Growing the Semantic Web with Inverse Semantic Search

Hans-Jörg Happel

FZI Research Center for Information Technology, Karlsruhe, Germany
happel@fzi.de

Abstract. Many use cases for the Semantic Web assume the availability of public metadata. However, research has not yet addressed in a satisfactory manner why and how metadata is published on the Semantic Web. We analyze several reasons and barriers for creating and sharing semantic metadata. In particular, we address the issue of how metadata from private spaces can diffuse into the public Semantic Web. Therefore we introduce the concept of inverse semantic search – an approach which aggregates information needs to motivate information providers to share private metadata.[1]

1 Introduction

The vision of the Semantic Web [1] describes a web populated by machine-understandable metadata based on which agents can reason and act to fulfill tasks for human users. However, the realization of the Semantic Web largely depends on the availability of such structured metadata.

While the usefulness of metadata has been claimed for many domains and applications, publicly available metadata in the Semantic Web is still scarce. Two main issues impede a widespread success of metadata [2]. First, metadata is additional, descriptive data on top of actual information resources by its very nature. Thus, it is not created for self-purpose but costs additional effort. Secondly, the creation of metadata often implies a disparity of providers and beneficiaries (i.e. people using metadata are different from people creating it) and between the time of creation and its use [3].

While a number of studies have investigated the forces that drive the creation of metadata by individual users (mostly w.r.t. tagging systems, c.f. section 2.2), there exists no unified theory why semantic metadata is created and how it is made available [2]. Especially the Semantic Web vision does not address the *creator* side of metadata, but focuses on the consumer side and its applications. This is quite similar to the domain of information retrieval, which also neglects the role of information *providers.*

Within this paper, we will line out an initial theory about why and how metadata is created and thus how the Semantic Web could be populated. We

[1] This work has been supported in part by the TEAM project, which is funded by the EU-IST programme under grant FP6-35111 and the BMBF-funded project WAVES.

therefore analyze different aspects of metadata in the following section. Based on those insights, we present the conceptualization and realization of our approach called *inverse semantic search*, which guides potential metadata providers using aggregated information needs. We claim that the design of inverse semantic search thus provides motivational incentives to help growing the Semantic Web.

2 On metadata

2.1 Usage of metadata

We distinguish two major scenarios that motivate the usefulness of metadata in the Semantic Web. The most prominent one is *resource description for information retrieval*. The need for metadata in this scenario stems either from resources which are not accessible by standard keyword-based search (i.e. photos or videos), or from the fact that resources might not contain certain keywords/conceptualizations by which they might be accessed. Metadata is thus added to provide descriptive information which can incorporate structured classifications (like in library catalogues) or synonym keywords.

The second case for metadata is rooted in *task automation*. This comprises a whole range from visionary agent-driven scenarios which automatically perform actions on behalf of their human owners down to mash-ups where data from different sources is joined to provide some extra functionality [4].

2.2 Creation of metadata

We distinguish three different ways of creating metadata: 1) either it comes for free and just needs to be exposed, 2) it can be generated automatically or 3) has to be created manually.

The *exposition* case is the most simple one. If data is already available in some highly structured form, such as in database systems, it can easily be exposed. An example for this could be a cinema which offers metadata about available films out of its existing booking system. Although supporting tools already exist (e.g. [5]) an initial technical investment might be necessary to make such data available for external users.

The *automatic creation* of metadata tries to generate descriptive metadata using certain algorithms. Typical examples are machine learning systems which analyze documents, pictures or other content to automatically assign topics or categories. Such techniques depend on the availability of sophisticated algorithms, suitable input and training data and suffer from potential impreciseness [6]. Furthermore, they can not create arbitrary metadata (e.g. movie ratings or reviews). Automatic metadata creation techniques are therefore often used semi-automatically to assist human metadata creators.

Despite of its cost, *human created metadata* is thus still an important issue. While human metadata creation has been common for specific tasks such as library management, it has seen a renaissance in recent years due to the emerging

Web 2.0 phenomenon. Applications like del.icio.us[2] or Flickr[3] collect small pieces of metadata from individual users and unfold their power by aggregating them.

Motivational issues have been discussed concerning tagging and photo sharing systems in recent years. Results highlight the important role of personal and social benefits as functional motivations [3, 7, 6]. However, tagging systems can not be directly compared to general metadata for the Semantic Web. The authors of [2] discuss motivations for metadata sharing on a more abstract level, identifying advertising and retrieval services as potential contributors.

