Developing Rich Internet Applications as Social Sites on the Semantic Web: a Model-Driven Approach

Jesús M. Hermida, Santiago Meliá, Andrés Montoyo, Jaime Gómez
Department of Software and Computing Systems, University of Alicante, Spain

ABSTRACT
Current Web 2.0 applications, either social sites or Rich Internet Applications, share several problems of interoperability when interconnecting different systems. It is therefore complicated to reuse (or export) the information between sources. In this context, where the value belongs to the data, not the application, the use of Semantic Web technologies opens a way of resolution with mature and standard technologies, thus leading to the Web 3.0.
This paper presents the application of Sm4RIA (Semantic Models for RIA), a model-driven design methodology that facilitates the development of semantic RIAs (SRIA), to the design of social network sites. The SRIA approach introduced herein combines the main advantages present in each of the current trends on the Web. In addition to these benefits, the application of a model-driven methodology can speed up the development process and simplify the reuse of external sources of knowledge.

Keywords: Sm4RIA; semantic RIA; Social Semantic Web; Web 3.0; rich internet applications; model-driven engineering; Web engineering

INTRODUCTION
A new wave of Web applications is currently under research. Web 2.0 applications (social sites and rich internet applications, RIA) are being enhanced with existing Semantic Web technologies, leading to the so-called Web 3.0, or Social Semantic Web (Mikroyannidis, 2007), combining technologies from the current Web trends (as stated by Murugesan, 2008). After years of development, all these technologies, techniques and applications are sufficiently mature to be tackled from an engineering viewpoint.
During the last decade, the Web 2.0 (O’Reilly, 2005) considerably extended the use of the Web among a large variety of users. Within this broad concept, at least two subtrends can be identified, one associated to a change in users’ behaviour and another technological: the Social Web and the Rich Internet Applications (RIA). Among the most representative types of application within the Social Web environment, social network sites (SNS; Boyd and Ellison, 2008), such as Facebook (http://facebook.com), received the largest amount of attention because of the number of users of different ages, gender and nationalities gained worldwide. At the same time, the Web 2.0 brought a significant change in the manner of interaction and communication between users and applications, and among applications as well. RIAs offer user interfaces with
a higher level of interactivity, similar to desktop interfaces, embed multimedia contents and minimise the communication between client and server components.

Nonetheless, both trends share some shortcomings that mainly limit the portability of data from their applications. On the one hand, social sites are normally born as proprietary sites where their API-based access methods do not share all their available information and the semantics of the data elements might vary between applications. Currently, it is usual that users have different profiles in several networks since they cannot reuse their own personal data (Breslin & Decker, 2007; Breslin, Passant & Decker, 2009). On the other hand, RIAs show the information in a user-friendly manner but, due to their intrinsic structure and technological issues, they also suffer these types of limitations. Plugin-oriented RIAs (e.g. those implemented using Flex or Silverlight) are similar to black boxes since they are stored as binary objects constraining the access to data only to the human users in a visual manner. However, as RIAs are event-driven applications, even exploring the content of browser-oriented RIAs, i.e. those based on HTML and Javascript technologies (e.g. AJAX, Asynchronous JavaScript And XML), in an automatic manner is not a straightforward process. HTML5 is an ongoing effort from the World Wide Web Consortium (W3C) to provide a modern platform to build Web applications, including RIAs, which could solve the issues introduced by the RIA technologies. However, this standard has not yet reached the level of maturity required to be widely adopted by the community.

In the context of the present Internet, where the value has been moved from the Web applications to the data managed by them, the use of open technological solutions is a need. In this way, the Semantic Web (SW; Berners-Lee, Hendler & Lassila, 2001) was aimed to resolve problems derived from the semantic incompatibility of systems by means of standard techniques and technologies (from knowledge representation and sharing to trust and security). These are the keys to solving the aforementioned issues and developing the Web to a further stage, i.e. systems containing our collective intelligence (Gruber, 2008).

The Social Semantic Web, as a combination of the Social and Semantic Web, is becoming a reality with new types of applications such as semantic blogs and wikis (described by Kinsella, Passant, Breslin, Decker & Jaokar, 2009), and different approaches that facilitate the reuse of knowledge between social platforms, such as FOAF (Friend of a Friend, http://foaf-project.org) or SIOC (Semantically-Interlinked Online Communities, http://sioc-project.org/).

