The Semantic Web: Collective Intelligence on the Web

Maciej Janik, Ansgar Scherp, and Steffen Staab

`lastname@uni-koblenz.de`
Institute for Web Science and Technologies, WeST
University of Koblenz-Landau

Abstract. The World Wide Web has turned hypertext into a success story by enabling world-wide sharing of unstructured information and informal knowledge. The Semantic Web targets the sharing of structured information and formal knowledge pursuing objectives of achieving collective intelligence on the Web. Germane to the structure of the Semantic Web is a layering and standardization of concerns. These concerns are reflected by an architecture of the Semantic Web that we present through a common use case. Semantic Web data for the use case is now found on the Web and is part of a quickly growing set of Semantic Web resources available for formal processing.

1 Introduction

"Which type of music is played in UK radio stations?" and "Which radio station is playing titles by Swedish composers?" are the type of questions that are very hard to answer using existing Web search engines. However, the upcoming Semantic Web provides better framework to easier answer such questions. The information required to answer these questions is already available. In fact, we will show that a large amount of it already exists in formats amenable to machine processing on the Semantic Web. The reason that Web search engines fail at answering such questions is that they are limited to analyzing Web content, mostly documents in natural language, one page at a time, while the Semantic Web allows for combining data that is distributed across many different sources and described in a machine-interpretable manner.

Let us consider how we may pursue answering the questions from above. Playlists of BBC radio shows are published online in Semantic Web formats. A music group such as "Abba" has an identifier (`http://www.bbc.co.uk/music/artists/d87e52c5-bb8d-4da8-b941-9f4928627dc8#artist`) that may be used to relate the music group to information at MusicBrainz[1], a music community portal exposing data on the Semantic Web. MusicBrainz knows about band members such as Benny Andersson as well as about genre of artists and songs. In addition, MusicBrainz aligns its information with Wikipedia, e.g., to be able to include the biography of an artist, or to add facts from DBpedia [2], a version

---

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

of Wikipedia in Semantic Web formats. Information about UK radio stations may be found in lists on web pages[2], which can be translated to a similar Semantic Web representation—descriptions of things and their relationships. MusicBrainz is connected via the Playcount ontology[3] with the broadcasting center of the BBC. Data at the BBC currently uses at least nine different ontologies, with varying degree of formality and different relationships to each others. The meaning of such relationships is explained online, too, using a set of ontologies that is available from the Semantic Web[4] such as Dublin Core[5] for describing general properties of information resources and specialized ontologies covering the music domain. Given the available data, one may answer questions about frequency of music genres played on UK radio stations, radio stations playing Swedish composers, and many more. Having access to and leveraging such data does not come for free. The outlined scenario and likewise other use cases require generic software components, languages, and protocols that must interact in a seamless manner to be able to answer such requests.

In this article, we explore how such example data may be managed (Section 3), structured (Section 4) and reasoned upon (Section 5) using ontologies. Common to these explorations is the nature of having data provided in a collective manner on the Web leading to new challenges such as provenance and trust (Section 7) and generic user interfaces (Section 8). In order to shape a picture of how these different requirements and building blocks fit together, we start by sketching an underlying Semantic Web architecture.

2 Architecture for Representing Knowledge on the Web

The scenario above illustrates *what* the Semantic Web as an infrastructure targets to achieve leaving open *how* to achieve its objectives. In fact, traditional knowledge systems with artificial intelligence capabilities already exhibited many of the required capabilities. However, traditional knowledge representation systems exhibited a lack of flexibility, robustness, and scalability. To quite some extent this problem stemmed from a lack of maturity together with a large complexity of the algorithmic methods. For instance, description logics systems, which are now the backbone of Web Ontologies, were severely limited in scale, typically capable of handling not more than a few hundred concepts in the mid 1990's (cf. [28]). In the meanwhile, several such problems have been overcome using much increased computational power as well as better understood and optimized algorithms leading to the practical handling of large ontologies such as SNOMED[6] with several hundreds of thousands of axioms (cf. (Baader et al., this issue)). Nevertheless, several severe practical problems have remained that are targeted by Semantic Web developments.

---

[2] For example: `http://www.listenlive.eu/uk.html`

[3] `http://dbtune.org/bbc/playcount/`

[4] `http://www.bbc.co.uk/ontologies/programmes`

[5] `http://dublincore.org/documents/dc-rdf/`

[6] `http://nbirn.net/research/ontology/snomed.shtm`

Bottlenecks of traditional knowledge representation for managing data and semantics revolve around issues about the large number of data sources with varying (1) underlying technologies, (2) geographically dispersed locations, (3) authorities, (4) service quality, and (5) high change rate of stored information. Thus, in analogy to the World Wide Web, the Semantic Web requires a computing mega system with the following five characteristics:

1. *Explicit, Simple Data Representation:* A common data representation should hide the underlying technologies and only capture the gist of the underlying data representations.
2. *Distributed System:* A fully distributed mega system of data sources without centralized control of who owns what type of information. Distributed ownership and control, if done properly, facilitates adoption and scalability.
3. *Cross-referencing:* In order to benefit from the network beyond the mere sums of its parts, the data must be cross-linked allowing for re- and cross-use of existing data and existing data definitions from different authorities.
4. *Loose Coupling with Common Language Layers:* In a mega system the components have to be only loosely coupled by communicating in standardized languages. The standardized languages must come with great flexibility such that they may be customized for specific systems, but the overall communication must not be jeopardized by such specialization.
5. *Ease of Publishing and Consumption:* A mega system that allows for easy publishing and consumption of simple data as well as for comprehensive publishing and consumption of complex data.

The required capabilities are provided by different components. A core set of such components have been standardized by the W3C specifying the formal semantics of exchange languages and protocols. Further components are still not agreed upon, but are foreseen in the Semantic Web layer cake drawn up by Tim Berners-Lee at `http://www.w3.org/2007/03/layerCake.png`. We depict a view of the Semantic Web in Figure 1 reflecting the status quo and distinguishing between language standards and future target capabilities. Each component is introduced briefly, starting at the bottom and working our way up.

HTTP/URI/IRI. Given the decentralised nature of the Semantic Web, data publishers require a way to identify entities unambiguously. Entities, also called ‘resources’, on the Internet are identified with Uniform Resource Identifiers (URIs)[3]. URIs on both the Web and the Semantic Web typically use identifiers based on HTTP, which allows for piggybacking on the Domain Name System (DNS) to ensure the global uniqueness of domain names and hence URIs. In our example, the URI `http://www.bbc.co.uk/music/artists/2f031686-3f01-4f33-a4fc-fb3944532efa#artist` denotes Benny Andersson of ABBA. Implicit in the use of URIs is a mechanism for retrieving content; assuming a HTTP URI denoting ABBA, a user has the ability to *dereference* the URI, that is, perform a lookup using HTTP and retrieving content which is associated with the URI. Internationalized Resource Identifiers (IRI) [12] complement URIs and allow for the use of characters from a large range of writing systems.

