Challenges in Using Linked Data within a Social Web Recommendation Application to Semantically Annotate and Discover Venues

Jakub Dzikowski, Monika Kaczmarek, Szymon Lazaruk, Witold Abramowicz

Challenges in Using Linked Data within a Social Web Recommendation Application to Semantically Annotate and Discover Venues

Jakub Dzikowski, Monika Kaczmarek, Szymon Lazaruk, and Witold Abramowicz

Department of Information Systems,
Faculty of Informatics and Electronic Commerce,
Poznan University of Economics
al. Niepodleglosci 10, 61-875 Poznan, Poland
{j.dzikowski,m.kaczmarek,s.lazaruk,w.abramowicz}
@kie.ue.poznan.pl
http://www.kie.ue.poznan.pl

Abstract. This paper focuses on a semantically-enhanced Social Web Recommendation application, called *Taste It! Try It!* It is a mobile restaurants' review and recommendation application based on a Linked Data source and integrated with a social network. The application is consuming Linked Data (while creating the reviews), producing semantic annotations (about the reviewed entities) and then, querying the gathered data in order to offer personalized recommendations. In this paper, we focus only on the consumption and usage of Linked Data for the needs of social recommendation system and point out the challenges and shortcomings that need to be addressed.

Keywords: Linked data application, semantic annotations, semantic content creation tool

1 INTRODUCTION

The Semantic Web paradigm constitutes a major step in the evolution of the Web. It is to enable machines to understand the meaning of information on the WWW. It is done via extending the network of hyper-linked human-readable web pages by inserting machine-readable meta-data, i.e., semantic annotations, about the Web content and information on how they are related to each other, thus, enabling automated reasoning [Berners-Lee et al., 2001]. A semantic annotation is machine processable, if it is explicit, formal, and unambiguous and this goal is usually reached by using ontologies [Uschold and Grüninger, 1996].

The Web has evolved into the Web of Data [Bizer et al., 2009] by using a set of best practices for publishing and connecting structured data on the Web, known as Linked Data. The content of the Linked Data cloud is diverse in nature [Bizer et al., 2009], comprising, e.g., data about geographic locations, people, companies, radio programmes, genes, proteins, census results, and reviews. Since

2007, the Linking Open Data cloud has expanded considerably. However, apart of some initiatives showing how to build applications using it, there is still plenty of space for more end-user applications operating on the Linked Data.

The above trends constitute a motivation for the development of a *Taste It! Try It!* application focusing on the creation of semantically annotated restaurants' reviews using concepts from DBpedia. The mentioned application is not only consuming the Linked Data (while creating the reviews), but also produces additional semantic annotations (about the reviewed entities) using either the mobile or WWW interface. As we are following the faceted-based approach to the review creation, we benefit from the additional information within the disambiguation process. That is why for the needs of the *Taste It! Try It!* application, a distinct disambiguation solution has been designed, adjusted to the specific needs of a mobile device. *Taste It! Try It!* is a real-world application and during the performed experiments it has been used by 180 users.

The goal of the paper is twofold. On the one hand, it is to show the user-friendly way of creating semantic annotations using Linked Data, and on the other, is to start a discussion on the consumption and usage of Linked data within the applications such as *Taste It! Try It!* and point the challenges and shortcomings that need to be addressed. Although the issues related to the efficiency of application of semantic technologies (reasoners, integration of data) are well investigated in the literature, there is still a number of issues left to be addressed. Thus, with this paper we aim at starting a discussion on the maturity of both the semantic data sources as well as tools that are to facilitate the Semantic Web and the Linked Data adoption. In addition, as the application is also integrated with the Facebook portal, the privacy related issues are also discussed.

In order to meet the above mentioned goal, the paper is structured as follows. We start with a short summary of the related work and position our application towards the work of others. Then, the vision of the tool, along with its architekture is shortly presented. Next, we focus on the interactions with the Linked Data sources within the *Taste It! Try It!* application and provide information on the semantic annotations' creation and usage process. Then, the challenges tackled in the context of DBpedia[1] and Facebook are discussed. The paper concludes with final remarks.

