried out on the outputs of such tools. Software tool qualification distinguishes between *design tools* which, by definition, may introduce new errors to the system and *verification tools* which may fail to reveal existing errors of the system.

Unsurprisingly, software tool qualification is extremely costly due to the high algorithmic complexity, tightly couple architecture and unexpected feature interaction of such tools. In fact, most companies rather opt for using tools just as aids to highlight errors quickly and then they carry out the traditional verification& validation process with thorough simulation and testing. Anyhow, systematic software engineering techniques to simultaneously improve quality and reduce the costs of software tool qualification would be highly beneficial. Existing software engineering practices may guarantee the quality of the system itself, but they frequently *fail to ensure the quality of the software tool* used in systems engineering. Furthermore, the rapid increase in the size and complexity of systems models introduces significant *scalability challenges* for these tools.

Language engineering aims to provide foundations, techniques and tools for domain-specific modeling languages to capture the models. *Model transformation engineering* aims to systematically develop queries and transformations used in automated code generators, debuggers to process these models. Of course, a seamless integration of these techniques is needed when developing real tools.

In this paper, I overview two open source Eclipse projects supporting model query and transformation techniques integrated into in various industrial tools for model-based systems engineering. EMF-IncQuery is an incremental model query framework while Viatra supports reactive, event-based transformations. Their scientifically well-founded basis enables *semantic integration of different tool features* to (i) complement structural integration provided by the component (plugin) architecture of Eclipse and to (ii) support tool qualification by precise specification and execution semantics of those features.

2 Incremental Model Queries in EMF-IncQuery

EMF-IncQuery is an open source Eclipse project[1] to define declarative graph queries over EMF models [33] without manual coding and execute them efficiently using incremental graph pattern matching techniques over an imperative programming language such as Java. The benefits of EMF-IncQuery include:

[1] https://www.eclipse.org/incquery.

(i) a high-level and powerful declarative *graph query language* [8,39];
(ii) a highly efficient *incremental query engine* capable of evaluating queries over models with millions of elements [7,39];
(iii) an advanced *integrated development environment* [39] to construct and validate model queries supported by state-of-the-art Xtext tooling.
(iv) its modular architecture enables *easy integration with existing EMF-based modeling tools* [39].

The primary use case for model queries is to support the live validation of well-formedness constraints and design rules of a domain in order to highlight and report inconsistencies as soon as they are introduced. Efficient incremental evaluation is based on adapting Rete networks [13] to change notifications sent by EMF-based models. Additional main use cases include advanced support for incremental calculation and maintenance of base model indexers [39], derived features [26], soft traceability links [14], or incremental view maintenance [12].

Detailed scalability assessment of EMF-IncQuery is carried out in numerous papers for validation of well-formedness constraints [7,39], detection of source code anti-patterns [40] or maintenance of soft traceability links [14] over models with 10 million elements. Ongoing development within the MONDO European project[2] aims to develop a distributed and incremental query engine [30] deployed over cloud based storages to further improve scalability.

Example. The definition of a sample well-formedness constraint (taken from [20]) for checking valid allocations of application instances to host instances (e.g. in a cloud application or a cyber-physical system) is listed in Fig. 1. The query notAllocatedButRunning captures an erroneous situation for allocation when an application *app* is running, but not allocated to a host instance (using another graph pattern allocatedApplication by negative composition). When checking this constraint on the instance model depicted in Fig. 2, app2 is the only ApplicationInstance which matches the pattern (thus violates the constraint) since app1 is allocated to a host instance ht1 while app3 is stopped.

```
1 //EMF-IncQuery pattern in the query definition file
2 @Constraint(
3   key = {"app"}, severity = "error"
4   message = "$app.id$ is not allocated but it is running",
5 )
6 pattern notAllocatedButRunning(app : ApplicationInstance) {
7   ApplicationInstance.state(app, ::Running);
8   neg find allocatedApplication(app);
9 }
10
11 private pattern allocatedApplication(app : ApplicationInstance) {
12   ApplicationInstance.allocatedTo(app, _host);
13 }
```

Fig. 1. Sample queries for well-formedness constraints (adapted from [20])

[2] http://www.mondo-project.org/.

