On-the-fly Data Integration for Personalized Television Recommender Systems

Pieter Bellekens, Kees van der Sluijs, Geert-Jan Houben[*], Lora Aroyo[†]
Technische Universiteit Eindhoven
P.O. 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

In this paper we present our iFanzy demonstrator, which aims at offering users television content in a personalized and context-sensitive way. With iFanzy we integrate different volatile content sources like video-on-demand, broadcast and Web information on-the-fly to disperse it on different clients on different platforms like set-top boxes and Web interfaces. Via Semantic Web techniques, TV content and background data from various heterogeneous sources is integrated into a transparent RDF/OWL knowledge graph, which allows the user to navigate and browse the vast content sets nowadays available. Via illustrative examples we explain how the main integrated data set is built and how we exploit it to provide iFanzy's main functionality, like semantic search of the available content and execution of context-sensitive recommendations.

1. Introduction

Television providers more and more turn to the Web to give users added functionality like personalization, recommendation, integration and navigation to the television platform [1], [5]. The projected goal is at one hand combining broadcasted video material with video-on-demand as for a large part can be offered via the Web, but also to extend the metadata from Electronic Program Guides (EPGs) with additional metadata that can be found on the Web. Furthermore, it allows to connect a television (e.g. via a set-top box) to the Internet, connecting and synchronizing various devices (like a mobile phone, a PC, etc.) with the TV system to provide a ubiquitous content framework for the user to connect to anywhere anytime. Each device in this setting has its own function based on its characteristics. For example the PC is an medium that users typically use to searching and navigating metadata, where watching TV remains typically a rather passive activity [2].

Our demo shows iFanzy[1], a personalized TV access point which consists out of a central server and a set of clients which helps users to find the programs they want from an enormous set of data sources. iFanzy currently has interfaces for two platforms up and running, namely one that runs on a set-top box and two that run as a Web application [2] while an interface for the mobile is under development. See for example Figure 1 for a screenshot of the redesigned iFanzy Web interface. On the surface the interface looks rather similar to a normal EPG. What one can see on the screenshot is an overview of programs on public Dutch for a certain evening. The programs are colored based on semantic match of the program with the user profile. For every program one can see a bar representation of how well the program fits with the current profile according to the recommendation system and a slider with which the user can influence this value if he does not agree with the recommendation (either positive and negative feedback).

All these interface clients are based on the services provided by our Web Server (called SenSee) as was for a great part introduced in our ICWE paper from last year [3]. SenSee includes services like synchronization, personalization and integration of the data sources. In order to achieve this integration of large data sources we used Semantic Web techniques, e.g. to connect program classes and instances, adding relations between abstract types like genres and formats, modeling locations and time, etc. This network allows us to infer new relations and to reason over the available knowledge structure. For example, when a user browses a program (e.g. a movie) broadcasted by the BBC we are able to show related movie information from IMDB[3] and we can show related programs (e.g. same genre, same director, etc)

[*] also affiliated with Vrije Universiteit Brussel, Brussels, Belgium
[†] also affiliated with Vrije Universiteit, Amsterdam, the Netherlands

[1] developed in collaboration with Stoneroos Interactive TV, Ltd., refer to *http://www.stoneroos.nl*
[2] For the new interface (work in progress) refer to *http://www.iFanzy.nl*. For our running prototype refer to *http://wwwis.win.tue.nl/SenSee*.
[3] *http://www.imdb.com*

978-0-7695-3261-5/08 $25.00 © 2008 IEEE
DOI 10.1109/ICWE.2008.20

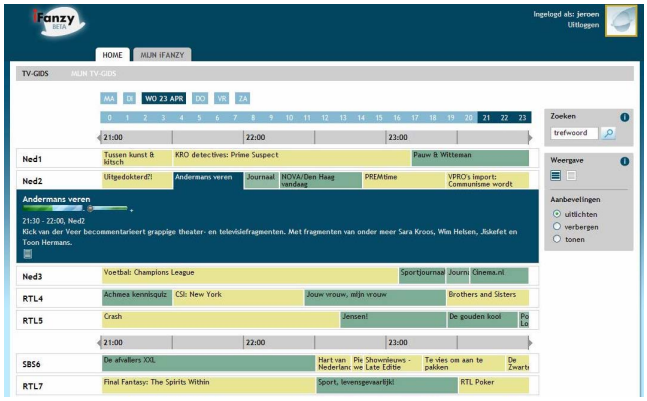

Figure 1. iFanzy EPG view

on different channels. These relations between data sources provide an excellent means to help the user in navigating and searching through the huge amount of available programs.

