Consuming multiple linked data sources: Challenges and Experiences

Ian C. Millard, Hugh Glaser, Manuel Salvadores and Nigel Shadbolt

School of Electronics and Computer Science
University of Southampton, UK
{icm, hg, ms8, nrs}@ecs.soton.ac.uk

Abstract. Linked Data has provided the means for a large number of considerable knowledge resources to be published and interlinked utilising Semantic Web technologies. However it remains difficult to make use of this 'Web of Data' fully, due to its inherently distributed and often inconsistent nature. In this paper we introduce core challenges faced when consuming multiple sources of Linked Data, focussing in particular on the problem of querying. We compare both URI resolution and federated query approaches, and outline the experiences gained in the development of an application which utilises a hybrid approach to consume Linked Data from the unbounded web.

Keywords: Linked Data, SPARQL, URI Resolution, Federated Query

1 Introduction

The vision of the Semantic Web is centred around the transition from a network of loosely interlinked text documents – the existing World Wide Web, suited primarily for human consumption – to a rigorously described and tightly interlinked 'Web of Data', intended for machine interpretation and automated processing.

In the past 5 or so years, the Linked Data community has worked hard to realise this vision. Combined with the push for 'Raw Data Now'[1], significant and increasing numbers of datasets are becoming available as Linked Data resources, as witnessed by the evolution of the Linked Data 'cloud diagram'[2].

However the main emphasis of these efforts has largely been focussed on publishing existing datasets, whereas the task of dataset integration and enrichment through cross-linkage has taken a lesser role until more recent years. While the cloud diagram may give the impression of nicely integrated data, analysis has shown that it is more sparsely connected than one might first think[3] [11].

Furthermore there are significant challenges in consuming multiple sources of Linked Data, due primarily to its distributed nature, and unfortunately there are still only a few applications or services which make use of the Web of Data

[1] http://www.ted.com/talks/tim_berners_lee_on_the_next_web.html
[2] http://lod-cloud.net/
[3] http://blog.larkc.eu/?p=1941

in the envisioned manner of accessing a generic homogeneous resource. While there are many examples of Linked Data being put to good use, they tend to be focussed on accessing a specific dataset, or pre-defined set of resources, utilising the benefits of easy access rather than the full power of data integration and interoperability. Querying the distributed resources which form the Web of Data is non-trivial, and still remains a largely unsolved problem.

This paper firstly explains challenges concerned with utilising the Web of Data in a distributed fashion, before outlining the experiences gained and methods employed in overcoming some of these issues during the development of the RKB Explorer platform and application [4,5].

2 Challenges in consuming Linked Data

Accessing the wealth of information provided in the Web of Data presents a wide range of challenging problems, not least of which include resource discovery, consolidation, and integration across a distributed environment in which little may be known regarding the makeup and content of the various sources which may be available. There are also many largely unresolved issues regarding versioning, changesets and the potentially dynamic nature of dataset content [13].

2.1 Co-reference

Co-reference, the problem of duplicate identifiers, is a critical issue within Linked Data. While it may be thought that the URI scheme used to identify resources will give a single ID for any given concept, in reality it gives *many* identifiers, potentially one or even more per source dataset.

While it is theoretically possible for all data providers to use a single URI globally to represent a concept, eg `http://example.com/id/Einstein`, in practice this is not a viable reality for several reasons.

Firstly, in the infancy of Linked Data, no such URIs existed; hence data publishers had to invent their own. The introduction of `dbpedia.org` went some way to resolving this bootstrap issue, giving identifiers for a broad range of concepts and entities from Wikipedia, and this has proved to be a common 'hub' in the Web of Data. However, it is difficult and time consuming for publishers to identify a 'better' or more common URI for a concept they are describing.

Indeed, many Linked Data sources have been created via exports or on-the-fly conversions of existing datasets, and utilise existing internal names or IDs; cross-linking to equivalent resources in other datasets often becomes an afterthought or separate follow-on activity. In fact, using external identifiers can significantly complicate the publishing of Linked Data, as it may add a layer of complexity to the publishing process. In addition, internal business processes of the unit that owns the data will be intimately concerned with identifiers, and disturbing these by introducing the need to consider external identifiers can be very expensive.

