# Constructing Scalable Domain-Specific Graphical Modelling Languages

Antonio Garmendia

Universidad Autónoma de Madrid (Spain)
antonio.garmendia@uam.es

**Abstract.** The adoption of Model-Driven Engineering (MDE) as software development paradigm, is recently growing. The creation of Domain Specific Modelling Languages (DSMLs) is a frequent task, because it has several benefits to describe a particular domain. However, the construction of graphical DSMLs is a technically challenging task, and the generated environments do not scale well for large models.
In order to improve this situation, in this thesis, we propose borrowing modularity concepts from programming languages. In this manner, DSMLs will be enriched with fragmentation strategies, which will produce environments able to seamlessly partition models across folders and files. To facilitate the definition of the graphical syntax, we propose a guided, heuristics-based approach, which is capable to profit from the defined fragmentation strategy.

**Keywords:** Domain-Specific Modelling Languages, Graphical Modelling Environments, Scalable Modelling, Meta-modelling, Modularity.

## 1 Introduction and Problem Statement

The Model-Driven Engineering (MDE) paradigm, proposes software development by the use of models of higher level of abstraction than code [13]. Hence, models are used to automate many activities, like code generation, system simulation or testing. One of the important concepts within MDE are Domain-Specific Modelling Languages (DSMLs), which focus on modelling using primitives of a particular domain.

Current applications become complex and MDE aims to reduce production costs. However, while models have a higher level of abstraction than code, for large systems, they may become large and unwieldy as well. There are different tools that focus on MDE, being Eclipse Modelling Framework (EMF) [14], a widespread technology and with a set of compatible plug-ins to facilitate the creation of DSMLs. The existing technology, including EMF, does not have appropriate mechanisms for fragmenting models, which are generally monolithic, making them heavier and difficult to process for tools and understanding for people.

With respect to models, there is an absence of a system that facilitates the modularity, therefore, the composition, extension and re-utilization, becomes difficult. Eclipse JDT for example, basically proposes the organization of Java projects in packages, fragments and compilation units. I propose to organise a model like a programming project, fragmenting it into folders and files. Once the fragmentation is applied, there is a need to define scoping rules for references, visibility rules for model elements and mechanisms for efficient check of OCL constraints. The modularity in models would bring as a benefit, the creation of libraries of abstractions, that provide standard solutions to already resolved problems.

A further difficulty is the definition of graphical editors for DSMLs. Graphical Frameworks do not scale well and do not support scalability mechanisms, just barely propose layers and hierarchical drill down, to describe the different levels of the model. Thus, I propose a guided, heuristics-based approach, to help users with the definition of the graphical syntax and afterwards the generation of these environments. Accordingly, the generated editor will be capable to profit from the defined fragmentation strategy.

The proposal is to establish modularity mechanisms for DSMLs, which provide scalability for DSMLs. In addition, facilitate the generation of graphical modelling environments that supports this modularity.

## 2 Related Work

This research attempts to provide scalability through modularity techniques to build modelling environments for DSMLs. Thereby, we are going to focus on works dealing with the need of process large models. In addition to this, we will review the use of frameworks for the generation of graphical editors. Our approach consists in generating structured model editors from meta-models and speed up the generation of graphical environments through patterns and wizards.

There are some works that use different techniques to process large models and obtain some performance gains. Among them, the work of Scheidgen and Zubow [12] which proposes a persistence framework that allows automatic and transparent fragmentation to create, update, traverse and query models, based on EMF framework. Another example is Morsa [5], which makes use of a NoSQL database to provide scalable access to large models. Finally, the most mature technology is CDO [4], that is a repository of EMF models, which needs to preprocess meta-models in order to persist their instances. The two works described above, compared their technology with CDO, finding scalability issues from the latter.

On the other hand, some works decompose models into sub-models for enhancing their comprehensibility. In [15] Strüber et al. describe a tool which use Information Retrieval Algorithms for model splitting and an engine to obtain several sub-models. Other example is described by Kelsen et al. [10], providing a mathematical description of linear-time algorithm for construction the decomposition of models, but there is no tool support.