In this context, this paper presents the application of $S^m4RIA$ (Semantic Models for RIA, Hermida, Meliá, Montoyo & Gómez, 2011), a model-driven design methodology that allows the creation of semantic RIAs, to the design of Web 3.0 RIA applications. In this case, the development of a social network site with a semantic RIA is employed as running example in order to explain the different stages of the methodology and the benefits of the use of SW technologies. This SNS will manage user profiles by means of FOAF and will integrate domain knowledge from the MusicBrainz knowledge base. The approach introduced herein will thus combine the main advantages present in each of the current trends on the Web. In addition to these benefits, the application of a model-driven methodology will speed up the development of these Web applications and simplify the reuse of external sources of knowledge. Developers will be able to specify the main features of their applications as a set of models and, subsequently, employ the OIDE CASE tool (OOH4RIA Integrated Development Environment; Meliá, Martínez, Mira, Osuna & Gómez, 2010) to obtain the final application from these models in a semi-automatic manner.

This paper is structured as follows: the next section describes the main features of semantic RIAs, the manner in which they can be used as a social site and the benefits of this approach.
Subsequently, the third and the fourth sections introduce the use case employed in this paper and an overview of the S4RIA methodology, respectively. Afterwards, the following sections describe the case study chosen and each of the activities necessary to develop the social site. The eighth section presents similar approaches and discusses their main benefits and shortcomings. Finally, the last section draws the main conclusions of the paper and explains the lines of future work.

**SEMANTIC RICH INTERNET APPLICATIONS AS SOCIAL SITES**

Despite the benefits of using Rich Internet Applications, these applications share two of the main shortcomings detected in social sites: data exportability and reusability. Although there currently exist several available solutions, they are dependent on the technology chosen for implementing the application. As mentioned before, the approach presented in this paper is based on the concept of Semantic Rich Internet Application (SRIA; Hermida et al., 2011), which can be defined as a RIA that extensively uses SW techniques and technologies to provide a representation of its contents and reuse existing knowledge sources on the Web.

![Figure 1. Schema of a semantic RIA and the connections to other applications. (Linking Open Data cloud diagram, by Richard Cyganiak and Anja Jentzsch. http://lod-cloud.net/)](image)

Figure 1 presents a conceptual schema of a SRIA with its relations to other components of the Web. The SRIA is illustrated as a grey box with a client-server architecture, whose components share data through asynchronous communication processes within the Web infrastructure. However, the final aims of a SRIA are (i) to open and share the working data of the application to some types of user that cannot currently get access to it without threatening the security of the application; and (ii) to reuse the available knowledge sources on the Semantic Web to enrich the
information shown to the users. In particular, SRIAs contain four software modules not present in RIAs:

- **Knowledge base management (server-side).** This module manages the Knowledge Base (KB) of the application. Since the application is going to retrieve knowledge from the SW, it will need a knowledge base (based on RDF, Resource Description Framework) as a storage system, which can be included in the database or managed separately.

- **SPARQL interface (server-side).** This module offers a service to access a part of the knowledge stored in the SRIA KB using SPARQL (SPARQL Protocol And RDF Query Language). In this case, this approach is aligned to Linked Data principles (Bizer, Heath & Berners-Lee, 2009), mainly due to the simplicity of their processes. However, this interface can be changed depending on the requirements of the application.

- **(Semantic) Web service client (server-side).** This client is actually a combination of clients for different types of services: web services, semantic web services (SWS) and linked data endpoints, which enable the application to access on demand ontologies and knowledge bases on the Web, even those published by other SRIAs.

- **HTML Interface Generator (server-side/client-side, plugin-oriented RIA interfaces).** This module generates an HTML representation of the ontology instances stored in the knowledge base. This view is annotated using RDFa and points to the mentioned KB instances. This HTML representation could be easily crawled and indexed by Web searchers using a process similar to the one carried out for traditional Web applications. Unlike the other SRIA modules, in this module, the communication between the server and the client part of the SRIA follows a synchronous process.

- **Semantic annotation generator (client-side, browser-oriented RIA interfaces).** This software module embeds RDFa semantic annotations (World Wide Web Consortium, W3C, 2008) within the user interface (UI) of the SRIA, which establishes a link to the instances stored in the local KB or external knowledge sources. The annotation model is based on two types of semantic annotations: content and context annotations. While the first ones are related to the application domain, the second ones provide knowledge about the navigational contexts and the visualisation elements, with knowledge that can help any users to understand and interpret the given information. These annotations are actually based on three OWL ontologies: domain –dependent from the application data–, navigation and visualisation –independent from the application developed.

