Semantic Web Rules and Ontologies for Developing Personalized Mashups

Aikaterini K. Kalou*, Dimitrios A. Koutsomitropoulos and Tzanetos Pomonis
High Performance Information Systems Laboratory (HPCLab), Computer Engineering and Informatics Dpt., University of Patras, Patras-Rio, Greece

Abstract. The current trends for the future evolution of the Web are without doubt the Semantic Web and Web 2.0. A common perception for these two visions is that they are competing. Nevertheless, it becomes more and more obvious that these two concepts are complementary. Semantic Web technologies have been considered as a bridge for the technological evolution from Web 2.0 to Web 3.0, the Web about recommendation and personalization. Towards this perspective, in this work we introduce a framework based on a 3-tier architecture that illustrates the potential for combining Web 2.0 and Semantic Web technologies. Based on this framework, we present an application of searching books from Amazon and Half eBay with a focus on personalization. This implementation purely depends on ontology development, writing of rules (for the personalization), and on creation of a mashup with the aid of Web APIs. However, there are several open issues that must be addressed before such applications can become commonplace. The aim of this work is to be a step towards supporting the development of applications which combine the two trends so as to conduce to the term Web 3.0, which is used to describe the next generation Web.

Keywords: Semantic Web, knowledge representation, ontology, rules, SWRL, personalization, Web 2.0, mashups, Web APIs, Web 3.0, 3-tier architecture

1. Introduction

The Semantic Web and Web 2.0 are two seemingly competing visions that dominate in Web research and development. It is our firm belief that the technologies and the core strengths of these visions are complementary, rather than in competition. In fact, both technologies need each other in order to scale beyond their own drawbacks, in a way that enables forthcoming web applications to combine Web 2.0 principles, especially those that set off notions such as usability, community and collaboration, with the powerful Semantic Web infrastructure, which facilitates the information sharing among web applications.

Recently, the term Web 3.0 came to be added in the glossary of Web and seems to describe the long-term future of the Web. By adding the Semantic Web to Web 2.0, we move conceptually closer to Web 3.0. The underlying technologies of the Semantic Web, which enrich content and the intelligence of the social web, pull in user profiles and identities, and must be combined for Web 3.0 to work [12]. Consequently, the incorporation of Semantic Web and Web 2.0 principles will conduce to the development of Web 3.0, the Web about personalization and recommendation.

Towards this direction, in this work we attempt to build a web application based on a 3-tier architecture (proposed by [32]), which combines basic principles of Semantic Web and Web 2.0, as mentioned above. We have named this application Books@HPClab, and at the rest of this paper, we refer to this application with this name. The implementation of Books@HPClab is focused mainly on the ontology development and on the mashup creation, since ontologies and mashups are the pillars for the Semantic Web and Web 2.0, respectively. A main characteristic of the application - and at the same time the primary benefit of Web 3.0 - is users’ personalization, which is implemented with the use of rules.

Users have the ability to search and find metadata and sell-offers for books which fit their personal preferences. These data are in fact mashed up from differ-
ent and heterogeneous sources on the Web, like Amazon and Half Ebay. The term *mashup*, one of the most popular Web 2.0 applications, is used to describe this heterogeneous combination of data and can be considered to have an active role in the evolution of Web 2.0. Book metadata are finally triplified, linked to their offers and kept in an ontology (*BookShop Ontology*). This makes the application’s presented information more reusable and effectively more sharable. At the same time, this allows us to easily open up this information to the *Linked and Open Data (LOD)* world, by providing a corresponding interface.

Users constitute autonomous entities for the application and a separate user profile is maintained for each of them, also in an ontology. These profiles are intended to capture user behaviour by keeping track of user preferences and their potential modifications. As a result, this kind of ontology-based user modeling allows the content of the application to be adapted to the profile of the user based on a set of SWRL rules.

The following text is organized in seven sections. In section 2, we start by providing some broad definitions and discussing the concepts of Semantic Web, Web 2.0, Web 3.0. Linked Data, putting a special focus on personalization based on semantic technologies. At the same time, we identify some major design decisions in our implementation. Furthermore, in section 3, we discuss related work and the theoretical background of the research area. In section 4, we describe in detail our application, its components, its architecture, the developed ontology, the rules for personalization and the current technological limitations. In section 5, we explain step-by-step the entire process of collecting data from online bookstores. Next, section 6 outlines some indicative application scenarios in order to illustrate the features and the functionality of the application. Based on our implementation, in section 7, we discuss and advise about some key issues that are to be met in similar ventures and identify potential future directions. Finally, section 8 summarizes our conclusions.