Furthermore, commercial or governmental interests are unlikely to adopt external or 'foreign' URIs from other datasets in their data, as they may have concerns of ‘ownership’ or the potential for a foreign resource not under their control to be changed or disappear. In many situations there may be doubt over the exact meaning or context in which an identifier and its description applies; for example does your notion of ‘London’ correspond exactly to my definition?

In such situations it is often the case that the easiest and most appropriate course of action for a data publisher is to mint their own URIs representing their understanding of each concept, entity or location, and to provide a simple label or description of that resource under their own domain space which they can control and maintain. Publishers or others can then (it is hoped) tie their local identifiers up with more generic or commonly used identifiers, hence forming the important cross-linkages within the Web of Data; however this task has not always received the attention and maintenance effort it deserves.

All of these issues compound the prevalence of multiple identifiers for resources within the Web of Data, leading to the issues of data discovery and having implications on the manner in which resources can be queried and consolidated as detailed in the following sections.

As early adopters of Linked Data, and in taking the less convenient but principled approach of storing and publishing different content sources separately, the team at Southampton have long been concerned with addressing the issues of co-reference identification and management. In more recent years, with the increasing availability, overlap and cross-linking of information in the Web of Data, these problems are now coming to the fore.

It is our belief that co-reference data should be treated as a first class entity, held and managed separately from the data itself. As a result, we have created a ‘Co-reference Resolution Service’ (CRS), which is described fully in [3]. In essence one or more CRS instances can be used to maintain sets of URIs that are deemed to be equivalent within a specific context. When queried with a URI, the CRS will return details of all other equivalent URIs. It is this technology which underpins the popular `sameAs.org` service, the leading source of co-reference data on the Semantic Web.

2.2 Ontology Mapping

There are many different sources of information within the Web of Data, expressed in a wide variety of different vocabularies. Due to similar reasons as to why common agreement of identifiers is not as prevalent as one might have thought, there are often a multitude of ontologies used to represent knowledge within Linked Data resources.

While the Linked Data infrastructure enables data items to be combined from across multiple sources, it is more often than not left to the end applications or consumers of the data to interpret the meaning of the different knowledge representations employed. This situation may improve over time as particular vocabularies become more established and gain popularity, however it is safe to assume that there will remain a number of ways in which resources are described, especially common concepts such as people, locations, organisations and topics or taxonomic classifications. These will correspond to the diversity of the organisations own and themselves consume the data.

The field of ontology mapping is a well established research topic, and so we shall not dwell on it here other than to say it remains a challenging issue that is not yet fully resolved, often requiring manual intervention and alignment between representations [8] or collaborative tools [1]. However the ability to interpret data from multiple vocabularies is a crucial element in the interoperation and transfer of knowledge across and between sources within the Web of Data. Mapping or translation services are likely to play a key part in the future development of applications and services which consume Linked Data from multiple sources.

2.3 Aggregation from distributed sources

Let us consider a trivial example: find all the people who a given individual claims to know. We may thus be looking for all triples that match the pattern:

```
ex:JoeBloggs foaf:knows ?x
```

There are two main approaches for accessing the Web of Data: the ubiquitous Linked Data URI resolution required by the founding principles, or the often more convenient direct access to SPARQL endpoint(s) if they are known.

A first sensible step for a system answering this request would be to resolve the URI `ex:JoeBloggs` and determine if any `foaf:knows` relationships exist. Alternatively, a SPARQL based approach may issue the obvious query to all known/configured endpoints; however it should be noted that provisioning a publicly available endpoint can be computationally expensive, complex queries often return partial results or time out, and service outages are not uncommon.

In fact, in a pure Linked Data world, where a URI appears in the subject position of the triple pattern, it is only necessary to either resolve the URI, or query such a SPARQL endpoint, since the endpoint is only standing as a proxy to serve the RDF for the URI. Of course, a consumer of Linked Data may choose to look for `foaf:knows` triples elsewhere, which leads to the complexities described below (Section 2.4). In this section we are only concerned with the simple case.