Semantic RIAs can be used a generic platform for the development of different types of applications, since they bring a technology-independent solution, using existing techniques, to solve the main shortcomings of RIAs. Among these possible applications, this paper is focused on the development of social sites, and more specifically, social network sites. The main benefits of using SRIAs as a social platform are the following:

**a) Benefits from the use of Semantic Web technologies:**

- **Improved interoperability among social sites.** The use of ontologies such as FOAF or SIOC, allows Web designers to describe all the personal data of the SNS (list of friends, contents from the wall, etc.) in a standard manner and facilitates the process of sharing among different social sites. While the FOAF ontology is widely used on the Web to describe people, documents and their relations, SIOC can complementarily describe all the types of possible contents of a social site.
a.2) Enrichment of the contents. SRIAs can reuse knowledge from other sites or existing knowledge sources in order to enrich the content presented to users. For instance, a SRIA can automatically access knowledge contained in DBpedia (http://www.dbpedia.org) to obtain supplementary information about a certain topic. Moreover, depending on the RIA technology, SRIAs can add semantic annotations to the content visualised by the users, which can be employed by different Web users, such as searchers or special clients for people with physical impairments.

b) Benefits from the use of RIA technologies

b.1) Improved UI usability. The use of a RIA as an interface of a SNS is not a novel approach. There currently exist some examples, such as facedeck on http://www.telerik.com/products/facedeck.aspx, that offer a higher degree of interactivity to users and a friendlier, desktop-like UI. The main difference is that this approach aims at building the complete SNS as a SRIA, not only the UI, thus improving the behaviour of the whole application.

AN OVERVIEW OF THE S^m4RIA METHODOLOGY

The S^m4RIA (/sem for RIA/) methodology extends the OOH4RIA development process (Meliá, Gómez, Pérez & Díaz, 2008) with a set of activities and modelling artefacts for the development of semantic RIAs. OOH4RIA is a proposal whose main target is to cover all the phases of the RIA lifecycle development by means of two server-side models (i.e. Domain and Navigation) and two RIA presentation models (i.e. Presentation and Orchestration), which specify all the aspects of a RIA. In S^m4RIA, these RIA functional models are extended using two mechanisms: (i) the extension of the OOH4RIA MOF meta-model by means of a set of new meta-classes, which relates the RIA client and server concepts to knowledge sources from the SW; and (ii) a set of model-to-model and model-to-text transformations to automatically obtain new ontology models and the final SRIA software modules.

Figure 2 illustrates this development process using the OMG’s standard Software Process Engineering Meta-model (SPEM, Object Management Group, 2008). In particular, the diagrams depicted were extended by introducing the stereotype ProcessRole to represent transformation engines (called Model Transformers) and defining a set of stereotypes of the meta-class TaskDefinition to represent different MDA transformations (Model-Driven Architecture, e.g. PIMToPIM).

Specifically, the S^m4RIA design process can be divided into three activities, which group processes with the same final goal: (i) designing the elements of the SRIA server; (ii) designing the elements of the SRIA client; and (iii) generating the final SRIA through a set of automatic model-to-text transformation processes.

The first activity starts with the specification of the Extended Domain model (EDM), which represents the domain entities and the relationships between them by means of a domain ontology. This model is divided into three views with specific goals: (a) establish the relations between the SRIA ontology and external ontology sources; (b) align the domain concepts with external concepts (or even reuse concepts from other ontologies); and (c) define the object-relational mappings and define CRUD operations (i.e. Create, Read, Update and Delete) and other custom operations (specified by the designer) over the SRIA data.

The resulting model is the input of the Define Navigation Model task, in which the server designer creates navigational links between domain concepts (internal or from external sources). The Extended Navigation model allows designers to invoke services on the SW (e.g. SWSs or
Linked Data Endpoints) using OCL (Object Contraint Language) filters and retrieve information (as ontology instances) from them. In addition, the designer can capture the manner RIAs publish their own structured knowledge and connect their information to other sources of knowledge on the Web.

The second activity continues by transforming the Extended Navigation model into the Presentation model and, subsequently, the Orchestration model, through two model-to-model transformations called Nav2Pres and Nav&Pres2Orch. The Presentation model provides a structural representation of the SRIA widgets and layouts, completed by the Orchestration model, which represents the interaction between these widgets and the rest of the RIA. Through the Enrich Presentation and Enrich Orchestration tasks Sm4RIA associates the UI elements included in the Presentation and Orchestration models to external ontology sources specified by navigational classes.

Finally, the last activity is aimed at generating the final SRIA from the set of models created in the first two activities through a set of PSM2Code transformations, which can obtain each of the software modules of the SRIA.

In the following sections, we develop a case study focusing on the description of those Sm4RIA activities and artefacts that address the problem of data portability on social sites.
CASE STUDY: SOCIAL NETWORK ON A SEMANTIC RIA

In order to explain the application of S^m4RIA to the development of RIA applications in the Social Semantic Web, this paper will hereafter use the development of a social network site (SNS) using a semantic RIA as use case. In this case, the application will be developed using Silverlight and it will be focused on the music domain (instead of having a general purpose). Moreover, it will embed a music player in case users need to play songs or complete albums, thus offering a platform for online music sales as well.