2. Background

For years, the World Wide Web has constituted a unique technological phenomenon with regard to the number of users and the vast available amount of information. At the same time, the lack of common terminology, organization and semantics of data, which are shared on the Web, induces difficulty at data exchange and processing. The necessity for development of the Web became imperative as long as never. As mentioned above, two distinct answers have emerged for the question “What is the next stage in the development of the Web?” from two different research groups: one, unmistakably Berners-Lee’s, advocates the Semantic Web, and the other, easily recognisable as O’Reilly’s, supports the so-called Web 2.0 [15].

Nowadays, the term Web 3.0 is used to describe the evolvement of the Web for the next decade (2010-2020) [23]. Web 3.0 will surely incorporate Semantic Web notions and Web 2.0 principles, but it will also include some more sophisticated concepts like artificial intelligence, as other researchers believe.

2.1. Semantic Web

The Semantic Web, proposed by Tim Berners-Lee (inspirator of the Web) and propagated by the World Wide Web Consortium (W3C), is not a separate Web but an extension of the current one, in which information is given well-defined meaning, better enabling computers and people to work in cooperation. In this context, the Web content will be presented in a form that is more easily machine-understandable, which means that machines will become much better able to process-to “understand” and to integrate the information that they simply display at present [8].

Ontology is without doubt the "backbone" of the Semantic Web. The most simple and easily understandable definition of the term *Ontology* is proposed by W3C: “An ontology formally defines a common set of terms that are used to describe and represent the basic concepts in a domain and the relationships among them". Since the Semantic Web relies heavily on formal ontologies, many languages for ontology expression such as SHOE, XOL, RDF, DAML and OWL and many tools for ontology engineering development have appeared, from WebOnto to Protégé. At the core of the Semantic Web architecture stack, as proposed by Tim Berners-Lee et al. [8], appears *reasoning*, the key component for the derivation of facts unexpressed explicitly in an ontology. Semantic reasoners and frameworks such as Pellet, FaCT++, Jena and others are pieces of software which implement the aforesaid task.

Note also that many ontology languages have restrictions on their expressiveness for the sake of decidability. One way to address this problem is to extend these languages with some form of rule languages. *Rules* have practical implementation in many domains, such as Engineering, Commerce, Law,
Medicine, Internet and so on. Especially for OWL, the extension of OWL DL with Horn-like rules gives an extended language, named SWRL (Semantic Web Rule Language), which is intended to be the rule language of the Semantic Web [18][4]. The unrestricted combination of formalisms leads to a very expressive formalism, which is, at the same time, unsurprisingly undecidable. To overcome this risk, a safety condition is imposed on SWRL rules. This safety condition is known as “DL-safety” and such rules are called “DL-safe SWRL rules” and are used in the context of our work.

Linked Data are also included among the means to reach the vision and accomplish the scope of the Semantic Web. In summary, the concept of Linked Data focuses on the creation of typed links between different data sources by using the Web. Linked Data, from the technical aspect, are data with explicitly defined meaning, published on the Web in a machine-readable format (data in RDF), which are linked to other external data sets and can also be linked to from external data sets. The result of this connection is the Web of Linked Data. ‘Linked Data principles’, a set of best practices by Berners-Lee, define how to publish and connect data on the Web, so that all published data becomes part of a single global data space.

The LOD project is a community effort which intends to bootstrap the Web of Linked Data and maintain the LOD cloud diagram, which includes all data sets that have been published in Linked Data format. It should be noted that the existent diagram contains data from many diverse domains, such as media, publications, life sciences, geographic data etc. Finally, a significant value of Linked Data is that of enabling new types of applications [17].

All these Semantic Web technologies constitute an environment capable of enabling efficient and personalized applications, since the idea of personalization is embedded within the very nature of the Semantic Web and set the state of the art [3].

2.2. Web 2.0

On the other side, the supposedly competing vision for the future of the Web is the Web 2.0, which is neither a new version of the Web nor a protocol, but concerns the changes in the way of using existing technologies. The main characteristic of Web 2.0 is that it provides great value to the end user Web utilization, by promoting notions such as interaction, dynamic content, collaboration, contribution, community and social computing. These concepts have led to the development and evolution of web-based communities and hosted services such as mashups, blogs, wikis, RSS, tagging, social bookmarking and social networking [28].

In particular, the development of mashups is full and anodic and many categories of mashup appear continuously, as mentioned thoroughly in ProgrammableWeb.com, a U.S. mashup portal site [28]. A mashup is a Web-based application that is created by combining and processing on-line third party resources, that contribute with data, presentation or functionality [22]. Even though Web 2.0 is often characterized by the aspect of community (social character) and user opinion expression (wikis, blogs), the high interest that academia and industry have shown in mashups, makes them valuable. The research on development tools to support mashups, the research projects in many countries, the presentation examples of mashups at international academic meetings and the effectiveness of mashups in the business field are some of their important aspects [16][22].

