Model Repositories: Will they become reality?

A Position Statement

Francesco Basciani[1], Juri Di Rocco[1], Davide Di Ruscio[1], Alfonso Pierantonio[1] and Ludovico Iovino[2]

[1] DISIM, University of L'Aquila, Italy.
name.surname@univaq.it
[2] Gran Sasso Science Institute, L'Aquila, Italy.
ludovico.iovino@gssi.infn.it

Abstract. Over the last years, several repositories have been proposed in response to the need of the MDE community for advanced systems supporting the reuse of modeling artifacts, and the adoption of model management tools as software-as-service. Even though the potential benefits of MDE repositories are valuable, researchers and practitioners seem to prefer the management of their modeling artifacts locally and do not use yet advanced mechanisms for sharing and reusing them. This paper discusses the opportunities related to the adoption of model repositories and identify research issues that have to be addressed in order to make model repositories a reality in MDE.

1 Introduction

Over the last years, several model-driven platforms and tools are available to simplify and automate many steps of MDE approaches. Even though existing modeling tools provide developers and users with features able to simplify and automate many steps of model-based development processes, empirical studies show that barriers still exist making a wider adoption of MDE technologies difficult [14, 15]. In particular, to deal with the growing complexity of software systems, it is necessary to enforce consistent reuse and leverage the interconnection of the modeling artifacts that are produced and consumed during the different phases of the applied development processes.

An increasing demand for exchanging, cross-linking, analyzing software models, and coordinating actions of manifold stakeholders working on such models is leading to the urge for more advanced and integrated model repositories. Interestingly, repositories already play an important role in the Enterprise Architecture (EA) domain, of which the Zachman framework is a good example [13]. A crucial aspect in EA is the heterogeneity of artifacts ("model types"), which requires to leverage the use of more sophisticated artifact keeping-and-governing instruments.

Event though several initiatives have been promoted to collect modeling artifacts, the current support for discovering and reusing already developed models is very limited, and as a result, similar metamodels, transformations and other model management

programs often need to be developed independently from scratch. Consequently the up-front investment is raised and the productivity benefits of model-based processes are compromised.

In this paper, we consider successfully adopted repositories in different application domains, and we draw a research agenda related to a number of issues that in our opinion have to be necessarily solved in order to make model repositories in MDE a realistic option.

2 Why some repositories are already a reality?

In different application domains repositories are used to support peculiar research and development activities. Just to give representative examples we can mention BioModels Database[3] and CellML[4] in the biology domain, and GitHub[5] in the domain of software development. BioModels Database is a repository of computational models of biological processes. It hosts ≈200K models that are collected from literature and that are manually enriched with cross-references from external data resources (such as publications, ontologies, etc.). The repository allows scientific community to store, search and retrieve mathematical models. Additional services are provided such as online simulation, export of models into different formats, and the possibility to access the repository via web services. The CellML model repository manages ≈500 mathematical models of cellular biological function. Models are submitted to the repository by CellML users for distribution and reuse by the wider scientific community. GitHub is the largest code host with over 25.3 million repositories. Powerful tools (e.g., collaborative code review, intelligent issue tracking, powerful search, and useful analytics) are provided to support the development of software systems, which can be both open to the community or private.

The popularity of such repositories has been gained thank to the opportunities offered to their users: nobody would be interested in sharing artifacts without envisioning an added value in doing so. The repositories mentioned above represent sources of valuable know-how: the contained artifacts encode knowledge that can be used for learning and further development purposes. Thus multitudes of users sharing the same interests can work on the same artifacts in collaborative ways. Especially in complex and business-critical applications, the need for trusted artefacts is more pressing than ever. The availability of models and tools that are continuously enhanced and improved by community of users permit to obtain outcomes of higher quality. The lack of such repositories would reduce the possibilities of different teams to work together on problems that are commonly of interest.

The benefits related to the adoption of repositories have been acknowledged also in the MDE community. In fact, in the past decade several model repositories have been introduced (see [1, 3, 4, 7–12] just to mention a few). However, all of them seem to struggle in attracting contributions from the community. In our opinion this is due to

[3] `http://www.ebi.ac.uk/biomodels-main/`
[4] `https://models.cellml.org/cellml`
[5] `https://github.com/`

a number of factors that have to be investigated by researchers and practitioners in the forthcoming years as discussed in the next section.

