Business Process -based Conceptual Design of Rich Internet Applications

Marco Brambilla[1], Juan Carlos Preciado[2], Marino Linaje[2], Fernando Sanchez-Figueroa[2]
[1]Politecnico di Milano. Dipartimento di Elettronica e Informazione. Milano (Italy)
[2]Escuela Politécnica. Universidad de Extremadura. Cáceres (Spain)
mbrambil@elet.polimi.it; {jcpreciado; mlinaje; fernando}@unex.es

Abstract

This paper presents a methodology and a mix of conceptual models for addressing design and development of Web applications supported by rich interfaces. For specifying the high level design of the user tasks, we exploit business process models. In particular, we describe how to model the business process, transform it into data and navigation model of a Web application, and apply a presentation model for obtaining a Rich Internet Application (RIA). A standard business modeling language (BPMN) is used for describing workflows, which are then translated to a WebML specification of a Web application implemented according to the Single Page Paradigm, typical of RIAs. Finally, by integrating the RUX-Method features, refined Rich Interface design can be achieved.

1. Introduction

Rich Internet Applications (RIAs) are Web applications that exploit the power of Web clients for increasing the responsiveness and usability of the Web user interfaces (UI), by offering functionalities similar to the ones of desktop applications. RIAs follow the client/server paradigm, but opposite to traditional Web applications, RIAs are able to transfer the processing of UI, business logic, and data management to the client, possibly using asynchronous communications.

The relevance of RIAs is continuously increasing, because users are becoming accustomed to good graphical appearance, advanced interaction patterns, and refined application behaviours on the Web. Thus, developers are nowadays requested to provide rich interaction on almost any Web application.

RIAs bring also great value in terms of productivity and efficiency of navigation. This is now fundamental because a lot of Web applications implement enterprise management systems, intranets, and extranets, where productivity is critical. Such systems typically implement strict business processes that dictate the user behaviour. Typically, this class of applications can be designed at an abstract level in terms of business process models. Our proposal consists of both a methodology and a mix of conceptual models for addressing design and development of business process-based Web applications supported by rich interfaces. Our contribution describes how to model the business process specified with BPMN, transform it into a WebML [3] specification of data and navigation models, and apply the RUX-Method [7] presentation model for obtaining a Rich Internet Application (RIA).

The paper is organized as follows: Section 2 presents the basics of RIAs; Section 3 describes how to exploit business processes in RIA design; Section 4 explains how BPMN models are transformed in WebML and RUX models; Section 5 presents a running case scenario; Section 6 describes the related work; and Section 7 concludes.

2. Basic concepts of RIAs

RIAs can be implemented using different techniques and development languages, that exploit a plethora of frameworks (FLEX, AJAX, LASZLO, XUL, XAML) helping the developers in sophisticated applications. The richness in RIA extends (at client and server side) the four main aspects of traditional Web applications [8]:

- **Data.** Through RIAs, client's memory is available to be used by the application; persistent and volatile content can be stored and manipulated on the client and finally sent to the server once the whole operation has been completed.
- **Business Logic.** RIAs have a different navigation structure from Web 1.0 applications. Due to the augmented process capability of the client, in RIA both the client and the server can carry out complex operations.

- **Communication.** RIAs allow both synchronous and asynchronous communications. Distribution of data and functionality across client and server broadens the features of the produced events as they can originate, be detected, notified, and processed in a variety of ways. Pull and push communication capabilities are available.
- **Presentation.** RIAs offer improved presentation and user interactions (e.g. multimedia support and drag&drop). They can operate as single page applications, avoiding unnecessary refreshments of the whole page and allowing progressive presentation loading when needed.

3. Representing RIA modeling foundations with BPMN

For specifying business process models, we adopt the OMG Business Process Management Notation (BPMN) and its companion metamodel Business Process Definition Metamodel (BPDM). Workflows are based on the concepts of *Process*, *Case* (a process instance, i.e., an execution), *Activity* (the elementary unit of work), *Activity instance* (an instantiation of an activity within a case), *Actor* (a user role), *Event* (some punctual situation that happens in a case), and *Constraint* (logical precedence among activities and rules enabling activities execution). Processes can be structured using a variety of constructs: sequences of activities; gateways implementing AND, OR, XOR splits, respectively realizing splits into independent, alternative and exclusive threads; gateways implementing joins, conditional flows, and loops.

