# Using Linked Data to build Recommender Systems*

Alexandre Passant Benjamin Heitmann Conor Hayes

Digital Enterprise Research Institute
National University of Ireland, Galway
Galway, Ireland
firstname.lastname@deri.org

## ABSTRACT

This paper describes how Semantic Web technologies and especially the Linked Open Data (LOD) project can be used to build a new generation of recommender systems. While most of the current recommender systems use private data sets, we show how to exploit the benefits of the LOD community effort to build recommender systems. By providing public, collaboratively created and semantically structured data, it enables cross-domain recommendations by providing data which can be exploited for different recommendation tasks and which is portable between systems, without changing the implementation of the recommendation algorithm.

The contributions of this paper are (i) an overview of the LOD community effort and of the data cloud generated by it, (ii) the description and evaluation of both a semantic-distance and a collaborative filtering approach based on Linked Data, (iii) a method for evaluating recommendations using existing categorical data and (iv) the description of a reference architecture for implementing cross-domain recommender systems using Linked Data.

## Categories and Subject Descriptors

H.3.3 [**Information Search and Retrieval**]: Information filtering; H.3.4 [**Systems and Software**]: Distributed system

## General Terms

Recommender Systems, Semantic Web, Linked Data

## Keywords

Linked Data, Semantic Web, Recommender systems, DBpedia, Semantic Distance, Architecture of Recommender Systems

*The work presented in this paper has been funded in part by Science Foundation Ireland under Grant No. SFI/08/CE/I1380 (Lìon-2).



## 1. INTRODUCTION

Currently, existing recommender systems use either private or public data sets. Private data sets, which are used by e.g. Amazon [22], are well structured, but usually only contain data from one domain and are not connected to data from other providers. Public data sets can be generated from mining the web [34], which results in an unstructured data set which is created in a collaborative way on a large scale. An orthogonal approach, is the exploitation of conceptual modeling for the recommendation task [24], by using ontologies to define the semantics of the data features. As described in this paper, we consider a different way to build recommender systems, by using data which combines advantages of both public and private datasets, i.e. being large-scaled, structured and using well-defined semantics. Such recommender systems can be built thanks to the ongoing and pragmatic Web of Data, that we consider being a subset of the Semantic Web vision, and especially via the Linking Open Data (LOD) project. We think this is the right time for the recommender system community to dive into such paradigms and this paper details how it might be done.

The rest of the paper is organized as follows: In Section 2, we describe current issues in recommender research and provide a context for the arguments in the following sections. Then, Section 3 explains how the standards of the Web of Data and the Linked Data principles allow the creation of structured data on the scale of the Web. Especially, we overview how it can be used to enable cross-domain recommendations by providing data which can be exploited for different recommendation tasks and which is portable between systems, without changing the implementation of the recommendation algorithm. In Section 4, we describe the implementation of two recommender systems based on these principles, i.e. a content-based and a collaborative-filtering approach that uses linked data. In Section 6 we propose a reference architecture to build such systems. Implementing this reference architecture allows recommender systems to use portable data for recommendation tasks from different domains without changing the recommendation algorithm. Then, in Section 5, we evaluate the results of the two recommendation approaches and we will then conclude the paper.

## 2. RECOMMENDER SYSTEMS SILOS

Research on recommender and personalization systems has tended to focus on centralized systems where the data representing user preferences, interactions and resource descriptions and usage are stored. This is entirely reasonable

as recommender algorithms tend to rely upon techniques closely related to machine learning, data-mining and information retrieval where access to centralized data storage is assumed. Furthermore, commercial organizations use their recommender data to develop valued-added services that give a competitive edge or reduce costumer churn. Understandably, there is little incentives to share such a valuable commodity. Thus, well established companies have tended to accumulate vast silos of user preference data while start-up companies and individuals have struggled to acquire enough data to get past the cold-start phase of the recommendation process.

Several research initiatives have been made to move recommender systems outside the centralised paradigm [17, 37]. In terms of content-based recommendation, a fundamental problem is matching domain-specific schemas. In distributed case-based recommenders, this problem is often ignored by assuming a common shared schema in order to concentrate on novel retrieval strategies [23, 29]. The conventional approach to the schema-matching problem is to employ a semi-automatic mapping process, either between domain schemas or between a domain schema and a reference schema. While the Database and Semantic Web communities have researched diverse approaches to mapping [15], this work has not made much impact on the recommender community.

In terms of social or collaborative recommendation, lack of agreed standards and industry support for data portability has meant that users cannot move preference data between domains or control the inferences that are made about them [10]. While this has stimulated research on privacy-enhanced recommender strategies [36, 21, 5], it has created obstacles for cross-domain recommendation, where the goal is to allow the user receive recommendations in different domain contexts [6, 16]. Bervkovsky et al. demonstrated a cross-domain approach to collaborative filtering by exchange and aggregation of user profile and neighbourhoods between domains [6]. Gonzalez et al. propose a model for integrating domain specific profile and generic profiles [16]. These approaches rely upon agreed user data formats and/or algorithms shared between domains. A similar problem is encountered in the active research area of hybrid recommenders [38, 13]. The issue of cross domain interoperability has tended to prohibit the combination of models developed in different domains.

