Model2Roo: Web Application Development based on the Eclipse Modeling Framework and Spring Roo

Juan Castrejón[1], Genoveva Vargas-Solar[2], and Rafael Lozano[3]

[1] Université de Grenoble, LIG-LAFMIA,
[2] Centre National de la Recherche Scientifique, LIG-LAFMIA
681 rue de la Passerelle, Saint Martin d'Hères, France
{Juan.Castrejon, Genoveva.Vargas}@imag.fr
[3] Instituto Tecnológico y de Estudios Superiores de Monterrey,
Campus Ciudad de México, Calle del Puente 222, México, México
ralozano@itesm.mx

Abstract. Inherent complexity in web application development is continually increasing, due to technical challenges, like new programming frameworks and tools. In this context, model-driven techniques can currently be used to guide the development of web systems, by focusing on different levels of modeling abstractions that encapsulate both implementation details and the definition of system requirements. This paper presents Model2Roo, a tool intended for Java web application development, that relies on the Eclipse Modeling Framework and on the Spring Roo project. In particular, this paper outlines key issues highlighted by previous users of the tool, and demonstrates recent implemented features.

1 Introduction

Web application development is one of the most evolving industries in software engineering [3]. However, this evolution also represents an increased complexity in functional and non-functional requirements associated to web applications [3]. In this environment, software engineers need to constantly evaluate new technical solutions for the development and maintenance of web applications, including a wide variety of programming languages, frameworks and tools.

To help overcome this complexity, the Model2Roo [1] project was presented in early 2011. Model2Roo uses a model-driven approach, in particular Model to Text (M2T) techniques, to transform UML and Ecore [10] class diagrams into appropriate Spring Roo commands [9], which can then be used to generate a full-blown Java web application. The generated applications are automatically built with a set of architecture patterns and industry best practices [9], and contain not only the static structure of the system, but also comprehensive support for the functionality associated to the Spring Framework module[4]. Model2Roo has also been available as an open-source tool since early 2011[5].

[4] http://www.springsource.org/
[5] http://code.google.com/p/model2roo/

This paper describes related work (Section 2), the main issues encountered by previous users of the tool (Section 3), demonstrates implemented features (Section 4), and finally, outlines conclusions and future work (Section 5).

2 Related work

Web application development based on model-driven techniques is a widely researched topic in software engineering. Representative results of this research include the Web Modeling Language (WebML) [2] and the UWE approach [5]. Both projects provide tool support to generate fully functional web applications, including not only presentation content, but also complex navigation features.

In comparison, tool support for Model2Roo is not yet as complete and mature as the one provided by similar projects, however the main advantage of our tool is that it provides an association to modern application development tools, by means of the Spring Roo project. As a result, developers can easily access the full potential of the Spring framework, and associated projects.

3 Previous tool issues

Three main issues have consistently been highlighted by previous users of the tool: (i) Insufficient support for graphical environments; (ii) Basic edition of Spring Roo properties; and, (iii) Troublesome installation procedure.

The first issue, support for graphical environments, was mainly associated to the use of UML tools within the Eclipse Modeling Framework (EMF). In this case, though Model2Roo could generate Spring Roo commands either from Ecore and UML files, users found it troublesome to generate these files with the basic *UML* and *Ecore Model Editors* [10], and wanted to use more complex tools, such as Papyrus [8]. However, when trying to use Papyrus to generate UML class diagrams, several incompatibilities with our tool were discovered, such as inappropiate use of numeric data types.

Model2Roo allows the specification of Spring Roo properties through *Ecore Annotations* and *UML Profiles*. However, in previous versions of the tool, the value for each property could only be set as free text. This led to subtle errors, such as assigning a *String* value to a property that required a *numeric* value, but more importantly, users that were not familiar with Spring Roo, did not know the possible domains for each property. For example, consider the property *debugLevel*, that configures the level of application debug traces. In previous versions of the tool, users had to know that this particular property could only be assigned one of the following values: *Debug, Info, Warn, Error* and *Fatal*.

The third issue highlighted by the users, was a troublesome installation of the tool. In this regard, though Model2Roo was distributed as a single plugin file, the main issue was the installation of the required dependencies, in particular, users were required to manually install the ATL [4] and Papyrus distributions.

These three main issues have been fixed in the last version of the Model2Roo tool. The details will be demonstrated in the following section.

4 Features demonstration

This section demonstrates the Model2Roo features, using the Spring PetClinic sample application[6]. This system is intended for clinic employees who need to view and manage information regarding veterinarians, clients, and pets.

A screencast detailing this demonstration is available in the project web site[7]. To try this demonstration locally, install Spring Roo, an Eclipse Modeling distribution[8], and finally, Model2Roo, using the project update site[9].

It should be noted that the update site already references all of the features that are required to execute Model2Roo, both over Ecore and UML projects.

To generate the test application, we must define a class diagram containing the system *Classes* and *Enumerations*, as well as their *properties* and the *associations* between them. In this case, we are going to use the Papyrus project, applying the *rooCommand* and *rooStructure* UML profiles, which are part of the Model2Roo installation. Fig. 1 depicts a fragment of the resulting class diagram.