2 RELATED WORK

Recommender Systems (RS) are information search tools that have been proposed to cope with the information-overload problem, i.e, the typical state of a consumer, having too much information to make a decision [Adomavicius and Tuzhilin, 2005, Burke, 2007]. Recommender Systems can be either [Pu et al., 2012]: *rating-based* (content-based or social/collaborative-based) – users explicitly express their preferences by giving binary or multi-scale scores to items

[1] `http://dbpedia.org`

that they have already experienced, or *feature-based* (case-based, utility-based, knowledge-based and critiquing-based) – evaluating the match between a user's need and the set of options available [Burke, 2002].

Recommender systems normally use software instead of users for the information filtering tasks [Peis et al., 2008]. This approach, however, has some disadvantages. The communications process, either between agents and users or agents only, is complicated because of the heterogeneity of information representation, which in turn leads to incapability of its reuse in other processes and applications. Thus, Semantic Web technologies are more and more often used within the recommender systems.

A particularly interesting example of ontology-based system has been introduced by Cantador and Castells in [Cantador and Castells, 2006] and extended in [Cantador et al., 2011]. In this work, a multi-layer semantic social network model has been proposed, based on a hypothesis that since user's interests are not made of a single piece, any approach that deals with them as such would have inevitable limitations. Thus, the system has been defined from different perspectives, splitting user profiles according to meaningful groups/layers of preferences shared among users, so that the similarities between users are to be established based on sub-profiles, rather than the global ones. This approach is also continued in the *Taste It! Try It!* application.

Semantic Web technologies have been introduced almost a decade ago, and yet, their real-life impact has been considerably limited for first few years. The situation has changed dramatically by an initiative called a Linked Data project. Based on the simple semantic technologies, like RDF and URIs, used along with Linked Data principles[2], a number[3] of datasets have been made available in a machine-understandable manner, eg., Wikipedia's resources are available on the Web of Data in the form of DBpedia.

Linked Data sets are used in more and more real-world application. Examples include [Hausenblas, 2009]:

- Faviki[4] – social bookmarking tool, utilizing semantic tags stemming from Wikipedia (via DBpedia) so that all concepts are ambiguously identified;
- DBpedia mobile[5] – mobile, location-based application presenting information from DBpedia on a map;
- Revyu[6] – a generic reviewing site based on the Linked Data principles and the Semantic Web technology stack.

Linked Data lowers the entry barrier for data providers by focusing on publishing structured data rather than, on the ontological level or inferencing, hence fosters a wide spread adoption. However, there exists some challenges that need to be tackled by developers of real-world linked open data applications, not

[2] `http://www.w3.org/DesignIssues/LinkedData.html`
[3] 295 datasets up to 2011 - source: `http://lod-cloud.net/`
[4] `http://faviki.com`
[5] `http://wiki.dbpedia.org/DBpediaMobile`
[6] `http://revyu.com/`

least of which include resource discovery, consolidation and integration across a distributed environment. Another group of challenges arise from the fact that building application based on foreign resources under control of third parties leads to unresolved issues regarding potentially dynamic nature of dataset content, meaning that they can be changed or even disappear [Umbrich et al., 2010]. Furthermore, problems of co-reference, ontology mapping, aggregation from distributed sources, resource discovery, queries spanning multiple datasets are also to be tackled [Millard et al., 2010].

The *Taste It! Try It!* application benefits from the already developed semantic technologies and tools, and offers an added value through their integration and usage in order to, on the one hand, contribute to the Linked Data by producing semantic annotations, and on the other, to offer personalized advanced discovery and clustering possibilities. For the needs of the *Taste It! Try It!* application, a distinct disambiguation solution has been designed, adjusted to the specific needs of a mobile device. All of these features together, make the *Taste It! Try It!* application a distinct solution.

3 TASTE IT! TRY IT! APPLICATION

Taste It! Try It! has been designed as a Web 2.0 application supporting the creation of semantic annotations describing various places and locations. It is targeted at end-users, among which two groups may be distinguished: data producers (contributors) – users providing reviews of places, i.e., people creating semantically annotated reviews, and data consumers (beneficiaries) – users interested in the content produced by the application, i.e., people looking for opinions about various places. Therefore, on the one hand, *Taste It! Try It!* enables data producers to contribute to a semantic content creation process using their mobile devices[7] or a WWW interface, and on the other, provides data consumers with personalized, semantic, context-aware recommendation process (i.e., offer a personalized semantic search mechanism).