Since our previous ICWE paper we made some big steps forward. One big improvement is the overall performance of the system. We managed to get response times from user queries down from minutes to subsecond. The main topic we will discuss in this paper however is data integration. One major challenge there was the volatility of our data sources. Every day new programs are broadcasted, alongside their respective streams of metadata, while yesterday's content is discarded. Even almost until the effective broadcast of a certain program, metadata can be changed and updated. This gives rise to need of frequent data updates and forces us to find relationships between metadata sources on the fly. In the rest of the paper we will describe, using illustrative examples, how we constructed our integrated content graph and how we can exploit to provide recommendations and personalization.

Like previously mentioned, other TV recommender systems exist, e.g. AVATAR [5] which has a focus on reasoning over TV content metadata and user preferences. iFanzy differs from AVATAR mainly because of our focus on combining and integrating the information from several large and live data sources. Many systems exist that focus on the recommendation part, for example the movie recommendation application MovieLens[4] that uses collaborative filtering. For an overview of different recommendation strategies e.g. refer to [6].

2 Semantic Structure

The use of semantics is an important instrument in order to combine and integrate the content from different applications and in this way to enhance personalization. In this sense iFanzy represents a large class of multi-device applications with a high degree of interactivity where semantics is key to effective integration [3]. We can differentiate three main steps to enable searching, recommendation and personalization strategies in a flexible way.

Step 1: Making TV metadata available in RDF/OWL
Adding a new TV metadata source to the system requires it to be in the RDF/OWL format making sure that it fits and can connects to already available data sources. In the current iFanzy demonstrator we use three live data sources, online TV guides (e.g. 1.2M triples for the daily updated programs), online movie databases such as IMDB (e.g. 83M triples, representing 1M movies and trailers from Videodetective.com), and broadcast metadata available from BBC-backstage (e.g. 92K triples, daily updated). All three sources were parsed and converted to our TV-Anytime[5] domain model in RDF/OWL.

Step 2: Making relevant vocabularies available in RDF/OWL
Having the metadata available, it is also necessary to make relevant vocabularies available in RDF/OWL. In iFanzy we did this in a SKOS-based manner for the genre vocabularies (resulting in 5K triples). All these genres play a role in the classification of the TV content and the user's likings (in order to support the recommendation). We also used the locations hierarchy used in IMDB (60K triples), WordNet 2.0[6] (2M triples) and the OWL Time Ontology[7] both as published by W3C.

Step 3: Aligning and enriching vocabularies/metadata
Here we did (1) alignment of Genre vocabularies (e.g. aligning `imdb:sci-fi` and `tva:science_fiction`), (2) semantic enrichment of the Genre vocabulary in TV-Anytime (e.g. relating `TVAGenres:Sport` and `TVAGenres:Sport_News`), and (3) semantic enrichment of TV metadata with IMDB movie metadata (e.g. connecting the broadcasted movie "Il Buono, il Brutto, il Cattivo" to the IMDB movie "The Good, the Bad and the Ugly").

