Virtual Composition of EMF Models [1]

Cauê Clasen — Frédéric Jouault — Jordi Cabot

AtlanMod Team (INRIA & École des Mines de Nantes)
4, rue Alfred Kastler
44307 Nantes Cedex 3
{caue.avila_clasen, frederic.jouault, jordi.cabot}@inria.fr

ABSTRACT. *Model composition is a very important modeling task as it allows to combine various perspectives of a system (represented by various models) into a single specialized view (a composed model). Several approaches have been proposed to tackle this problem, but they present some important limitations concerning efficiency, interoperability, and/or synchronization issues (mainly due to the element cloning mechanism used to create the composed model).*
In this paper we propose a new model composition method based on the virtualization of the composition mechanism. In our approach, the composed model is in fact created as a virtual model that redirects all its model access and manipulation requests directly to the set of base models from which it was generated. This is done transparently for the designer. Our mechanism improves the composition process with relation to the limitations mentioned above. The solution has been implemented and validated in a prototype tool on top of EMF.

RÉSUMÉ. *La composition de modèles est une tâche de modélisation très importante car elle permet de combiner différents points de vue d'un système (qui est représenté par divers modèles) en une seule vue spécialisée (un modèle composé). Plusieurs approches ont été proposées pour aborder ce problème, mais elles présentent d'importantes limitations concernant l'efficacité, l'interopérabilité, et/ou les problèmes de synchronisation (principalement en raison du mécanisme de clonage d'éléments utilisé pour la création du modèle composé).*
Dans cet article nous proposons une nouvelle méthode de composition des modèles basée sur la virtualisation du mécanisme de composition. Dans notre approche, le modèle composé est en fait créé comme un modèle virtuel qui redirige toutes ses demandes d'accès et de manipulation directement à l'ensemble des modèles de base à partir desquelles il a été généré. Cela se fait d'une manière transparente pour le concepteur. Notre mécanisme améliore le processus de composition par rapport aux limitations mentionnées ci-dessus. La solution a été implémentée et validée dans un prototype développé sur EMF (Eclipse Modeling Framework).

KEYWORDS: *Model composition, Virtual model, EMF, Model synchronization*

MOTS-CLÉS : *Composition de modèles, Modèle virtuel, EMF, Synchronisation de modèles*

1. This work has been partially supported by the CESAR European project and by the Galaxy French project.

1. Introduction

Model composition in its simplest form is *a modeling process that combines two or more input (contributing) models in order to generate a single output (composed) model*. Models are composed to provide users with a specific view of a system from a particular viewpoint (more adequate for a certain scenario, according to the user's needs), integrating information from a set of models that represent the various system perspectives. This can be a very challenging problem due the heterogeneous nature of models and the complex relationships that may exist between them.

Model composition has been studied from different perspectives: its formal semantics and possible composition operators [HER 07, VAL 10]; algorithms for automatically identifying relations between elements and composing them [FLE 07, KOL 06]; or also targeting specific families of models as ADLs [RUS 10], or UML [XIN 05]. A commonality between most of these approaches is that they generate a new composed model by copying/cloning contributing model elements into the composed model, completely discarding contributing models once composition is finished.

This data duplication mechanism is inefficient both in terms of memory usage (cloning elements can be a serious bottleneck when composing large-size models), and creation time (copying a large number of elements can be very time consuming, and composition needs to be re-executed every time contributing models are modified). Besides, synchronization issues can also arise as composed and contributing models do not share the same model element instances (updates in contributing models are not propagated to composed model, or vice-versa), thus leading to inconsistency.

In order to overcome these limitations, we introduce the concept of a virtual composed model, which is a model that does not contain concrete data, but redirects all its model manipulation requests to the (contributing) models from which it was generated. We realize this concept as an extension of the *Eclipse Modeling Framework* (EMF) [1] to showcase how it improves the overall quality of model composition [2].