3.1 Storyboard

The storyboard supported by the system is as follows. A user goes to a restaurant. While being at the restaurant, the user decides to share his opinion on the restaurant with other members of the community. He uses Taste It! Try It! to express this opinion and creates a review by providing values to selected features suggested by the application. In particular, the review edition screen is divided into 3 tabs:

Main tab containing basic and obligatory information such as: name of the place being reviewed; type of location; GPS location which is to be provided using the mobile devices built-in GPS module; and star ratings that allow the user to express his Overall, Service, Atmosphere, Food impression in the quantitative manner, by assigning from 1 to 5 stars in each category.

[7] The application is developed to work with the Android system

Details tab allowing the user to assess a wide range of qualitative features of the place, which are grouped in intuitive categories such as: Dining options, Entertainment or Good for. In this tab, the user is also able to select the cuisine type and best dishes/drinks served. Values of those fields are suggested from DBpedia.

More tab containing some additional star ratings and features together with a free-text comment field.

The review is then uploaded to a *Taste It! Try It!* server and in the background, the semantic representation is created. Based on the quantity and quality of created annotations, the user may be awarded with a special title e.g., *Polish-cuisine expert*, *International-food expert*. This title is visible to his friends at the Facebook portal, with which the application is integrated. Moreover, a user may check the ranking among his friends on Facebook.

In addition, based on the user behaviour and data made available by the Facebook portal, the user profile is created, which is then used in the personalization process.

3.2 Architecture

Taste It! Try It! consists of three main components: an Android client, a Facebook client and a Server. Moreover the *Taste It! Try It!* application communicates with three other components: Facebook (FB), the INSEMTIVES platform and DBpedia (see fig. 1). The first three components are to provide basic functionalities of the application. Communication with the other ones is to connect the application with Web 2.0 services (the social aspect) and Linked Data (the semantics).

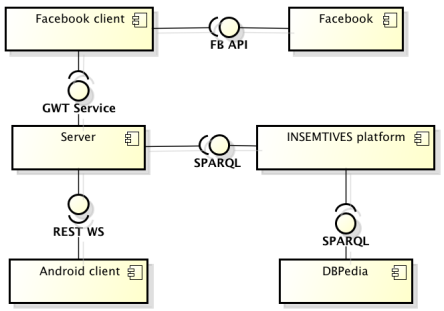

Fig. 1. UML component diagram of *Taste It! Try It!*

The Android client provides a user with a front-end to manage reviews. For communication with the Server a RESTful Web Service interface is used. The

Facebook client is a web front-end embedded in the Facebook canvas[8] and written with the use of Google Web Toolkit framework[9]. Via the FB API interface[10] the Facebook client authenticates and authorises the user, as well as retrieves basic information about the user (name, gender, list of friends). The GWT Service interface provided by the Server is to retrieve and store informations about user's interaction with the application, including restaurant reviews and user's personal information.

To support the semantic content creation process, the *Taste It! Try It!* application integrates with the INSEMTIVES platform and DBpedia. The INSEMTIVES platform is a tool created by the INSEMTIVES consortium[11] on top of the OWLIM semantic repository[12], which uses native RDF engines implemented in Java and both Sesame[13] and Jena[14] frameworks. Among others it consists of a SPARQL endpoint, that is used to store and retrieve RDF triples. A part of semantic data is retrived and cached from DBpedia via the DBpedia SPARQL endpoint.

3.3 Semantic annotations based on Linked Data

An ontology is a formal, explicit specification of a shared conceptualization [Gruber, 1995]. It provides a data model, i.e., shared vocabulary that may be used for describing objects in the domain (their type, properties and relations). The important part of every ontology are the instances forming a knowledge base. Instances refer to a concrete object being an instantiation of an object type represented by the ontology.

After conducted analysis of the coverage and popularity of the currently available semantic contents providers, the decision was taken to consume data from DBpedia while creating the reviews in order to provide unambiguous values to the reviewed features of a venue. After the careful analysis of the structure of relevant concepts from DBpedia and schema.org, as well as types of assigned properties to the Restaurant concept, and comparing it to the Data model used by the *Taste It! Try It!* application, we have noticed that values of only few elements/fields require disambiguation (e.g., features expressed using stars or having binary value can be directly mapped to DBpedia concepts within the application model and no-aid from a user is necessary). Thus, only the following facets from the review need to be linked to the concepts from DBpedia by users of the application: category of restaurant, type of cuisine, food and drinks served. Values of all other features are mapped automatically.

