Developing Semantic Rich Internet Applications Using a Model-Driven Approach

Jesús M. Hermida, Santiago Meliá, Andrés Montoyo and Jaime Gómez

Department of Software and Computing Systems, University of Alicante
Apartado de correos 99, 03080 Alicante, Spain
{jhermida, santi, montoyo, jgomez}@dlsi.ua.es

Abstract. At present, the Web sometimes behaves as a heterogeneous mixture of several technologies with distinct purposes. On the road to the so-called Web 3.0, it is essential to integrate and harmonize techniques and technologies from its main branches. In this context, this paper focuses on two of these: Rich Internet Applications (RIA) and the Semantic Web. Although RIAs are the most distinctive and used types of Web 2.0 applications, owing to their intrinsic characteristics, some type of clients such as, search engines and readers for disabled people cannot explore a relevant subset of them. Semantic Web technologies can be the key for opening RIA contents to any client. Specifically, the paper proposes the concept of Semantic RIA as an extension of traditional RIAs that can reuse and share structured knowledge on the Web. Furthermore, it presents S^{m}4RIA, an extension of OOH4RIA that supports and speeds up the development of these applications.

Keywords: RIA, Semantic Web, MDE, OOH4RIA, S^{m}4RIA, Linked Data.

1 Introduction

During the last decade, the Web has experienced a continuous evolution from several viewpoints that led into different branches with overlapping aims, oriented to satisfy the more and more complex necessities of knowledge and services of the end users within a context in which the value of information is increasing endlessly.

At present, there exist three main evolution branches of the Web tree that emphasise different aspects: *(i)* the Semantic Web [1], which proposes a set of techniques to provide explicit, disambiguated meaning to the information on the Web for user agents; *(ii)* the Social Web, as a part of the so-called Web 2.0 [20], in which the end users gained importance as content providers (e.g. social networks, blogs, wikis, etc.); and *(iii)* Rich Internet Applications (RIAs), which embraced Web applications with desktop-like interfaces that embed multimedia elements and minimise the client-server communication. In this context, the concept of Web 3.0 (or Social Semantic Web [19]) as a combination of these branches is beginning to emerge even though the term remains ambiguous. However, this technological and social

evolution of the Web is not harmonized and, therefore, some advances produced in one branch not always yield profits in the rest.

To be specific, our work proposes a harmonized evolution in which one important branch, in this case, the Semantic Web (together with one of its most active branches, i.e. the Web of Linked Data [3]), can solve the most important lacks of another, RIAs. Despite the breakthroughs brought by RIAs, they also reintroduced an involution in some aspects already solved in the traditional Web interfaces, such as Web indexing and accessibility. The current solutions to these issues are not completely satisfactory since they cannot be adapted to plugin-oriented RIAs, which are indeed processed by means of an interpreter, such as Flex, OpenLazlo or Silverlight.

The lack of HTML code in RIAs has a strong, negative influence on some of the most common applications of the Web. The current search engines (Google, Yahoo!, Bing) are not capable of indexing the mentioned subset of RIAs correctly. This limits the quality of web searches and, thus, prevents several users from finding them and reading their contents, which is a cyclic problem because it also prevents developers from creating new RIAs. Furthermore, this problem is partially shared with those clients used by disabled people. At present, there is no standard client that can interpret the contents of any type of RIA. Clients are tightly coupled to the technology of the RIA, i.e. each family of RIAs needs of its specific type interpreter.

To overcome these deficiencies and provide users a better experience, this paper presents a new type of RIA, called *semantic RIA*, which extensively uses Semantic Web technologies for overcoming the problematic issues just mentioned. From our viewpoint, ontologies [24] and Linked Data principles [3] have to be two pillars of semantic RIAs that can actually provide an accurate, disambiguated representation of their contents. At present, there exist some approaches [23,26] that principally focus on browser-oriented RIAs (e.g. AJAX), but they do not address the problem of plugin-oriented RIAs. The approach offers a sufficient degree of abstraction for adapting the representation to the different characteristics of each type of RIA. Moreover, not only do semantic RIAs provide a simple set of semantic annotations, but they offer a complete representation of a RIA from different viewpoints: domain knowledge, navigational contexts and visualization elements, which can boost the interoperability between systems, allowing that a certain application could be visually reinterpreted from its semantic representation.

In order to efficiently address the design and development of semantic RIAs, we present *Semantic Models for RIA* (S^m4RIA), a new model driven methodology designed as an extension of the OOH4RIA methodology [15]. Due to its characteristics and after a process of adaptation, S^m4RIA can be supported by OIDE (OOH4RIA Integrated Development Environment [17]).

Finally, the structure of the present paper is divided as follows. Section 2 defines the main characteristics and requirements of semantic RIAs. Section 3 presents the S^m4RIA methodology and describes each of its processes and models. The next two sections address the problem of designing the server side and the client side of semantic RIAs. In Section 6, we present an analysis of the bibliography focusing on model-driven methodologies. Finally, Section 7 draws the main conclusions of the paper and the future lines of research.