Due to the high level of abstraction of BPMN, its notation is suitable to many purposes. The main idea of this work is to provide an additional abstraction level to be used over existing Web modeling approaches (such as OO-H [5], OOHDM [11], WebML [3], UWE [12], and others) to represent the behaviour of RIAs for the aspects not available in traditional Web engineering.

We exploit the BPMN specification to generate the design of data, business logic, communication, and presentation. We provide a special interpretation of the BPMN models that fits the needs of RIAs and we define a translation to the various aspects of the application. Separation of concerns is preserved in the separated models used for process, content, navigation, and presentation design.

One of the issues to take into account when creating BPMN diagrams is the level of granularity to be applied. For our purposes, an intermediate level of abstraction is required: activities and events specified in the business model should be the ones that trigger the behaviour of the business logics and of the presentation components.

The first purpose of the BPMN models used for RIAs is to specify the distribution of the computation and data storage between client and server. To achieve this goal, pools and lanes must be organized as follows: a pool represents an actor/role of the Web application. Each pool is composed a set of predefined lanes that must be marked as: server, client, and (optionally) client-server. *Client* and *server* lanes will comprise aspects that will stand/occur just in one of the sides (client or server respectively), while the *client-server* lane may be used to represent common issues.

3.1. Data modeling

BPMN *data objects* can be exploited for inferring the basic information concepts in the data model of the application. Data distribution (client or server) and persistence (persistent or volatile) can be described in the BPMN: data objects can be represented inside the client or the server lanes; while persistence is described by the *state* attribute (‘P’ or ‘V’) within the data object. Persistent data objects will be tagged with ‘P’, while volatile data will be tagged as ‘V’. When the same name is used by two data objects, these data objects are considered the same (names are repeated only for graphical convenience). If the same data object has two different persistence properties, it can be put twice in the diagram, with different markers.

3.2. Business Logic

BPMN allows representing the business logic by using the control flow (arrows), connection objects, artifacts, and properly configured lanes. *Client* and *Server* lanes indicate those elements that will stand/occur at one of the sides, while mixed business logic can be specified in the *Client-server* lane. The client-server lane is useful for two different scenarios: firstly, when the application needs to perform the same operation twice (once at client and later at server side); secondly, on BPMN collapsed sub-processes where their expansions involve processes both in the client and in the server lanes.

To establish the relation between the activities and the data we use the standard relationship proposed by BPMN through *associations*. Directionality (depicted by an arrow) added to an association shows whether the data object is an input or an output to the activity. Three types of association are available: an activity can read (incoming arrow), write (outgoing arrow from activity to data), or read/write (no arrows).

3.3. Communication

To represent the complex communication patterns that may appear into RIAs, we focus in the native communication mechanisms that these applications provide. We must notice that in RIAs the servers are also able to begin a conversation, by means of push communication paradigms (obviously, the traditional client-server pull communication is allowed too).

BPMN *control flow* arrows can be used to describe communication between activities of the processes. From a technical point of view, the only interesting communications are the ones traversing the border between the client and the server. The kind of communication (push or pull) can be immediately derived from the direction of the arrow from the origin to the target. Control flows can be refined by a *text annotation* specifying whether the transmission is synchronous ('S') or asynchronous ('A').

Another facet of communication is related to the information shared or transmitted between different actors of the application. This addresses collaborative applications. Although we are currently investigating the use of *message flows* at this purpose, this issue is out of the scope of this paper.

4. Generating the WebML+RUX models from BPMN

- In our proposal, the information specified in the BPMN business process is used by the WebML model as a high level specification of the hypertext behaviour. The design steps can be summarized as follows: (1) design of the BPMN model; (2) automatic generation of preliminary data model and navigation model, which need further refinement by the designer; (3) extraction of the abstract RUX-Method user interface model; (4) definition of the RUX-method concrete model; (5) implementation of the application through automatic code generation .

If needed, the design can re-cycle on one or more steps.