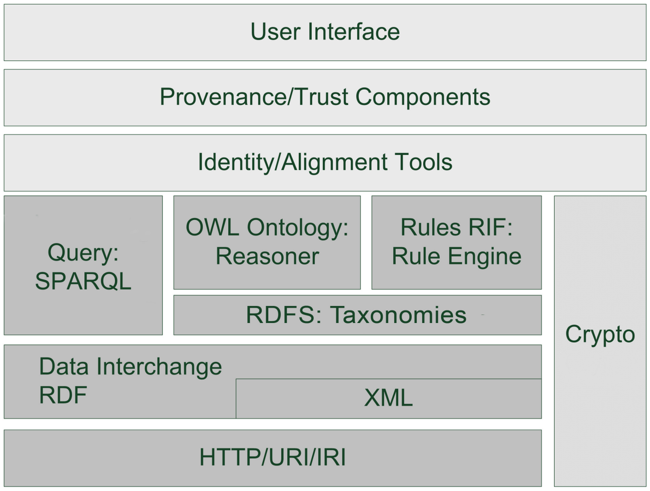

Fig. 1. Semantic Web Layer Cake: W3C language standards are depicted in dark grey. Further targeted capabilities are indicated in light grey.

XML. Having the ability to point to resources unambiguously and dereference them is a first step. Next, we require a language to exchange description of resources. The Extensible Markup Language (XML) is used for encoding documents and provides means for specifying and serializing structured documents.

RDF. Having a referencing and a document exchange mechanism, we require means to encode descriptions about resources. Given that the data on the Semantic Web is highly distributed, the description of resources should be encoded in a way that facilitates integration from multiple sources. A graph-structured data format [34] achieves the easy integration of data from multiple sources. The W3C standard for encoding such data is the Resource Description Framework (RDF). RDF graphs can be serialized in multiple ways; one of the most commonly used is the XML serialization mentioned above.

SPARQL. Having integrated data we require means for querying the integrated graphs. SPARQL[7] (a recursive acronym for SPARQL Protocol and RDF Query Language) is a declarative query language similar to SQL and allows for specifying queries against data in RDF.

RDFS. Encoding data as graph covers only parts of the meaning of the data. Often, constructs for modelling class or property hierarchies provide machines and, subsequently, humans a more sapient understanding of data. In order to provide a comprehensive model for a domain of interest, ontology languages can be employed. RDF Schema (RDFS) is a language which can be used to express class and property hierarchies as well as domain and range of properties [9].

---

[7] `http://www.w3.org/TR/rdf-sparql-query/`

OWL. Data originates from multiple sources and is thus highly heterogeneous. RDFS is not expressive enough to efficiently reconcile data from different sources and to check consistency of combined data. The Web Ontology Language (OWL) provides such means for authoring ontologies. It allows, e.g., for specifying equality of resources or cardinality constraints of properties [24]. Ontology languages allow for automated inferences, i.e., drawing conclusions based on existing facts.

RIF. An alternative way for specifying logical inferences are rules. Often, users require to specify rules which transform data. To this end, the Rules Interchange Format[8] (RIF) allows for encoding logical rules for exchange [6] .

Crypto. Future capabilities are encryption and authentication technologies to ensure that data transmissions cannot be intercepted and read or altered. Crypto modules such as Secure Socket Layer processors verify digital certificates and provide cryptographic privacy and authentication.

Identity and Alignment. Content aggregated from a large number of sources often use multiple identifiers to denote the same real-world object. An integration and alignment layer provides for consolidation and tighter integration of data.

Provenance and Trust. A provenance and trust layer analyzes the data in conjunction with additional information to provide the user with a notion of trust associated to individual data items.

User Interfaces. Finally, the user interface allows for users to interact with Semantic Web data. From a functionality viewpoint, some user interfaces are generic and operate on the graph structure of the data, while others are tailored to a certain task, domain or ontology. Novel paradigms are currently investigated that can trade-off between the generality required from changing schema and the dedicated needs of end users.

3 Distributed Data on the Web

The Semantic Web exploits several logical paradigms (cf. Section 5). However, its main emphasis is on relationships between resources. While Artificial Intelligence concentrates mostly on logic reasoning and individual intelligence, the Semantic Web goes towards collective intelligence, distributed data, and interlinking of information. The Semantic Web can be characterized by knowledge sharing, intensive collaboration, cooperation, and information processing activities. Anybody can create an ontology, link it to other data sources, and finally, make it available on the Web for others to re-use. The large mass of interlinked information can lead to discovery of previously unknown relations and facts. Such collective knowledge must be supported by efficient access to distributed data, appropriate publishing guidelines, and querying capabilities.

---

[8] `http://www.w3.org/2005/rules/wiki/RIF_Working_Group`

3.1 Linked Data

Data can be exposed and shared using different techniques, yet simplicity of accessing information stimulates linking activity. One of the approaches is to implement data access using *Linked Data* principles[9]. They can be summarized with the following guidelines:

- use URIs as names for things,
- use HTTP URIs so that they can be looked up,
- provide useful information for URI lookup using standards (RDF, SPARQL),
- include links to other URIs to discover more things.

Publishing data according to Linked Data principles allows for accessing them remotely in a very easy way (just using HTTP). This facilitates navigation and exploration of semantic resources on the web. Most important is a possibility to dereference an URI by posting an HTTP query and obtaining additional information associated with the resource, especially other resources that it links to. An example of linked data about ABBA taken from MusicBrainz is presented in Figure 2. It defines different properties that relate entities to the URI representing ABBA like `foaf:member` and `rdf:type`. Please note the `owl:sameAs` property that enables linking to other datasets, which will be presented further below.

ABBA

Resource URI: **http://dbtune.org/musicbrainz/resource/artist/d87e52c5-bb8d-4da8-b941-9f4928627dc8**

Property	Value
vocab:alias	Abba
vocab:alias	Björn + Benny + Anna + Frieda
bio:event	<http://dbtune.org/musicbrainz/resource/artist/d87e52c5-bb8d-4da8-b941-9f4928627dc8/birth>
bio:event	<http://dbtune.org/musicbrainz/resource/artist/d87e52c5-bb8d-4da8-b941-9f4928627dc8/death>
foaf:homepage	<http://www.abbasite.com>
foaf:member	<http://dbtune.org/musicbrainz/resource/artist/042c35d3-0756-4804-b2c2-be57a683efa2>
foaf:member	<http://dbtune.org/musicbrainz/resource/artist/2f031686-3f01-4f33-a4fc-fb3944532efa>
foaf:member	<http://dbtune.org/musicbrainz/resource/artist/aebbb417-0d18-4fec-a2e2-ce9663d1fa7e>
foaf:member	<http://dbtune.org/musicbrainz/resource/artist/ffb77292-9712-4d03-94aa-bdb1d4771d38>
mo:musicbrainz	<http://musicbrainz.org/artist/d87e52c5-bb8d-4da8-b941-9f4928627dc8>
foaf:name	ABBA
owl:sameAs	<http://dbpedia.org/resource/ABBA>
owl:sameAs	<http://sv.wikipedia.org/wiki/Abba>
owl:sameAs	<http://www.bbc.co.uk/music/artists/d87e52c5-bb8d-4da8-b941-9f4928627dc8#artist>
rdf:type	mo:MusicArtist

Fig. 2. Linked data example for ABBA.