2 Characteristics of Semantic Rich Internet Applications

During the last years, several approaches [15, 23, 11] have dealt with defining a set of characteristics desired for any RIA and how they should be modelled. However, the inclusion of Semantic Web technologies in RIA is a field in which current efforts are in early stages. Considering the requirements for Semantic Web engineering defined in [5], we would like to point out what characteristics transform a RIA into a Semantic RIA from our approach. We specifically propose to extend current RIAs from two general requirements:

A) High level of exportability, understandability and reusability of its contents. The capacity of providing contents in a meaningful, non-ambiguous and structured manner to external software clients, such as autonomous agents or even other Semantic RIAs. This high-level requirement can be divided into two characteristics:

A.1) Ontologies as knowledge representation formalism. All the data stored and managed by a RIA has to be represented by means of ontologies.

A.2) Provide semantic annotations of the content. Ontologies just provide a method to represent the underlying knowledge used by a RIA. However, it is also important to annotate certain chunks of information to effectively represent what information is being shown. In order to avoid ambiguities, it is necessary to define the concept of semantic annotation that will be used and the annotation model [2] chosen for semantic RIAs:

A.2.1) Definition of **semantic annotation**: reference (in this case, HTTP URIs) to an ontology or its instances, attached to a chunk of information or a complete RIA.

A.2.2) The **annotation model**, i.e. what type of annotations and how to include them within a RIA, can be briefly summarised in three aspects:

- Three types of ontology-based annotations: domain, navigational and visualization annotations, each of which associated to a specific ontology or set of instances, thus providing a complete view of the RIA.
- Embedded annotations (principally for browser-oriented RIAs). The possibility of including text annotations (through the current standard RDFa) within the text-based contents of the RIA (mainly HTML).
- Service for knowledge sharing (for any RIA). Opening an point of access to the contents, based on the principles of the Linking Open Data Project [3] for data sharing and reuse, can simplify and boost the retrieval of knowledge (ontologies and instances) from the RIA.

B**) High level of reusability of external knowledge**. Following the philosophy behind the Semantic Web and the Linking Open Data project, semantic RIAs might be enriched with other knowledge sources. This requirement can be split into two subrequirements:

B.1) Reuse existing ontologies. This process can lead into richer contents, a simpler process of knowledge sharing and a less time-consuming design phase.

B.2) Reuse existing knowledge bases. In order to reduce the complexity of resusing knowledge in an open domain we initially propose to enrich RIAs from two types of knowledge sources:

B.2.1) Reuse of available Linked Data sources. Use of the large datasets available on the Linked Data cloud (http://linkeddata.org).

B.2.2) Reuse of knowledge from semantic RIAs.

3 The S[m]4RIA Development Process

RIA development methodologies are relatively new and one yet unsupported aspect of these methodologies is the addition of semantic information to RIAs. Therefore, this paper presents a new proposal called *Semantic Models for RIA* (S[m]4RIA), which extends the OOH4RIA development process [15] with a set of modelling artefacts and activities in order to create semantic RIAs. OOH4RIA is a proposal whose main target is to cover all the phases of the RIA lifecycle development. It specifies an almost complete RIA through two server-side models (i.e. Domain and Navigation) and two RIA presentation models (i.e. Presentation and Orchestration). In this work, we propose to extend these RIA functional models with a new viewpoint using two mechanisms: *(i)* the extension of the OOH4RIA MOF metamodel by means of a set of metaclasses, which relates the RIA client and server concepts to semantic sources; and *(ii)* a set of model-to-model transformations to automatically obtain new ontology models from the functional RIA models.

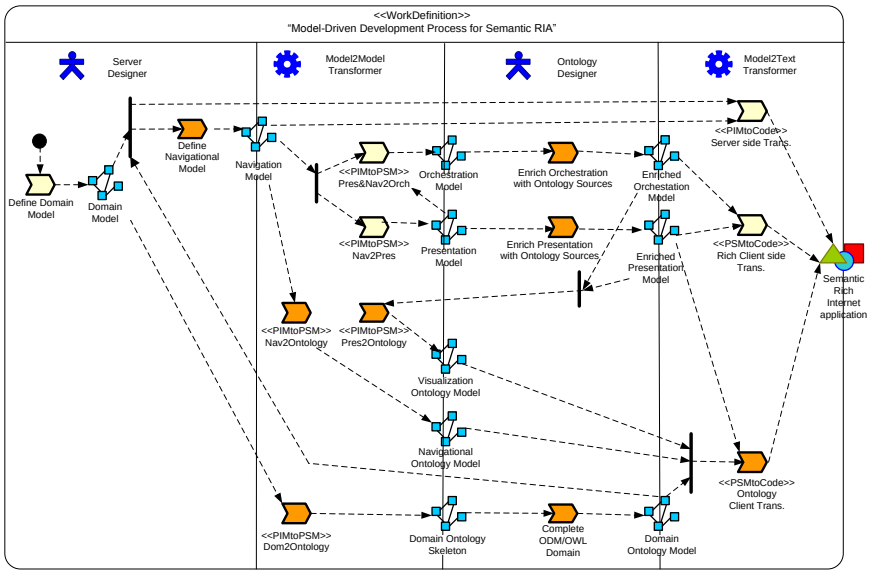

Fig 1. S[m]4RIA development process.

To represent this process (see Fig. 1), we used an OMG[1] standard called Software Process Engineering Metamodel (SPEM) [22]. Specifically, we have extended the SPEM profile introducing the stereotype *ProcessRole* to represent transformation engines (called *Model Transformers* in Fig. 1) and defining a set of stereotypes of the metaclass *Activity* to represent different MDA transformations, such as PIMToPIM, PIMToPSM, PIMToCode, PSMToCode, etc.

