Link Services for Linked Data

Yang Yang

Faculty of Physical Sciences and Engineering

University of Southampton

A thesis submitted for the degree of

Doctor of Philosophy

2014

Abstract

This thesis investigated the concept of building link services as an extension of Linked Data to improve its navigability (thus improving the linking of the Web of Linked Data). The study first considered the Semantic Web URI and how an agent understands what a URI refers to when dereferencing it. As a result, a generic URI dereferencing algorithm was designed which can be used by any agent to consume Linked Data. The navigability of the Web of Linked Data was then defined - how an agent can follow the links to discover more data. To understand how the Web of Linked Data is connected, this study found 425 million across-datasets URIs (URIs link two different datasets and enable discoverability between datasets) on the Linked Data cloud and only 7.5% of resources are linked to non-local datasets. To improve the navigability of the Web of Linked Data, a list of link services was built. These link services are RESTful services, and takes a link as input and provides a RDF document as output with linking information of the requested URIs. They are: resolution service (retrieves the RDF description of the requested URI for agents), Link extraction service (extracts URIs from a RDF), Linkbase service (third party hosting link relations between datasets, especially for those data which were not originally linked), Reasoning service (applies rules of reasoning to generate a new RDF), Composition service (compose multiple RDF documents into one documents), and Link injection service (inject extra links relations into the client requested RDF document). To use link services, it is almost always requires multiple requests from the clients. Thus, to make the service transparent to the clients and to enable clients to orchestrate link services easily, a link service proxy was built that can be used from the client side with

any Linked Data application. When clients request a URI via HTTP, the proxy injects link relations to the requested RDF documents on the fly, hence augmenting Linked Data. The link service proxy was evaluated using four services we built during the enAKTing project: PSI backlink service, sameAs co-reference service, geo-reasoning services, and a link injection service. This work showed that these these services alone added 373 million across datasets foreign URIs, which almost doubles the previously mentioned 7.5% across-datasets foreign URIs coverage to the 14%. We also demonstrated how the linked service proxy works dynamically with the Web browser to enrich the Web of Linked Data. As all link services can be easily reused, and programmed to navigate the Web of Linked Data as well as generating new link services, we believe this provides a basis for agents to consume Linked Data. Following this trend, the Linked Data consumers will only need to orchestrate or create the link services to consume the Web of Linked Data. Any other Web-based Linked Data applications can be understood as specialised services to be built on top of the link services.

Contents

List of Figures

Acknowledgements

Foremost, I would like to express my sincere gratitude to my supervisor Dr Nicholas Gibbins, Prof. Dame Wendy Hall, Prof. Sir Nigel Shadbolt, for the continuous support of my Ph.D. I need to thank Dr. Gibbins for giving me a great research topic, his immense technical knowledge (especially on the W3C standards), brilliant ideas and hands-on help with writing this thesis. I thank Prof. Hall for her management and her vision - this thesis is heavily influenced by her previous work on the Microcosm. She is inspirational and a living legend in the hypertext world. Things I leant from her are beyond this thesis and valuable for a lifetime. I also need to thank Prof. Shadbolt for funding this study and making everything possible. Prof. Shabolt is a charismatic leader who initialised the enAKTing project and bound everyone together.

Secondly, I would like to thank Prof. Sir Tim Berners-Lee and Prof. Ted Nelson, for their previous work. I have always admired Prof. Berners-Lee for his wisdom and philosophy and I have learnt a lot from him by reading his free online design issue notes. I thank Prof. Ted Nelson for making Computer Science imaginative, fun and colourful.

Finally, I would like to thank colleagues, Dr. Salvadores, Dr. Gorrendo, Dr. Glaser and Prof. Carr for their expertise and works on the link services. I thank Quintin Gee for proofreading this thesis, and Jiadi Yao and Xin Wang for listening my research. The University of Southampton has a reputation and history on researching the hypertext topic. Although there are no many people who researching on this topic, I feel very honoured to be chosen to do this research. I also hope someone will receive the torch to continue this research.

Chapter 1

Introduction

"Data is a precious thing and will last longer than the systems themselves."

