Adapting the Presentation Layer in Rich Internet Applications

Irene Garrigós[1], Santiago Meliá [1], and Sven Casteleyn[2]

[1] Universidad de Alicante, Campus de San Vicente del Raspeig,
Apartado 99 03080 Alicante, Spain
{igarrigos,santi}@dlsi.ua.es
[2] Vrije Universiteit Brussel, Department of Computer Science, Pleinlaan 2,
1050 Brussel, Belgium
Sven.Casteleyn@vub.ac.be

Abstract. Rich Internet Applications offer Web surfers a richer user experience, mainly due to better responsiveness and enhanced user interface capabilities. In recent years, existing design methodologies targeting traditional Web 1.0 applications were extended to also support RIAs. These extensions do not yet cover all design concerns typically encountered in state-of-the-art Web applications. One yet unsupported aspect is the personalization of content and presentation to the specific user and his/her context, exploiting the extra capabilities offered by RIAs. This article addresses this hiatus and presents an extension of the OOH4RIA approach to include presentation personalization support, focusing on Rich Internet Applications.

1 Introduction

Rich Internet Applications are an answer to the growing demand for Web applications offering better responsiveness and an extended UI experience. They keep the middle between the traditionally sober (HTML-based) Web applications and the interface, interaction and functionality capabilities of traditional desktop applications.

When designing and implementing Rich Internet Applications, several new requirements and concerns come into play [1, 13], complicating the task of a Web engineer. The Web engineering community is well-aware of these difficult challenges, extending the design methodologies that target traditional Web 1.0 applications to also support RIAs [2, 5, 9, 12]. However, due to their relative recentness, these extensions do not yet cover all design concerns usually encountered in state-of-the-art Web applications. One yet unsupported aspect is the personalization of content and presentation to the specific user and his/her context, specifically for RIAs. UIs of RIAs are typically dependent on the context device rendering them and vulnerable to the limitations they impose: limited screen size, more difficult interaction and poorer multimedia support. In this paper, we aim to overcome some of these problems by personalizing the UI depending on the specificities of the device (i.e. the device context). This device context personalization must consider two important aspects: (1) an interface re-organization to fit the UI layout to the device dimensions, and (2) the

M. Gaedke, M. Grossniklaus, and O. Díaz (Eds.): ICWE 2009, LNCS 5648, pp. 292–299, 2009.

transformation of some origin widgets into specific widgets that work more efficiently in the target device.

With this goal in mind, we present an extension to an existing RIA design method called OOH4RIA [5, 8], to support personalization of the RIA user interface for different devices. OOH4RIA defines a model-driven development process based on a set of models and transformations allowing to easily introduce new concerns to the RIA development process. We thus adapt the OOH4RIA process by (1) introducing new personalization models (such a User Model and personalization rules), (2) defining transformations that reduce the effort to redefine new presentation models for each device . These extensions process allows us to obtain different device-aware versions of the same RIA project.

The remainder of this paper is organized as follows. Section 2 discusses how personalization gets differenced in RIA from traditional Web applications and outlines related approaches. Section 3 presents the extensions done in OOH4RIA to integrate personalization. Section 4 presents the main contribution of the paper: the personalization of the RIA user interface to different contexts. Finally, Section 5 provides conclusions and future research lines.

2 Personalization: From Traditional Web Applications to RIAs

Personalization has been intensively studied in traditional Web application methods. Typically, content, navigation and presentation are personalized to tailor to the specific user based on his/her preferences, characteristics, context and browsing behavior. Traditional Web applications limit the possibilities to track the user browsing to the requests performed to the server. RIAs provide new client-side capacities, new presentation features and different communication flows between the server and client side. These differences with respect to traditional Web applications must be taken into account in RIAs design, as well as in the specification of personalization strategies.

RIA applications provide richer and more interactive user interfaces, similar to desktop applications. They offer multimedia native support (i.e. no plug-ins are needed to show video and audio) and support animations. As a consequence, from a personalization point of view, the layout and look-and-feel of the application can be personalized but also the system's reaction to user interaction has to be specified accordingly. Recently, existing Web design methodologies were extended to also support RIAs. The most relevant ones are (1) OOHDM [12] which provides the use of ADVcharts to model widget interaction. (2) WebML which extends its conceptual modeling primitives for RIA's [2] and provides support for distributed event-driven RIA's and specific interaction patterns typically occurring in RIAs. (3) RUX [9], a method independent presentation framework for RIAs tackling presentational specificities of RIAs. RUX has been applied to WebML and UWE, lending its presentational capabilities to these approaches and (4) OOH4RIA which we will extend and use as a RIA method in this article.