The S[m]4RIA process develops the original OOH4RIA process by extending existing activities and introducing a collection of new ones (coloured in grey). This

[1] Object Management Group. http://www.omg.org/

process starts when the server designer defines the *Domain model* for representing the domain entities and the relationships between them. The Domain metamodel introduces the necessary information to obtain a complete server side design with CRUD operations. At this point, a subprocess starts at the *Dom2Ontology* transformation, which is a variation of the UML2OWL [21] that converts the Domain model into a *Domain Ontology Skeleton*. The ontology skeleton is completed when the ontology designer defines semantic relations to external ontology sources in such a way that the concepts of the RIA can be aligned to concepts from other applications. This *Domain Ontology* is also an input of the *Define Navigation Model* activity, in which the server designer creates navigational links between domain concepts (internal or from external sources). This model allows designers to query ontology sources using OCL filters and retrieve the precise information from them.

The process continues by converting the *Navigation model* into other two different models. Specifically, the *PIM2PSM* transformation called *Nav2Pres* (see [15] in detail) obtains the *Presentation model,* and the *Nav2Ontology* PIM2PSM transformation generates the *Navigational Ontology model,* which defines how RIAs publish its own structured knowledge and connect its information to other sources of knowledge on the Web.

Another important contribution of this work is the application of UI semantic annotations within the Presentation and Orchestration models. The Presentation model provides a structural representation of the RIA widgets and layouts that is completed by the Orchestration model, which represents the interaction between these widgets and the rest of the RIA.

These UI models are the input of the *Enrich Presentation* and *Enrich Orchestration* activities, in which the ontology designer adds semantic annotations to the GUI widgets. These enriched Presentation and Orchestration models are the source of the PSMtoCode *Rich Client side* transformation, which obtains a RIA client side with semantic references. Finally, the set of ontology models generated during this process is the input of the PSMtoCode *Ontology RIA* transformation, which generates a RDF[2] representation from these models that will be made available to semantic agents on the Web.

In the following sections, we introduce several artefacts generated during the S^m4RIA design process by means of a case study: the RIA MediaPlayer[3], a typical multimedia player system that allows users to manage and play a collection of media files (e.g. music, films and document files). It aims at demonstrating the benefits of integrating, at design phase, RIA and ontology development.

4 The Business Logic and Data Persistence of Semantic Rich Internet Applications

As is well known, the evolution of RIA was mainly produced by the introduction of richer and more interactive UIs within Web applications. However, considering that a RIA is a special type of client-side application, it is also essential to describe, in a first

[2] Resource Description Framework. http://www.w3.org/standards/techs/rdf#w3c_all
[3] http://suma2.dlsi.ua.es/ooh4ria/Projects.html (Revised: June 21st, 2010)

stage, its business logic and a persistence method in order to provide a complete view of its functionalities. To carry out this task, S[m]4RIA reuses OOH4RIA's DSL (Domain-Specific Language) server models, the Domain and the Navigation models, which are actually based on OOH [12] server-side models, to obtain a CRUD Server (Create-Read-Update-Delete) implementation..

The Domain model represents the principal domain entities and it does not include technical nor implementation details, thus representing an ideal class model. In order to remove some ambiguities of the subsequent Object-Relational mapping, we have introduced several fundamental modifications on this model: *(i)* the Domain model defines a topology of different operations, such as create, delete, relate, unrelate, modify, readAll, etc., each of which is linked to a specific functionality in the final implementation; *(ii)* it deals with different collection types such as set, bag, list, etc.; and *(iii)* the model defines object identifiers for generating primary keys, and database aliases within classes, attributes and roles for naming tables, columns and foreign keys.

Fig. 2 depicts the Domain model of the MediaPlayer application, through which a user will manage and play a collection of MediaProducts such as, MusicTracks, Films or Documents. Moreover, the application will allow users to create personalized PlayLists that reference a set of preferred MediaProducts. Finally, MusicTracks will also be organized in Albums, each of which has references to its artists.

After specifying the Domain model, the process starts to define the server-side semantic information. To do so, we propose a model-to-model transformation called *Dom2Ontology*, which establishes a mapping from the elements of the Domain model (i.e. class entities and their relationships) to an ODM[4] model [21] called *Domain Ontology Skeleton*, which represents the ontology generated from the model. This transformation has certain similarities with the UMLtoODM transformation provided by the ODM specification. The model generated is a skeleton of the final domain ontology, represented by the *Domain Ontology Model*. Once generated, the ontology designer has to complete it in two stages. Firstly, external knowledge sources have to be found and linked to the base ontology using a package diagram. Specifically, from a public list of available Linking Open Data datasets[5], the designer can choose those ones that fit best with the requirements of the RIA and use them within the current workspace, thus importing the definition of their data, i.e. their ontology. In this case, it is supposed that the designer decided to reuse four existing ontologies: FOAF[6], which represents people and relations between people; DBLP[7] ontology, which describes scientific documents; MusicOntology[8], elements from the music domain: authors, groups, discs, styles, tracks, etc.; and LMDB[9], which represents elements related to the film sector.