It is quite possible that Joe Bloggs is described in more than one dataset, using different URIs, so a co-reference service such as `sameAs.org` may be consulted, returning one or more identifiers equivalent to `ex:JoeBloggs`. Each of these new URIs can now be considered using the same method. Assuming that we only consider the one vocabulary, we may now have results for `?x` from each subject URI and/or endpoint. Depending on the intended use, it may be sufficient to simply perform a union over these results and return them to the client. However it is possible that the same information has been duplicated in more than one dataset, for example two sources may imply that Joe Bloggs knows David Smith. In some scenarios, such as citation counting analysis, this replication of knowledge should be considered only once, even though it is represented multiple times using different identifiers and potentially in a different vocabulary. In addition to co-references in the subject position, there may well be co-references within the results.

Even in the simplest of cases, careful consideration must be given as to where URI expansion should be performed to potentially increase the number of subject resources queried, or equivalent vocabulary/predicate terms, along with the need to collapse responses from various datasets to nominated or preferred identifier to alleviate the issue of co-references in the final result.

2.4 Resource discovery

The example query used in the above section was straightforward, and could be answered by simply resolving the requested subject URI and optionally each known co-referent identifier. Let us now consider the following seemingly trivial query, to determine who lives near London –

```
SELECT ?who WHERE { ?who foaf:based_near dbpedia:London }
```

In this case, the known resource is in the object position of the query, and we are asking for matching subjects. This presents a problem when accessing the Web of Data through URI resolution, as the URIs to resolve are not immediately obvious. Again, the first logical step is to resolve the `dbpedia:London` URI, which, if it returns a full concise bounded description, may indeed yield matching triples identifying resources within the `dbpedia` dataset that are near London. However, in the Web of Data, publishers are encouraged to re-use resources and cross link concepts, resulting in a large number of documents scattered across the Linked Data cloud which link to `dbpedia:London`. Note that these are not bi-directional, there is no means of traversing the reverse direction from the object to discover all subjects.

To overcome these resource discovery issues, a number of 3rd party services can be employed. Semantic Web search engines, such as `Sindice.com`, index Linked Data resources that are found by link-traversing spiders or bots. Such services may be able to return a list of documents in which a given URI exists, however functionality varies between the services and each may require a different access mechanism. Furthermore the number of results returned may be large (approx 20K for `dbpedia:London`) and without any ordering, prioritisation, or even an indication of where in a document the requested URI exists. As a result, such services must be used with caution, with careful attention paid to the number of resources that a client is willing to access or resolve.

In a more specialised field, domain specific 'backlinking' services may be employed to index triples across datasets enabling a lookup to discover subject resources that have triples containing a specified URI in the object position [12]. Backlinking services may additionally hold information about the subject resources they index, such as recording the `rdf:type`(s) and/or `rdfs:label`. These facilities may greatly reduce the number of URIs that would otherwise be performed unnecessarily by simply consulting a search engine, as the scope can be limited to only those types of resource which are of interest.

Finally, when taking a SPARQL based approach to accessing Linked Data resources, a different set of resource discovery problems arise. Without having to perform URI resolution, it makes little difference as to whether unbound variables are in the subject or object positions. Rather the discovery problem is determining which SPARQL endpoint(s) to consult in answering a given query.

The Vocabulary of Interlinked Datasets (voiD) is an emerging ontology that describes the contents and composition of Linked Data resources. voiD documents, expressed in RDF, contain properties which outline the main themes or subjects of the dataset, the ontologies being used, statistics relating to the number of triples, classes, and instances, and cross-linkages to other datasets. Additional useful features may be included, such as the availability and location of a SPARQL endpoint, and regular expressions defining the URI patterns in use within the dataset.

A voiD service[4] may collect a number of voiD documents in a single repository, and serviced by a SPARQL endpoint or REST API facilitate queries to be made which easily identify those datasets which are likely to hold information regarding a certain resource or topic. As a result, such services can be used to help identify valid SPARQL endpoints for use with a given query, although there is no guarantee they will yield results.

