Engineering semantic-based interactive multi-device web applications

Citation for published version (APA):
Bellekens, P. A. E., Sluijs, van der, K. A. M., Aroyo, L. M., & Houben, G. J. P. M. (2007). Engineering semantic-based interactive multi-device web applications. In L. Baresi, P. Fraternali, & G. J. Houben (Eds.), *Proceedings of the 7th International Conference on Web Engineering (ICWE 2007) 16-20 July 2007, Como, Italy* (pp. 328-342). (Lecture Notes in Computer Science; Vol. 4607). Springer. https://doi.org/10.1007/978-3-540-73597-7_27

DOI:
10.1007/978-3-540-73597-7_27

Document status and date:
Published: 01/01/2007

Document Version:
Publisher's PDF, also known as Version of Record (includes final page, issue and volume numbers)

Engineering Semantic-Based Interactive Multi-device Web Applications

Pieter Bellekens, Kees van der Sluijs, Lora Aroyo[⋆], and Geert-Jan Houben[⋆⋆]

Technische Universiteit Eindhoven
PO Box 513, NL 5600 MB, Eindhoven, The Netherlands
{p.a.e.bellekens,k.a.m.sluijs,l.m.aroyo,g.j.houben}@tue.nl

Abstract. To build high-quality personalized Web applications developers have to deal with a number of complex problems. We look at the growing class of personalized Web Applications that share three characteristic challenges. Firstly, the semantic problem of how to enable content reuse and integration. Another problem is how to move away from a sluggish static interface to a responsive dynamic one as seen in regular desktop applications. The third problem is adapting the system into a multi-device environment. For this class of personalized Web applications we look at an example application, a TV recommender called SenSee, in which we solve these problems in a metadata-driven way. We go into depth in the techniques we used to create a solution for these given problems, where we particularly look at utilizing the techniques of Web Services, Web 2.0 and the Semantic Web. Moreover, we show how these techniques can also be used to improve the core personalization functionality of the application. In this paper we present our experience with SenSee to demonstrate general engineering lessons for this type of applications.

1 Introduction

The ICT landscape is developing into a highly-interactive distributed environment in which people interact with multiple devices (e.g. portable devices such as mobile phones or home equipment such as TVs) and multiple applications (e.g. Web browsers and dedicated Web services). Globally the industry is being driven by the shift away from old models - moving from physical to digital. New methods emerge for getting content such as TV programs via the Web. For example, more and more people want to watch or manage TV content on their PCs, and thus make a bridge between a TV and a PC: each device doing what it can do best. As we can see, technologies in these fields are rapidly progressing, but the user is lost. The information overload is enormous and the content presented is hardly adapted to the prior knowledge, to the preferences and to the current situation of the user. Not only the users, but also the industry comes to

[⋆] Also affiliated with Vrije Universiteit, Amsterdam, The Netherlands.
[⋆⋆] Also affiliated with Vrije Universiteit Brussel, Brussels, Belgium.

the realization that integrated content services and personalized user experience are becoming more important.

Personalization in information retrieval [1] and information presentation [2] has therefore become a key issue. Successful personalization experiments have been done, in e-commerce [3] and news websites [4], where the most common example is the Amazon.com recommendations. Personalization is seen as a key ingredient of the so-called Web 2.0 applications [5] [6]. However, such personalization is still local. As a nice example, on the Web users are increasingly involved in multiple virtual environments (e.g. MySpace, Flickr, YouTube, Amazon, entertainment sites) in each of them with a different identity (e.g. login information, preferences). There is very limited integration between them, and if there does exist some integration, it is not always under the user's control. Moreover, there remains a great lack of transparency in the use of personal data between different applications [7].

The main objective in this research is to provide techniques to improve personalization in the world of multi-device access to interactive Web applications connected via semantic-based integration. This world can be characterized with three key terms: *Semantic-based, Interactive and Multi-device*. We use the acronym *SIM* to refer to this application setting. In SIM Web applications, people do use multiple devices to interact with multiple distributed Web applications. It results in complex interaction patterns requiring integrated views of distributed data collections, multiple modeling perspectives of content, user and environment data, as well as an increased need for personalized information presentation. To realize this kind of personalization, we need to address challenges concerning the integration of content services, the integration of user modeling and personalized presentation. In our approach we exploit recent developments in the field of *Web services*, *Web 2.0* and *Semantic Web* to meet these challenges.

In this paper we present the core elements of this approach. In Section 2 we explain the rationale behind our focus on the SIM setting. In Section 3 we introduce the SenSee application that we use to illustrate the SIM approach. SenSee is a personalized content recommender for multimedia consumption in an ambient home media-centre and is created within the context of the Passepartout project [8]. We then discuss its architecture and subsequently in Sections 4, 5 and 6 we show how we in SenSee implemented the Web Service, Web 2.0 and Semantic Web solutions for SIM applications. In Section 7 we show how the combination of the techniques can in addition be used to aid in the core functionality of SenSee. We finally summarize our experience for the general class of SIM applications in Section 8.