In our application, Web 2.0 is represented by the notion of a mashup that uses Web APIs, as the most common technique for mashups' creation. Web APIs define a generic set of methods and functionalities which enable applications to call remote procedures and to exchange data by passing well-defined messages from a web service in a transparent manner. The set of messages delivered to a web service is named request and the set of messages delivered to clients is named response. In order to achieve the interaction or communication with a web service, protocols such as SOAP (Simple Object Access Protocol) and REST (Representation State Transfer) are used. For the basic unit of communication, i.e. the messages (request and response), JSON and XML formats are used. For our application, we have chosen Amazon Web API and EBay API, since Amazon and Half.com are among the 20 top book sites, as mentioned in http://books.nettop20.com.

2.3. Web 3.0

According to the most recent research, the Web evolution is expected to take us into the era of Web 3.0, which adds the properties of the Semantic Web, better enabling computers and people to work in corporation, to the bidirectional knowledge exchange structure of Web 2.0 [21], as depicted in Figure 1.

It is anticipated that Web 3.0, the “intelligent web” as it is usually named in the literature, will address
the lack of structure and organization, as it has come up by Web 2.0 technologies, by linking information from disparate sources and systems. Thus it will make the web even easier to use, more efficient and more valuable to its users [6].

Generally, the main features emphasized by the concept of Web 3.0 are the following three:

- The capability of obtaining contextual information from a web search.
- The ability to obtain information drawn from a variety of previously incompatible applications or sources.
- The engagement of all types of devices and machines in the data creation, data use, and communication process that informs our daily lives, our work, our business.

The result of such evolution will be the creation of smarter, more efficient web programs that could drastically reduce the time it takes to compile and post information to the Internet and the time it takes users to search for it. The key for Web 3.0 is efficiency and personalization [6].

Towards the concept of Web 3.0, problems may emerge when there are alternative techniques by the competing visions of Web 2.0 and Semantic Web, which can serve the same purpose. Take for example Linked Data and Web APIs, which both can be used as primary data sources for mashup development.

Besides technical details, there is a major conceptual difference between Web APIs and Semantic Web. The available data items via Web APIs are not assigned with globally unique identifiers. Therefore, it is not possible to set links between items in different data sources. Applications based on Linked Data, can draw on an unbounded, global data space. Unlike Web 2.0 mashups that can use a part or total of a fixed set of data sources, Linked Data mashups can discover new data sources at runtime by following data-level links, and can thus deliver more complete answers as new data sources appear on the Web.

Therefore, Linked Data technologies can contribute to connecting the different data silos that currently exist on the Web back into the single global information space. The Web of Linked Data gives also to anyone the opportunity consume and aggregate information without having to fully understand its schema [17].

Nevertheless, in our work data integration is based on the use of Web APIs. The main reason is that Amazon and Half EBbay resources, where our data are

---

Fig. 1. Web Evolution.
drawn from, are only accessible via Web APIs. In addition, the methods to aggregate data by using APIs are generally considered more straightforward and have grown more technically mature, as compares to data linking [16].

Even though our implementation may not consume Linked Data, data ingested from the APIs are triplified in web ontologies, where the application’s logic is based upon. As a step ahead, we actually make these data readily available to the LOD cloud by exposing them also as Linked Data, thus evidently supporting third-party applications to “plug-in” and meaningfully reuse this information.

2.4. Personalization based on Semantic Technologies

Due to the increasing amount of available informational resources, the one-size-fits-all paradigm seemed to be inadequate to fit the users’ needs. The appropriate solution to this problem was to bring the user’s needs into the center of interaction processes and provide individually optimized access to Web data and information.

Adaptive Hypermedia (AH) and Web mining constitute the two major approaches to personalization, since the early days of World Wide Web [3]. Particularly AH is a research area that can utilize Semantic Web technologies in an attempt to address some of its drawbacks, which include limited degree of interoperability and reusability, difficulties in the acquisition of model information and the lack of control and transparency of the system’s adaptive behaviour [38].

Generally, existing approaches to personalization are based on three different axes:

- **Adaptivity dimensions**: The adaptive behavior is realized by either collaborative-filtering (identifies content found relevant by similar users) or content-based filtering (exploits similar content to identify relevant resources for a certain user) or combination of these two.

- **Representation formalism**: This is how the aspects of the adaptive system, such as the content, the users, the system itself, are represented within a specific formalism.

- **Exploitation techniques**: Techniques which are used to perform the underlying logic of adaptation.

With respect to adaptivity dimensions, content-based filtering and collaborative filtering are two basic categories of adaptation systems, which differ from an architectural and algorithmic point of view [5]. A content-based filtering system selects items based on the correlation between the content of the items and the user’s preferences. It makes recommendations by comparing a user profile with the content of each document in the collection [30]. On the other hand, in collaborative filtering, the system not only uses the profile for the active user but also maintains a database of other users’ profiles, so as to have the possibility to detect user-to-user similarities [35].