The present case study does not introduce new social features, which have been exploited by existing SNSs such as Apple’s Ping (http://www.apple.com/itunes/ping/) or lastFM (http://www.last.fm/). Instead, it is aimed at two issues: (i) the interconnection (interoperability) between social network sites using open SW techniques and resources, already studied in other works, e.g. Kinsella et al. (2008), and (ii) the analysis of the development process proposed. Security and trust issues are out of the scope of this paper in spite of their importance in this type of applications.

The final application will manage the basic features of current SNSs, i.e. (a) the management of a user profile, (b) the connections with other users and (c) a wall where users can share thoughts/impressions/comments with their connections. Moreover, users will be able to (i) share their music preferences, groups and songs, (ii) follow their favourite artists and (iii) read and publish news on all these topics.

Figure 3 depicts a screenshot of the final application, whose UI can be divided in four different zones: (1) the music player, located on the top of the application, together with the main menu; (2) the user profile, right area, with the main user information; (3) the wall, in the central area, with the main user’s and user friend’s stories and comments; and (4) the search form, on the left part of the UI, with a form to search friends, artists or songs within the SNS.

Figure 3. Screenshot of the main context of the Social Network Site.

The following sections will apply the S^m4RIA methodology to develop the case study emphasising those aspects related to the interchange of knowledge between applications.
DESIGN OF THE SERVER SIDE OF THE SOCIAL NETWORK SITE

As introduced in previous sections, the first activity of the Sm4RIA design process consists in designing the server side of a SRIA, in our case the social network site.

Building the domain ontology and specifying the data persistence methods

The process starts when the server designer builds the Extended Domain model of the application (see a subset in Figure 4, 5 and 6), which represents the domain ontology of the application, i.e. the concepts internally managed by the SNS. This model is specified by means of three complementary views: the Source, Ontology and Class views, which correspondingly specify the ontologies to be imported, the domain ontology itself and the operations over the application data and the object-relation mappings. The model is based on the OntoDom meta-model, which is an extension of the OOH4RIA domain meta-model and the OMG’s Ontology Definition Meta-model (ODM; Object Management Group, 2010). The OntoDom simplifies the definition of elements and includes a new element, the Domain Ontology Source (DOSource), to represent knowledge sources with different features –i.e. SWSs, SPARQL endpoints, etc.

The first view to be defined is the Source view, by which the designer indicates the ontologies that will be imported and the location of their instances on the Web. Figure 4 illustrates the Source view of the EDM for the example introduced in the previous section. The DOM DODefinition element represents the domain ontology of the application, while the DOMSource DOSource element indicates where their instances are stored in. Both of these must be included by default in any EDM. In this example, the DOM imports three ontologies (DODefinition FOAF, SIOC and MusicOntology): the FOAF ontology, which represents people, documents and their relationships; SIOC, which represents concepts related to social sites and the Social Web; and the MusicOntology ontology, which represents concepts about the music domain. The instances of FOAF and MusicOntology are stored in another social network site (called SNS_FaceRIA) and MusicBrainz (DOSource elements MusicBrainz and SNS_FaceRIA). This will enable the application to search friends in another social site and retrieve information about music from MusicBrainz.

Figure 4. Source view of the Extended Domain Model of the case study.
Once completed the first view, the designer specifies the second one, i.e. the Ontology view. In this step, the ontology designer actually builds the domain ontology (contained in the DOM DODefinition element) creating new ontology elements, reusing elements from the ontologies already imported or mapping local concepts to the concepts from external ontologies. Figure 5 depicts a fragment of the final Ontology view of the case study. Following the example, the elements of this view can be divided into two groups: a social one (classes sioc__UserAccount, foaf__Person, sioc__Role, Story, Comment, Tag, Invitation and Message), which would be similar for any SNS and contains those classes required for the social interaction of a community of users; and another dependent from the application domain, which is, in this case, associated to the music domain (classes mo__MusicArtist, mo__Track and UserAlbum) and determines the main aim of the application. In this case, the concepts used for the representation of people, their user accounts and their role (foaf__Person, sioc__UserAccount and sioc__Role) as well as the concepts for representing artists and songs were reused from the FOAF and SIOC ontology (the prefixes “sioc__” and “foaf__” correspond to the namespaces of these ontologies respectively, previously defined in the Source view).

This will also enable the generation of the information contained in the SNS as instances of FOAF and MusicOntology, facilitating the interchange of the knowledge contained in this SNS with other social sites.