To the best knowledge of the authors, there is only one approach [11] that provides personalization support specifically targeting Rich Internet Applications. This approach is not in the context of Web engineering and performs on-the-fly adaptation over AJAX

pages. The authors combine ontologies to annotate RIAs and adaptation rules which are derived from semantic Web usage mining techniques. This approach however, does not contemplate the personalization of the presentation features, which is exactly the focus of this paper. We thus present a personalization approach founded in a Web application method, and specifically focus on the RIA-specific elements of the presentation layer.

In the next section, we explain how to integrate personalization in the OOH4RIA development process.

3 Integrating Personalization in the OOH4RIA Development Process

OOH4RIA [5] is a model-driven approach whose main target is to cover all the phases of the Rich Internet Application (RIA) lifecycle development for a GWT-based application [4]. This paper presents an extension of the OOH4RIA development process, focusing on the models and artifacts that allow us to introduce the personalization concern into OOH4RIA. The personalization extended OOH4RIA process starts specifying the *OOH Domain Model* in order to represent the domain entities and the relationships between them. This model is the starting point of the main subprocesses: (1) the definition of the RIA server side where a model-to-text transformation generates the business logic and persistence from the domain and navigational entities, (2) the RIA user interface, defining *the OOH Navigation Model* which represents the navigation through the domain concepts and establishes the visualization constraints. Starting from the Navigation Model, the different screenshots, which represent spatial distributions of the widgets rendered in a given moment, of the Presentation Model are defined. A detailed overview of OOH4RIA can be found in [5].

The personalization extension introduced by this work begins defining *the OOH User Model* where the Domain, Navigation and Presentation models are taken as input. The User Model represents the dynamic data structures where the information about the user is stored. This information is used to personalize the website containing information about the user preferences and widget mappings of different contexts. The actual personalization is defined over the Presentation Model for different devices. To do that: we have used the marking technique defined by the MDA guide [6] to introduce information about the spatial arrangement of the layout widgets, which are to be reorganized in the target Presentation Model. This marked Presentation Model, together with the User Model, are the inputs to obtain one or more presentation models according to the devices defined by the User Model. For this aim, a set of transformation rules was defined.

On the other hand, the User Model together with the Navigation, Presentation and Orchestration models are the input that permits to define Personalization Rules, specifying runtime personalization strategies based on user preferences, goals and context. To define these rules we use the PRML language [3], which was defined in the context of OOH to extend it with personalization support. However, due to space constraints, we do not elaborate these rules in this paper; instead, we focus solely on presentation personalization targeting different devices.

The last step consists of defining the model-to-text transformations that will grant us the personalized RIA implementation. The *GWT Server Side* transformation generates the server code from the OOH Domain and the Navigation models, while the *GWT client side* transformation generates the client side code using a specific GWT framework. These model-to-text transformations are written in the MOFScript language which follows the OMG ModelToText RFP for the representation of model-to-text transformations.

4 Device Context Adaptation of the Presentation Model

In this work, we are focused on the device context personalization of the Presentation layer of a RIA, and for this purpose, we must reorganize the layout widgets in the user interface depending on the screen size, and some widgets may need to be transformed into others that better fit the new device screen dimensions.

In OOH4RIA, the layout of the RIA is represented by the Presentation Model, so the adaptation of its elements should be done in order to cope with these issues. As explained, the OOH4RIA Presentation Model is based on the GWT framework, which is composed of widgets and panels (i.e. layout widgets) where the widgets are placed. For personalization purposes, the designer has to specify how these panels and widgets are transformed and/or reorganized for the target application (i.e. specific device). For instance, one screenshot element specified in the original Presentation Model may be split into different screenshots in a mobile screen device. As already explained, we reduce the effort to redefine new presentation models for each device, including these transformations in the OOH4RIA development process.