2 SIM Web Application Requirements

A main concern in modern Web applications is *personalization* and adapting the application in all its facets to the user's current situation. In this research we consider the development of personalized Web applications in the light of three characteristic trends and properties of modern Web applications.

(1) One aspect that has definitely changed in the recent years is that users no longer access the Web via PCs only. The Web has become much more a multi-device environment. Many types of devices can connect to the Web nowadays: think of PDAs, mobile phones, GPS-systems, TVs, game consoles, etc. As a consequence, Web applications can no longer be engineered with only one device, with its inherent properties, in mind. Designing for using an application on different devices is one step, e.g. allowing the user to choose between PC, PDA or mobile phone, but the consequent step and the one that we are now confronted with is to consider the user's interaction with an application via the current wealth of devices in an integrated fashion. So, instead of using one device we are faced with a multi-modal context in which for example the TV-content viewer uses the PC to manage personal preferences and the mobile phone to carry these preferences around to pass them then to a TV-set for viewing the content.

The Web applications that operate in such a *multi-device* setting bring challenges as well as benefits, particularly when we consider personalized access. The main challenge is the integration of the different parts of the application, i.e. to adapt the application aspects to different environments and different capabilities. Think of differences in video, sound and data processing capabilities: when distributing the application functionality over the device-related application parts these capabilities are leading. A major benefit for personalization is that we can learn from the user (assess the user's situation) in a more unobtrusive way, since the user will spend more time with all of his devices combined than with the PC alone. Furthermore, we can utilize online sensing devices like GPS functionality through a mobile phone, RFID tags, and biometric sensors, to improve the quality and quantity of the information we have on the user and therefore improve personalization in a context-sensitive way. If we, for example, know the user is driving in a car we could infer that we don't want to bother him with video messages as his attention is needed elsewhere.

(2) In the integrated context of our applications, the increasing need for *semantic-based* reuse of information and application functionality for personalization is obvious. In terms of content, one reason is that it is time-consuming to reproduce data that is already available. Furthermore, specialist data sources often are created by domain experts or crafted collaboratively, making it hard for non-domain experts to recreate something of the same quality. Also, one might want to use external data that can not be under the direct control of the application itself and therefore has to be fetched on demand. Think, for example, of reusing information from Wikipedia[1] or IMDB[2]. When integrating content in an application we are faced with the typical heterogeneous nature of the content, specially in the different terminology that is used from the different domains involved.

Besides reuse of content, also reuse of functionality becomes an even more relevant issue. Being able to share and combine application logic (components) in the integrated application enables to configure the application and its personalization

[1] http://wikipedia.org/
[2] http://imdb.com

on demand. The applications we consider do not just “import” content and functionality, but also “export” aspects to other applications. Most prominently, one can see this for the sharing and exchange of the user model information that is used in the personalization process. Applications running on different devices that are able to share the user’s preferences and situation description can improve their (individual) service to the user.

(3) A third trend in Web applications is the ever growing demand for *richer user interaction*. This interactivity can be used for personalization to offer the user more control, for example by allowing richer and more detailed feedback on metadata, both content-metadata and user model data. Modern Web applications should also no longer suffer from an unresponsive user interface between page loads, e.g. because of complex time-consuming operations. Instead, they should move towards the responsiveness of dynamic desktop applications. If a user triggers a query that gets distributed over different sources on the Web, the user should not need to wait until the slowest source responds and sends all its data. It would be much better to show as quickly as possible the first results, integrating results of slower sources as they arrive.

Enabling personalization in the next generation of these SIM Web applications where *semantics*, *interactivity* and *multi-device* are important properties, various new and more elaborate requirements surface. Tackling these challenges in a natural unobtrusive way is key for the success of such personalized applications. In the next section we introduce SenSee that helps to illustrate our target class of applications.

3 SenSee

SenSee is a personalized Web-based recommender for digital television content and an example of a SIM application that we use to illustrate our case. It is developed in the context of the Passepartout [8] project, which investigates the future of digital television. SenSee collects information about TV programs as well as related data from various sources on the Web. Furthermore, the user can access his personalized TV guide from various devices at any time, so not only on the TV itself. To provide high-quality personalization SenSee collects as much data on the user as possible. It provides the user with direct feedback options to provide his information manually, but also observes the user’s behavior and with help of sensor information like GPS determines the user context.