In this paper, we will introduce the concept of how the Linked Data principles can open new and rewarding opportunities in recommender systems research. We do not propose to solve all the problems that we have identified above. However, our perspective is to present low cost techniques for making both content-based and collaborative recommendations across several domains using the growing networks of collaboratively produced, structured linked-data in the public domain. By focusing on developing algorithms for this environment, researchers in recommender systems can develop and evaluate systems that can be exploited immediately by organizations and individuals without having to accrue silos of content and usage data, enabling new ways to design such systems.

# 3. LINKED DATA PRINCIPLES

The current Web is mainly a Web of Documents connected together via untyped hyperlinks. While computers have excelled at extracting and making searchable syntactic information in documents, the Semantic Web vision aims to create Web of interoperable and machine-readable data, enabling better integration of resources between applications [8]. The move to bridge the gap between the Web of Documents and the Web of Data is underpinned by (1) common representation formats for data, typically in RDF [20], that provides a language to represent machine-readable statements in the form of `<subject> <predicate> <object>`; (2) common semantics to represent this data, using ontologies and related languages, i.e. RDFS [2] and OWL [1]; and (3) a means to query this information, using the query language SPARQL [3].

More recently, and based on this vision, a more pragmatic approach has emerged that focuses on providing structured and interlinked data rather than on high-level formal semantics and reasoning. This vision is termed the *Web of Data* and is based on the *Linked Data* principles defined by Tim Berners-Lee [7]: (1) use URIs as names for things; (2) use HTTP URIs so that people can look up those names; (3) provide useful information at each URI (4) include links to other URIs so that they can discover more things. For example, considering `http://dbpedia.org/resource/Johnny_Cash`, (1) it is defined as a URI to represent Johnny Cash[1]; (2) it is a HTTP URI, so that one can look it up in a browser, contrary to, for instance, isbn URIs defined as `isbn://`; (3) when accessing this URI, or *dereferencing* it, information is provided about the thing it represents: the name of the artist, his label, records, or other ontological and category information, as follows; (4) finally, it also provides link to other resources, so that it enable interactive browsing between various interlinked resources with Semantic Web browsers such as the Tabulator[2].

```
dbpedia:Johnny_Cash a dbpedia-owl:MusicalArtist ;
  dbpedia-owl:label dbpedia:American_Recordings ;
  skos:subject dbpedia:Category:Sun_Records_artists .
```

We will now provide an overview of the Linking Open Data initiative and examine how it offers an alternative to the data silo paradigm by providing semantically rich data that is interoperable and reusable.

## 3.1 Linked Data: Alternative to Data Silos

The Linking Open Data (LOD) project[3] started as a community effort in 2007, and helped to produce billion of RDF statements that are now published on the Web. The LOD objective is to interlink data that is available on the Web for free (e.g. under Creative Commons license), but which is locked in independent data-silos. This data is then linked and published in RDF format providing what is commonly known as the LOD cloud (Fig. 1).

To date, several knowledge bases have been exposed in RDF and interlinked together. Information from DBpedia[4] [4] (an RDF export of Wikipedia) is linked to geographical information from Geonames[5] as well as music-related information from MusicBrainz and its RDF export via DBtune[6].

[1]Different URIs can be used to identify the same resource; however, we will not address this concert in this paper.
[2]`http://www.w3.org/2005/ajar/tab`
[3]`http://linkeddata.org`
[4]`http://dbpedia.org`
[5]`http://geonames.org`
[6]`http://dbtune.org`

It then enables various advanced browsing capabilities and mash-ups, *e.g.* geolocation of artists of a particular music genre, as we previously exposed in [28]. In order to provide these links between resources, various techniques can be used, from manual and user-driven interlinking to automatic methods using heuristics to identify that two entities from different data sets are the same or related, as described in [31] for music-related data, one key feature of the success of the LOD cloud is mutual agreement on URIs to represent data as well as on lightweight ontologies to model these data rather than monolithic ones.

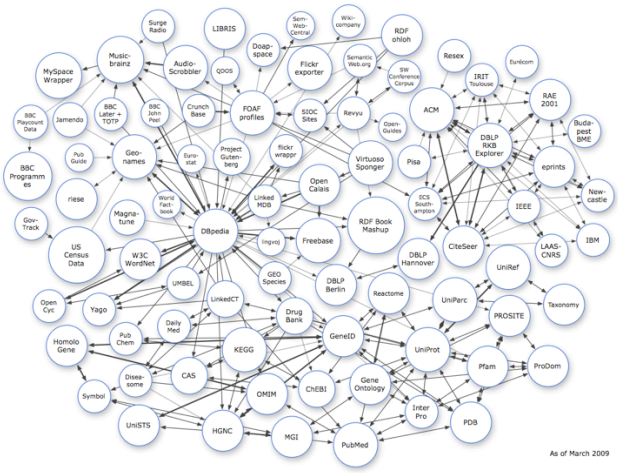


**Figure 1: The Linking Open Data cloud, March 2009**

As well as legacy data, LOD project has enabled the publishing of social data in RDF, such as user profiles, social networks and interests of people expressed on Web 2.0 sites[7], e.g. Flickr profiles or Last.fm playlists. As the example in Figure 2 illustrates, a single unified network can be created on the top of various applications interlinked thanks to this data: a blog post on a Drupal website can be linked to Flickr pictures via DBpedia. This scenario is already possible as data exporters for these services are already available on the Web.