The OOH4RIA device context adaptation is made up of following steps:

- *Marking the Presentation Model* in order to determine the spatial arrangement of the panels in the target Presentation Model
- *Define the User Model* with the purpose of specifying the target device(s) and to provide specific information on how to transform certain widget types in order to influence the set of transformation rules to be performed.

a) Marking the Presentation Model

The first step the designer has to do is to mark the Presentation Model in order to mark how the elements will be reorganized. Depending on these markings a set of transformation rules will be executed for modifying the spatial arrangement of the elements. Other rules have to be explicitly selected by the designer or are selected depending on the data stored in the User Model (this will be further developed in the ongoing section).

To allow the designer marking the elements, the metamodel of the Presentation Model is extended: each of the panels has a new attribute called *Location* which indicates whether the panel will be placed in a new screenshot or it will be shown in an existing one. The location attribute can have different values:

- **inherits:** this is the default value for all the panels. The panels are nested, all the nested panels will be placed in the same screenshot that their upper panel unless the designer specifies a different value.

- **new:** in this case the designer specifies that the panel will be placed in a separate screenshot.
- **none:** this value would be assigned when the designer wants to exclude the panel, so it will not be visible (and all what is in it) from the target application.
- **all:** the designer would assign this value when he wants to include this panel in all the screens of the target application.
- **containerID:** the designer may also want to show the panel within of another concrete panel of the website.

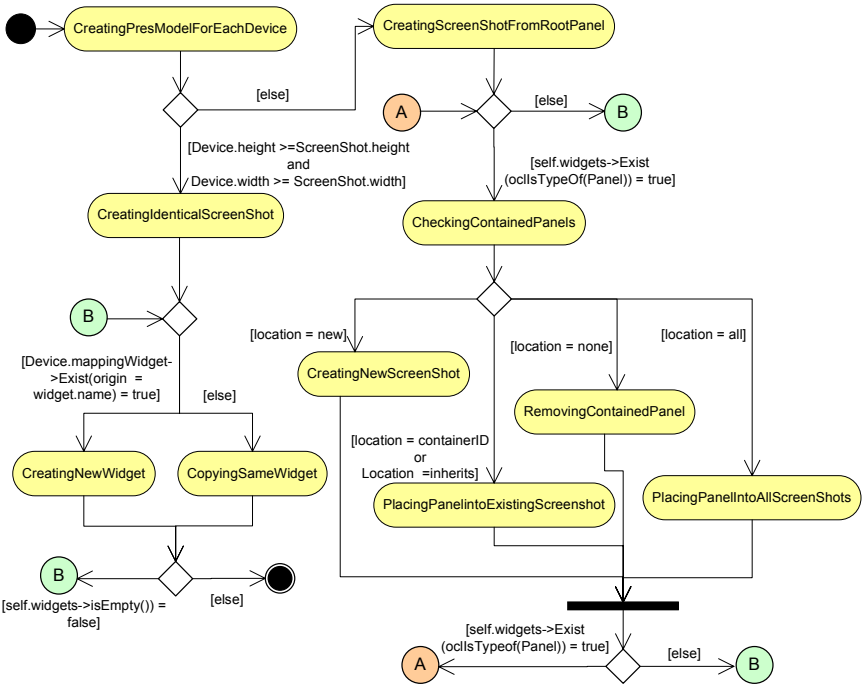

Fig. 1. Rule Map of the ObtainSpecificDevicePres QVT Transformation

Depending on the markings done in the Presentation Model different transformation rules are to be performed. They allow to convert a generic Presentation Model into a specific device Presentation Model. Fig. 1 presents an activity diagram that establishes the execution workflow of the transformation rules defined for this purpose. The execution starts with the root rule called *CreatingPresModelForEachDevice* which creates the Presentation Model element for each Device defined in the User Model. When the dimensions (height and width) of the device are larger than the definition, the transformation invokes the *CreatingIdenticalScreenShot* rule which creates Screenshots identical to the destination model. On the contrary, if the device dimensions are smaller, then the *CreatingScreenShotFromRoolPanel* rule establishes a Screenshot from the container panel with the dimensions adjusted to the device.