2.5 Queries spanning multiple datasets

With the increasing wealth of dataset cross-linkage, by means of direct inclusion of 'foreign' URIs, `owl:sameAs` links, and co-reference services, the Web of Data is becoming more tightly interwoven. This leads to a situation in which SPARQL queries cannot readily be executed as their constituent triple patterns span across multiple datasets.

Consider the example in Figure 1, where some activity `a:Activity1` is stated as having two related documents in other datasets, namely `b:Doc1` and `c:Doc2`. One may wish to ask the question, 'who has authored documents related to this activity', as expressed in the following simple SPARQL query –

```
SELECT ?p WHERE { a:Activity1 eg:related ?d . ?d eg:author ?p }
```

If all of the triples from namespaces `a:`, `b:` and `c:` were contained within the same dataset and serviced by a single SPARQL endpoint, then this query would be trivial. However, if as in the case shown in Figure 1 where each namespace is a separate dataset, then the above query cannot be executed by standard means even if each dataset provides an endpoint, as no one store contains all the facts required to answer the query.

This query can however be executed by means of repeated URI resolution, where the query is evaluated in stages. The first pattern can be evaluated by resolving `a:Activity1`, yielding pointers to the two documents `b:Doc1` and `c:Doc2`. Each of these URIs can then be resolved from their respective datasets to find the authors. It may also be possible for a similar step-by-step approach to be employed in conjunction with the available SPARQL endpoints, where the

[4] http://void.rkbexplorer.com/

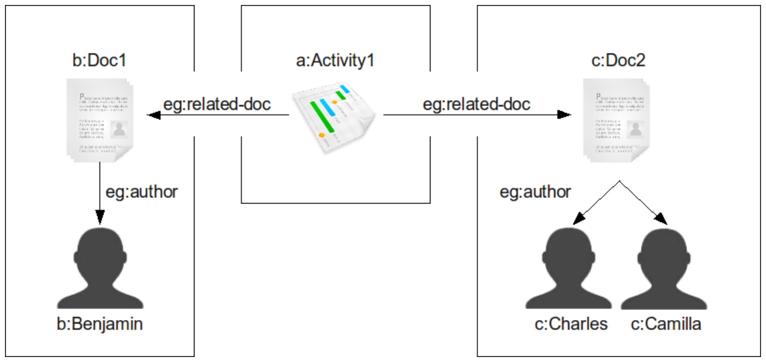

Fig. 1. An example of dataset inter-linkage, referencing 'foreign' URIs.

initial query is broken down and each step executed against the relevant domain endpoint.

The benefits and pitfalls of the URI resolution and SPARQL approach are discussed in the next section.

3 Related work

The problems of distributed query are not new, indeed for many years the database community have successfully produced high performance systems with data spread across a cluster of nodes. However, there are a number of important differences to note between such systems and the Web of Data. Firstly, in a database cluster, all storage elements are under the control of one organisation. They are almost always co-located, and connected by a fast network. All aspects of the data partitioning are carefully controlled, often with detailed statistics calculated and maintained to help inform the planning and execution of queries.

Conversely in the Web of Data, each node or dataset may be under different control. There may be network lag, and often there are no statistics available at all describing the size or makeup of the data.

We shall briefly consider two projects which have attempted to overcome the problems of distributed query within the Web of Data: The Semantic Web Client Library, which takes a URI resolution based approach, and DARQ, which utilises a federated SPARQL approach.

3.1 Semantic Web Client Library

The Semantic Web Client Library (SWCL) is an attempt to solve the problem of executing arbitrary queries over Linked Data resources [7]. The aims of the SWCL are to provide access to the entire Web of Data as if it were a single RDF graph, and to enable the execution of SPARQL queries over this graph.

Given a SPARQL query, the SWCL sets about retrieving the RDF by URI resolution that is considered to be necessary to provide the results, storing the resolved documents into a single RDF cache, over which the SPARQL query is finally performed. In this sense it is firmly in the Linked Data world – all RDF is fetched via HTTP using Linked Data resolution, and no remote SPARQL queries are executed. It is therefore unable to fetch RDF data that is not exposed as Linked Data, excluding that which is only available via a SPARQL endpoint.