There are a collection of tools that propose MDE solutions for the development of graphical editors compatible with EMF. There are some Eclipse plug-ins

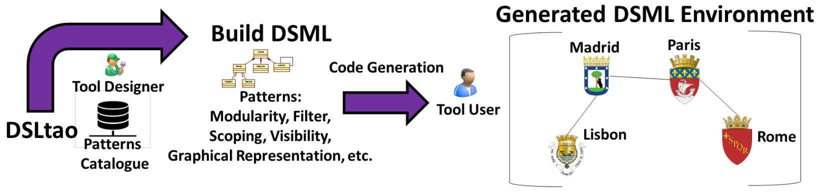
DSLtao
Tool Designer
Patterns Catalogue
Build DSML
Patterns: Modularity, Filter, Scoping, Visibility, Graphical Representation, etc.
Code Generation
Tool User
Generated DSML Environment
Madrid
Paris
Lisbon
Rome

Fig. 1: General scheme for the semi-automatic generation of the modelling environment.

to construct diagram editors, such as: GMF [1] , Eugenia[2], Graphiti[3] and Sirius [4]. Graphiti provides a Java API for coding, the other frameworks are model-based. Sirius, is the graphical modelling framework that is currently becoming popular in the MDE community. However, we need some expert knowledge to use these frameworks, sometimes they require the configuration of complex environment specifications. Besides that, the constructed editors do not scale well, because it is based on build monolithic models. Our approach is to generate the scalable environment with the required settings.

## 3 Proposed solution

We are going to base our tool in DSLtao, which provides an environment to design DSMLs through patterns. DSLtao proposes a catalogue of patterns divided into domain, design, concrete syntax, dynamic semantics and infrastructure patterns [11]. These patterns may include services which can contribute to the functionality of the generated environment. For this thesis, we are going to provide a number of patterns and services, to create scalable environments for graphical DSMLs.

We plan to create the environment for DSMLs according to the scheme in Fig. 1. Our tool will take as input a meta-model, with instances of patterns, such as: Modularity, Filter, Scoping, Visibility and its corresponding graphical representation. The interaction of the DSML designer with the tool, would be through wizards, that facilitate the creation of diagram editors, proposing efficient and flexible schemes that may be adapted to different problems. After applying the patterns to the DSML meta-model, the modelling environment will be automatically generated.

In terms of fragmentation, the modularity mechanism that we currently support [6] fragments models across composition relationships between classes. To make the approach more flexible, we are also planning to provide some fragmentation strategy which does not depend only on composition relationships. We are planning to apply graph theory based cluster algorithms to models, and also some heuristics to detect that the models need to be split due to their large size

[1] `https://wiki.eclipse.org/Graphical_Modeling_Framework`
[2] `http://eclipse.org/epsilon/doc/eugenia/`
[3] `http://eclipse.org/graphiti/`
[4] `https://www.eclipse.org/sirius/`

[16]. With this approach, we can provide fragmentation strategies which do not depend on composition and therefore a non-hierarchical fragmentation can be carried out.

Using this modular organization, we plan to add scoping mechanism, to allow objects being visible or not, between model elements that belong to different locations in the file system or satisfy certain query. We intend to use Epsilon Object Language (EOL) [5] and Hawk [1], to query the fragmented models and obtain the elements efficiently.

In the case of graphical syntax, we propose a wizard to define the concrete syntax. Here, we are going to develop a dedicated wizard to apply the graphical pattern over the elements of the DSMLs meta-model. This wizard will contain some heuristics, that generate a graphical representation model, from which the environment is synthesized, using different graphical editor frameworks. Using this model, we can improve the cognitive effectiveness, using Moody's principles for the evaluation of graphical notations [9].

Finally, we are going to do some research in Model Transformations (MTs). Taking advantage that the model is fragmented, it could be possible to put forward a formal description which explains how we can take advantage of the splitted model and improve the performance of MTs, perhaps in connection with distributed models for transformation execution [2].

## 4 Preliminary Work

We have developed *EMF Splitter* [6], which works both as an independent Eclipse plug-in, and as a plug-in of DSLtao. *EMF Splitter* supports both, the definition of a modular structure for EMF models [6] and the definition of their graphical syntax relying on Sirius [8]. We made some initial research by defining modularity mechanisms for meta-models developed by us and third parties, and by generating graphical environments in the context of the MONDO EU project [7].

First of all, we define a modular structure based on how Eclipse JDT organized projects. This Modularity Pattern has three main concepts *Project*, *Package* and *Unit*, from which we generate an Eclipse plug-in that enables the editing of models according to this structure [6]. The model content can be organized in folders and files, being possible the composition of the parts to build a monolithic one. The generated plug-in also provides the decomposition of the model, according to the selected fragmentation strategy.

