Emergent Model-Driven Engineering Ecosystems by means of Patterns

Antonio Cicchetti[1]

Mälardalen University, IDT,
Västerås, Sweden,
antonio.cicchetti@mdh.se

Abstract. In Model-Driven Engineering (MDE) design of modelling languages and model transformations is still an expert's task. Domain experts, i.e. the stakeholders of languages and transformations, would like to independently define and use their own MDE ecosystem, but can only support those activities.
In this position paper we discuss this problem from a more general perspective, arguing for the need of emergent MDE, that is modelling languages and transformations shall be inferred from the usage it is done of modelling concepts. Moreover, the paper outlines a possible research agenda with corresponding challenges towards the goal of emergent MDE ecosystems.

Keywords: bottom-up (meta-)modelling, modelling languages, model transformation, emergent languages, emergent transformations, modelling patterns, MDE ecosystems

1 On the need for emergent MDE

Model-Driven Engineering [1] techniques have done remarkable progresses since the dawn of software modelling methodologies. Thanks to generative approaches it is possible to derive a whole language editor based on a metamodel definition, add views, create concrete syntaxes, and so forth [2]. Even more advanced, recently a whole set of by-example mechanisms have been proposed to automatically infer modelling languages and model transformations starting from a set of exemplary models or mappings, respectively [3]. However domain-experts, possibly agnostic of MDE, still find it very difficult to define their own language or necessary transformations. Additionally, current MDE frameworks expose little flexibility, meaning that once a metamodel is defined, any need for changes has as side effects changes to the existing models and transformations complying to the old version of the metamodel [4]. Also in this respect, recent proposals have tried to enhance frameworks malleability, notably by allowing the handling of partial models or trying to separate linguistic aspects from ontological (i.e., domain-specific) aspects [5, 6].

In all the available solutions, including by-example mechanisms, the support for MDE is supposed to be top-down, that is a domain is studied, and then metamodels and model transformations are created correspondingly. In other words, the creation of an MDE framework is conceived as a bootstrap activity, preliminary of any application design [7]. This kind of approach leaves MDE mechanisms vulnerable to evolutionary pressures: in fact, very often metamodels and model transformations become monolithic, meaning that refinements and extensions are difficult to manage especially when involving some form of semantic reconsideration [4]. Such a weakness is exacerbated in industrial contexts: to create a MDE framework experts have to rely on their own personal interpretation of modelling requirements, which in turn are gathered from a selected (group of) domain-expert(s) that might not be representative of the general case, or might not be skilled enough to foresee needs for refinements and extensions [3]. As a consequence, frameworks are perceived as scarcely flexible, usable, and integrable with the existing development processes [8].

In this paper, we argue that the support for MDE should be reconsidered to enable also *emergent* approaches. More precisely, languages and transformations should incrementally inferred as the result of modelling needs and uses rather than being fixed a-priori. Therefore, an ecosystem grows and is refined in order to satisfy the modelling needs conveyed by users' usage. Emergent ecosystems would be naturally agile and flexible, since they would include all and only the needed features, as captured and encoded through usage patterns. Admittedly, this paradigm shift does not come for free: in the remainder of this work, first a discussion on the concepts of syntax, semantics, and ontology is proposed, and then a research agenda together with expected challenges and possible solution is illustrated.

2 Levering ontology information in MDE

In general, modelling activities have to deal with the interplay of syntax, semantics, and ontology. The boundaries between the three are quite blurry. The abstract syntax of a language, i.e. its metamodel, is supposed to be – indeed – only syntax. However, well-formedness constraints do include domain-specific information (e.g., types or multiplicities). With respect to semantics, it is usually distinguished in structural and behavioural [9]: the former describes a language in terms of its model instances, while the latter defines how the state of a model evolves over time. In both case, semantics relies on mappings between syntactical structures [10]. The concrete syntax of a language is a special case: it was intended to be *syntactic sugar*, analogously to programming languages, to ease the task of modellers and hence avoid them to deal with verbose, difficult to read, metamodelling structures. Moreover, diagrammatic representations, i.e. a combination of textual and graphical objects, are usually very effective in supporting communication and information exchange [11]. However, abstractions in general have to deal with ontology. More precisely, the rendering of syntactic structures is naturally prone to multiple interpretations.

In software engineering ontological information is typically seen as a uniquely defined representation of a certain domain for reasoning purposes. On the contrary, other fields of science as philosophy consider ontology as agent-based, that is strictly related to the observer [12]. Going into more details, depending on the amount of knowledge about a certain domain, different agents, or modelling stakeholders in our case, would interpret the same information in different ways. As an example, a sub-atomic physician would have a quite different perception of a rainbow if compared to someone who is completely agnostic about physics. This is also the cause of the intrinsic ambiguity of abstractions, because some details relevant for the former observer would be totally irrelevant for the latter, respectively [7].