A key factor within the SWCL is the determination of how much RDF to dereference before attempting to execute a query on the local cache. Initially any URIs present in the query are resolved, along with any pre-determined graphs that are specified to be fetched. A second phase of resolution is then performed, dereferencing those URIs that were discovered in the first round of returned graphs. The query is executed in stages, with further resolution steps performed on intermediary results as required.

The SWCL can execute complex queries over large portions of the Semantic Web, utilising many different data sources. However, there are high overheads of performing significant numbers of HTTP resolutions, and indeed much time can be wasted in transferring, parsing and importing data which is not required to answer the query. For example, in executing a query which asks for the email address for all members of a university, it is likely that the URI for each individual would be resolved. Each such resolution will potentially return a large and complex document detailing all aspects of each individual's activities, including their publications, interests, and teaching duties, where all that was required to answer the query was a single triple containing their address. There is no way to predetermine the quantity of data that will result when resolving a URI, or standard means to restrict or filter the data returned – an issue facing any Linked Data resolving client.

Futhermore, if the ontological relationship was `<person> works-at <uni>`, and resolving the university URI does not return a symmetric concise bounded description, then there may be resource discovery problem, requiring a fall-back to utilise a search engine.

Due to the expense of performing numerous URI resolutions, and depending on the queries executed, performance with SWCL can be generally poor.

3.2 DARQ

DARQ is in some sense at the opposite end of the spectrum from the SWCL. Where the SWCL accesses all the Linked Data by HTTP resolution, DARQ relies solely on querying remote SPARQL endpoints [10]. The aim is to provide perhaps the ideal facility for distributed query – an application presents the DARQ engine with a SPARQL query, and the system analyses it and plans and transparently executes the required individual queries over all the required SPARQL endpoints.

However, there are a number of shortcomings. DARQ requires a SPARQL endpoint for all resources it consumes, whereas a large proportion of the Web of Data is available only as resolvable URIs. Furthermore, a significant barrier is that in order to use a SPARQL endpoint, DARQ requires a detailed Service

Description to describe the capabilities of that endpoint, and details of the predicates and resources contained within it to inform the query planning engine. This both requires significant (and computationally intensive) statistical work, and also limits the endpoints available to those that have been registered and for which the Service Descriptions are defined.

While the approach appears promising, unfortunately development on the DARQ project appears to have stopped around 2006. This lack of ongoing development also means that it is not compatible with the later versions of various libraries on which it is dependent. Finally, it should be noted that DARQ does not deal with the full range of the SPARQL query language.

4 Experiences gained with RKBExplorer

As part of the ReSIST project[5] the team at Southampton were tasked with providing a range of knowledge management technologies to both support the activities of the project and enhance dissemination of the outputs. We chose to utilise a semantic approach, compliant with the emerging Linked Data best practises. It was particularly exciting to be putting the *web* into Semantic Web, at a time where the majority of Semantic Web research was focussed on storing or caching large quantities of data all in one repository.

One of the core outputs of this work was the production of the RKBExplorer application[6], which uses community of practice (CoP) style analyses to identify different types of resource that are related to a given person, publication, project or research topic. It provides a simple user interface giving a coherent view over a multitude of data sources, with little indication of the underlying semantic infrastructure [4,5].

During the project, we produced more than 20 different Linked Data sources, each hosted separately as a sub-domain of `rkbexplorer.com`. Focussing largely on academics, their institutions and scholarly works, there was significant overlap between a number of datasets, hence our early work on co-reference [3].

However given our distributed datasets, and co-reference knowledge, we required a solution to enable the RKBExplorer application to interact with our 'web' as if it were a single entity. The identification of other resources related to the given focus resource is achieved by means of a domain and type specific configuration, defining queries that represent the relationships which indicate relevance between two types of resource. A hybrid approach was developed to enable efficient execution of such queries, drawing on both federated SPARQL, URI resolution and co-reference expansion. The resulting system has a number of limitations, but offers excellent performance in the intended purpose, so long as queries are carefully constructed.