3 Research agenda

In order to gain the advantages related to the adoption of model repositories in MDE, in our opinion there are many challenges that still have to be properly addressed [6] as discussed below.

Management of different kinds of modeling artifacts: most of the existing model repositories provide persistence options to models only. Consequently, reusability of other modeling artifacts and tools like modeling editors, model transformations, and code generators is not supported.

Advanced query mechanisms: advanced query techniques are of crucial relevance to retrieve artifacts according to different criteria. For instance, models can be searched by considering the corresponding metamodels, domain type, the particular development phase, by exploiting some tagging mechanisms, or even querying the repository (or a part of it) by means of logical predicates (e.g., OCL). Queries might consider also relations among different kinds of artifacts, e.g., to search for some modeling editor that can be used to produce models that can be automatically processed by some specific transformations available in the repository.

Model management and analysis tools as service (MaaS): modelling and model management tools are commonly distributed as software packages that need to be downloaded and installed on client machines, and often on top of complex software development IDEs (e.g., Eclipse). Given the non-trivial implicit and explicit dependencies of such tools, this can often be a burden, particularly for non-technical stakeholders (e.g., domain experts) with average IT skills. Consequently, it is necessary to have cloud-based installations of model repositories and give the possibility to remotely use the stored modeling artifacts over the net. In particular, the repository should provide APIs to programmatically and remotely adopt the previously deployed model management and analysis tools. This would permit to integrate such tools within other existing modeling environments and software systems.

Extension mechanisms: model repositories should provide core functionalities and extension mechanisms, which permit to build new applications in order to manage new kinds of modeling artifacts (e.g., conforming to different meta-metamodels or technical spaces) and to provide additional services. In this way, when new application requirements occur, the core services of the repository remain unaffected and they can be exploited to implement new functionalities able to address the new requirements.

Heterogeneity: model repositories should provide the means to enable the interoperability of model management tools possibly relying on different meta meta-models or even belonging to different technical spaces [5].

Scalability: it is an aspect, which is orthogonal to all the previous challenges. In particular, as MDE is increasingly applied to larger and more complex systems, current generation of model management technologies are reaching their limits and a new line of research is required in order to achieve scalability and to enable efficient management

and persistence of models larger than hundreds of megabytes in size, thus to cope with industry-scale artifacts.

Further than the previous challenges there are also non-technical ones that necessarily have to be met in order to enable a wider adoption of model repositories.

Incentives to share modeling artifacts: there are several public model repositories around, however keeping them alive and solicit contributions from user communities is a hard task especially because each business entity might not see any benefit from sharing their own developed artifacts. Consequently, it is necessary to conceive rewarding mechanisms that can motivate users to share their artifacts. For instance, providing additional services like remote validation of modeling artifacts, automated chaining of model transformations and their remote execution, might represent examples of added services that could motivate users to upload their internally developed artifacts and to take advantage of them.

Licensing related to the shared artifacts: it is necessary to manage the intellectual property of the shared artifacts. Similarly to what occurs in the domain of open source software, it is necessary to identify, assess, enforce, and inferring licensing schemes under which modeling artifacts are uploaded and maintained in model repositories.

Guidelines to manage the sharing of artifacts and to keep their quality under control: users that would like to reuse artifacts available in repositories should be somehow guaranteed that the available artifacts satisfy quality requirements that has to be defined for each kind of modeling artifacts. To this end, the sharing phase has to be moderated in a way that users upload their artifacts, which are analyzed and tested before making them available to other users. This process resembles what already happens in commonly used stores of mobile applications.

Federation of model repositories: Having only one public repository would be very difficult to achieve for different reasons like e.g., intellectual property and licensing issues. Instead, a more realistic scenario would consist of multitudes of model repositories that interact each with others according to project or organization driven policies. Thus, each organization might have its private repositories able to interact with public ones in order to retrieve modeling artifacts that are publicly available. Consequently, federation mechanisms have to be properly conceived in order to seamlessly aggregate modeling artifacts with respect to specific access and licensing policies and to ensure business entities that they would remain the owner of the produced modeling artifacts while still benefiting the use of other model repositories.