---

[9] `http://www.w3.org/DesignIssues/LinkedData.html`

Over the last two years the publication of Linked Data has proliferated especially in the area of bibliographic information, life-science data, and e-government data. Thereby, DBpedia acts as a nucleus connecting these different domains (cf. [4]).

3.2 SPARQL Query Language

The Linked Data principles, however, do not support posting more complex queries. In order to handle complex queries to RDF knowledge bases, the commonly used standard is SPARQL. Like in SQL, the `WHERE` clause defines how data is joined. Additionally, it specifies what graph templates are matched during the search process. Similar to specifying SQL tables in the query, users can specify the source of data by selecting a specific named graph.

In addition to a query language, SPARQL defines the access protocol and interoperability data formats. A repository that supports SPARQL must implement querying of the underlying data using a specific syntax and protocol. The most popular repositories are Sesame [10], Jena [47], Virtuoso[10] or OWLIM[11]. They do not only store and allow for accessing graphs encoded in RDF, but also support inference and rules. SPARQL enables only defining specific graph templates for querying. The repositories are responsible for supporting the semantics of RDF/S or OWL (cf. Section 5) and implementing the appropriate inference models using specialized reasoners. Datasets exposed as a *SPARQL endpoint* to the outside world are accessible using the REST protocol [16].

Let us refer back to the example from MusicBrainz where we were looking for information about ABBA. We would like to know if singers from ABBA were also members of other music bands. Using only Linked Data access, we would have to navigate from the URI representing the music group ABBA to each of its members and later link to all bands she or he was a member of. SPARQL allows to specify patterns in the graph that should be matched in the knowledge base together with additional constraints. The graph pattern querying for different bands that musicians from ABBA were singing in is depicted in Figure 3 and a corresponding SPARQL query is presented in Figure 4.

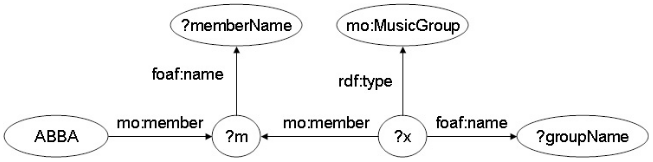

Fig. 3. Graph representing query for music groups that members of ABBA sing in.

---

[10] `http://virtuoso.openlinksw.com`

[11] `http://www.ontotext.com/owlim`

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX mo: <http://purl.org/ontology/mo/>
PREFIX foaf: <http://xmlns.com/foaf/0.1/>
PREFIX bbc: <http://www.bbc.co.uk/music/>
SELECT ?memberName ?groupName
WHERE  { bbc:artists/d87e52c5-bb8d-4da8-b941-9f4928627dc8#artist mo:member ?m .
         ?x mo:member ?m .
         ?x rdf:type mo:MusicGroup .
         ?m foaf:name ?memberName .
         ?x foaf:name ?groupName }
FILTER (?groupName <>  "ABBA")
```

Fig. 4. SPARQL query for music groups that members of ABBA sing in.

The core of `SPARQL` matching is defined in the `WHERE` clause. It defines the graph pattern that has to be matched against the knowledge base. The graph pattern consists of joined individual statements (subject, predicate, object) forming a template that is filled during the matching process. In this example, joining resources include the unknown band and members of ABBA. The `WHERE` clause is followed by a `FILTER` expression that narrows the returned results to those structures that fulfill specific criteria.

3.3 Querying Linked and Distributed Data

Distributed queries are not yet included in the SPARQL standard. The mediation architecture, index structures, and algorithms for executing distributed path queries were proposed in [45]. It was further refined and extended in Networked Graphs [41]. Networked Graphs do not only allow for querying of remote repositories in a unified manner and for building dynamic views of remote graphs but also for joining them together using recursive views (defined as `CONSTRUCT` queries) and applying rules.

An example query for the extended wikipedia pages of artists singing in ABBA is presented in Figure 5. Results join information about artists from two graphs: Musicbrainz and Wikipedia. Musician (`?member`) is the element joining two datasets as presented in Figure 6. Although URIs for the same musician in Wikipedia and MusicBrainz may be different, it is possible to define their equivalence by using `owl:sameAs` relationship and answer queries that cross datasets. From the user perspective, the query is executed by one SPARQL endpoint that extracts data from two named RDF graphs. Networked Graphs hide the complexity of remotely accessing multiple repositories and creating a joined result.

While the querying of distributed RDF sources described above is strongly related to querying of schemaless and distributed or peer-to-peer databases (cf. [25]), recently a novel paradigm for querying has been developed to directly take advantage of the availability of Linked Data. The querying paradigm proposed by Hartig et. al. [27] combines the basic property of Linked Data being URI dereferencable with data crawling. It overcomes the lack of data indexing and direct SPARQL querying by traversing of RDF links to discover information that may be relevant for the query during the query execution. The search