4.1. WebML Essentials

WebML (Web Modeling Language) is a suite of models that cover the specification of the contents, business logics, hypertexts, and of the presentation of a Web application. The content can be modeled using E-R diagrams, UML class diagrams, or ontologies. Upon the same data model different hypertext models (site views targeted to different user roles) can be defined. A site view is a graph of pages, possibly organized into sub-pages. Pages comprise units, i.e., components for publishing database contents. Units are connected to each other through links, which carry parameters and allow the user to navigate the hypertext. WebML also allows specifying operations implementing arbitrary business logic (e.g., to manipulate content instances) and Web service invocation and publishing. In this work we exploit the approach that map business processes to Web application models [2], that defines:

- A standard content metamodel for recording activities, activity instances, and process cases;
- Some new hypertext primitives for describing the workflow behaviour. Activity areas represent hypertext fragments implementing a workflow activity. Start link and End link declare the entry and exit points of activity areas, update the status of the execution and check its correctness.
- A set of automatic rules that transform BPMN workflow models into WebML hypertexts

To manage the peculiar characteristics of RIAs, some aspects must be changed in the existing WebML methodology for business-process based applications.

4.2. WebML Workflow Data Model

The business process metamodel is composed by a set of general purpose entities and relationships that are needed for any kind of application [2], extended with some new properties that are specific for RIAs. All this entities and relationships are depicted in Figure 3.

The *Process* describes a business process type and is associated with entity *ActivityTypes* that can be executed in a process. A *Case* is an instance of a Process, and is composed by the *ActivityInstances* denoting the occurrences of activities in the execution. Every *ActivityInstance* is associated with one *ActivityType*. *User* and *Group* represent the process actors, as individuals clustered in groups. *ActivityType* is related to entity Group to denote that the users of the group are entitled to perform that kind of activity, while *ActivityInstances* are associated with individual users that are assigned to execute them and that actually perform them. Satus attributes store the execution status of *Case* and *ActivityInstances*.

The additional information that is needed for developing RIA applications is shown in underlined bold font in Figure 1: *ActivityType* entity includes two new attributes: *CreationTime* that specifies when an activity instance will be created, and *Visibility* that specifies when an activity will be shown. The value of the *CreationTime* attribute can be *ProcessStart* (the instances of the activity will be created when the case starts) and *OnExecution* (the instances will be created when it has to be executed). The value of the *Visibility* attribute can be *BeforeActive* (the instances of the activity will be visible to the user before or during the execution of the activity), *OnlyActive* (the instances will be visible to the user only when active), *AfterActive* (the instances of the activity will be visible only when active or completed), *Always* (the instances of the activity are always visible). Notice that some constraints hold between the values of the two attributes. If *Visibility* is set to *BeforeActive*, the *CreationTime* must be *ProcessStart*, to allow showing the activity before its activation. The *ActivityInstance* entity includes the new *Visible* attribute, that states whether the activity instance is currently shown.

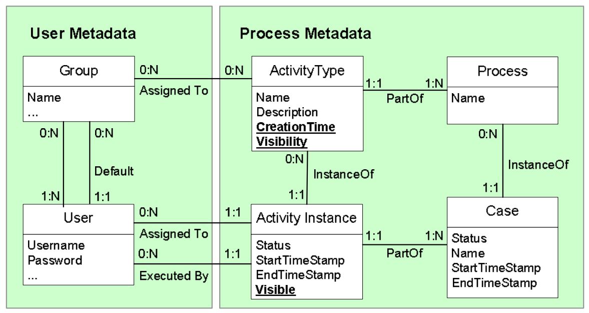


Figure 1. Process metadata model (RIA specific information is in bold red font).

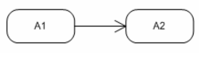
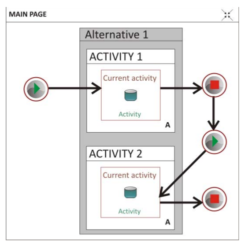
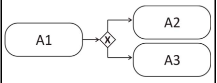
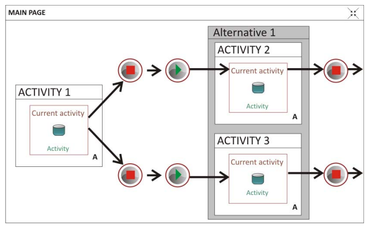
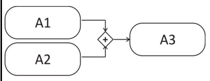
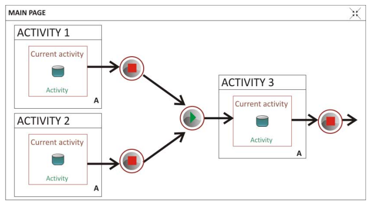
4.3 WebML Hypertext Model Updates

The WebML hypertext model, even if extended with workflow-enabling capabilities, still misses some important aspects for supporting full fledged RIAs.