2.3 Visibility of metadata

Even if metadata has been created and is in place, it needs to be available for all its potential consumers. Like any kind of digital resource, metadata can be kept in arbitrary spheres of access – ranging from the private sphere of an individual user up to public visibility in the internet.

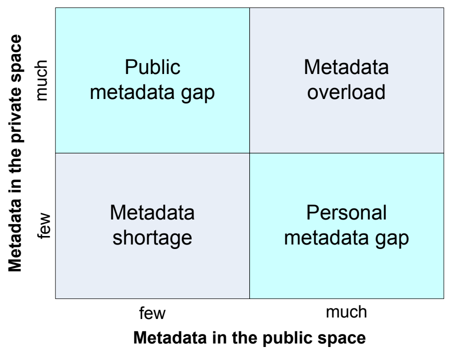

Fig. 1. Possible distributions of metadata in private vs. public information spaces (adapted from [8])

Private spheres are commonly used because users often hesitate to share data openly. Reasons are low motivation due to a lack of personal benefit [9–11], privacy concerns [12, 13] and effort for sharing (e.g. capturing, categorization and setting access rights) [13, 14]. Thus, even many open "Web 2.0" applications such as Flickr or del.icio.us allow for storing metadata privately.

Figure 1 illustrates this situation. It distinguishes the amount of metadata available for a certain information resource in the private space of a particular user vs. the public space. Four general situations are depicted: in a balanced situation, there either exists few metadata (*Metadata shortage*) or lots of metadata (*Metadata overload*) in both, the private and public space. If there is more

[2] http://del.icio.us
[3] http://www.flickr.com

metadata in the public space than in the private space, we call this a *personal metadata gap*. The case of a *public metadata gap* describes that no or only few metadata concerning an information resource exists in the public space, but in the private space of at least one particular user.

When considering the Semantic Web, the situations of a *public metadata gap* and *metadata shortage* are the most unfortunate ones, since potentially useful metadata is hidden in private spaces or does not exist at all.

2.4 Conclusion

Contrasting this section with the vision of the Semantic Web, the fact that the Semantic Web so far neglects the perspective of metadata *providers* has two consequences:

- The creation of metadata should be guided resp. focused, since it is a costly process.
- Feedback channels and easy sharing facilities should be incorporated in Semantic Web tool design.

In the following section, we will describe a general framework and an approach called *inverse semantic search* which tries to address incentives for sharing and creating semantic metadata.

3 Inverse semantic search

In this section we will describe a concept called *inverse semantic search* in order to help growing the amount of metadata in the Semantic Web. Therefore, we differentiate between *consumers* and *providers* of semantic metadata.

We will begin the section with a motivating scenario for our approach, followed by a specificiation of requirements and use cases. Afterwards, we will describe the realization of *inverse semantic search* in terms of its architecture and process steps.

3.1 Motivational example

As-is situation Our scenario involves two persons: Chrissy, who wants to buy a birthday present for her boyfriend, and Dave, who is a movie enthusiast. Chrissy's initial idea is to buy a trip to one of the locations mentioned in the movie "Casablanca" of which her boyfriend is a big fan. Thus, Chrissy queries her favourite Semantic Web search engine for "All locations mentioned in Casablanca". To her surprise, the application only returns the obvious "Casablanca" as a result – no additional metadata seems to be available on the web. However, Dave maintains his own local movie application, where he keeps data about his favourite films. His application actually contains "Paris" and "Lissabon" as additional locations mentioned in Casablanca. Since this data is within Dave's private space, Chrissy is not able to retrieve that information. Thus, she finally decides to buy a different birthday present.

To-be situation In order to improve knowledge sharing in the described situation, we propose that Chrissy's query is not just matched against the available metadata corpus (yielding only one result in our example), but also stored in a central query log. This information can then be made available to interested clients. Thus, Dave's movie application can retrieve this list of queries and automatically compare it to the metadata in his private space. In our example, this would reveal that information from Dave's computer could help satisfying Chrissy's information need. The movie application would present a list of metadata items to Dave, indicating that there is an information need that can be satisfied by sharing them. Dave may then choose to contribute this metadata to the public space. Once Dave shares the information, Chrissy could be notified about the new results.