As a consequence of this decision, users while filling in the above mentioned aspects of a review, are pointing to the concepts from DBpedia. As the process

[8] `http://developers.facebook.com/docs/guides/canvas/`
[9] `https://developers.google.com/web-toolkit/`
[10] `http://developers.facebook.com/docs/reference/api/`
[11] `http://insemtives.eu/`
[12] `http://www.ontotext.com/owlim`
[13] `http://www.openrdf.org/`
[14] `http://jena.apache.org/`

of assigning semantic annotations to the created reviews was to be on the one hand, user-friendly and almost invisible to users, and on the other, work well on a mobile device (specific way of introducing data), we have decided to take advantage of an auto-completion mechanism suggesting possible tags to be used. The auto-completion mechanism is shown in figure 2.

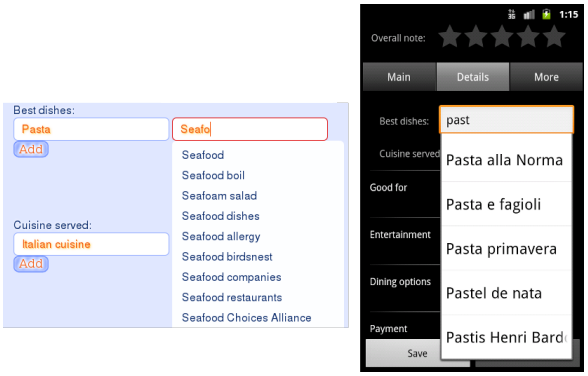

Fig. 2. Autocompletion mechanism

As we are following the faceted-based approach to the review creation, we can benefit from the additional knowledge in order to disambiguate and limit the potential tags (concepts from DBpedia) to be presented to users as an option to choose from. This becomes even more important in case of a mobile device – the presented list of tags to choose from, should not be too long. Therefore, for the needs of the *Taste It! Try It!* application, a distinct disambiguation solution has been designed. It works as follows – once a user starts to type in the first characters of the tag for the selected feature, the disambiguation takes place in the following steps:

- an appropriate SPARQL query limited to the branch of interest (e.g., in case a user is providing a tag to the *best dishes* feature, only the concepts related to Food and drinks are suggested) is created automatically by the application,
- the obtained result is filtered using the preferred language of the user as well as the first characters typed in and as a result the list of concepts with their labels is retrieved from DBpedia,
- the proposed list of suggestions is sorted by Levenshtein distance between suggestions and typed characters and presented to the user.

The SPARQL queries used to gather the relevant concepts for auto-completion, depending on the type of field being annotated, needed to be created in a semi-automated manner – using appropriate scripts to generate queries based on the manual analysis of the structure of DBpedia.

The reviews provided by users are stored on the server and then serialized into the RDF format, so that they can be later on published as linked dataset. In order to check, whether such a venue is already defined within the ontology (i.e., in order to disambiguate the object being reviewed), the server performs the following steps:

- Taking into account the data provided by a user, i.e., the geographical coordinates of the location, the SPARQL query is formulated automatically by the application in order to retrieve all already described places of interest (restaurants and all subcategories) within specific radius from the coordinates provided. As a result a set of objects is being retrieved.
- If the set is not null, the Levenshtein measure is applied to the name of the concepts being retrieved and the name provided in the review in order to identify similar objects (e.g., in case of a typo in a name of the place).
- Based on the above, the ranking list is created. In case only one object reaches the defined similarity threshold, the disambiguation is automatically performed, in all other cases – the system is assuming it is a new concept as long as a user being presented with the ranked list of restaurants will not decide otherwise.

Once we know whether the RDF triples will concern a new or already existing concept, the further serialization is performed.

Another issue that needs to be addressed while creating the RDF triples is reaching the consensus in case contradictory information is provided by different users on the same venue. Therefore, the tool needed also to incorporate a feature supporting consensus creation while providing annotations. It is worth noting that the created semantic annotations that are to be made available outside the *Taste It! Try It!* application, are anonymous (no information about the author of the review is revealed outside the *Taste It! Try It!* application) and the subjective evaluation of the venue is expressed in an aggregated form (e.g., by showing an average number of assigned stars).