2. The Notion of a Virtual Model

A virtual model is a model whose (virtual) elements are proxies to elements contained in other models. It provides to tools/users the *illusion* of working with a regular model whereas, in fact, all model manipulation requests are transparently redirected to elements contained in the virtualized contributing models. This behaviour allows to seamlessly integrate contributing model elements into the composed model, so they serve no longer as mere inputs for generating the composed model, but rather as core artefacts during its whole lifecycle. The virtualization of the composed model allows the improvement of model composition through the following properties:

– It is viewed and handled as a normal model (a tool/user does not know it is dealing with a virtual model). This guarantees their interoperability;

1. `www.eclipse.org/emf/`
2. Prototype available in `www.emn.fr/z-info/atlanmod/index.php/VirtualModel`

– It actually uses the same element instances as the contributing models. This means: perfect synchronization between contributing and composed models (guaranteeing consistency); no duplicated data (less memory usage); and faster creation of the composed model (no need of cloning elements);

– Complex inter-model relationships can be defined as in any other composition solution. E.g., virtual elements may refer to two contributing model elements (merging); or associations may be created between different contributing elements.

3. Virtualizing EMF Models

In this section we realize the concept of virtual models in practice using EMF as the targeted modeling environment. EMF (as any modeling framework) provides an API to manipulate its models and model elements. Roughly put, EMF uses the `Resource` and `EObject` interfaces to represent, respectively, models and model elements. Some `Resource` methods are `load` and `save` models, or get (`getContents`) its root model elements (in EMF, elements are organized hierarchically in a tree). `EObject` provides methods to, e.g., `get` or `set` model elements and its properties, get an element's container (`eContainer`), or its metaelement (`eClass`). These (and others) operations must be refined to provide an adequate support for the creation, retrieval, and manipulation of virtual models. We achieve this through three main steps:

3.1. *The Composition Metamodel*

As any composed model, a virtual model conforms to a composition metamodel that defines the concepts that can be presented in the composed model (e.g., elements conforming to a given type can be discarded, and others can be merged into a single one). The composition metamodel could be also virtualized, but as metamodels are considerably smaller than models, the benefits of virtualizing them are not so sound.

We automatically generate the composition metamodel through an ATL [3] transformation that – receiving as input the contributing metamodels and a weaving model [4] establishing links between them – generates, as transformation output, the composition metamodel populated with the selected contributing metaelements, possibly linked according to the weaving model elements.

3.2. *The Model Virtualization API*

The Model Virtualization API works as a refinement of the standard EMF API to transparently support operations on virtual models. We provide, mainly, implementations of the `Resource` and `EObject` interfaces, making them capable of identifying the correct concrete model element(s) a given virtual model elements refers to, and then delegating the request to it by using the standard EMF API (as shown in Fig. 1).

3. `http://www.eclipse.org/atl/`
4. Defined with the AMW technology: `www.eclipse.org/gmt/amw/`

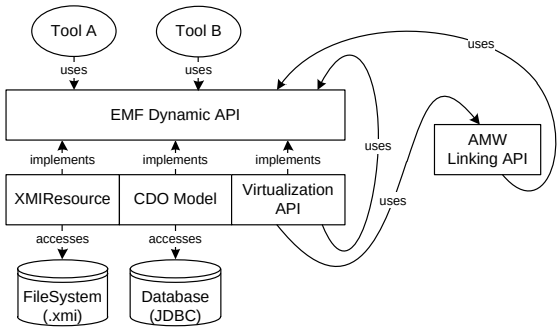

Figure 1. *API Relationship for EMF Model Virtualization.*

In a first step, the virtual model is created by invoking the `load` method of our `Resource` implementation, whose arguments are simply the persistence locations (stored in a `.virtualmodel` file) of all resources involved in the composition (i.e., contributing models, metamodels, and weaving models). Our `load` method then loads each of these resources (by using EMF's standard `load` method), and builds a series of internal mappings (mostly relating composition metaelements and contributing metaelements) that will aid in the task of identifying to which concrete element a virtual element refers to, and in which concrete model this element is contained.

Similarly, at the element level we provide a specific implementation of `EObject` that allows to navigate from a virtual to a corresponding concrete element (or elements, in the case of element merging). As virtual elements do not hold concrete data, our `EObjects` are created lazily, on demand, using the mentioned internal mappings to identify and retrieve the correct concrete model elements they refer to.