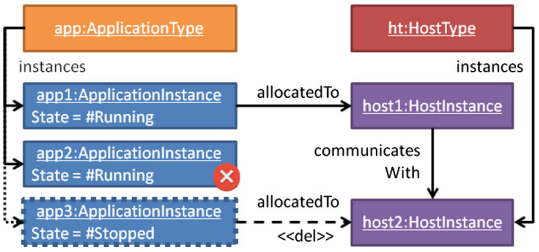

Fig. 2. Sample instance model

By using a @Constraint annotation, EMF-IncQuery will automatically integrate the query into a model editor using the Eclipse Modeling Framework (EMF) [33] as underlying model representation. As a result, an error marker will immediately be placed on the model whenever this consistency constraint is violated, which is removed automatically once the source of the problem is corrected (e.g. the application instance is stopped or allocated to a host instance).

3 VIATRA: A Reactive Transformation Platform

Viatra is a reactive, event-driven model transformation platform [6] where transformations are executed continuously as *reactions* to changes of the underlying model. The Viatra project[3] provides:

(1) An *internal domain-specific language* over Xtend [38] to specify both *batch and event-driven, reactive transformations*.
(2) A *complex event-processing engine* [11] over EMF models to specify reactions upon detecting complex chains of events.
(3) A *rule-based design space exploration* framework [4,15] to explore design candidates as models satisfying multiple criteria over states and trajectories.
(4) A *model obfuscator* to remove sensitive information from a confidential model (e.g. to create bug reports).

Viatra adopted the principles of reactive programming [5]. The core concept of reactive programming is *event-driven behavior*: components are connected to event sources and their behavior is determined by the event instances observed on *event streams*. Compared to sequential programming, the benefits of reactive programming are remarkable especially in cases when continuous interaction with the environment has to be maintained by the application based on external events without a priori knowledge on their sequence.

The *specification* of a Viatra transformation program contains (1) *rule specifications* consisting of *model queries*, which serve as a precondition to the transformation, and *actions*, which typically prescribe model manipulations. Furthermore, (2) *execution schemas* are defined in order to orchestrate the reactive

[3] http://www.eclipse.org/viatra/.

behavior. VIATRA uses an internal domain-specific language for specifying transformations, i.e. an advanced API over Java and Xtend [38]. Viatra has proven to be an efficient execution platform for incremental transformations in [20,24].

Example. A sample event-driven transformation rule (adapted from [20]) is illustrated in Fig. 3, which removes a stopped *ApplicationInstance* from the model if it is no longer allocated to a host instance. The execution of this rule is triggered by a disappearance of a match of its precondition pattern stoppedApplInstance. If the application instance is still stopped after the observed change, then we remove appInst from appType.

When executing the rule over the model of Fig. 2, the rule is triggered when the allocatedTo reference is removed between app3 and host2. Then the rule action removes app3 together with the incoming instances reference from app (illustrated by dotted lines in Fig. 2).

```
// Located query definition file .eiq
// Finds an application instance which is stopped and allocated to a host instance
pattern stoppedApplInstance(appInst: ApplicationInstance,
            appType:ApplicationType) {
  ApplicationType.instances(appType, appInst);
  ApplicationInstance.state(appInst, AppState::Stopped);
  ApplicationInstance.allocatedTo(appInst, host);
}
--------------------------------------------------------------------------------
// Located query definition file .xtend
val deleteApplInstanceRule = createRule().name("deleteApplInstance")
  .precondition(stoppedApplInstance) //a graph pattern as precondition
  .lifeCycle(ActivationLifecycles.default)
  //action to be executed when a pattern match gets lost
  .action(ActivationStates.DISAPPEARED) [
    // remove application instance from application type
    if (appInst.state == AppState::Stopped) {
      appType.remove(ApplicationType_Instances, appInst);
    }
  ].build
```

Fig. 3. A sample event-driven transformation rule

4 Selected Recent Applications

The EMF-INCQUERY and VIATRA frameworks have actively been used in different research and industrial projects carried out by various researchers and practitioners. Below we provide a short overview of selected applications of these frameworks within our own projects.