The first need is multithreading in the execution of server-side operation chains. Currently, within a siteview, WebML allows to trigger operation chains only by means of user clicks that can trigger only one execution path, which may include branching points but not execution forks. Instead, actual multithreading must be supported now to allow execution of several activities at the same time at server side.

Another important aspect is the granularity of the Activity in the hypertext model. Up to now, Activity elements in WebML can only be hypertext areas. An activity may include one or more pages, but the minimum granularity is of 1 page (or one operation) per activity. Correspondingly, start and stop links can be defined when traversing an Activity Area border. Viceversa, RIAs implemented in a Single Page Paradigm require activities to correspond to pieces of content smaller than one page. Therefore, hypertext activity must be available also at subpage level. A single page can include several *activity subpages*.

Table 1. Translation of Workflow patterns

BPMN pattern	WebML model correspondence
1. Sequence 	 MAIN PAGE Alternative 1 ACTIVITY 1 ACTIVITY 2 Current activity Activity A
2. XOR split 	 MAIN PAGE Alternative 1 ACTIVITY 1 ACTIVITY 2 ACTIVITY 3 Current activity Activity A
3. AND join 	 MAIN PAGE ACTIVITY 1 ACTIVITY 2 ACTIVITY 3 Current activity Activity A

4.4 BPMN-WebML Translation

The translation of BPMN models into WebML hypertext and navigation models can be achieved with the approach proposed in [2]. However, new translation rules must be defined for supporting RIA features. The new translation covers the new primitives previously introduced. Table 1 presents the translation of some of the most used business process patterns (sequence, split, and join). For space reasons the translation of the whole set of pattern is only available online at [13].

The translation of elementary concepts is quite straightforward: a pool represents the behaviour of the system with respect to a single user (with the two lanes *Client* and *Server*). For each pool a single siteview is created and composed of one page (Main Page) according to the single page application paradigm. Both client and server activities, spread on the two corresponding lanes, will be placed inside the single page. An activity is represented: *(i)* as a subpage marked with an “A” symbol (Activity) within the main page, if it contains user interaction elements; or *(ii)* as an operation chain, if it only contains execution logic.

Control flows are translated as a combination of start and stop links that indicate the beginning and the end of the activities. Events are not fully exploited in the translation: only start event are currently transformed in a special Start link that starts the first activity of the process and also the process case. This can then be exploited as a special event in the presentation model.

The translation of the patterns (Table 1) covers a subset of the large set of BPMN possible patterns (for further details on these limitations see [2]). For instance, all the activities of a sequence flow (1) become mutually exclusive subpages. An activity instance can be automatically hidden according to its current execution status and to the value of the *Visible* attribute. Thanks to the alternative element and to the start/stop links, an activity will be started if and only if the previous activity is completed. A *split* (branching or parallel execution) can be implemented with two possible hypertext behaviours: the first (not shown in the table) envisions an automatic choice of activities by the system without involving the user. In this case, only one anchor exists in the page and a complex logic on the link leads to the proper executions; the second (2) allows the user to manually activate each single activity, by clicking on the proper link. *Join* points are managed based on the kind of gateway: AND joins must check that all the parallel branches are completed before activating the next activity, while OR or XOR branches can just allow the user to proceed. From the same BPMN model, different hypertexts can be generated. For instance, if the activity must be always visible (*Visibility* attribute), no alternative pages will be used.