Many studies have pointed out the advantage of the use of semantic technologies in the last two aforementioned axes of personalization approaches. The use of ontologies, in order to represent the different adaptivity dimensions, increases meaningfully the interoperability and the reusability of model information.

Especially for the user model, many efforts have been made to standardize the information about the user in terms of an ontology, such as FOAF, LOM etc. Therefore, ontologies can play many potential roles to support user modeling: i) providing a mechanism for reasoning about the users ii) supporting scrutability for aiding the user in getting a better understanding of the domain and iii) defining a set vocabulary to enable metadata annotation of the content [20].

In the axis of exploitation techniques, rules are employed to represent the adaptation logic. Similar to ontologies, which substitute vectors, matrices, Bayesian networks etc as formalisms for the representation of different aspects, the usage of rules, as the logic underlying the adaptation, replaces adopted techniques from statistics and machine learning, for the same purpose. Adaptation by using rules is accomplished in a more transparent manner to the users which can better inspect and understand the entire process.

Besides the aforementioned contributions, semantic technologies also help to solve the challenging problem of developing open-corpus AH systems [3]. In a traditional closed-corpus AHS, all the documents and the relationships between them are known at the design time. However, this no longer holds when considering an open corpus of documents, and mashups can be considered to fall into this category. In this case, Semantic Web can offer a certain context-of-use by allowing programs to reason about content and its meaning [11].

In terms of the above, this work introduces a content-based filtering adaptation system; the representation formalism adopted is offered by web ontologies and OWL semantics; and the exploitation logic is captured by a rule-based model.
3. Related Work

Our work focuses on two distinct axes: i) the integration of Semantic Web and Web 2.0 so as to develop a prototype Web 3.0 application and ii) the usage of ontologies and rules, pillars of semantic technologies, in order to achieve more efficient personalization.

The concept of combining Semantic Web technologies and Web 2.0 has been investigated from various different angles. For example, semantic blogging constitutes an effort to enhance blogs with semantic, machine-understandable metadata and has attracted quite a lot of interest. We report indicatively some scenarios of semantic blogging such as the semiBlog editor proposed in [26], the Semblog platform in [29], the prototype semantic blogging system OntoBlog [36] and so on.

Towards the integration of mashups with the notion of the Semantic Web [14], many attempts have also been made, such as: (i) semantic mashup for tourism proposed by [42], (ii) semantic map mashups, (iii) semantic mashups for several scenarios in life sciences [7], [26], and (iv) the use of mashup architecture in more sophisticated tasks, like business processes [1].

Traditional personalization methods, as discussed in section 2.4, are already widely used. Semantic personalization on the other hand has recently come to attract research focus [40].

Similar to our approach, most of these techniques focus on the ontology-based establishment and enrichment of user profiles either on FOAF [2] or on custom methodologies [39][33] and then employ content-based filtering.

In [24] this approach is further extended by utilizing user ratings for items; however issues like the new user problem or impeding document ratings remain unresolved.

Social networking data and rating records can be utilized in collaborative filtering contexts [25]. The authors report improvement in scalability and accuracy of personalization, although incompatible to the Semantic Web.

In the following we single out and discuss a couple of semantic personalization approaches that are most closely related to our work.

Blanco-Fernández et al. [10] present a procedure to automatically compose interactive applications that provide personalized commercial functionalities to the users, gathering content from multiple sources and with a back-end of semantic web services. The procedure is driven by SWRL rules and similarity metrics based on semantic reasoning.

In addition, Wang et al. [41] propose a tourism system based on an ontology. This system allows integration of heterogeneous online travel information and recommends tourist attractions to a user based on the Bayesian network.

The above efforts differ in two distinct dimensions, namely the representation formalism and the exploitation technique used, as discussed in Section 2. The tourist information and the content of advertised items in [41] and in [10] respectively, are both represented as basic ontology components. In [41] besides the travel ontology, a user ontology is also constructed. In contrast, the user profile features in [10] are captured in a data structure. Conversely, [41] employs a Bayesian network to perform recommendation, while in [10] Semantic Web rules decide the suitable sources for a given user.

In our work, an ontology is being used as the representation formalism both for the content sources and for user profiles. Furthermore, personalization tasks are carried out solely by the inference power of rules. Thus, all beneficial characteristics of Semantic Web personalization can be combined into a single unified approach. To our knowledge, there is no other intelligent mashup that performs user modeling and adaptation relying purely on the Semantic Web stack.

Finally, it should be mentioned that there is also another Book Mashup [9] which integrates book data from Web APIs into the Semantic Web. However, it appears to lack personalization features.