In RKBExplorer, thousands of queries can be performed for a single CoP, looking at 100Ms triples from 30+ local stores and other remote SPARQL endpoints and URI resolution. Where performance is sometimes slow for very well-

[5] ReSIST, EU Network of Excellence, 2006–2008. http://resist-noe.eu/

[6] http://www.rkbexplorer.com/explorer/

connected resources, sophisticated caching and refresh infrastructure means that users rarely wait very long while they browse.

Firstly, a generic query library was created which attempts to execute a SPARQL query in the best manner possible, as follows: Internal configuration details a number of available SPARQL endpoints, both local and externally. If a query contains URIs from only one dataset, the query is simply passed to the appropriate endpoint, if known. In the situation where either no endpoint is known, or there are URIs from multiple domains, then the URIs are resolved via HTTP and stored in a local cache repository over which the query is then made. Later versions utilise the voiD store to identify additional relevant endpoints.

This facility enables very simple queries to be submitted and transparently executed by the library by whichever means is most suitable. It does not provide any co-reference or cross-repository functionality; these capabilities lie within the carefully structured operation of the CoP engine.

Pair-wise configuration files specify how to relate one type of resource to another, each containing a number of searches or 'rules' which prescribe relationships that support a notion of relatedness between those types of resource. In the example below we pass four arguments: the input URI, identifying the currently focussed resource; two query 'snippets'; and a weighting to be applied to results of this rule.

```
doCOP(
  $inputURI,                                    // input
  "%targetURI% eg:related-doc ?intermediate",   // snippet 1
  "%intermediate% eg:author ?result",           // snippet 2
  5                                             // weighting
);
```

This rule is executed as follows. Firstly the input URI is expanded to a set of all equivalent identifiers. The first query snippet is fired replacing `%targetURI%` for each URI in the target set, resulting in a set of `?intermediate` bindings. For the second phase, each intermediary result undergoes similar co-reference expansion, before replacing `%intermediate%` in the second query snippet before execution. Note that we constrain ourselves to two query snippets, however each may contain more than one triple pattern.

The resulting set of `?result` values represents resources that are related to the initial `$inputURI`. To prevent false counting the result set is checked for co-references, with duplicates removed. Each URI result is then scored by attributing the weighting specified for the rule. Subsequent rules in the configuration are then considered, which may further increment the score for individual URIs, before finally sorting all results by their total score to give an ordered notion of 'relatedness'.

Each query is executed as appropriate by the library, usually via a SPARQL endpoint. While the process may involve a large number of queries in total, they are usually very simple 'atomic' statements or simple triple patterns which can be performed very quickly. A key benefit of segregating the rule into two phases is the handling of co-reference equivalences, and due to the multi-phase execution results can span multiple repositories. Indeed, the example discussed in Section 2.5 can be handled by this system.

While this hybrid system cannot execute arbitrary queries across multiple datasets, the CoP engine with carefully constructed queries can perform complex analyses in a very efficient manner over distributed resources. Performance is much improved over using a URI-only based approach such as SWCL [9], and detailed DARQ-like statistics are not required to configure endpoints (indeed with the voiD store, endpoints can be dynamically included). The system is successfully used by the RKBExplorer application, performing many hundreds of queries to deduce related resources for each subject viewed.

To tackle the issues of ontology mapping, specific datasets can be configured within the RKBExplorer platform to be accessed by URI resolution through an on-the-fly mapping service [6]. Work has also been undertaken on re-writing SPARQL queries [2] to achieve similar cross-vocabulary data inclusion.