4.5 WebML and RUX-Method integration for presentation specification

The WebML presentation model, as well as many other Web models, is not able to deal with all the rich features of RIA interfaces. Therefore, we integrate it with the RUX-Method, a visual method that allows model-driven design of multimedia, multimodal, multi-device, and multi-platform UIs for RIAs. RUX-Method is a set of multi-level user interface models with distinct responsibilities and abstraction levels. RUX-Method must be used over a Web model that provides content and functionality to the Web application. Currently, RUX-Method is being used over some well-know Web models (e.g., WebML or UWE) and, conceptually, can be used over other ones. The RIA UI development process in RUX-Method has four main stages: connection with the previously defined Web model (called Connection Rules), definition of the Abstract Interface, definition of the Concrete Interface, and specification of the Final Interface, which ends in UI code generation.

The Connection Rules are composed of a set of transformation rules specific for the adopted Web model. The objective of the Connection Rules is to extract the available information in the Web model in order to build a first version of the Abstract Interface. In previous works with WebML, Connection Rules have been used for extracting the information only from the hypertext model [9]. The RIA specification proposed in this paper requires minor enhancements in the Connection Rules to take advantage of the new information offered by the business process model:

- The business logic distribution (client vs. server) must be extracted from the BPMN. Server-side operations are supported by RUX-Method CALLActions, while functions at the client-side can be based on the RUX-Method OCLActions.
- The communication (both asynchronous and synchronous) aspects can be extracted from the BPMN flow, instead of specifying it at the RUX-Method Concrete Interface level.

Data duration and distribution are not taken into account by the RUX-Method, because they are directly managed by the business logic layer. For more information about RUX-Model Interactive Presentation we recommend to read [7].

5. Case Study

In order to explain the design concepts described in this document, we introduce an example based on the Adobe Gadget Store demo Web application (`http://flexapps.macromedia.com/flex15/flexstore/flexstore.mxml`) that allows exploring and buying products. It offers a rich interaction (e.g., drag&drop) and advanced presentation features (e.g., partial page refresh) typical of RIAs. The main page of this application is structured in three parts (Figure 2): on the left side (1) a product list on which filters, searches and selections can be applied; on the right side (2) the details of the selected product; and at the bottom (3) the shopping cart. For space reasons, we cannot discuss the whole application design. Figure 3 depicts only the BPMN diagram for the subprocess called “Checkout”, that is located completely at the client side. Three activities can be performed in parallel or sequence (General Info, Shopping Info, and Payment Info) and finally checkout is completed.

The data model of the application is shown in Figure 4: the *Order* of the user is described by the chosen *ShipmentOptions* and by its *OrderLines* that contain the quantities and the product IDs. The *Order* is connected to the owner *User* and to the *Activity* of the workflow that manipulates it.

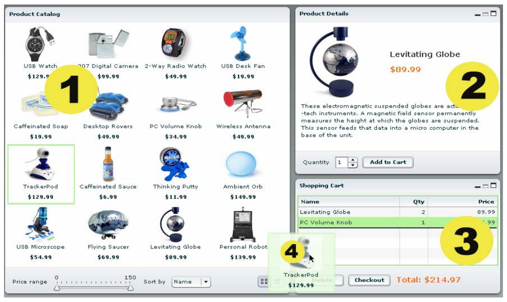


Figure 2. Gadget Store case study application.

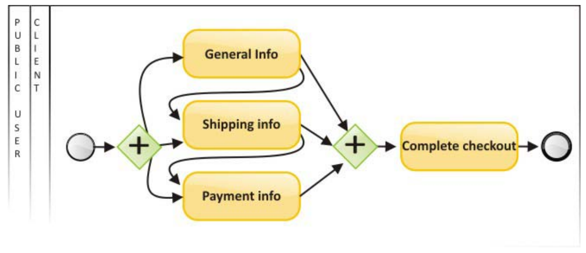


Figure 3. Detailed diagram of the "Checkout" BP.

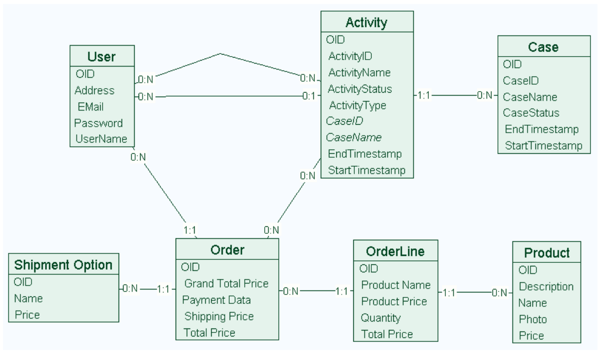


Figure 4. Client-side data model of shopping cart.

