Megamodeling Software Platforms:
Automated Discovery of Usable Cartography from Available Metadata

Vincent Mahé
INRIA Rennes – Bretagne Atlantique
Rennes, France
vincent.mahe@inria.fr

Frédéric Jouault, Hugo Bruneliere
AtlanMod Team (INRIA & Ecole des Mines de Nantes)
INRIA Rennes – Bretagne Atlantique
Nantes, France
{frederic.jouault | hugo.bruneliere}@inria.fr

Model-driven reverse-engineering focuses on automatically discovering models from different kinds of available information on existing software systems. Although the source code of an application is often used as a basic input, this information may take various forms such as: design "models", bug reports, or any kind of documentation in general. All this metadata may have been either built manually or generated (semi-)automatically during the whole software life cycle, from the specification and development phase to the effective running of the system. This paper proposes an automated and extensible MDE approach to build a usable cartography of a given platform from available metadata by combining several MDE techniques. As a running example, the approach has been applied to the *TopCased*™ MDE platform for Embedded & Real-Time Systems.

Keywords: Reverse-engineering, Software platform, Cartography, Metadata, Megamodeling, Model Transformation

I. Introduction

As an effective solution for designing, developing and managing software systems, Model-Driven Engineering is gaining more and more interest from the industry. However, switching from classical processes to MDE requires dealing with the huge amount of existing legacy in terms of applications, documentation, raw data, etc. As a consequence, reverse-engineering techniques [3] [4] are now quite commonly applied. To this intent, all kinds of software artifact should be able to be reverse engineered as usable models. Generally, these artifacts have either been created manually (by architects, designers, developers, users, etc) or (semi-)automatically generated by a tool. There are already several tools which provide the capability to discover models from legacy source code in the *MoDisco* project on *Eclipse.org* [5]. However, tools offering similar reverse engineering features on other kinds of software artifact, such as documentation, are still rare. This paper focuses on the second category of artifact, independently of the way they have been built and modified afterward.

In this paper, we present an extensible MDE approach which is based on the combined use of megamodeling [1] and model transformation [6] techniques. Its goal is to allow the automated cartography of existing software platforms by merging generated metadata with user-specified metadata. Thus, a megamodel of a given platform (i.e. a global model of its artifacts) is automatically built from data contained in *Excel*™ files. They are first injected into models and then successively transformed. As a concrete industrial example of a software platform, we consider the *TopCased*™ platform (an MDE platform for embedded & real-time systems) [2], on which we have directly applied our approach, in order to realize its cartography.

This paper is organized as follows. Section 2 describes the scope of our work and motivates our approach within this reverse engineering context. Section 3 provides more details about the actual tooling developed as an implementation of this approach. Section 4 illustrates how our solution has been concretely applied on the previously stated example. Section 5 presents some related works while Section 6 concludes the paper.

II. Motivation & Approach

Software platforms are actually composite systems which are becoming more and more complex. These systems are usually built on top of a generic platform by plugging on it many different software components or tools which have to be compatible. Such platforms are often extended or specialized for a given domain by respectively adding or removing one or several components. In other words, they are intended to be customized, of course without breaking any required dependency, in order to create more dedicated platform.

Within the context of such platforms, numerous inter-related artifacts of several different kinds have to be considered: for instance Eclipse plug-ins and associated files, services offered to end users, modelers and editors, source code, documentation, etc. Thus, there is a significant amount of information on these platforms (and corresponding artifacts) which is already available in various heterogeneous formats.

Being able to homogeneously manage all the metadata on these software platforms in order to ensure their continuous validity and consistency is still an open challenge.

As a partial solution to this problem, this paper proposes to apply an MDE approach. The underlying idea is to make

an intensive use of modeling techniques in order to: 1) get the required metadata as models, 2) generate from them a complete cartography (i.e. a megamodel) of the platform, 3) use it for checking this platform.

The well-known *TopCased* E/RT platform is based on the Eclipse Modeling Platform, which is itself a basic Eclipse platform fitted with different MDE-specific plug-ins. Within this paper, this industrial platform is used as a demonstrator for applying our approach on a real-life use case.

III. An MDE Tooling for Megamodeling Software Platforms

We present here the tooling which has been developed for dealing with consistency and validity of models built from heterogeneous inputs, as illustrated in Fig. 1: injection of metadata in various formats as models (text-to-model transformations), generation of a megamodel of the platform (model-to-model transformation [6] + megamodeling [1]), use of this megamodel for various goals (model-to-model transformation). These steps are detailed below.