---

[4] Ontology Definition Metamodel

[5] http://esw.w3.org/TaskForces/CommunityProjects/LinkingOpenData/DataSets (Revised: June 21st, 2010)

[6] Friend of a Friend. http://www.foaf-project.org/ (Revised: June 21st, 2010)

[7] http://dblp.rkbexplorer.com/ (Revised: June 21st, 2010)

[8] http://musicontology.com/ (Revised: June 21st, 2010)

[9] Linked Movie DataBase. http://www.linkedmdb.org/ (Revised: June 21st, 2010)

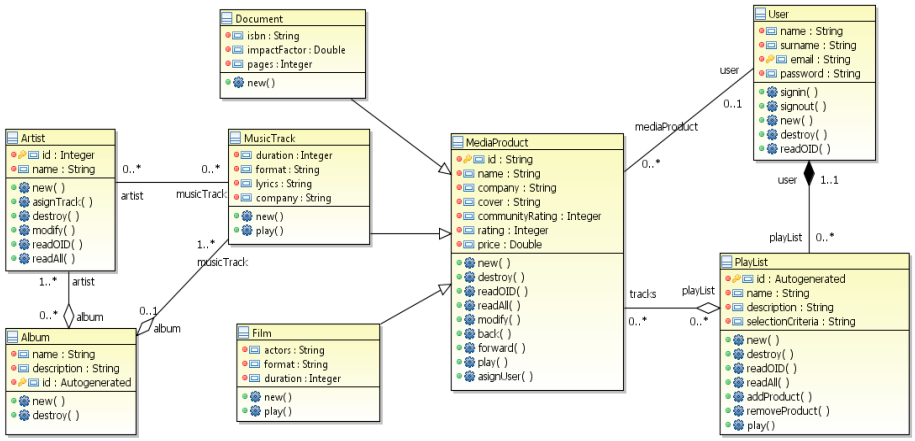

Fig 2. The Domain Model of the RIA MediaPlayer.

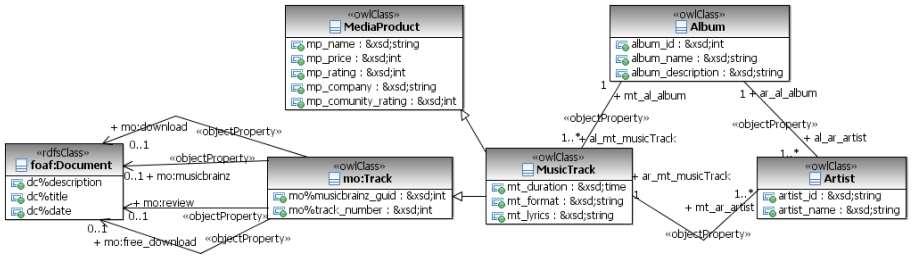

Fig. 3. Fragment of the Domain Ontology Model of the MediaPlayer RIA.

Subsequently, subsets of the concepts from each source can be aligned using different strategies (inheritance, equivalency, etc.; see Fig. 3). The designer has to link concepts among ontologies, choosing which properties are due to be used internally (since some of them might be duplicated in both concepts). Fig. 3 depicts a part of the final Domain Ontology model. In this case, the concept of 'MusicTrack' has been linked to the concept of 'Track' from MusicOntology (namespace *mo*) through a relation of inheritance (*owl:subClassOf*). Once the concepts have been linked, it is possible to specify what parts of this knowledge can be locally managed by the RIA. For instance, for each music track, it is relevant to keep information about its external ID (*mo:musicbrainz_guid*) and where it is stored on-line (*mo:free_download*).

After the definition of the ontology domain models, the server designer must design the Navigation model, which establishes the most relevant knowledge paths through the application space, thus filtering the domain elements that can be shown in the client side. This model is formalized by a MOF metamodel, which establishes a set of different types of navigational elements (as can be seen in [6]).

Fig. 4 shows the Navigation model for the current case study. The navigation starts with the Login NavigationalClass, through which a person has to identify as a User of the Mediaplayer in order to gain access. If the person introduces a valid email and password, the system navigates to the MailPlayer NavigationalClass, which is the

main page. It contains the name of the user and automatically shows him/her a list with his/her own playlists, to which he/she can create, add or edit new MediaProducts.

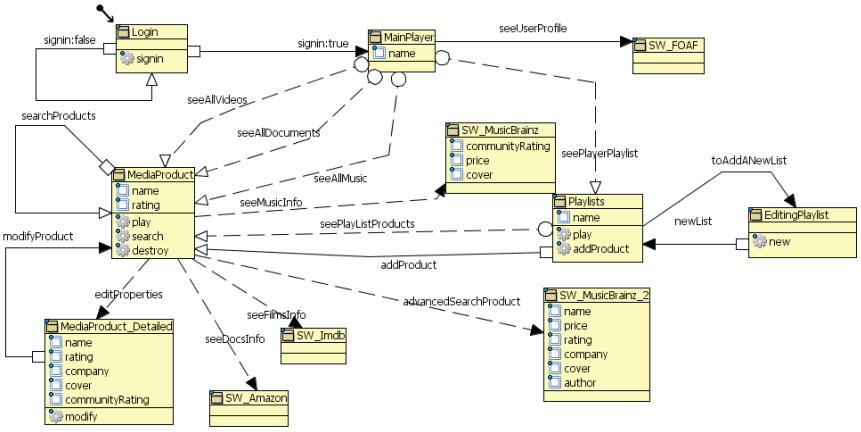

Fig. 4. The Navigation Model of the RIA MediaPlayer

This main page also allows the user to directly access information from different MediaProducts, which can be listed according to their type (i.e. videos, documents or music tracks). Then, he/she can select a specific mediaproduct to see more details (such as, its company, cover and community rating) or more information from the ontology sources related to each type of mediaproduct (DBLP for documents, LMDB for films and Musicbrainz for music). Moreover, the user can carry out an advance search by author or title within these ontology sources, which will retrieve URI references to new MediaProduct elements.