Furthermore, not only academic projects are involved in this LOD community efforts, but also major Web 2.0 players. For instance, BBC programmes are available as RDF and companies such as Freebase also natively provide RDF data of their content. In addition, companies such as Yahoo! with the SearchMonkey search engine also emphasize the growing interest of the market for the Linked Open Data.

## 3.2 Linked Data and recommendation systems

As an outcome of these technologies, recommender systems that are not dependent on data solos can be realised. As Figure 2 demonstrates, the interlinking of various user profiles, social networks and related social data enables collaborative recommendation algorithms that work on several sources of user information.

Moreover, it enables new possibilities regarding how to build such systems and process the data. By developing systems dealing with RDF(S)/OWL data, and generally relying on the standardized SPARQL query language, a single algorithm can be applied to various data sources. For example, a user-based collaborative filtering algorithm that relies on FOAF (a vocabulary to describe people and their social networks [12]) and SIOC (a vocabulary to describe activities of online communities [11]) to infer neighborhoods can be deployed on any source of data using these models, from Last.fm to Flickr, content from both sites being available in RDF thanks to wrappers developed within the LOD project.

Nevertheless, as the Web of Data being highly-distributed, we need also to consider approaches for distributed querying and scalability and in section 6, we propose an architecture to build such systems taking into account both the advantages and the shortcomings of the Web of Data and the decentralized Web architecture.

# 4. LINKED DATA RECOMMENDATIONS

In order to provide practical implementations of the previous ideas, we have developed two different music-recommendation systems using Linked Data: the first approach is content-based and exploits semantically structured data about artists represented in DBpedia, the second one exploits the relationships between authors and topics on Wikipedia for a collaborative recommendation approach. Both algorithms are implemented on top of a similar infrastructure for accessing two sources from the LOD data cloud:

- The first one relies on **DBPedia**, which provides RDF data extracted from the infoboxes of Wikipedia pages in a structured way;
- the second one uses the **MediaWiki SIOC** exporter[8] [27] that provides RDF data about the authors and revisions of each page of a MediaWiki-powered website, and in particular can be used with Wikipedia.

## 4.1 Content-based Recommendation

As we mentioned earlier, data provided within the LOD project is structured data, in the sense that it is define using common semantics. RDF being the underlying model of the LOD cloud, this cloud can be considered as a set of RDF statements that link nodes together, a statement between node $A$ and node $B$ being represented by a relation $Ri(A, B)$, nodes being represented by URIs. Hence, our proposal is to apply semantic distance algorithm (often used in ontology matching techniques [15]) to compute the relatedness between entities of the Web of Data and hence build a recommender system using it.

Contrary to semantic distance as provided by [30] or a weighted version by [39], we do not consider the hierarchy of concepts in a taxonomy of classes to find the distance between two concepts, but compute the distance based on the different properties shared by the object. Indeed, as we mention earlier, the power of Linked Data does not rely on the accuracy of ontologies but on the interlinks that exist between resources, either direct links or indirect ones. Then, our algorithm considers all the links available between resources, no matter if it these links involve simple relationships or instantiation information. For instance, we consider that artists $A$ and $B$ are connected not only because they are instances of `dbpedia:artist` but because they share a concert with artist $C$, e.g. being linked via

[7]We will not discuss privacy and trust issues in this paper.

[8]`http://ws.sioc-project.org/mediawiki/`

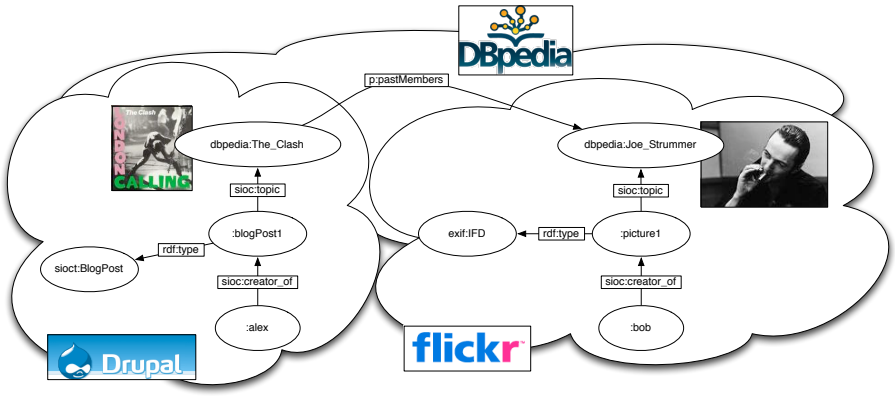


**Figure 2: Cross-services integration of information thanks to Linked Data**

a `dbpedia:associatedActs` relationship. Hence, while ontology usage in recommender systems has already been discussed [25], we consider a different approach.

