An Eclipse Plug-in for Model-Driven Development of Rich Internet Applications*

Santiago Meliá, Jose-Javier Martínez, Sergio Mira,
Juan Antonio Osuna, and Jaime Gómez

Universidad de Alicante, IWAD, Campus de San Vicente del Raspeig,
Apartado 99 03080 Alicante, Spain
{santi,jmartinez,smira,jaosuna,jgomez}@dlsi.ua.es

Summary. Rich Internet Applications (RIAs) have recently appeared in the Internet market offering a rich and efficient User Interface similar to desktop applications. However, these applications are rather complex and their development requires design and implementation tasks that are time-consuming and error-prone. In this paper, we present a tool called OIDE (OOH4RIA Integrated Development Enviroment) aimed at accelerating the RIAs development through the OOH4RIA approach which establishes a RIA-specific model-driven process.

1 Introduction

In the last few years, a new type of sophisticated Web applications called Rich Internet Applications (RIAs) are breaking through the Internet market offering better responsiveness and a more extended user experience than the traditional Web applications. In essence, RIAs are client/server applications that are at the convergence of two competing development cultures: desktop and Web applications. They provide most of the deployment and maintainability benefits of Web applications, while supporting a much richer client User Interface (UI).

However, RIAs are complex applications, with time-consuming and error-prone design and implementation tasks. They require designing a rich user interface based on the composition of Graphical User Interface (GUI) widgets and event-based choreography between these widgets.

Therefore, RIA development requires new design methods and tools to represent this complex client/server architecture and to increase the efficiency of the development process through code generation techniques able to accelerate it and reduce errors. To achieve this goal we propose a seamless and domain-specific RIA development approach called OOH4RIA [4], which proposes a model-driven development process based on a set of models and transformations to obtain the implementation of RIAs. This approach specifies an almost complete Rich Internet Application (RIA) through the extension of the OOH server-side models (i.e. domain and navigation) and with two new platform-specific RIA presentation

* This work is supported by the Spanish Ministry of Education under contract TIN2007-67078 (ESPIA).

B. Benatallah et al. (Eds.): ICWE 2010, LNCS 6189, pp. 514–517, 2010.

models (i.e. presentation and orchestration). In order to give support to this approach, we have implemented a Rich Client Platform tool called OOH4RIA Integrated Development Environment (OIDE) [6] defined by a set of model-driven frameworks developed in Eclipse. Currently, this approach has been extended by introducing artifacts that gather new concerns: a RIA quality model and the technological and architectural RIA variability [5].

For an adequate comprehension of the OIDE tool, next section 2 presents a general perspective of OIDE development process which implements partially the OOH4RIA approach process. And finally, section 3 shows the most important technological features of OIDE.

2 An Overview of the OIDE Development Process

OIDE implements partially the OOH4RIA development process specifying an almost complete Rich Internet Application (RIA) through the extension of the OOH [2] server-side models (i.e. domain and navigation) and with two new RIA presentation models (i.e. presentation and orchestration).

This OIDE process starts by defining the OOH domain model, which is based on the UML class diagram notation, to represent the domain entities and the relationships between them. To do so, we have defined an extended domain MOF metamodel to obtain an Object-Relational Mapping without ambiguities. To improve the quality of the server side, the domain model has introduced several fundamental adjustments: (1) defining a topology of different operations such as create, delete, relationer, unrelationer, etc. to generate the CRUD (i.e. Create, Read, Update and Delete) functionality of a data-intensive server-side. (2) Dealing with a complete collection datatypes such as set, bag, list, etc. (3) Introducing concepts to remove the ambiguity of the Object-Relational Mapping (ORM) such as the object identifier to obtain the primary key, mapping from UML datatypes to database datatypes, database aliases in class (to name the table), in attribute (to name the column) and in roles (to name the foreign keys).

After specifying the domain model, the developer must design the OOH Navigation Model to define the navigation and visualization constraints. The navigation model – a DSL model – uses a proprietary notation defined by the OOH method formalized by a MOF metamodel. This model establishes the most relevant semantic paths through the information space filtering the domain elements available in the RIA client-side. It also introduces a set of OCL filters permitting to obtain information from the domain model.

At this point, the UI designer begins the definition of the RIA client-side establishing the complete layout by means of a structural representation of different widgets (e.g. dimensions x, y, position), panels (e.g. horizontal, vertical, flow, etc.) and style (e.g. colour, background, fonts, etc). There are many RIA frameworks, each with a different set of widgets with their own properties and events. For this reason, we have defined a platform-specific model called Presentation Model, which can be instantiated for each supported RIA framework (currently, OIDE gives support to Google Web Toolkit and Silverlight) thus obtaining similarity with the look and feel of a RIA UI.

To complete the information needed by an interactive UI, OOH4RIA incorporates a platform-dependent model called Orchestration Model, which helps introduce the dynamic behavior of the RIA UI. This model does not have a graphical representation in the OIDE tool, being defined by a set of Event-Condition-Action (ECA) rules defined by a property form. They determine how different widgets receive the events from users and if an OCL condition is accomplished they invoke a set of Actions that correspond to one or more widget methods or to one or more services offered by the server-side business logic. These events and methods offered by the OIDE tool are RIA framework specific (i.e. Silverlight or GWT) permitting the user to specify the proper arguments accurately.