In this SNS, there are two types of possible members: community members (or simply, users, class sioc__UserAccount) and corporative members (artists, i.e. users associated to an object of

Figure 5. Ontology view of the Extended Domain Model of the case study.
the class Artist). The first ones are the core of the SNS, i.e. those who interact, produce and consume the contents of the SNS. Corporative users also have the possibility of managing an official profile to interact with their fans or followers. Users can invite other users to become members of the community and friends. However, only the SNS administrator can assign the Artist profile to normal users. All the members own a wall where they publish stories (class Story) and comments about their stories or friends’ stories (class Comment). Moreover, users can send private messages (class Message) to their friends when required. As this SNS is focused on the music domain, each user can create his/her own personal music albums (class UserAlbum) from a list of songs (class mo__Track), which can be tagged using his/her own personal tags (class Tag).

After building the domain ontology of the application, the designer builds the Class view of the Extended Domain Model, which is depicted in Figure 6.
In this last view, which follows the notation of UML class diagrams, the designer defines the behaviour of each domain class by means of class operations (adding CRUD or custom operations), specifies compositions (part-of relations) and aggregations, and indicates which properties will be used as object identifiers (represented with a “key” icon, e.g. attributes “OID”) in the database, which could be a URI or another field. Elements imported from external sources are compulsory identified by URIs (attribute uri). This will simplify the creation of the final object-relational mappings, carried out in the last activity of S^m4RIA to generate the database/knowledge base management modules.

User Navigation and Access to External Services

The server designer continues by designing the Extended Navigational model, which represents how users navigate between the different data (or knowledge) contexts of the application. The Extended Navigational model defines the user navigation from the designer’s perspective in terms of (a) navigational classes, which are views of the data instances associated to a class of the EDM, and (b) navigational links, which define the manner users move through the data and invokes different services. Navigational classes (represented by boxes) offer certain attributes and services that could be accessed by RIA UIs or by other external applications. In fact, these correspond to attributes and operations of the EDM class. The initial navigation class, in which the user starts the navigation through the application, is denoted by a small arrow on the top left corner of the box.

There exist two types of navigational classes depending on the domain class they were created from: local (coloured in grey), created from local classes of the EDM DODefinition element; and external classes (coloured in white), imported from external DODefinition elements.

Navigational links, depicted as arrows, can be classified in transversal, which represents the transition between contexts, or service links, associated to an operation (those whose origin is drawn as a square). With navigational links, the designer can specify and publish the operations defined in the domain model as (public) Web services, using a specific architecture such as SOA (Service-Oriented Architecture). Black arrows represent that the data from the navigational classes is shown in a different context, while the white ones indicate that data will be visualised in the same context. When accessing an external navigational class, the designer can choose from which knowledge source the results should be obtained, already defined in the EDM Source view (DOSource elements), thus defining local or external transversal links (“E” and “L” links).

For each user role available in the application the designer can specify a new view of the Extended Navigational model. At least, there always exist two different views: one for human users and another for Semantic Web agents, which will represent the access to the ontology instances of the application by external web agents.

The first of these views is partially illustrated in Figure 7. In this case, in order to access the wall, (human) users (class AnonymousUser) need to be registered first. After the process of logging in, the user will be able to navigate to different areas from the User class. The first one will include the wall with the last stories and comments (navigational classes Story and Comment) and a list of his/her friends (navigational class Friend), which lead to their walls. A user can also search and invite new users to become his/her friends (navigational classes PossibleFriend, Invitation) or accept others’ invitations (navigational class UserInvitation). Regarding domain-specific features, a user can manage his/her personal albums (and play them, class UserAlbum) and tag the songs contained in them (classes MusicBrainz_Track, MusicArtist_MB_Info, Tag).

Moreover, users will be able to search and play songs on MusicBrainz (by means of the
searchSongs service link). The classes for the management of the artists’ profiles are included in the ArtistManagement package.

Those traversal links with an external class as destination are associated to a knowledge source (DOSource) either local or external (L or E links). For instance, the get_AllFriends link is associated to the DOMSource source (local KB of the SRIA) and get_AllMakers linked to the MusicBrainz KB, on an external Web location.

![Diagram](image)

**Figure 7. View (partial) of the Extended Navigational Model for human users.**

Once completed, the server designer builds the view for SW agents, which defines a navigational ontology representing a map of the application and constrains the domain ontology instances that will be generated for each navigational class by the final software modules (mainly the SPARQL endpoint and the HTML view, see the second section of this paper). Figure 8 presents a fragment of this view together with an OCL constraint to restrict the access to all the instances.

According to this model, the system will grant access to the user profiles (personal information and friends) and the stories published by each user. However, this access is limited by the OCL constraint included in the access traversal link. This constraint limits the generation of instances to those user accounts whose property sharing was set to “open”. This property can be managed by human users in the other view of the model depending on their preferences.
DESIGN OF THE CLIENT SIDE OF THE SOCIAL NETWORK SITE

The second activity of the Sm4RIA design process uses the server-side models already defined as the basis for the design of the client interface. The first task of this activity is the definition of the Presentation model, which is specific for each target technology/platform (in this case, Silverlight). With this model, the UI designer creates and organises the elements of the user interface (panels or widgets) and associates each UI element to an element of the Extended Navigational Model. Subsequently, with the Orchestration model, the UI designer describes the behaviour of the interface and links events occurred in the UI to server actions.