Here begins the reorganization of the containers or panels where the transformation checks whether the root panel contains in turn inside panels. If this is the case, the CheckingContainedPanels rule is executed and it decides the destination of the panel according to the value of the location attribute. (1) If location is equals to *new* then the panels requires a new Screenshot, thus executing the CreatingNewScreenshot rule. (2) If the location is equal to the *ID* of a pre-existing panel or is equal to intherits then a new Screenshot will be created within it. (3) However, if we want to eliminate the panel (location equal to *none*), we execute RemovingContainedPanel rule. (4) Finally, if we want the panel to appear in all the Screenshots (location equal to *All*), the PlacingPanelScreenshot rule is executed.

b) Specifying the User Model

In the User Model, information regarding the user characteristics, interest, preferences or context is stored. In this case we store information regarding the device context of the user.

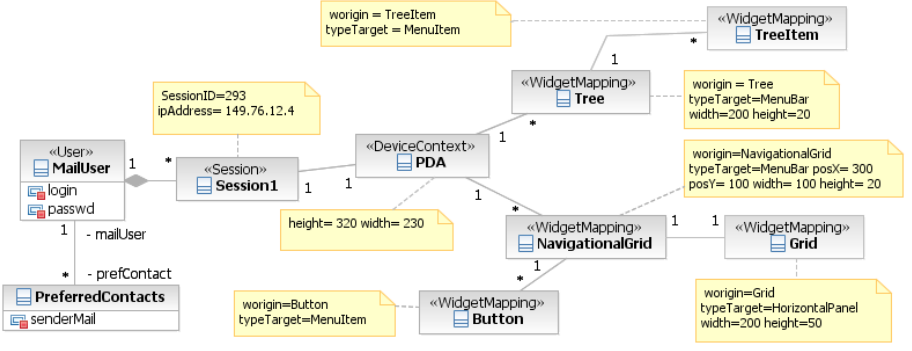

Fig. 2. The User Model of GWT Mail application

In Fig. 2 we can see the User Model needed for defining a sample mobile-aware RIA. In order to deal with the personalization at widget level, the personalization designer must introduce the *WidgetMapping* concept into the User Model which proposes the widget conversion to another widget, giving it similar functionality in the target device. Thus, by defining the User Model, the designer provides information that influences which set of rules is to be executed.

Let's recover the *Pres2DevicePres* transformation at point B where the transformation of simple widgets starts. Here, the transformation checks if there is a WidgetMapping into the User Model for the current Widget, if this is the case, the *CreatingNewWidget* (Fig. 3) rule is executed converting a Widget into another one. However, when there is not a WidgetMapping defined, the original Widget is copied into the target Presentation Model.

Figure 3 presents the *CreatingNewWidget* rule which converts a Widget into another one gathering the information from the WidgetMapping defined by the User Model. Firstly, this rule checks that the source Widget is not a panel in the *When* sentence. From here, the rule creates a new widget that maintains the same name, position and isDisable properties. However, the rule introduces the personalization

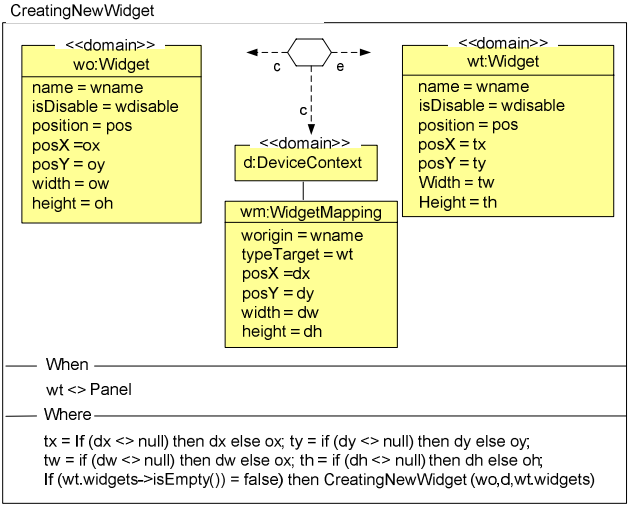

Fig. 3. Example of Pres2Device: CreatingNewWidget QVT Transformation Rule

information from the User Model (see Fig. 2), where the WidgetMapping defines a new Widget by means of the typeTarget attribute, and establishes the new location of the widget with the posX and posy, and the new dimension with height and width attributes. Finally, the rule checks if the Widget contains other nested Widgets in the Where clause of the QVT rule, in this case, this rule is invoked recursively in order to transform the contained widgets.