Clearly this is a rather simplified example, which could probably be solved without any Semantic Web technologies at all. However, it illustrates the key principle of sharing and matching information needs asynchronously which is also applicable to scenarios utilizing more structured metadata.

3.2 Specification

In this section we specify our envisioned functionality by introducing a number of use cases and non-functional requirements.

Requirements As lined out in section 2, metadata provision suffers from a number of barriers. We want to address these barriers by satisfying the following set of non-functional requirements:

R1. Retain privacy An information provider must not expose information to others by default. Knowledge sharing systems often lack acceptance, since contributing information to the public space means losing control about it. However, many information providers want to retain such control, since information might be premature or sensitive [15].

R2. Minimize effort The effort for both, information providers and information seekers should be minimized. There should not be much redundant information provision [6].

R3. Motivate to share Information providers should be motivated to share relevant information with information seekers. Traditional knowledge sharing applications usually require to share information without signaling any benefit to the provider. Thus, those practices are often perceived as self-purpose with an unclear value. In opposite to this, we want to give the potential information provider more concrete information that can help to estimate the benefit of sharing certain metadata. Research targeting movie rating systems has shown that design features motivated by social psychology such as highlighting the uniqueness [16] or value [17] of a contribution can significantly increase information provision.

Use cases Metadata is typically queried in structured query languages such as SPARQL [18]. For the scope of this paper, we restrict ourselves to a fragment which allows to query for either *instances* or *literal values*.

Information need	Informal	Semi-formal
L	Chrissies Phone number	?x: ns:Chrissie ns:phoneNumber ?x
L	Speed of all cars	?x: ?y rdf:type ns:Car . ?y ns:hasSpeed ?x
I	Locations mentioned in Casablanca	?x: ?x ns:mentionedIn ns:Casablanca
I	All movies	?x: ?x rdf:type ns:Movie
I	Chrissies birth town	?x: ns:Chrissie ns:bornIn ?x
I	All videos tagged with "Chrissie"	?x: ?x ns:hasTag "Chrissie"
I	Places where Popes were born	?x: ?y ns:bornIn ?x . ?y rdf:type ns:Pope
I	All persons that own a car	?x: ?x ns:owns ?y . ?y rdf:type ns:Car

Table 1. Use cases (*ns* stands for an arbitrary namespace)

Table 1 shows example queries to illustrate eight different kinds of triple patterns which we consider in this paper. The first column shows the type of information need (instances or literal). The second column contains a written description of the information need. In the last column, a simplified formal representation of these information needs is shown. It contains the queried variable (*?x*) followed by constraints on this variable. Constraints are either concrete values for object or datatype properties (e.g. *?x hasTag "Chrissie"*) or types of object property values (*?x rdf:type ns:Movie*).

3.3 Realization

In this section we give a short definition of our approach. We then discuss architectural implications and describe the process steps involved.

Definition As already lined out, common retrieval models follow a *provide first – retrieve then* approach. This means that they do not conceptualize the provision of information but assume that information exists at the time of retrieval. Information seekers can then query this information to retrieve results satisfying their information need.

The basic underlying idea of *inverse search* is that (potential) providers of information do not have to reveal or capture their information beforehand, but can use data about actual information needs to evaluate demand [19].

We thus conceptualize inverse search as information providers, matching their information against a given set of information needs - in opposite to conventional search, where information seekers match their information needs (i.e. queries) against a given set of information (i.e. documents). While users "import" public information into their private space in conventional search, inverse search helps to move information from the private space to the public space, where it might satisfy the information needs of other users.

When we talk about inverse *semantic search*, we consider SPARQL-like structured queries on structured RDF-like data[4]. SPARQL-like queries will be used to estimate the demand for certain metadata triples in a certain knowledge base. With inverse semantic search, we aim to address how and why metadata moves from private to public spaces. Besides sharing existing metadata, demand information can also be used to signal metadata which is not yet captured at all.

Architecture We will now describe a system architecture that supports the envisioned metadata sharing process.

In order to differentiate between metadata in public and in private spaces and to fulfill requirement R1, the system distinguishes a public metadata space ($MSpace_{Public}$) and a private metadata space for each user (e.g. $MSpace_{Chrissy}$ and $MSpace_{Dave}$), which is not accessible to any other user. Technically, this can be realized either by physical or logical separation. Physical separation means, that the private space is an independent system running on the local machine of a user (e.g. a Semantic Desktop system). Logical separation does not require two separate applications, but can be implemented as a feature in a server-based system – e.g. by offering "private" and "public" sharing options.