The Enrich Presentation and Orchestration Model tasks (i) establishes a relationship between the RIA UI and the external navigational classes to gather information from the server side; (ii) connects UI actions from the Orchestration model to navigational links, which define OCL filters for retrieving ontology instances (from external or local sources) on demand; and (iii) defines direct links from UI widgets to ontology elements, thus allowing the retrieval of information about this UI elements.

Figure 8. View (partial) of the Extended Navigational Model for Semantic Web agents.

Figure 9. Main screenshot of the Presentation Model of the case study and diagram legend.
In this case (see Figure 9), the interface is divided into four main areas: user’s profile, user’s wall, media player and search form. The users’ profile (with all his/her information) is located on the left part of the UI, while the users’ wall is in the main central area. Users can search friends, songs or artists using the “Advance search” form on the bottom right corner of the application and the results are visualised on a list located over the form. Last but not least, there is a media player to play the users’ favourite songs.

As a part of the Extended Orchestration model, Figure 10 shows the tree-based representation of an event-condition-action rule in OIDE. This rule is defined on the “Search” button and triggered when the OnClick event occurs. It was created to specify the behaviour of the “Advance Search” form. When the user chooses “Songs” from the SLComboBox element (combo box) and clicks on the target button, the application will use the content of the text box as a parameter for the searchSongs server action, which corresponds with the external service link with the same name from the Extended Navigational model and whose behaviour is defined by the EDM operation with the same name. The results will be shown in the SLLListBox1 list box, which is just above the Advance Search form in Figure 9.

**Figure 10. Event-Condition-Action rule from the Extended Orchestration model.**

**GENERATION OF THE SRIA SOFTWARE MODULES FROM THE S\textsuperscript{M}4RIA MODELS**

**The Visualisation Ontology model**
The last activity of S\textsuperscript{M}4RIA starts with the automatic generation of the Visualisation Ontology model (VOM) from the Extended Presentation and Orchestration models. This model is used for the generation of a visualisation ontology that can be used to annotate the application (depending on the final implementation technology). Further on, it can be aligned to existing accessibility approaches such as, W3C WAI-ARIA (Accessible Rich Internet Applications Suite, http://www.w3.org/WAI/intro/aria), in which case the ontology designer would establish those relations. The VOM is based on the OntoVisu metamodel, which extends the OMG’s ODM to include the characteristics managed by the Presentation and Orchestration models. The VOM combines the information of the structure, design and behaviour of the user interface in a technology-independent manner. Unlike the Presentation model, whose elements are adapted to a specific, target RIA platform/technology, the VOM specifies generic elements or elements that are present in any of the available technologies. These elements will be the basis of the visualisation ontology generated during the last activity of the process. This model can be automatically generated from the Extended Presentation and Orchestration models by means of a set of QVT (Query-View-Transformation) rules.
In this case, Figure 11 shows a fragment of the VOM, specifically focused on the search form and the event of clicking the “Search” button. The main elements of the diagram are (a) screenshots, represented by an eye (the open eye represents the principal screenshot); (b) complex visualisation elements (such as SearchForm), which contain (c) simple visualisation elements (e.g. SearchButton); (d) events, such as OnClick1, which is connected to (e) different actions (e.g. SetBinding), which can be carried out by the server part (S arrow) or the client part (C arrow) of the application.

**Generation of the final SRIA software modules**

All the models already created are (semi-)automatically transformed into the final software modules of the application. Depending on the design solutions taken in the previous activities some parts of these modules should be manually implemented before deploying the application, e.g. custom operations defined in the EDM.

---

Figure 11. Fragment of the Visualisation Ontology Model of the application.

Figure 12. Schema of the main model-to-text transformations of the Sm4RIA process.
Figure 12 offers an overview of this transformation process, highlighting those elements generated specifically for SRIAs. As can be noticed, not only are these new elements software modules, but also ontologies and mappings needed for the implementation of the annotation model proposed. The transformation process defines and applies a set of XPand transformation rules (http://www.eclipse.org/modeling/m2t/?project=xpand) for the generation of the whole modules (or the main code structures in those cases when generating all the code automatically is not possible). The modules depicted in Figure 12 correspond to those illustrated in Figure 1 as parts of a typical SRIA.