A new venue is being added to the database as soon as it reaches the limit of 3 reviews being assigned. Each new review added about the already existing venue within the knowledge base, may result in updating the information on the restaurant stored there. The created RDF triples are then uploaded to the INSEMTIVES platform via the SPARQL interface and stored in the local RDF repository.

Although our application exploits semantic datasets, the complex semantic nature of the underlying information is hidden to the end users who do not interact directly with the Semantic Web languages and technologies such as RDF or SPARQL. It is also visible while performing the search of a restaurant of interest. The following scenarios are currently supported:

- Searching for a restaurant with some quantitative criteria (non-semantic, e.g., number of stars assigned (not less than...)).
- Searching for a restaurant with some qualitative (non-semantic) criteria added, e.g., wi-fi zone, live sport events transmissions etc.

Fig. 3. *Taste It! Try It!*– searching for a restaurant – choosing location, cuisine and dishes

- Searching restaurants near some location – a map and coordinates (see fig 3).
- Searching for a restaurant with some criteria requiring reasoning (semantic ones from DBpedia) – type of cuisine and type of dishes (see fig 3).

Additionally the *Taste It! Try It!* application enables personalisation in the searching process. Thus, the following personalisation-enhanced search scenarios are supported:

- Searching for a restaurant I may like, i.e., recommended by people with a similar profile.
- Searching for a restaurant that my friends from the Facebook recommended (criteria – author of the review).
- Searching for a restaurant that one specific persons (that I trust) likes.
- Hang-out (recommend a restaurant for n-number of *Taste It! Try It!* users).

When it comes to specifying the semantic criteria, a user searches for it by typing characters in the corresponding text field, as indicated on fig 3. The application returns an auto-complete list of suggested concepts retrieved from DBpedia. Once the list has been populated, the user can select one (or more) of the suggested concepts.

As already mentioned, while returning the search results, the additional personalization may be applied. Thus, in fact while searching, the personalized recommendation exploits both the knowledge base – information gathered by the *Taste It! Try It!* application and DBpedia (content-based approach), and the similarities between users (collaborative-filtering approach).

4 CHALLENGES

Implementation of scenarios and functionalities described shortly in the previous section, required addressing several issues. In the case of integration with DBpedia we had to deal with poor reliability of public SPARQL endpoints while trying to find an efficient solution to a well known problem of querying semantic data coming from various repositories. In the case of integration with the Facebook, we had to handle problems resulting from a restricted Facebook privacy policy.

In order to test our application and evaluate the usability of the developed tool, experiments with 180 participants were conducted during December 2011 and January 2012. The conducted experiments not only allowed to improve the application and verify its performance, but also as a result, 2274 reviews on approximately 900 different restaurants with 5667 semantic concepts coming from DBpedia were created. The challenges presented below encompass also few selected findings from the experiments, however, due to the limited space the experiments themselves are not further described.

4.1 Integration with DBpedia

During the process of searching for restaurants both the original concepts from DBpedia and the ones created by the *Taste It! Try It!* application need to be considered. As already mentioned in the previous section, search criteria include e.g., cuisine types and dishes from DBpedia (being used within the created reviews) as well as restaurants, which could be already defined within DBpedia or constitute new venues added by the *Taste It! Try It!* application. In this case two approaches may be followed: (1) performing a federated query to DBpedia and our local SPARQL endpoint or (2) storing part of the data from DBpedia in our endpoint. We decided to test both of them.

At the time of writing, neither DBpedia nor the SPARQL enpoint in the INSEMTIVES platform supports federated queries. It is possible to query DBpedia and the INSEMTIVES platform using another endpoint supporting federated queries[15]. However, there are only few public endpoints of that kind and we find their performance unsatisfactory.

After unsuccessful experiments with Virtuoso[16] and Sesame[17] we have decided to use – Apache Jena Fuseki[18]. An instance of the endpoint has been deployed and tested as an interface to perform federated queries. The endpoint itself turned out to be a feasible solution, however, the following performance issues have occurred:

– Because of the obscure structure of DBpedia, queries for retrieving cuisine or dishes types had to be quite complex. Performing these queries took long time and often led to time-out errors.