After the business process and the data model specification, the design focuses on the hypertext navigation model. The coarse navigation design can be automatically derived from the business process, according to the rules presented in section 4.4.

The final result of the navigation model (already refined by the designer after the automatic translation) is shown in Figure 5. The translation is obtained as follows: the start event becomes a *Start Case* link; the three parallel activities are rendered as three pages (marked with "A") within an alternative page; the sequential behaviour is implemented by the *Continue* links, directly connecting an activity to the following one, through a *Stop* action on the current activity and a *Start* on the next one (links on the top of Figure 5). The parallel behaviour is implemented by the remaining links: the *Complete Purchase* links on the top lead to the Complete Checkout activity (not shown), whose starting is conditioned on the completion of all the three parallel activities. For moving from one parallel activity to the other the *Parallel Activity* links are designed on the bottom part of the figure.

All these features can be generated automatically by the translation of the workflow and do not need refinements or updates by the designer, while the remaining details are specified by the designer in the WebRatio environment. In this example, forms are located into the pages and also contain hidden fields for tracking information submitted in the other activities. This allows the final validation of the submitted data, which is finally performed only by the Complete Purchase links. When exiting each activity, the submission is saved by proper operations. The main page also contains a fixed part that displays the current user and the overall status of the order.

For the presentation model, RUX-Method is able to extract the information available in the WebML specification (Section 4.5) to build a first version of the abstract interface of the application by means of the Connection Rules. This first Interface level is depicted in Figure 6, where alternative and simple views are shown for the check-out process. Connectors are depicted as boxes with a white circle. Text elements for input (incoming arrows) and output (outgoing arrows) are represented as light boxes with a "Aa" label.

The second step is the applications of the Transformation Rules that provide a draft version of the Concrete Interface, which needs to be refined by the modeller, who can select the interface components to use. Operation chains can be triggered by normal WebML links or by Start events when a process needs to be triggered (e.g. the Start Case Link in Figure 5). RUX-Model is able to specify both kinds of triggers using the CALLActions primitive.

When using the RUX-Method CASE tool (RUX-Tool [10]), the first step can be performed automatically and the second one semi-automatically.

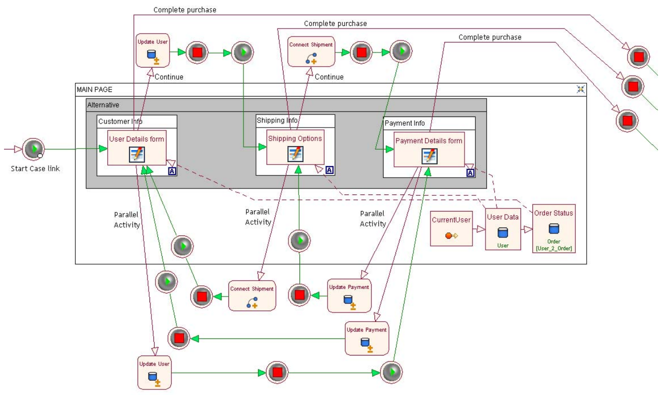


Figure 5. Hypertext model of the Checkout.

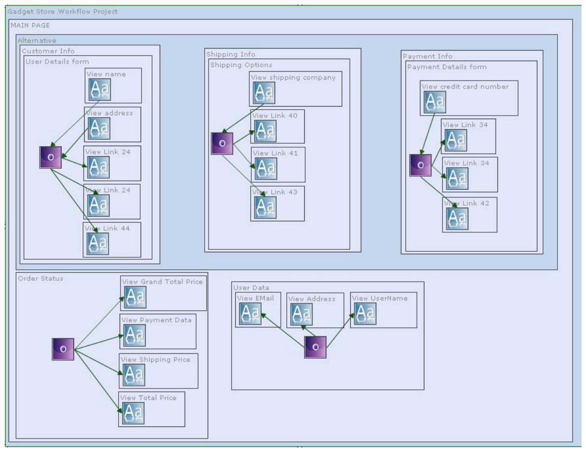


Figure 6. RUX-Method Abstract Interface model.