Many related TV recommender systems can be found in literature, like in [9], [10] and [11]. However, those systems mainly focus on the recommendation part. In AVATAR [10], for example, the authors make use of a combination of TV-Anytime [12] metadata fields and a custom-made genre hierarchy. Their recommendation algorithm is an effective hybrid combination of content-based and collaborative filtering, although there is no inclusion of any kind of user context. To get an overview of the various kinds of recommendation strategies and combinations that exist both for selection, structuring and presentation of content refer to [11]. In this paper, however, we focus on the properties of the

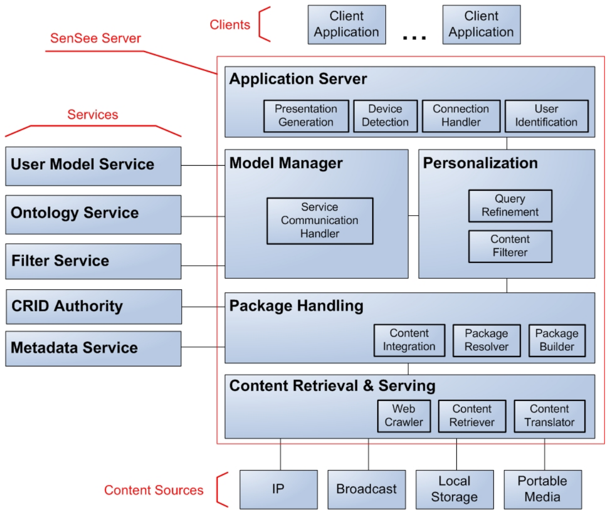

Fig. 1. SenSee Architecture

SIM framework which serves as the backbone of the SenSee recommender and how we can utilize the content metadata together with the user model metadata to solve the issues at hand.

The SenSee architecture, as depicted in Figure 1, is a layered one. At the bottom part of the figure, we find the various heterogeneous content sources. This can include regular TV broadcasts, but also movie clips from the Web, media stored on the local computer or physical media like DVD and Blu-ray discs. Metadata about the content originates from the bottom, propagates up while at each layer gaining semantics and context, and is finally used by an application at the top. In this process first information about the content is collected, and then converted to our internally used vocabulary and aggregated into *packages*. In order to do so we use two external services that are specified by the TV-Anytime specification, namely the *CRID Authority* for uniquely identifying multimedia objects and the *Metadata Service* that maintains metadata on these objects. Since typically there are numerous content packages available, we use personalization and recommendation based on the context and the user model with the aim to prevent that the user gets lost in the information abundance. Furthermore, we offer the user a user-guided search to assist the user in finding what he or she is looking for by refining a user query (keywords) with help of ontological domain knowledge. Both for personalization and user-guided search we use supporting services, each with their own responsibility and functionality. The *User Model Service* (UMS) maintains and retrieves data from the user model (including the current user context), the *Filter Service* (FS) provides filters needed to eliminate content unsuitable for the current user, and the *Ontology Service* (OS) maintains and manages all vocabularies and ontologies defining the

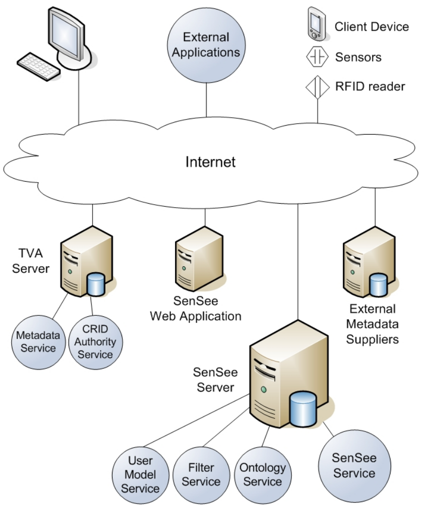

Fig. 2. SenSee Connection Architecture

semantics of concepts and properties. Finally, the metadata is shown to the user in a hierarchically structured way: the user can browse through the data and select the desired multimedia object. The object can then be retrieved via its unique CRID identifier [13] for consumption (viewing).

Figure 2 shows how the different parts of the SenSee framework are connected. The user can access SenSee via a client application. We currently implemented two client applications: The SenSee Web application[3] and a commercial application called iFanzy[14] that runs on a set-top box and is controlled by the television remote control. In this paper we will concentrate on the SenSee Web application which runs in a Web-browser[4].

The SenSee Web application connects to the SenSee server, which besides the SenSee Service (which contains the main application logic) also contains the three supporting services: UMS, FS and OS. The TV-Anytime services, CRID Authority Service and Metadata Service, as well as the content metadata that is not stored locally, are accessible via the Internet. Sensors and device information are accessed via external applications which are responsible to reading the sensor values and adding or updating them in the user model. Having SenSee running as a service enables these various applications to quickly make a connection and push their readings to the system. SenSee on its turn processes these readings and can immediately take them into account in upcoming actions.