The next step of this process is to obtain the Navigational Ontology model, which defines navigational constraints to the knowledge sources, either internal or external, and aims at capturing and representing all the accesses to these sources and how knowledge is modelled and filtered within the different contexts of a RIA. It is worth highlighting that this is an internal model, only used by Model2Text processes and, thus, cannot be modified by designers. Moreover, the OWL Navigational model specifies which parts of the domain of a RIA are going to be generated for software agents through a Model2Text process, which would create an internal SPARQL endpoint. Specifically, this model organises knowledge within contexts and explicitly represents the use of SPARQL endpoints, as a knowledge management method, and SPARQL queries. To carry out this, we designed a new UML profile called *NavODM*, which is an extension of the current ODM UML profile that partially captures the semantics of the OOH4RIA Navigational Metamodel. NavODM defines the primitives needed to model knowledge contexts, their interrelation and the use of SPARQL endpoints. In addition, this profile provides a primitive for explicitly connecting the OWL Navigation model with the Domain Ontology Model, thus integrating knowledge from both views.

However, this model is not obtained from scratch. Our process establishes a model-to-model transformation called *Nav2Ont* (see process in Fig. 1), which automatically

generates the S[m]4RIA Navigational Ontology model from the Navigational model. Fig. 5 illustrates an example of QVT rule of the Nav2Ont transformation called *ServiceLink2SparqLLink*. This transformation rule checks whether there exists some instance of ServiceLink in the Navigation model (see Fig. 4) that connects an origin Navigational Class related to a Domain class to a target NavigationalClass from a external source. Only if these conditions are satisfied, will this rule create a SparqLLink, which would establish a relationship between a NavContext (with the name *nc1* of the origin NavigationalClass) and a new SparqEndPoint (with the NavigationalClass *nc2 as target*), which would be connected to the same OWLClass *c1*. Additionally, the "where" clause invokes the *SLArgumenttoQueryParam rules,* which generate a QueryParam set in a SparqLink for all the SLArguments of each ServiceLink.

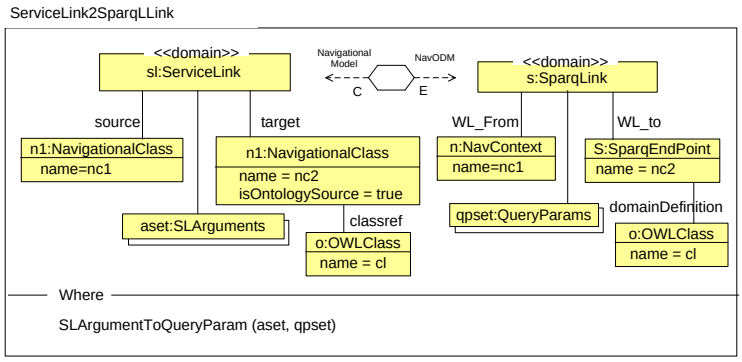

Fig. 5. The QVT ServiceLink2SparqlLink Rule of the Nav2Ontology Transformation.

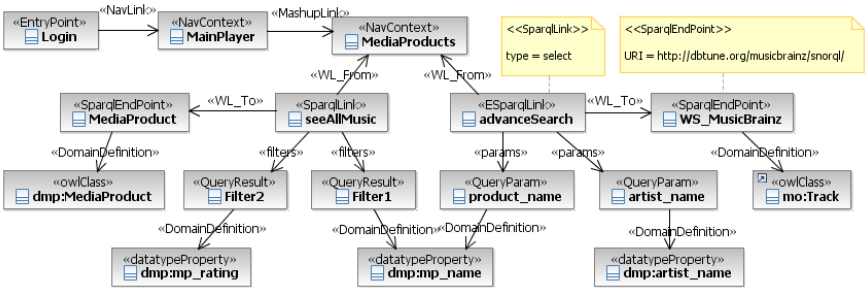

Fig 6. Fragment of the Navigation Ontology model of the RIA MediaPlayer.

The result of the Nav2Ontology transformation is the model depicted in Fig. 6, whose main elements are the navigational contexts (<<*NavContext*>> classes) and the SPARQL endpoints (<<*SparqlEndPoint*>> classes). Each endpoint within the model is connected to a concept definition on the Domain Ontology model (namespace *dmp*), which provides the type of elements that are going to be obtained from the source. SPARQL Links (<<*SparqlLink*>> and <<*ESparqlLink*>> elements) are the primitives for querying and filtering knowledge from the endpoints.

The next section illustrates how to design a RIA User Interface in which the S[m]4RIA process will introduce semantic extensions.

5 Designing a RIA User Interface with Semantic Extensions

In the last section, we have carried out the integration of semantic sources with a RIA server side. However, to our viewpoint, the most significant feature of S^{m}4RIA is how RIA User Interfaces (UI), similar to desktop UIs, can be actually enriched with semantic annotations, which will allow us to combine a good user experience with the semantic richness.