By looking to the current MDE techniques, what is missing is a solid support of ontological concerns and the customization demands they introduce. Languages and transformations specification facilities should be aware of agent-based ontology and allow adaptability to the different degrees of knowledge a stakeholder might have. Therefore, ontology represents the main factor to be taken into account for customization purposes. This concept is instantiated quite naturally in the form of views, i.e. given a certain language different experts might need different information depending on their perspective. Moreover, consistency issues should be addressed as limited to the information available to each particular view.

An appropriate support for emergent MDE ecosystems should be able to infer languages and transformations as derivable from users' usage patterns, and hence relieve stakeholders from customization efforts. The ecosystem should be defined in a user-specific ontological layer, over the base syntax and semantics specifications, and not as proper new languages and transformations [13].

3 Towards emergent MDE: a research agenda

As discussed so far, in this paper we propose to tackle flexibility and usability issues of MDE ecosystems by adopting an induction strategy [3] and exploiting ontological information for the customization. This solution poses several challenges, which are discussed in the following together with available solutions and open problems.

3.1 The base knowledge: i) MDE repositories, ii) composition operators, iii) common semantics.

An emergent approach relies on the existence of base/concrete knowledge that can be used to create further abstractions. One option to store such knowledge is to rely on a repository of existing languages and transformations, and consequently to create a *lattice* of relationships between domain-specific languages [14]. Graph databases are a suitable support to manage these mechanisms since they rely on nodes (portions of languages) and edges (relationships between languages); moreover, they naturally allow pattern-based reasoning about the information they store. In this scenario, the major problems would be due to the creation of operators for (sub-)languages composition, and in particular for semantics, that is not guaranteed to be compositional [15, 16]. In this respect, interesting solutions enable re-use of metamodels and transformations portions through adaptation [5, 17]. Moreover, model matching engines and by-example techniques can be adapted to support the navigation of the graph and the collection/elicitation of candidate usage patterns. Whereas, the definition and adoption of a suitable semantics is a highly debated and controversial problem in MDE, and therefore represents an open issue for emergent mechanisms.

3.2 The ontological layer: iv) ontology-based matching of modeling patterns, v) multilevel transformations, vi) base knowledge evolution and ontology layer co-evolution.

The ontological layer is meant to support user-tailored customizations, and is incrementally extended and refined depending on the patterns traversed by a certain user. In particular, users can list a personal set of concepts and relationships that have to be matched in the repository. Once the set is considered satisfactory, a user has the option of renaming elements and relationships between them, and to define a custom concrete syntax. As previously discussed, these aspects pertain to the user ontology, i.e. the way in which a certain observer perceives the reality, and therefore do not change the base knowledge. Technically, these layers can be implemented by means of multi-level modelling solutions, which support multiple instantiation layers: existing works already deal with creating ontological layers of metamodels, including the definition of concrete syntaxes [5]. However, incremental refinements of ontologies as well as transformation definitions at the ontological level need further investigations.

Another major concern is represented by possible changes in the base knowledge and their propagation to existing ontological layers [12]. In fact, there shall be a way to assess if changes are breaking a certain ontological layer, both in terms of model entities and transformations. Moreover, it should be possible to understand if the chosen concrete syntax might be affected by the changes. This issue shares a lot of open research problems with metamodel evolution and the co-evolution of related artefacts [4].

4 Conclusions

This position paper discusses the problems related to MDE ecosystems usability and flexibility and proposes a research direction aiming at realizing emergent customization mechanisms. We advocate the definition of a clearer distinction between syntax, semantics, and ontological aspects of modelling and illustrate a research agenda towards emergent MDE. It is interesting to notice that most of the intricacies discussed in the research agenda raise also in the development of cyber-physical systems (CPSs): as a matter of fact, their modelling demands the integration of multi-paradigm modelling techniques, different levels of abstractions, and disparate semantics. Available solutions tend to create tool chains in which a shared semantics, typically a model of computation, is exploited as a common denominator devoted to consistency management and synthesis of the resulting integration. The repository mechanism mentioned in this work can be conceived as a generalization of modelling solutions adopted for CPSs, and the adoption of models of computation as shared semantics is an interesting investigation direction [6, 18].

Acknowledgements

The author would like to thank Federico Ciccozzi and Alfonso Pierantonio for the interesting discussions on the topics covered in this work.