Based on this idea, we designed a set of Linked Data Distance algorithms, named *LDD* that compute the distance between one seed entity and all the other entities appearing in the same dataset. Our six algorithms, for which the results are normalized between 0 and 1, are defined as follows. (1) *LDDs* is a simple algorithm simply using the direct relationships that exist between entities. A weighted version is provided as (2) *LDDws* to be more accurate, in which we give more weight to the properties that are present less time. (3) *LDDo* is an *object-based* distance algorithm, computing the number of entities in indirect relation with the seed entity, i.e. sharing a common property with the seed entity. It also features a weighted version, (4) *LDDwo*. Finally, our complex algorithm (5) *LDDc* mixes the two first ones, as well with a weighted version, (6) *LDDwc*.

For example, the following equation describes how to compute the distance between two entities $A$ and $B$ using $LDDws$. We weight each relation $Ri$ between $A$ and $B$ by the number of relationships $Ri$ that exist in the whole graph between $A$ and any other node, i.e. $Ri(A, N)$.

$$LDDws(A,B) = \frac{1}{1 + \sum_i \frac{Ri(A,B)}{1+log(\sum_N Ri(A,N))}} \quad (1)$$

Another interesting aspect of these algorithms is that they simply require a seed entity, defined by its URI, and a dataset. As they do not require an eager pre-computing phase, recommendations can be run at query time. It also imply, in case the dataset is highly distributed, that we must deal with scalability issues, and this is why we also propose an architecture to build such system later on section 6

### 4.1.1 Use-case and Implementation

In order to demonstrate our findings, we developed a recommender system based on the algorithms described and using data from DBpedia. DBpedia allows us to make recommendations for more than 30.000 artists and hence offers a useful testing ground for recommender development and testing. We asked a set of users to submit a list of five to ten bands and we then ran the six algorithms on a total of 39 bands, from styles as diverse as Country music, Metal or New-wave. Our first step was to ask the users to choose the most relevant algorithm. Table 1 shows that users considered *LDDwc* to be most accurate algorithm.

| Algorithm | Times ranked first |
|---|---|
| *LDDs* | 10 |
| *LDDws* | 1 |
| *LDDo* | 2 |
| *LDDwo* | 8 |
| *LDDc* | 5 |
| *LDDwc* | **13** |

**Table 1: User evaluation for *LDD* variants**

The recommendation resultset for the *LDDwc* algorithm for the query "Johhny Cash" is shown on the left side of Table 4.2).

In addition, it is interesting to compare these results with existing recommender systems. By considering the first 15 recommendations for Johnny Cash on Last.fm, six of them are included in the *LDDwc* resultset (actually, five of our first ten recommendations are also in the Last.fm list). The *LDDwc* algorithm discovered bands that were not in the Last.fm suggestions but, on inspection, are good recommendations.

For example, The Tennessee Three, the backing band of Johnny Cash, appear in the resultset but not in the Last.fm recommendation set even though they have a last.fm profile. An interesting feature of our algorithm is that, since distance is computing based on the paths that exist in the dataset, all recommended artists share something in common with the query URI, e.g. the artists may have recorded together.

As the resultset items are themselves provided as Linked Data URIs, an advantage of this approach is that associated resources for each recommended artist can be displayed in the user interface. We simple query DBpedia to retrieve a related picture, biography, link to the homepage, etc. As DBpedia is built thanks to various Wikipedia exports (i.e. from multiple languages), our interface can be instantaneously translated into various languages, without having to relaunch the algorithm or query another dataset. Figure 3 demonstrates a simple user-interface to browse the results in English and Spanish.

### 4.1.2 Cross-domain recommendations


**Figure 3: Multilingual interface for recommendations (English and Spanish)**

Additionally, we ran this algorithm to get book recommendations for the book query "Fight-Club". As Table 4.1.2 shows, the results are quite accurate and related to the seed object, since they are all books written by the same author. Most importantly, computing these recommendations did not require any change in the algorithm, since it is completely agnostic of the data and simply computes the distance based on the links available between entities according to the Linked Data principles.

| Artist | Distance |
|---|---|
| Invisible Monsters | 0.2275 |
| Survivor | 0.2290 |
| Choke | 0.2734 |
| Diary | 0.2880 |
| Lullaby | 0.2880 |

**Table 2: Recommendations for "Fight Club", ordered by distance from the original book**

## 4.2 Collaborative Recommendation

One of the most popular recommendation algorithms is collaborative filtering [32]. The recommendations are usually mined from a database of preferences for items by users. Based on the preferences, a matrix of distances between different users or different items is created, which is then used as the basis for recommendations. In order to use a collaborative filtering approach for topics on Wikipedia, we exploit the editing history of the pages which reveals the trace of the authors' collaboration processes. We can view this as a type of social network based where links are reveal shared interest and expertise in page topics.