- A recent project aimed to define a model-driven approach and tool chain for the synthesis of complex, integrated Matlab Simulink models capable of simulating the software and hardware architecture of an airplane [14,16].
- As a bi-product of the project, the Massif (Matlab Simulink Integration Framework for Eclipse)[4] framework [16,19] was developed, which provides a bidirectional bridge between Matlab Simulink models and their EMF model counterpart by calling the Matlab API.

[4] https://github.com/FTSRG/massif/wiki.

- Formal validation of domain-specific languages is carried out in [29] by using back-end logic solvers where derived features and well-formedness constraints are captured by queries.
- Incremental queries and transformations provide foundations for incremental code generators [18] to avoid complete regeneration in case of small changes.
- Incremental recomputation of graphical views of Sirius [17] can be also be driven by reactive transformations.
- Live detection of human gestures and movements are carried out in [11] by using streaming transformations and complex event processing. Similar technology is used in ongoing work for runtime verification of cyber-physical systems and detecting critical situations in IoT applications [27].

In addition, EMF-IncQuery and/or Viatra is known to be *integrated into popular open source modeling tools* such as Papyrus UML [36], Capella [25], mbeddr [3], Sirius [37] or Artop [1].

5 Related Work

There are, of course, other open source technologies which support model queries or transformation used in Eclipse based tooling.

Query Technologies. EMF Model Query 2 [31] provides simple query primitives for selecting model elements that satisfy a set of conditions. The OCL development environment of the Eclipse OCL project [34] provides different ways to edit OCL constraints: an Xtext-based editor for file-based editing, an embedded editor inside Ecore model editors. The Epsilon Validation Language is dedicated to support the construction of validation rules within the Epsilon family [22], while the Acceleo Query Language (AQL) is heavily used within the Sirius project [37] to populate views from underlying models. However, relatively few academic approaches support incremental evaluation [10,28].

Transformation Technologies. The development environment of *EMF-based model transformation tools* provide support for specifying, executing and evaluation of transformations including frameworks such as ATL [32], Henshin [9], QVTo [35] or eMoflon [2]. Many industrial applications rely on Xtend [38] as a code generation and transformation language based on Java. Epsilon [22] provides the Epsilon Transformation Language and the low-level Epsilon Object Language with an advanced execution platform. Recently introduced new features of ATL include target incremental computation [21] combined into the ReactiveATL transformation engine.

6 Conclusions

While a multitude of design are verification tools is used in model-driven systems engineering of critical systems, the complexity of those tools is frequently comparable to the system under design. Certification standards of critical systems

necessitate to qualify those tools, i.e. to justify that the tools themselves do not introduce new errors to the design. The complexity of the tools makes tool qualification extremely costly, and provides a strong motivation for solid foundations of integrated tool features. The paper overviews two open source projects, EMF-IncQuery and Viatra to serve as a precise and efficient basis by incremental model queries and reactive transformations as illustrated on various industrial applications.

Our ongoing research and development aims to develop systematic approaches to the verification and validation of tool features and language specifications. This primarily includes the automated synthesis of a large, well-formed and diverse set of instance models to serve as test cases or scalability benchmarks. Furthermore, as distinction between design-time and run-time models are being more and more blurred [23] for smart cyber-physical systems, incremental query and transformation techniques will likely be used as part of the underlying middleware, which triggers further open challenges.

Acknowledgments. The author is indebted for the continuous and deep contributions of all contributors of the EMF-IncQuery and Viatra project teams. In particular, I would like to highlight the 8+ year involvement of Gábor Bergmann, Ábel Hegedüs, Ákos Horváth, István Ráth and Zoltán Ujhelyi (listed in alphabetic order).

This work was partially supported by the MONDO Project (EU ICT-611125) and the MTA-BME Lendület 2015 Research Group on Cyber-Physical Systems.