**Implementation of the S^m4RIA Processes in OIDE**

To conclude this section, it is worth pinpointing once again that the proposed case study was developed using an extension of OIDE (http://suma2.dlsi.ua.es/ooh4rial), which is an Eclipse-based tool that implements the OOH4RIA methodology generating RIAs using Silverlight or GWT (Google Web Toolkit). All its models (i.e. OOH4RIA’s models) were implemented using EMF (Eclipse Modelling Framework) and GMF (Graphical Modelling Framework), which manage the elements of a model in Eclipse and depict them on a canvas. In order to support the S^m4RIA method, it was necessary to extend the existing OOH4RIA meta-models and define new ones for the S^m4RIA models (e.g. OntoVisu). As regards the transformation and generation processes, the model-2-model and model-2-text transformations were implemented using QVT and Xpand rules, both also supported by the Eclipse framework.

**RELATED WORK & DISCUSSION**

Although, to our knowledge, the extension of the already existing model-driven methods for engineering Web 3.0 applications has not been deeply addressed in the bibliography, the goals of this section are: (i) to analyse the different solutions adopted by the best-known model-driven design methodologies with regard to the development of Web 2.0 (RIA and Social Web) and SW applications; and (ii) to compare them with the S^m4RIA approach.

The first method to be analysed is WebML (Web Modelling Language, Ceri, Fraternali & Bongio, 2000), which is the only one that has been extended with primitives for modelling either rich user interfaces, SW applications or community portals (among others, see Ceri, Brambilla & Fraternali, 2009). Brambilla and Facca (2007) defined the requirements and features of semantic Web portals and proposed an extension of WebML in order to develop this new type of application. At the same time, Brambilla et al. (2007) established the requirements for designing the access to SWSs from Web applications. However, these proposals are oriented to traditional Web interfaces and do not consider the existence of RIAs. Regarding Web 2.0 applications, on the one hand, Fraternali, Comai, Bozzon and Carughi (2010) proposed a specific extension of WebML for designing RIAs, aimed at browser-oriented RIAs. On the other hand, Fraternali, Tisi, Silva and Frattini (2010) defined a set of patterns used in community-based Web applications and exemplified how to design them using WebML. Nonetheless, the WebML approaches remain unconnected. On the contrary, S^m4RIA combines primitives to model RIAs and Semantic Web applications that can be applied, as demonstrated in this paper, to the development of new Web 3.0 social sites (or community sites, as stated by Fraternali et al., 2010). In addition, our method offers a better separation of concerns so that it clearly divides the elements that model the data persistence, business logic and user interface.
At present, there exist other methodologies that combine part of the requirements for the development of Web 3.0 applications. Among these, we would like to highlight RUX-Method, SHDM (Semantic Hypermedia Design Method), WSDM (Web Semantic Design Method) and Hera. Despite the fact that, to our knowledge, all these lack specific adaptations for designing social sites, once analysed their elements, it could be possible to apply some of the design patterns (Fraternali et al., 2010) using these methods. Having said that, this analysis is focused on those that merge primitives for designing RIAs and the use Semantic Web technologies in a certain manner, thus pointing out those approaches that might be used for developing our vision of Web 3.0 application.

RUX-Method (Preciado, Linaje, Comai & Sanchez-Figueroa, 2007) is a model-driven methodology for the development of RIA interfaces. As an extension of the method, Linaje, Lozano-Tello, Preciado, Rodríguez & Sanchez-Figueroa (2009) 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.

SHDM (Lima & Schwabe, 2009), created as 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 by the W3C Model-based User Interfaces Incubator Group, 2009). This extension mainly increased the number of elements of the presentation ontology and the functionalities of the Web generator, such as the generation of RDFa annotations.

The aforementioned methods address the inclusion of semantic annotations into RIAs only in the case of HTML-based UIs such as, AJAX interfaces. This fact has a strong influence in the design of these methods and evidently limits the technologies that can be used to generate semantic RIAs. However, Sm4RIA provides a more generic solution, based on the principles of the Linking Open Data project (Bizer, Heath, Berners-Lee, 2009) and, consequently, there is no strong dependence between the method and the technology of those RIAs generated. Furthermore, not only does Sm4RIA use ontology models in the design tasks, but it also makes these ontologies (domain, navigation and visualisation) available to the end users, so that they can manage a complete machine-understandable three-layer semantic representation of the SRIA.

Finally, WSDM (De Troyer, Casteleyn & Plessers, 2006) and Hera (van der Sluijs, Houben, Broekstra & Casteleyn, 2006) are capable of reusing existing knowledge on the Web and generating semantic annotations within Web applications. However, the analysis of both approaches is out of the scope of this paper, since they do not have primitive to design rich user interfaces.