Unlike traditional Web applications, RIAs follow a "simple page application" [18] strategy, in which an UI is constituted of a single page with a set of stateful widgets. Moreover, this RIA UI has complex interaction dependencies between its widgets and the user, the RIA server side and other widgets. Following this assumption, OOH4RIA proposes two RIA-specific models to represent the UI: *(i)* the Presentation model, which defines a structural representation of different widgets based on a WYSWYG representation, thus allowing designers with a low level of training in programming to specify a RIA UI; and *(ii)* the Orchestration model, which represents the interactions between the UI widgets and the rest of the system using a collection of Event-Condition-Action rules formalized by a MOF metamodel.

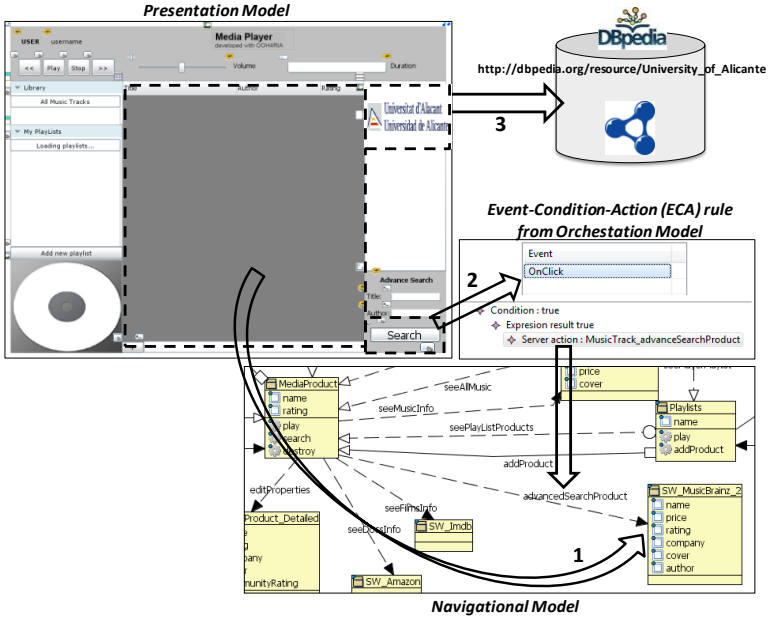

Fig. 7. Patterns of extension of the Presentation and Orchestration models.

The S^{m}4RIA extension of the client layer started with the definition of two new activities that enrich the corporative RIA UI and relate them to the semantic sources. The *Enrich Presentation and Orchestration Model* activities add semantic annotations to the GUI widgets by means of three patterns (see Fig. 7): *(i)* establishing a relationship between the RIA UI and the semantic navigational classes to gather semantic information from the server side (see in detail in Section 3); *(ii)* connecting UI actions from the Orchestration model to navigational links, which define OCL

filters for retrieving semantic information (from external or internal sources) on demand; and *(iii)* defining direct links from UI widgets to ontology elements, thus allowing the retrieval of information about this UI elements.

Once the semantic annotations have been introduced within the Presentation and Orchestration models, a QVT model-to-model transformation, called *Pres2Ontology, is applied to* the Enriched Presentation and Orchestration models for automatically generating a new semantic representation of the RIA UI called *Visualization Ontology Model*. This model gathers information from the abstract Presentation and Orchestration concepts and, consequently, it provides an abstract representation of the widgets presented to users, independent from the implementation of the UI. This model merges the structure and behaviour of UIs and represents the relations between UI widgets and the elements previously defined within the Navigational Ontology model (namespace *nav*). Moreover, it indicates which elements, such as images or labels, have to be explicitly annotated with semantic annotations. To model it, *VisuODM* was designed, which is a new ODM-based UML profile based on the OOH4RIA Abstract Presentation metamodel.

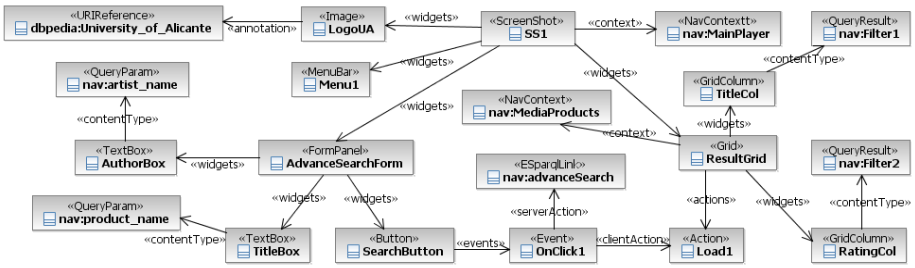

Fig. 8. Fragment of the Visualization Ontology Model for the RIA MediaPlayer.

Fig. 8 depicts a diagram showing part of the Visualization Ontology Model, specifically those widgets and events involved in the advance search of elements using external sources of knowledge. The main element is denoted with the stereotype <<*ScreenShot*>>, which represents a screenshot of the RIA. Each widget is denoted with subtypes of the stereotype <<*Widget*>>, such as <<*Grid*>> or <<*TextBox*>>. Each of these stereotypes has a set of tagged values for representing the aesthetic characteristics of each widget, such as relative position or background colour. Widgets have a default set of available events (<<*Event*>> elements) and methods (<<*Method*>> elements) that might be raised and actions that might carry out. The Visualization Ontology model is tightly linked to the Navigational Ontology model through different patterns: *(i)* container widgets can be linked to any <<*NavContext*>> element only once; *(ii)* an event has to be linked with a server action, represented in the Navigational model as a <<*SparqlLink*>> class; and *(iii)* the characteristics of the parameters and the results of a certain server action are explicitly represented through the <<*ContentType*>> relations.