```
PREFIX mo: <http://purl.org/ontology/mo/>
PREFIX foaf: <http://xmlns.com/foaf/0.1/>
PREFIX bbc: <http://www.bbc.co.uk/music/>
CONSTRUCT { ?member mo:wikipedia ?biography . ?member foaf:name ?name}
FROM NAMED :Musicbrainz   FROM NAMED :DBpedia
WHERE {
    GRAPH :Musicbrainz {
        bbc:artists/d87e52c5-bb8d-4da8-b941-9f4928627dc8#artist mo:member ?member .
        ?member foaf:name ?name }
    GRAPH :DBpedia {
        ?member mo:wikipedia ?biography }
}
```

Fig. 5. SPARQL query for Wikipedia pages of ABBA members

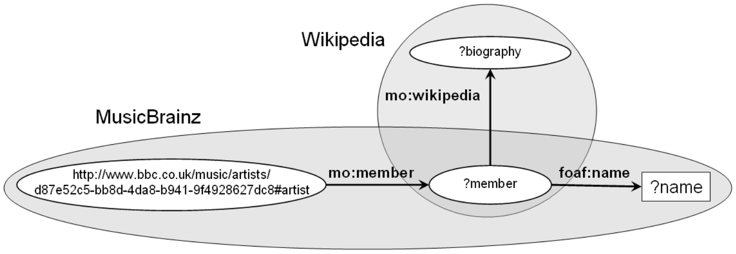

Fig. 6. Joined information about ABBA from two data sources

space is extended based on the given query pattern to next linked resources that can contribute to the final answer. It is accompanied by an indexing mechanism that utilized data summaries combining description of instance and schema-level elements [25]. Such indexing helps to select relevant data sources to answer the query that can span over multiple, initially possibly unknown, sites. The approach allows for querying also the Linked Data sources that do not offer SPARQL endpoints, however at the cost of being incomplete.

4 Ontologies on the Web

For modeling the scenario of interconnecting MusicBrainz with the BBC programme introduced in Section 1, several ontologies are involved and connected into a network of ontologies. Networked ontologies are a novel approach of interconnecting knowledge and spanning it over the web. An excerpt of the network of ontologies used in such a scenario is depicted in Figure 7. The artist from MusicBrainz is represented by the concept `MusicArtist` taken from the music ontology[12]. It is connected via an `object` property to the `Playcount` concept of the playcount ontology, which represents the number of times an artist is played. The playcount ontology connects via the `Brand` concept with the BBC ontology[13]. The BBC programme ontology allows for representing the `Brand` in

---

[12] `http://musicontology.com`

[13] `http://www.bbc.co.uk/ontologies`

the context of some `Service` in which the artist is played. Such a `Service` can be a customized version of the BBC programme for a specific region. It is broadcasted by some `Broadcaster`, which is a specialization of the `Organization` concept of the FOAF ontology. In addition, the event of a `Broadcast` is modeled in the BBC ontology, which is a specialization of the `Event` concept of the event ontology[14]. The `Broadcast` may be a `broadcast_of` some specific version of the broadcasted item such as a version of a song shortened for a radio programme. The `Version` has a `time` property that associates it with a temporal `Interval` for temporal annotations such as subtitles or played track. `Interval` is a concept taken from the timeline ontology[15].

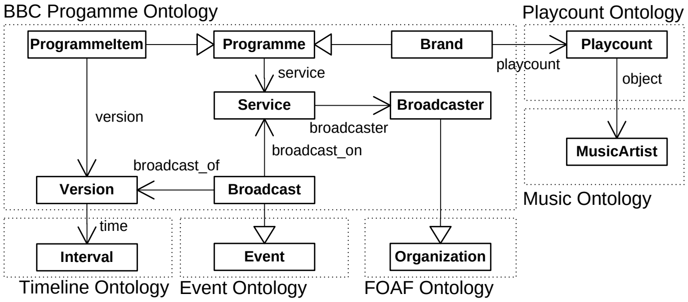

Fig. 7. Excerpt of the BBC ontology showing the interconnection with other ontologies

4.1 Ontologies

A network of ontologies such as the example depicted in Figure 7 can consist of many ontologies created by different actors and communities. Each individual ontology is a formal representation of the relevant concepts and relations of a domain in a machine readable format [33, 32]. Ontologies are specified to represent a joint point of view [33], which means that the formal conceptualization of the ontologies expresses a consensus and shared view between the different creators. The ontologies may be the result of some transformation or re-engineering tasks of legacy systems such as existing relational database schemata or existing taxonomies like the Dewey Decimal Classification[16] or Dublin Core. Other ontologies are crafted from scratch applying one of the existing ontology engineering methods and methodologies, leveraging modern ontology development tools, and choosing an appropriate ontology representation language (see Section 5). The ontologies may be very simple such as FOAF or the event ontology mentioned before or they may be very complex and designed by domain experts

---

[14] `http://motools.sourceforge.net/event`

[15] `http://motools.sourceforge.net/timeline`

[16] `http://dewey.info/`

like the medical ontology SNOMED. Ontology engineering is the field that studies the methods and methodologies for building ontologies [23]. It originates from domain modeling in software engineering and conceptual modeling in databases. A good overview of the support for ontology engineering is provided by reference books such as [43, 23]. The structure of ontologies, their size, and their methodology of engineering varies greatly with their scope and purpose. Complex ontologies are further distinguished according to their purpose and level of generality.

Domain ontologies like SNOMED are a representation of knowledge that is specific for a particular domain [14, 32] and can be very complex. Domain ontologies can be used as external sources of background knowledge [14], e.g., in combination with core ontologies.

Core ontologies provide a precise definition of structured knowledge within a specific field that spans across several application domains [32]. An example is the core ontology for software components and web services [32]. Core ontologies can be based on foundational ontologies by refining and adding concepts and relations in their specific field.

Foundational ontologies have a large scope and are highly reusable in different modeling scenarios [8]. Thus, foundational ontologies serve reference purposes [32] and aim at modeling the very basic and general concepts and relations [8, 32] that make up our world, e.g., objects and events. An example is the Descriptive Ontology for Linguistic and Cognitive Engineering (DOLCE) [8]. Foundational ontologies with their rich axiomatization are important during the design time of ontologies. They help the engineer in defining a sound and formal conceptualization of the considered part of the world that is to be modeled and to validate that it is free of inconsistencies. For later use of the foundational ontology in a concrete application, i.e., during runtime of the application, the rich axiomatization can be removed and replaced by a lightweight version of the foundational ontology.

Such kind of use stands in contrast to the one of domain ontologies. They are built specifically for reasoning during the runtime of an application. Thus, when designing and applying ontologies it is important to trade-off completeness and complexity with efficiency while distinguishing the modeling and validation support that is needed during design time and the reasoning support that is required during runtime. To support the representation of structured knowledge in scenarios such as the example in Figure 7, the ontologies need to be connected and integrated into a network that spans over the web. To this end, the ontologies used in the network need to be matched and aligned to each other.

4.2 Networked Ontologies