4. System Architecture and Design

In this section, we describe the 3-tier architecture, which constitutes the base of our application and may also underlie many sophisticated Web 3.0 applications, such as semantic wikis, semantic portals, semantic mashups etc. Then, we focus on the design of the ontology and the philosophy behind our personalization rules. Finally, some concrete implementation details are given.

4.1. Architecture overview

As illustrated in Figure 2, the layers of the architecture can be distributed both at the logical and physical level: (i) the front-end layer, (ii) the application logic layer and (iii) the knowledge management layer.
The knowledge base of our application resides on the lower part of the architecture, namely the knowledge management layer. This is represented by our core ontology, BookShop, which includes rules for personalization and the individuals (instances of classes).

Then, the middle layer of the 3-tier architecture, the application logic layer, is responsible for managing and uploading the ontological model of the lower layer. The operation of this layer is implemented with the use of the OWL API [19] that serves also as a means of communication with the lower, knowledge management layer. The middle layer is also responsible for the ontological data loading, coming through the front end and based on the ontological schema of the back end. Ontological data include all the instances of the four main classes User, Book, Author and Offer and are stored in an OWL file.

Finally, the upper layer of the proposed architecture is in general the layer which enables users to fully interact with the knowledge base, by adding, eliciting and incrementing the ontological data model. This layer may also interact with web services, programs, scripts, and other interoperability interfaces. It performs the following tasks:

- Communication/Interaction of the application with the Amazon and Half EBay Web APIs.
- User Interface of the application with the ability of interaction with the knowledge base, presentation and navigation of semantic data.

4.2. BookShop Ontology

In this subsection, we describe how we collected the structural information and designed the core ontology of our application.

Although there are many formal methodologies for developing ontologies [13] (TOVE, KACTUS, METHONTOLOGY, On-To-Knowledge, etc), we preferred a more simplified and intuitive approach.
for this purpose [27]. In short, it is an iterative approach to ontology development, starting with a rough first pass at the ontology, then revising and refining the evolving ontology and filling in the details. This iterative designing process will likely continue through the entire lifecycle of the ontology.

To design our book ontology, we took into account the kind of metadata offered by Amazon and Half Ebay responses. Our design process has resulted in the core ontology BookShop and part of this is shown in Figure 3. BookShop contains four main classes: Book, Author, Offer and User.

The class Book is enriched with relations such as title, publisher, dimensions, ISBN, publication year, number of pages, format, rating, images in various sizes and a URL – corresponding to the Amazon online bookstore – so as to describe the instances of this class. All these relations are captured in the ontology as datatype properties.

For our purposes, keeping record of each author’s name and surname suffices. Therefore, we decided to define the class Author as a subclass of the class Person, which is included in the FOAF (Friend of a friend) ontology. FOAF is an ontology, which provides a unified way to describe persons, expressing their interests, their activities and their relations to other people and objects. The FOAF properties foaf:surname and foaf:firstName have been used to model author last and first name.

Items for sale on Amazon and Half Ebay can be sold by more than one seller for different prices and in different conditions (‘New’ or ‘Old’). Thus, any item – any book in this case – is associated with an offer. An offer is a combination of price, condition and vendor/seller. Therefore, to find a book’s price, we have to get the offers made by the vendors selling the book on online bookstores. The concept of an offer is represented by the class Offer. Datatype properties such as Condition, Price and Origin express the price, the condition of the offered book and the seller URL in these bookstores, respectively.

The class User is meant to express user profiles. We capture the preferences of each user in this class, such as preferable condition, preferable rating, pref-

![Fig. 3. BookShop Ontology.](image-url)
erable publication year and preferable maximum price (preference criteria). All this data about users are represented as datatype properties.

The User class and related properties in the ontology reflect a snapshot of the preferences for each register user. Changes in user behavior are easily captured by allowing the users to interact with the ontology, through the “settings” menu option that effectively allows the update or modification of their preferences.

Relationships between instances of the BookShop ontology are represented by object properties that express relations between instances of two classes. In this context, a Book must have at least one Author (hasAuthor and inversely isAuthorOf) and there is at least one Offer for a Book (isOfferOf and inversely hasOffer).

Object properties that link an instance of the class Book to an instance of the class User and inversely, are defined to express the user’s preference for a book depending on which preference fields of the user's profile are covered. Then, a set of DL-safe SWRL rules is responsible for actually populating these properties. For example, the object property prefersBookbyCondition would relate a user with books being in a condition the user prefers. There are also properties for the case when more than one preference criteria are met. For example, prefersBook_byRate2 would hold in case two criteria are satisfied together (no matter which).