[15] For example `http://sparql.org/query.html`
[16] `http://virtuoso.openlinksw.com/`
[17] `http://www.openrdf.org/`
[18] `http://incubator.apache.org/jena/documentation/serving_data/`

- The performance of the DBpedia SPARQL endpoint depends of its current usage and often is very poor (participants of experiments complained having to wait up to 30 seconds or more for a result). Sometimes the endpoint is not available at all.
- The way SPARQL federated queries work is not efficient enough. For example a federated query (as shown in listing 1.1) is performed as follows. The first sub-query is performed on the first endpoint (in our case the INSEMTIVES platform). Then, for every result of the sub-query further sub-queries are performed on the second endpoint (the DBpedia). At the end the results of the queries are integrated by the endpoint that supports federated queries (Apache Jena Fuseki).

The first of the issues enumerated needs to be stressed. An important finding from the conducted experiments is the poor quality of information provided by DBpedia (mainly the lack of consistency regarding the structure to which concepts of the same type are being assigned). The coverage of DBpedia was deemed as unsatisfactory by most of the users. It does not necessarily result from the fact that the required concept is not present in the data gathered by DBpedia, but the concept could have been assigned to a not-intuitive place in the structure, which makes it difficult to be discovered

As a consequence, we decided to follow the second approach and to duplicate the part of DBpedia. Although all the data from DBpedia is available to download[19], required by our application RDF triples (covering such concepts as dishes, cuisine types and restaurants) are defined in multiple files. Thus, the whole English version of DBpedia has been downloaded and integrated with the RDF repository in our instance of the INSEMTIVES platform. During the first launch of the platform indexing of 111 GB text files with RDF triples in the NT format took about 3.5 day. Additionally, the response-time of the copy of DBpedia was too long (because of the large amount of data to process and complex queries to handle). Thus, we decided to retrieve via the DBpedia SPARQL endpoint only required concepts, generate *.nt files and insert them to our RDF repository.

Listing 1.1. SPARQL federated query to retrieve from DBpedia `skos:broader` concepts of dishes from the INSEMTIVES platform

```
select ?uriBroader {
  { service <http://insemtives.example.com/sparql>
    { select ?uri where {
        <http://insemtives.eu/tasteit/DishesType>
        <http://insemtives.eu/tasteit/instance>
        ?uri .   } } }
  { service <http://dbpedia.org/sparql>
    { select ?uriBroader ?uri where {
        ?uri
        <http://www.w3.org/2004/02/skos/core#broader>
        ?uriBroader . } } } }
```

[19] `http://wiki.dbpedia.org/Downloads37`

To retrieve only relevant concepts and relations the public DBpedia SPARQL endpoint, our instance of the INSEMTIVES platform and our instance of the Apache Jena Fuseki were used. At first queries for restaurants, dishes and cuisine types were performed on the DBpedia. Then, we generated RDF triples to store them in our repository in the INSEMTIVES platform. The example of the triples is shown in listing 1.2.

Listing 1.2. An example of generated RDF triples for dishes types (NT format)

```
...
<http://insemtives.eu/tasteit/DishesType>
        <http://insemtives.eu/tasteit/instance>
                <http://dbpedia.org/resource/Pasta_salad> .
<http://insemtives.eu/tasteit/DishesType>
        <http://insemtives.eu/tasteit/instance>
                <http://dbpedia.org/resource/Waldorf_salad> .
...
```

Storing presented RDF triples in our repository enabled us to display in the application auto-suggestions for cuisine types and dishes, in efficient manner. Instead of complex query to the DBpedia SPARQL endpoint, a simple query to our local endpoint is being performed to retrieve a list of suggestions.

Additionally, our reasoning mechanism requires an access to all concepts C and C' that fulfil the following conditions:

- A given cuisine type, dish or restaurant is in the relation `skos:broader` with C.
- A given cuisine type, dish or restaurant is in the relation `dcterms:subject` with C.
- C is in the relation `skos:broader` with C'.

To retrieve appropriate RDF triples we performed federated queries to the DBpedia and the INSEMTIVES platform on our Apache Jena Fuseki endpoint. An example of the query is presented in listing 1.1.