A network of ontologies such as the example outlined in Figure 7 needs to be flexible with respect to the functional requirements that are emposed on it. This is due to evolution of systems over time, being extended, combined, and integrated. In addition, networked ontologies must lead to a coherent understanding of the modeled domain. Such coherence may be supported by a sufficient level of formality and axiomatization as well as by following a pattern-based approach that allows for selecting and reusing all or parts of the original ontologies in the network in an overall coherent manner.

In order to engineer a network of ontologies capable to represent and exchange structured knowledge in a complex domain an approach based on core ontologies may be followed. Well designed core ontologies fulfill the requirements discussed in the preceding paragraph and allow for an easy integration and smooth interplay. The approach for networked ontologies depicted in Figure 7 results in a flat structure where all ontologies reside on the same level. While such structures may be mastered up to some degree of complexity, a principled approach based on networked, formally defined core ontologies provides an abstraction layer that serves as a guidance for all the domain and structure ontologies below.

We exemplify the approach of using networked core ontologies with a stack of ontologies from foundational, over core, to domain ontologies. The stack includes DOLCE as a foundational ontology, M3O [38] as a core ontology for multimedia metadata, and its extension for the music domain is sketched in Figure 8.

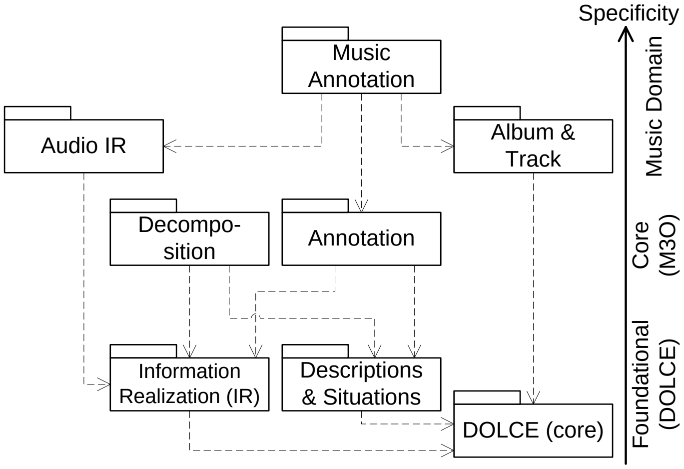

Fig. 8. Ontology stack combining DOLCE, M3O, domain specific extensions of M3O for audio and annotating music, and a domain ontology of albums and tracks

Core ontologies—even when well-designed—are usually large and cover more knowledge than might be required in a specific application domain [19]. Concrete systems will commonly use only portions of it. To provide a modular design, core ontologies should follow a pattern-based approach. Like design patterns in software engineering, design patterns in ontology engineering provide modeling solutions to recurrent ontology design problems [19]. DOLCE provides such a pattern-based approach. The precise alignment of the concepts defined in a core ontology with a foundational ontology provides a solid basis for future extensions. New patterns can be added and existing patterns can be extended by specializing the concepts and properties defined in the patterns. Figure 8 shows different patterns of the M3O and DOLCE as well as their relationships. In the ideal case, domain ontologies reuse the design patterns defined in core ontologies [19] as depicted in Figure 8. As one cannot assume that all domain ontologies are aligned with a foundational ontology or core ontology, both options need to be considered explicitly. Reuse of domain knowledge by core ontologies can be achieved, e.g., by applying the design pattern Descriptions and Situations (DnS) defined in DOLCE. It provides an ontological formalization of context [32] by defining different views through roles. These roles can refer to a domain ontology and allow for a clear separation of the structured knowledge defined in the core ontology from the domain-specific knowledge.

In future, we expect further ontology design patterns and core ontologies to emerge. For example, the music scenario in Section 1 could benefit from a core ontology on media management.

5 Knowledge Representation

Ontologies and the semantics of data may be formally specified using means of knowledge representation. In the Semantic Web, formal representation languages from several knowledge representation paradigms are used, especially languages that are based on description logics and logic programming. The core languages for ontology modeling in the Semantic Web are RDF/S and OWL (see Section 2). RDF allows to express simple facts (statements with subject, predicate, and object), e.g., ‘Anni-Frid Lyngstad’ ‘is a member of’ ‘ABBA’. Entities are connected by different named relationships resulting in a directed graph representing factual knowledge. RDFS can be used to define entity types (classes), relationship types between them, and a subsumption hierarchy between types.

In the following, we illustrate drawing conclusions from example facts by the use of the RDFS construct `rdfs:subClassOf` that can be used to model class hierarchies and the OWL construct `owl:sameAs` to state that two resources are identical. For the first example, we consider the concept `foaf:Person` defined by FOAF and the Music Ontology which specifies classes `mo:Musician` and `mo:Group`. Additionally, the Music Ontology contains an axiom specifying that `mo:Musician` is a subclass of `foaf:Person` via the `rdfs:subClassOf` property. Such a construct allows a reasoner to deduce that instances of `mo:Musician` are also of type `foaf:Person`. Given the definition of such a hierarchy and a fact stating that Anni-Frid Lyngstad is a `mo:Musician`, a reasoner can draw the conclusion that Anni-Frid Lyngstad is also of type `foaf:Person`. Consequently, applications that query for all resources of type `foaf:Person` will also get Anni-Frid Lyngstad as a query result even if the instance itself is not directly of that type. Figure 9 depicts these facts and hierarchy in RDF/S as directed, named graph.

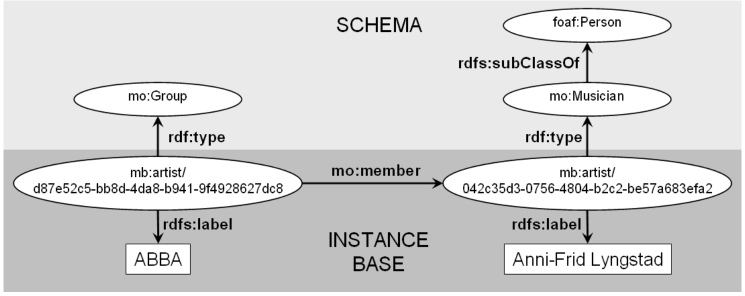

Fig. 9. Visualization of example RDF data about ABBA and Anni-Frid Lyngstad.

In the second example, the OWL construct `owl:sameAs` is used to specify that two resources are identical, e.g., `http://www.bbc.co.uk/music/artists/d87e52c5-bb8d-4da8-b941-9f4928627dc8#artist` and `http://dbpedia.org/resource/ABBA`. It allows a reasoner to consolidate information about ABBA from multiple sources (see Section 3.3). Using `owl:sameAs` instead of reusing the same URI across sources allows for sources being connected after they coined the URIs for the respective thing. In addition, by coining separate URIs, the different sources can provide information on the resource via HTTP lookups on their own URI.

Besides the construct `owl:sameAs`, OWL comprises a large number of constructs for defining classes and properties as well as concrete facts. For instance, OWL enables to define transitivity (transitive property), relationship inversion (like is-member-of is an inverse of has-member), or union and intersection of class expressions. Using OWL, it is not only possible to reason with ontology schema or a knowledge base of instances, but also to check consistency of an ontology or satisfiability of a specific concept in an ontology. For a detailed discussion we refer the interested reader to (Baader et al., in this issue).

Further expressiveness for knowledge representation on the Semantic Web is offered by RIF. However, at the time being there is less agreement on its use and fewer tool standards for its use.

6 Identity and Alignment

* (semi-)automatic, adaptive, or machine learning approaches for ontology alignment [15, 13, 5]

* core ontologies presented in Section 4.1 are able to incorporate linked open data