Our collaborative-filtering recommendation algorithm uses each edit of an author on a topic page, as an indicator for the expertise and preference of the author for that page. In order to create the item-item comparison matrix, we first create a list to associate authors to the topics they have edited, and which are relevant for the recommendation task and domain:

1. aggregate all topics which are relevant to the recommendation domain, e.g. all musicians on DBpedia. We use a SPARQL query to do this.

2. for each topic, get the last $n$ edits and the name of the author doing the edit. We use the SIOC Mediawiki exporter to get the last 500 edits, excluding minor edits and edits by bot accounts.

3. for each edit on each topic, identify if the topic which was edited is from the recommendation domain, e.g. determine if the topic is about a musician, by using a SPARQL query to DBpedia.

This author-topics list is then used to create the item-item comparison matrix using a binary-weighted cosine score [33].

We implemented the outlined recommendation algorithm for musicians and musical artists from DBpedia and Wikipedia. The SIOC MediaWiki exporter is used as the data source for the connections between authors and topics, while DBpedia is used for accessing the semantically structured features of a topic. Table 4.2 shows the recommendation results for "Johnny Cash", of which 6 are present in the first 30 recommendations from Last.fm. The results are also evaluated against the results of the content-based algorithm and against a set of random musicians in section 5.

Just like the content-based recommendation algorithm from section 4.1, this collaborative recommendation algorithm can be used for cross-domain recommendations, by changing the criteria by which the topics are selected. Instead of using musicians, e.g. books or cities can be used, by specifying a different topic type in the SPARQL queries.

| Results of content-based recommendation | | Results of collaborative filtering recommendation | |
|---|---|---|---|
| **Artist** | **Distance** | **Artist** | **Distance** |
| June Carter Cash | 0.1281 | Willie Nelson | 0.0119 |
| Kris Kristofferson | 0.1340 | Dolly Parton | 0.0125 |
| Elvis Presley | 0.1398 | George Strait | 0.0138 |
| Glen Campbell | 0.1541 | Hank Williams | 0.0147 |
| Willie Nelson | 0.1608 | Jimmie Rodgers | 0.0148 |
| The Highwaymen | 0.1666 | George Jones | 0.0227 |
| Tennessee Three | 0.1737 | Reba McEntire | 0.0277 |
| Dolly Parton | 0.1778 | Tim McGraw | 0.0416 |
| Jerry Lee Lewis | 0.1792 | Les Paul | 0.0500 |
| Jack Clement | 0.1853 | Dave Dudley | 0.0500 |
| Bob Dylan | 0.1931 | Waylon Jennings | 0.0588 |
| Louis Jordan | 0.1951 | Roy Acuff | 0.0625 |
| Charlie Rich | 0.1966 | Dixie Chicks | 0.0625 |
| Carlene Carter | 0.1967 | George Harrison | 0.0625 |
| Al Green | 0.1996 | Garth Brooks | 0.0625 |

**Table 3: Recommendations for Johnny Cash from the content-based (left table) and collaborative filtering (right table) algorithms.**
Note that distance scores are not directly comparable, as the are derived in different ways.

# 5. PRELIMINARY EVALUATION

## 5.1 Using semantic features for the evaluation

Evaluation of recommender systems can be done in two ways [18]: The users of a running recommender system can evaluate the system in an on-line evaluation, whereas for off-line evaluation the performance of the recommender system is evaluated through existing data. A key issue for off-line evaluation is the need for labelled or preference data in the domain, which is often not available. A key advantage of working with Linked data is the availability of different category schemas with collaboratively produced category data

that is associated with resources.

In addition to the on-line evaluation of the content-based approach from section 4.1, in which we compared six different algorithms for recommendations, we evaluated both the content-based approach and the collaborative filtering approach against a random selection of artists, by quantifying the similarity of the recommended items through their SKOS categories.

SKOS – Simple Knowledge Organisation System [26] – is a formal language for representing controlled and structured vocabularies, like thesauri and taxonomies. SKOS itself is expressed as an OWL vocabulary. The DBpedia project uses the SKOS vocabulary to express the tree of hierarchical Wikipedia topic categories as semantically structured RDF. For instance, the following statement indicates that the category American_country_guitarists is narrower than the American_guitarists one:

```
dbpedia:Category:American_country_guitarists
  skos:broader dbpedia:Category:American_guitarists .
```

## 5.2 Evaluation method

In order to evaluate the result sets, we use the number of shared SKOS categories among each of them to quantify their homogeneity. While SKOS categories are hierarchical, the hierarchy of SKOS categories generally only needs to be traversed for about 5 levels to reach very broad terms. Therefore we exclude too generic SKOS categories, e.g. Living things. In order to determine the distribution of SKOS categories among a result set, we use the following algorithm:

- for each item of the result set, get all its SKOS categories, by following the SKOS category tree for 4 broader levels;
- exclude all categories which have a number of descendants which are 100% higher than the number of descendants of any of their subcategories;
- determine the number of categories which is shared amongst a percentage level of the whole result set. Start at a 100% and decrease in 10% steps.