Queries to the public space are automatically saved to a public *query log*. Both, the public space and the query logs can be accessed by any user. In order to retain privacy (R1), queries may be anonymous and must not contain information about the querying user. However, if users like to receive automatic notifications when new metadata arrives, they might need to reveal their identity.

As functional modules, our approach requires a *SearchApplication* which allows to query both the local and the public space and a *SharingEngine*, which periodically compares the local with the public space and the query log. Again, this can be realized either within a web-application or by combining a public web-based space with an application running on a local machine.

Thus, the sharing engine can provide an estimation of how useful it would be to share certain metadata. This helps to satisfy requirement R3, since the user is guided in her decision which metadata is worth sharing. In order to minimize the effort of sharing (requirement R2), several ways are possible to suggest sharing certain metadata to the user. This might either happen by enriching existing interfaces (e.g. by blending metadata with information about its value [17]) or by periodically presenting a ranked list of sought metadata.

Process At runtime, inverse semantic search comprises a number of subsequent steps, which are lined out in the following. As in the example in section 3.1, this process starts with the collection of information needs and its aggregation. Afterwards, information needs are retrieved by potential information providers

[4] We are well aware that there are many other notions of semantic search

and matched against their private metadata. Finally, they can decide to share or create certain metadata, if it matches some demand.

Information need The information need of information seekers drives our knowledge sharing process. As it is the most convenient source of information needs, we will stick to *queries* resp. *query logs* as our main input and do not discuss other possible sources in this paper.

As driven by the use cases in section 3.2, we assume a fragment of common metadata query languages in the scope of this paper. Based on SPARQL, the most common and standardized language, queries in our approach are restricted concerning the free variables they can contain. We assume a single variable in the result set, which can have arbitrary constraints concerning object property values, literal values and property value types. We allow an additional second free variable to help defining type constraints for object properties. The resulting eight query archetypes haven been presented in Table 1. We now describe the internal ("query log") storage format for these queries.

Result type Instances or literals
List<Type> Type constraints for the resulting instance (?x rdf:type t; not applicable to queries for literals)
List<Instance, Property> List of tuples of instances and properties constraining the result (i, p, ?x)
List<Property, Object> List of tuples of properties and instances constraining the result (?x, p, i; not applicable to queries for literals)
List<Property, Literal> List of tuples of properties and literals constraining the result (?x, p, l; not applicable to queries for literals)
List<Property, Type> List of properties and their type constraining the result (?x, p, t; not applicable to queries for literals)
List<Type, Property> List of types and properties constraining the result (t, p, ?x)
Timestamp Timestamp of the query
User Concrete or abstract user id
Number of results Current number of results for the query in the knowledge base (i.e. size of the result set for the most recent query)

Each query archetype listed in Table 1 will be logged in one of the *List* fields of the log. Combinations of constraints will result in several entries. We refer to query instances in this log (Q) by using the variable i. A query q denotes a set of query instances i, which is similar in all fields except of timestamp, user and number of results.

Need aggregation Need aggregation targets the ranking of queries in terms of identifying those queries which information need is only badly satisfied by the underlying public knowledge base. We therefore apply two processing steps to the data in the query log.

First, identical queries are aggregated on a per-user basis to calculate a *personal information need.* For the sake of simplicity, we only consider identical queries in the scope of this paper (q; see above). Second, the different personal information needs concerning a particular query are aggregated into an *aggregate information need*. We shortly motivate this distinction, before we describe how to calculate these values.

The information need of a user is a primary subject of investigation in information retrieval (IR). The main purpose of IR systems is to help users satisfying their information needs by providing a set of relevant documents. A personal information need can be defined as information which a user requires to complete a specific task [20]. To use an IR system, the user typically has to express this information need in terms of the query language which can be interpreted by the search system. In most systems, this is a textual, "keyword-based" representation.

Based on this definition of personal information need, we conceptualize *aggregate information need* (AIN) as an aggregate of the personal information needs of members in a group. By *group* we mean the group of users which are able to access a certain public space. Depending on the concrete setup, this can be a team, an organization or the web as a whole. The aggregate information need thus denotes the overall amount of information which the members of this group require to complete their particular tasks.