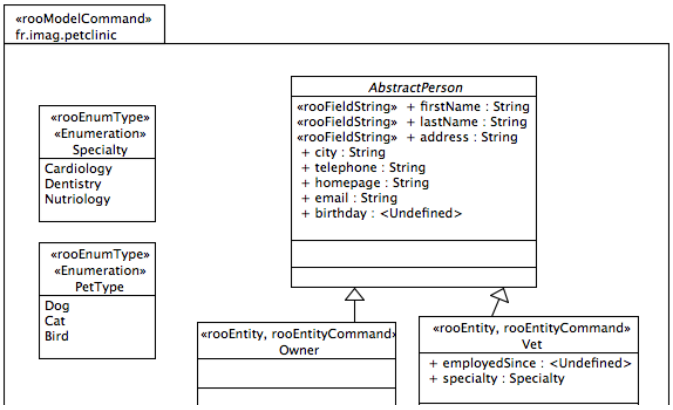

Fig. 1. Fragment of the PetClinic class diagram.

As opposed to previous versions of Model2Roo, the current version fully supports Papyrus profiles. As a consequence, property values can now be specified using drop-down menus. Also, in the current version of Model2Roo, the Acceleo project [6] is used to generate the Spring Roo commands that correspond to UML diagrams, as opposed to ATL queries in previous versions. This change was motivated to increase maintainability, considering that Acceleo provides an implementation of the MOF Model to Text Language (MTL) standard [7].

Finally, the generated commands can be executed by the Spring Roo console, in order to create the corresponding Java web application, as depicted in Fig. 2.

[6] http://static.springsource.org/docs/petclinic.html
[7] http://code.google.com/p/model2roo/
[8] http://eclipse.org/downloads/packages/eclipse-modeling-tools/indigosr1
[9] http://model2roo.googlecode.com/svn/trunk/fr.imag.model2roo.update.site/

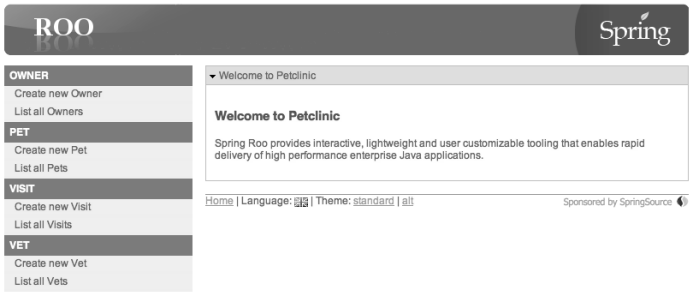

Fig. 2. Generated web application.

5 Conclusions and Future work

This paper described Model2Roo, a tool for web application development that relies on UML and Ecore class diagrams, in order to generate Spring Roo commands. The main issues highlighted by previous users were discussed, as well as recent implemented features. Support for the full gamut of Spring Roo commands is intended for future work.

Acknowledgments

This work is funded by the French National Research Agency, through the UBIQUEST project (http://ubiquest.imag.fr) ANR-09-BLAN-0131-01, and by the STIC-AMSUD program, within the CLEVER project (http://clever.imag.fr).

References

1. Castrejón, J., López-Landa, R., Lozano, R.: Model2Roo: A Model Driven Approach for Web Application Development based on the Eclipse Modeling Framework and Spring Roo. In: Electrical Communications and Computers (CONIELECOMP), 2011 21st International Conference on. pp. 82 –87 (March 2011)
2. Ceri, S., Fraternali, P., Bongio, A., Brambilla, M., Comai, S., Matera, M.: Designing Data-Intensive Web Applications. Morgan Kaufmann, San Francisco, CA (2003)
3. Jazayeri, M.: Some Trends in Web Application Development. In: Future of Software Engineering, 2007. FOSE '07. pp. 199 –213 (May 2007)
4. Jouault, F., Allilaire, F., Bézivin, J., Kurtev, I.: ATL: A model transformation tool. Science of Computer Programming 72(1-2), 31–39 (June 2008)
5. Kraus, A., Knapp, A., Koch, N.: Model-Driven Generation of Web Applications in UWE. In: 3rd International Workshop on Model-Driven Web Engineering (2007)
6. Obeo: Acceleo. http://www.eclipse.org/acceleo/ (November 2011)
7. Object-Management-Group: MOF Model to Text Transformation Language, v1.0. http://www.omg.org/spec/MOFM2T/1.0/ (January 2008)
8. Papyrus: Papyrus. http://www.eclipse.org/modeling/mdt/papyrus/ (April 2012)
9. SpringSource: Spring Roo. http://www.springsource.org/spring-roo/ (May 2012)
10. Steinberg, D., Budinsky, F., Paternostro, M., Merks, E.: EMF: Eclipse Modeling Framework. Addison-Wesley Professional, Boston, Massachusetts, 2nd edn. (2008)