For mass multimedia content consumption, people prefer the television as main target device. Since SenSee targets users from all kinds of social classes and age groups these sensors could make *interactivity* easier. Because of the personal nature of this application, people are required to log in, either by login/password

[3] http://wwwis.win.tue.nl:8888/SenSee/

[4] Note: Currently only tested in Firefox and Internet Explorer.

or through an RFID sensor. RFID recognition has been implemented to login people automatically when they for example enter the living room. Knowing who is watching, allows SenSee to optimize recommendations, the look-and-feel, and privileges that are appropriate for the current user or user group.

To personalize access to various online content sources and services, a certain sense of *semantics* is required for interpretation of the metadata. The domain model used by SenSee, to discern between for example 'actor' and 'director' or a 'western'- and 'action'-movie, is specified in the TV-Anytime specification [12]. TV-Anytime was specifically tailored to describe future multimedia content and provides constructs to describe all kinds of content elements as well as hierarchical containers suited to cluster and organize these numerous elements. To be able to exploit external content, metadata is required for interpretation. However, using various heterogeneous content sources implies that available metadata might be complying with different conceptual schemas which in turn might be not compatible with the TV-Anytime specification. In such a case SenSee uses transformations to migrate this content to the TV-Anytime platform. Currently, SenSee retrieves content from online Electronic Program Guides[5], BBC backstage[6] and various Web sites like IMDB and Wikipedia.

Since TV-Anytime focuses on describing multimedia content, which gives a rather restricted perspective, other concepts related to, for example, time and geographical locations are less profound. Therefore, in SenSee we fill those gaps by adding external knowledge coming from publicly available ontologies, schemas and thesauri by relating their concepts to existing ones in TV-Anytime, giving the domain more profundity. The most important knowledge structures, which are all maintained in the Ontology Service (OS), are among others:

- The TV-Anytime and MPEG7[7] schemas to describe content
- The W3C OWL-Time ontology [15] providing time conceptualizations
- A geographic thesaurus to describe locations like e.g. [16]
- WordNet[8,9] to provide synonyms, hyponyms and other lexical relations
- Topic hierarchies to identify e.g. program genres (TV-Anytime provides several topic classifications among which are content and action classification)

Maintaining the various RDF[17]/OWL[18] repositories we chose for the metadata, such as the user models and the various contexts in the UMS, and the various ontologies and thesauruses in the OS, is done through a Sesame [19] triple repository.

4 Web-Service Architecture

The SenSee application *shares* all of its main parts to encourage interoperability with others. Web services enable the interconnection between connected devices,

[5] http://xmltv.org/wiki/
[6] http://backstage.bbc.co.uk/
[7] http://www.chiariglione.org/mpeg/standards/mpeg-7/mpeg-7.htm
[8] http://wordnet.princeton.edu/
[9] http://www.semanticweb.org/library/

allow third parties to adapt their existing software to the provided interfaces and enable lightweight sensor devices to provide input.

Communication between all services and clients is established via XML-RPC[10]. XML-RPC is a remote procedure call protocol making use of XML to encode its messages and HTTP as transportation layer. XML-RPC was chosen because of its simplicity. It is designed to be as simple as possible, while allowing complex data structures to be transmitted, processed and returned. The implementation we use is version 3 of Apache XML-RPC[11]. The main advantage of this implementation is its ability to transmit any Java object, taken that it is serializable, and its ability to handle exceptions with ease. SOAP[12] was also considered, but discarded again because of the higher complexity in use, while the functionality was the same. To handle the various incoming XML-RPC requests and have them executed in such a way that different calls do not interfere with each other while maintaining good performance, we chose to use the Jetty Web server[13]. Jetty is a highly customizable standalone Web server completely build in Java, known for its scalability and efficiency.

Via this universal communication protocol we enable multiple devices, understanding the HTTP protocol, to be able to contact the SenSee server. Choosing lightweight XML-RPC messages to build and receive requests enables us to even use devices with limited processor power and bandwidth, like for example mobile phones. Furthermore, working with services allows us to run multiple instances of a specific service at the same time in different locations. The advantage of such a setting is that we enable load balancing, to avoid that one particular server chokes under too much requests. The disadvantage of such an approach is that we then also need a synchronization algorithm to keep all service instances up-to-date.