- Tim Berners-Lee

The World Wide Web [18] is a hypertext system which constructs a network of documents by embedding hyperlinks on Web pages, so that users can follow links to discover other web pages. This hyperlink network idea enables the discovery of information resources [117] as well as creating network effects [134], and has proven its effectiveness in daily use of the Web. Although the Web is the most successful hypertext system built, researchers consider that it has a number of limitations [88, 93]. For instance, it does not provide any information structuring facilities beyond hyperlinks; and its one-way links have no ability to determine which other documents refer to a particular document, therefore leading to inconsistencies when documents are moved or deleted. These can be considered either

as limitations or as missing features. Many pre-Web hypertext systems often had built-in features that enabled users to retrieve, navigate and create links. In contrast, on the Web, we observed patterns that these features emerged as third party services. Google is an example of a link service, which indexes large numbers of documents with metadata and helps users retrieve and navigate between web documents. Google, ranked by Alexa as the second most visited website globally[1], plays a crucial role on the Web, but is not a part of the Web infrastructure. There are many other link services, as shown in Figure 1.1, that play a role which extends the Web to make it a more powerful hypertext system.

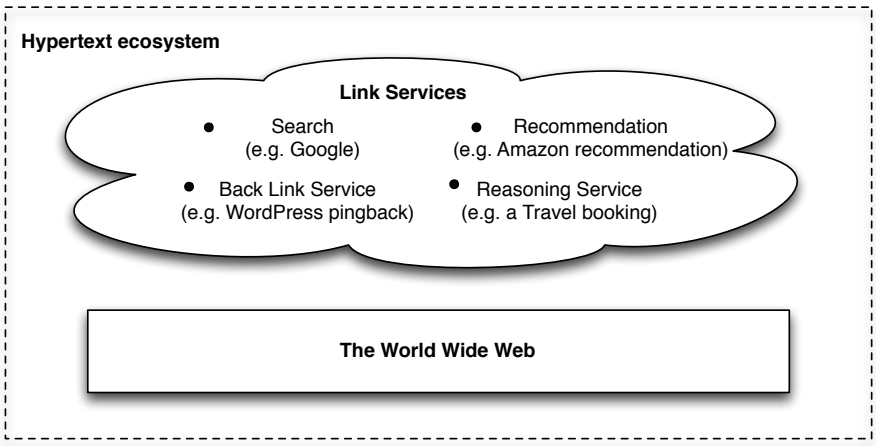


Figure 1.1: The hypertext ecosystem

The Semantic Web [26] is an extension to the Web that enables people to share content beyond the boundaries of applications and websites. Linked Data [15] at its core, aims at the creation of a network of data. Linked Data achieves this by using the (Semantic) Web standards URI and RDF to publish data in a similar fashion to publishing web pages by using URIs and HTML. Just as the

[1]http://www.alexa.com/siteinfo/google.com as 09/2013

link structure of the Web affect the discoverability of Web documents, the way in which this network of data is linked will determine how data can be discovered. Discoverability (navigability of the data) is an important issue because a piece of data can only be used after it has been discovered. According to the State of the Linked Data Cloud report [29] that about about 41% of datasets have fewer than 1000 outgoing links to other datasets which is rather low compared with the number of resources within the datasets. In other words, the Web of Linked Data is currently not well linked. Can we build link services similar to the Web link service which extend Linked Data to make it a better linked hypertext system?

1.1 Research Question

This thesis studies the connectivity of the Web of Linked Data and proposes building a number of link services to improve its linkage. This is an important research issue, as the link services will play a crucial role to in helping agents consume the Web of Linked Data and in making it a more powerful hypertext system. The first study is of navigability of the Web of Linked Data, how agents can navigate the web of data by dereferencing URIs. Then the idea of building link services to improve the navigability of the Web of Linked Data is investigated. To that end, the following research questions have been identified.

Q1: How can an agent understand what a URI refers to by dereferencing it?

The hyperlinks on the Web are different from those on the Semantic Web. A URI is used to identify a real world object on the Semantic Web instead of a document locator on the Web. A 'paramount requirement' of the Semantic Web is that an agent can follow a URI to understand what that URI refers to. This is the fundamental basis of the navigability of Linked Data.

Q2: How is the Web of Linked Data interlinked?

The idea of Linked Data is that by resolving a URI and retrieving the RDF descriptions, the links in an RDF document can be followed to discover other data. By following the links in the RDF documents, can an agent navigate and discover other datasets?

Q3: What services can be built to extend Linked Data to make the data better linked?

Based on studying link services for the Web, services are summarised that can be built on Linked Data. It is hypothesised that using link services dramatically improves the current linkage of the Web of Linked Data.

This thesis is structured to answer each of above research questions in Chapters 2 to 5.

