# DESIGN, DEVELOPMENT AND DEPLOYMENT OF AN INTELLIGENT, PERSONALIZED RECOMMENDATION SYSTEM

by

VICTOR CODINA BUSQUET

A THESIS

submitted in partial fulfillment of the requirements for the degree of

MASTER IN ARTIFICIAL INTELLIGENCE

UNIVERSITAT POLITÈCNICA DE CATALUNYA (UPC)

BARCELONA, SPAIN

SEPTEMBER 2009

Approved by:

Dr. LUIGI CECCARONI

# Abstract

Personalization and recommendation systems are a solution to the problem of content overload, especially in large information systems. In this thesis, a personalized recommendation system enhanced with semantic knowledge has been developed in order to overcome the most common limitations of traditional approaches: the cold-start and the sparsity problems. The recommender consists of the following two main components. A user-profile learning algorithm combines user's feedback from different channels and employs domain inferences to construct accurate user profiles. A recommendation algorithm, using content-based filtering, exploits the semantic structure of the domain to obtain accurate predictions and generate the corresponding recommendations. The system's design proposed is flexible enough to be potentially applied to applications of any domain that can be properly described using ontologies. In addition to the development of the recommendation system, an existing Web-application in the tourism domain has been extended and adapted in order to be able to integrate the recommender into it. The overall recommendation system has been evaluated and the results obtained indicate that it satisfies the requirements established.

# Table of Contents

# List of Figures

# List of Tables

# Acknowledgements

I would like to thank my supervisor, Dr. Luigi Ceccaroni at the Universitat Politècnica de Catalunya (UPC) for his advice throughout the course of my work and for his precious support for reviewing my thesis writing. Also, I would like to thank Maria Salamó, an expert in the field of recommendation systems, for her valuable ideas and support for designing the recommender algorithms. Special thanks also to my girlfriend Gemma, for her support during the course of the thesis.

# CHAPTER 1 - Introduction

The general purpose of this thesis was the development of a personalized recommendation system and its integration within a specific tourism-domain Web-application that is part of the INREDIS research project (explained in the following section 1.1). However, because the project is doing research in more domains such as the e-commerce and finances, it was desirable that the recommender design was flexible enough to be easily integrated within any application domain. This particular requirement mainly led the way in how the system was designed and made it especially different from the existing approaches.

This chapter presents the starting point of this work, the INREDIS project, the main motivations for developing adaptive[1] Web systems (in particular, the ones behind the undertaken work), as well as the general and specific objectives of the thesis.

## 1.1. Starting point: the INREDIS project

INREDIS (Interfaces de Relación entre el Entorno y las personas con Discapacidad) is a CENIT (Consorcios Estratégicos Nacionales de Investigación Técnica) project partially funded by the government of Spain with a budget of 23M €, whose consortium includes 14 companies, such as TMT Factory, Technosite, Vodafone, e-laCaixa, Moviquity, and Barclays; and 18 research organizations, such as UPC, UCM, URL, and UV. INREDIS is a project that does basic research in the field of accessible and interoperable technologies. The project's goal is to develop base technologies that allow building communication and interaction channels between people with special needs and their environment.

As a demonstration of what the accessible and interoperable technologies developed can be used for, INREDIS produced a Web-based prototype based on tourism information services using Interactive Community Displays (ICD) located in public spaces of the city as main platform. ICDs, integrated with posters, city information panels, bus stop shelters, kiosk systems, and interior panels, are an ideal channel to provide the city semantically-rich services through map-based interfaces. An example of ICD integrated with an interior panel can be observed in Figure 1.1.

[1] In this thesis, the term "adaptive" is used as a synonym of "personalized".

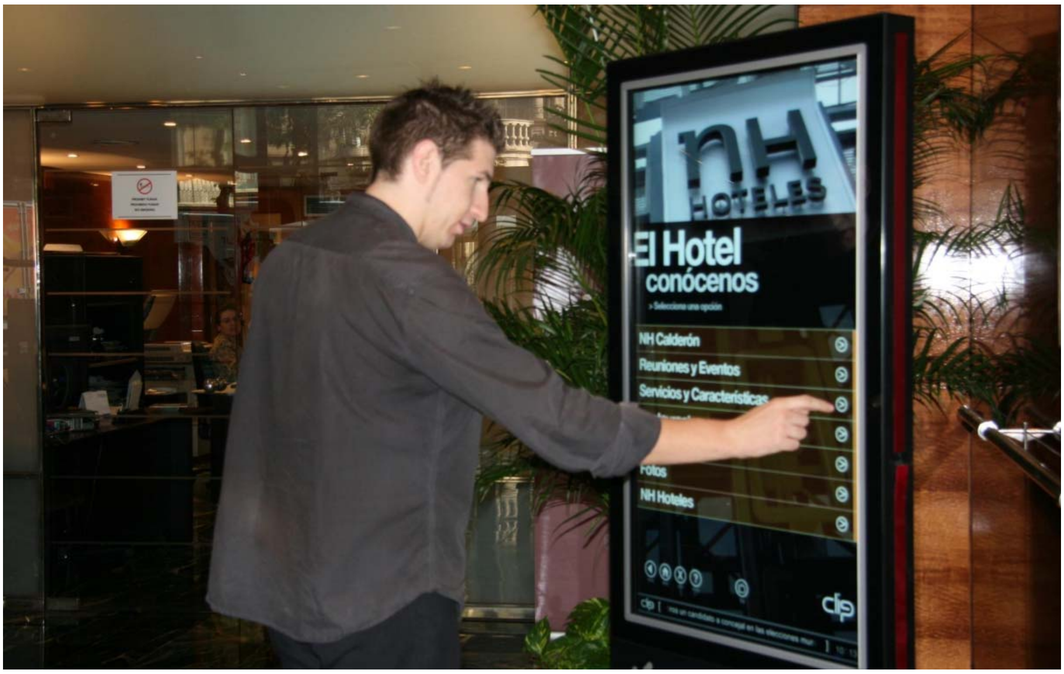


**Figure 1.1** An example of ICD

Before starting this work, the prototype consisted of an online Web system which provided and facilitated access to urban information services in real time through multimodal interfaces, a system which allowed the users haptic interactions as well as speech and sign recognition. Concretely, the services implemented were the following: an *interactive map*, in which the user had information available about all surrounding locations all the time; and *description of places of interest*, providing the user a brief description of places tagged on the map manually gathered from the Barcelona Council's website[2]. In the Figure 1.2 the front-page of the tourism service is presented. The key components of the interface are: the interactive map, in the uppermost panel, the command panel, which facilities the access to the main functionalities of the information service; and the category-based search-interface, situated in the lowermost panel.

[2] See [http://www.bcn.cat/] (last access on September 3, 2009)

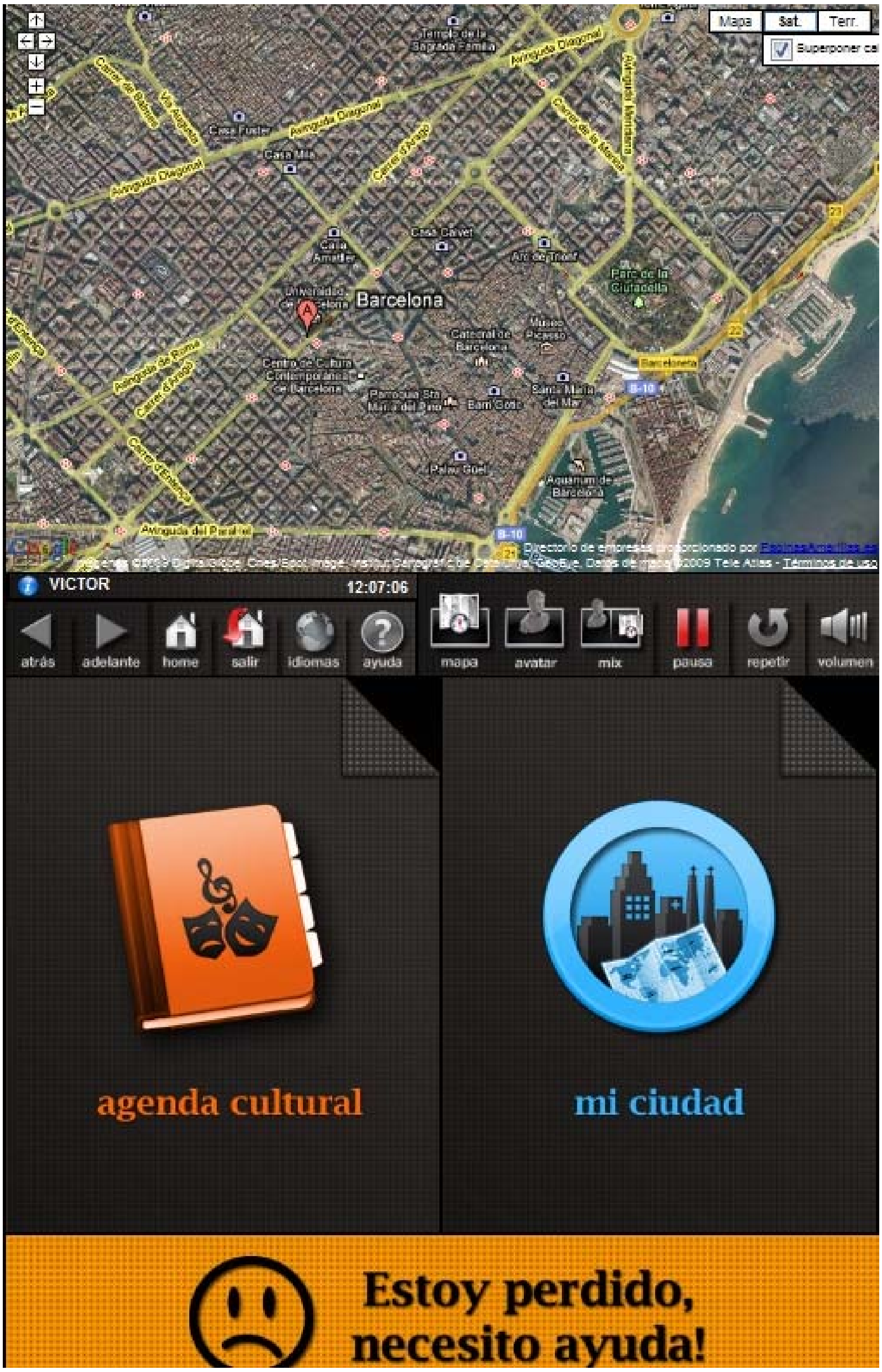
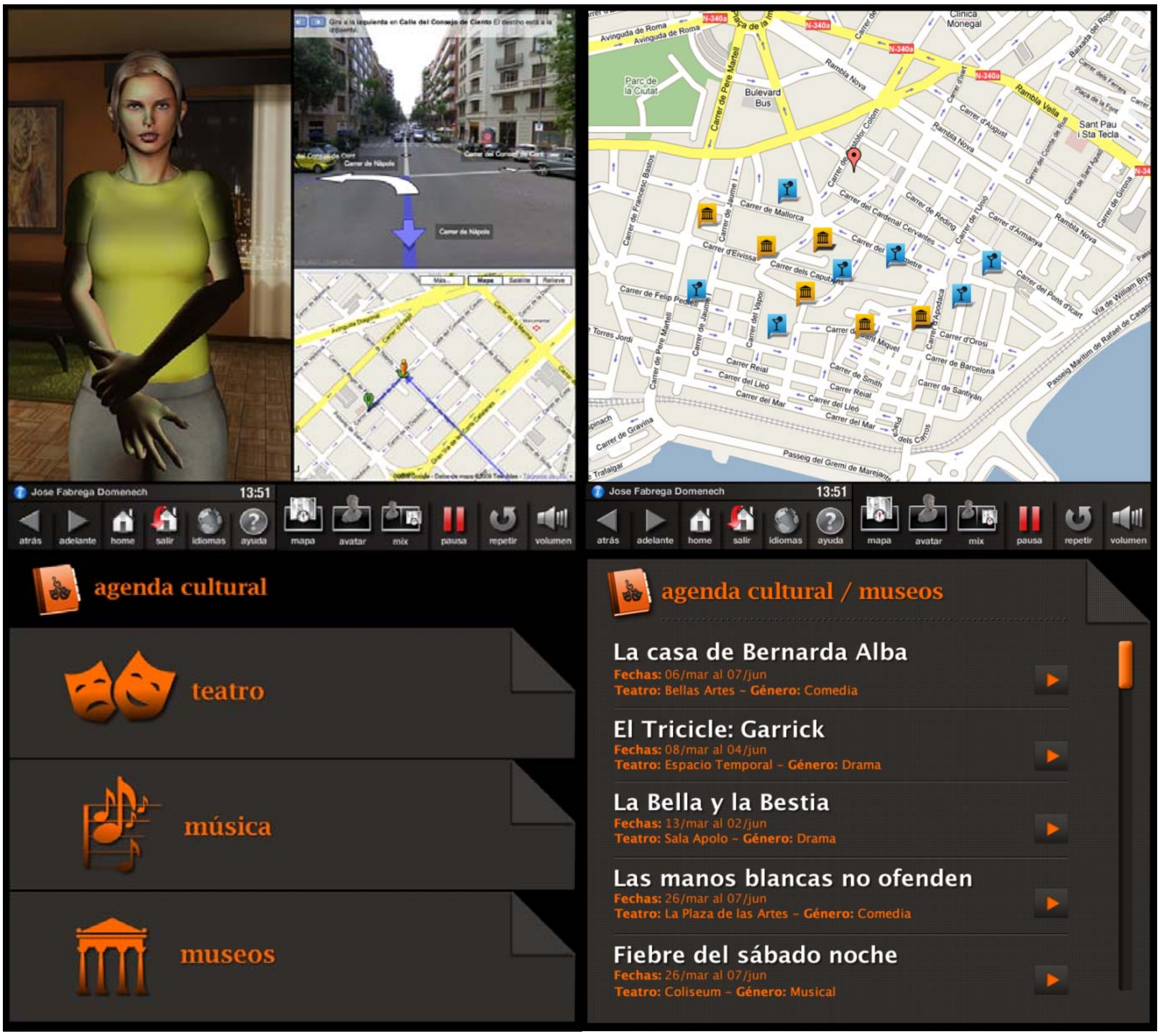


**Figure 1.2** Front page of the INREDIS prototype

In Figure 1.3 the navigation interface in Spanish is presented in more detail. In the left part, the main categories of the Agenda information service can be observed, in conjunction with the avatar interaction mode. In the right part, the result of the query “museos” (Spanish for “museums”) is shown to the user by using a plain list of events ordered by proximity.



La casa de Bernarda Alba
Fechas: 06/mar al 07/jun
Teatro: Bellas Artes - Género: Comedia

El Tricicle: Garrick
Fechas: 08/mar al 04/jun
Teatro: Espacio Temporal - Género: Drama

La Bella y la Bestia
Fechas: 13/mar al 02/jun
Teatro: Sala Apolo - Género: Drama

Las manos blancas no ofenden
Fechas: 26/mar al 07/jun
Teatro: La Plaza de las Artes - Género: Comedia

Fiebre del sábado noche
Fechas: 26/mar al 07/jun
Teatro: Coliseum - Género: Musical

**Figure 1.3** Interface design of the INREDIS prototype

Part of the work of this thesis consisted of extending the INREDIS prototype in order to be able to integrate a *personalized recommendation service*, providing different options for places of interest, according to the profile of the user identified and the user's context, such as the current user's location, the current weather and the time of the day).

## 1.2. Motivations for adaptive Web systems

In an era of increased availability of digital content, there is a need of personalized tools to help people select what they consume. This problem, known as information overload, is yet more exposed in information systems that cover a large information space, like the Web, where it is supposed that individual users have different knowledge and information needs. Traditional Web-based information systems suffer from an inability to satisfy these heterogeneous needs, providing general interfaces and the same information for the same query for all the categories of users. Thus systems capable to adapt the results to the particular user's interests are necessary.

Web personalization [1] is a recent field which was originated to address the deficits of traditional Web systems. Researchers started developing adaptive Web systems that tailored their appearance and behavior to each individual user and, in the last years, the field has evolved into a large research field attracting scientists from different communities such as hypertext, user modeling, machine learning, natural language generation and information retrieval. Adaptive systems have been designed for different usage contexts and explored different kinds of personalization. The most typical usage contexts are: the *adaptive search systems*, promoting items in result lists that they estimate more relevant to the user's interests and needs than others; *adaptive hypermedia systems,* tailoring page content to the respective user and emphasizing recommended links; *adaptive filtering and recommendation systems*, complementing search and browsing based information access by proactively recommending items that seem most relevant to users' interests and might otherwise be missed due to information overload.

Personalized recommendation systems are one possible solution to the problem of content overload, whose main objective is to present to the users information-items (such as movies, music, books, news and Web pages) that may be appealing to them taking into account their personal preferences. These technologies are generally based on content, previous cases, cooperation among users or human-generated links between content items, and may take into account that taste is context-sensitive and evolves over time. A key piece of these technologies for obtaining successful personalization is the maintenance of the user model that reflects the real user's interests and preferences at a specific moment.

The current trend is to develop hybrid recommender systems that combine characteristics of different filtering methods (see section 2.3), in order to minimize the disadvantages of each of them and thus improve the overall efficiency of the system's performance in terms of accuracy and comprehensiveness. The dominant methods used in many of top e-commerce sites in order to improve the sales are the collaborative-filtering algorithms (CF). The main reason is that they are not item domain bound. Basically, these methods express user preferences as item ratings and recommendations are based on matching users with similar ratings, assuming that high correlation in ratings among users is an indicator of taste overlap.

The two major technical problems with CF methods are 1) the *cold-start problem* [2] and 2) the *sparsity problem* [3]. The former refers to the fact that the system cannot compute any recommendations for a new user because it has no information about his preferences. The latter is about the fact that the number of people who have rated particular items in the database might be relatively small compared to the total number of items. This means that might not be significant similarity among users leading to possibly lower quality recommendations as they are based on poor information.

## 1.3. Objectives

The general objective of this thesis is to develop a personalized recommendation system and integrate it into the INREDIS prototype: a Web-application in the tourism domain. This adaptive Web system combines two different usage contexts: the adaptive search, in which the users receives a personalized list of places related with a given query; and the adaptive filtering, in which the system proactively recommends a list of places based on the user's context and interests.

The specific objectives accomplished by this work are:

1) To overcome the typical problems of current recommenders: the *sparsity problem* and the *cold-start problem*. This means that, in addition to using rich data domains with a high density of user-item ratings, the system can also produce effective personalized recommendations in these situations:
   - domains with sparse data, such as the tourism domain;
   - new users whose profile information is very poor.

   As a consequence, the recommender system cannot only rely on typical CF techniques in which the quality of the recommendations is highly dependent to the available density of rates. A recent approach which seems to be more suitable than CF methods dealing with these limitations is to incorporate semantics into the recommender system. For this reason, this thesis has developed a semantic recommender system (see section 2.5) and tries to demonstrate the hypothesis that this approach can be a better option in order to overcome these problems.
2) To design a domain-independent recommendation system. Instead of designing a recommender uniquely thought to be used for a tourism-domain Web-application, one of the objectives of this work is that the recommender can be employed for any Web-application domain without too much effort. Therefore, the approach is flexible enough to be applied in diverse domains.

3) To acquire and learn the user model in an unobtrusive manner. This implies that the system preferably uses implicit knowledge acquisition methods in order to build the user model. The basic idea is that the recommender system has the less impact on the user regular activities. Another hypothesis that this work assumes based on recent studies [4], in favor to use implicit knowledge as a primary source of information, is that explicit feedback based on ratings, typically used in CF methods, might not be a confident indicator of user’s tastes in some domains because most users are inconsistent in giving their feedback.
4) To be able to construct a user model even in absence of previous usage data from the users. Therefore, typical techniques used in adaptive Web systems based on Web usage mining [5] cannot be applied to construct the user profile because these kinds of methods need large volumes of user historic data to obtain reliable results.

## 1.4. Organization of the thesis

This thesis is organized as follows. Chapter 2 presents a general review of the state-of-the-art in personalized recommendation systems and, concretely, in semantic recommenders. Chapter 3 describes the main elements of the recommendation system developed. Chapter 4 presents the key aspects of the development of the Web-application in the tourism domain as an extension of the INREDIS prototype, and how the recommender has been integrated into it. Chapter 5 presents the undertaken experimental evaluation and discusses the results obtained. Finally, Chapter 6 and 7 draw some conclusions and possible future work respectively.

# CHAPTER 2 - State of the art

## 2.1. Recommendation and personalization

Recommendation and personalization are concepts inter-related. Actually, personalization can be seen as a type of recommendation in which the objective is to provide a personalized experience to the user. Different definitions of both concepts recommendation [6] [7] and personalization [8] can be found in literature.

A recommendation can be considered "non-personalized" if it does not depend on a user profile. In these cases, recommender systems do not distinguish users as individuals and normally provide the same recommendations to users with different characteristics. In contrast, personalized recommendations are those based on user data which is collected and represented into user profiles. In this work, the recommenders whose main objective is to provide a personalized experience to the users by means of modeling their interests and preferences are referred to as *personalized recommendation systems*. The general process that follows this kind of automatic personalization systems consists of an iterative process that can be defined by two main stages (see Figure 2.1):

1. the user modeling process, in which the system creates and maintains an up-to-date user profile by collecting data from various sources of feedback that may include implicitly observing user behavior and explicitly requesting direct input from the user (see section 2.2);
2. the content adaptation or recommendation process, in which the system delivers personalized recommendations based on the knowledge contained in users' profiles by means of combining different recommendation techniques (see section 2.3).

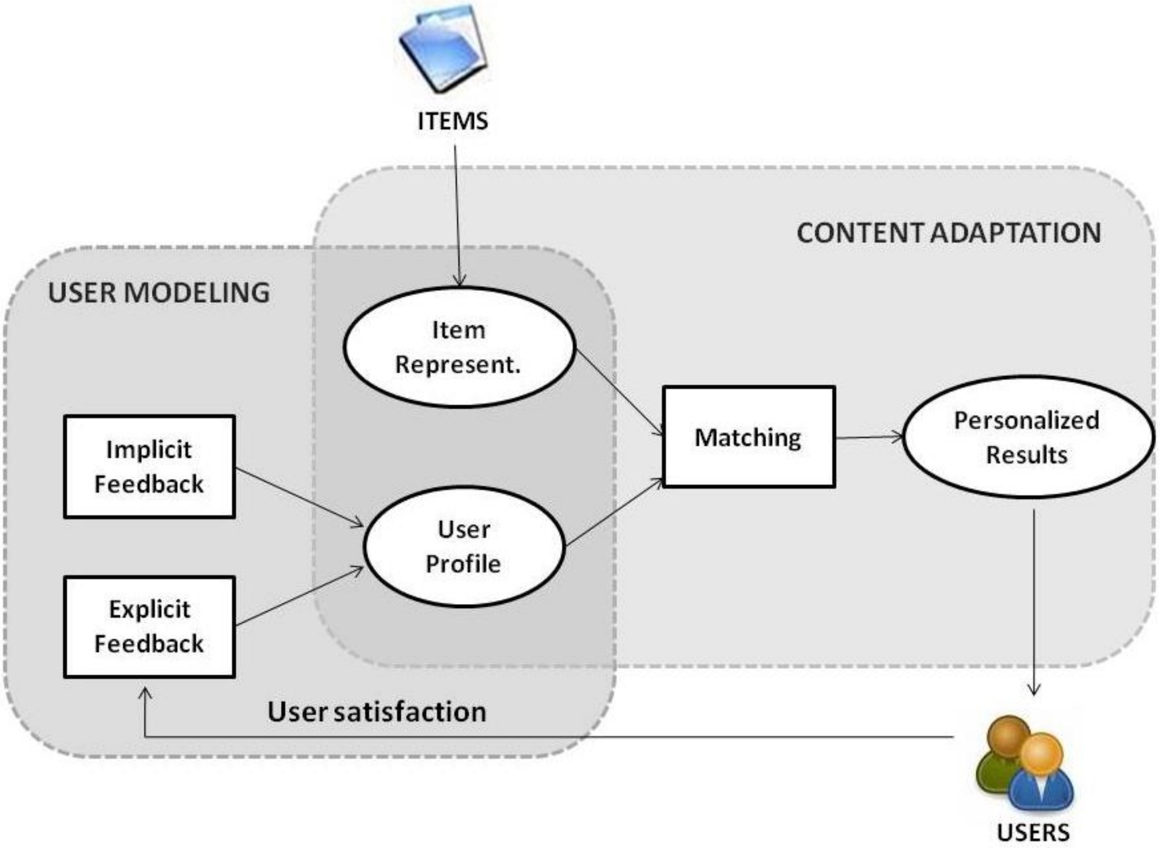


**Figure 2.1** General personalization process

## 2.2. User modeling and ontologies

One distinctive feature of a personalized recommendation system is a *user model (*also called *user profile* in the context of recommenders*)*. The user profile is a representation of information about an individual user that is essential for an adaptive system to provide *the personalization effect*, that is, to behave differently for different users.