(todo) neeed to read these ones: * object identification [37]

* Yago ontology [46], generated from Wikipedia and Wordnet

* sameas.org service - duplicate detection and duplicate discovery service; helps you to find co-references between different data sets.

* work by Andrew McCallum:

* manual alignment at DNB

* Ontology Alignment Evaluation Initiative (OAEI) - establish a consensus for evaluation of methods for ontology matching

* OKKAM project: entity detection such as people, locations, organizations or events; with the goal to publish and link information on the web and to reuse identifiers for entities; Entity Name System (ENS), a scalable and sustainable infrastructure to make systematic reuse of entity identifiers

* SKOS - skos:exactMatch, skos:broader, ...

* problems such as in the German national library; publication of norm data about persons and corporations; deduplicate persons

* projects such as HP's efforts on determining global identifiers - Maciej, do you know these ones?

7 Trust and Provenance

The relevance ranking of Web pages by search engines, their URL location and their layout, their forward and backward links, and the provisioning of certificates, all this allows the Web page visitors to assign some—albeit limited and varying—amount of trust into the contents of a Web page. In the semantic Web, many of these characteristics, e.g. layout, may be missing. Then it becomes difficult to credit trustworthiness even to a single source of data. In the distributed setting of the Semantic Web, the use of data queries and logical inference mechanisms further implies the necessity to aggregate trust accredited to individual data items from distinct sources. Trust may be derived on the reputation of agents claiming a fact ("who said it?"), based on the temporal validity ("when was a fact asserted"), based on a fuzzy truth value ("to which extent is this true"?) or based on the dispute involving a fact ("who challenges the truth of a fact?"). "Trust is not a new research topic in computer science, spanning areas as diverse as security and access control in computer networks, reliability in distributed systems, game theory and agent systems, and policies for decision making under uncertainty. The concept of trust in these different communities varies in how it is represented, computed, and used" (cf. [1]). However, trust in the Semantic Web adds unique challenges that have to do with the distributed provenance of data *and* ontologies from different sources, different actors involved in providing the data and ontologies, and the objective of reasoning on such integrated knowledge. Key factors are: (i) provenance of data, (ii) reputation gained by previous interaction or social influence, (iii) credentials assigned through policies enforced by system(s), (iv) access control mechanisms, and sometimes the (v) certainty or importance of information also serves as an important measure to accredit trust to a set of facts. Under any circumstances, the possible costs and their probability of occurring, i.e., the risks arising from accrediting trust to another agent must be considered. However, to our knowledge such a risk management strategy has not been applied to semantic Web data, yet.

The key factors enumerated above are managed in a set of different systems. Provenance of data in the Semantic Web has been investigated along the lines of data lineage for RDF based data [11, 17] as well as for OWL and rules [11]. Reputation in social and semantic networks has been investigated [22, 21] exploiting the topology of links in these networks. Semantic Web policies have been defined as representations in the Semantic Web [42] and representations for the Semantic Web [7]. Some of them are serving access control to distributed Semantic Web data [20]. Finally, computations of degrees [44] and importance of resources [18] and subgraphs [30] help to contribute achieving appropriate trust estimations.

8 User Interfaces for the Semantic Web

The increasing popularity and availability of various kinds of semantic data and linked data raises new challenges for user interface design. User interfaces for Semantic Web applications differ significantly from traditional applications. This is due to the requirements that arise from the characteristics of semantic data. In contrast to traditional data provided by relational databases or originating from XML schema, one cannot assume to have any predefined knowledge about the schema and the type of data provided. One does also not know how many resources exist in terms of both amount of triples as well as number of repositories providing those triples. Finally, repositories in the Semantic Web can also be arbitrarily extended or modified.

The dynamics of semantic data and semantic repositories needs to be reflected by the applications that are retrieving, processing, and visualizing the data to end users. Thus, one of the central challenges the Semantic Web community is currently tackling is the development of innovative user interfaces and applications that can deal with this flexibility of the data schema and the huge amount and heterogeneity of the data provided. At the same time, such semantic applications must be easy to understand and to use. They need to be extensible and offer specialized capabilities for data from domains which were not anticipated during design-time of the interface [36]. This puts much more requirements and efforts in designing the application's user interface.

A key element to reach the goal of developing such flexible, *living* user interfaces for the Semantic Web is the use of faceted search and navigation. With facets, users can iteratively explore and visualize semantic data following a blended browsing and querying (BBQ) approach [31]. This means that the users search and navigate through the data in iterations of refinement. Within each iteration, the users apply a facet acting as filter on the data set. A query to the repository returns only those data that match the selected facet(s). By this, browsing the data is blended with querying for data from the repositories. Unfortunately, we cannot give a strict definition of facets [39]. Basically, one can say that a facet is a single criteria that can be used to subdivide a given data set.

Thus, in the context of the Semantic Web, a facet is basically a criteria to split a RDF graph. Facets shall be orthogonal to each other and behave as coordinates in a multidimensional space. For example, music data can be characterized by artist, album, title, label, and year. The Fresnel Display Vocabulary [35] can be used to specify how a facet should be rendered. They are used to select and order parts of an RDF graph, add content formatting, and hooks to CSS styling instructions. Facets often encode complex data that require dedicated visualization widgets such as the SIMILE Timeline widget [29] for temporal information and Google Maps[17] to render spatial information. Another major task in building flexible, living user interfaces over semantic data is to decide on how to order query results. As in Web search, ranking plays an important role to prioritize and order items for display. Ranking offers the application to decide on orderings in lieu of a fixed schema which is typically used to decide on ordering data items.