The last step consists in executing the model-to-text transformation to obtain the RIA implementation. The process defines a transformation that generates the RIA server-side from the domain and navigation model, and a second transformation obtains the RIA client-side from the Presentation and Orchestration models.

3 The OIDE Design and Technological Features

The OIDE is a tool developed like an Eclipse plug-in based on other open-source relevant tools in the Eclipse Modeling Project: the Graphical Modeling Framework (GMF) [3] to represent domain-specific models and the Xpand language of the Model to Text (M2T) project [1] to develop the transformations that carry out the OIDE development process. This tool is a user-friendly IDE which specifies easily and seamless the server and the client side of a RIA. Its main characteristics are: (1) a WySWyG UI presentation model which emulates the same appearance and the layout spatial distribution than a specific RIA design tool (see Fig. 1). (2) An intuitive ECA rules Tree which permits to define the UI behavior selecting events and methods of a RIA framework widgets. (3) An OCL checking which avoids introducing invalid model configurations. (4) A rapid prototyping that helps reducing the development iterations. (5) The integration of the model transformation editor which allows developers to modify them for a specific application. To do so, the tool generates a set of default transformation rules for each new OIDE project, thus allowing developers to manipulate these transformations introducing exceptions or a specific code for this project. OIDE integrates the following transformation languages provided by the Eclipse Modeling Model to Text project [1]: Xpand, a model-to-text transformation language, Xtend, a model-to-model transformation language and Check to represent OCL-like constrains. Specifically, Xpand provides two interesting features: On the one hand, a polymorphism rules invocation that allows developers to introduce new rules without having to modify the default rules provided by the tool. And the other hand, an incremental generation facility that detects which parts of a model have changed since the last generation process and determines which files need to be generated due to that change and which are unaffected by it.

To implement the OIDE DSLs, we have defined an EMOF metamodel (specifically an EMF metamodel) which establish the metaclasses, attributes and relationships between the elements of the OOH4RIA models. At this point, we use the GMF framework to generate automatically the graphical editor for each different models and the EMF to produce a set of Java classes that enable us to view and edit this

metamodel. Next, we must establish a correspondence between the relevant MOF metaclasses and a graphical element (a node or a link) using a set of XML files. This generates a graphical editor (canvas and tool) based on GEF.

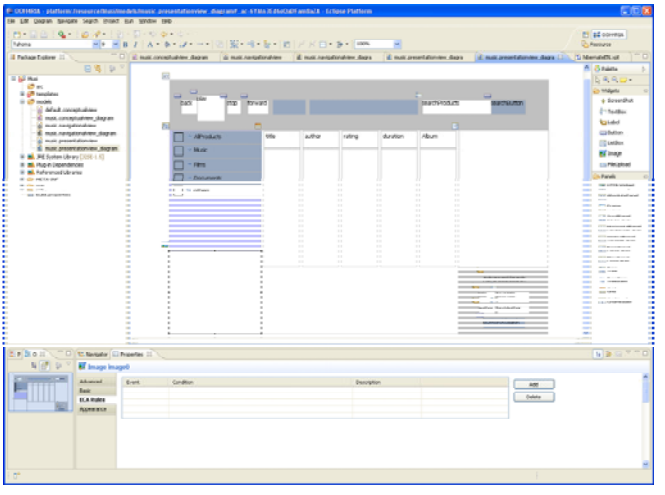

Fig. 1. A snapshot of the Presentation Model in OIDE tool

Currently, the OIDE tool permits to generate the final implementation in two alternative RIA frameworks: GWT and Silverlight. GWT is a DOM-based RIA framework developed by Google, permitting developers to program with a Java API by generating a multi-browser (DHTML and Javascript) code. Thus, OIDE integrates a GWT plug-in which permits to prototype the RIA client-side in the Eclipse IDE. On the other hand, Microsoft Silverlight is a very promising plugin-based RIA framework that integrates multimedia, graphics, animations and interactivity into a single runtime environment.

References

1. Eclipse Modeling Model to Text Project, http://www.eclipse.org/modeling/m2t/
2. Gómez, J., Cachero, C., Pastor, O.: Conceptual Modeling of Device-Independent Web Applications. IEEE Multimedia 8(2), 26–39 (2001)
3. Graphical Modeling Framework (GMF), http://www.eclipse.org/modeling/gmf/
4. Meliá, S., et al.: A Model-Driven Development for GWT-Based Rich Internet Applications with OOH4RIA. In: ICWE '08, pp. 13–23. IEEE Computer Society, USA (2008)
5. Meliá, S., et al.: Facing Architectural and Technological Variability of Rich Internet Applications. IEEE Internet Computing 99, 30–38 (2010)
6. OOH4RIA Integrated Development Enviroment (OIDE), http://suma2.dlsi.ua.es/ooh4ria/