5 Interface Asynchronicity and Ajax

A SIM application like SenSee uses data and application logic of a fundamentally distributed nature, so we have to solve distribution-related problems. One of the main problems is that we cannot rely on those sources. Data and applications can be taken off-line without notice or might be very badly reachable because of network problems. However, what we want to minimize is that the user has to wait indefinitely because of one unavailable or slow source. This problem in the user interface can be tackled by making use of parallelism. To be able to use parallelism in a Web interface setting we used the asynchronous Ajax technology. In this way the user can see results that arrive quickly immediately, while data that arrives later can be easily blended in within the current user interface, without reloading the whole page. If the user clicks 'Submit' with the keyword query 'bike', different ontologies are checked for possible conceptualizations for

[10] http://www.xmlrpc.com/
[11] http://ws.apache.org/xmlrpc/
[12] http://www.w3.org/TR/soap12-part0/
[13] http://jetty.mortbay.org/

this keyword. Due to the asynchronicity in the interface, we can check various ontologies at the same time while results arrive in a undetermined order.

For the implementation we used the Google Web Toolkit[14] (GWT). GWT provides a Java API which enables programming in Java, while afterward the GWT compiles the classes and dynamically converts them to regular HTML and Javascript that handles both the user interface and the asynchronous XML-RPC calls.

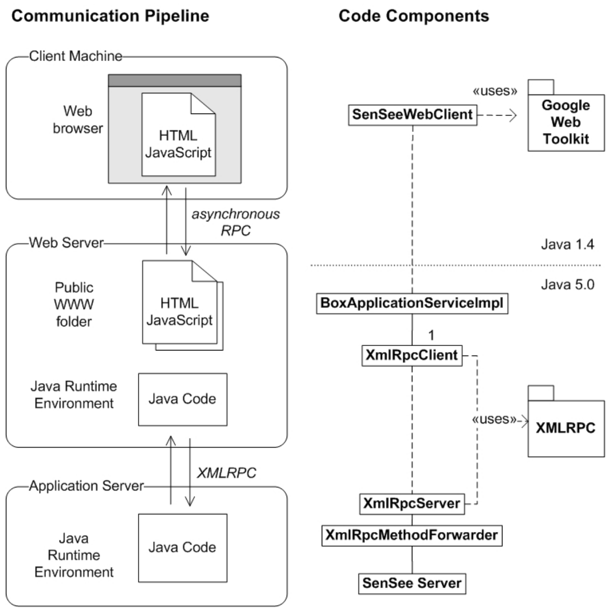

Fig. 3. SenSee communication pipeline

In Figure 3, we see the communication pipeline from a client Web application to the SenSee server. The Web browser initiates the communication by loading a Webpage from the Web server. Via asynchronous RPC calls Java methods are executed on the Java back-end of the Web server. Each one of those method-calls themselves, again forward an XML-RPC call to the appropriate service (the SenSee server, UMS, OS or FS) running on an application server.

6 Semantic-Based Content Integration

When a user fires a request, SenSee responds with an integrated set of content containing pieces from various sources. Like previously mentioned, SenSee retrieves content and metadata descriptions from these various sources which comply with

[14] http://code.google.com/webtoolkit/

different data schemes. However, since SenSee is completely built around the TV-Anytime scheme, we chose to transform all incoming content to TV-Anytime. The TV-Anytime specification consists of two parts, “phases” in TV-Anytime speak. The first phase consists of a synchronized set of specification documents for metadata, content referencing, rights management, and content protection. The second phase defines open standards that build on the foundations of the first phase and includes areas such as targeting, redistribution and new content types. A central concept in the second phase is packaging. A package is a structure which contains a set of unambiguous identifiers referring to content-elements which are somehow related. An example would be a ‘Lord of The Rings’ package containing all the movies, Making-Ofs, the text of the books, images, the soundtracks, etc. To be able to reuse this generated content we add a sense of context to each package from which we can benefit in future requests.