5 Conclusions and Future Work

In this paper, we presented, in the context of the existing Web design method supporting Rich Internet Applications OOH4RIA, a personalization approach specifically targeting the enhanced presentational capabilities of RIA's. We elaborated on the models and artifacts needed to support personalization in the overall RIA design process. Our approach consists of two steps. During the first step, the personalization designer marks in the Presentation Model which elements will be subject for transformation and what will be their target designation. The second step consists of rule selection. This is done partly automatically, for general rules, based on the information specified in the User Model.and partly manually, for specific rules that will personalize the interface at runtime for each specific user which is out of the scope of the present paper.

Currently, we are developing the OOH4RIA tool which is based on the Eclipse Graphical Modelling framework (GMF). This tool is being completed with the specified personalization transformations presented in this work. Furthermore, we are working on defining the transformation rules that should be performed over the Orchestration Model to complement the work described here.

Acknowledgements

We would like to thank our colleague Sandy Pérez for his pointers on implementation issues and his comments on the work presented.

This work has been co-supported by the ESPIA project (TIN2007-67078) from the Spanish Ministry of Education and Science and the DEMETER (GVPRE/2008/063) project from the Valencia Ministry of Enterprise, University and Science (Spain).

References

1. Bozzon, A., Comai, S., Fraternali, P., Carughi, G.T.: Conceptual Modeling and Code Generation for Rich Internet Applications. In: 6th International Conference on Web Engineering (2006)
2. Comai, S., Carughi, G.T.: A Behavioral Model for Rich Internet Applications. In: Baresi, L., Fraternali, P., Houben, G.-J. (eds.) ICWE 2007. LNCS, vol. 4607, pp. 364–369. Springer, Heidelberg (2007)
3. Garrigós, I. A-OOH.: Extending Web Application Design with Dynamic Personalization, Phd thesis, University of Alicante (2008)
4. Google. Google Web Toolkit (GWT), `http://code.google.com/webtoolkit`
5. Meliá, S., Gómez, J., Pérez, S., Diaz, O.: A Model-Driven Development for GWT-Based Rich Internet Applications with OOH4RIA. In: Eighth International Conference of Web Engineering, Yorktown Heights, USA (2008)
6. Object Management Group (OMG). MDA Guide (version 1.0.1) (June 2003), `http://www.omg.org/docs/omg/03-06-01.pdf`
7. Object Management Group (OMG). Software Process Engineering Metamodel (version 1.1) (January 2005), `http://www.omg.org/docs/formal/05-01-06.pdf`
8. Pérez, S., Díaz, O., Meliá, S., Gómez, J.: Facing Interaction-Rich RIAs. In: The Orchestration Model Eighth International Conference of Web Engineering, Yorktown Heights, USA (2008)
9. Preciado, J.C., Linaje, M., Comai, S., Sánchez- Figueroa, F.: Designing Rich Internet Applications with Web Engineering Methodologies. In: 6th International Conference on Web Engineering (2006)
10. Rossi, G., Urbieta, M., Ginzburg, J., Distante, D., Garrido, A.: Refactoring to Rich Internet Applications. A Model Driven Approach. In: Proceedings of the Eighth International Conference of Web Engineering, ICWE (2008)
11. Schmidt, K., Stojanovic, L., Stojanovic, N., Thomas, S.: On Enriching Ajax with Semantics: The Web Personalization Use Case. In: Franconi, E., Kifer, M., May, W. (eds.) ESWC 2007. LNCS, vol. 4519, pp. 686–700. Springer, Heidelberg (2007)
12. Urbieta, M., Rossi, G., Ginzburg, J., Schwabe, D.: Designing the Interface of Rich Internet Applications. In: 5th Latin American Web Congress (2007)
13. Wright, J.M., Dietrich, J.B.: Requirements for Rich Internet Application Design Methodologies. In: Bailey, J., Maier, D., Schewe, K.-D., Thalheim, B., Wang, X.S. (eds.) WISE 2008. LNCS, vol. 5175, pp. 106–119. Springer, Heidelberg (2008)