An example for an interactive application of faceted search and navigation following the BBQ approach is SemaPlorer [40]. The SemaPlorer application allows to interactively explore and visualize a very large and semantically heterogeneous semantic data set in real-time. The data set used consists of DBpedia, GeoNames[18], WordNet[19], and personal FOAF files. In addition, a live wrapper to the application programming interface of Flickr[20] is employed. SemaPlorer defines four facets of general interest, namely location, time, people, and tags. A screenshot of the SemaPlorer application is depicted in Figure 10. Users can state text queries on the left hand side, and select a query result from a proper facet. The content of a current facet such as location is shown on the right hand side of Figure 10. Finally sights and images of a certain city are depicted in the middle. From a software engineering point of view it is generically implemented and flexible. However, the different facets and the data that can be presented by the facets are hard-wired with the application. Thus, similar to the majority of applications in the Semantic Web, the data used as input for the SemaPlorer application is directly connected to the features how it can be searched, explored, and visualized by the user interface.

Examples of applications with more flexible, living user interfaces are Paggr[21] and Sigma. Paggr is the winner of the Semantic Web challenge 2008. It makes use of structured, self-describing data on the Web to create ad-hoc semantic mashups and organizes them in personalized dashboards. Also Sigma (see Section 7) combines different data sources and is flexible with respect to different data schemata. Finally, Lena[22] is an RDF browser that allows to present a particular view onto the RDF data described by the Fresnel Display Vocabulary. An adaptive version of Lena supporting the ranking of the RDF data has been developed using the TripleRank algorithm [18]. Applications providing such living

---

[17] `http://maps.google.com`

[18] `http://geonames.org`

[19] `http://wordnet.princeton.edu`

[20] `http://flickr.com`

[21] `http://www.paggr.com/`

[22] `http://code.google.com/p/lena/`

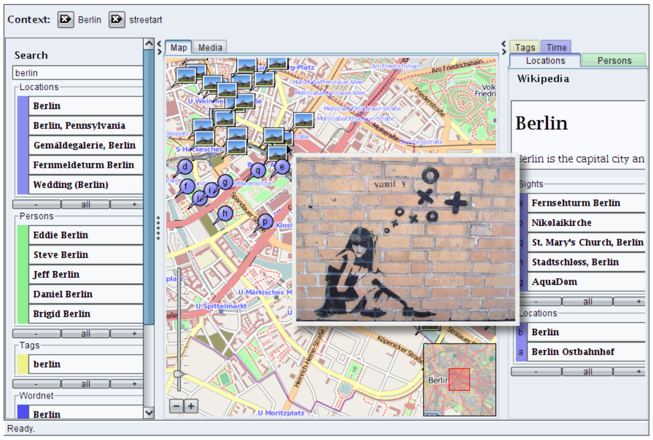

Fig. 10. Faceted search and navigation of semantic data in SemaPlorer [40]

user interfaces do not require any predefined knowledge about the data schema but adapt themselves to the actual data that is provided, or can easily be reconfigured. Thus, if a data source is extended or modified, applications providing living user interfaces are able to instantly reflect this on the user interface.

9 Conclusion

As we have presented in this article, the Semantic Web is an interesting technology that is highly influenced by the long-term research and achievements of Artificial Intelligence. It extends Artificial Intelligence by a general architecture for modeling, representing, and reasoning on distributed knowledge, i.e., knowledge that is spanned over the World Wide Web. Based on the current Web, the Semantic Web adds a multitude of standards and languages to provide machine-accessible semantics to the documents and data on the Web. The full potential of the Semantic Web has yet not been exploited as important layers of the Semantic Web architecture such as provenance, trust, and crypto are currently researched and have yet not been standardized. However, the importance of the Semantic Web is constantly growing. In particular, the dramatic growth of the Linked Open Data cloud[23] has already begun impacting the industry. For example, Yahoo has developed with its SearchMonkey platform[24] a search engine that exploits among others semantic data for web retrieval. With SearchMonkey, one can add descriptions to a website to influence its visual appearance on the search result list. Studies on selected services have shown that websites providing appropriate metadata for SearchMonkey have a 15% higher

---

[23] The growth of the Linked Open Data cloud is constantly documented on `http://linkeddata.org/`.

[24] `http://developer.yahoo.com/searchmonkey/`

click-trough-rate once the website made it to the top 10 results.[25] Other companies such as BestBuy.com even report higher traffic of about 30% since they have started publishing semantic data with their website in September 2009.[26] BestBuy uses the GoodRelations vocabulary[27] gathered by SearchMonkey to describe its e-commerce offerings. Also Google has started exploiting semantic data from e-commerce websites describing their products with the GoodRelations vocabulary.[28] Another success story is the release of governmental data to the general public. For example, Data.gov[29] is an effort undertaken by the US government to release federal datasets and US Census[30] provides various statistics about the United States. In UK, data.gov.uk[31] is the key element of the Government's Transparency programme in making available data of the UK public sector. The Open Public Sector Information project[32] is an initiative to release public data of the UK government.

Finally, we observe an explosion of biomedical semantic data available on the Web. In the collaborative effort of Bio2RDF[33] various bioinformatics databases are connected and made available. The Transinsight GmbH offers a search tool called GoPubMed[34] to find biomedical research articles and to collaboratively modify an ontology describing the data. As we have shown, Semantic data on the web has begun to have a real impact on providers of commercial services and products as well as governmental organizations. This suggests a successful future of the Semantic Web.

Acknowledgements. This paper has greatly benefitted from discussions we had with Andreas Harth, KIT, when developing our common view to a Semantic Web architecture [26].

References

1. D. Artz and Y. Gil. A survey of trust in computer science and the semantic web. *J. Web Sem.*, 5(2):58–71, 2007.
2. S. Auer, C. Bizer, G. Kobilarov, J. Lehmann, R. Cyganiak, and Z. Ives. DBpedia: A nucleus for a web of open data. In *Semantic Web Conference and Asian Semantic Web Conference*, pages 722–735, November 2008.

---

[25] `http://developer.yahoo.net/blog/archives/2008/07/`

[26] `http://priyankmohan.blogspot.com/2009/12/online-retail-how-best-buy-is-using.html`

[27] `http://www.heppnetz.de/projects/goodrelations/`

[28] `http://www.ebusiness-unibw.org/wiki/GoodRelationsInGoogle#GoodRelations_in_Google_Rich_Snippets`

[29] `http://www.data.gov/`

[30] `http://www.rdfabout.com/demo/census/`

[31] `http://data.gov.uk`

[32] `http://www.openpsi.org/`

[33] `http://bio2rdf.org/`

[34] `http://www.gopubmed.org/`

3. T. Berners-Lee. Universal resource identifiers in WWW: a unifying syntax for the expression of names and addresses of objects on the network as used in the World-Wide web. RFC 1630, Internet Engineering Task Force, June 1994.