Figure 4 shows the results of the evaluation using these three distinct datasets. As expected the random set of musicians has the least homogeneity of categories, with only 2 categories shared by 30% of the results. The musicians which are recommended by the content based algorithm have the highest homogeneity of categories, with 80% of musicians sharing one category and 70% sharing 6 categories. This can be explained by the fact that this algorithm exploits the features of the content for the recommendations. The musicians which are recommended by the collaborative filtering algorithm have more shared categories than the random set of musicians, while being less homogeneous then the content based results: 50% of musicians share 1 category and 40% share 11 categories. This is in line with the expectation that a collaborative filtering algorithm will tend to contain more serendipious results than the results of a content-based algorithm, which will tend be closer to the query artist.

# 6. TOWARDS A REFERENCE ARCHITECTURE

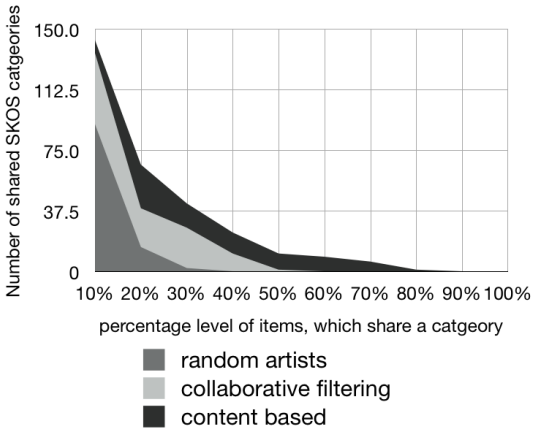


**Figure 4: Distribution of SKOS categories amongst the two recommendation result sets and a random set of musicians**

In the field of software engineering, a reference architecture, also known as a reference model, describes high-level concepts and terminology of a system, without fixing interfaces [14, page 242]. It enables discussing and comparing the common aspects of multiple implementations from a particular domain, and can be used as a blueprint for implementing a single application instance from that domain.

Using a reference architecture in this way can significantly reduce development and maintenance costs [35] of the implementation. A reference architecture also provides a common terminology for communicating concepts related to the implementation of a system, in this case for recommendation systems which are based on linked data.

## 6.1 Empirical basis

The reference architecture is based on a survey of 98 applications utilizing Semantic Web standards from two key demonstration challenges in the Semantic Web domain: the "Semantic Web challenge"[9], organised as part of the International Semantic Web Conference (2003–2008), and the "Scripting for the Semantic Web challenge"[10], organised as part of the European Semantic Web Conference (2006–2008).

The surveyed applications share a significant amount of functionality: **Data interfaces** provide an abstraction over remote and local data sources, a **persistence layer** stores data and run time state, and the **user interface** provides access for the user. This functionality has been implemented by **more than 90%** of surveyed applications. The **integration service** provides a unified view on heterogeneous data, and the **search engine** allows searching in data. This functionality has been implemented by **70% to 80%** of surveyed applications. The **crawler** discovers and retrieves remote data, and the **annotation user interface** allows creating new data. This has been implemented by **30% to 40%** of surveyed applications.

## 6.2 The components

The reference architecture and its components are based on the Linked Data community best practices [9] and on a

[9] http://challenge.semanticweb.org/
[10] http://www.semanticscripting.org

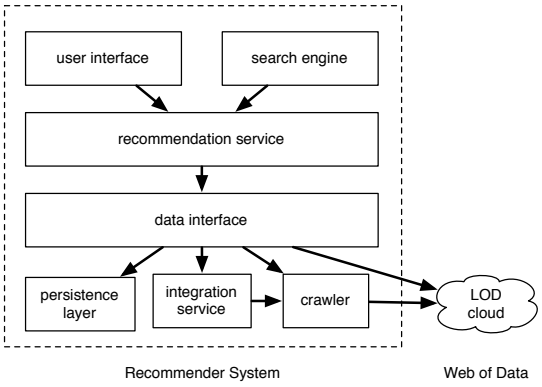


**Figure 5: A reference architecture for using linked data with recommender systems**

reference architecture for Semantic Web applications [19], which has been adapted for the domain of recommender systems. As the survey shows, there is a large area of functionality that is shared between Semantic Web applications. This common functionality can be abstracted into components which together form a reference architecture. In order to adapt the reference architecture for the domain of recommender systems, we introduce a "recommendation service" and merge the user interface with the annotation interface, as shown in figure 5. In order to change the recommendation algorithm for a different domain, e.g. from books to music, it is only necessary to exchange "recommendation service" component.