4.3. Rule-based Personalization

In order to obtain efficient personalization for our application, a set of DL-safe rules was written, using SWRL. These rules “match” user’s preferences (user profile) with the features of books, which are returned as a result from the search in Amazon and Half EBay web services. These “personalization rules” essentially distinguish those books, which satisfy user’s preferences from the entire set of books after the searching process. Initially, four rules were written to check the satisfiability of each preference criterion separately. Take for example the case of a rule about preferred book condition. The SWRL description for this rule is depicted in Figure 4.

In addition, a SWRL rule was defined in order to associate a user with all books having his/hers preferred max price, as in Figure 5. The rule about a user, which prefers books with rating equal or greater than a certain degree, is described in Figure 6. Finally, user preference for a book published within a certain publication year can be illustrated with the rule in Figure 7.

---

**SWRL Rule #1**

Book(?y) AND Offer(?z) AND User(?x) AND isOfferOf(?z, ?y) AND BookCondition(?z, ?condition) AND prefersCondition(?x, ?preferred_condition) AND swrlb:equal(?condition, ?preferred_condition) → prefersBookbyCondition(?x, ?y)

---

**Fig. 4. Rule for book condition.**

**SWRL Rule #2**

Book(?y) AND Offer(?z) AND User(?x) AND isOfferOf(?z, ?y) AND OfferPrice(?z, ?price) AND prefersMaxPrice(?x, ?max_price) AND swrlb:lessThanOrEqual(?price, ?max_price) → prefersBookbyPrice(?x, ?y)

---

**Fig. 5. Rule for book max-price.**

**SWRL Rule #3**


---

**Fig. 6. Rule for book rating.**

**SWRL Rule #4**

Book(?y) AND User(?x) AND PublicationDate(?y, ?date) AND prefersPublicationDate(?x, ?preferred_date) AND swrlb:equal(?date, ?preferred_date) → prefersBookbyPublicationDate(?x, ?y)

---

**Fig. 7. Rule for book publication year.**

For the case where two or three or four preference criteria are satisfied together, we wrote more rules so as to check the number of satisfied criteria. An example of such a rule is in Figure 8. There are also more rules to check all possible combinations.

**SWRL Rule #5**

Book(?y) AND User(?x) AND prefersBookbyCondition(?x, ?y) AND prefersBookbyPrice(?x, ?y) AND prefersBookbyRating(?x, ?y) AND prefersBookbyPublicationDate(?x, ?y) → prefersBook_byRate3(?x, ?y)

---

**Fig. 8. Rule for three preference criteria.**

As an example, Rule #5 can be translated in natural language as follows: “If a user has defined at his profile that prefers books rated equal or greater than 3.5, cost equal or less than 40.00 USD and published in 2000, then a book matching all these conditions would be linked to this user via the object property prefersBook_byRate3".
4.4. Implementation Issues

The interface and the functionality of the application are implemented, as it is usual for web applications, in PHP and in HTML. For the ontology development, we use OWL in order to ensure the maximal possible expressiveness and the efficient support of reasoning. The definition of rules is implemented in SWRL. Protégé 4 (http://protege.stanford.edu) has been selected as the appropriate ontology environment for the ontology and rules development.

We make use of Pellet (http://clarkparsia.com/pellet), an open-source Java based OWL reasoner, which guarantees sound and complete OWL reasoning. The special feature of Pellet is its support for reasoning with DL-safe rules which makes it particularly suitable for our needs.

OWL API and Pellet are Java applications. On the other hand, the functionality of Books@HPClab is implemented in PHP. The problem was how these three components of the whole application could "communicate" and "collaborate" with each other, since they are implemented in different programming languages. One way to overcome this problem is to use the PHP/Java Bridge (http://php-java-bridge.sourceforge.net/pjb/), an implementation of a streaming, XML-based network protocol, which can be used to connect a native script engine, for example PHP, Scheme or Python, with a Java virtual machine.

Finally, since our application handles many XML documents (OWL ontology and XML responses from Amazon and eBay APIs), we make use of XSLT and XML DOM for this purpose.

5. Collecting Data from Bookstores

In this section, we review the process of searching data about books from the web data sources, in other words application’s interaction with Amazon and Half EBay Web APIs. Whenever the user sends a searching call, the searching process starts to query data from Amazon Web Services (AWS), and especially from the US E-Commerce Service (ECS). In order to extract the appropriate data for our application, we choose the ItemSearch operation, among the set of available ECS operations.

A request may return many thousands of items in a response. Returning all these results at once may be inefficient and impractical. In order to alleviate this, we combine all these files in a single XML file using DOM XML in PHP. This file is further processed by an XSLT in order to rule out redundant data that are not useful in our implementation. For example, information, such as images height and width, number of used, new, collectible or refurbished books, has not added value for our application.

Once our application completes the search process at Amazon, it starts searching Half Ebay; for each book returned by Amazon, we find additional offers that may be available at Half Ebay. We use the eBay Shopping Web Services and particularly, the FindHalfProducts operation. The interaction with the eBay Shopping API is based also on the REST-protocol and the exchange of URL requests and XML files-responses.