6. Related work

The main innovation of our approach is the exploitation of BPM for RIA design, allowing partial RIA code generation using WebRatio and RUX-Tool with a MDA life-cycle, a missing feature in the related work, with the exception of [1]. With respect to [1], we address a better separation of concerns regarding the data and business logic distribution due to the introduction of business processes for those issues. However, a missing feature in [1] is RIA communication specification.

As [4] claims, presentation model is an aspect which has been so far ignored in the Web Engineering, where many hypertext/navigation models mix UI presentation features. Many RIA UI capacities like animations or progressive downloads are missing also in [1]. RUX-Method allows a more refined and independent specification of the presentation model.

Our approach is also different with respect to other recent proposals in the Web Engineering field to represent the RIA foundations (e.g. [12]), because we include a more abstract level through the introduction of business processes, which are used specifically for RIAs, while other works (e.g., [6]), generically introduce business processes for Web applications.

7. Conclusions

In this paper we presented a methodology and a set of conceptual models for addressing design and development of RIAs by means of a high level design view that exploits business process models. We have shown how BPMN can be translated to a WebML specification of a single-page paradigm Web application. Then, by integrating the RUX-Method features, we have shown how refined rich interface implementation can be achieved.

The proposed mix of models allows a fine grained design of rich interaction. Due to the generality of the approach, the business process-based design can be applied to other hypertext navigation models and other presentation models, by simply specifying the new mappings.

8. References

[1] Bozzon A., Comai S., Fraternali P., and Toffetti Carughi G., "Conceptual Modeling and Code Generation for Rich Internet Applications", *International Conference on Web Engineering*, Springer, 2006, pp. 353-360
[2] Brambilla M., Ceri S., Fraternali P., and I. Manolescu, "Process Modeling in Web Applications", *ACM Transactions on Software Engineering and Methodology (ACM TOSEM)*, vol. 15, no. 4, 2006, pp. 360-409
[3] Ceri S., Fraternali P., Bongio A., Brambilla M., Comai S., and Matera M., *Designing Data-Intensive Web Applications*, Morgan Kauffmann, San Francisco, 2002
[4] Daniel F., Yu J., Benatallah B., Casati F., Matera M., and Saint-Paul R., "Understanding UI Integration: A Survey of Problems, Technologies, and Opportunities", *IEEE Internet Computing*, vol. 11, no. 3, 2007, pp. 59-66
[5] Gómez J. and Cachero C., "OO-H Method: extending UML to model web interfaces", *Information modeling for internet applications*, Idea Group Publishing, 2003
[6] Koch N., Kraus A., Cachero C. and Meliá S., "Integration of Business Processes in Web Application Models", *Journal of Web Engineering*, Rinton Press, vol. 3 is.1, 2004, pp. 22 - 49.
[7] Linaje M., Preciado J.C., and Sánchez-Figueroa F., "Engineering Rich Internet Application User Interfaces over Legacy Web Models", *Internet Computing*, vol.11, iss. 6, 2007, pp. 53-59
[8] Preciado J.C., Linaje M., Comai S., and Sanchez-Figueroa F., "Designing Rich Internet Applications with Web Engineering Methodologies", *International Symposium on Web Site Evolution*, IEEE, 2007, pp. 23-30
[9] Preciado J.C., Linaje M., and Sánchez-Figueroa F., "An approach to support the Web User Interfaces evolution", *Adaptation and Evolution in Web Systems Engineering ICWE Workshop on*, Springer, 2007, pp. 94-100
[10] RUXProject Homepage: www.ruxproject.org
[11] Schwabe, D., Rossi, G. and Barbosa, S., "Systematic Hypermedia Design with OOHDM", *International Conference on Hypertext*, ACM Press, 1996, pp. 116 - 128
[12] Urbieta M., Rossi G., Ginzburg J. and Schwabe D., "Designing the Interface of Rich Internet Applications", *Latin-American Conference on the WWW*, IEEE, 2007, pp.144-153
[13] WebML, Mapping BPMN - WebML, full description http://home.dei.polimi.it/mbrambil/RIA/bpml2webml.pdf