References

1. Artop: The AUTOSAR tool platform (2015). https://www.artop.org/
2. eMoflon (2015). http://www.moflon.org/
3. mbeddr (2015). https://mbeddr.com/
4. Abdeen, H., Varró, D., Sahraoui, H., Nagy, A.S., Hegedüs, Á., Horváth, Á., Debreceni, C.: Multi-objective optimization in rule-based design space exploration. In: 29th IEEE/ACM International Conference on Automated Software Engineering (ASE 2014), pp. 289–300. IEEE, Vasteras (2014)
5. Bainomugisha, E., Carreton, A.L., Cutsem, T.V., Mostinckx, S., Meuter, W.D.: A survey on reactive programming. In: ACM Computing Surveys (2012)
6. Bergmann, G., Dávid, I., Hegedüs, Á., Horváth, Á., Ráth, I., Ujhelyi, Z., Varró, D.: VIATRA 3: a reactive model transformation platform. In: Kolovos, D., Wimmer, M. (eds.) ICMT 2015. LNCS, vol. 9152, pp. 101–110. Springer, Heidelberg (2015). http://dx.doi.org/10.1007/978-3-319-21155-8_8
7. Bergmann, G., Horváth, A., Ráth, I., Varró, D., Balogh, A., Balogh, Z., Ökrös, A.: Incremental evaluation of model queries over EMF models. In: Petriu, D.C., Rouquette, N., Haugen, Ø. (eds.) MODELS 2010, Part I. LNCS, vol. 6394, pp. 76–90. Springer, Heidelberg (2010). http://dx.doi.org/10.1007/978-3-642-16145-2_6
8. Bergmann, G., Ujhelyi, Z., Ráth, I., Varró, D.: A graph query language for EMF models. In: Cabot, J., Visser, E. (eds.) ICMT 2011. LNCS, vol. 6707, pp. 167–182. Springer, Heidelberg (2011)
9. Biermann, E., Ermel, C., Taentzer, G.: Precise semantics of EMF model transformations by graph transformation. In: Czarnecki, K., Ober, I., Bruel, J.-M., Uhl, A.,

Völter, M. (eds.) MODELS 2008. LNCS, vol. 5301, pp. 53–67. Springer, Heidelberg (2008)
10. Cabot, J., Teniente, E.: Incremental integrity checking of UML/OCL conceptual schemas. J. Syst. Softw. **82**(9), 1459–1478 (2009)
11. Dávid, I., Ráth, I., Varró, D.: Streaming model transformations by complex event processing. In: Dingel, J., Schulte, W., Ramos, I., Abrahão, S., Insfran, E. (eds.) MODELS 2014. LNCS, vol. 8767, pp. 68–83. Springer, Heidelberg (2014). http://dx.doi.org/10.1007/978-3-319-11653-2_5
12. Debreceni, C., Horváth, A., Hegedüs, A., Ujhelyi, Z., Ráth, I., Varró, D.: Query-driven incremental synchronization of view models. In: 2nd Workshop on View-Based, Aspect-Oriented and Orthographic Software Modelling (VAO 2014), pp. 31:31–31:38. ACM (2014). http://doi.acm.org/10.1145/2631675.2631677
13. Forgy, C.L.: RETE: a fast algorithm for the many pattern/many object pattern match problem. Artif. Intell. **19**(1), 17–37 (1982)
14. Hegedüs, Á., Horváth, Á., Ráth, I., Starr, R.R., Varró, D.: Query-driven soft traceability links for models. Softw. Syst. Model. 1–24 (2014). http://dx.doi.org/10.1007/s10270-014-0436-y
15. Hegedüs, Á., Horváth, Á., Varró, D.: A model-driven framework for guided design space exploration. Autom. Softw. Eng. **22**(3), 399–436 (2015). http://dx.doi.org/10.1007/s10515-014-0163-1
16. 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 (2014)
17. Horváth, A., Ráth, I.: IncQuery gets Sirius: faster and better diagrams. In: EclipseCon Europe (2015). https://www.eclipsecon.org/europe2015/session/incquery-gets-sirius-faster-and-better-diagrams
18. Horváth, A., Ráth, I., Hegedüs, A., Balogh, A.: IoT supercharged: complex event processing for MQTT with eclipse technologies. In: EclipseCon France (2015). https://www.eclipsecon.org/france2015/session/decreasing-your-coffee-consumption-incremental-code-regeneration
19. Horváth, A., Ráth, I., Starr, R.R.: Massif - the love child of Matlab Simulink and Eclipse. In: EclipseCon NA (2015). https://www.eclipsecon.org/na2015/session/massif-love-child-matlab-simulink-and-eclipse
20. IncQuery Labs Ltd.: CPS Demonstrator: a model transformation benchmark (2015). https://github.com/IncQueryLabs/incquery-examples-cps/wiki/
21. Jouault, F., Tisi, M.: Towards incremental execution of ATL transformations. In: Tratt, L., Gogolla, M. (eds.) ICMT 2010. LNCS, vol. 6142, pp. 123–137. Springer, Heidelberg (2010)
22. Kolovos, D., Rose, L., Garcia-Domnguez, A., Paige, R.: The Epsilon Book (2015). http://www.eclipse.org/epsilon/doc/book/
23. Lee, E.A., Hartmann, B., Kubiatowicz, J., Rosing, T.S., Wawrzynek, J., Wessel, D., Rabaey, J.M., Pister, K., Sangiovanni-Vincentelli, A.L., Seshia, S.A., Blaauw, D., Dutta, P., Fu, K., Guestrin, C., Taskar, B., Jafari, R., Jones, D.L., Kumar, V., Mangharam, R., Pappas, G.J., Murray, R.M., Rowe, A.: The swarm at the edge of the cloud. IEEE Des. Test **31**(3), 8–20 (2014). http://dx.doi.org/10.1109/MDAT.2014.2314600
24. van Pinxten, J., Basten, T.: Motrusca: interactive model transformation use case repository. In: 7th Doctoral Symposium on Computer Science and Electronics, p. 57 (2014)
25. Polarsys: Capella (2015). https://www.polarsys.org/capella/