3 Building the Graph

Constructing the RDF/OWL graph out of very different data sources and vocabularies creates a whole new set of relations and interconnections between previously completely separate television program instances. Let us look at the following example. Imagine we retrieve metadata of two separate TV programs broadcasted on different channels retrieved from different sources (do note that some syntax is abbreviated due to space constraints):

[4] http://www.movielens.org/

[5] *http://www.tv-anytime.org/*

[6] *http://www.w3.org/2006/03/wn/wn20/*

[7] *http://www.w3.org/TR/2006/WD-owl-time-20060927/*

```
<program channel="NED1">
    <source>http://foo.bar/</source>
    <title>Sportjournaal</title>
    <start>20080309184500</start>
    <end>20080309190000</end>
    <genre>sport nieuws</genre>
</program>

<program title="Match of the Day">
    <channel>BBC One</channel>
    <start>2008-03-09T19:45:00Z</start>
    <duration>PT01H15M00S</duration>
    <genre>sport</genre>
</program>
```

Both programs are sport programs broadcasted on the same day. However, since the data is coming from different sources, the syntax and semantics differ. Like explained in 2, all instances from our different sources need to be converted to the central TV-Anytime domain in RDF/OWL. After this transformation the generated RDF/OWL looks (for the first program) like:

```
<TVA:ProgramInformation ID=
        "crid://foo.bar/0001">
    <hasTitle>Sportjournaal</hasTitle>
    <hasGenre rdf:resource=
        "TVAGenres:Sport_News"/>
</TVA:ProgramInformation>

<TVA:Schedule ID="TVA:Schedule_0001">
    <serviceIDRef>NED1</serviceIDRef>
    <hasProgram crid=
        "crid://foo.bar/0001"/>
    <startTime rdf:resource=
        "TIME:TimeDescription_0001"/>
</TVA:Schedule>
```

While doing so, references to time like the start time of a program are transformed to a Time ontology instance and source specific genres are matched to their TV-Anytime counterparts.

```
<TIME:TimeDescription ID=
    "TIME:TimeDescription_0001">
    <year>2008</year>
    <month>3</month>
    <day>9</day>
    <hour>18</hour>
    <minute>45</minute>
    <second>0</second>
</TIME:TimeDescription>
```

The RDF/OWL graph constructed out of the vocabulary knowledge and the TV metadata sources now contains a set of triples which models each of the two originally retrieved programs. Because both of them are now a part of the same graph, new relations can be derived through reasoning. For example, since both now have a time description modeled according to the Time ontology specification, we can compare their starting times and conclude that ”Match of the Day” on BBCOne starts 45 minutes later than ”Sportjournaal” on the same day.

Also the genres of the two programs are now mapped to a TV-Anytime equivalent. Above we can see that the ”Sport nieuws” genre (in Dutch) has been mapped to the `TVAGenres:Sport_News` genre. Analogously the ”sport” genre from BBC is mapped to the `TVAGenres:Sport` genre. However, like we saw in 2, new relations were made between different genres in the TVA hierarchy by means of the SKOS relations: `skos:related`, `skos:narrower` and `skos:broader`. For these specific genres the following relation exists:

```
<TVAGenres:genre ID="TVAGenres:Sport">
    <rdfs:label>Sport</rdfs:label>
    <skos:related rdf:resource=
        "TVAGenres:Sport_News"/>
</TVAGenres:genre>

<TVAGenres:genre ID=
        "TVAGenres:Sport_News">
    <rdfs:label>Sport News</rdfs:label>
    <skos:related rdf:resource=
        "TVAGenres:Sport"/>
    <skos:broader rdf:resource=
        "TVAGenres:News"/>
</TVAGenres:genre>
```

These relations in turn help in making the graph more interconnected allowing us to find more related content and use these relations to find even more relations through custom reasoning (e.g. if all movies directed by Director D are action movies, then there must be a relation between D and the genre ’Action’).

4 Exploiting the Graph

To recommend TV programs or movies, the resulting RDF/OWL graph is extended with the user model such that the eventual RDF/OWL knowledge structure can be directly used for the recommendation. When the user rates a program P this is stored in the user model. When the system then gives the user program recommendations, the rating for program P is propagated via all relations between P and other programs in the graph. In this way a user can get recommendations for programs that have for example the same

or a related genre as P. Which relations to use and with which propagation weights is configurable in SenSee. We for instance use `skos:related` and `skos:narrower` relations in our genre hierarchy to find related genres.