4 Conclusion

In different application domains like biology and in source code development, repositories are already a reality and they are continuously used to share, learn, reuse, and improve artifacts. Even though several initiatives have been promoted to support the adoption of model repositories in MDE, we are still far from a concrete adoption of them. As discussed in the paper, in our opinion this is due to a number of issues which represent interesting research themes for the next few years. To this end the authors

have recently started the MDEForge project[6] [2], with the aim of addressing the problems outlined in the previous section and also discussed in [6].

Acknowledgment

This research was supported by the EU through the Model-Based Social Learning for Public Administrations (Learn Pad) FP7 project (619583).

References

1. The Open Models Initiative. `http://www.openmodels.at/`. Last visited on September 2015.
2. F. Basciani, J. Di Rocco, D. Di Ruscio, A. Di Salle, L. Iovino, and A. Pierantonio. MDEForge: an Extensible Web-Based Modeling Platform. In *Procs of CloudMDE@MoDELS 2014, Valencia, Spain, September 30, 2014.*, pages 66–75, 2014.
3. R. Breu, B. Agreiter, M. Farwick, M. Felderer, M. Hafner, and F. Innerhofer-Oberperfler. Living models - ten principles for change-driven software engineering. *Int. J. Software and Informatics*, 5(1-2):267–290, 2011.
4. P. Brosch, P. Langer, M. Seidl, and M. Wimmer. Towards end-user adaptable model versioning: The by-example operation recorder. In *Procs.of CVSM '09*, pages 55–60, Washington, DC, USA, 2009. IEEE Computer Society.
5. J. Bzivin. Model driven engineering: An emerging technical space. In R. Lmmel, J. Saraiva, and J. Visser, editors, *Generative and Transformational Techniques in Software Engineering*, volume 4143 of *Lecture Notes in Computer Science*, pages 36–64. Springer Berlin Heidelberg, 2006.
6. J. Di Rocco, D. Di Ruscio, L. Iovino, and A. Pierantonio. Collaborative repositories in Model-Driven Engineering. *IEEE Software*, 32(3):28–34, May 2015.
7. R. B. France, J. M. Bieman, S. P. Mandalaparty, B. H. C. Cheng, and A. Jensen. Repository for Model Driven Development (ReMoDD). In *2012 34th International Conference on Software Engineering (ICSE)*, pages 1471–1472. IEEE, 2012.
8. C. Hein, T. Ritter, and M. Wagner. Model-driven tool integration with ModelBus. *Workshop Future Trends of Model-Driven*, 2009.
9. T. Holmes, U. Zdun, and S. Dustdar. Automating the Management and Versioning of Service Models at Runtime to Support Service Monitoring. In *EDOC*, pages 211–218, Sept. 2012.
10. B. Karasneh and M. R. V. Chaudron. Online img2uml repository: An online repository for UML models. In *Procs of the 3rd International Workshop on Experiences and Empirical Studies in Software Modeling - MoDELS 2013*, pages 61–66, 2013.
11. M. Koegel and J. Helming. EMFStore: a model repository for EMF models. In *Software Engineering, 2010 ACM/IEEE 32nd Int. Conf. on*, volume 2, pages 307–308, May 2010.
12. R. Kutsche, N. Milanovic, G. Bauhoff, T. Baum, M. Cartsburg, D. Kumpe, and J. Widiker. BIZYCLE: Model-based Interoperability Platform for Software and Data Integration. In *Procs.of the MDTPI at ECMDA*, 2008.
13. J. F. Sowa and J. A. Zachman. Extending and formalizing the framework for information systems architecture. *IBM Syst. J.*, 31(3):590–616, June 1992.
14. J. Whittle, J. Hutchinson, and M. Rouncefield. The state of practice in model-driven engineering. *Software, IEEE*, 31(3):79–85, May 2014.

[6] MDEForge: `http://www.mdeforge.org`

15. J. Whittle, J. Hutchinson, M. Rouncefield, H. Burden, and R. Heldal. Industrial adoption of model-driven engineering: Are the tools really the problem? In *MODELS*, volume 8107 of *LNCS*, pages 1–17. Springer Berlin Heidelberg, 2013.