![Sample record of the XML file.](image)

When the searching process at the Half EBay is complete, results are once more unified into a single XML file (Fig. 11). In order to do this, we actually append more Offer sub-elements for each Book ele-
ment in the Amazon’s results document. Finally, the elements of this XML file are converted into an OWL file with individuals, matched against our BookShop ontology schema. An excerpt of the final OWL file is shown in Fig. 12.

A diagram, that depicts the data flow upon interaction with the Web APIs, appears in Figure 13.

6. Functionality and Usage

In this section, we demonstrate the functionality of the application. Moreover, we outline an indicative usage case in order to point out its capabilities and features.

6.1. Functionality

The first page of the application includes two choices for the visitor, “New User” and “Registered User”. New users are presented with a completion form, which includes fields such as Book Condition, Maximum Book Price, Publication Year and Maximum Book Rating, thus forming the user profile.

After successful authorization, the user is directed to the main page of the application, which includes a search form.

For efficiency and demonstration purposes there is also the choice to select between “search on the fly” and “prefetched search”, where the user interacts with a prefetched dataset.

The search form consists only of a “Search” button and a text field, where the user types the keyword or the key-phrase to initiate the book searching process.

When searching ends, results are imported into the core BookShop ontology as individuals. Next, the ontology is classified by the reasoner and the rules, that would determine how user preferences are matched, are fired against the ontology. Search results are ranked based on the rules outcome, as shown in Figure 14. In particular, the more criteria a Book satisfies (the more preference rules it triggers), the higher it appears in the results.

Each table’s row includes information, such as an auto-incrementing number, the book’s title and a number of exclamation marks which express the number of satisfied criteria. Each book’s title is a link and clicking it, a page appears with all the available features of the specific book, like Title, Author, ISBN (Figure 15). The title and image of the book are links to its “official” page at Amazon. Finally, the Offers for this book are presented in a table. Following the Search link (see Figure 14) the user is taken back to the main page in order to initiate a new query.

Finally, book information are made available as Linked Data in RDF, where books’ and offers’ links are resolvable within our application or within their original context, respectively (Fig. 16). This is also true for the search results list, which is exposed as RDF along with the inferred book rankings, for other applications to be able to consume and reuse.

Fig. 13. Diagram of communication with Web APIs.
6.2. An Example Usage Case

This subsection contains a brief example of Books@HPCLab personalization features, so as to demonstrate that different users can be treated differently, even though they are searching for the same key-phrase. We shall consider two users, Mary and George. According to Mary’s profile, she prefers books that were published in 2009, their price does not exceed 15.00 USD, their condition is “New” and their rating is at least 4. On the other hand, George’s “best” books should be in “Old” condition, their price...
should be lower than 45.50 USD, have been published in 2000 and their rating should be over 4.5.

Mary and George are searching for books with the topic Semantics, so they are typing in the text box of the search form the same keyword; "Semantics". At this moment, the application gets a lookup call, so it decodes the keyword and starts to query the original data sources (Amazon and Half Ebay). When the collection data process is complete, a XML file with 100 book nodes is returned in both cases, with the same content, as it was expected.

In the case of George, the procedure goes on as follows:

- Our DL reasoner, Pellet, finds that, for the books returned by the query, George’s profile matches the conditions of those SWRL rules which apply when two preference criteria are satisfied together (Subsection 4.3).
- Next, Pellet finds the applicable SWRL rules for George, which satisfy just one of the four preference criteria. None of the other SWRL rules is applicable for George.

The total number of books, which are shown to George is 74, therefore there are 26 books returned from the collection process, which satisfy none of George's preference criteria. From the 74 books, there are only two which satisfy exactly two criteria and the rest of them satisfy just one criterion at a time (see Table 1).

<table>
<thead>
<tr>
<th>Set 1</th>
<th>Number of returned books which satisfies exactly two criteria</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>2</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Set 2</th>
<th>Number of returned books which satisfies exactly one criterion</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>72</td>
</tr>
</tbody>
</table>

On the other hand, the procedure for Mary goes on as follows:

- The reasoner finds that Mary’s profile matches the conditions of those SWRL rules which check that three preference criteria are satisfied together (Subsection 4.3). So the instance Mary is being related to six instances of the class Book through properties prefersBook_byRate3.
- In addition, the rules, which check the simultaneous satisfiability of two preference criteria, are triggered.
- Next, Pellet finds the applicable SWRL rules for Mary, which satisfy just one of the four preference criteria. For Mary, there is no book that satisfies all four preference criteria at the same time.

Table 2 summarizes the total number of returned books and the number of books with the respective amount of satisfied criteria in the case of Mary.