The user modeling process is concerned with several issues which designers of adaptive systems have to deal. In the following sections, different methods dealing with these issues are presented: Which user features to model (section 2.2.1)? How to collect information about the user (section 2.2.2)? How to represent this information (section 2.2.3)? How to construct/learn the user profile (section 2.2.4)? How to adapt the user profile to changes over time (section 2.2.5)?

This chapter discusses user profiles specifically designed for providing personalized information access in Web-based systems, since this is the application the recommendation system of this work has been developed for. Moreover, as the recommendation system developed use semantic technologies to model the user profile, analysis will focus on the ontology-based user modeling approach.

### *2.2.1. What is being modeled?*

The kind of information that is being modeled in personalized Web systems mainly depends on the application domain and the kind of personalized services. In general, most adaptive Web systems represent features of the user as an individual; although mobile and ubiquitous adaptive Web systems, where the context is essential, also represent the current user's context. Typical context features are: the user location, the user platform, the physical environment, the social context, and effective state.

This section focuses on the main features describing the user as an individual, since how to deal with groups of users is outside of the scope of this work. The five most popular features are [9]:

- the *user's knowledge,* which represents the expertise level of the user in a specific subject or domain. This feature appears to be the most important user feature for existing adaptive educational and hypermedia systems, in which the knowledge is frequently the only user feature being modeled;
- the *user's interests,* which always constituted the most important (and typically the only) part of the user profile in adaptive information retrieval and filtering systems that dealt with large volumes of information. Normally, it represents the long-term users' interests and preferences in a specific domain. Due to the characteristics of the recommendation system developed, in this work, this is the only user feature modeled (see section 3.2.1);
- *the user's goal* or *need,* which represents the immediate purpose for a user's task within an adaptive system. Depending on the kind of system, it can be an immediate information need (in information access systems), or a learning goal (in educational systems). The user's goal is the most changeable user feature: it almost always changes from session to session;
- the *user's background*, which represents the user's previous experience outside the core domain of a specific Web system. A range of backgrounds that have been used in adaptive Web systems includes the user's profession, job responsibilities, experience of work in related areas, and even specific view on the domain. Background information is used most frequently for content adaptation, although there are examples of the use of it within adaptive search and adaptive navigation support;

- and the *user's individual trait,* which define the user as an individual. Examples are personality traits (e.g., introvert/extravert), cognitive styles, cognitive factors (e.g., working memory capacity) and learning styles. Similar to user background, individual traits are stable features of a user that either cannot be changed at all, or can be changed only over a long period of time.

### *2.2.2. User profile acquisition*

Once we know the user features we need to model for providing personalized recommendations in a specific domain or domains, the next issue is how to collect information about individual users. A basic requirement is that the system must be able to uniquely identify users.

The information collected may be explicitly input by the user or implicitly gathered by a software agent. It may be collected on the user's client machine or gathered by the application server itself. Depending on how the information is collected, different data about the users may be extracted. In this section explicit and implicit feedback methods are briefly discussed.

#### *2.2.2.1. Explicit Feedback*

Explicit user-information collection-methodologies rely on personal-information input by the users. The two typical methods to capture explicit feedback are: via *Web forms* such as MyYahoo![3], in which the users can provide personal and demographic information such as birthday, current job, personal interests or personal data (e.g., stock portfolios); and via *ratings* such as MovieLens[4] or Netflix[5], which allows users to express their opinions by selecting a value from a range.

One problem with explicit feedback is that it cost time and, if users do not voluntarily provide personal information, it is not possible to build any profile for them. In addition, though the users provide some feedback, this could be inconsistent or not properly updated causing the profile to become increasingly inaccurate over time.

[3] See Yahoo personalized portal [http://my.yahoo.com/] (last access on August 19, 2009).
[4] See [http://www.movielens.org] (last access on September 3, 2009).
[5] See Netflix website [http://www.netflix.com/] (last access on August 19, 2009).

#### *2.2.2.2. Implicit Feedback*

Implicit user-information collection-methodologies are based on usage-data of the users. From this data the system tries to predict user interests taking into account implicit indicators associated to specific *patterns of user behavior* [10]. Web usage mining is the process of automatic discovery and analysis of patterns and associated data collected from the user interactions with Web resources. The typical heuristic indicators used by implicit user modeling methods are the time spent “viewing” a specific item, the frequency of item selection, and if the item is consumed or acquired.

The main advantage of this technique is that it does not require any additional intervention by the user during the user modeling process. One drawback of implicit feedback techniques is that they can typically only capture positive feedback. When a user clicks on an item, it seems reasonable to assume that this indicates some degree of interest in the item. However, it is not as clear, when a user fails to examine some data item, that this is an indication of disinterest.

#### *2.2.2.3. Stereotypes approaches*

The acquisition of user profiles in a stereotype approach is based on generalizations about communities of users [11]. A stereotype contains the typical characteristics of a group of users in a particular application domain along with a set of activation conditions, which make it possible to identify users belonging to this group.

The application of stereotypes for user profile acquisition has been shown to be useful in areas where a fast, but not necessarily precise, assessment of user interests is required. In such situations, stereotypes are a basic information source that is used initialize a default profile about the user when nothing else is available [12].

An obvious disadvantage of this approach is the necessity for a pre-definition of stereotypes, whose construction is almost exclusively manual as this is a process that involves the classification of users by an expert and the analysis of individual interests of users. A detailed survey of toolkits for deploying stereotypes can be found in [13].

### *2.2.3. User profile representation*

As unstructured Web documents are generally not suitable as inputs for machine learning algorithms, preprocessing steps are needed to transform text into more treatable representations. Traditional user profile representations are those using sets of weighted keywords or semantic networks. A more recent approach, in which this work is more focused on, is the use of weighted concepts.

### *2.2.3.1. Keyword- and semantic network-based profiles*

The most common representation for user profiles in personalized Web systems is sets of keywords, which can be automatically extracted from Web documents or directly provided by the user. Each keyword can represent a topic of interest or also can be grouped in categories to reflect a more standard representation of user's interests. Usually keywords are associated with weights that are quantifiers indicating the degree of interest or disinterest in a specific topic, such as in Fab [14], a Web page recommender.

Another approach quite similar to *keyword-based profiles*, which tries to address the polysemy problem inherent in this kind of representations, is the *weighted semantic network* in which each node represents a concept and particular words with the same meaning are connected by means of arcs. The ifWeb recommender [15] uses this approach.

The main drawback of these user profile representations is that they require a large amount of user's feedback in order to learn the terminology by which a topic may be discussed in future Web documents. Concept profiles, explained in the next section, is an approach that overcomes this limitation.

### *2.2.3.2. Concept-based profiles*

Concept-based profiles are trained on examples for each concept a priori, and thus begin with an existing mapping between vocabulary and concepts. These profiles are robust to variations in terminology and need less user feedback than the above approaches.

Concept-based profiles are similar to semantic network-based profiles in the sense that both are represented by conceptual nodes and relationships between those nodes. However, in concept-based profiles, the nodes represent abstract topics the user considers interesting, rather than specific words or sets of related words.

Although concept-based profiles can be modeled using vector models [16] (set of unrelated concepts), the most common approach is to use connected models such as the taxonomy and the ontology models. The *ontology model* is based on a rich ontology[6] in which concepts are explicitly specified and the resulting profile may include a variety of relationships types, allowing better interest tracking and propagation. This kind of model is

[6] In this thesis, we adopt the following definition for ontology: A formal and explicit specification of a shared conceptualization, which is readable by a computer [58].

preferred by closed corpus personalization systems, in which documents and their relationships are known to the system at design time such as tourist guides [17] or store catalogues [18]. The *taxonomy model* can be seen as a simple case of an ontology model in which concepts are modeled hierarchically with only parent-child relationships [19].

### *2.2.4. User profile construction*

A variety of construction techniques based on machine learning and statistical methods used in Web personalization systems to construct the user profile. Different techniques can be more appropriate depending on two main factors: the type and availability of information source (explicit or implicit), and the user profile representation used.

From the information source viewpoint, Web systems that have in place a large volume of usage data normally use Web usage mining techniques [5], which consist of off-line learning methods based on past user interactions to construct the user profile. As one of the objectives of this work is that the system does not depend of the availability of historic usage data to construct accurate user profiles, this kind of techniques are not presented in detail. In contrast, in this section are shown different typically online construction techniques more suitable to each of the user profile representations seen in the previous section.

#### *2.2.4.1. Building profiles based on keywords and semantic networks*

Keyword-based profiles are normally created by extracting keywords from Web documents collected from some information source, and then, using some form of keyword weighting to identify the most important keywords. The typical techniques used for these tasks are the prototype-based classifiers or *tf-idf classifiers* widely used in information retrieval [20], which represent user's interests in terms of a prototype vector in the same dimensional space as Web documents, facilitating the similarity calculation between the user profile and documents.

Semantic network-based profiles are typically built by collecting explicit positive and/or negative feedback from users, extracting keywords from the user-rated Web documents, similar to keyword profile construction techniques. However, these techniques differ because, rather than adding the extracted keywords to a vector, the keywords are added to a network of nodes representing group of words or concepts, what allows the system to deal more effectively with the inherent ambiguity and synonymy of natural language.

Normally these techniques exploit an existing mapping between words and concepts like the one of WordNet[7] such as the SiteIF project [21].

#### *2.2.4.2. Building profiles based on concepts*

The general technique used to construct concept-based profiles is the overlay approach, in which user features are represented as an overlay of a concept-level model of the domain that the system covers. As mentioned before, the most common used models with this approach are the ontology-based models.

Basically, the overlay approach consists of mapping collected feedback on visited Web documents to concepts of a specific domain associated with a weight, which indicates the degree of interest for each concept. Different techniques to construct weighted concept profiles are used such as: variations of a *tree coloring method* [19], which involves tagging nodes representing domain concepts in a general n-tree with information (usually a weight); *domain inferences* [22], which consist of a weighting propagation method of user's interests by applying domain inferences based on the hierarchical structure provided by the ontology or taxonomy model (also known as *upward and sideward propagation)*; and the *spreading activation algorithm* [23], which is a generalization of the previous weighting propagation technique in that the propagation is based on pre-computed weights of the concept relationships, not necessarily based on the hierarchical structure of the ontology.

Although some systems collect feedback on pre-classified documents, many collect feedback on a wide variety of documents what implies they rely on text classification in order to map the information collected about the user into the appropriate concept(s) in the concept model. Text classification is a supervised approach that attempts to assign documents to the best matching concept(s) from a predefined set of concepts. A very complete survey and comparison of such methods is presented in [24].

### *2.2.5. User profile adaptation*

Adaptation of user profiles is an essential requirement for personalized systems that need to be capable of adjusting to changes quickly in order to reflect the user's interests accurately. Profile updating can be done automatically and/or manually. Automatic methods are preferred because it is less intrusive to the end user.

[7] The *Wordnet* project at Princeton University is an online lexical reference system that organizes English words into synonym sets [59]

In most personalized systems adaptation is restricted to the incorporation of new information acquired through user feedback. The main disadvantage of this method of adaptation is that old interests are not forgotten, causing not only an exponential growth of the user profile, but also a decrease in precision since the recommendation system continue recommending information matching the old interests.

Several forgetting mechanisms have been proposed in the literature to adapt user profiles, imitating the gradual process of natural forgetting [25]. A simple approach is to consider a time window of fixed or adaptive size and learn the description of the user interests from only the latest observations.

## 2.3. Content adaptation

The content adaptation or recommendation process has been the main focus area of research over the past decade in recommendation systems. Different recommendation approaches have been developed using a variety of methods from such disciplines as human-computer interaction, statistics, data mining, machine learning, and information retrieval. In this section is presented a detailed review of the traditional approaches based on user and item information, and also some description of the current trend in recommenders that try to incorporate contextual information to the recommendation process.

### *2.3.1. Traditional approaches*

#### *2.3.1.1. Classification*

Tradition recommendation methods are often classified into broad categories according to the nature of their algorithmic technique as well as to their knowledge source. Based on the kind of algorithmic technique two main categories can be distinguished [26]:

- **Memory-based**. This approach memorizes all the previous historical data (such as ratings) and operates over this data to make recommendations. Therefore, these techniques are more prone to scalability issues, and generally adapt better to changes in user interests as more data becomes available.
- **Model-based**. It consists of using the available data to learn a model, which is the used for recommendations. In these approaches the computationally expensive learning phase is usually realized offline and thus they generally tend to scale better than memory-based approaches.

Four different classes of recommendation techniques can be identified on the basis of their knowledge sources [6], as can be observed in Figure 2.2:

- **Knowledge-based**. These systems make use of explicit domain knowledge about items or users to generate recommendations. This knowledge sometimes contains explicit functional knowledge about how certain item features meet user needs. Typically, these systems are quite “static” in the sense that they only learn the short-term user’s preferences associated to the current session and often employ case-based reasoning (CBR) [27] during the recommendation process such as the NewsDude system [28].
- **Content-based filtering**. It consists of recommending items matching user’s interests implicitly or explicitly collected and item’s features. The key element of this method is the similarity measure that indicates how related is some item to a certain user. Model-based content recommenders usually treat recommendation as a user-specific classification problem and learn a classifier for the user's likes and dislikes based on item features [29].
- **Collaborative filtering**. It uses data about the preferences of a set of users to recommend content to a target user with similar tastes. Typically, these approaches do not use any information regarding the actual content, but are rather based on user’s opinions (typically ratings explicitly collected). Memory-based collaborative recommenders usually employ heuristic techniques such as correlation analysis and vector similarity and can be distinguished by two different types depending on what is based the similarity: User-based, when the algorithm consist of finding similar users to the active one [30]; Item-based, when consist of finding similar items to the ones that the active user likes [31]. Model-based collaborative recommenders usually employ probabilistic classifiers such as Bayesian networks [26] as well as clustering models [32].
- **Demographic filtering**: A demographic recommender use descriptions of people to learn the relationship between a single item and the type of people who like it. Generally, these recommenders use some kind of stereotype approach to acquire the user profiles and form different groups of users. Once the user is classified into one group, the opinions of users belonging to the same group are combined for generating recommendations [33].

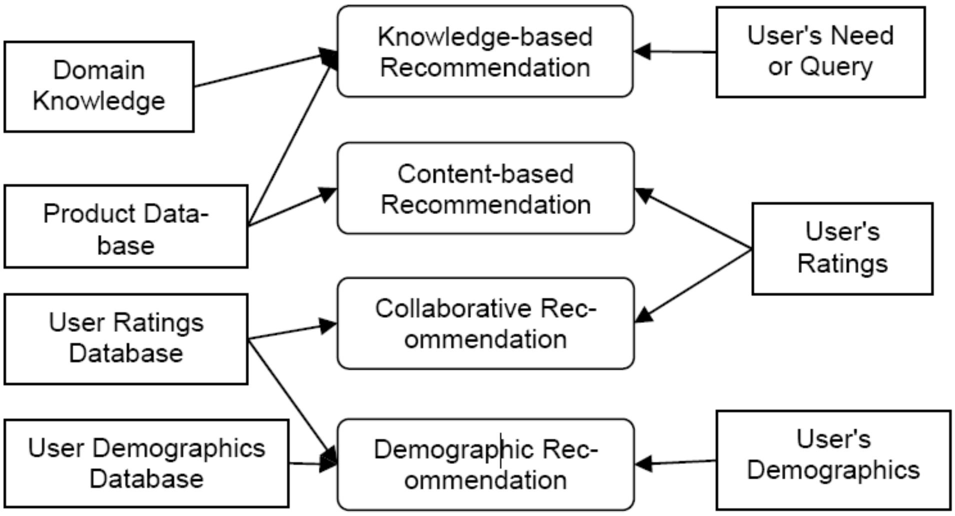

Domain Knowledge
Knowledge-based Recommendation
User's Need or Query
Product Data-base
Content-based Recommendation
User's Ratings
User Ratings Database
Collaborative Rec-ommendation
User Demographics Database
Demographic Rec-ommendation
User's Demographics

**Figure 2.2** Recommendation techniques and their knowledge sources

### *2.3.1.2. Recommendation techniques tradeoffs and hybrid approaches*

All previous recommendation techniques have been the subject of active exploration since the mid-1990's and their capabilities and limitations are well known (see Table 2.1).

| **Recommendation Tradeoffs** | | **KB** | **CB** | **CF** | **DF** |
|---|---|---|---|---|---|
| Capabilities | Can identify user groups precisely | | | X | X |
| | Domain knowledge not needed | | X | X | X |
| | Recommendation quality improver over time | | X | X | X |
| | Implicit feedback is sufficient | | X | X | |
| | No ramp-up required | X | | | |
| | Sensitive to preference changes (short-term profile) | X | | | |
| | Can map from user needs to items | X | | | |
| Limitations | New user cold-start problem | | X | X | |
| | New item cold-start problem | | | X | X |
| | Sparsity problem | | | X | X |
| | Gray-sheep or generalization problem | | | X | X |
| | Insensitive to preference changes | | | X | X |
| | Overspecialization problem | | X | | |
| | Demographic information is required | | | | X |
| | Knowledge model and domain experts are required | X | | | |
| | Do not learn user long-term preferences | X | | | |

**Table 2.1** Recommendation systems tradeoffs

All the learning-based techniques: content-based filtering (CB), collaborative filtering (CF), and demographic filtering (DF), suffer from the *cold-start* problem in one form or another. CB is not affected when a new item is introduced into the system because it uses item's features for recommending. DF is not affected when a new user registers into the system because users are associated to stereotypes. CF is affected in both cases because the recommendation is only based on ratings of users (user and item descriptions are not employed).

CF and DF approaches suffer from the sparsity and generalization problems. The former is due to the need of large quantities user's ratings to generate quality recommendations. The latter is due to these approaches are based on generalizations of user's interests and therefore the system is not able to provide accurate recommendations to users with particular interests. Furthermore, these recommenders cannot adapt the recommendations to any individual interest changes.

CB recommenders avoid the sparsity problem because their recommendations do not rely on opinions of others users. In contrast, these approaches tend to suffer from the overspecialization problem, gradually providing more specialized recommendations according to the user's interests over time, due to the syntactic nature of the existing similarity metrics.

KB recommenders avoid all the previous mentioned problems as their recommendations are independent of individual tastes and also do not depend on a base of user ratings. In contrast, these approaches do not have the capability of increasing the quality of their recommendations because they rely on immediate user's needs and therefore do not learn long-term user's interests. Moreover, the domain knowledge of these systems usually is manually maintained by domain experts and this can be very expensive depending on the domain.

Hybrid recommender systems are those that combine two or more of the techniques described above to overcome its main limitations and improve recommendation performance. Different strategies of hybrid recommendation have been used in literature and the most common ones are those that combine information across different sources. Some typical strategies are: the weighted strategy, in which the score of different recommendation components are combined numerically [34]; the feature augmentation strategy, in which the recommendation technique is used to compute a feature or set of features, which is then part of the input to another technique [35]; and the cascade strategy, in which recommendations made by one technique are refined by another technique [36].

### *2.3.2. Context-aware approaches*

Traditionally recommendation systems have been focusing on recommending the most relevant items to users based on the available information about them. While the traditional recommenders have performed reasonably well in several applications such as e-commerce, in many other applications, such as location- and time-based services, including travel recommendations, it may not be sufficient to consider only users and items, being also important to incorporate contextual information into the recommendation process.

Context, besides information on users and items, is additional information relevant to generate contextual recommendations. Contextual information can be explicitly obtained from direct inputs of the user or implicitly from the user behavior as well as by using different environment and position sensors such as the GPS; this last context acquisition method is specifically useful in mobile applications.

Context-aware recommendation systems can be classified according to how contextual information is integrated in the recommendation process, tightly integrated with the user preferences or independently used complementing the outcomes of traditional recommenders; and how is the context used from an algorithmic viewpoint. Three different strategies are identified from the algorithmic perspective [37] (see Figure 2.3):

- Contextual pre-filtering. A weak coupling context integration strategy in which contextual information is used to select the data that will be recommended by using traditional recommendation techniques [38].
- Contextual post-filtering. A weak coupling context integration strategy in which contextual information is used to adjust the resulting recommendations of traditional approaches to the user's context [39].
- Contextual modeling. A tight coupling context integration strategy in which context is used directly during the user modeling and recommendation process. These recommenders are known as multidimensional (MD) recommendation systems [40].

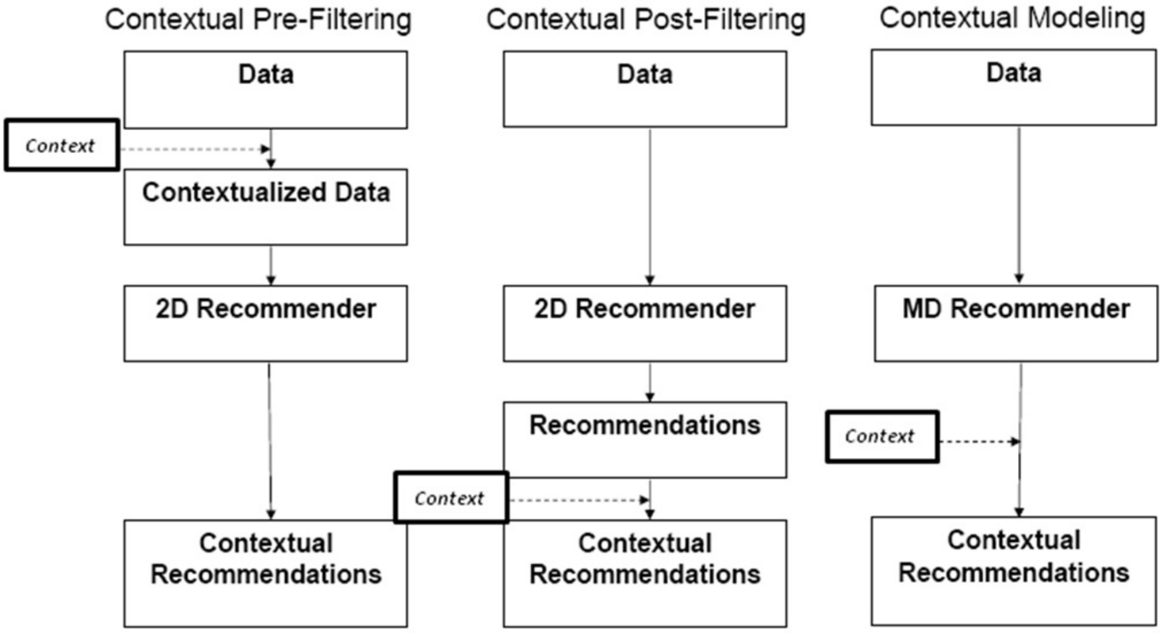


**Figure 2.3** Strategies for incorporating context in recommendation systems

## 2.4. Semantic Web technologies

The Semantic Web project [41] aims at enriching Web data, which is usually represented in (X)HTML or other XML formats, by meta-data specifying the meaning of such data and thus allowing Web based systems to take advantage of “intelligent” reasoning capabilities. In the context of personalized recommendation systems semantic Web technologies provide several advantages:

- *Better interoperability*. Semantic Web representation models provide uniform ways to describe, share and exchange knowledge about: information resources, domains they describe, users who use them, and further knowledge needed and acquired automatically in Web systems.
- *Explicit semantics*. Domain models which are used to describe and index information resources provide semantics about them which helps personalization systems to better understand how they fit to user query and user’s interests.
- *Formal representation*. Semantic Web vocabularies and ontologies provide means to formalize information resources about some specific domain knowledge. On the Web, each information resource has its own identifier provided, specified as a *Unified Resource Identifier (URI)* which is globally unique. Different formalisms have been proposed to represent information resources: on the one hand, there are basic

languages that provide a syntax for describing assertions about resources such as the Resource Description Format (RDF[8]) and Topic Maps [42]; on the other hand, there are schema or ontology languages that allows describing properties and relationships about resources in some specific domain such as the RDF Schema (RDF-S[9]) and the Web Ontology Language (OWL[10]) which is an extension of RDF-S and incorporates different levels of logics.