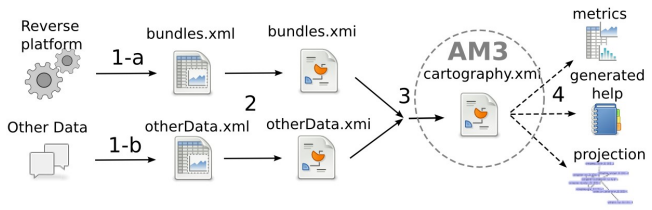

Figure 1. Main scheme

A. Injection of metadata

Thanks to its GUI, *Excel* makes data human-readable and human-writable. This data-centric application dedicated to end-users and its clear and powerful interface makes information capture easier and simpler.

Office OpenXML format gives *Excel* the ability to save data sheets into structured XML files. These files are computer-readable and computer-writable. This means that hand-typed data can be processed by programs, and that automatically discovered data (i.e. by reverse-engineering) can be stored in a human readable format. For our purpose, we use the ATL *Microsoft Excel Injector* [7] to transform *Excel* 2003 XML textual files into *SpreadsheetSimplifiedML* models.

B. Generation of the megamodel

ATL, the *AtlanMod Transformation Language*, is a hybrid language (with declarative and imperative constructs) to write model transformations and queries. An Eclipse environment for ATL users is also available.

Models from the previous step can be processed as usual with specific ATL transformations in order to get a model representing the platform to examine. This tool chain is based on the *Eclipse Modeling Framework* (EMF) and its XMI file format. We obtain a megamodel of the system under examination.

C. Use of the megamodel

The *AtlanMod MegaModel Management* (AM3 [8]) tool provides a practical support for modeling in the large, to deal with global resources in a MDE environment. In AM3, our megamodel can be processed in order to get information and to do some works and verifications on it, like metrics, visualizations and constraint checking.

IV. Concrete Example: Building a Cartography of the TopCased Platform

We first present here the application of the previously described approach on the example of the *TopCased* platform (inputs: merged *Excel* stylesheets, output: megamodel of the platform). In a second part, we present examples of what can be done with such a megamodel of *TopCased* (concrete examples of metrics, visualizations, etc.).

A. Obtaining the TopCased megamodel

The transformation of *TopCased* metadata (including reverse engineering of the platform) into a megamodel of this platform consists of four steps. The first step consists in capturing data about the platform and is split into two parts which deliver *Excel* files in XML format. The second step converts these files into EMF models which conform to the *SpreadsheetSimplifiedML* metamodel [7]. Once we get these models, the third step merges them and transforms them into a platform cartography model. In a forth step, many computations and works can be applied on this model, like metrics, generation of documentation, graphical views, dependencies validation. The three first steps are described in this section. The fourth step is detailed in the next section.

Step 1-a: platform reverse engineering. We write a program running on the platform under examination, which looks for available information and generates a textual file with ad-hoc separators (similar to *Comma-Separated Values* or CSV), as shown in fig. 2.

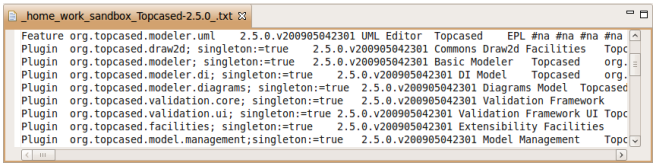

Figure 2. Textual result of reverse engineering

Microsoft™ Excel™ application easily opens this file (fig. 3) and we save it in *Excel* 2003 XML flat format.

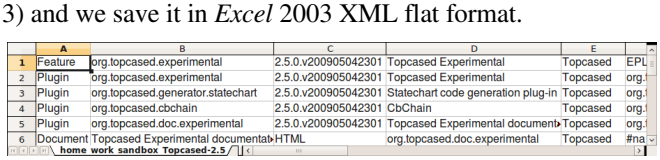

Figure 3. Excel view of reverse engineered data

Thanks to *Excel* GUI, we get human-readable data as result of the reverse engineering process.

You may notice that a more sophisticated discovery program could directly generate an XML file conforming to the *Excel* 2003 XML specification but we prefer to leverage the capabilities of Excel and save ourselves some work.

Step 1-b: typing data in Excel. Data about the platform, which is needed into the platform model but misses from the automatically discovered information, is captured by people who know such data, using the *Microsoft™ Excel™* application. They must use the same column format than the one of the generated files. Below is an excerpt of the specified syntax of our *Excel* files:

'Model2Model' row structure

A cell *Type* = "Model2Model"

B cell *Name* : the official label of the transformation. Sample: "MARTE to SynDEx transformation"

C cell *URI* : the short identifier under which the transformation is known. Sample: "MARTE2SynDEx"

D cell *Language URI* : the short identifier of the language used to write the transformation. Sample: "ATL"

E cell *Source URI* : the URI of the metamodel the input model conforms to. Sample: "UML" *[...]*

The collection of data is done using the Excel GUI, as shown in fig. 4.