Consequently, our application returns more books to Mary and more of Mary’s preference criteria are satisfied together, in contrast to George. Finally,
there are books, which are returned both to Mary and George, but in different order. Take for example the book with title "HTML Mastery: Semantics, Standards and Styling", which belongs to Set 2 (Table 2) in Mary’s case. In the case of George, this book appears in the set of books which satisfy only one preference criteria (Fig. 17 and Fig. 18).

Table 2.
Application response for Mary

<table>
<thead>
<tr>
<th>Set 1</th>
<th>Number of returned books which satisfies exactly three criteria</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>6</td>
</tr>
<tr>
<td>Set 2</td>
<td>Number of returned books which satisfies exactly two criteria</td>
</tr>
<tr>
<td></td>
<td>71</td>
</tr>
<tr>
<td>Set 3</td>
<td>Number of returned books which satisfies exactly one criterion</td>
</tr>
<tr>
<td></td>
<td>22</td>
</tr>
</tbody>
</table>

7. Discussion and Future Directions

In this work, we have described a concrete scenario of how Semantic Web technologies could enhance Web 2.0 tools and especially mashups. The resulting semantic mashup has been enhanced with personalization features based on the use of rule filtering as a recommendation technique.

The integration of semantics and, mainly, ontologies within Web 2.0 applications has also been studied before (Section 3). Light-weight ontology languages like RDF are often easier to handle, both for machines and for humans, than more complex formalisms like OWL. The fact that even simple machine-readable data can bring benefits, may lead someone to believe that the Semantic Web doesn’t need more expressivity than RDF. However, depending on practical uses, further expressivity is necessary to express complex knowledge. Examples of this set of applications include semantic mashups in life sciences or in business areas.

Our engineering effort to enhance our book mashup application with web semantics is primarily based on the development of an OWL ontology. In contrast to [9], which uses RDF, we use much more complex formalisms, in order to extend our mashup ontology with rules so as to achieve personalization.

The usage of rules, as an adaptation logic technique, achieves a substantial level of personalization in a manner more understandable and transparent to users. The power of ontologies in such personalized mashups lies in the interoperability and in the reusability of information as well as in a sound and expressive logic framework to perform fine-grained adaptations upon.

During the ontology creation process, it is worth noting that one must check first if there is any existing ontology that would fit application’s needs. In case of creating the ontology from scratch, careful design is necessary so as to increase the quality of semantic annotations and ontology reusability.

As a lesson-learned, we advise that it is not desira-
ble to specify too large a number of rules, in an attempt to check all the possible combinations of user preference criteria. Most of the time just a few rules are necessary and the rest of the task can be accomplished programmatically.

With the aid of Semantic technologies we can get from closed-corpus to open-corpus adaptive systems, where the set of documents is not known a priori and can be extended and expanded. In fact, the notion of open-corpus seems to be inherent in mashups, where the integration of disparate and evolving sources is involved.

In terms of performance, the manipulation and merging of large XML files using DOM can often be slow. A relational database or other persistent store for keeping XML responses can be used as an intermediary data source for populating ontology documents dynamically, as well as for caching purposes. Alternative exchange formats can also be investigated, such as JSON or the asynchronous processing of XML documents.

Finally, Books@HPCLab treats all rules with equal importance. It might be worth considering to put weights on rules in a fashion similar to [31], though not ranking rules in terms of certainty, but in regard to their individual value in a particular user’s profile. These weights could be user-defined and depend on user scores or implicitly adjusted, as a means to produce adaptations based on observable user behavior.

8. Conclusions

In this work, we have shown that Semantic Web and Web 2.0 can be complementary visions for the future of Web, rather than in competition. This was achieved by the development of an application which unifies successfully the philosophy of Web 2.0 applications (mashup) and the powerful technical infrastructure of the Semantic Web (ontologies and rules). Such Web applications are considered to be part of the next generation Web, usually referred as Web 3.0.

In particular, we presented a prototype web application, which integrates information from Web APIs, such as Amazon Web API and HalfEBay API, converting them to individuals of an ontology schema, forming a kind of semantic mashup for books. At the same time, information are elicited from their original sources, acquire semantic structure and exposed as Linked Data. The integration of rules into this semantic mashup serves as a viable and straightforward alternative towards personalization features.

Our literature survey reveals that a strong interest exists in filtering content ‘mashed up’ from various sources using semantics; this is mainly due to a growing need for eliciting value and meaning out of the plethora of unauthoritative and community-oriented information on today’s Web 2.0. To this end, the Semantic Web and the LOD paradigm appear to be worth investigating as almost natural means to achieve these goals. We believe that our dealing with
the various conceptual and technological challenges and the lessons-learned in this process may serve as a useful guidance towards developing tailored Semantic Web applications in the Web 3.0 framework.

References