- *Formal reasoning.* These formal representations for knowledge enables formal reasoning top of them. Several query languages have been introduced to query for metadata providing efficient and effective access to data on the Semantic Web such as the SPARQL[11], the most recent RDF query languages. In addition to query languages, different reasoning technologies are available. The most common used reasoners are those that use Description Logics reasoning (DL) such as the OWL-DL ones: Pellet[12], Racer[13] and Fact++[14].

[8] RDF [http://www.w3.org/RDF/] is a foundation for processing meta-data; it provides interoperability between applications that exchange machine understandable information on the Internet. RDF uses XML to exchange descriptions of Internet resources but the resources being described can be of any type, including XML and non-XML resources. RDF can be used in a variety of application areas, for example: in resource discovery to provide better search engine capabilities; in cataloging for describing the content and content relationships available at a particular website or digital library; by intelligent software agents to facilitate knowledge sharing and exchange; in content rating; in describing collections of pages that represent a single logical *document*; for describing intellectual property rights of web-sites; and for expressing the privacy preferences of a user as well as the privacy policies of a website. RDF provides the means for adding semantics to a document without making any assumptions about the structure of the document. RDF is an infrastructure that enables the encoding, exchange and reuse of structured meta data.

[9] RDF-S [http://www.w3.org/TR/rdf-schema/] provides a basic type schema for RDF. *Objects*, *classes*, and *properties* can be described. Predefined properties can be used to model *instance of* and *subclass of* relationships as well as domain restrictions and range restrictions of attributes (D. Brickley and R.V. Guha: RDF Vocabulary Description Language 1.0: RDF Schema, W3C Recommendation 10 February 2004).

[10] OWL [http://www.w3.org/TR/owl-features/] is designed for use by applications that need to process the content of information instead of just presenting information to humans. OWL facilitates greater machine interpretability of Web content than that supported by XML, RDF and RDF-S by providing additional vocabulary along with a formal semantics.

[11] SPARQL [http://www.w3.org/TR/rdf-sparql-query/] is used to express queries across diverse data sources, whether the data is stored natively as RDF or viewed as RDF via middleware

[12] See [http://clarkparsia.com/pellet/] (last access on September 3, 2009)

[13] See [http://www.sts.tu-harburg.de/~r.f.moeller/racer/] (last access on September 3, 2009)

[14] See [http://owl.man.ac.uk/factplusplus/] (last access on September 3, 2009)

## 2.5. Semantic recommendation systems

In this section it is presented a survey of semantic recommenders. These systems are characterized for incorporating semantic knowledge in their recommendation processes to generate more quality recommendations than traditional recommenders by taking advantage of current Semantic Web technologies (briefly presented in previous section).

Semantic recommendation systems presented in this survey share the characteristic of using concept-based user modeling techniques based on the overlay approach (presented in section 2.2.3.2 and 2.2.4.2 respectively) to enhance the recommendation process, which usually rely on traditional recommendation techniques. Currently, two main specializations of semantic recommender can be distinguished [43]: *context-based recommenders,* which try to model accurately the user's context using concept-based models in order to adapt the recommendation to these circumstances [44]; and *trust network-based recommenders* that, in addition to take advantage of semantic modeling, offer an addition filtering level based on trust networks [45]. As the recommender developed in this work does not belong to any of the previous specializations (see chapter 3), this survey is focused on general concept-based semantic recommenders.

As can be observed in Table 2.2, approaches of different application domains exist in literature.  Although in most recommenders semantics are used to improve the similarity estimations of content-based recommendation techniques, some in the context of e-commerce have been employed to enhance collaborative filtering recommendations [16] [18] (something quite logic because e-commerce applications mostly employ CF techniques). In both approaches the user profile is modeled using OWL ontologies and exploited to find similar users (the neighborhood).

In the personalized Web search domain most of recommenders map users' interests implicitly collected to open concept hierarchies such as the Open Directory[15] available in RDF format on the Web. Particularly, in [23] a short-term user's interests are learnt using the spreading activation algorithm and then used to re-rank the search results. And in [46] a combination of keyword- and concept-based user modeling techniques is used, using an id-tdf classifier and cosine similarity measure in the vector space model.

In the domain of scientific papers two approaches based on concept taxonomy models are presented: the QuickStep system [47] and the ePaper [48]. Both use a tree coloring

[15] The **Open Directory Project** is the largest, most comprehensive human-edited directory of the Web available at [http://www.dmoz.org] (last access on September 3, 2009)

method based on a specific correlation measure to weighting the concepts, which the user is interest in, using the collected feedback. Furthermore, in the former the user profile is initialized via stereotypes and also upwards domain inferences through the hierarchy are employed. The latter approach is more focused on the recommendation process, in which a hierarchy-based semantic similarity algorithm is used, supported by the IPTC[16] news ontology, and the user profile is learnt via implicit feedback. Another approach with similar characteristics is the SemMF [49] which also takes advantage of the job domain taxonomy to compute semantic similarities for the recommendation.

Finally, there are some approaches that attempt to exploit rich ontology models taking advantage of semantic descriptions of the items, besides the hierarchical relations used in the previous ones. The Travel Support System [17] is a tourism domain recommender that uses RDF ontologies to represent all features of the user profile. The system use a stereotype approach to create the initial profile, a variation of tree coloring method using a specific correlation measure based on statistical heuristics among the domain concepts and the implicit and explicit feedback collected, and also applies upwards domain inferences through based on the hierarchical structure of the ontology. Foafing the music project [50] is a music recommender that employs the FOAF[17] vocabulary to represent user profiles mapping music-related concepts from a OWL-DL music ontology. The recommender combines the implicit feedback based on listening habits and the semantic descriptions of the music available. AVATAR [51] is a TV program recommender that takes advantage of OWL ontology in the TV domain to recommend items semantically associated with the user's preferences collected via explicit and implicit feedback. The system uses a tree coloring method that is based on a combination of different semantic associations among the domain concepts, and also employs upward domain inferences exploiting the hierarchical structure of the defined ontology classes (movie genres).

[16] See [http://www.**iptc**.org/**News**Codes/] (last access on September 3, 2009)

[17] Friend of a Friend (FOAF) vocabulary [http://xmlns.com/foaf/spec/] (last access on August 31, 2009)

| Application domain | Rec. technique | Semantic Web technologies | Concept-based User modeling techniques | | Examples |
|---|---|---|---|---|---|
| Web Search | Content-based | RDF<br>(Open Directory) | Implicit feedback (user behavior) | Taxonomy model<br>Spreading activation<br>Tree coloring method | A. Sieg et al [23] |
| | | | | Combines keyword-based & concept-based vector model | Challam et al. [46] |
| TV program | Content-based (case base) | OWL,<br>TV-Anytime standard | Ontology model<br>Explicit and Implicit feedback<br>Domain inferences (upwards)<br>Tree coloring method | | AVATAR [51] |
| e-commerce | Collaborative Filtering (User-based) | OWL | Vector Space Model<br>Explicit feedback (ratings) | | Farsani et al. 2006 [16] |
| | | | Ontology model<br>Implicit feedback<br>Tree coloring method | | P. Liu et al. [18] |
| Tourism | Content-based | RDF<br>RDQL | Ontology model<br>Explicit and Implicit feedback<br>Stereotype approach<br>Domain inferences (upwards)<br>Tree coloring method | | Travel Support System [17] |
| Music | Content-based | OWL-DL,<br>FOAF, RDF | Ontology model<br>Created by the user<br>Implicit feedback (listening habits)<br>Tree coloring method | | Foafing the music project [50] |
| Scientific papers | Content-based | OWL ,<br>IPTC NewsCodes | Taxonomy model<br>Tree coloring method | Implicit feedback | Epaper [48] |
| | | Frame-based ontology | | Explicit and Implicit feedback<br>Domain inferences<br>Stereotype approach | QuickStep system [47] |
| Jobs | Content-based | OWL, RDF | Taxonomy model<br>Explicit feedback (via web forms)<br>Created by the user | | SemMF [49] |

**Table 2.2** Analysis of semantic recommendation systems.

# CHAPTER 3 - A recommendation system for the semantic Web

## 3.1. General design

In order to accomplish the specific objectives of this work (defined in section 1.3), the general design of the recommendation system has to satisfy two main requirements. On the one hand, a semantic-recommender approach has been employed in order to overcome the most common limitations of traditional approaches: the sparsity and cold-start problems (see section 3.1.1). On the other hand, the system design is based on the Service Oriented Architecture (SOA) paradigm [52] allowing semantic applications of different domains to easily integrate the recommendation system developed into their information services (see section 3.1.2).

### *3.1.1. Semantic approach*

As it has been seen in section 2.5, recent recommender approaches have been trying to obtain more accurate personalized recommendations than traditional approaches by exploiting semantic descriptions of the application domain using semantic Web technologies, some of them with promising results. This is because in contrast to classic representations, ontology-based models present several advantages, including the followings:

- To allow inferring incomplete information by applying domain inferences what reduces the cold-start problem.
- To guarantee the inter-operability of system resources and the homogeneity of the representation of information.
- To allow for the dynamic contextualization of user preferences in specific domains.
- To improve the representation and description of different system elements.

In this work, a semantic Web recommender that exploits an ontology-based model in all the stages of the personalized recommendation process is presented: the user modeling component takes advantage of the semantic relationships of domain concepts to acquire a more accurate user profile, and the content adaptation component based on a content-based filtering approach exploits hierarchy-based semantic similarity to retrieve the most suitable items according to the user profile.

### *3.1.2. SOA-based architecture*

In order to develop a flexible enough recommender easy to integrate within different application domains, the recommendation system has been designed as a service provider following a SOA-based architecture. Thus, Web-applications just need to call to the public interface of the recommendation service to obtain personalized recommendations for their users at the desired moment. In Figure 3.1 is presented an abstract representation of the architectural design.

As the recommender use the semantic descriptions of domain concepts represented via ontologies to make recommendations, the only requirement that Web-applications must satisfy in order to be compatible with the recommender is to class their items as a set of concepts belonging to a pre-defined ontology (in OWL or RDF format). Once the ontology domain of a certain application is available for the recommendation service, the system is ready to start generating personalized recommendations to their specific users.

As a consequence of using this decoupled architecture, a design decision that has been made is to delegate any contextual-based filtering to Web-applications, following the previously commented contextual pre-filtering strategy (see section 2.3.2). Thus, each Web-application should provide a pre-filtered list of items to the recommendation service based on its particular contextual information.

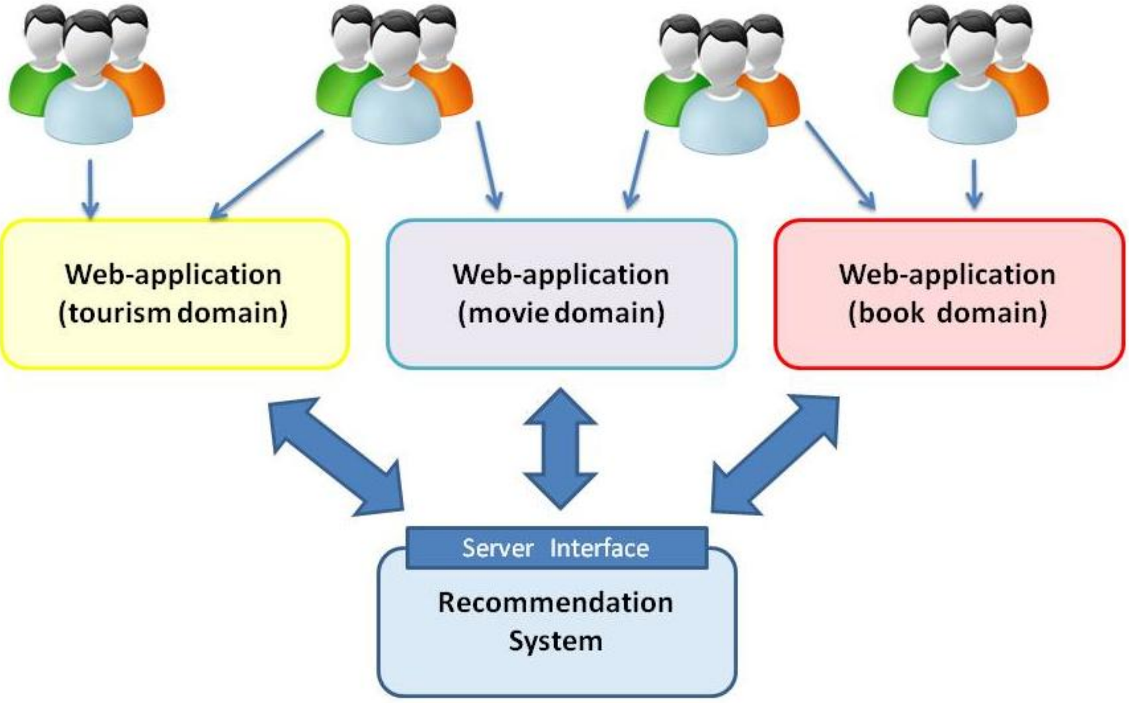

Web-application (tourism domain)
Web-application (movie domain)
Web-application (book domain)
Server Interface
Recommendation System

**Figure 3.1** Recommendation system design based on SOA architecture

## 3.2. The user modeling process

In this section, a detailed explanation of how the main issues of the user modeling process have been addressed is presented.

### *3.2.1. How is the user profile acquired?*

Because of the recommendation system need not to model contextual information, since with the above architecture design this task is done by each Web-application, the recommender only is focused on modeling long-term user's interests. For this task, a hybridization of user-information collection-techniques is used by the system in order to be able to learn the most accurate user models for each kind of application whose availability of information sources may be totally different.

#### *3.2.1.1. Collecting explicit feedback*

There are two types of explicit feedback that the recommendation system processes.

1. The information manually provided by the users when these change their degree of interest (DOI[18]) in existing concepts of the application domain, for instance using a numeric value. In general, this feedback may be acquired at the first use of the system; although also at any moment once the user is registered, if the Web application offers some management functionality for changing the user profile.
2. The information provided by means of rating specific items of the domain whether they have been previously recommended by the system or not. In this case, the degree of interest in the concepts associated with each item rated are updated properly depending on if the rating has been positive or negative. The basic assumption is that repeated negative or positive ratings of items with some particular topic can be an indicator of how much the user likes or dislikes the concept.

#### *3.2.1.2. Collecting implicit feedback*

As one of the requirement of the recommendation system is that can also construct accurate user profiles for users reluctant to provide explicit feedback, the recommender is specially thought to extract user interest patterns from the user behavior. The basic idea of the implicit collection-method is to maintain some statistics about the concepts that the users have marked interest. Depending on the type of user behavior related with the concept, higher

[18] In this thesis, DOI is used as an abbreviation of "degree of interest"

or lower is its impact on the user statistics. Currently, the recommendation system processes two types of user behaviors commonly used in Web environments:

- The user query, in which the user searches items associated with a particular domain concept. The basic assumption in which this behavior relies on is that the repeated search of a certain user for items with some specific feature can be used as a strong indicator that the user is interested in it.
- The item selection, in which the user is asking for detailed information about a specific item. In this case, the time spent by the user viewing the item information can be taken into account to give more or less importance to the particular behavior and therefore to the statistics associated. The less the time of viewing the less important the impact on the statistics of the concepts associated with the item selected.

In contrast to the explicit-feedback collection-methods, the implicit feedback collected is only used to infer positive evidences of user interest, since it has been demonstrated that this kind of feedback is not a good indicator for negative evidences [22].

### *3.2.2. How is the user profile represented?*

As in most semantic recommenders, the user profile is represented by an ontology-based model. Concretely, the ontology employed has been defined in OWL format and consists of an extension of standard vocabularies such as the FOAF and DOAC[19] ontologies because thus many concepts describing the users can be reused. The ontology is publicly available at [http://research.tmtfactory.com/ont/user_model.owl].

FOAF cover general descriptions of people and it has been defined as the standard for representing user profiles in most of the actual social networks with approximately 20 millions of FOAF profiles counted. Reusing this vocabulary allows the recommender to exploit the existing user profiles to extract the required user information. In addition, these vocabulary supports the OpenID authentication method (for more information see section 4.1.2) allowing a transparent registration and authentication process to the user.

However, still some pieces are missing in order to obtain complete personal information related with the user education and professional career that could be useful for the recommendation system in order to apply the stereotype approach. For this reason, the

[19] Description Of A Career (DOAC) vocabulary: [http://ramonantonio.net/doac/] (last access on August 31, 2009)

DOAC ontology also has been reused because is a RDF vocabulary compatible with FOAF specification that describes professional and educational user information.

### *3.2.2.1. User model ontology*

The main concepts and relationships of the user model are shown in Figure 3.2. User information can be divided in domain-dependent, such us user's interests, stereotypes, ratings, statistics and session information; and domain-independent, such as user personal and demographic data extracted from the FOAF- and DOAC-based profiles.

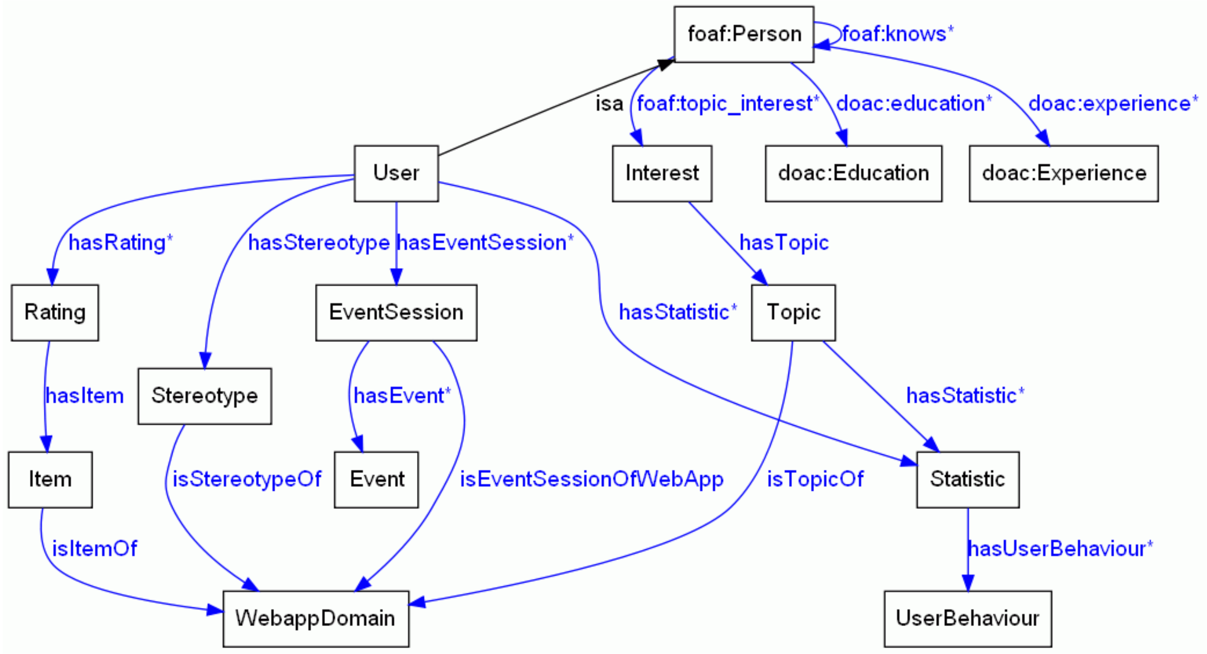


**Figure 3.2** Main concepts and relationships of the user model

The *User* concept is which represents a certain user of the system and is related with the information collected by the recommendation system about him/her. The domain-independent information is represented by the properties inherited from the *foaf:Person* concept. And for the particular case of the interests' representation, the FOAF property *topic_interest* has been reused to associate the interests learned with a certain user.

Domain-dependent information is always related with an instance of the *WebappDomain* concept. These specific concepts are: the *Topic*, which represents the concepts used to classify the items of a particular domain; the *Item*, representing items of a specific domain; the *Stereotype*, whose instances represent the matching stereotypes of the user for a specific domain; and the *EventSession* that represent the user sessions composed of a set of user events associated with a specific application domain (the usage data).

A particular user is related with as many instances of *Interest* as instances of *Topic* the user is supposed to be interested. Each *Interest'*s instance contains information about the DOI and the information sources that have been used for the prediction (see section 3.2.3.1 for a detailed explanation).

Users and topics are related with statistics about them. Basically, an instance of *Statistic* consists of a counter of the occurrences for a particular topic, user and behavior. These statistics are used for calculating the DOI according to the implicit feedback collected.

### *3.2.3. How is the user profile constructed and adapted?*

As most semantic recommendation systems that use ontology-based profiles to model the user profile, the recommender employ the overlay approach to exploit the semantic knowledge of the ontologies mapping user's interests to specific topics of the application domains. The user profile construction-techniques employed to construct the user profile are the following:

- a variation of tree-coloring method, in which each user's interest is weighted with a real value with range [-1, 1] indicating the DOI for a particular topic;
- a weighting propagation method based on domain inferences, in which the DOI for a specific topic is propagated to the parent and sibling topics exploiting the hierarchical structure of the ontology domain.

#### *3.2.3.1. User's interests modeling*

The DOI prediction for a particular topic (DOI_weight) is calculated by means of combining a fixed set of weights consisting of real values that are obtained from different information sources and learning approaches. Moreover, for each type of information source is calculated a confidence level with range [0, 1], which is an indicator of the reliability of the particular source. From the combination of the partial confidence levels is obtained the global confidence level for a specific DOI prediction (DOI_CL). Confidence levels are associated with the following abstracted values:

- LOW = [0, 0.4];
- MEDIUM = (0.4, 0.7];
- HIGH = (0.7, 1.0].

In conclusion, each user's interest is composed of the global DOI_weight and DOI_CL, as well as set of partial weights with their confidence levels according to the information available about the user. Next, it is explained how each partial weight and its confidence level is calculated and updated over time.

- Feedback manually provided by the user.
  - The weight (**ew**) is set when the user manually assigns a DOI for a particular topic through the Web application. The range of possible values is between [-1, 1]; where -1 indicates the user does not like at all items related with the topic, and 1 that is very interested in items related with the topic.
  - The confidence level (**ewCL**) is a global indicator (topic independent) set to 1 each time the user directly updates the user profile, since the system assumes that the user always is providing trustworthy information. Then, a forgetting factor is periodically applied to reduce progressively the confidence level of old interests.
- Ratings-based information.
  - The weight (**rw)** is calculated using the average of past ratings of the items related with the topic. The range of possible values also is between [-1, 1]; and the meaning is the same than in the previous case.
  - The confidence level (**rwCL**) is a measure that is calculated for each user's interest based on the number of ratings the user has associated with a given topic.
- Usage-data-based information.
  - The weight (**iw**) is calculated as the probability that the user is interested in the topic based on a weighted sum of the number of its occurrences according to the user's statistics in relation to the occurrences distribution for all users (also called normalized probability). This probability is calculated using a sigmoid function, so if the number of occurrences is greater/lower than the standard deviation of the distribution, then the value is near to 1 or 0 respectively. Depending on the type of user behavior (query or item selection) a different weight is given to the specific statistic. When the number of users and events in the system is lower than a threshold, the normalized probability is calculated using the number of occurrences distribution of each particular user. The range of values is [0, 1]; where 0 indicates non-interest, and 1 that the user is completely interested (as it was mentioned in section 3.2.1.3, this type of feedback does not provide reliable negative evidences of user interest).