However, the presented example is still too complex to perform it as one query. The public DBpedia SPARQL endpoint was not able to handle about 10,000 sub-queries (for all pre-fetched dishes types) without time-out. In consequence, we had to put limits and offsets after first sub-query and generate queries automatically. We have prepared UNIX/Linux shell scripts to generate and perform queries to count entities from the first sub-queries, retrieve concepts for particular subsets of concepts for the sub-queries and generate appropriate RDF triples in the NT format. A part of the script is presented in listing 1.3. The parameter `LIMIT` contains limit for the results in the first sub-query. In our case value of this parameter differs according to the complexity of the query (from 1 to 50). It had to be adjusted to the DBpedia endpoint performance for particular queries. The `COUNT` variable contains a number of results of the first sub-query and the `s-query` is included in Fuseki distribution Ruby script for querying SPARQL endpoints.

Listing 1.3. Part of a shell script to perform federated queries

```
for OFFSET in `seq 0 $LIMIT $COUNT`
do
    QUERY="""
select ?uri ?uriBroader {
{ service <http://insemtives.example.com/sparql>
  { select ?uri where {
      <http://insemtives.eu/tasteit/DishesType>
      <http://insemtives.eu/tasteit/instance>
      ?uri . }
    limit $LIMIT offset $OFFSET   } }
  { service <http://dbpedia.org/sparql>
    { select ?uriBroader ?uri where {
        ?uri
        <http://www.w3.org/2004/02/skos/core#broader>
        ?uriBroader . } } } }
    """
    ../s-query --output=tsv --service $FUSEKI_URL "$QUERY"
done
```

Thus, the implementation of two possible scenarios proved that neither of them works fully in practice, in this way limiting the potential usage of the Linked Data. Although, all of the content of DBpedia is available to download and free to use, a number of issues needs to be considered in order to take advantage of it.

4.2 Integration with the Facebook

In addition, the restrictions resulting from the Facebook privacy policy – the user's personal data is not allowed to be stored by other applications – needed to be tackled. In consequence, in case user's data needs to be utilised, two independent requests to two components (the Facebook and the Server) are required. In figure 4 we present an UML sequence diagram of displaying statistics of the user's friends. This is a good example of the enforced redundancy in HTTP calls. The first call occurs when the user visits the appropriate *Taste It! Try It!* webpage. The user has to be authorised by the Facebook, so the first call occurs when the Facebook client requests Facebook for user credentials. Therefore, the Facebook client uses FB API interface to request Facebook for user authentication and authorisation (see fig. 1 and 4).

When the user is authorised, the Facebook client uses the Google Web Toolkit client-server communication interface to retrieve a user profile from the Server. These two calls have to be performed each time – it is significantly slowing down the loading of particular web pages (user profile, review list, search form, friends statistics and user settings). Although these calls are inherently asynchronous, the application has to wait for the results of the former call to perform the latter one. Thus, they are executed as if they were synchronous.

As has been already mentioned, according to the Facebook privacy policy the application is not allowed to store a list of user's friends. Thus, the list as well

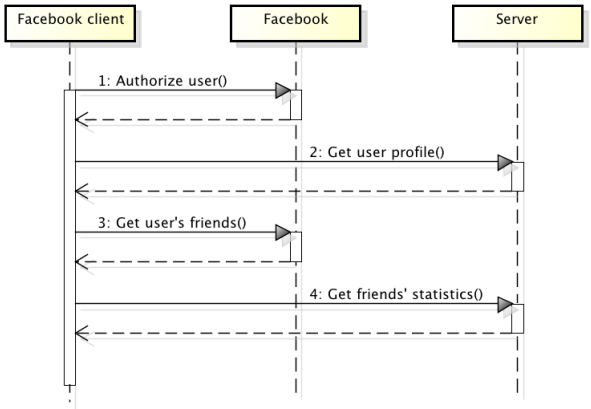

Fig. 4. UML sequence diagram of retrieving statistics of friends using the *Taste It! Try It!* application

has to be retrieved each time from the Facebook portal. During the third call the Facebook client uses the FB API to obtain the list of friends' Facebook IDs. The fourth call, when the Facebook client calls the Server, is to retrieve statistics of user's Facebook friends, who are using the *Taste It! Try It!* application.