Another feature that should be highlighted is the capacity of representing a semantic annotation over an element of the interface. In this case, Fig. 8 captures that the image *Image1* is represented by the resource *University_of_Alicante* of DBpedia, which contains all the information related to the actual content of the image.

6 Related Work

Although, to our knowledge, the combination of already existing model-driven techniques for developing Semantic RIAs has not been deeply addressed in the bibliography, the aims of this section are to analyse the different solutions adopted by the best-known model-driven design methodologies with regard to the development of RIAs and Semantic Web applications, and compare them with the S^m4RIA approach. Specifically, the present section will explore the following three methodologies: WebML [8], RUX-Method [23] and SHDM [13].

The first method to be analysed is WebML, which has been extended with primitives for modelling either rich interface or Semantic Web applications (among others, [7]). [5] defined the characteristics of a semantic Web portal and proposed an extension of WebML in order to develop this new type of application. Concurrently, [4] established the requirements for including the use of semantic Web services in web applications. However, these proposals are oriented to the traditional Web interfaces and do not consider the existence of RIAs. Although [10] did propose a specific extension of WebML for designing RIAs, it still remains unconnected to the aforementioned approaches.

RUX-Method [23] is a model-driven methodology for developing RIA interfaces. As an extension of the method, [14] presented the requirements and the changes needed to include semantic annotations within AJAX user interfaces by applying the W3C WAI-ARIA standard for RIA accessibility. The authors proposed a solution actually tied to another tool, EditSAW, with its own underlying methodology for creating accessible web sites.

The last one, SHDM [13], which is an ontology-driven design methodology for building Web applications on the Semantic Web, was extended with the necessary primitives for designing RIAs some years after its invention (as described in [26]). This extension mainly increased the number of elements of the presentation ontology and the functionalities of the Web generator, such as the inclusion of RDFa annotations within RIA interfaces. While ontologies are only part of design and generation processes in SHDM, S^m4RIA makes these ontologies (domain, navigation and visualization) available to the end users, so that they can manage a complete machine-understandable three-layer semantic representation of a certain RIA.

We would like to mention, as a general observation, that the aforementioned methods fundament the inclusion of semantic annotations in RIAs on the use of HTML UIs, i.e. AJAX interfaces. This fact has a strong influence in the design of these methods and evidently limits the type of technologies that can be used to generate semantic RIAs. However, S^m4RIA provides a more general solution, based on the philosophy of the Linking Open Data project and, consequently, there is no strong dependence between the method and the technology of those RIAs generated. S^m4RIA can be applied to model plugin-oriented semantic RIAs, thus solving important limitations in accessibility and Search Engine Optimization techniques.

Finally, it is worth pointing out that there are other model-driven methodologies such as, WSDM [9] and Hera [25], which are capable of reusing existing knowledge on the Web and generating semantic annotations within Web applications. Nevertheless, they have been excluded from the analysis in this section since, to our knowledge, they cannot model and develop rich user interfaces.

7 Conclusions and Future Work

This paper introduces the concept of semantic RIA and S^{m}4RIA, an extension of the OOH4RIA methodology for designing and developing this new type of RIA. This extension defines the models and processes required for represent how the knowledge from the Web can be imported into a RIA and which knowledge a RIA can share using different techniques. ODM-based diagrams make the generation of the semantic representation easier and, at the same time, contain structured knowledge about the design of the whole application, which could be reused in other designs.

Specifically, the benefits of the S^{m}4RIA extension are the following: (i) it reduces the complexity of developing of semantic RIAs by applying a model-driven development process, which brings a reduction in the cost of developing and maintaining semantic RIAs; *(ii)* it allows obtaining two synchronized, complementary views (one RIA-oriented and another semantics-oriented) of the application; *(iii)* it enables non-expert users to use the knowledge bases available on the Web in their applications and to create new ones; and *(iv)* as is a model-driven approach, it makes possible to define the whole application at design time.

Future work aims at *(i)* implementing PSM2Code processes presented in Section 3 in the OIDE CASE tool; *(ii)* aligning semantic RIAs and S^{m}4RIA to current Web accessibility standards, such as WAI-ARIA; *(iii)* studying optimization techniques for retrieving knowledge from external sources; and *(iv)* analysing the use of semantic RIAs within the Social Web environment.

Acknowledgements. The present paper has been supported by the Spanish Ministry of Education under the FPU program (AP2007-03076) and the contract TIN2007-67078 (ESPIA).