    - The confidence level (**iwCL**) is a topic dependant measure that relies on the well known statistical method Univariate Significance Analysis [53], which is based on the idea that attribute values in random samples are normally distributed. Thus, if the weighted sum of number of occurrences of a certain topic is higher or lower than in a random sample according to some thresholds, then the weight is considered statistical significant and therefore the confidence level is set to 1. In contrast, if the number of occurrences cannot be considered statistical significant, then the confidence level is calculated as the distance between the sample and the center of the occurrences normal distribution.
- Stereotypical-based information.
    - The weight (**sw**) is set using the predictions on interest defined in the stereotypes in which the user better fits. As the stereotype approach is based on generalizations about the users and it cannot be considered as a trustworthy prediction, only positive evidences of user's interest should be taken into account. The range of possible values is between [0, 1].
    - The confidence level (**swCL**) is a global indicator based on the matching measure between the user and the stereotype. The higher is the matching stereotype-user, the higher is the confidence level for the stereotypical-based weights.
- Domain-inference-based information.
    - The weight (**dw**) is updated when the weighting propagation algorithm based on domain inferences is applied (for upward or sideward propagation) and the value is calculated as the average DOI_weight for the direct sub-topics. The range of values is between [-1, 1] because the weight is calculated using the average DOI_weight that has also this range of values.
    - The confidence level (**dwCL**) is a topic-dependant measure calculated as a combination of two factors: the number of direct sub-topics the user is interested in with respect to the total number of sub-topics, and the average DOI_CL of the sub-topics the user is interested in. The higher the proportion and the average DOI_CL, the higher the confidence level of the prediction based on domain inferences.

### *3.2.3.2. Initial user profile generation*

When a new user is registered into the recommendation system there are two possibilities to initialize its user profile. The more simple and reliable one is that the user provides explicitly their DOI in some concepts of the domain. Each Web-application can obtain this explicit feedback differently depending on how are the hierarchies of the domain concepts. At the end, the user is registered into the recommendation system with an initial set of user's interests with their respective **ew** values and the **ewCL** is set to 1.

The second option and the less intrusive by the user is to employ the stereotype approach that exploits some specific user-demographic data contained into the FOAF profile to classify the users into the best fitting stereotypes of a certain domain. Each stereotype has a set of predictions on interest in some particular topics. The basic idea of the stereotype approach is to complete the unknown information that the user has not wanted to provide explicitly during the registration process (filling blanches).

Each stereotype profile is described by a set of characteristics and probabilities (see the tourism stereotypes as example in section 4.2.3), and has a predefined list of topics of interests, indicating for each one the DOI predicted. The range of values is discrete using this abstraction (HIGH = 1; MEDIUM = 0,5; NULL = 0). The algorithm to initialize the user profile using the stereotype approach works as follows:

```
Input:
    • userData (personal user information extracted from FOAF profile such as
      age, gender, education level, profession)
    • stereotypeSet (set of possible stereotypes for the particular domain
      defined in the application domain ontology)
Output:
    • interestsPredicted (set of interests predicted with sw and swCL values)

Local variables:
    • DOM($S_i$) (normalized degree of match between the user and the stereotype)

    1. To determine the DOM($S_i$) according to the stereotype characteristics and
       userData:
       FOR EACH stereotype of stereotypeSet: $S_i$ DO
              DOM($S_i$) = Product of feature probabilities based on userData
       END FOR

    2. To calculate sw and swCL values of the interestsPredicted by combining the
       DOI predicted for each topic according to the DOM($S_i$) of each stereotype
```

### *3.2.3.3. User profile learning algorithm*

From the time a user profile is initialized for a specific application domain, the recommender updates the DOI_weight and DOI_CL of the user's interests over time. These DOI values are calculated using a linear combination of the available partial weights (defined in section 3.2.3.1) based on their confidence levels and some priority rules according to type of feedback. The priority rules about the partial weights have been defined based on common sense and it might be refined according to further experimentation:

1. The weights considered less trustworthy are the **sw** and **dw**, since both predictions rely on generalizations: one about the users and the other about the domain;
2. Due to the first rule, when the **ew** is available for a given topic, the **sw** is not taken into account at all.
3. When the **ew** exists and its confidence level is HIGH, then the **rw** is not used, since the **ew** is considered the most trustworthy explicit-feedback source.
4. In the case of implicit-feedback sources, if the **iw** exists and its confidence level is HIGH, then the **dw** is not taken into account.

Each time the user exits the Web-application and the user session is closed, the learning algorithm is executed in order to update the DOI values of the new possible user's interests from the user events associated with the last session. Once the partial weights have been updated, the algorithm calculates the new DOI_weight and DOI_CL for each modified interests by means of the linear combination of partial weights and the weighting propagation method based on domain inferences. The algorithm works as follows:

```
Input:
    • ew, rw, iw, sw, dw weights of an user's interest;
    • ewCL, rwCL, iwCL, swCL, dwCL confidence levels of for the particular user
      and interest
    • propagation (Boolean enabling/disabling weighting propagation)
Output:
    • DOI_weight (updated weight indicating the DOI for a particular topic)
    • DOI_CL (confidence level for the calculated DOI)
Local variables:
    • explicitW, explicitCL (represent the weight and confidence level predicted
      from the 'explicit' feedback sources: ew, rw and sw)
    • implicitW, implicitCL (represent the weight and confidence level predicted
      from the 'implicit' feedback sources: iw and dw)
Constants:
    • SIDEWARD_INFERENCE_THRESHOLD = 0.75; UPWARD_INFERENCE_THRESHOLD = 0.6
```

**Phase 1**: Linear combination of partial weights based on their confidence levels to update the DOI values of a given user's interest

```
1. To calculate explicitW and explicitCL:
   IF exist(ew) THEN
           IF is_HIGH(ewCL) OR ¬(exists(rw)) THEN
                   explicitW = ew; explicitCL = ewCL;
           IF is_MEDIUM(ewCL) OR (is_LOW(ewCL) and is_LOW(rwCL)) THEN
                   explicitW = $\frac{ew*ewCL+rw*rwCL}{ewCL+rwCL}$;
                   explicitCL = $\frac{ewCL+rwCL}{2}$;
           IF is_LOW(ewCL) AND ¬(is_LOW(rwCL)) THEN
                   explicitW = rw; explicitCL = rwCL;
   ELSE IF exist(rw) THEN
           IF is_HIGH(rwCL) OR ¬(exists(sw)) THEN
                   explicitW = rw; explicitCL = rwCL;
           ELSE
                   explicitW = $\frac{rw*rwCL+sw*swCL}{rwCL+swCL}$;
                   explicitCL = $\frac{rwCL+swCL}{2}$;
   ELSE IF exist(sw) THEN
           explicitW = sw; explicitCL = swCL * 0.5; (less trustworthy weight)

2. To calculate implicitW and implicitCL:
   IF exist(iw) THEN
           IF is_HIGH(iwCL) OR ¬(exists(dw)) THEN
                   implicitW = iw; implicitCL = iwCL;
           ELSE
                   implicitW = $\frac{iw*iwCL+dw*dwCL}{iwCL+dwCL}$;
                   implicitCL = $\frac{iwCL+dwCL}{2}$;
   ELSE IF exist(dw) THEN
           implicitW = dw; implicitCL = dwCL*0.5; (less trustworthy weight)

3. To calculate DOI_weight and DOI_CL:
   IF is_HIGH(explicitCL) AND is_LOW(implicitCL) THEN
           DOI_weight = explicitW;
           DOI_CL = explicitCL;
   ELSE IF is_LOW(explicitCL) AND is_HIGH(implicitCL) THEN
           DOI_weight = implicitW;
           DOI_CL = implicitCL;
   ELSE
           DOI_weight = $\frac{explicitW*explicitCL + implicitW*implicitCL}{explicitCL + implicitCL}$;
           DOI_CL = $\frac{explicitCL + implicitCL}{2}$;
4. IF has_changed(DOI_weight) AND is_TRUE(propagation) THEN
           Execute Phase 2;
```

```
Phase 2: Domain-inference weighting-propagation algorithm

  IF exist(parent’s interest topic) THEN
    1. To calculate proportion of topic siblings with DOI value:
         $proportion = \frac{Number\ of\ \ topic's\ siblings\ that\ belong\ to\ the\ user's\ interests}{Total\ number\ of\ topic's\ siblings}$;

    2. IF proportion > UPWARD_INFERENCE_THRESHOLD THEN
         To do upward propagation (the parent’s interest topic):
         •  $dw_{\text{parent's topic}} = \frac{Summatory\ of\ sub-topics\ DOI}{Total\ number\ of\ subtopics\ belonging\ to\ the\ user's\ interests}$;

         •  $infDOI\_CL_{\text{parent's topic}} = \frac{Summatory\ of\ sub-topics\ DOI\_CL}{Total\ number\ of\ subtopics\ of\ interest}$

         •  $dwCL_{\text{parent's topic}} = \frac{goodness(proportion)+infDOI\_CL}{2}$;

         •  Set to parent’s interest topic the dw, dwCL and infDOI_CL
            values;
         •  Execute Phase 1 for the parent’s topic interest;

    3. IF proportion > SIDEWARD_INFERENCE_THRESHOLD AND ¬is_ROOT(parent) THEN
         To do sideward propagation (The same that for the upward
         propagation but in this case is executed for all the siblings of
         the interest topic and using the SIDEWARD_INFERENCE_THRESHOLD)
   ELSE
     Case in which we are in the root concept of the feature type (do nothing)
```

## 3.3. The content adaptation process

As in most semantic recommendation systems presented in section 2.5, the basis of the recommendation method developed in this work consist of exploiting the hierarchical information contained in the ontology models to enhance the traditional content-based filtering. Therefore, the recommendation algorithm filters and ranks the items by measuring the similarity between the user’s interests and the topics that represent the items.

### *3.3.1. Items’ representation design*

In order to exploit the semantic similarities between items and users’ interests, the items have to be classified as a set of topics that represents its particular features. Each type of feature is composed of a hierarchy of topics that belong to it, and this is basically the semantic information that the recommender exploits.

About the item’s representation, it is assumed that the topics representing an item always are leaves of the feature hierarchy, that is, they are always specific topics and it is assumed that general topics cannot represent an item feature. This decision was taken in order to simplify the hierarchy-based similarity algorithm, since in this manner it is not possible that a given user's interest is more specific than the topic of the item.

As a consequence of the decoupled architectural design of the recommendation system to allow that diverse application domains can use it at the same time, the task of classify the items with the particular topics of the domain is delegated to each Web-application that will provide the correspondent pre-classified list of items in every recommendation request.

### *3.3.2. The enhanced content-based filtering method*

In this section the content-based algorithm that exploits the hierarchical classification of domain topics is presented. First, the hierarchy-based similarity measure is described, then the algorithm for calculating the score of an item for a particular user, and finally the complete recommendation algorithm that generates the diversified top-n recommendations based on the similarity and scores measures presented in the following sections.

#### *3.3.2.1. Measuring the item-user similarity*

The similarity measure used to see how a particular item matches with the user's interests is based on the hierarchical classification of the topics. Basically, the method consists of calculating for each item's topic (belonging to a feature hierarchy) a similarity value according to the type of matching with the set of user's interests. This value is calculated taking into account the depth of the topic within the hierarchy.

As it was explained in section 3.3.1, it is assumed that the topics representing the item's features are always at the lowermost level in the hierarchy, therefore user's interests cannot be more specific than the item's ones. With this assumption, three different types of matching can be produced between an item's topic and a certain user profile.

1. The item's topic appears in the user profile. In this case, the matching is perfect and the value is set to 1.
2. Some ancestor of the item's topic appears in the user profile. In this case, the matching is partial and the similarity value is calculated according to two different factors: on the one hand, the distance between the item's topic and its ancestor; and on the other hand, the depth of the item's topic in the hierarchy to which belongs.
3. Neither the item's topics nor some of its ancestors appears in the user's profile. In this case, the similarity is considered as null and the value is set to 0.

In order to calculate the similarity value when there is a partial matching (case 2) the following function is used:

- $X_{n=0} = 1$;   ($n = 0$, is the case of perfect match);
- $X_n = X_{n-1} - K * X_{n-1}$.

Where:

- 'n' is the distance between the item's topic and its ancestor (e.g., when the ancestor is the direct parent, then n = 1).
- 'K' is the decreasing factor with range [0.1, 0.5] that marks the rate at which the similarity values decrease as higher is 'n'. This factor is recalculated taking into account the depth of the item's topic in the hierarchy. The deeper is the item's topic; the lower is the decreasing factor K. This is based on the assumption that semantic differences among upper-level topics are bigger than those among lower-level topics. In other words, two general topics are less similar than two specialized ones. In Figure 3.3 and 3.4 the two different cases of decreasing functions according to the depth of the item's topic are shown.

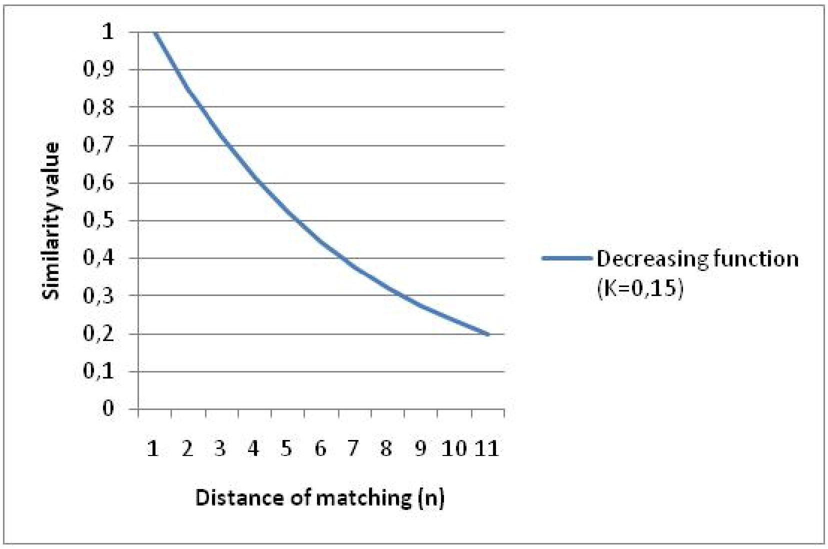


**Figure 3.3** Decreasing function in a deep case (when item's topic depth is 7)

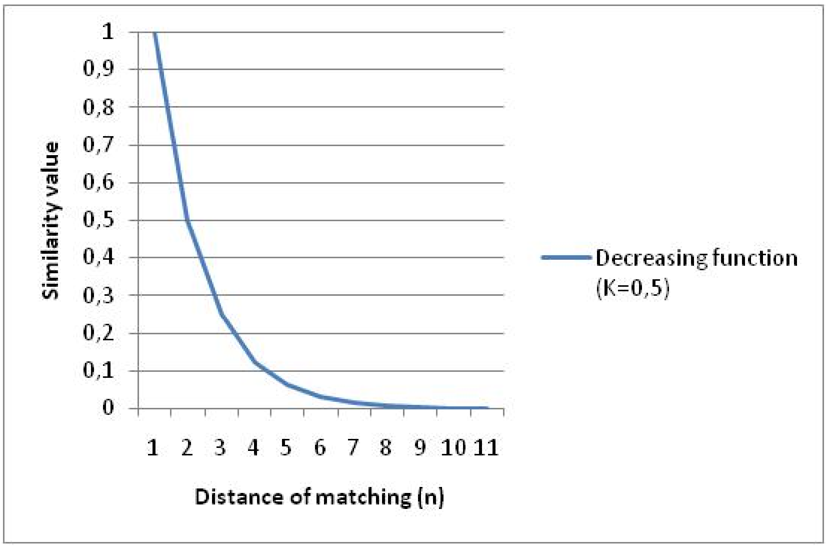


**Figure 3.4** Decreasing function in a uppermost-case (when item's topic depth is 2)

#### *3.3.2.2. Measuring the relevance of the feature's type*

Taking advantage of the hierarchical classification of the features and the interest of a given user in some topics classified into these hierarchies, a relevance value for each feature's type is calculated. The feature's relevancies are used in the item-score calculation-algorithm to give more weight to the relevant topics and less to the irrelevant ones.

The relevance of a certain feature's type for a particular user profile is obtained combining two factors:

1. A measure taking into account the number of topics appearing in the user profile (user's interests) and their confidence levels according to the characteristics of the hierarchy (size and depth). The higher the proportion of topics the user is interested in, the DOI_CL average and the size of the hierarchy, the higher the relevance.
2. A value based on the DOI_CL values (the infDOI_CL), which is inferred during the upward propagation in the domain-inference weighting-propagation algorithm (see Phase 2 of the learning algorithm) from the leaves of hierarchy to the root (a certain feature's type), indicating how extended is the interest of the user in topics of the feature's type. The higher the infDOI_CL value, the higher the feature's relevance.

As this relevance value reflects the general confidence level for the topics of a certain feature type, it indicates how much each feature influences the item-score calculation.

### *3.3.2.3. Item score calculation*

The similarity and relevance measures previously commented are used to reduce or increase the influence of a particular item's topic in the final item score. The basic information used to calculate the score of an item's topic is the DOI_weight and DOI_CL from the user's interests. Depending on the DOI_CL of each user's interest, which indicated how trustworthy the prediction is, the influence of the DOI_weight in the score of the item's topic is different. The algorithm to calculate the item score for a particular user works as follows:

```
Input:
    •   item (represents an instance Item object);
    •   user (represents the user profile)
    •   featuresRelevances (set of feature's type relevances)
Output:
    •   itemScore (represents the predicted score associated with item)
Local variables:
    •   iconcept (represents an item's topic)
    •   mconcept (represents a topic that appears in the user profile)
    •   isTrustworthy (is TRUE when the a DOI_weight of a mconcept is trustworthy
        enough and it implies stopping the search of iconcept's ancestors)
    •   parentLevel  (is TRUE when the iconcept's ancestor is the root of the
        hierarchy)
        conceptMatching (is TRUE when there iconcept or some of its ancestors
        appears in the user profile)
```

```
1. FOR EACH item's concept: iconcept DO

     a. To obtain user's DOI for the matching concepts in user profile:
        WHILE parentLevel==TRUE && isTrustworthy==FALSE DO
              IF has_matching(iconcept) THEN
              •  Set conceptMatching to TRUE
              •  Get the DOIweight of the user's interest
              •  Get the DOI_CL of the user's interest
              •  IF is_HIGH_CL(DOI_CL) THEN
                 /* The algorithm is exploring the iconcept's ancestors
                 till one of the matching concepts is trustworthy (DOI_CL
                 = HIGH) */
                    o  Set isTrustworthy to TRUE
              •  IF isTrustworthy==FALSE THEN
                    IF is_TOPLevel(iconcept) THEN
                       o  Set parentLevel=FALSE
                    ELSE
                       •  Obtain the parent concept of iconcept
        END WHILE
     b. To calculate the parcial item score (concept score):
        IF conceptMatching == TRUE THEN
              FOR EACH matching concept related with iconcept : mConcept
        DO
              •  Get the similarity score of mConcept
              •  Get the DOIweight and DOI_CL of mConcept
              END FOR
              •  conceptScore = $\frac{\sum_i^{mConcepts} DOIweight[i]*(DOI_{CL[i]}*similarity[i])}{\sum_i^{mConcepts}(DOI_{CL[i]}*similarity[i])}$;
        ELSE
              •  conceptScore = 0;
   END FOR

2. To calculate the item's core as the normalized average of concept scores
   according to their associated feature type relevance:

   itemScore = $\frac{\sum_j^{iConcepts} conceptScore[j]*featureRelevance[j]}{\sum_j^{iConcepts} featureRelevance[j]}$;
```

### *3.3.2.4. The filtering algorithm*

The recommendation algorithm can be used to obtain the score of a single item for a particular user, as well as to obtain a top-n recommendation whose items are ranked in descent order by their score value. In this case, the item's score obtained with the above algorithm is considered as the item's suitability for a certain user.

When the algorithm is used to rank a set of items, whether it consists of a proactive location-based recommendation or a user request, apart from calculating the ranked list of items, the algorithm generates a diversified recommendation list based on the user's interests in the sub-topics associated with the request. This process is also known as topic-diversification and it has been used in similar approaches [54]. The method is used to avoid the well-known overspecialization problem of content-based filtering, in which items of the recommendation are too similar. Basically, the method is useful when the user query is about a top-level or general topic, and it consists of assigning a max number of items for each sub-topic in relation to their respective DOI values in the user profile, and the size of the recommendation.

The algorithm to generate the top-n recommendation from a pre-classified set of items when the size of the recommendation is greater than 1 works as follows:

```
Input:
   • itemsList (set of items to be recommended)
   • N (size of the recommendation list)
   • query (general topic associated with the user query)
   • userInterests (represents the user profile)
Output:
   • TOPNrecommendation (resulting top-N recommendation)
Local variables:
   • rankedList (the itemsList rank in order by item's score)

   1. To calculate the score of all items using the item-score-calculation
      method:
      FOR EACH item of itemsList DO
            Get the itemScore
      END FOR
   2. Once  all item's scores are processed, items are rank in order by score
   3. IF isGeneric(query) THEN
         o  To create the diversified TOPNrecommendation list from the
            rankedList using the topic-diversification method
      ELSE
         o  To obtain the N first items of the rankedList
```

## 3.4. Recommender implementation

In this section, the main technologies and tools employed to implement all the components of the recommendation system are presented. In particular, a detailed description of the implementation using Web services, the most common strategy to implement SOA-based architectures. The exact versions and configurations of the software used are described in the Appendix A.

### *3.4.1. Technologies and tools*

In Figure 3.5 the overall vision of the technologies and tools that have been used to implement the different components of the recommendation system is shown. Basically, the recommendation service is composed of three general components: the core of the recommender (Logic Layer) implemented in Java, where the user modeling and recommendation processes are found; the Server Interface, which implements the JAX-WS Web service endpoints based on the SOAP protocol; and the Jena framework and Jastor tool, that allows the system to work with ontology models from the Java classes in a transparent way.