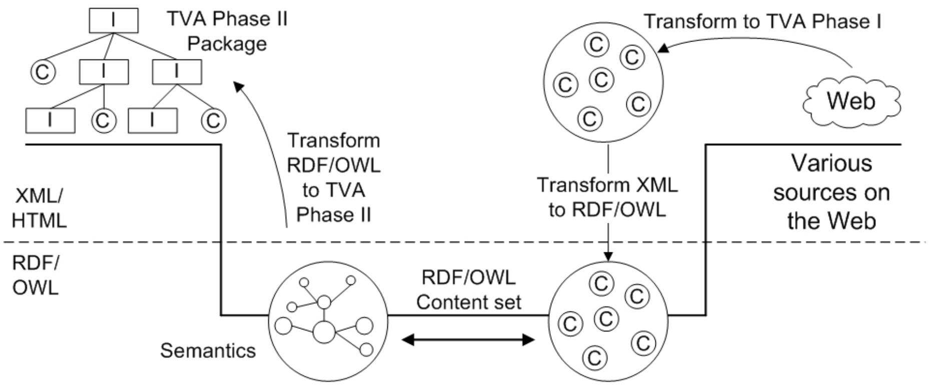

Fig. 4. SenSee data integration scheme

In Figure 4 we see the complete SenSee data integration scheme. In this figure we see that there are two layers visible, namely XML/HTML at the top and RDF/OWL at the bottom of the figure, depicting a clear separation between the two. On the right hand side we see our various sources located on the Web, these sources are typically retrieved in HTML (Wikipedia, IMDB, ...) or XML (XML-TV and BBC Backstage content) formats. This content is then transformed to TV-Anytime Phase I (XML). However, in order to apply our semantics available in the OS which are typically modelled in RDF/OWL, we need to transform the TV-Anytime XML (Phase I). This second transformation crosses the dashed line bringing the TV-Anytime Phase I content set into the RDF/OWL layer. All content here is stored in a Sesame RDF store which serves as a temporal cache to augment and work with the retrieved data. The advantage of our approach of using RDF is that the resulting integrated datamodel can be easily adapted for plugging in and removing data sources and relating those to supporting ontologies because of the open world nature of RDF. In this way we can use inference techniques (e.g. subsumption) to get more useful data for the user and we can utilize mechanisms like the context feature in Sesame 2.0[15] to track the origin of information.

Once here, we can apply the semantics stored in the OS to enrich the current content set. For example, every TV-Anytime Phase I content set always has a set of keywords to describe this content in its metadata. The TV-Anytime (TVA) field for this is:

```
<element name="Keyword" type="tva:KeywordType" minOccurs="0" maxOccurs="unbounded"/>
```

We can make a simple enrichment by searching for synonyms of these keywords, in the WordNet dictionary, and expand the existing set. Similarly, we can also add semantics to time specifications. In TVA every time-related field is modeled by an MPEG-7 'timePointType', where a time point is expressed with the regular expression:

```
'-'?yyyy'-'mm'-'dd'T'hh':'mm':'ss('.'s+)?(zzzzzz)?
```

The date '2007-01-01T12:30:05' is an example. By mapping this string to a time description in our Time ontology we can obtain the following time specification (do note the abbreviated syntax):

```
<CalendarClockDescription rdf:ID="example">
    <second>05</second>
    <hour>12</hour>
    <minute>30</minute>
    <year>2007</year>
    <month>01</month>
    <week>01</week>
    <day>01</day>
</CalendarClockDescription>
```

This richer time description enables us for example to easily cluster all content produced in a particular year or all content being broadcasted before or after a certain point in time.

After this enrichment phase, our content is ready to be used to create a TVA Phase II package for a particular user query. The retrieval of the content was triggered by a user query and the package will now be the (rich) answer to this query. A package is constructed hierarchically and consists out of items, nodes in the hierarchy to enable browsing and navigation, and components which contain the effective content like a movie, a piece of text, an audio file or any other multimedia type.

The simplest package we can generate in SenSee is a package with one item that contains one component, which is effectively an arrangement of the TVA Phase I metadata. However, we are also looking into experimenting with more advanced automatic clustering. We try to cluster content based on relationships of its metadata with helper ontologies, e.g. we relate the TVA Phase I metadata to ontology concepts with string matching techniques. Like this we can make use of the fact that ontologies are well-crafted by domain experts, and therefore

[15] http://www.openrdf.org/doc/sesame2/users/ch04.html#d0e836

the content grouping will be well-structured. Sorting the grouping structure can be done by applying traditional recommendation techniques, e.g. by taking the user model into account.

7 Application in User-Guided Search Support

In the previous sections we demonstrated solutions for the SIM issues. However, the described techniques could also be used to improve the core personalization functionality of the application. Besides offering 'passive' recommendations, the user can also actively search through the available content. Our approach is based on semantic faceted browsing techniques (e.g. refer to [20] or [21]). The idea is to go beyond a pure keyword-based search. Instead the user is assisted to search through different *facets* of the data. To accomplish this, additional metadata fields as defined in TV-Anytime specification were used. Moreover, like previously mentioned, ontological sources to make a further semantical connection between the terms in TV-Anytime play a important role. Consider for instance the time facet, TV-Anytime uses the XML datetime datatype to express the time when a program is broadcasted. To semantically enrich time information we use the W3C OWL-Time ontology, adding for instance that the concept 'afternoon' ranges from 1400h-1800h. In this way we can easily search for programs in the afternoon by looking for all programs broadcast within this 1400h-1800h interval.