Additionally, we explored how to combine fragmentation strategies with visualization mechanisms. The feasibility of this combination is confirmed based on an evaluation over a synthetic models, and the model sets of the GraBaTs'09 contest. In [7], we applied fragmentation strategies to large models to obtain more manageable chunks. The exploration is done by applying different abstraction strategies to visualize big models in the forms of graphs.

[5] `http://www.eclipse.org/epsilon/doc/eol/`
[6] `http://antoniogarmendia.github.io/EMF-Splitter`
[7] `http://www.mondo-project.org/`

The construction of modelling environments to create and edit models, is usually an ad-hoc, technically difficult process. In [8], we proposed the generation of modelling environment using a wizard to define the concrete syntax and to automate the process. We have a platform independent GraphicalRepresentation meta-model and then this representation is transformed into a technology-specific editor. Currently we support Sirius, but in the future we plan to support other graphical frameworks. In addition to this, we have decided to support manual modification of the generated editor, trying to avoid overriding the manual changes.

## 5 Expected Contribution

We will provide developers with advanced tools, to facilitate the work with large models, facilitating distributed development, through division of models in more manageable chunks.

We are also implementing a pattern based approach for the automatic generation of graphical DSMLs. We are currently working on new features and developing further services like layers and abstractions. In the future, we hope to contribute to the generation of advanced graphical and textual environments. In addition, the approach could be used to the modernization of existing editors. In this moment, the generation of editors is very costly for developers. For that reason, the creation of the wizard to define the concrete syntax aims to reduce this effort.

In this research we will generate the plug-ins that produce the environment using the defined patterns. These patterns may be applied in combination, then the developer can reuse the functionality implemented. Our goal is to generate scalable modelling environments.

## 6 Plan for Evaluation and Validation

We plan some experiments to test the performance of the generated tools and also validate the construction process. We already made some test with the GraBaTs'09 models, in which we evaluate the performance of model fragmentation. In the future, we plan to choose other case studies like the Knowledge Discovery Metamodel (KDM) [8] models, which provides a representation related to the existing software assets. The companies use these technology to understand the functionality and the data of their legacy systems [3]. With our approach, these models will not be monolithic. Instead, our framework will be able to decompose them, into folders and files, to facilitate their edition, navigation and comprehension.

One of the main purpose of this approach is to make easier the tasks to developers. For that reason, it is necessary to test the tool with users have some experience in the creation of environments for DSMLs. We will evaluate the

[8] `https://www.eclipse.org/gmt/modisco/infrastructure/KDM/`

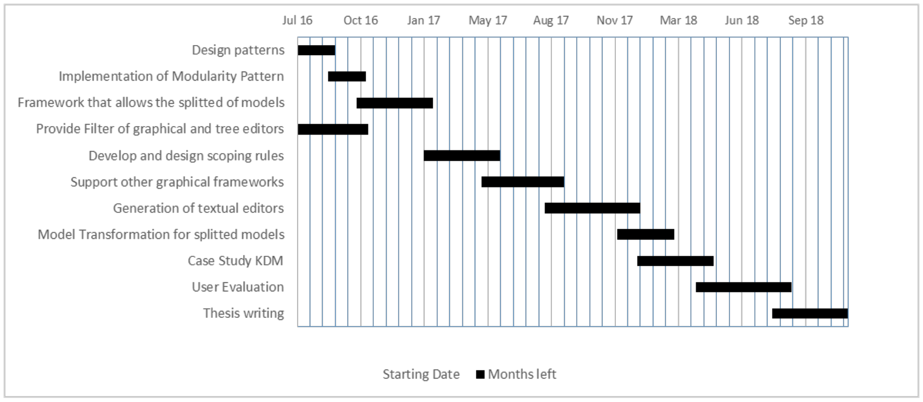

Fig. 2: Gantt Diagram-Research Schedule.

process performing a case study and additionally a test to assess the quality of our tool. For instance, the wizard for defining the concrete syntax has been partially assessed in the MONDO project.

## 7 Current Status

At this moment, we are developing *EMF Splitter* that enables developers to build models in a structured way. We are in an early stage of development, but already we made some experiments with large models and how to visually explore them. We also provide a wizard for a semi-automatic generation of graphical editors. Currently, we only support Sirius.

We aim to continue working on this tool, to provide more advanced features in the generation of modelling environments using patterns. The tasks are summarized in Figure 2, with an approximation of the estimated time. We plan to finish the thesis by the end of 2018.