4. C. Bizer. The emerging web of linked data. *IEEE Intelligent Systems*, 24(5):87–92, 2009.
5. E. Blomqvist. Ontocase-automatic ontology enrichment based on ontology design patterns. In *International Semantic Web Conference*, pages 65–80, 2009.
6. H. Boley, G. Hallmark, M. Kifer, A. Paschke, A. Polleres, and D. Reynolds. RIF core dialect. W3C candidate recommendation, W3C, October 2009. http://www.w3.org/TR/rif-core/.
7. P. A. Bonatti and D. Olmedilla. Rule-based policy representation and reasoning for the semantic web. In *Reasoning Web Summer School*, volume 4636 of *Lecture Notes in Computer Science*, pages 240–268. Springer, 2007.
8. S. Borgo and C. Masolo. *Handbook on Ontologies*, chapter Foundational choices in DOLCE. Springer, 2nd edition, 2009.
9. D. Brickley and R. Guha. RDF vocabulary description language 1.0: RDF schema. W3C recommendation, W3C, February 2004. http://www.w3.org/TR/rdf-schema/.
10. J. Broekstra, A. Kampman, and F. V. Harmelen. Sesame: A generic architecture for storing and querying RDF and RDF schema. In *International Semantic Web Conference*, pages 54–68. Springer, 2002.
11. R. Q. Dividino, S. Schenk, S. Sizov, and S. Staab. Provenance, trust, explanations - and all that other meta knowledge. *KI*, 23(2):24–30, 2009.
12. M. Duerst and M. Suignard. Internationalized resource identifiers (IRIs). RFC 3987, Internet Engineering Task Force, Jan. 2005.
13. M. Ehrig. *Ontology Alignment: Bridging the Semantic Gap*, volume 4 of *Semantic Web and Beyond*. Springer, 2007.
14. J. Euzenat and P. Shvaiko. *Ontology matching*, chapter Classifications of ontology matching techniques. Springer, 2007.
15. J. Euzenat and P. Shvaiko. *Ontology matching*. Springer, 2007.
16. R. T. Fielding. *Architectural Styles and the Design of Network-based Software Architectures*. PhD thesis, University of California, Irvine, USA, 2000.
17. G. Flouris, I. Fundulaki, P. Pediaditis, Y. Theoharis, and V. Christophides. Coloring RDF triples to capture provenance. In *International Semantic Web Conference*, volume 5823 of *LNCS*, pages 196–212. Springer, 2009.
18. T. Franz, A. Schultz, S. Sizov, and S. Staab. Triplerank: Ranking semantic web data by tensor decomposition. In *International Semantic Web Conference*, volume 5823 of *LNCS*, pages 213–228. Springer, 2009.
19. A. Gangemi and V. Presutti. *Handbook on Ontologies*, chapter Ontology Design Patterns. Springer, 2nd edition, 2009.
20. R. Gavriloaie, W. Nejdl, D. Olmedilla, K. E. Seamons, and M. Winslett. No registration needed: How to use declarative policies and negotiation to access sensitive resources on the semantic web. In *European Semantic Web Symposium*, volume 3053 of *LNCS*, pages 342–356. Springer, 2004.
21. J. Golbeck and J. A. Hendler. Inferring binary trust relationships in web-based social networks. *ACM Trans. Internet Techn.*, 6(4):497–529, 2006.
22. J. Golbeck and A. Mannes. Using trust and provenance for content filtering on the semantic web. In *Models of Trust for the Web*, CEUR Workshop Proceedings. CEUR-WS.org, 2006.
23. A. Gomez-Perez, M. Fernandez-Lopez, and O. Corcho. *Ontological engineering*. Springer, 2004.

24. W. O. W. Group. OWL 2 web ontology language document overview. W3C recommendation, W3C, October 2009. http://www.w3.org/TR/owl2-overview/.
25. A. Harth, K. Hose, M. Karnstedt, A. Polleres, K.-U. Sattler, and J. Umbrich. Data summaries for on-demand queries over linked data. In *World Wide Web*. ACM, 2010.
26. A. Harth, M. Janik, and S. Staab. Semantic Web architecture. In *Handbook of Semantic Web Technologies*. Springer, 2010/2011.
27. O. Hartig, C. Bizer, and J. C. Freytag. Executing SPARQL queries over the web of linked data. In *International Semantic Web Conference*, pages 293–309, 2009.
28. J. Heinsohn, D. Kudenko, B. Nebel, and H.-J. Profitlich. An empirical analysis of terminological representation systems. *Artif. Intell.*, 68(2):367–397, 1994.
29. D. F. Huynh, D. R. Karger, and R. C. Miller. Exhibit: lightweight structured data publishing. In *World Wide Web*, pages 737–746. ACM, 2007.
30. G. Kasneci, S. Elbassuoni, and G. Weikum. Ming: mining informative entity relationship subgraphs. In *Information and Knowledge Management*, pages 1653–1656. ACM, 2009.
31. K. D. Munroe, B. Ludscher, and Y. Papakonstantinou. Blending Browsing and Querying of XML in a Lazy Mediator System. In *Extending Database Technology*, 2000.
32. D. Oberle. *Semantic Management of Middleware*. Springer, 2006.
33. D. Oberle, N. Guarino, and S. Staab. What is an ontology? In S. Staab and R. Studer, editors, *Handbook on Ontologies*. Springer, 2nd edition, 2009.
34. Y. Papakonstantinou, H. Garcia-Molina, and J. Widom. Object exchange across heterogeneous information sources. In *Data Engineering*, pages 251–260, Washington, DC, USA, 1995. IEEE Computer Society.
35. E. Pietriga, C. Bizer, D. R. Karger, and R. Lee. Fresnel: A browser-independent presentation vocabulary for RDF. In *International Semantic Web Conference*, pages 158–171, 2006.
36. D. A. Quan and R. Karger. How to make a semantic web browser. In *World Wide Web*, pages 255–265, New York, NY, USA, 2004. ACM.
37. S. Rendle and L. Schmidt-Thieme. Object identification with constraints. In *Proceedings of the 6th IEEE International Conference on Data Mining (ICDM 2006), 18-22 December 2006, Hong Kong, China*, pages 1026–1031. IEEE Computer Society, 2006.
38. C. Saathoff and A. Scherp. Unlocking the semantics of multimedia presentations in the web with the multimedia metadata ontology. In *World Wide Web*, 2010.
39. G. M. Sacco and Y. Tzitzikas, editors. *Dynamic Taxonomies and Faceted Search : Theory, Practice, and Experience*. Springer, Berlin, 2009.
40. S. Schenk, C. Saathoff, S. Staab, and A. Scherp. SemaPlorer—interactive semantic exploration of data and media based on a federated cloud infrastructure. *Journal of Web Semantics*, 2009.
41. S. Schenk and S. Staab. Networked graphs: a declarative mechanism for SPARQL rules, SPARQL views and RDF data integration on the web. In *World Wide Web*, pages 585–594. ACM, Apr. 21-25, 2008.
42. F. Schwagereit, A. Scherp, and S. Staab. Representing distributed groups with dgFOAF. In *Extended Semantic Web Conference*, LNCS. Springer, 2010.
43. S. Staab and R. Studer, editors. *Handbook on Ontologies*. Springer, 2009.
44. G. Stoilos, G. B. Stamou, J. Z. Pan, V. Tzouvaras, and I. Horrocks. Reasoning with very expressive fuzzy description logics. *J. Artif. Intell. Res.*, 30:273–320, 2007.

45. H. Stuckenschmidt, R. Vdovjak, J. Broekstra, and G.-J. Houben. Towards distributed processing of RDF path queries. *Int. J. Web Eng. Technol.*, 2(2/3):207–230, 2005.
46. F. M. Suchanek, G. Kasneci, and G. Weikum. Yago: a core of semantic knowledge. In *WWW '07: Proceedings of the 16th international conference on World Wide Web*, pages 697–706, New York, NY, USA, 2007. ACM.
47. K. Wilkinson, C. Sayers, H. A. Kuno, and D. Reynolds. Efficient RDF storage and retrieval in Jena2. In I. F. Cruz, V. Kashyap, S. Decker, and R. Eckstein, editors, *SWDB*, pages 131–150, 2003.