26. Ráth, I., Hegedüs, A., Varró, D.: Derived features for EMF by integrating advanced model queries. In: Vallecillo, A., Tolvanen, J.-P., Kindler, E., Störrle, H., Kolovos, D. (eds.) ECMFA 2012. LNCS, vol. 7349, pp. 102–117. Springer, Heidelberg (2012)
27. Ráth, I., Horváth, A.: IoT supercharged: complex event processing for MQTT with eclipse technologies. In: EclipseCon Europe (2015). https://www.eclipsecon.org/europe2015/session/iot-supercharged-complex-event-processing-mqtt-eclipse-technologies
28. Reder, A., Egyed, A.: Incremental consistency checking for complex design rules and larger model changes. In: France, R.B., Kazmeier, J., Breu, R., Atkinson, C. (eds.) MODELS 2012. LNCS, vol. 7590, pp. 202–218. Springer, Heidelberg (2012)
29. Semeráth, O., Barta, A., Horváth, Á., Szatmári, Z., Varró, D.: Formal validation of domain-specific languages with derived features and well-formedness constraints. Softw. Syst. Model. 1–36 (2015). http://dx.doi.org/10.1007/s10270-015-0485-x
30. Szárnyas, G., Izsó, B., Ráth, I., Harmath, D., Bergmann, G., Varró, D.: IncQuery-D: a distributed incremental model query framework in the cloud. In: Dingel, J., Schulte, W., Ramos, I., Abrahão, S., Insfran, E. (eds.) MODELS 2014. LNCS, vol. 8767, pp. 653–669. Springer, Heidelberg (2014)
31. The Eclipse Foundation: EMF Model Query 2 (2012). http://wiki.eclipse.org/EMF/Query2
32. The Eclipse Foundation: ATL (2015). http://www.eclipse.org/atl/
33. The Eclipse Foundation: EMF: The eclipse modeling framework (2015). http://www.eclipse.org/emf
34. The Eclipse Foundation: MDT OCL (2015). http://www.eclipse.org/modeling/mdt/?project=ocl
35. The Eclipse Foundation: Model to model project (2015). http://www.eclipse.org/m2m/
36. The Eclipse Foundation: Papyrus (2015). https://eclipse.org/papyrus/
37. The Eclipse Foundation: Sirius (2015). http://www.eclipse.com/sirius/
38. The Eclipse Foundation: Xtend (2015). http://www.eclipse.org/xtend
39. 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). http://dx.doi.org/10.1016/j.scico.2014.01.004
40. Ujhelyi, Z., Szoke, G., Horváth, Á., Csiszár, N.I., Vidács, L., Varró, D., Ferenc, R.: Performance comparison of query-based techniques for anti-pattern detection. Inf. Softw. Technol. **65**, 147–165 (2015). http://dx.doi.org/10.1016/j.infsof.2015.01.003