By incorporating all these results, as is shown in the screenshot in Figure 5, we try to close the gap between the keywords the user types in and what the user intends to search for. As an example, consider a user making the following keyword request: "sports Scotland 'Friday evening'" when he looks for Scottish sports games to watch during Friday evening when his friends come over. The system reacts by trying to find any ontological matches for these keywords, through the use of the OS. In Figure 5 the results are shown. Under the tab 'Genres' we see that the system found some matches for the keyword 'sport' in the genre classification and in the Geo ontology a match was found for 'Scotland'. In the time ontology a match was found for 'Friday evening' and automatically translated to an interval from 18pm until 23pm as shown in the calender under the 'Time' tab. With this system-assistance the user is able to manually refine the original request. With the genre hierarchy it is possible now to specify a more particular sport type the user might be looking for, or just select the broad term 'sports'. In the geographical hierarchy either a narrower term, like a region in Scotland, or a broader term like 'United Kingdom' can be chosen. Via the calender the user can revise his time interval.

After the system-assistance, the newly refined query is taken as input for the retrieval process which tries to find relevant content made available by the various sources. When the sources start responding with multimedia content, the retrieved content needs to be filtered and personalized by leaving out items deemed unsuited, and ranking important item higher up the results list. This process, which forms the hart of the personalization process, uses mainly the UMS to find the user's preferences, the FS to provide the appropriate filter, and

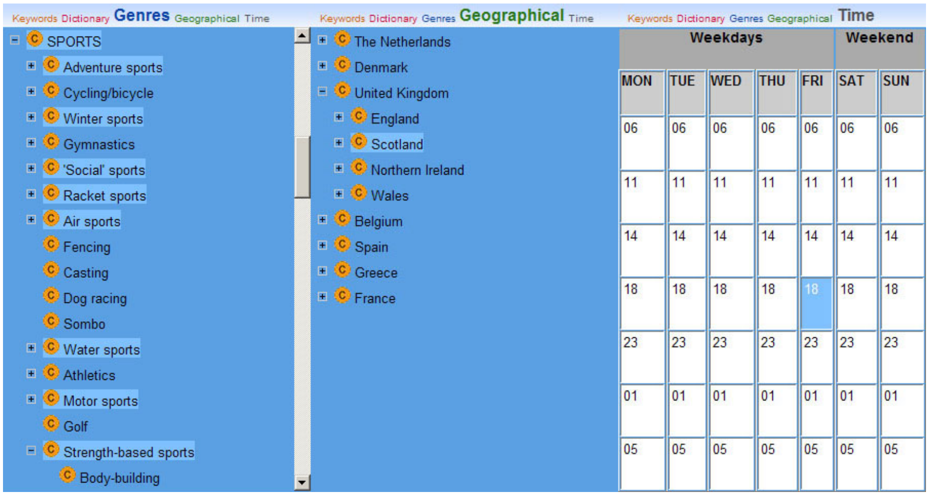

Fig. 5. User-driven concept refinement

the OS to give correct semantics throughout the complete sequence. The filter provided by the FS basically acts as the glue between values in the user model and fields in the content's metadata. It decides for every content-element whether or not it should be discarded after comparing its metadata to the interests and preferences of the user. Lastly, the resulting content set needs to be presented to the user, by means of the TV-Anytime packaging concept. Packages allow us to create for example a 'Marlon Brando' package were everything known from this person can be bundled and browsed. Packages can contain all kinds of content-elements ranging from pure text to full HD (High Definition) movie clips. Like this, the results are clustered and hierarchically structured to obtain an easy-to-browse content-container where the user finds what he or she was looking for or at least is interested in.

8 Conclusion

We have considered a growing class of Web applications that share three characteristic demands that are typical for these emerging applications: semantics for integration, richer interactivity and multi-device adaptation. We have taken an illustrative example of such an application, a TV recommender called SenSee, that tackles these problems with a metadata-driven approach. The main technologies that have been used in the realization of SenSee were techniques from Web Services, Web 2.0 (more specific Ajax technology) and Semantic Web. With Web Services we were able to flexibly connect functional blocks together, e.g. for dynamically choosing the appropriate modality for the appropriate devices. Furthermore, we saw that our approach allowed plugging-in new functionality, but also makes our functionality available to others. Web 2.0 technology like

Ajax was used to improve the user experience. By applying asynchronicity we could accommodate external content sources that may be slow or unavailable, while at the same time the application was staying as responsive as a regular desktop application. Semantic Web technologies were exploited to relate concepts from various heterogeneous sources to the used TV-Anytime specification on which SenSee operates. We also showed that we had an extra benefit as we could use these same semantic techniques to improve the core business of the recommender: by dynamically plugging-in extra knowledge in form of domain ontologies we could give the user more control over the process of finding what they really wanted.

The experience gained in developing SenSee was achieved within the Passepartout project where multiple partners were working closely together. In the project other applications were developed by companies such as Philips Research[16], Stoneroos[17], V2_[18], Henri Tudor[19] and CWI[20], and we observed that our solution proved to be valuable as framework to use, and that SenSee could be extended with new third-party algorithms and add-ons, improving interoperability. The service-based architecture helped us in many ways facilitating these cooperations. We learned how the growing class of Web applications with SIM characteristics can be built. Although there might also be other ways to accomplish this, the semantic-based, metadata-driven approach we took allows the application of technologies that can get the job effectively done.