The **recommendation service** implements the recommendation algorithm of the recommender system, by accessing linked data via the data interface and by providing recommendation results to the user interface. The **data interface** provides the **interface** needed by the recommendation service to **access local or remote data sources**, with the distinction based on either physical remoteness or administrative and organisational remoteness. The **persistence layer** provides persistent **storage for data and run time state** of the application. It is accessed via the data interface. The **user interface** provides a **human accessible interface** for using the application and **viewing recommendations**. The **search engine** provides the ability to **perform searches** on the data and on the recommendations, based on the content, structure or domain specific features of the data. The **integration service** provides the means for **addressing structural, syntactic or semantic heterogeneity** of data, caused by accessing data from multiple data sources using diverse kinds of format, schema or structure. The desired result is a **homogeneous view on all data** for the recommendation service. The **crawler** implements **automatic discovery and retrieval of linked data**. It is required if data needs to be found and accessed in a domain specific way before it can be integrated.

Applications do not need to implement all components of the architecture, they can choose the components which are appropriate for the recommendation task. For instance, our content-based system implements this architecture in the following way: The **recommendation interface** is accessing remote data from DBpedia using a **data interface**, which is actually build using SPARQL queries. The **persistence layer**, i.e. an RDF store, is used to store the recommendations in combination with the **integration service** which integrates external data corresponding to artists (from DBpedia) in the same store. The (multilingual) **user-interface** can be defined without having to remotely query DBpedia each time a page must be rendered and a **search engine** allows to find artists for which we can provide recommendations. There is no dedicated **crawler** as we rely on external sources that provide a SPARQL endpoint as a way to run distributed queries.

# 7. CONCLUSION

In this paper, we presented some recent findings regarding Linked Data and recommender systems. Our goal is to explore how recommender research can benefit from potentially rich sources of new data and from new research on decentralised recommmendation. As such we demonstrated how data from the LOD project can be efficiently used to build new recommender systems that can operate independently of a particular domain data. We carried out a preliminary evaluation using the hierarchical categories associated with Linked Data resources. We believe that more refined evaluation techniques are possible using this category data, creating richer possibilities for algorithm design and testing. We also defined how such recommender systems could be designed by providing a reference architecture, based on empirical analysis of existing applications.

We do not claim to solve all the issues of current recommender systems, but we hope that this work can lead to new possibilities for recommender research, particularly in developing more open, distributed and interoperable systems that use various interlinked datasets to recommend items, based both on the content and the social interactions between people.

# 8. REFERENCES

[1] OWL Web Ontology Language Overview. W3C Recommendation 10 February 2004 , World Wide Web Consortium, 2004. `http://www.w3.org/TR/owl-features/`.

[2] RDF Vocabulary Description Language 1.0: RDF Schema. W3C Recommendation 10 February 2004, World Wide Web Consortium, 2004. `http://www.w3.org/TR/rdf-schema/`.

[3] SPARQL query language for RDF. W3C Recommendation 15 January 2008, World Wide Web Consortium, 2008. `http://www.w3.org/TR/rdf-sparql-query/`.

[4] S. Auer, C. Bizer, J. Lehmann, G. Kobilarov, R. Cyganiak, and Z. Ives. Dbpedia: A nucleus for a web of open data. In *Proceedings of the 6th International Semantic Web Conference (ISWC2007)*, volume 4825 of *LNCS*, pages 715–728. Springer, 2007.

[5] S. Berkovsky, Y. Eytani, T. Kuflik, and F. Ricci. Enhancing privacy and preserving accuracy of a distributed collaborative filtering. In *RecSys*, pages 9–16, 2007.

[6] S. Berkovsky, T. Kuflik, and F. Ricci. Cross-domain mediation in collaborative filtering. In *User Modeling*, pages 355–359, 2007.

[7] T. Berners-Lee. Linked Data. Design issues for the world wide web, World Wide Web Consortium, 2006. `http://www.w3.org/DesignIssues/LinkedData.html`.

[8] T. Berners-Lee, J. A. Hendler, and O. Lassila. The Semantic Web. *Scientific American*, 284(5):34–43, 2001.

[9] C. Bizer, R. Cyganiak, and T. Heath. How to Publish Linked Data on the Web. Technical report, FU Berlin, 2007.

[10] U. Bojars, A. Passant, J. G. Breslin, and S. Decker. Social Network and Data Portability using Semantic Web Technologies. In *SAW 2008 - BIS*, 2008.

[11] J. G. Breslin, A. Harth, U. Bojārs, and S. Decker. Towards Semantically-Interlinked Online Communities. In *Proceedings of the 2nd European Semantic Web Conference (ESWC2005)*, volume 3532 of *LNCS*, pages 500–514. Springer, 2005.

[12] D. Brickley and L. Miller. FOAF Vocabulary Specification. Namespace Document 2 Sept 2004, FOAF Project, 2004. http://xmlns.com/foaf/0.1/.

[13] R. D. Burke. Hybrid web recommender systems. In *The Adaptive Web*, pages 377–408, 2007.

[14] A. Endres and D. Rombach. *A Handbook of Software and Systems Engineering*. Pearson Education, 2003.

[15] J. Euzenat and P. Shvaiko. *Ontology Matching*. Springer-Verlag, Berlin-Heidelberg, 2007.