3.3. *The Linking API*

An auxiliary Linking API is in charge of managing *virtual links*: relationships between elements contained in different contributing models. The Model Virtualization API identifies if a given operation is requested on a virtual link, and, if positive, delegates its management to the Linking API. Maintaining it as a separated component was a choice to gain in modularity, as different implementations (e.g., different kinds of virtual links, or automatic composition scenarios) could be provided without having to modify the main virtualization mechanism.

The Linking API uses AMW to load, save, and manipulate virtual links in the form of weaving model elements. The possible types of virtual links (e.g. inter-model inheritance, simple or bidirectional references,...) are defined in a weaving metamodel, and, by checking its type, the Linking API identifies how to manage each kind of virtual link. For instance: when the Virtualization API identifies that a given requested reference is in fact a reference between elements in different models (i.e. a virtual association), it delegates its management to the Linking API. The Linking API then finds the weaving model element referring to this particular association, retrieves the referenced concrete model element, and returns it to the Virtualization API.

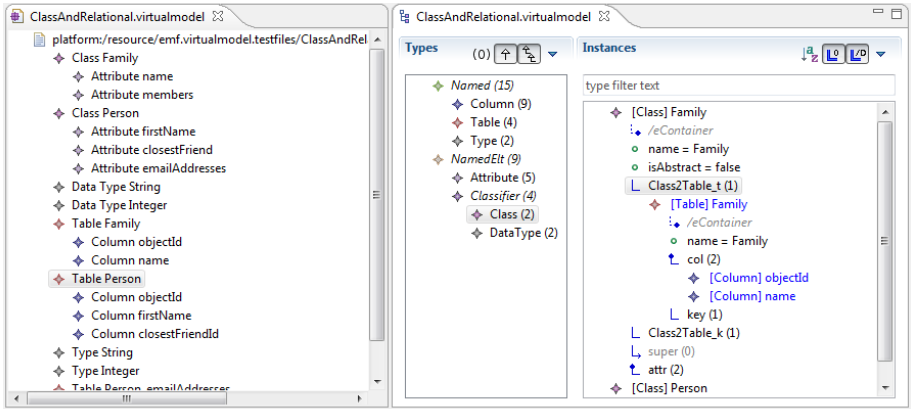

Figure 2. *Screenshot of a virtual composed model – resulting from the composition of a Class model with a Relational model – handled with two EMF-based tools: the Ecore Sample Model Editor (left), and the MoDisco Model Browser (right).*

4. Conclusion

In this paper we have presented a new approach for model composition based on a model virtualization mechanism that offers a direct and transparent access to the contributing models used in the composition process. An EMF-based prototype (presented in Fig. 2) has been developed to translate, through a Virtualization API, manipulation requests on virtual elements to appropriate operations on contributing ones. The approach can be also extended to support different composition techniques by providing alternative implementations of our Linking API. As further work, we plan to extend the list of possible inter-model relationships to enrich our virtualization mechanism, and to provide an implementation of the Linking API based on a state-of-the-art automatic composition technique.

5. References

[FLE 07] FLEUREY F., BAUDRY B., FRANCE R. B., GHOSH S., "A Generic Approach for Automatic Model Composition", *MoDELS Workshops*, 2007, p. 7-15.

[HER 07] HERRMANN C., KRAHN H., RUMPE B., SCHINDLER M., VÖLKEL S., "An Algebraic View on the Semantics of Model Composition", *ECMDA-FA*, 2007.

[KOL 06] KOLOVOS D. S., PAIGE R. F., POLACK F., "Merging Models with the Epsilon Merging Language (EML)", *MoDELS*, 2006.

[RUS 10] RUSCIO D. D., MALAVOLTA I., MUCCINI H., PELLICCIONE P., PIERANTONIO A., "Developing next generation ADLs through MDE techniques", *ICSE*, 2010.

[VAL 10] VALLECILLO A., "On the Combination of Domain Specific Modeling Languages", *ECMFA*, 2010.

[XIN 05] XING Z., STROULIA E., "UMLDiff: an algorithm for object-oriented design differencing", *ASE*, 2005.