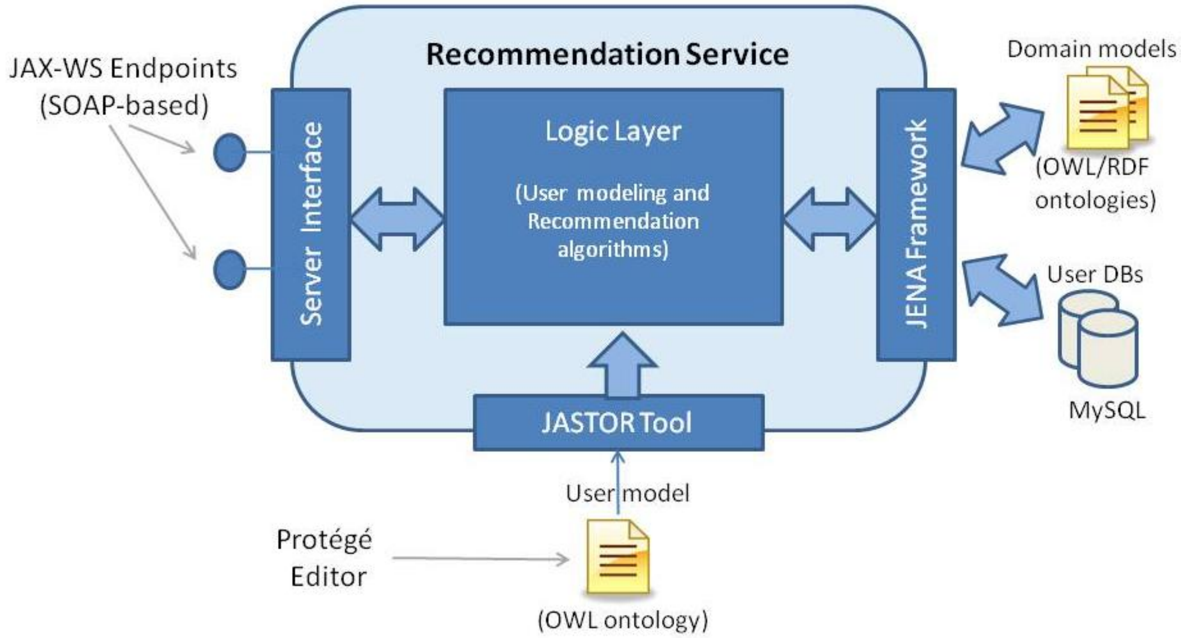


**Figure 3.5** Overall vision of the recommendation service implementation

#### *3.4.1.1. Working with semantic technologies in Java*

In order to be able to work with persistent ontology models from a program is necessary to use some framework acting as gateway between the application and the relational database. Jena[20] is the open source Java framework for building Semantic Web applications. It provides a programmatic environment for RDF, RDFS and OWL, SPARQL and includes a rule-based inference engine. In this particular implementation, two different ways of working with ontology models have been used: on the one hand, the domain

[20] See [http://jena.sourceforge.net] (last access on August 31, 2009)

ontologies such as the tourism domain are directly loaded from the ontology file as memory models, since is the most efficient way of working with non-persistent models; on the other hand, user profiles are loaded by means of persistent ontology models because in this case is needed to modify the data. For scalability reasons the user data has been distributed into different persistent ontology models; the data about interests and user events has been separated from the data about the statistics.

The Jastor[21] is an open source tool that has been used to generate automatically the java interfaces and beans from the user model ontology in OWL format, which has been developed using the well-known ontology editor Protégé[22]. Thus, it has been possible to work directly with the vocabulary and descriptions of the user model without having to create the Java classes manually.

#### *3.4.1.2. Working with SOAP Web services*

The combination of technologies that has been employed to develop the server interface is: the Java API for XML Web Services specification (JAX-WS), for the implementation of the Web service endpoints; the GlassFish v3 enterprise server, for the deployment of the Web service developed; and the Java EE version of the Eclipse IDE, for the development of the recommendation service. This combination has been chosen because the three technologies are perfectly integrated and offers several facilities for the implementation and debugging of Web services that with others technologies are not available.

The fact of building the recommendation service using Web service endpoints based on the SOAP[23] protocol, a lightweight protocol intended for exchanging structured information in a decentralized, distributed environment, allows the interoperability among systems developed in any platform, programming language and hardware. And this is a desired characteristic of a system that has to interoperate with independent Web-applications as the one developed in this work.

### *3.4.2. Recommendation service implemented as Web services*

As it has been previously introduced, the recommendation system has been implemented as a SOAP Web service that can provide a personalized recommendation service to the users of different semantic Web-applications. Web services are usually stateless

[21] See [http://jastor.sourceforge.net/] (last access on August 31, 2009)
[22] See [http://protege.stanford.edu/] (last access on August 31, 2009)
[23] Simple Object Access Protocol (SOAP) [http://www.w3.org/TR/soap/]

because of its independent and decoupled design. This implies that the service endpoint receives a request and responds back without preserving any state among different requests, and has to serve all incoming requests concurrently. But, this approach is quite inefficient when the service to implement is relatively complex and also if the service can take advantage of user sessions to be more efficient in its response. For this reason, a stateful web service approach has been used to implement the recommendation service, in which an independent instance of the service is created for each user and application domain identified.

### *3.4.2.1. The implemented stateful Web service approach*

The JAX-WS API supports the stateful Web service approach by using an implementation of the WS-Addressing protocol [24] that provides transport-neutral mechanisms to address Web services and messages. In particular, the used construct in the implemented stateful approach is the Endpoint Reference (EPR) that conveys the information needed to address a Web service endpoint.

The implemented strategy consist of having two services endpoints: a stateful one, which provides all the recommendation service functionality; and a stateless one, whose main purpose is to check the user's authentication and create a new instance of recommendation service (the Stateful one) associated with the identified user and application domain. In addition to this, it is required the compatibility with the addressing protocol of the Web service client used in the Web-application, being able to maintain the session ID among service calls that is sent in the header part of the SOAP message.

### *3.4.2.2. Public Web service operations*

In this section, the public operations of the two service endpoints are briefly described. More detailed information of the service descriptions can be found in their respective WSDL[25] files publicly available at the URL:

- [http://research.tmtfactory.com/wsdl/ RecommendationServiceLogin.wsdl]
- [http://research.tmtfactory.com/wsdl/GetRecommendation.wsdl]

The Login and Registration service (the stateless endpoint) has the following operations:

[24] WS-Addressing protocol [http://www.w3.org/2002/ws/addr/]

[25] Web Services Description Language (WSDL) [http://www.w3.org/TR/wsdl]

- *Login* (request-response-operation). This is the first operation that has to be used by the Web-app when a user is login in the system using his/her openID. It consist of checking the existence of a user with the given user id into the users database of the recommendation service, which stores all the information about the registered users. Moreover, it also checks if the application domain ID corresponds with some of the registered domain models into the system. If all the conditions are satisfied a new instance of recommendation service is created for this user and application domain.
    - Input – The user ID and the application domain ID.
    - Output – The EPR to the new instance of recommendation service.
    - Fault – The user ID or the domain ID are not valid.
- *User registration* (request-response-operation). With this operation a new user is registered into the recommendation service from the user FOAF profile that is associated with the user ID sent as a parameter. If demographic information can be extracted the user profile is initialized by using the stereotype algorithm (see section 3.2.3.1).
    - Input – The user ID and the URI of the public FOAF profile associated with it.
    - Output – A message informing that the user registration has been done correctly.
    - Fault – There have been some error during the registration.
- *Application domain registration* (request-response-operation). With this operation a new application domain is registered into the recommendation service from the domain model in RDF/OWL file format sent as a parameter.
    - Input – The ontology model and the domain ID that will be associated with the registered domain model.
    - Output – A message informing that the application domain model registration has been done correctly.
    - Fault – There have been some error during the registration.

The recommendation service (stateful endpoint): whose all operations have in common that sends a Fault message if either, the user ID or the domain ID, does not correspond with the user or application domain associated with the current service instance.

This is done to avoid "incoherent states" caused by the Web-applications if the EPRs are swap among users. The operations can be grouped by type of functionality associated.

- User modeling operations:
  - *Start new user session* (one-way operation). It is responsible of creating a new user session to which all user events will be associated. This operation is usually called after the login service when the new instance of the recommendation service is created for the given user and application domain.
  - *New query event* (one-way operation). A new event with the corresponding type of user behavior (query) is created and associated with the active user session. In addition, the event is set with the query information.
    - Input – Query information related with some domain concept.
  - *New selected item event* (one-way operation). A new event with the corresponding type of user behavior (item selection) is created and associated with the active user session. In addition, the event is set with the item information composed of its ID and features.
    - Input – Information of the Item selected by the user.
  - *New item rating event* (one-way operation). A new event with the corresponding type of user behavior (item rating) is created and associated with the active user session. In addition, the event is set with the item information composed of its ID and features as well as the user rating (an scalar value)
    - Input – Information of the Item rated and the user's rating.
  - *Close active user session* (one-way operation). It is responsible of updating the user's interests from the user events associated with the active session by executing the learning algorithm (explained in section 3.2.3.2). This operation should be called when the user logs out of the Web-application.

- User profile management operations:
  - *Manually update user's interests of domain* (request-response-operation).
  - *Get user's interests of domain* (request-response-operation).

- Content adaptation operations:
  - *Get recommendation* (request-response-operation). Filter and ranks the item's list generating a diversified top-N recommendation according to topic of the query. The operation executes the filtering algorithm (explained in 3.3.2.3).
    - ▪ Input – The size of the recommendation (N), the query associated, and the list of items, each one of which is composed of the item ID and its topics.
    - ▪ Output – The top-N recommendation, in which recommended items are ordered by their score predicted.
    - ▪ Fault – There have been some error during the recommendation.

# CHAPTER 4 - A Web application in the tourism domain

In this chapter, the design and implementation of the Web-application in the tourism domain and how the recommendation service (described in the previous chapter) has been integrated into its architecture is presented in detail.

## 4.1. Extending the INREDIS prototype

In order to be able to apply the semantic recommendation system within the INREDIS prototype, a PHP-based Web-application deployed in Apache HTTP Server[26], some architectural- and technological-based extensions have been carried out. Basically, the main extensions are found in: the content retrieval component, supporting ontology-based semantic data; the authentication method, supporting straightforward user identifications by using the OpenID authentication protocol; and the user interface, allowing the users to navigate through the hierarchical structure of the domain and make simple topic queries.

Following a modular design, the architecture of the system is presented in the Figure 4.1. In the left part, the existing services of the prototype are shown. The well-known Google Maps service is used to visualize places of interest into the city map. And the multimodal interaction service, which has been completely developed in the INREDIS project, is used to adapt the interaction mode to the user capabilities in order to ease the information access to people with special needs, such as deaf-mute or blind people.

[26] See [http://httpd.apache.org/] (last access on September 1, 2009)

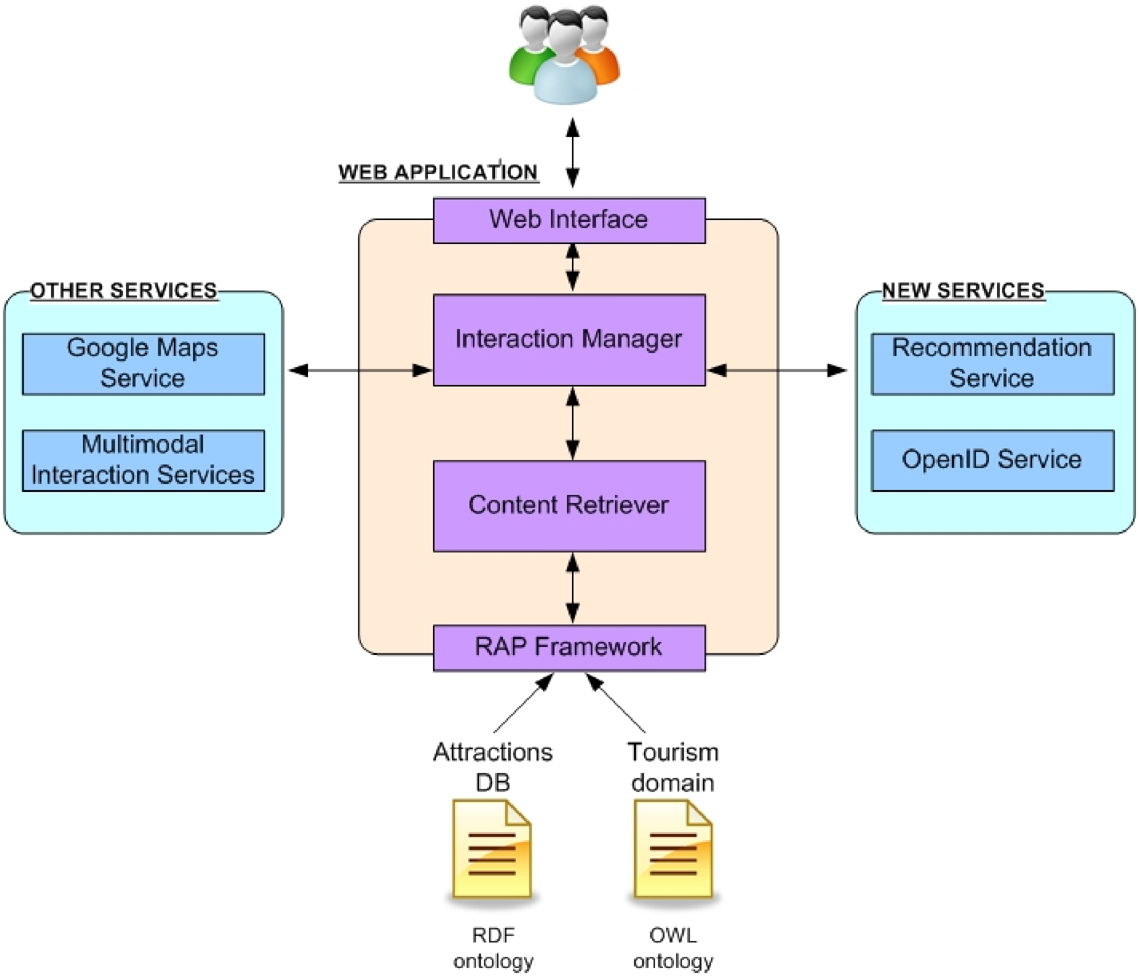


**Figure 4.1** Architecture of the tourism-domain Web-application

Apart from the recommendation service that has already been explained in detail in chapter 3 and whose integration is commented in section 4.3, the rest of the main changes and extensions are presented in the following sections.

### *4.1.1. The RDF-based semantic database*

Existing conventional database servers are not prepared to work with semantic models such as the ones defined by means of OWL and RDF ontologies. For this reason, in the last years some frameworks have appeared to offer this semantic support, acting as a gateway between semantic Web applications and relational database servers.

The framework chosen for this work is the RDF API for PHP (RAP)[27], since its design is based on the Jena framework used in the recommendation service, and in addition, it is the only one that supports ontology-based models. The reason behind this is that most of

[27] See [http://www.seasr.org/wp-content/plugins/meandre/rdfapi-php/doc/index.html] (last access on September 1, 2009)

semantic frameworks for PHP are focused on give support for the Linked Data principles [55] that still work in the low level of the Semantic Web, in which the semantics are in the RDF-based vocabularies and connections among RDF resources.

The only information source used in the prototype to obtain the information about places of interests is the data available at Barcelona Council's website, since for the evaluation purposes of the thesis there was enough information. However, for a commercial version of the system, some extra information about the places would be needed to offer a complete tourism-information-service, such us their location in GPS coordinates, images and descriptions. Taking advantage of the Linked Data principles, the information could be obtained from public RDF resources such as the ones available in DBPedia[28] (more information in section 7.2).

In the case of the Web-application developed in this work, the semantic framework RAP was used to load two different memory-based ontology models: one from the attractions DB (the instances of POI) and the other from the tourism ontology. Because the application do not require modify the models in runtime, persistent ontology models were discarded by reasons of efficiency. The process to semantify the tourism data consisted of mapping the data available at Barcelona Council's website about tourism attractions in XML format to the topics and relationships defined in the tourism ontology. This mapping was quite tedious because of such unstructured state of the original data that not allowed a straightforward mapping.

### *4.1.2. User authentication with OpenID*

Most websites ask for an extended, repetitive amount of information in order to use their application. OpenID[29] accelerates that process by allowing users to sign in to websites with a single click. Moreover, it reduces the frustration associated with maintaining multiple usernames and passwords. OpenID is a decentralized standard, meaning it is not controlled by any one website or service provider, that allows the users to use it as a portable identity across the web. In addition, this authentication method can be used in conjunction with FOAF-based user profiles, which is the standard vocabulary used in the recommendation system to model the user profiles.

[28] See [http://dbpedia.org] (last access on August 27, 2009)

[29] See [http://openid.net/] (last access on August 27, 2009)

Basically, with this authentication mechanism the following steps would be necessary for a new user to sign in to the Web-application and, at the same time, to the recommendation service.

1. The user needs to get an OpenID identity from an OpenID service provider[30], which typically is his/her home page.
2. When the user introduces the OpendID identity in the Web-application, whether it is the first use the application redirects the user to the OpenID provider's website, where is asked to submit his/her credentials and to validate the registration process into the new website. Finally, the user is sent back to the Web-application.
3. Once the user has signed in correctly, the Web-application uses the OpenID identity to obtain the FOAF profile associated with the user. At this point, the *Interaction Manager* component calls to the recommendation service passing as parameters the OpenID and the FOAF profile that will be used by the recommender to obtain the user's demographic information and to predict an initial user profile using the stereotype approach (mentioned in section 3.2.3.1).

### *4.1.3. Changes in the search interface design*

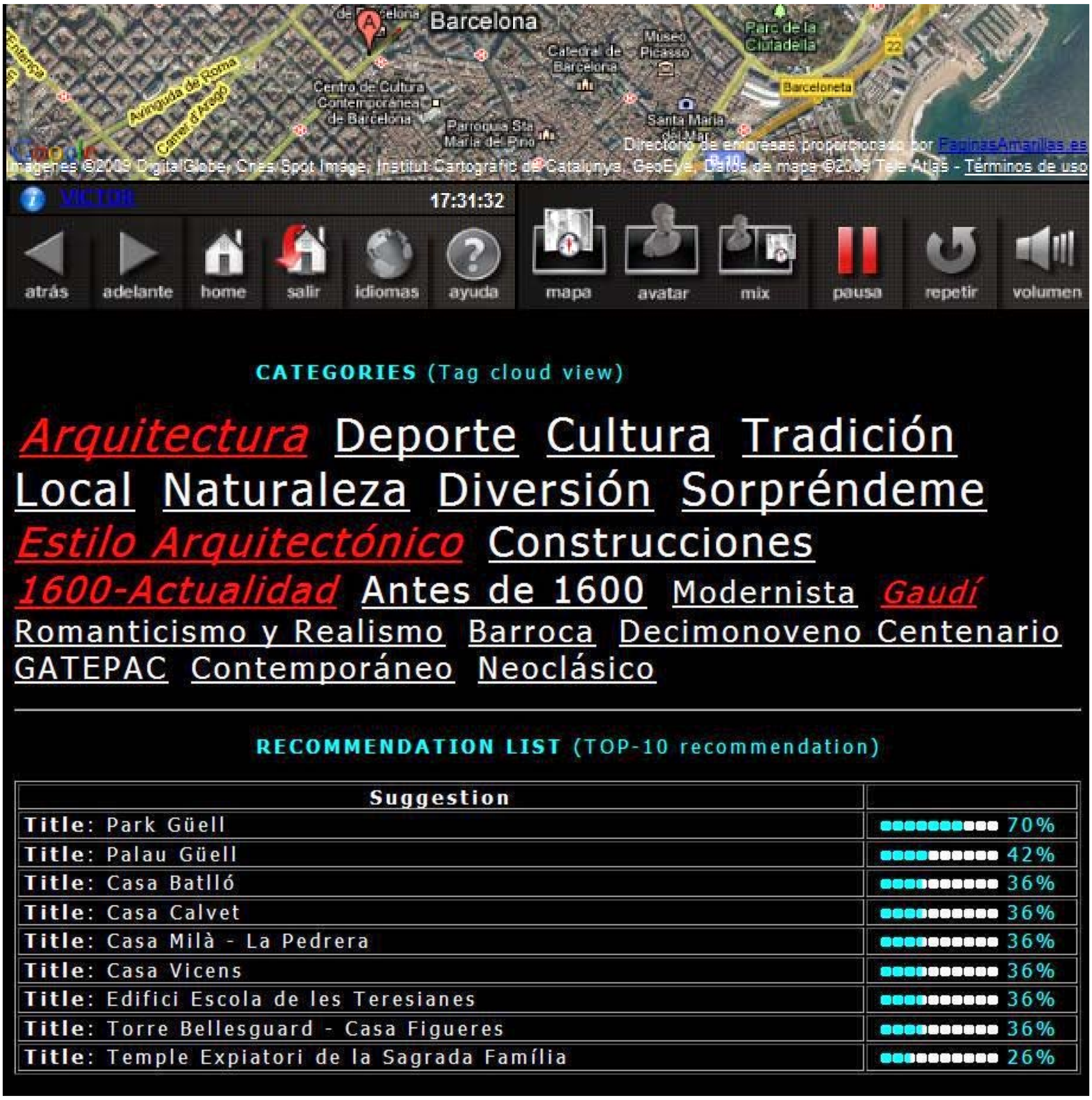
The original design of the interface was based on a simple hierarchical navigation by topic using pagination, in which each query loaded a new page. As this interface design can be tedious when the hierarchy is relatively depth, a new search-interface was implemented exploiting the hierarchies of topics defined in the tourism ontology.

The new navigation interface tries to combine the well-known tag-cloud and tree-view interface in order to exploit the strengths of both designs: on the one hand, a unique navigational panel that shows the relevant topics for the current user session; and on the other hand, a hierarchical-based navigation which allows the user to make progressively more accurate searches.

In the screenshot of the Figure 4.2 the new navigation interface and the top-10 recommendation panel presenting the result of a specific query are shown. The basic idea of the tree-view interface is to show the sub-topics associated with the topic that has been selected, and to maintain the navigation path followed by the user marking in red the clicked topics that belongs to the same branch of the hierarchy. In addition, each time the user selects

[30] A list of OpenID providers can be found in [http://openid.net/get-an-openid/] (last access on August 27, 2009)

a deeper topic their sub-topics are presented in smaller size. In the particular case of the Figure 4.2  the user has navigated through the '*Arquitectura'* branch till the '*Gaudí*' topic.


Barcelona
Parc de la Ciutadella
Museo Picasso
Catedral de Barcelona
Centro de Cultura Contemporánea de Barcelona
Parroquia Sta María del Pino
Santa María del Mar
Barceloneta
Avinguda de Roma
Carrer d'Aragó
Google
Imágenes ©2009 DigitalGlobe, Cnes/Spot Image, Institut Cartogràfic de Catalunya, GeoEye, Datos de mapa ©2009 Tele Atlas - Términos de uso
Directorio de empresas proporcionado por PaginasAmarillas.es
VICTOR
17:31:32
atrás
adelante
home
salir
idiomas
ayuda
mapa
avatar
mix
pausa
repetir
volumen

**CATEGORIES** (Tag cloud view)

*Arquitectura* Deporte Cultura Tradición Local Naturaleza Diversión Sorpréndeme *Estilo Arquitectónico* Construcciones *1600-Actualidad* Antes de 1600 Modernista *Gaudí* Romanticismo y Realismo Barroca Decimonoveno Centenario GATEPAC Contemporáneo Neoclásico

**RECOMMENDATION LIST** (TOP-10 recommendation)

| Suggestion | |
|---|---|
| **Title**: Park Güell | 70% |
| **Title**: Palau Güell | 42% |
| **Title**: Casa Batlló | 36% |
| **Title**: Casa Calvet | 36% |
| **Title**: Casa Milà - La Pedrera | 36% |
| **Title**: Casa Vicens | 36% |
| **Title**: Edifici Escola de les Teresianes | 36% |
| **Title**: Torre Bellesguard - Casa Figueres | 36% |
| **Title**: Temple Expiatori de la Sagrada Família | 26% |

**Figure 4.2** Navigation interface developed for demonstration purposes

## 4.2. Tourism domain semantification

As it has mentioned before, the Web-applications that want to use the recommendation service must define the formal semantics of its particular data by means of an OWL or RDF ontology.  In this section, it is explained how the tourism data describing places of interest and tourist stereotypes have been modeled. The complete OWL ontology is publicly available at [http://research.tmtfactory.com/ont/inredis.owl].

### 4.2.1. Conceptual modeling guidelines

In order to ease the work to the recommendation service when some new application domain is registered to the system, the ontology descriptions have to follow some simple rules so all ontologies can be uniformly interpreted. Basically, two main concepts representing the domain-dependent information have to appear in the ontology: the *Feature* concept, where are located the different feature types and their respective topic hierarchies that are used to classify the items of a particular domain; and the *Stereotype* concept, whose instances represent the possible set of user stereotypes of the domain and are associated with a set of characteristics describing the stereotype profile (instances of *StereotypeCharacteristic*) and a set of predictions on interests (instances of *InterestOpinion*) with their respective DOI, whose range of discrete values is [*High=1; Medium=0,5*; *Null=0*]. The rest of possible concepts describing the domain are not taken into account for the recommendation system.