[16] G. Gonzalez, B. Lopez, and L. de la Rosa. A multi-agent smart user model for cross-domain recommender systems. In *IUI'05 Workshop: Beyond Personalization 2005*, 2005.

[17] P. Han, B. Xie, F. Yang, and R. Sheng. A scalable p2p recommender system based on distributed collaborative filtering. *Expert systems with applications*, 2004.

[18] C. Hayes, P. Massa, P. Avesani, and P. Cunningham. An online evaluation framework for recommender systems. In *Workshop on Personalization and Recommendation in E-Commerce*. Springer Verlag, 2002.

[19] B. Heitmann, C. Hayes, and E. Oren. Towards a reference architecture for semantic web applications. 2009.

[20] G. Klyne and J. J. Carroll. Resource Description Framework (RDF): Concepts and abstract syntax. W3C Recommendation 10 February 2004, World Wide Web Consortium, 2004. `http://www.w3.org/TR/rdf-concepts/`.

[21] A. Kobsa. Privacy-enhanced web personalization. pages 628–670. 2007.

[22] G. Linden, B. Smith, and J. York. Amazon.com recommendations: Item-to-item collaborative filtering. *IEEE Internet computing*, 7(1):76–80, 2003.

[23] L. McGinty and B. Smyth. Collaborative case-based reasoning: Applications in personalised route planning. In *ICCBR '01: Proceedings of the 4th International Conference on Case-Based Reasoning*, pages 362–376, London, UK, 2001. Springer-Verlag.

[24] S. Middleton, D. De Roure, and N. Shadbolt. Capturing knowledge of user preferences: ontologies in recommender systems. In *Proceedings of the 1st international conference on Knowledge capture*, pages 100–107. ACM New York, NY, USA, 2001.

[25] S. E. Middleton, H. Alani, and D. D. Roure. Exploiting synergy between ontologies and recommender systems. *CoRR*, cs.LG/0204012, 2002.

[26] A. Miles and J. Pérez-Agüera. SKOS: Simple knowledge organisation for the web. *Cataloging & Classification Quarterly*, 43(3):69–83, 2007.

[27] F. Orlandi and A. Passant. Enabling cross-wikis integration by extending the SIOC ontology. In *Proceedings of the Fourth Workshop on Semantic Wikis (SemWiki2009)*, 2009.

[28] A. Passant and Y. Raimond. Combining Social Music and Semantic Web for music-related recommender systems. In *Proceedings of the ISWC2008 Workshop on Social Data on the Web*, volume 405 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2008.

[29] E. Plaza and L. McGinty. Distributed case-based reasoning. *The Knowledge Engineering Review*, 20(03):261–265, 2005.

[30] R. Rada, H. Mili, E. Bicknell, and M. Blettner. Development and application of a metric on semantic nets. 19(1):17–30, 1989.

[31] Y. Raimond, C. Sutton, and M. Sandler. Automatic Interlinking of Music Datasets on the Semantic Web. In *Proceedings of the WWW2008 Workshop Linked Data on the Web (LDOW2008)*, volume 369 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2008.

[32] P. Resnick, N. Iacovou, M. Suchak, P. Bergstrom, and J. Riedl. GroupLens: An Open Architecture for Collaborative Filtering of Netnews. In *In Proceedings of ACM 1994 Conference on Computer Supported Cooperative Work*, 1994.

[33] B. Sarwar, G. Karypis, J. Konstan, and J. Reidl. Item-based collaborative filtering recommendation algorithms. In *Proceedings of the 10th international conference on World Wide Web*, pages 285–295. ACM New York, NY, USA, 2001.

[34] G. Shani, M. Chickering, and C. Meek. Mining recommendations from the web. In *Proceedings of the 2008 ACM conference on Recommender systems*, pages 35–42. ACM New York, NY, USA, 2008.

[35] M. Shaw and D. Garlan. *Software Architecture. Perspectives of an Emerging Discipline*. Prentice Hall, 1996.

[36] B. Smyth and E. Balfe. Anonymous personalization in collaborative web search. *Inf. Retr.*, 9(2):165–190, 2006.

[37] J. Wang, J. Pouwelse, R. L. Lagendijk, and M. J. T. Reinders. Distributed collaborative filtering for peer-to-peer file sharing systems. In *SAC '06: Proceedings of the 2006 ACM symposium on Applied computing*, pages 1026–1030, New York, NY, USA, 2006. ACM.

[38] K. Yoshii, M. Goto, K. Komatani, T. Ogata, and H. G. Okuno. Hybrid collaborative and content-based music recommendation using probabilistic model with latent user preferences. In *ISMIR*, pages 296–301, 2006.

[39] J. Zhong, H. Zhu, J. Li, and Y. Yu. Conceptual graph matching for semantic search. In U. Priss, D. Corbett, and G. Angelova, editors, *Proceedings of the 10th International Conference on Conceptual Structures (ICCS 2002)*, volume 2393 of *LNCS*, pages 92–196. Springer, 2002.