5 Future work

The key drawback of the CoP engine system is the hard-coded limitation of a two-phase query execution. Work is underway to improve the flexibility, in a more DARQ-like fashion by breaking an arbitrary query down into constituent triple patterns and executing them separately. The current prototype is somewhat naïve in the order of triple execution, which can lead to poor performance with some queries where there are a large number of intermediary resources to be joined or filtered. The system would benefit from a more intelligent query planning algorithm, yet this requires details of either the data makeup in each repository, or perhaps typical patterns to be expected when using particular vocabularies or predicates.

voiD documents are increasing in number as dataset publishers recognise their importance, and tool support for automatically generating such descriptions improves. However one issue in using a voiD store to identify relevant endpoints is the prevalence of URI patterns from common hubs – the majority of Linked Data sources may contain references to datasets such as `DBPedia`. Hence when determining which endpoints may match a given `DBPedia` URI, the voiD store is likely to return a large number of endpoints, some of which may actually only contain a handful of URIs from that domain, making a successful match rather unlikely.

Nevertheless, voiD documents are increasingly containing more statistical information regarding the datasets they describe, which may both help inform query planning and enable prioritisation of endpoints in the voiD store.

Finally, further work is required with regards to investigating the trade-offs between performance and query completeness when dealing with the distributed resources on the Web of Data. At each stage of a query execution plan, decisions must be made as to whether co-reference expansion or collapse is performed, and/or how many resources are considered for querying or URI dereference.

6 Conclusions

In this paper we have introduced the major challenges in consuming linked data from multiple sources, namely co-reference, resource discovery, and the issues involved with spanning queries over multiple endpoints. We have outlined our experiences in building an application which utilises the unbounded Web of Data, employing a hybrid query engine to execute restricted format queries efficiently. The benefits and drawbacks of this system have been discussed, and similar systems have been compared. Finally we looked forward at our ongoing work in developing a more generic solution to facilitate distributed query over Linked Data resources, highlighting key issues and potential solutions.

7 Acknowledgements

This work has been supported with finance and time by many projects, organisations and people over the years, most recently the EnAKTing project funded by the UK's EPSRC under contract EP/G008493/1.

References

1. Correndo, G., Alani, H., Smart, P.R.: A community based approach for managing ontology alignments. In: 3rd International Workshop on Ontology Matching (2008)
2. Correndo, G., Salvadores, M., Millard, I.C., Glaser, H., Shadbolt, N.: SPARQL Query Rewriting for Implementing Data Integration over Linked Data. In: 1st International Workshop on Data Semantics (DataSem 2010) (2010)
3. Glaser, H., Jaffri, A., Millard, I.C.: Managing Co-reference on the Semantic Web. In: WWW2009 Workshop: Linked Data on the Web (LDOW2009)
4. Glaser, H., Millard, I.C.: RKBPlatform: Opening up Services in the Web of Data. In: International Semantic Web Conference (2009)
5. Glaser, H., Millard, I.C., Jaffri, A.: RKBExplorer.com: A Knowledge Driven Infrastructure for Linked Data Providers. In: European Semantic Web Conference (2008)
6. Glaser, H., Millard, I.C., Sung, W.K., Lee, S., Pyung, Kim, You, B.J.: Research on Linked Data and Co-reference Resolution. In: International Conference on Dublin Core and Metadata Applications (2009)
7. Hartig, O., Bizer, C., Freytag, J.C.: Executing SPARQL Queries over the Web of Linked Data. In: International Semantic Web Conference (2009)
8. Kalfoglou, Y., Schorlemmer, M.: Ontology mapping: the state of the art. The Knowledge Engineering Review 18(1), 1–31 (2003)
9. Querying: RKBExplorer -vs- SWCL, http://www.rkbexplorer.com/blog/?p=43
10. Quilitz, B., Leser, U.: Querying Distributed RDF Data Sources with SPARQL. In: European Semantic Web Conference (2008)
11. Rodriguez, M.A.: A Graph Analysis of the Linked Data Cloud. CoRR (2009)
12. Salvadores, M., Correndo, G., Szomszor, M., Yang, Y., Gibbins, N., Millard, I.C., Glaser, H., Shadbolt, N.: Domain-Specific Backlinking Services in the Web of Data. In: Web Intelligence (2010)
13. Umbrich, J., Hausenblas, M., Hogan, A., Polleres, A., Decker, S.: Towards Dataset Dynamics: Change Frequency of Linked Open Data Sources. In: WWW2010 Workshop: Linked Data on the Web (LDOW2010)