A similar sequence of calls occurs each time the application requires additional information from the Facebook portal and Server.

5 CONCLUSIONS

The *Taste It! Try It!* application, presented in this paper, is a semantic content creation tool for a mobile device. It is to support users in the process of creation of semantically annotated reviews of various venues. It uses DBpedia as a source of data and is integrated with the Facebook portal. In this paper, we have shown how the application is consuming the Linked Data, and how additional semantic annotations are created. As they are to be made available outside the *Taste It! Try It!* application, they are anonymous and the subjective evaluation of the venue is expressed in an aggregated form.

In order to verify the mechanisms applied within the application as well as evaluate the usability of the developed tool, experiments with 180 participants were conducted. The evaluation of the proposed solution has shown that the application constitutes a good compromise between the power of semantic annotations and the difficulty of creating and maintaining them, and in addition, allowed to identify new directions of the application evolution.

During the implementation of the application, a number of problems related to the lack of maturity of the semantic technologies has been encountered. They show that apart of the already well known problems with the federated queries, also the problems related to the quality of the linked data, its coverage, as well

as the performance of the semantic technologies, hamper greatly the evolution of the Web. Users will not accept fully the Semantic Web paradigm, if semantic applications will burden them with additional interactions with RDF or SPARQL technologies, or will exhibit lower performance than the traditional applications.

References

[Adomavicius and Tuzhilin, 2005] Adomavicius, G. and Tuzhilin, A. (2005). Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. *IEEE Trans. on Knowl. and Data Eng.*, 17:734–749.

[Berners-Lee et al., 2001] Berners-Lee, T., Hendler, J., and Lassila, O. (2001). The semantic web. *Scientific American*, 284(5):34–43.

[Bizer et al., 2009] Bizer, C., Heath, T., and Berners-Lee, T. (2009). Linked data - the story so far. *International Journal on Semantic Web and Information Systems*, 5(3):1–22.

[Burke, 2002] Burke, R. (2002). Hybrid recommender systems: Survey and experiments. *User Modeling and User-Adapted Interaction*, 12:331–370.

[Burke, 2007] Burke, R. (2007). Hybrid web recommender systems. In *The Adaptive Web: Methods and Strategies of Web Personalization*, chapter 12, pages 377–408. Berlin.

[Cantador and Castells, 2006] Cantador, I. and Castells, P. (2006). Multilayered semantic social network modeling by ontology-based user profiles clustering: Application to collaborative filtering. In *EKAW*, pages 334–349.

[Cantador et al., 2011] Cantador, I., Castells, P., and Bellogn, A. (2011). An enhanced semantic layer for hybrid recommender systems: Application to news recommendation. *Int. J. Semantic Web Inf. Syst.*, 7(1):44–78.

[Gruber, 1995] Gruber, T. (1995). Toward principles for the design of ontologies used for knowledge sharing. *International Journal of Human-Computation Studies*, 43:907928.

[Hausenblas, 2009] Hausenblas, M. (2009). Exploiting linked data to build web applications. *IEEE Internet Computing*, 13:68–73.

[Millard et al., 2010] Millard, I., Glaser, H., Salvadores, M., and Shadbolt, N. (2010). Consuming multiple linked data sources: Challenges and experiences. In *Proceedings of the First International Workshop on Consuming Linked Data (COLD2010)*, Shanghai, China.

[Peis et al., 2008] Peis, E., del Castillo, J. M. M., and Delgado-Lpez, J. A. (2008). Semantic recommender systems. analysis of the state of the topic. *Hipertext.net*, 6:(online).

[Pu et al., 2012] Pu, P., Chen, L., and Hu, R. (2012). Evaluating recommender systems from the user's perspective: survey of the state of the art. *User Modeling and User-Adapted Interaction*, pages 1–39. 10.1007/s11257-011-9115-7.

[Umbrich et al., 2010] Umbrich, J., Hausenblas, M., Hogan, A., Polleres, A., and Decker, S. (2010). Towards dataset dynamics: Change frequency of linked open data sources. In *Proceedings of the WWW2010 Workshop on Linked Data on the Web (LDOW2010)*.

[Uschold and Grüninger, 1996] Uschold, M. and Grüninger, M. (1996). Ontologies: principles, methods, and applications. *Knowledge Engineering Review*, 11(2):93–155.