	A	B	C	D	E	F
1	Type	Name	URI	Language URI	Source URI	Target URI
2	Model2External	BNF to SableCC Export	BNF2Sable	ATL	BNF	SableCC
3	Model2Model	UML profiled SDL to Fiacre	SDL2Fiacre	ATL	UML	Fiacre
4	Model2Model	M2M Ecore to XSD	Ecore2XSD	Xtend	Ecore	XSD
5	Model2Model	M2M News (OPML, Atom, RSS) to XH	News2XHTML	Xtend	OPML	XHTML
6	Model2Model	M2M XML to SVG via Xtend	XML2SVG	Xtend	XML	SVG
7	Model2External	M2T custom XML to Java via Xpand	XML2Java	Xpand	XML	Java

Figure 4. Excel view of hand-made data

Step 2: transform Excel into model.

Microsoft™ Excel™ application uses the *SpreadsheetML* part of the *OpenXML* specification as format to export its workbook into textual XML files. This format has been introduced in *Microsoft™ Office™* 2003. The *AtlanMod* research team has developed a metamodel for a subset of this specification, called *SpreadsheetSimplifiedML*, and tools to switch from *Excel* file to model and back (*Excel* injector and extractor). The *Excel* injector [7] consists of few technical steps: the raw XML file is injected in the corresponding XML model, thus entering the homogeneous MDE world. Then a *SpreadsheetSimplifiedML* model is extracted from this XML model using a specific ATL transformation:

bundles.xml → bundles-XML.xmi → bundles-SimplifiedML.xmi

otherData.xml → otherData-XML.xmi → otherData-SimplifiedML.xmi

The *XML2SpreadsheetSimplifiedML* ATL transformation is, as usual, a Model2Model set of declarative rules.

You may see this transformation as a refinement of a raw XML model into an effective spreadsheet model. The two files could be merged during this jump-to-model step, but we choose to use the generic *Excel* injector tools as-is, and to perform the merge in a separate step, with an application-specific transformation.

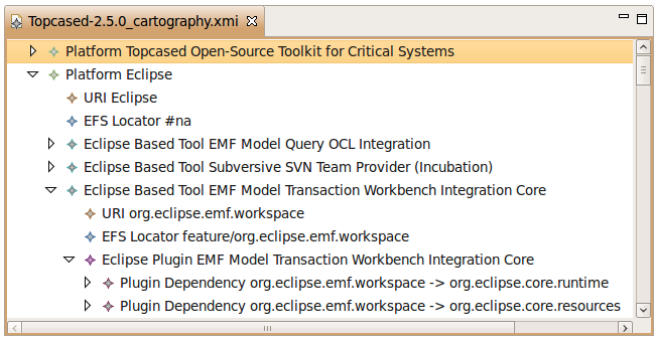

Figure 5. AM3 cartography model (with full merged data)

Step 3: acquire AM3 cartography from model. Both generated and handled data models are subsets of the total information about the platform under study. As serialized parts of a whole model, they can be processed in the same MDE transformation, which takes *SpreadsheetSimplifiedML* rows and cells and generates corresponding domain specific *GMM4Cartography* megamodel.

The result is a full *GMM4Cartography* megamodel containing instances of both *Excel* data files and their crossed relationships, as we can see in fig. 5.

B. *Exploiting the TopCased megamodel*

Once we have built the central *GMM4Cartography* megamodel of the platform, we can define all MDE tools we need to take benefits of this information.

As an example, we wrote an ATL query to get some metrics about the *TopCased* platform (and its underlining *Eclipse* tools). The resulting text file is shown in fig. 6:

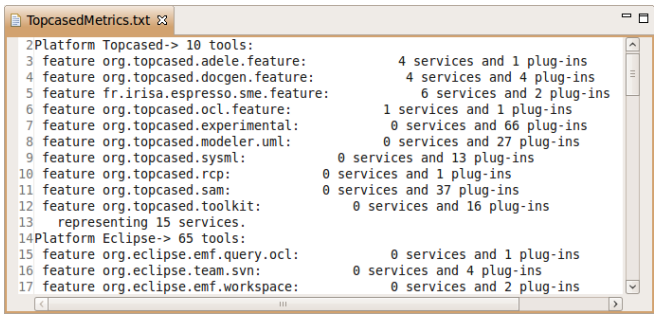

Figure 6. Some metrics about the Topcased platform

A tool has been build upon the AM3 megamodeling environment in order to visualize the megamodel entities and their relationships. As our platform cartography conforms to a metamodel, which is based on the underlying AM3 metamodel, we can use this tool directly. The result is a visualization of the *TopCased* plug-ins dependencies as seen in Fig. 7:

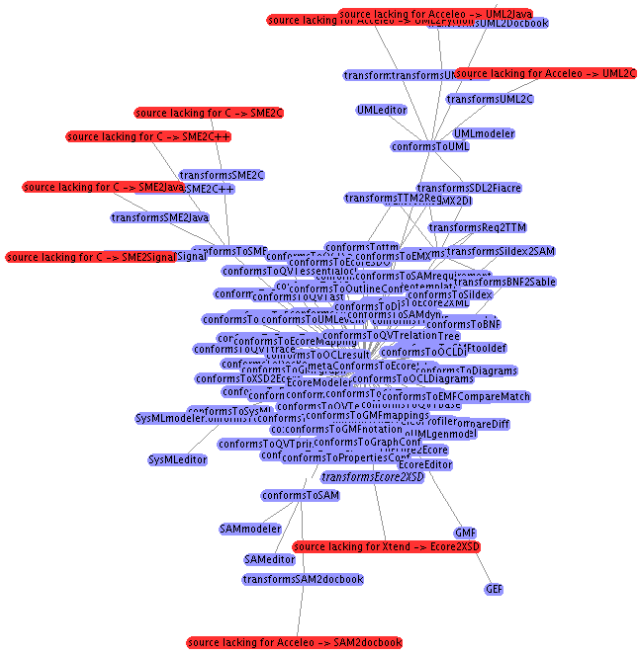

Figure 7. View of the dependencies in the TopCased platform

The red elements correspond to lacking targets of required dependencies. Such a work displays easily problems which have not been detected previously by the Eclipse plug-in system.

V. Related Works

Another way to merge automated and hand-made could be to use a Text-to-Model approach [9] and tools like TCS [10] or EMFText [11]. But it assumes a concrete syntax, which was only implicit in our case, as well as some unusual skills from users.

Some research works explored the question of tools interoperability [12]. We used the corresponding tools in our use case, emphasizing the interest of such an approach.

VI. Conclusion

Software platforms are increasingly complex and some of these platforms are even built on top of each other (e.g., *TopCased* on top of *Eclipse Modeling*). Moreover, in industrial use cases, it is sometimes necessary to integrate several platforms together. For these reasons, generic and extensible tools are needed to understand software platforms. The approach introduced in this paper brings a partial answer to this important requirement. The presented MDE tool chain performs platform cartography. It produces models of software platforms by combining information from two kinds: 1) automatically discovered in the software platforms themselves, and 2) manually entered by experts on these platforms.

Such models of platforms may be browsed (e.g., in AM3) or further processed in order to visualize their structure, measure their complexity, check some consistency rules, etc. Generic and extensible modeling tools like AM3 [8] may be leveraged for these purposes.

Additionally, this work emphasizes the fact that model-driven techniques (e.g., model transformation, or megamodeling) enable the homogeneous integration of heterogeneous information on platforms by using models as first-class entities.

Acknowledgment

The present work is being supported by the CESAR European ARTEMIS-JU project, and relies on the AM3 tooling developed within the MODELPLEX European IST-FP6 research project.

References

[1] Bézivin, J, Jouault, F, and Valduriez, P , “On the Need for Megamodels” In: Proceedings of the OOPSLA/GPCE: Best Practices for Model-Driven Software Development workshop , OOPSLA, 2004

[2] Topcased: http://www.topcased.org/ (Open Source toolkit for critical systems)

[3] Favre, J.M, Musset, J, “Rétro-ingénierie dirigée par les métamodèles,”. In: IDM'06, Lille – Secondes Journées sur l'Ingénierie Dirigée par les Modèles

[4] Rugaber, S, Stirewalt, K, "Model-Driven Reverse Engineering," *IEEE Software*, vol. 21, no. 4, pp. 45-53, July/August, 2004.

[5] Eclipse-GMT MoDisco: http://www.eclipse.org/gmt/modisco/

[6] Jouault, F, and Kurtev, I “Transforming Models with ATL” In: Proceedings of the Model Transformations in Practice Workshop at MoDELS 2005, Montego Bay, Jamaica, 2005

[7] http://www.eclipse.org/m2m/atl/atlTransformations/#MSOfficeExcelInjector

[8] Eclipse-GMT AM3 Project: http://www.eclipse.org/gmt/am3/

[9] Wimmer, M, and Kramler, G “Bridging Grammarware and Modelware”. In: Proceedings of the 4th Workshop in Software Model Engineering (WiSME 2005), Montego Bay, Jamaica.

[10] Jouault, F, Bézivin, J, and Kurtev, I “TCS: a DSL for the Specification of Textual Concrete Syntaxes in Model Engineering”. In: Proceedings of GPCE’06, Portland, USA

[11] Eclipse EMFText Project: http://www.emftext.org

[12] J. Bezivin, H. Bruneliere, F. Jouault, and I. Kurtev: "Model Engineering Support for Tool Interoperability". In: Proceedings of the 4th Workshop in Software Model Engineering (WiSME 2005), Montego Bay, Jamaica.