References

1. Allan, J., B.C.(eds.).: Challenges in information retrieval and language modeling. SIGIR Forum 37(1), pp. 31–47 (2003)
2. Brafman, R.I., Domshlak, C., Shimony, S.E.: Qualitative decision making in adaptive presentation of structured information. ACM Trans. Inf. Syst. 22(4), 503–539 (2004)
3. Ardissono, L., Goy, A.: Tailoring the interaction with users in web stores. User Modeling and User-Adapted Interaction 10(4), 251–303 (2000)
4. Ardissono, L., Console, L., Torre, I.: An adaptive system for the personalized access to news. AI Communications 14(3), 129–147 (2001)
5. Webster, D., Huang, W., Mundy, D., Warren, P.: Context-orientated news riltering for web 2.0 and beyond. In: WWW '06. Proceedings of the 15th international conference on World Wide Web, New York, NY, USA, pp. 1001–1002. ACM Press, New York (2006)
6. Millard, D.E., Ross, M.: Web 2.0: hypertext by any other name? In: HYPERTEXT '06. Proceedings of the seventeenth conference on Hypertext and hypermedia, New York, NY, USA, pp. 27–30. ACM Press, New York (2006)

[16] http://www.research.philips.com/
[17] http://www.stoneroos.nl/
[18] http://www.v2.nl/
[19] http://www.tudor.lu/
[20] http://www.cwi.nl/

7. Jones, M.B.: The identity metasystem: A user-centric, inclusive web authentication solution. In: Toward a More Secure Web - W3C Workshop on Transparency and Usability of Web Authentication (2006)
8. Passepartout: Itea passepartout project. (2005-2007) `http://wwwis.win.tue.nl/~ppartout`
9. Ardissono, L., Kobsa, A., Maybury, M.T. (eds.): Personalized Digital Television. Human-Computer Interaction Series, vol. 6. Springer, Heidelberg (2004)
10. Blanco Fernández, Y., Pazos Arias, J.J., Gil Solla, A., Ramos Cabrer, M., López Nores, M.: Bringing together content-based methods, collaborative filtering and semantic inference to improve personalized tv. 4th European Conference on Interactive Television (EuroITV 2006) (May 2006)
11. van Setten, M.: Supporting people in finding information: Hybrid recommender systems and goal-based structuring. Telematica Instituut Fundamental Research Series, No.016 (TI/FRS/016). Universal Press (2005)
12. TV-Anytime. The TV-Anytime Forum, Requirement Series: RQ001v2.0, TV140. (April 2003) Available at: `ftp://tva:tva@ftp.bbc.co.uk/pub/Plenary/0-Plenary.html`
13. Earnshaw, N.: The tv-anytime content reference identifier (crid) (2005)
14. Akkermans, P., Aroyo, L., Bellekens, P.: European Semantic Web Conference 2006 (demo presentation) (2006), `http://www.eswc2006.org/demo-papers/FD36-Lora.pdf`
15. Hobbs, J.R., Pan, F.: An ontology of time for the semantic web. ACM Transactions on Asian Language Information Processing (TALIP) 3(1), 66–85 (2004)
16. Chipman, A., Goodell, J., Johnson, R., Ward, J.: Getty thesaurus of geographic names: editorial guidelines (2005) `http://www.getty.edu/research/conducting_research/vocabularies/guidelines/tgn_1_contents_intro.pdf`
17. Manola, F., Miller, E.: Resource Description Framework (RDF). `http://www.w3.org/TR/rdf-primer/`
18. McGuinness, D.L., van Harmelen, F.: OWL web ontology language. `http://www.w3c.org/TR/owl-features`
19. Broekstra, J., Kampman, A., van Harmelen, F.: Sesame: A generic architecture for storing and querying rdf and rdf schema (2002)
20. Yee, K.P., Swearingen, K., Li, K., Hearst, M.: Faceted metadata for image search and browsing. In: CHI '03. Proceedings of the SIGCHI conference on Human factors in computing systems, New York, NY, USA, pp. 401–408. ACM Press, New York (2003)
21. Hildebrand, M., van Ossenbruggen, J., Hardman, L.: /facet: A browser for heterogeneous semantic web repositories. In: Cruz, I., Decker, S., Allemang, D., Preist, C., Schwabe, D., Mika, P., Uschold, M., Aroyo, L. (eds.) ISWC 2006. LNCS, vol. 4273, pp. 272–285. Springer, Heidelberg (2006)