In summary, as demonstrated in the paper, using the Sm4RIA approach it is possible to successfully engineer Web 3.0 applications, resolving the problems of data portability found in RIAs and SNSs. Nevertheless, Sm4RIA could be also applied to other scenarios within the Semantic Web, such as the development of semantic blogs, wikis or even Web applications with no social component. As mentioned before, SRIAs are actually a generic platform that opens the RIA data to the Web by means of the three ontologies proposed. In this way, Web users can obtain an accurate view of the application data, structure and visualisation.

While SRIAs give a solution reusing known techniques from the SW, Sm4RIA facilitates the implementation of this new kind of applications even to developers who are not experts in the field. Using both together, developers can create new Web 3.0 applications, combining features
from RIAs, SW applications and social sites, by means of a reduced collection of models and transformations, supported by an Eclipse-based tool that automates its processes.

At the present, the whole approach is oriented to the principles of the Linked Data approach, which is growing in popularity and success in the SW. This fact also restricts the type of SW sources that SRIAs can be attached to and the manner SRIA provides its internal data. Moreover, the proposal does not define Xpand transformation rules for any possible target RIA platform/technology, thus limiting the functionalities of OIDE. At present, the implementation is focused on the generation of Silverlight SRIAs, as an example of plugin-oriented SRIA, and the generation of browser-oriented SRIAs is under development.

CONCLUSION

This paper applies the concept of semantic RIA and the S^m4RIA methodology to the development of a Web 3.0 social site, which combines the benefits of the current Web trends: RIA, Social Web and Semantic Web. The use of SRIAs as social sites solves the existing interoperability issues between social sites and offers new features for software agents. In addition, it allows users to enrich their contents from structured knowledge sources or creating new sources, thus going a step further to the development of systems with collective intelligence.

S^m4RIA defines a collection of models and processes required for the specification of the processes of knowledge importation and sharing from the (Semantic) Web within a RIA. ODM-based diagrams facilitate the generation of the semantic representations 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 S^m4RIA are the following: (a) it reduces the complexity of developing of Web 3.0 applications by applying a model-driven development process, since it brings a reduction in the cost of development and maintenance; (b) it standardises the quality of the resulting applications; (c) it enables designers, without experience in semantics technologies, to use the knowledge bases available on the Web in their applications and to create new ones; and (d) being a model-driven approach, it facilitates the definition/specification of the whole application at design time, thus reducing the time needed for development.

Future work aims at (i) applying the design method with other types of application such as semantic blogs or wikis, which use other types of social patterns; (ii) aligning semantic RIAs and S^m4RIA to current Web accessibility standards, such as WAI-ARIA; (iii) studying optimisation techniques for retrieving knowledge from external sources; and (iv) adapting the S^m4RIA approach to other knowledge sources of the Semantic Web.

REFERENCES


ACKNOWLEDGEMENTS
This paper has been supported by the Spanish Ministry of Education under the FPU program (ref. AP2007-03076) and the SONRIA project (ref. TIN2010-15789). The authors would also like to thank the University of Alicante for the economic support given through the DIMENRIA research project (ref. GRE10-23).

Jesús M. Hermida is a PhD student at the University of Alicante, Spain, supported by the FPU program of the Spanish Ministry of Education. His current research interests include the application of Semantic Web technologies, techniques and resources (especially ontologies) to different areas, such as model-driven engineering, rich Internet applications, Natural Language Processing or knowledge representation in the health domain. Hermida received a 5-year degree in Computer Engineering from the University of Alicante, Spain, and the Diploma of Advanced Studies in the area of Software and Computing Systems at the same university (2009).

Santiago Meliá is an Assistant Professor at the Universidad de Alicante, Spain. His research topics of interest include Rich Internet Applications, Model-Driven Engineering, Generative
Andrés Montoyo is a Professor of Databases at the Computer Science School and of Ontology Design in Natural Language Processing and Semantic Web at the MSc in Computing Technologies at the University of Alicante. He received his Bachelor and Master’s degree in Computer Science from the Polytechnic University of Valencia and his PhD in Computer Science from the University of Alicante, Spain. His research interests include Information Extraction, Word Sense Disambiguation, Opinion Mining, Ontologies, and the Semantic Web. He is the author of more than 70 scientific publications in international journals and conferences on many topics, including sentiment analysis and opinion mining. He has been involved both in national and international research projects and co-organised several conferences and workshops on these topics.

Jaime Gómez is a Professor of Software Engineering at the Computer Science School at the University of Alicante. He received a B.S. in Computer Science from Technical University of Valencia in 1992 and Ph.D. in Computer Science from the University of Alicante in 1999. His current research interests include Web Engineering, Model-Driven Development, Rich Internet Applications Design and Semantic Web. He is the head of Web Engineering, Applications and Developments Research Group at University of Alicante. For more than ten years, his research group has developed methods and tools to facilitate software development in Industry. He is author of more than 50 papers in Journals and International Conferences.