### 4.2.2. Representing points of interest

The items of the developed tourism-domain-Web-application are represented with the *POI* concept (an abbreviation of Point Of Interest). A given POI can be of four different types: a *Restaurant*, an *Event* such as a live concert, an *Accommodation*, or an *Attraction* such as a museum). Because the data available when the prototype was developed only was about tourist attractions, only the features that describe this type of items were modeled in the domain. Four different feature types are represented: the *FunctionalType*, whose hierarchy of concepts classifies attractions by their functional type; the *EducationalSubject*, which classifies the attraction by the type of education offered to the tourists; the *POI_Facility*, which classifies the items by their facilities available; and the *EntranceType,* which describes the type of discounted or free entrance of an attraction. In the Figure 4.3 are represented the concepts and relations previously mentioned.

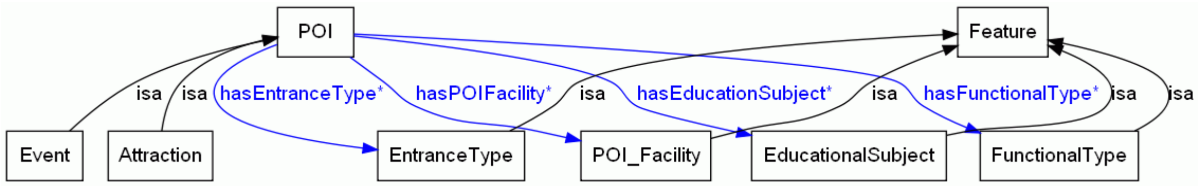


**Figure 4.3** Representation of points of interest (POI) and features

Most of the feature hierarchies have been designed using a bottom-top approach according to the available information about attractions extracted from the Barcelona Council's website. Although there is some exception in which the hierarchies are based on taxonomies of reference. Next, the four hierarchies are presented in more detail.

The *FunctionalType* hierarchy is the biggest and has a max depth of four levels. Its design is in part based on a larger taxonomy developed in the PICTURE project[31], which was financed by the European Commission in the Sixth Framework Programme of Research. In the Figure 4.4 the hierarchy is only partially shown by reasons of space.

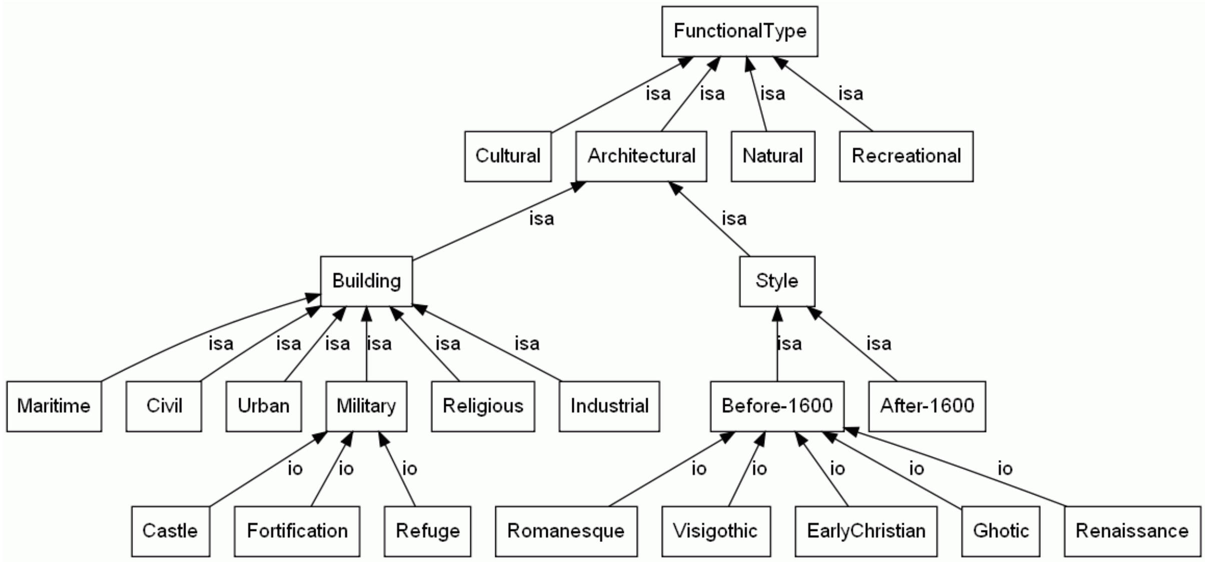


**Figure 4.4** Hierarchy of the feature *FunctionalType*

The *EducationalSubject* hierarchy only has a first level of depth and is represented in the Figure 4.5.

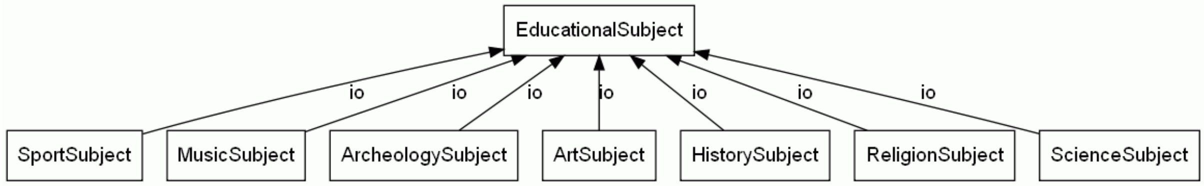


**Figure 4.5** Hierarchy of the feature *EducationalSubject*

The *POI_Facility* hierarchy has two levels of depth and is shown in the Figure 4.6.

[31] See [http://www.picture-project.com] (last access on September 3, 2009)

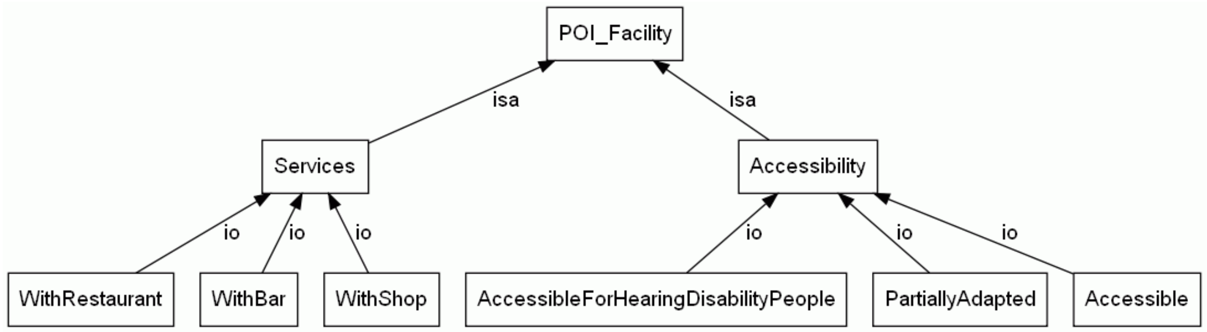


**Figure 4.6** Hierarchy of the feature *POI_Facility*

The *EntranceType* hierarchy has three levels of depth and only some of its concepts are shown in the Figure 4.5 for reasons of space.

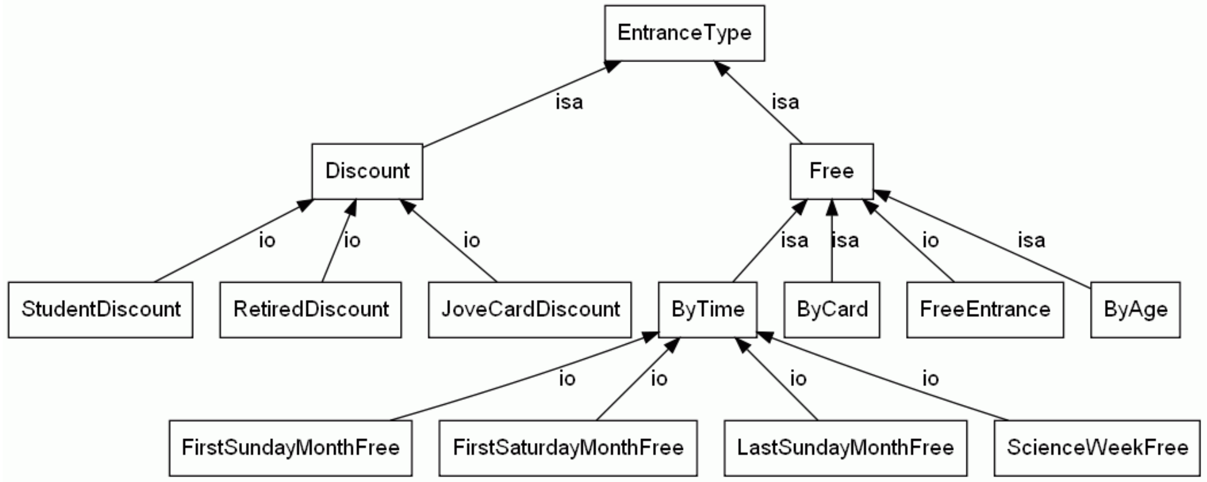


**Figure 4.7** Hierarchy of the feature *EntranceType*

### *4.2.3. Identifying and modeling tourist stereotypes*

Defining user stereotypes for an application domain is usually a laborious task that requires experts of the domain. In this work, the chosen stereotype classification in the tourism domain is based on a statistical analysis of cultural tourism in Europe that distinguishes two broad types of tourist [56]:

- The *specific cultural tourist*, for whom visiting cultural sites and attractions is the primary reason for the journey. This type of tourists is drawn mainly from the middle classes and usually has a higher level of education. Also they are usually well-off. According to some statistics presented in [56], the percentage of specific cultural tourists is small: they may represent as little as 10% of the cultural tourism market; and most of the half have an age between 23 and 40 years.

- The *general cultural tourist*, who take in cultural tourism as part of their broader interest in going on holidays. In this stereotype are classified all the tourists that does not fit with the characteristics of the specific ones, which according to the statistics this group is the most common with approximately a 90% of the tourists. In contrast to *specific cultural tourists*, *general cultural tourists* tend to regard cultural tourism as a secondary activity, subordinate to sporting activities, shopping or general sightseeing, visiting only the iconic and emblematic attractions.

In the Figure 4.8 the main concepts of the stereotype model in the tourism domain are shown. A particular stereotype (instance of *Stereotype*) is composed of two main types of information: a list of predictions on interest (instances of *InterestOpinion*), and the set of characteristics that identify the stereotype (instances of *StereotypeCharacteristic*). An interest opinion is mapped to a topic belonging to a certain feature hierarchy and its DOI value (*InterestDegree*). A stereotype characteristic is of a certain *StereotypeData* weighted with a float value indicating the probability that a given user with this characteristic will pertain to the stereotype. The weights of characteristics belonging to the same type are normalized. Although more types of *StereotypeData* could be used to describe tourist stereotypes such as the travel company, the job position and hobbies; only these were used because it is the information supposed to be found in an existing FOAF profile using the extensions described in section 3.2.2.

A possible definition for the two broad tourist stereotypes identified is presented in the tables 4.1 and 4.2. Because the available description of each stereotype profile is too general and the differences in interest only refers to the *FunctionalType* feature, the prediction on user's interests is only based on the first level of concepts of the this feature.

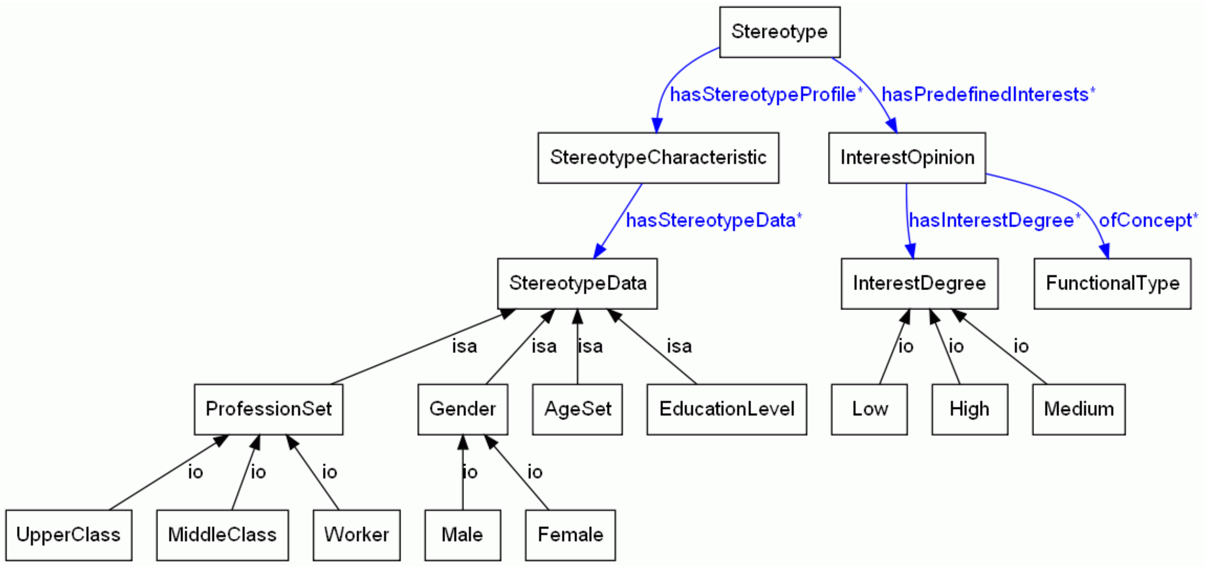


**Figure 4.8** Representation of user stereotypes

<table>
<tr><th colspan="11">SPECIFIC CULTURAL TOURIST</th></tr>
<tr><th colspan="11">Profile characteristics</th></tr>
<tr><th colspan="3">Profession Set</th><th colspan="2">Gender</th><th colspan="3">Age Set</th><th colspan="3">Education Level</th></tr>
<tr><td>UpperClass</td><td>MiddleClass</td><td>Worker</td><td>Male</td><td>Female</td><td>< 23</td><td>23-40</td><td>>40</td><td>Higher</td><td>Secundary</td><td>Primary</td></tr>
<tr><td>0.55</td><td>0.40</td><td>0.05</td><td>0.5</td><td>0.5</td><td>0.05</td><td>0.6</td><td>0.35</td><td>0.80</td><td>0.15</td><td>0.05</td></tr>
<tr><th colspan="11">Predictions on general interests <i>(FunctionalType)</i></th></tr>
<tr><th colspan="2">Architectural</th><th>Cultural</th><th colspan="2">Natural</th><th colspan="2">Recreational</th><th colspan="2">Traditional</th><th colspan="2">Sportive</th></tr>
<tr><td colspan="2">HIGH</td><td>HIGH</td><td colspan="2">MEDIUM</td><td colspan="2">LOW</td><td colspan="2">MEDIUM</td><td colspan="2">LOW</td></tr>
</table>

**Table 4.1** Description of the specific cultural tourist stereotype

<table>
<tr><th colspan="11">GENERAL CULTURAL TOURIST</th></tr>
<tr><th colspan="11">Profile characteristics</th></tr>
<tr><th colspan="3">Profession Set</th><th colspan="2">Gender</th><th colspan="3">Age Set</th><th colspan="3">Education Level</th></tr>
<tr><td>UpperClass</td><td>MiddleClass</td><td>Worker</td><td>Male</td><td>Female</td><td>< 23</td><td>23-40</td><td>>40</td><td>Higher</td><td>Secundary</td><td>Primary</td></tr>
<tr><td>0.5</td><td>0.10</td><td>0.85</td><td>0.5</td><td>0.5</td><td>0.50</td><td>0.15</td><td>0.35</td><td>0.05</td><td>0.10</td><td>0.80</td></tr>
<tr><th colspan="11">Predictions on general interests <i>(FunctionalType)</i></th></tr>
<tr><th colspan="2">Architectural</th><th>Cultural</th><th colspan="2">Natural</th><th colspan="2">Recreational</th><th colspan="2">Traditional</th><th colspan="2">Sportive</th></tr>
<tr><td colspan="2">MEDIUM</td><td>LOW</td><td colspan="2">MEDIUM</td><td colspan="2">HIGH</td><td colspan="2">MEDIUM</td><td colspan="2">MEDIUM</td></tr>
</table>

**Table 4.2** Description of the general cultural tourist stereotype

## 4.3. Integration of the personalized recommendation service

In this section, it is presented how the recommendation service, exposed via Web services, has been integrated into the tourism-domain-Web-application. Although there are some common design patterns that all domain application should be employ, depending on their specific needs and use cases each application could use its ad-hoc integration design.

In Figure 4.7 a sequence diagram in UML[32] showing how the tourism-domain Web-application and the recommendation service interact with each other in order to offer a personalized recommendation to a given user, from the user login to the user logout, is presented. In the left part, the main components of the tourism Web-application are represented: the *Interaction Manager*, which is in charge of process the different user events caught by the Web page; the *Content Retriever*, which retrieves the set of items related with a specific query according to the particular user context; and the *Stateful Webservice Client,* which is the component in charge of managing the interaction with the Web service endpoints exposed by the recommendation service. In the right part, there are the two Web service endpoints of the recommender (explained in detail in section 3.4.2): the stateless endpoint *RecommendationServiceLogin*, and the Stateful endpoint *RecommendationService*.

Basically, the diagram shows the following sequence of operations:

1. When the user is logged in, a new instance of *Stateful Webservice Client* is created and used to start a new user session into the recommendation service. First, the *InteractionManager* calls to the *RecommendationServiceLogin Endpoint* in order to obtain the new instance of *RecommendationService Endpoint,* and then it calls the one-way operation for creating the new user session.
2. When the user makes a search query, two different operations of the *RecommendationService Endpoint* are called sequentially. First of all, the *ContentRetriever* filters the list of items to be recommended based on the search query and the user context, and calls the *getRecommendation* operation passing the items list that returns the top-10 recommendation. Then, the *newQueryEvent* operation is called for registering the user event.

   3. When the user logout, the *closeUserSession* operation of the *RecommendationService Endpoint* is called, ending the user session in both the recommendation service and the Web-application.

[32] Unified Modeling Language (UML) [http://www.uml.org/]

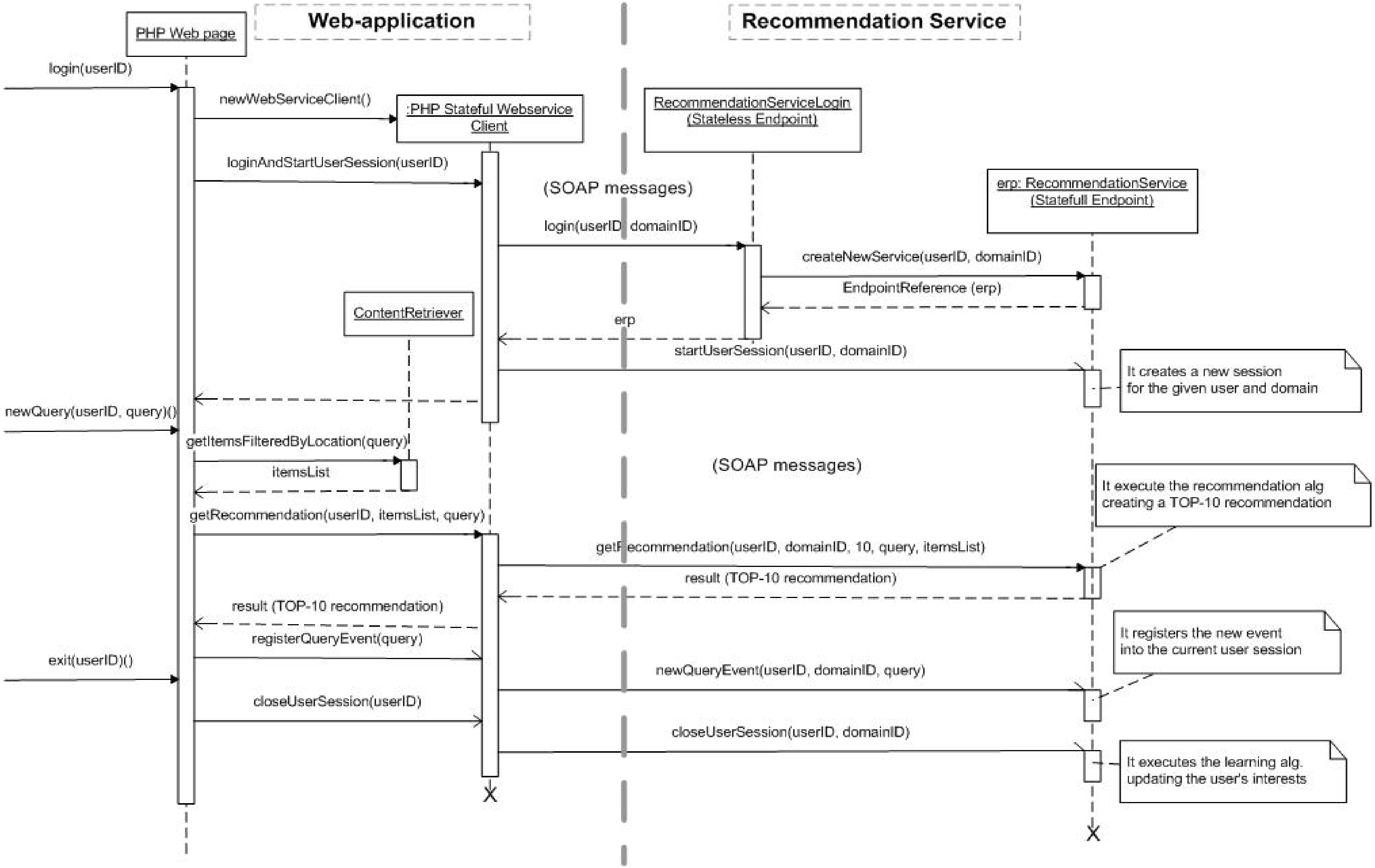

Web-application
Recommendation Service
PHP Web page
:PHP Stateful Webservice Client
RecommendationServiceLogin (Stateless Endpoint)
erp: RecommendationService (Statefull Endpoint)
ContentRetriever
login(userID)
newWebServiceClient()
loginAndStartUserSession(userID)
(SOAP messages)
login(userID, domainID)
createNewService(userID, domainID)
EndpointReference (erp)
erp
startUserSession(userID, domainID)
It creates a new session for the given user and domain
newQuery(userID, query)()
getItemsFilteredByLocation(query)
itemsList
(SOAP messages)
It execute the recommendation alg creating a TOP-10 recommendation
getRecommendation(userID, itemsList, query)
getRecommendation(userID, domainID, 10, query, itemsList)
result (TOP-10 recommendation)
result (TOP-10 recommendation)
registerQueryEvent(query)
It registers the new event into the current user session
exit(userID)()
newQueryEvent(userID, domainID, query)
closeUserSession(userID)
closeUserSession(userID, domainID)
It executes the learning alg. updating the user's interests

**Figure 4.9** UML's Sequence diagram showing the recommendation service integration

# CHAPTER 5 - Experimental evaluation

The main objectives of the undertaken experimental evaluation are:

- to evaluate the correctness of the user profile learning and recommendation algorithms; and
- to evaluate the improvement of the recommendation performance in terms of accuracy with respect to traditional content-based recommenders.

A strong limitation for the evaluation of the recommendation system has been not to dispose of suitable, real-usage data in the tourism domain. For this reason, an artificial user profile has been used in the experiments. This limitation restricts the set of possible experiments; however, the ones carried out and their evaluation have been useful to refine some parameters and formulas of the learning and recommendation algorithms implemented.