Our basic rationale for computing the AIN is that it is higher, 1) the more often and more recently a term has been part of a query, 2) the more different users used the term in a query and 3) the more seldom a term is in the local space of the users. Based on this, we define the AIN as the weighted sum of individual information needs of the querying users.

We propose the following four measures as signals for an aggregate information need:

Frequency We assume that the AIN regarding a query is the higher, the more often it has been executed.

Availability If few results are returned for a query, the availability of metadata is low which indicates a higher demand. For availability, the number of results for a query is normalized into an interval $[0, 1]$.

Freshness Since the AIN regarding some query is a dynamic value, we also assume that the AIN is higher, the more recently the query has been executed. This allows recent information needs to score a relatively higher value.

Universality We define that an AIN is the higher, the more different users issued the same query. The rationale behind this is that an information provider may only receive a limited set of sharing recommendations (see section 3.2). Thus, in order to maximize the overall benefit for the organization, such metadata should be prioritized, which is relevant for a large number of different information seekers.

In order to formally define the AIN, we group the first three signals into a personal information need. Thus, the personal information need for a user concerning a specific query consists of the availability, frequency and freshness of queries:

$$PIN(q, user) = (1 - r) \cdot \frac{Q_{q,user}}{Q} \cdot (1 + \frac{Q_{q,user-recent}}{Q_{recent}}) \tag{1}$$

Accordingly, the AIN is the sum of the values for PIN, normalized by the total amount of querying users:

$$AIN(q) = \frac{Users_q}{Users} \cdot \sum_{user_q} PIN(q, user) \tag{2}$$

Further need aggregation could be done by aggregating structured queries using similarity measures leveraging taxonomic knowledge from a background ontology. This kind of aggregation could potentially be done at server- or client-side. However, further considerations in this direction are out of the scope of this paper.

Need retrieval The "query log" as presented before needs to be available for retrieval by interested metadata providers. Therefore, we define two major services:

List<InformationNeed> getTopInformationNeeds() returns the most desired information needs from the metadata repository under consideration.
List<InformationNeed> getInformationNeedsRelatedTo(URI) returns the information needs w.r.t. a certain instance URI.

Both services return a list of *InformationNeed* objects, which basically represent entries from the query log.

Local matching In the local matching step, the retrieved information need is matched against the private metadata of the information provider. Therefore, the *InformationNeed* objects are transformed back into SPARQL queries, where all attributes are marked as optional. The results are finally ranked by the number of constraints they fulfill. Again, more sophisticated matching approaches are possible, but their discussion is beyond the scope of this paper.

Sharing In terms of the actual user interaction for sharing, several possibilities exist. One option could be to embed the described knowledge sharing mechanism in an existing application (e.g. some kind of knowledge browser). This browser occassionally triggers the information need backend and matches it against the private metadata of the user. Once metadata is identified to be worth sharing, the user interface indicates this e.g. by highlighting the respective data. A concrete example could be a Semantic Wikipedia [21] browser, which identifies sought metadata for a browsed page.

The second option would be to provide an explicit sharing mechanism which presents the user a raw list of sought metadata. The user could then decide to generate this list (e.g. once a week or each time a certain program starts) and share respective metadata. A Semantic Wikipedia example could here be a list of desired metadata within the overall Wiki, similar to the existing list of "Wanted pages" in the MediaWiki software (Special:WantedPages).

4 Conclusion

In this paper, we addressed the issue of why and how metadata is provided for the public Semantic Web. In particular, we introduced a mechanism called *inverse semantic search* which targets to support *knowledge providers*. It is based on the principle of aggregating unsatisfied information needs in order to recommend the sharing or capturing of information. By considering requirements rooted in studies on knowledge sharing (c.f. section 3.2), our system design explicitly considers user incentives [7, 16, 6].

Since a concrete evaluation of this system would be a challenge of its own, it was not in the scope of this paper and is left to future research. However, since related research has shown that meta-information can foster user contributions [17, 16], we are confident that our approach will have practical value. Evaluation would require to incorporate design choices based on different motivational factors into the user interface which allows to test according hypothesises at system runtime (similar to [16]).

Regarding the level of granularity, our discussion was based on the vision of the Semantic Web as such. However, we think that our approach can also be beneficial in more restricted settings such as organizations or teams. Furthermore, the described mechanism could also be built into applications such as Semantic Wikis to guide and foster metadata generation.

Finally, this paper focused on describing the general motivation, architecture and design principles of inverse semantic search. Several technical issues such as the modelling of information needs based on more complex structured queries or the semantic aggregation of queries should be addressed by future work.