## References

1. K. Barmpis, S. Shah, and D. S. Kolovos. *Towards Incremental Updates in Large-Scale Model Indexes*, pages 137–153. Springer International Publishing, Cham, 2015.
2. A. Benelallam, A. Gómez, M. Tisi, and J. Cabot. Distributed model-to-model transformation with ATL on mapreduce. In *Proceedings of the 2015 ACM SIGPLAN International Conference on Software Language Engineering, SLE 2015, Pittsburgh, PA, USA, October 25-27, 2015*, pages 37–48, 2015.
3. H. Bruneliere, J. Cabot, F. Jouault, and F. Madiot. Modisco: A generic and extensible framework for model driven reverse engineering. In *Proceedings of the IEEE/ACM International Conference on Automated Software Engineering*, ASE '10, pages 173–174, New York, NY, USA, 2010. ACM.

4. Connected Data Objects (CDO). `http://www.eclipse.org/cdo/`.
5. J. Espinazo-Pagán, J. S. Cuadrado, and J. G. Molina. Morsa: A scalable approach for persisting and accessing large models. In *Model Driven Engineering Languages and Systems, 14th International Conference, MODELS 2011, Wellington, New Zealand, October 16-21, 2011. Proceedings*, pages 77–92, 2011.
6. A. Garmendia, E. Guerra, D. S. Kolovos, and J. de Lara. EMF splitter: A structured approach to EMF modularity. In *Proceedings of the 3rd Workshop on Extreme Modeling co-located with ACM/IEEE 17th International Conference on Model Driven Engineering Languages & Systems, XM@MoDELS 2014, Valencia, Spain, September 29, 2014.*, pages 22–31, 2014.
7. A. Garmendia, A. Jiménez-Pastor, and J. de Lara. Scalable model exploration through abstraction and fragmentation strategies. In *Proceedings of the 3rd Workshop on Scalable Model Driven Engineering part of the Software Technologies: Applications and Foundations (STAF 2015) federation of conferences, L'Aquila, Italy, July 23, 2015.*, pages 21–31, 2015.
8. A. Garmendia, A. Pescador, E. Guerra, and J. de Lara. Towards the generation of graphical modelling environments aided by patterns. In *SLATE*, volume 563 of *CCIS*, pages 160–168. Springer, 2015.
9. D. Granada, J. M. Vara, V. A. Bollati, and E. Marcos. *Enabling the Development of Cognitive Effective Visual DSLs.* Springer International Publishing, Cham, 2014.
10. P. Kelsen, Q. Ma, and C. Glodt. Models within models: Taming model complexity using the sub-model lattice. In *Fundamental Approaches to Software Engineering - 14th International Conference, FASE 2011, Held as Part of the Joint European Conferences on Theory and Practice of Software, ETAPS 2011, Saarbrücken, Germany, March 26-April 3, 2011. Proceedings*, pages 171–185, 2011.
11. A. Pescador, A. Garmendia, E. Guerra, J. S. Cuadrado, and J. de Lara. Pattern-based development of domain-specific modelling languages. In *MoDELS*, pages 166–175. IEEE, 2015.
12. M. Scheidgen, A. Zubow, J. Fischer, and T. H. Kolbe. Automated and transparent model fragmentation for persisting large models. In *Proceedings of the 15th International Conference on Model Driven Engineering Languages and Systems*, MODELS'12, pages 102–118, Berlin, Heidelberg, 2012. Springer-Verlag.
13. T. Stahl, M. Völter, J. Bettin, A. Haase, and S. Helsen. *Model-driven software development - technology, engineering, management.* Pitman, 2006.
14. D. Steinberg, F. Budinsky, M. Paternostro, and E. Merks. *EMF: Eclipse Modeling Framework, $2^{nd}$ Edition.* Addison-Wesley Professional, 2008. See also `http://www.eclipse.org/modeling/emf/`.
15. D. Strüber, J. Rubin, G. Taentzer, and M. Chechik. Splitting models using information retrieval and model crawling techniques. In *Fundamental Approaches to Software Engineering - 17th International Conference, FASE 2014, Held as Part of the European Joint Conferences on Theory and Practice of Software, ETAPS 2014, Grenoble, France, April 5-13, 2014, Proceedings*, pages 47–62, 2014.
16. R. Xu and D. C. W. II. Survey of clustering algorithms. *IEEE Trans. Neural Networks*, 16(3):645–678, 2005.