## 5.1. Experimental data sets

The experimental data set contained 180 topics in the feature hierarchies of the tourism domain (see section 4.2.2) and a total of 1288 tourist attractions indexed under several of these topics. A negative characteristic of this dataset is the variability of the number of topics and features, which implies that some attractions are better classified than others. In Figure 5.1 two examples of tourist attractions with a different number of topics and features is shown. Due to this, items with fewer topics will have more possibilities to obtain higher scores and this could affect the accuracy of the recommendation algorithm.

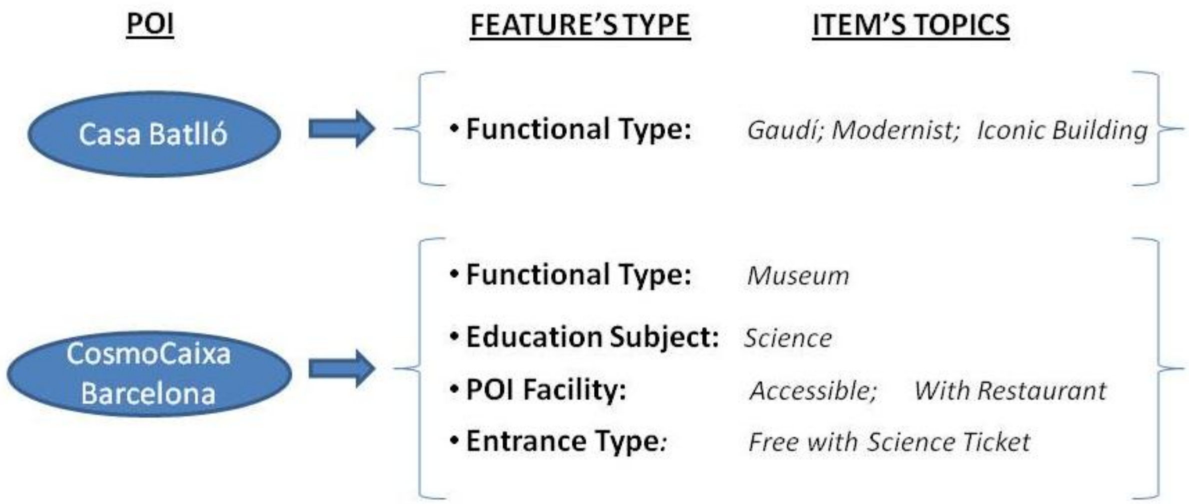


**Figure 5.1** Example of two items with different number of topics and features

As real user-data (either explicit or implicit feedback) were not available relative to the tourism data set used, it was decided to create an artificial user profile using a set of user behaviors related with specific topics. The types of user events that have been used are the ones in the recommendation-service description: the user query and the item-selection event. The method to simulate the predefined user-behavior is the following:

1. Definition of the normalized weights for each direct sub-concept of the *Functional Type* feature (Architectural, Cultural, Natural, Sportive, Traditional, Recreational), representing the "real" user's interests. This feature was chosen because is the only that appears in all item's representation and therefore the amount of items associated with each *FunctionalType* topic is so higher than using topics of the other features.
2. Automatic creation of a set of queries and item-selection events for each topic proportional to the predefined weights. Although the selection of queries is done randomly, in order to get a more real user behavior patterns the selection process follows the hierarchical structure of the domain similar to the type of interaction that allows the new search interface of the prototype (see section 4.1.3). Thus, if a selected query has sub-topics, then one of them is chosen and so on.
3. Execution of all the selected events distributed in different user session of 30 events each. For each query executed, two item-selection events are registered into the recommendation service. Each event is randomly chosen. The process works until the all selected events are registered into the recommendation service. For the experiments a total of 300 user events were used to learn the user profile.

## 5.2. Experimental methodology and discussion of results

In this section, the undertaken experiments to evaluate some aspects of the recommendation system, as well as the discussion of their respective results are presented.

### *5.2.1. Evaluating the user profile learning algorithm*

The objective of this experiment is to evaluate the user-profile learning-algorithm in terms of accuracy and to ensure that the predicted DOI values reflect the long-term user's interests according to the predefined user-behavior.

The method consists of executing the previously mentioned simulation and, after each session, obtaining the predicted DOI values for the analyzed topics (direct sub-topics of *FunctionalType* feature). The results showing the predictions, calculated as the product between the DOI_weight and DOI_CL values, in each user session are shown in Figure 5.2.

Note that the predicted DOI values are always positive, since the learning algorithm is only using the implicit-feedback source.

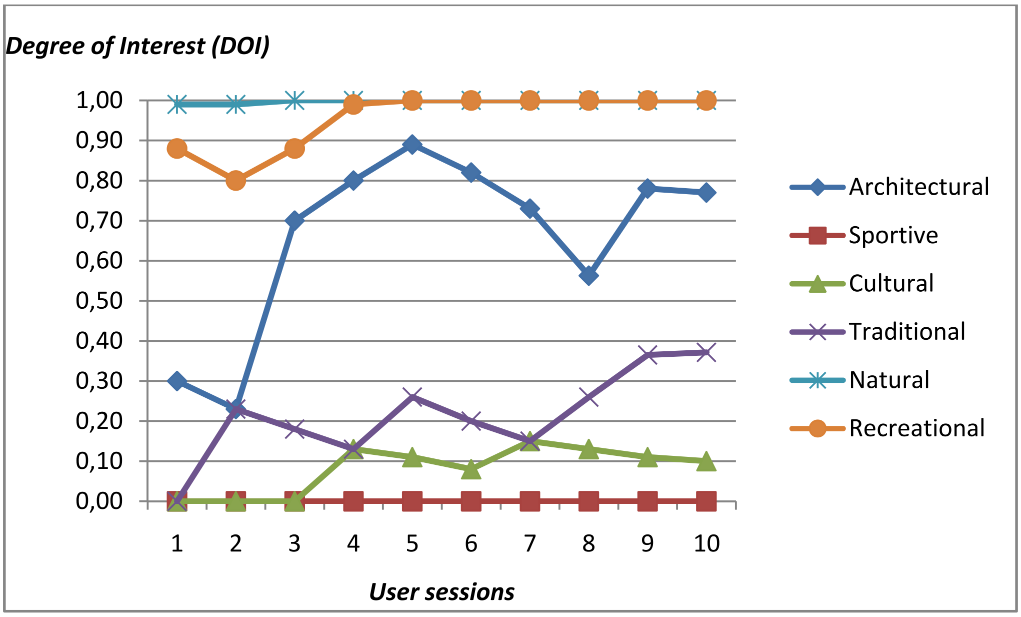


**Figure 5.2** Chart showing the predicted degrees of interest over time

In addition, a comparison between the DOI values predicted after the last session and the predefined ones are shown in Figure 5.3 and Table 5.1. The values have been normalized.

| | Architectural | Sportive | Cultural | Traditional | Natural | Recreational |
|---|---|---|---|---|---|---|
| **Predefined DOI values** | 23,00% | 0,00% | 3,00% | 8,00% | 31,00% | 35,00% |
| **Predicted  DOI values** | 23,76% | 0,00% | 3,09% | 11,46% | 30,85% | 30,85% |

**Table 5.1** Results of the learning algorithm evaluation

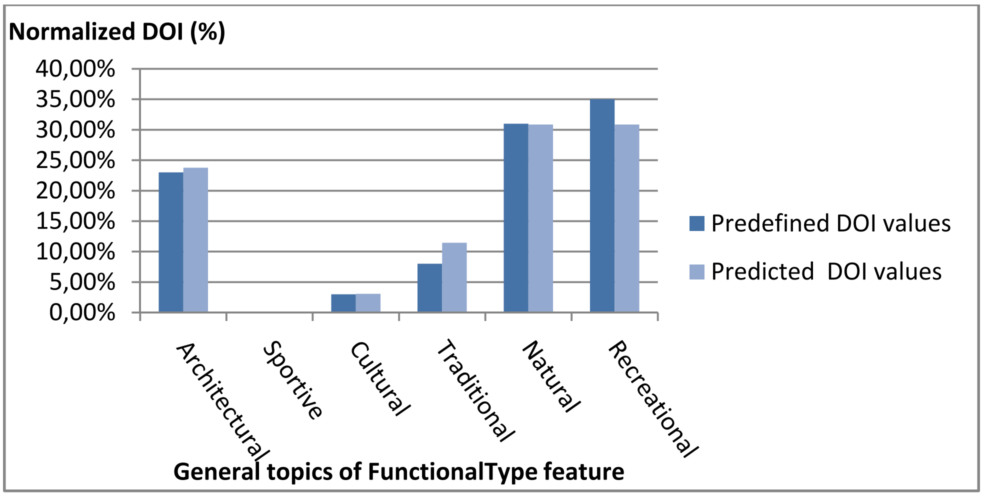


**Figure 5.3** Bar chart showing the accuracy results of the learning algorithm

From the results of the experiment can be concluded that the learning algorithm works properly and reflects quite accurately the predefined long-term user's interests. As the events are randomly selected, the proportions of events related with each topic are not maintained after each session, what explains the changes of the predictions between sessions observed in Figure 5.2. However, the global interest priority is almost maintained during all the simulation (*Recreational* > *Natural* > *Architectural* > *Traditional* > *Cultural* > *Sportive*), what means that the predictions are correctly updated over time.

Due to the random nature of the undertaken experiment, only analyzing some general topics do not reflect the user's interests in detail. But for our evaluation purposes it gives us enough information to evaluate if the learning algorithm evolves adequately over time.

In the Figure 5.3 the predicted DOI values for each topic are compared with the predefined ones. It can be observed that in most cases the prediction virtually matches with the ideal one. The two particular cases with a higher error are the *Traditional* and *Recreational* topic. There are two main assumptions that explain these deviations:

- The fact that the implicit weight (**iw**) is calculated by means of a sigmoid function implies that the user's interests in topics with most number of occurrences have very similar weights (near to 1). And this is what happens in the case of the *Natural* and *Recreational* topics, in which the predictions are very similar.
- Variations in depth and the size of the hierarchy branches to which the topics belong, also imply variations in the inferred weights (**dw**), though the proportion of user events is the same. As the events are randomly selected, the deeper and larger is the branch, the lower is the value of **iw** and DOI_weight for each topic and therefore the lower is the **dw** value. In the experiment, the *Traditional* topic has the smallest branch with only 1 level of depth and 2 sub-topics. For this reason, it is the most overestimated predictions. In the case of the *Recreational* topic, the effect is the contrary. The topic has been underestimated respect to the predefined number of events, in which the predicted DOI should be greater than in the *Natural* topic. This is because the branch of the *Recreational* topic is deeper and larger. From these results is concluded that the accuracy of the learning algorithm is closely related with the quality of the domain classification, in which the most important thing is to employ the same level of detail in all the branches of the hierarchy in order to develop more balanced classifications.

### *5.2.2. Evaluating the recommendation algorithm*

In this experiment the objective is to evaluate the quality of top-n recommendations over time. Because the absence of explicit feedback, such as user ratings, the most common used accuracy metrics for evaluating recommenders [57] cannot be employed in this experimental evaluation. Hence, some ad-hoc metrics have been designed in order to be able to evaluate the system using only simulated implicit feedback, not in terms of recommendation accuracy but in terms of correctness. The metrics that have been used are the followings:

- The *relevancy ratio*, which measures the ratio of items considered relevant in relation with the size of the recommendation list. An item is relevant when the score predicted is greater than a threshold. As the experiment has been done with implicit feedback that only predicts positive evidences of interests, the threshold is equivalent to a prediction of 0.75, which in the typical five-star scale would be equal to items with 4-5 stars of suitability. The range of possible values is [0,1].
- The *relevant items-proportion*, which measures the proportion of relevant items in relation with the total number of items to recommend. The range of values is [0,1];
- The *user satisfaction* that tries to measure how much the user is satisfied by the recommendation. It is calculated as a combination of two factors:
    1. a measure indicating the goodness of the recommendation taking into account its topic diversification;
    2. and the *relevancy ratio* metric of the recommendation.

  In order to calculate the first factor, the ideal topic proportions are pre-calculated based on the predefined DOI values that can be observed in Table 5.1. In this particular case, the 'ideal' top-10 recommendation has the following proportions: (*Recreational=4; Natural=3; Architectural=2; Traditional =1; Cultural=0; Sportive=0*). Then, this ideal topic-diversification is compared with the one of the top-10 recommendation generated. The higher the similarity between the two proportions, the higher the value of the user satisfaction metric. The range of values is also [0,1]; where values close to 1 indicates a good matching. The second factor is less important in this case because is measuring the relevance from the system's viewpoint instead of the user's viewpoint.

The method of this experimental evaluation consists of executing the most general query (all items with some *FunctionalType* feature) for each user session of the predefined simulation. The results of the experiment can be observed in the Table 5.2 and represented in Figure 5.4.

| | S1 | S2 | S3 | S4 | S5 | S6 | S7 | S8 | S9 | S10 |
|---|---|---|---|---|---|---|---|---|---|---|
| **Relevancy ratio** | 0,7 | 1 | 1 | 1 | 0,9 | 1 | 1 | 1 | 1 | 1 |
| **Relevant items proportion** | 0,13 | 0,49 | 0,49 | 0,55 | 0,45 | 0,51 | 0,51 | 0,52 | 0,52 | 0,59 |
| **User satisfaction** | 0,32 | 0,7 | 0,55 | 0,77 | 0,83 | 0,92 | 1 | 1 | 1 | 0,92 |

**Table 5.2** Results of the recommendation algorithm evaluation

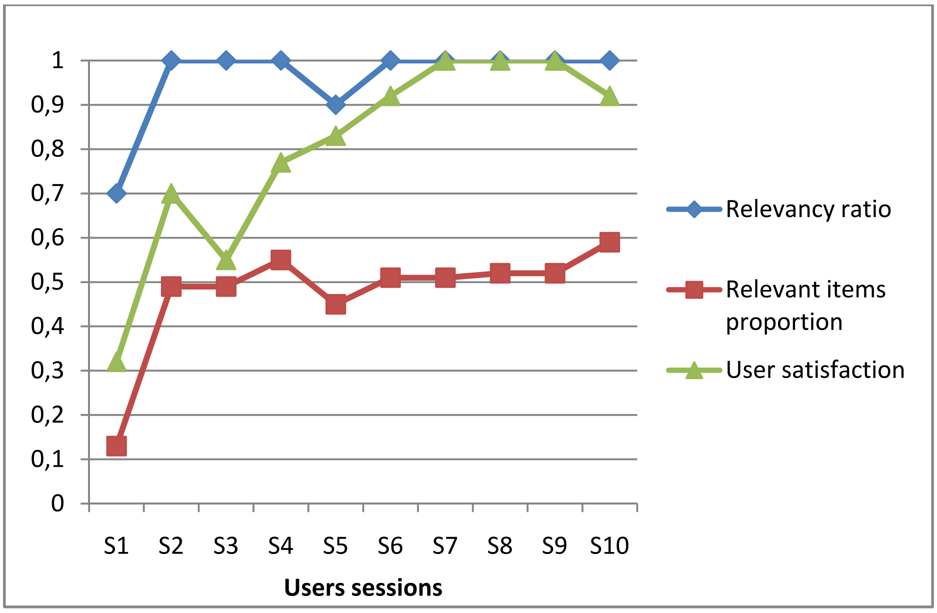


**Figure 5.4** Chart showing the results of the recommendation algorithm over time

Analyzing together the three metrics, it can be observed that the tendency over time is the improvement of the recommendation's quality. As it was expected, the more accurate the user profile, the better the top-10 recommendations. In particular, from the session 5 is when the quality of the recommendation in terms of user satisfaction is practically perfect according to the metrics employed.

The relevancy ratio and the relevant items-proportion are strongly related with the quantity of information available about the user, and their values tend to grow over time. This is, in part, due to the characteristics of the experiment that is only based on implicit feedback information, what implies in general that the values increase after each session.

Respecting to the user satisfaction metric, which measures the quality of the recommendation in terms of ideal topic diversification and relevancy, the tendency of growth is similar to the other metrics. In this case, the more similar the predicted DOI to the predefined ones, the more similar the topic-diversification to the ideal recommendation list.

### *5.2.3. Evaluating the semantic recommendation system*

The objective of this experiment is to evaluate in which circumstances and how the ontology-based learning algorithm and the content-based recommendation algorithm enhanced with semantic information improve the performance of the recommendation system in terms of recommendation quality and accuracy. To evaluate this, two different configurations of the recommender have been set up:

- the SemRec, which is the configuration working with the semantic components developed in this work and presented in chapter 3;
- the Rec, which is the same recommender but without exploiting the ontology-based components: the weighting propagation based on domain inferences in the learning algorithm, and the hierarchy-based similarity measure and the topic diversification method in the recommendation algorithm.

#### *5.2.3.1. Comparing the user profile learning algorithm*

In order to compare the learning accuracy between the ontology-based learning algorithm (SemRec) and the configuration only using the statistical model (Rec), the two configurations were executed with the same simulated user profile (the one of section 5.2.1). In Figure 5.5 the resulting DOI predictions for each configuration are compared with the predefined ones, and their respective error rate percentages are shown in Table 5.3.

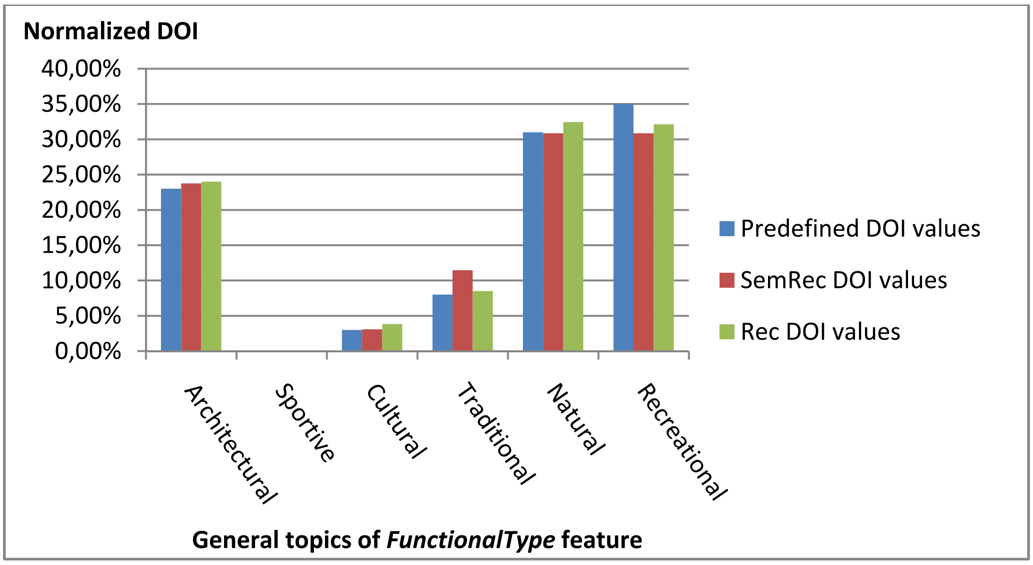


**Figure 5.5** Bar chart comparing the results of the learning algorithm

| | Architectural | Sportive | Cultural | Traditional | Natural | Recreational | Avg. error |
|---|---|---|---|---|---|---|---|
| **SemRec** | 0,76% | 0,00% | 0,09% | 3,46% | 0,15% | 4,15% | 1,43% |
| **Rec** | 1,01% | 0,00% | 0,84% | 0,52% | 1,44% | 2,88% | 1,11% |

**Table 5.3** Results with error rates of the learning algorithm predictions

In addition to these results, after each user session the number of learned interests was analyzed in order to compare the learning rate of the two configurations. The results can be observed in the Figure 5.6.

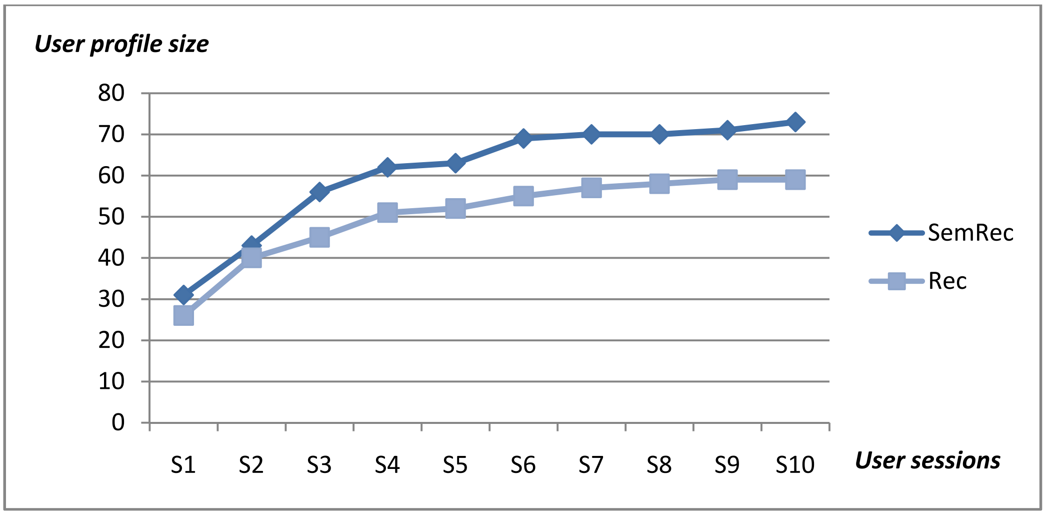


**Figure 5.6** Chart comparing the number of learned interests over time

Observing the error rate percents of the two configurations in the Table 5.3, it can be observed that the Rec configuration has obtained slightly more accurate predictions, although the difference is minimal. These results are not very surprising, since the only source of information used in the experiment is usage data and, as the Rec configuration is only based on this feedback, the predictions reflect more accurately the user-behavior. This fact implies that the small differences are due to the weighting propagation technique. Although this technique introduces some deviations in the predictions because it is based on domain generalizations, it achieves an increase in the number of learned user's interests (as can be observed in Figure 5.6). This is an important strength of personalized recommendation systems to overcome the typical cold-start problem. The main issue here is to find the trade-off between the use of domain inferences or more trustworthy information sources such as ratings and user behavior. In this work, its use is restricted to specific condicions and, in general, the **iw** values have a low influence in the predictions when other partial weights are available.

### *5.2.3.2. Comparing the quality of top-n recommendations*

In this experiment the recommendation algorithm of the two configurations are compared: one is the algorithm presented in this work with all the semantic components activated (SemRec); and the other is the same algorithm but with the hierarchical-based similarity and topic diversification disabled (Rec). The average values for each metric are presented in Table 5.4 and compared in Figure 5.7.

| | Relevancy ratio | Relevant items proportion | User satisfaction |
|---|---|---|---|
| **SemRec** | 0,95 | 0,40 | 0,71 |
| **Rec** | 1,00 | 0,49 | 0,62 |

**Table 5.4** Results of the recommendation algorithm comparison

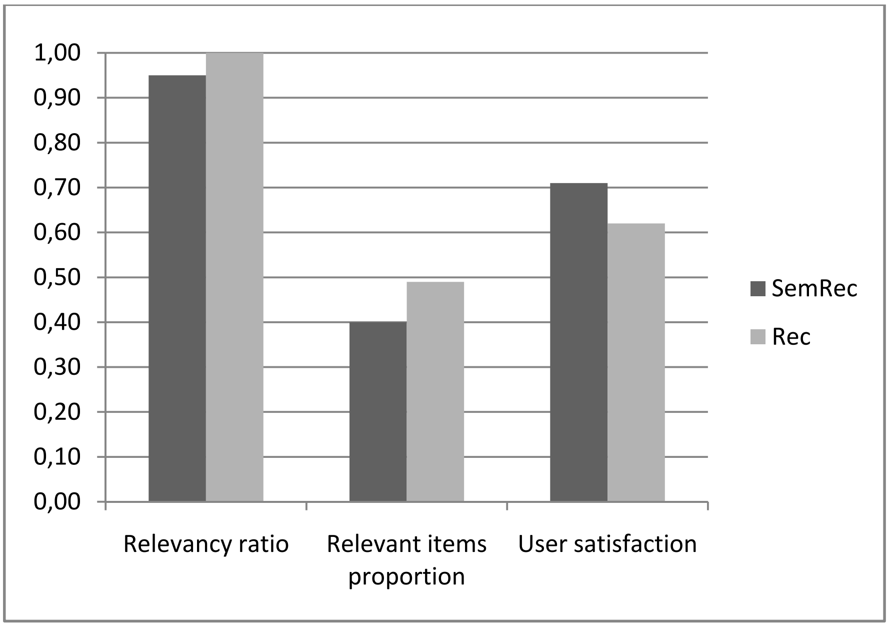


**Figure 5.7** Bar chart comparing the results of the recommendation algorithm

From the above results, the following conclusions can be obtained about the type of improvement achieved using the content-based recommendation algorithm enhanced with the semantic similarity and topic diversification methods:

- The fact that the SemRec configuration obtained, on average, a lower “Relevant items proportion” is an indicator that the item-score calculation algorithm using the hierarchy-based similarity and the domain-based feature-weighting mechanism (explained in section 3.3.2.2) is more restrictive. In this case, the reduction of the number of relevant items is considered as an improvement in accuracy, since the algorithm is reducing the set of possible items to recommend.
- The high values in user satisfaction of the SemRec configuration are due to the topic-diversification method that constructs more diversified top-n recommendations than in the Rec configuration, in which the metric remains constant over time because the algorithm always return the most relevant items.
- The slightly lower value in relevancy ratio of SemRec is the consequence of generating topic-diversified recommendations. It is common sense that the more diversified a recommendation, the higher the probability of recommend irrelevant items. This is the well-known trade-off between finding accurate recommendations and avoiding the overspecialization by generating diversified recommendations that usually stimulate serendipity and novelty.