References

1. Berners-Lee, T., Hendler, J., Lassila, O.: The Semantic Web. Scientific American 284(5), 34–43 (2001), http://www.sciam.com/article.cfm?id=the-semantic-web
2. Bettencourt, N., Maio, P., Silva, N., Rocha, J: A systematization and clarification of semantic web annotation terminology. In proceedings of the International Conference on Knowledge Engineering and Decision Support, ICKEDS'06, (2006), 27-34
3. Bizer, C., Heath, T., Berners-Lee, T. Linked Data - The Story So Far. Int. Journal Semantic Web Information Systems, 5 (3), (2009) 1-22.
4. Brambilla, M., Ceri, S., Facca, F.M., Celino, I., Cerizza, D., Valle, E.D.: Model-driven design and development of semantic web service applications. ACM Trans. Internet Technol. 8(1) (2007) 3
5. Brambilla, M., Facca, F.M.: Building semantic web portals with WebML. In: ICWE'07: Proceedings of the 7th international conference on Web engineering, Berlin, Heidelberg, Springer-Verlag (2007) 312–327
6. Cachero, C., Meliá, S., Genero, M., Poels, G., Calero, C.: Towards improving the navigability of Web applications: a model-driven approach. *European Journal of Information Systems*, 16, ISSN 0960-085X/07, (2007) 420-447

7. Ceri, S., Brambilla, M., Fraternali, P.: The history of WebML. In Conceptual Modeling: Foundations and Applications: Essays in Honor of John Mylopoulos, Springer-Verlag, Berlin, Heidelberg, (2009) 273–292
8. Ceri, S., Fraternali, P., Bongio, A.: Web Modeling Language (WebML): a modeling language for designing Web sites. Computer Networks (Amsterdam, Netherlands: 1999) 33(1–6) (2000) 137–157
9. De Troyer, O., Casteleyn, S., Plessers, P.: WSDM: Web Semantics Design Method. In Web Engineering: Modelling and Implementing Web Applications. Human-Computer Interaction Series Springer London, (2007) 303-351
10. Fialho, A.T.S., Schwabe, D.: Enriching hypermedia application interfaces. In Proceedings of the ICWE 2007, Berlin, Heidelberg, Springer-Verlag (2007) 188–193
11. Fraternali, P., Comai, S., Bozzon, A., Carughi, G.T.: Engineering rich internet applications with a model-driven approach. ACM Trans. Web 4(2) (2010) 1–47
12. Gómez, J., Cachero, C., Pastor, O.: Conceptual modeling of device-independent web applications. IEEE MultiMedia 8(2), 26–39 (2001)
13. Lima, F., Schwabe, D.: Modeling applications for the semantic web. In Proceedings of ICWE 2003, Berlin, Heidelberg, Springer-Verlag (2003) 417–426
14. Linaje, M., Lozano-Tello, A., Preciado, J.C., Rodríguez, R., Sanchez-Figueroa, F.: Obtaining accessible RIA UIs by combining RUX-Method and SAW. In Proceedings of the International Workshop on Automated Specification and Verification of Web Systems, Linz, Austria, (2009), 85-97
15. Meliá, S., Gómez, J., Pérez, S., Díaz, O.: A model-driven development for GWT-based rich Internet applications with ooh4ria. In Proceedings of the ICWE 2008, Washington, DC, USA, IEEE Computer Society (2008) 13–23
16. Meliá, S., Gómez, J., Pérez, S., Díaz, O.: Architectural and technological variability in Rich Internet Applications. IEEE Internet Computing 14(3) (2010) 24–32
17. Meliá, S., Martínez, J., Mira, S., Osuna, J.A., and Gómez, J. An Eclipse Plug-in for Model-Driven Development of Rich Internet Applications. In Proceedings of the ICWE 2010, Vienna, Austria, IEEE Computer Society (2010) XX–XX
18. Mesbah, A., van Deursen, A.: Migrating multi-page web applications to single-page AJAX interfaces. In Proceedings of the 11th European Conference on Software Maintenance and Reengineering. IEEE Computer Society, Washington, DC, USA (2007) 181–190
19. Mikroyannidis, A. Toward a Social Semantic Web. *Computer* 40, 11 (Nov. 2007), 113-115. (2007) DOI= http://dx.doi.org/10.1109/MC.2007.405
20. O'Reilly, T.: What is web 2.0? Design patterns and business models for the next generation of software. Tech. rep. (2005), http://oreilly.com/web2/archive/what-is-web-20.html
21. Object Management Group: Ontology Definition Metamodel Version 1.0. Object Modeling Group (May 2009), http://www.omg.org/spec/ODM/1.0/PDF (Revised: June 21st, 2010)
22. Object Management Group: Software Process Engineering Meta-Model, version 2.0. Object Modeling Group (April 2008), http://www.omg.org/cgi-bin/doc?formal/08-04-01.pdf
23. Preciado, J.C., Linaje, M., Comai, S., Sanchez-Figueroa, F.: Designing rich internet applications with web engineering methodologies. In Proceedings of the WWV 2007, Washington, DC, USA, IEEE Computer Society (2007) 23–30
24. Studer, R., Benjamins, R., Fensel, D.: Knowledge Engineering: Principles and Methods. *Data Knowl. Eng.* 25(1-2) (1998) 161-197
25. van der Sluijs, K., Houben, G., Broekstra, J., Casteleyn, S.: Hera-S: web design using sesame. In Proceedings of the ICWE 2006. Palo Alto, California, USA. ACM, New York, 263, (2006) 337-344. DOI= http://doi.acm.org/10.1145/1145581.1145646
26. W3C Model-based User Interfaces Incubator Group. SHDM - Semantic Hypermedia Design Method (2009), http://www.w3.org/2005/Incubator/model-based-ui/wiki/SHDM_-_Semantic_Hypermedia_Design_Method (Revised June 21st, 2010)