References

1. Berners-Lee, T., Hendler, J., Lassila, O.: The semantic Web. Scientific American **284**(5) (May 2001) 34–43
2. Thomas, C.F., Griffin, L.S.: Who will create the metadata for the internet? First Monday **3**(12) (1998)
3. Ames, M., Naaman, M.: Why we tag: motivations for annotation in mobile and online media. In: CHI '07: Proceedings of the SIGCHI conference on Human factors in computing systems, New York, NY, USA, ACM (2007) 971–980
4. Ankolekar, A., Krötzsch, M., Tran, T., Vrandecic, D.: The two cultures: mashing up web 2.0 and the semantic web. In: WWW '07: Proceedings of the 16th international conference on World Wide Web, New York, NY, USA, ACM (2007) 825–834
5. Bizer, C., Cyganiak, R.: D2r server-publishing relational databases on the semantic web (poster). In: International Semantic Web Conference. (2006)
6. Kustanowitz, J., Shneiderman, B.: Motivating annotation for personal digital photo libraries: Lowering barriers while raising incentives. Technical Report HCIL-2004-18, University of Maryland, College Park, MD, USA (01 2005)
7. Marlow, C., Naaman, M., Boyd, D., Davis, M.: Ht06, tagging paper, taxonomy, flickr, academic article, to read. In: HYPERTEXT '06: Proceedings of the seventeenth conference on Hypertext and hypermedia, New York, NY, USA, ACM (2006) 31–40

8. Happel, H.J., Stojanovic, L.: Analyzing organizational information gaps. In: Proceedings of the 8th Int. Conference on Knowledge Management. (2008) 28–36
9. Cress, U., Hesse, F.W.: Knowledge sharing in groups: experimental findings of how to overcome a social dilemma. In: ICLS '04: Proceedings of the 6th international conference on Learning sciences, International Society of the Learning Sciences (2004) 150–157
10. Cabrera, A., Cabrera, E.F.: Knowledge-sharing dilemmas. Organization Studies **23** (2002) 687–710
11. Wasko, M.M., Faraj, S.: Why should i share? examining social capital and knowledge contribution in electronic networks of practice. MIS Quarterly **29**(1) (2005) 35–57
12. Ardichvili, A., Page, V., Wentling, T.: Motivation and barriers to participation in virtual knowledge-sharing communities of practice. Journal of Knowledge Management **7**(1) (2003) 64–77
13. Desouza, K.C.: Barriers to effective use of knowledge management systems in software engineering. Commun. ACM **46**(1) (2003) 99–101
14. Desouza, K.C., Evaristo, J.R.: Managing knowledge in distributed projects. Commun. ACM **47**(4) (2004) 87–91
15. Orlikowski, W.J.: Learning from notes: organizational issues in groupware implementation. In: CSCW '92: Proceedings of the 1992 ACM conference on Computer-supported cooperative work, New York, NY, ACM Press (1992) 362–369
16. Beenen, G., Ling, K., Wang, X., Chang, K., Frankowski, D., Resnick, P., Kraut, R.E.: Using social psychology to motivate contributions to online communities. In: CSCW '04: Proceedings of the 2004 ACM conference on Computer supported cooperative work, New York, NY, USA, ACM (2004) 212–221
17. Rashid, A.M., Ling, K., Tassone, R.D., Resnick, P., Kraut, R., Riedl, J.: Motivating participation by displaying the value of contribution. In: CHI '06: Proceedings of the SIGCHI conference on Human Factors in computing systems, New York, NY, USA, ACM (2006) 955–958
18. Prud'Hommeaux, E., Seaborne, A.: SPARQL query language for RDF. World Wide Web Consortium, Recommendation REC-rdf-sparql-query-20080115 (January 2008)
19. Happel, H.J.: Closing information gaps with inverse search. In: Practical Aspects of Knowledge Management, 7th International Conference. Lecture Notes in Computer Science, Springer (2008) 74–85
20. Baeza-Yates, R., Riberio-Neto, B.: Modern Information Retrieval. ACM Press (1999)
21. Völkel, M., Krötzsch, M., Vrandecic, D., Haller, H., Studer, R.: Semantic wikipedia. In: WWW '06: Proceedings of the 15th international conference on World Wide Web, New York, NY, USA, ACM (2006) 585–594