### *5.2.3.3. Comparing the algorithm performance*

A comparison of the time required for the learning and recommendation algorithm between the two configurations is shown in the Table 5.5. The time for the learning algorithm has been calculated as the average time required in process 30 user events for ten sessions. The time for the recommendation algorithm has been calculated as the average time required to generate a top-10 recommendation during ten sessions and for different sizes: a 1 item prediction, a query with 260 items and a query with all the available items (1288).

| | **Elapsed time** (ms) **of the recommendation service** | | | |
|---|---|---|---|---|
| | ***Learning alg.*** *(30 events)* | ***Recommendation alg.*** (query size) | | |
| | | 1288 items | 260 items | 1 item |
| **SemRec** | 15.596 | 1.431 | 289 | 1,1 |
| **Rec** | 12.162 | 1.288 | 218 | 1,0 |

**Table 5.5** Results of the algorithm performance comparison

From the above results, it can be concluded that the differences between the two configurations are minimal. However, as the ontology-based algorithms depend on the size and depth of the domain ontology, it would be necessary to evaluate the performance of the SemRec with a larger ontology in order to see how the size increases the computational cost.

Another issue to comment is the high computational cost of the learning algorithm in comparison with the time required for the recommendation algorithm. This is because the learning algorithm has to access to the persistent ontology model associated with the users database in order to be able to update the user profiles in real time, and this implies a high cost in time. In the first version of the recommender implementation, in which the tourism model also was loaded as a persistent ontology model, the time required for the recommendation algorithm was 56 times longer.

## *5.2.4. Evaluating the recommendation service performance*

In this experiment the objective is to measure the total average time required for obtaining a top-10 recommendation from the user sends the query to the user receives the recommendation, in order to evaluate the overall performance of the recommendation service integrated within the Web-application. The results are shown in Table 5.6.

| | **Total elapsed time using the tourism Web application** (ms) | | |
|---|---|---|---|
| | **Web Application**<br>(Content retrieval) | **Recommendation service**<br>(Top-10 recommendation) | **TOTAL time** |
| **1288 (items** | 6.950 | 1.431 | 8.381 |
| **260 (items)** | 1.630 | 289 | 1.919 |

**Table 5.6** Results of the overall system performance in a top-10 recommendation request

From the above results, it can be observed that the semantic-based content retrieval component of the Web-application is the most time consuming, and this is due to the semantic queries used to retrieve the items by category. This type of queries requires a lot of time because in RDF-based databases the system cannot use indexing methods as in relational databases to reduce the time per access, and it has to find statement by statement. In RDF graphs of more than 5000 statements, like the one used in this experiments, this process has a high computational cost.  In contrast, the time required for the recommendation algorithm, which has a linear order cost O(N) where N is the number of items used for the recommendation, is quite more efficient in comparison to the time of the content retrieval component.

Taking into account the overall time required for all the personalization process from the user's viewpoint, the online system developed is able to provide personalized recommendations in acceptable times with the existing sizes of item's list. However, queries with a greater number of items than 1500 items should be avoided in online services, since the response time will be longer than 10 seconds, and this would probably affect the user satisfaction.

# CHAPTER 6 - Conclusions

In this work, a personalized recommendation system has been developed and integrated into a Web-application in the tourism domain. The recommender, a content-based filtering approach that exploits semantic knowledge, overcomes or reduces the most common limitations of traditional approaches: the cold-start problem and the sparsity problem. A potential limitation of the proposed approach is overspecialization, that is, recommendations including excessively similar items. This problem has been partially solved by using a method that diversifies recommendations according to the sub-topics the user is more interested in with respect to the topic of the user query. The semantic recommender reduces the new-user cold-start problem by means of a weighting-propagation technique based on domain inferences. As it has been observed in the results of the experimental evaluation in Figure 5.6, user's interests are always better learned using the proposed technique than using only an approach based on statistics. This implies that the time needed to learn an accurate user profile and therefore to generate high-quality recommendations is reduced. Although the undertaken experimental evaluation of the accuracy of the content-based algorithm enhanced with semantic knowledge was limited (because it used simulated data), the results in Figure 5.7 have shown that the item-score predictions are more accurate than the ones obtained with a traditional approach. However, an evaluation based on real usage data would be necessary in order to confirm these preliminary results.

The architectural design of the recommendation system, based on the SOA paradigm and implemented as a Web service, allows the easy integration of the recommender into any Web-application. Moreover, the use of FOAF profiles as the basis of the user model facilitates the reuse of existing user profiles, and allows taking advantage of existing identification protocols. In this sense, the OpenID service is proposed for the implemented tourism application to avoid the users the typical, sometime tedious process of registering to a new website. Thanks to the high interoperability of the SOAP protocol, the integration of the recommendation service is quite straightforward and Web-applications need not too many changes in their source code. The only requirements that Web-applications must satisfy in order to integrate the recommendation service are the following ones: an OWL ontology must be available, describing the domain-dependent information, the items' features and the stereotypes; the items of the domain must be classified using the concepts of the domain defined in the ontology; a Web-service client able to maintain the user session must be implemented in order to use the public operations of the recommendation service.

Due to the variety of user-information collection techniques that the recommender employs to acquire and construct the user profile, the system is able to model the long-term user's interests without using intrusive methods. In the worst case, in which a new user provides no explicit feedback, the combination of the predictions made by the stereotype approach, the usage-data statistical method and the domain-inferences allow generating acceptable recommendations in a relatively short time, depending on the domain ontology used. However, the time required to learn an accurate user model in these conditions is higher than with a user that provides explicit feedback from the beginning.

The recommendation system is able to start recommending from scratch, with no previous usage-data coming from collaborative filtering, since it only relies on learning methods mainly focused on the individual user information. Only in the case of the usage-data statistical method, the recommender needs data from other users and events in order to be reliable. But this situation is solved by temporally using only the individual information to build the statistical model.

As for the overall performance of the system, the results of the experimental evaluation of the recommendation service integrated into the Web-application in the Table 5.6 show that the computational cost of generating a recommendation is minimal in comparison with the time needed for the Web-application to obtain the list of items to be recommended. Although the time needed by the user-profile learning algorithm is quite long according to the results in Table 5.5, the user is not affected because the learning algorithm is executed after the user ends the session. Moreover, it also has been shown that the computational cost associated with the semantic-based techniques is insignificant using the tourism domain ontology of 180 concepts and 4 levels of depth developed in this work. Despite this, it would be interesting to evaluate the recommender performance using larger ontologies in order to see how the size and depth of the ontology affects the computational cost in each stage of the recommendation process.

## 6.1. Exploitation of results

The company TMT Factory aims to exploit **Streetbox**, one of the company's main products, enhancing its functionalities using the personalization and recommendation system defined in this thesis. **Streetbox** is an ICD deployed in the urban environment which offers multimedia content to people living in or visiting a city. The personalization system can be then adapted and extended to other TMT Factory's products such as **Lobbybox**, **Beebox** or **DSbox**. **Lobbybox** is an ICD deployed in the lobby of the hotels, i.e. indoor places, **Beebox** is an interactive television offering advanced services to hotel and hospital clients, and **DSbox** is a content management system.

# CHAPTER 7 - Future work

As often happens, some ideas and improvements have remained to be done. In the following sections possible future work about the recommendation system and the tourism-domain Web-application developed are presented.

## 7.1. About the recommendation system

One of the most important things to complete the analysis of the recommendation system is to evaluate the accuracy of the recommender by combining: the results obtained from an existing dataset, with explicit and implicit feedback of real users available; and the results obtained from a live user-experiment where the users are explicitly asked for giving their feedback. In this manner, more reliable accuracy metrics may be obtained about the quality of the recommendation in terms of accuracy and real user satisfaction.

In order to develop a commercial version of the recommendation service, some more architectural extensions and careful analysis would be needed. On the one hand, the Web-service endpoints should be extended with more operations offering user-profile management functionality, which for the purposes of this work more focused on the recommendation algorithms, it has not been implemented. On the other hand, privacy and security issues about the implementation should be analyzed in more detail, as well as evaluating the system's performance and behavior in extreme conditions, such as the one having a high number of concurrent users and Web-applications.

A possible improvement of the recommendation system that should be evaluated is to exploit social-network information, which could be extracted from the FOAF profiles, to offer trust-based collaborative-filtering recommendations (see section 2.5). The idea would be to develop a hybrid recommendation strategy, in which the content-based and trust-based recommendations are combined to generate high-quality recommendations. A solution would be to use a feature-augmentation strategy, in which the recommendation generated by the content-based algorithm is complemented with the opinions or textual recommendations obtained from the trust-network of the user. One of the tasks of the system would be to maintain a trust-network of possible recommender-users for each user of the system taking into account the similarity of their opinions.

## 7.2. About the tourism domain Web-application

The Web-application in the tourism domain that has been extended and adapted for demonstrations purposes is far to be a final product and could be improved in various aspects. This section is only focused on future work related with the personalized service developed.

The current version of the experimental navigation interface, which has been implemented combining a tag-cloud and tree-view design, is not personalized to the user. A possibility to personalize the user interface is to use the user's interests modeled by the recommendation service to change the size of the topic-words according to the DOI predicted for the particular topic and user.

For a commercial version of the system some extra information about the places would be needed in order to offer a complete tourism information-service, such us the location in GPS coordinates of the places, detailed descriptions and images. Taking advantage of the developed semantic-database based on RDF resources, this extra information could be dynamically obtained from public semantic-datasets such as the one of DBPedia. Basically, the process would consist of two steps: first, the items of the current dataset, represented as RDF resources, has to be linked to the public resources representing the same object using the adequate properties, such as the *owl:sameAs* property; then, once the items are linked to their respective public resources, the system would be able to access to their extra information at runtime by using SPARQL queries.

# References

[1] P. Brusilovsky, A. Kobsa, and W. Nejdl, *Adaptive Web*, Germany: Springer-Verlag Berlin Heidelberg 2007, 2007.

[2] D. Maltz and K. Ehrlich, "Pointing The Way: Active Collaborative Filtering," *Proceedings CHI '95*, Denber: 1995.

[3] B.M. Sarwar, G. Karypis, J.A. Konstan, and J.T. Riedl, "Application of Dimensionality Reduction in Recommender System - A Case Study," *In ACM WebKDD Workshop*, 2000.

[4] X. Amatriain, J.M. Pujol, and N. Oliver, "I like it... I like it not: Evaluating User Ratings Noise in Recommender Systems," *User Modeling and Adaptive Hypermedia (UMAP'09)*, Trento, Italy: 2009.

[5] B. Mobasher, " Data Mining for Web Personalization," *Adaptive Web*, Berlin: Springer-Verlag Berlin Heidelberg 2007, 2007, pp. 90-135.

[6] R. Burke, "Hybrid Web Recommender Systems," *Adaptive Web*, P. Brusilovsky, A. Kobsa, and W. Nejdl, 2007, pp. 377 - 408.

[7] P. Lamere and O. Celma, *Tutorial on music recommendation*, ISMIR'07, 2007.

[8] G. Adomavicius and A. Tuzhilin, "Personalization Technologies: A Process-Oriented Perspective," *Communications of the ACM*, vol. 48, 2005, pp. 83-90.

[9] P. Brusilovsky and E. Millán, "User Models for Adaptive Hypermedia and Adaptive Educational Systems," *Adaptive Web*, 2007, pp. 3 - 53.

[10] D. Kelly and J. Teevan, "Implicit feedback for inferring user preference," *ACM SIGIR Forum*, vol. 37, 2003, pp. 18-28.

[11] E. Rich, "Users are individuals: individualizing user models," *International Journal of Man-Machine Studies*, vol. 18, 1983, pp. 199-214.

[12] L. Ardissono, L. Console, and I. Torre, "An adaptive system for the personalized access to news," *Ai Communications*, 2001, pp. 1-19.

[13] J. Kay, "Lies, damned lies and stereotypes: pragmatic approximations of users," *Proc. of Fourth International Conference on User Modeling*, MITRE, 1994, pp. 175-184.

[14] M. Balabanovic and Y. Shoham, "Fab: content-based, collaborative recommendation," *In: Communications of the ACM*, vol. 40, 1997, pp. 66-72.

[15] F. Asnicar and C. Tasso, "ifWeb: A Prototype of User Model-Based Intelligent Agent for Documentation Filtering and Navigation in the World Wide Web," *In: Proceedings*

*of the 6th International Conference on User Modeling*, Chia Laguna, Sardinia, Italy,: 1997, pp. 3-11.

[16] H.K. Farsani and M. Nematbakhsh, "A Semantic Recommendation Procedure for Electronic Product Catalog," *Journal of Applied Mathematics*, vol. 3, 2006, pp. 86-91.

[17] M. Gawinecki, Z. Vetulani, M. Gordon, and M. Paprzycki, "Representing Users in a Travel Support System," *Proceedings of the Intelligent Systems Design and Applications*, Wrocław, Poland: 2005.

[18] P. Liu, N. Guihua, and D. Chen, "Exploiting Semantic Descriptions of Products and User Profiles for Recommender Systems," *Proceedings of the 2007 IEEE Symposium on Computational Intelligence and Data Mining (CIDM 2007)*, 2007, pp. 179-185.

[19] F. Tanudjaja and L. Mui, "Persona: A Contextualized and Personalized Web Search," *HICSS*, 2002.

[20] G. Salton and M.J. McGill, *Introduction to Modern Information Retrieval*, McGraw-Hill, 1986.

[21] A. Stefani and C. Strapparava, "Personalizing Access to Web Sites: The SiteIF Project," *Proceedings of the 2nd Workshop on Adaptive Hypertext and Hypermedia*, Pittsburgh, USA: 1998.

[22] J. Fink and A. Kobsa, "User Modeling for Personalized City Tours," *Artificial Intelligence Review*, vol. 18, 2002, pp. 33-74.

[23] A. Sieg, B. Mobasher, and R. Burke, "Learning Ontology-Based User Profiles: A Semantic Approach to Personalized Web Search," *IEEE Intelligent Informatics Bulletin*, vol. 8, 2007.

[24] D. Godoy and A. Amandi, "User profiling in personal information agents: a survey," *The Knowledge Engineering Review*, vol. 00, 2005, pp. 1-33.

[25] G.I. Webb and M. Kuzmycz, "Feature Based Modelling: A methodology for producing coherent, consistent, dynamically changing models of agents' competencies.," *Methodology*, vol. 5, 1996, pp. 1-32.

[26] J.S. Breese, C. Kadie, and D. Heckerman, "Empirical Analysis of Predictive Algorithms for Collaborative iltering," In Proceedings of Fourteenth Conference on Uncertainty in Artificial Intelligence, M. Kaufmann, 1998, pp. 43-52.

[27] J. Kolodner, *Case-based reasoning*, San Francisco: Morgan Kaufmann Publishers Inc., 1993.

[28] S. Schmitt and R. Bergmann, "Applying case-based reasoning for product configuration in mass customization environments," *Twelfth International Bled Electronic Commerce Conference*, Slovenia: 1999.

[29] D. Billsus and M.J. Pazzani, "A Hybrid User Model for News Story Classification," *7th International Conference on User Modeling (UM'99)*, Springer, 1999.

[30] P. Resnick, N. Iacovou, M. Suchak, P. Bergstrom, and J. Riedl, "GroupLens: An Open Architecture For Collaborative Filtering Of Netnews," *Proceedings of the 1994 ACM conference on Computer supported cooperative work - CSCW '94*, Chapel Hill, North Carolina: ACM Press, 1994, pp. 175-186.

[31] G. Linden, B. Smith, and J. York, "Amazon.com Recommendations: Item-to-Item Collaborative Filtering," *IEEE Internet Computing*, vol. 7, 2003.

[32] L.H. Ungar and D.P. Foster, "Clustering Methods for Collaborative Filtering," *Proceedings of the 1998 Workshop on Recommender Systems*, Menlo Park, California: AAAI Press, 1998, pp. 114-129.

[33] B. Krulwich, "LIFESTYLE FINDER Intelligent User Profiling Using Large-Scale Demographic Data," *AI Magazine (AAAI)*, vol. 18, 1997, pp. 37-46.

[34] B. Mobasher, X. Jin, and Y. Zhou, "Semantically Enhanced Collaborative Filtering on the Web," *Lecture Notes in Computer Science*, 2004.

[35] P. Melville, R. Nagarajan, and R.J. Mooney, "Content-Boosted Collaborative Filtering for Improved Recommendations," *American Association for Artificial Intelligence AAAI-02*, 2002.

[36] R. Burke, "Hybrid Recommender Systems: Survey and Experiments," *User Modeling and User-Adapted Interaction*, vol. 12, 2002.

[37] G. Adomavicius and A. Tuzhilin, "Context-Aware Recommender Systems" , *Tutorial presented at the 2008 ACM Conference on Recommender systems*, Lausanne, Switzerland: ACM, 2008, pp. 335-336.

[38] M. Setten, S. Pokraev, and J. Koolwaaij, "Context-Aware Recommendations in the Mobile Tourist Application COMPASS," *Springer-Verlag*, 2004, pp. 235-244.

[39] F. Wang, "Content Recommendation Based on Education-Contextualized Browsing Events for Web-based Personalized Learning," *Educational Technology & Society*, vol. 11, 2008, pp. 94-112.

[40] G. Adomavicius, R. Sankaranarayanan, S. Sen, and A. Tuzhilin, "Incorporating contextual information in recommender systems using a multidimensional approach," *ACM Transactions on Information Systems (TOIS)*, vol. 23, 2005.

[41] T. Berners-Lee, J. Hendler, and O. Lassila, "The Semantic Web A new form of Web content that is meaningful to computers will unleash a revolution of new possibilities," *Scientific American*, 2001.

[42] M. Biezunski, S. Newcomb, and M. Bryan, "Guide to the topic map standards," *ISO/IEC 13250 Projects*, 2002.

[43] E. Peis, J.M. Morales-del-castillo, and J.A. Delgado-López, "Semantic Recommender Systems. Analysis of the state of the topic," *"Hipertext.net"*, vol. 6, 2008, pp. 1-9.

[44] S. Kim and J. Kwon, "Effective Context-aware Recommendation on the Semantic Web," *IJCSNS International Journal of Computer Science and Network Security*, vol. 7, 2007, pp. 154-159.

[45] C. Ziegler, "Semantic Web Recommender Systems," *Proceedings of the Joint ICDE/EDBT Ph. D. Workshop*, Springer, 2004, pp. 78-89.

[46] V. Challam, S. Gauch, and A. Chandramouli, "Contextual Search Using Ontology-Based User Profiles," *RIA2007*, 2007.

[47] S.E. Middleton, H. Alani, and D.C. Roure, "Exploiting Synergy Between Ontologies and Recommender Systems," *Semantic Web Workshop*, vol. 55, 2002, pp. 41-50.

[48] P. Shoval, V. Maidel, and B. Shapira, "An Ontology-Content-Based Filtering Method" , *International Journal*, vol. 15, 2008, pp. 303-314.

[49] C. Bizer, R. Oldakowski, R. Tolksdorf, M. Mochol, R. Heese, and R. Eckstein, "The Impact of Semantic Web Technologies on Job Recruitment Processes," *International Conference Wirtschaftsinformatik (WI'05), 2005*, Springer, 2006, pp. 17-26.

[50] O. Celma and X. Serra, "Foafing the music: Bridging the semantic gap in music recommendation," *Web Semantics: Science, Services and Agents on the,* vol. 6, 2008, pp. 250-256.

[51] Y. Blanco-Fernández, J.J. Pazos-Arias, A. Gil-Solla, M. Ramos-Cabrer, M. López-Nores, J. García-Duque, A. Fernández-Vilas, R.P. Díaz-Redondo, and J. Bermejo-Muñoz, "A flexible semantic inference methodology to reason about user preferences in knowledge-based recommender systems," *Knowledge-Based Systems*, vol. 21, 2008.

[52] C.M. Mackenzie, K. Laskey, F. Mccabe, P.F. Brown, R. Metz, and B.A. Hamilton, *Reference Model for Service Oriented Architecture 1.0*, OASIS Open, 2006.

[53] I. Schwab and W. Pohl, "Learning User Profiles from Positive Examples," *UM99 Workshop on Machine Learning for User Modeling*, Banff, Canada: 1999, pp. 1-6.

[54] C. Ziegler, S.M. McNee, J.a. Konstan, and G. Lausen, "Improving recommendation lists through topic diversification," *Proceedings of the 14th international conference on World Wide Web - WWW '05*, 2005, p. 22.

[55] C. Bizer, T. Heath, and T. Berners-Lee, "Linked Data - The Story So Far," *International Journal on Semantic Web and Information Systems (IJSWIS)*, 2009.

[56] G. Richards, *Cultural Tourism in Europe*, Wallingford: CABI, 1996.

[57] J.L. Herlocker, J.A. Konstan, L. Terveen, and J.T. Riedl, "Collaborative Filtering Recommender Systems," *ACM Transactions on Information Systems*, vol. 22, 2004, pp. 5-53.

[58] L. Ceccaroni, "*Ontowedss* - An Ontology-based Environmental Decision Support System for the Management of Wastewater Treatment Plants" PhD thesis. Departament de Llenguatges i Sistemes Informàtics. Universitat Politècnica de Cantalunya, 2001. (ISBN: 84-688-1569-1)

[59] C. Fellbaum, *WordNet: An Electronic Lexical Database (Language, Speech, and Communication)*, MIT Press, 1998.

# Appendix A - Software implemented

In this section, the detailed information about the revisions of the software employed for the development of the recommendation system and the Web-application in the tourism domain are presented.

## Recommendation system development

The main software that has been used for the development of the recommendation system is composed of the following programs:

- Ontology editor Protégé 3.4 with OWL plugin
- Eclipse IDE 3.4.2 for Java EE developers
- Glassfish v3 Enterprise Server
- JAX-WS RI 2.1 (nigthly version)
- Jena framework 2.5.7
- Jastor tool 1.0.4
- MySQL JDBC driver 5.1.7
- MySQL database server 5.1
- Java JDK 1.6.0_13

## Web application development

The main software that has been used for the development of the Web-application is composed of the following programs:

- Eclipse IDE 3.4.2 for PHP developers with Zend Debugger
- PHP 5 with XAMPP distribution
- Apache HTTP Server 2.2
- PHP SOAP Client library for web services (integrated with PHP 5)
- RAP framework 0.9.6