References

1. Schmidt, D.C.: Guest editor's introduction: Model-driven engineering. Computer **39**(2) (February 2006) 25–31
2. Kolovos, D., Rose, L., Paige, R.: The epsilon languages and tools framework (2013)
3. López-Fernández, J.J., Cuadrado, J.S., Guerra, E., Lara, J.: Example-driven metamodel development. Softw. Syst. Model. **14**(4) (October 2015) 1323–1347
4. Di Ruscio, D., Iovino, L., Pierantonio, A.: Evolutionary togetherness: How to manage coupled evolution in metamodeling ecosystems. In: Proceedings of the 6th International Conference on Graph Transformations. ICGT'12, Berlin, Heidelberg, Springer-Verlag (2012) 20–37
5. de Lara, J., Guerra, E.: Generic meta-modelling with concepts, templates and mixin layers. In: Proceedings of the 13th International Conference on Model Driven Engineering Languages and Systems: Part I. MODELS'10, Berlin, Heidelberg, Springer-Verlag (2010) 16–30
6. Degueule, T., Combemale, B., Blouin, A., Barais, O., Jézéquel, J.M.: Melange: A meta-language for modular and reusable development of dsls. In: Proceedings of the 2015 ACM SIGPLAN International Conference on Software Language Engineering. SLE 2015, New York, NY, USA, ACM (2015) 25–36
7. Salay, R., Chechik, M.: Supporting agility in mde through modeling language relaxation. In Juan de Lara, Davide Di Ruscio, A.P., ed.: XM 2013 Extreme Modeling Workshop Proceedings at MoDELS, September 29, 2013 Miami, Florida (USA). (2013) 20–27
8. Whittle, J., Hutchinson, J., Rouncefield, M., Burden, H., Heldal, R.: Industrial Adoption of Model-Driven Engineering: Are the Tools Really the Problem? In: Model-Driven Engineering Languages and Systems: 16th International Conference, MODELS 2013, Miami, FL, USA, September 29 – October 4, 2013. Proceedings. Springer Berlin Heidelberg, Berlin, Heidelberg (2013) 1–17
9. Chen, K., Sztipanovits, J., Abdelwalhed, S., Jackson, E.: Semantic anchoring with model transformations. In: Proceedings of the First European Conference on Model Driven Architecture: Foundations and Applications. ECMDA-FA'05, Berlin, Heidelberg, Springer-Verlag (2005) 115–129
10. Harel, D., Rumpe, B.: Meaningful modeling: What's the semantics of "semantics"? Computer **37**(10) (October 2004) 64–72
11. Johnson, G., Gross, M.D., Hong, J., Yi-Luen Do, E.: Computational support for sketching in design: A review. Found. Trends Hum.-Comput. Interact. **2**(1) (January 2009) 1–93

12. Ermolayev, V., Davidovsky, M.: Agent-based ontology alignment: Basics, applications, theoretical foundations, and demonstration. In: Proceedings of the 2Nd International Conference on Web Intelligence, Mining and Semantics. WIMS '12, New York, NY, USA, ACM (2012) 3:1–3:12
13. Izquierdo, J.L.C., Cabot, J., López-Fernández, J.J., Cuadrado, J.S., Guerra, E., Lara, J.: Engaging End-Users in the Collaborative Development of Domain-Specific Modelling Languages. In: Cooperative Design, Visualization, and Engineering: 10th International Conference, CDVE 2013, Alcudia, Mallorca, Spain, September 22-25, 2013. Proceedings. Springer Berlin Heidelberg, Berlin, Heidelberg (2013) 101–110
14. Hebig, R., Seibel, A., Giese, H.: On the Unification of Megamodels. In Amaral, V., Vangheluwe, H., Hardebolle, C., Lengyel, L., Magaria, T., Padberg, J., Taentzer, G., eds.: Proceedings of the 4th International Workshop on Multi-Paradigm Modeling (MPM 2010). Volume 42 of Electronic Communications of the EASST. (2011)
15. Jézéquel, J.M., Combemale, B., Barais, O., Monperrus, M., Fouquet, F.: Mashup of metalanguages and its implementation in the kermeta language workbench. Softw. Syst. Model. **14**(2) (May 2015) 905–920
16. Vallecillo, A.: On the combination of domain specific modeling languages. In: Proceedings of the 6th European Conference on Modelling Foundations and Applications. ECMFA'10, Berlin, Heidelberg, Springer-Verlag (2010) 305–320
17. Basciani, F., Di Ruscio, D., Iovino, L., Pierantonio, A.: Automated Chaining of Model Transformations with Incompatible Metamodels. In: Model-Driven Engineering Languages and Systems: 17th International Conference, MODELS 2014, Valencia, Spain, September 28 – October 3, 2014. Proceedings. Springer International Publishing (2014) 602–618
18. Bryant, B.R., Gray, J., Mernik, M., Clarke, P.J., France, R.B., Karsai, G.: Challenges and directions in formalizing the semantics of modeling languages. Comput. Sci. Inf. Syst. **8**(2) (2011) 225–253