We also look at all persons connected to a certain TV program. Persons can be connected to programs via different relations like: actor, director, guest, presenter, writer, etc. In the same way, every person associated with P is indirectly rated, where the type of relation influences the eventual rating given (this is again configurable). For example, we consider a person always presenting a certain program is more closely related to this program than a guest who only appears once.

The more connected the graph is the further we could propagate a certain rating. Obviously, we could keep on propagating a rating throughout the entire graph, although, with every step the relevance of the rating diminishes. Therefore, we maintain a propagation threshold for which given the weights of the relations stop the propagation.

We also store contextual information in the user model. Context is important in this application because people tend to like different programs in different situations. While a father likes to watch the morning news to anticipate possible traffic jams, a mother will be more interested in the weather forecast to adapt her children's clothing before being send of to school. In the evening program preferences might be entirely different, as the entire family might want to sit back to enjoy a movie. Therefore, we store for a rating the context in which this rating is given, if this is desirable by the user. It is the responsibility of the client application (currently used to connect to the server) to see in which context the user or group of users is/are residing. Contexts which are currently supported are 'current time', 'current location' and 'current audience'. To see who is watching the login feature is used (only the people logged into the system are considered), although for future work we consider scenarios to let people automatically be recognized by their mobile phones, a personal RFID tag or via a fingerprint reader (e.g. on the remote control).

For querying we use query expansion, based on again exploiting ontology relationships. We do this to heavily improve recall as well as to enlarge the serendipity of the result. As the size of available data sources grows we need to be able to come up with surprising results which still have some (maybe faint) relation to what the user really wants, so that the user does not get stuck stuck with recommendations for a number of programs they happen to already know without considering other possibly interesting programs that might be interesting to the user, but that the user until now was unfamiliar with. For every user request we look at the input and try to extend the concepts and/or keywords with connected concepts and keywords. To accomplish this we look at relations like the synonym relation from WordNet and the `skos:narrower` and `skos:related` relationships from the vocabularies. Also in this case we carefully look at the user's profile to rank the result set. Results which are liked less by this user (or group of users) or are liked less in the current applicable context can be pushed lower down the result set.

5 Conclusions

In this paper we discussed our iFanzy demonstrator and we more specifically zoomed in on the topic of on-the-fly data integration in the television domain. The integrated data set has many interconnections which lead to better (semantic) recommendations, richer navigation and allows us to put in a serendipity factor.

Different versions of the different clients and server systems have been implemented and successfully evaluated in collaboration with our commercial partner Stoneroos. Based on our practical experiences we are currently redesigning the iFanzy frontend and SenSee backend, improving amongst others the user interface and further optimizing the performance [4](e.g. by parallelization). Furthermore, an evaluation trial with 500 set-top boxes in Dutch households is prepared together with Stoneroos, in which we will look at how users appreciate the functionality as well as real-time performance benchmarking.

References

[1] L. Ardissono, A. Kobsa, and M. T. Maybury, editors. *Personalized Digital Television*, volume 6 of *Human-Computer Interaction Series*. Springer, 2004.

[2] L. Aroyo, P. Bellekens, M. Bjorkman, G.-J. Houben, P. Akkermans, and A. Kaptein. Sensee framework for personalized access to tv content. In *Interactive TV: a Shared Experience*, pages 156–165, Amsterdam, the Netherlands, 2007. Springer.

[3] P. Bellekens, K. van der Sluijs, L. Aroyo, and G.-J. Houben. Engineering semantic-based interactive multi-device web applications. In *ICWE*, pages 328–342, 2007.

[4] P. Bellekens, K. van der Sluijs, W. van Woensel, S. Casteleyn, and G.-J. Houben. Achieving efficient access to large integrated sets of semantic data in web applications. In *Proceedings of the 8th International Conference on Web Engineering (ICWE'08), to be published*.

[5] Y. Blanco Fernandez, J. J. Pazos Arias, A. Gil Solla, M. Ramos Cabrer, and M. Lopez Nores. Bringing together content-based methods, collaborative filtering and semantic inference to improve personalized tv. *4th European Conference on Interactive Television (EuroITV 2006),*, May 2006.

[6] M. van Setten. 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.