1.2 Contributions

This thesis studies Linked Data as a hypertext system and proposes link services to improve the Linked Data system as whole. By answering the research questions above, our major research contributions in the area of the Web hypertext system are:

How can an agent understand what a URI refers to by dereferencing it? (Chapter 3)

URIs are used with Linked Data to identify resources instead of to identity web documents. Since it is used differently from the URI on the Web, when building it on the Web, the URI identity must kept consistent. W3C proposed two approaches to publishing URIs which identify real world objects. However, quite a few issues of URI resolution are not addressed. In order to study this identity issue systematically, the HTTP URI dereferencing process was formally modelled according to its specification and summarised in a set of URI Rewriting Rules. This is essentially a generic dereferencing algorithm that can be used for agents to dereference a URI and to understand what the URI refers to. Based on the algorithm, a Linked Data validator named Hyperthing was implemented that can be used to help data publishers to publish Semantic Web URIs as well as differentiate whether the requested resource is a real world object or a web document. To evaluate the validator and the current deployment of the Semantic Web URIs, an empirical study was carried out into dereferencing the most frequently referenced 25 ontology URIs and their properties. It was found that, on average, more than

half of Linked Data ontology URIs did not follow the published guidance to make the URI identity consistent. The Hyperthing validator can be used to avoid these publishing mistakes.

How is the Web of Linked Data inter-linked? (Chapter 4)

To study this issue, the navigability of the Web of Linked Data was first modelled. Two studies were then carried out to analyse how the Web of Linked Data are linked. It was found that only 7.5% resources on the Linked Data cloud had across datasets foreign URIs (URIs enabled to discover other datasets), which is relatively low. The case study demonstrated that the datasets exhibited two extreme patterns: either the majority of data instances are one-to-one parallel linked from one dataset to another, or the majority of the data instances from one dataset only link to a few data instances of another dataset. This means that, for those linked datasets, an agent can only follow some of the resources in a dataset to discover other resources and their navigability could be rather limited. The linking patterns of these across-datasets URIs seem monotonous. More diverse linking patterns are desired to enable the discovering of data.

What services we can be built to extend Linked Data to make the data better linked (Chapter 5)

To improve the navigability of the Web of Linked Data, a list of link services was built. The link services are RESTful services, and takes a link as input

and provides a RDF document with linking information of the requested URIs. They are: Resolution service (retrieves the RDF description of the requested URI for agents), Link extraction service (extracts URIs from a RDF), Linkbase service (third party hosting link relations between datasets, especially for those data which were not originally linked), Reasoning service (applies rules of reasoning to generate a new RDF), Composition service (compose multiple RDF documents into one documents), and Link injection service (inject extra links relations into the client requested RDF document). To use the link service, it is almost always requires multiple requests from the clients. Thus, to make the service transparent to the clients and to enable clients to orchestrate link services easily, a link service proxy was built that can be used from the client side with any Linked Data application. When clients request a URI via HTTP, the proxy injects link relations to the requested RDF documents on the fly, hence augmenting Linked Data. The link service proxy was evaluated using four services we built during the enAKTing project: PSI backlink service, sameAs co-reference service, geo-reasoning services, and a link injection service. These services alone added 373 million across datasets foreign URIs, which almost doubles the previously mentioned 7.5% across-datasets foreign URIs coverage to the 14%. We also demonstrated how the linked service proxy works with the Web browser to enrich the Web of Linked Data dynamically. As all link services can be easily reused, and programmed to navigate the Web of Data as well as generating new link services, we believe this provides a basis for agents (both human and machine) to consume Linked Data. Following this trend, the Linked Data consumers will only need to orchestrate or create the link services to consume Linked Data. Any other web-based Linked Data applications can be understood as specialised

services to be built on top of these link services.

1.3 Declaration

I, Yang Yang, declare that the work presented in this thesis is my own original research. The works presented in Chapters 2 to 5 were completely carried out on my own. The major contribution of this thesis is the link service proxy as presented in Chapter 6 which was designed and implemented completely on my own. I had some collaboration within in section 6.1.2.1 Backlink Service and 6.1.3 Reasoning Service, although in this section the description and summary of these services are my own original contribution. However, a few service exemplars used to describe services, were collaboration work I did during the enAKTing project by which I am funded. I was only partially involved in the design and implementing these services and they constitute a relatively minor contributions towards this thesis.

1.4 Publications

Y. Yang, P. Singh, J. Yao, C. Yeung, A. Zareian, X. Wang, Z. Cai, M. Sal- vadores, N. Gibbins, and W. Hall. (2011) Distributed human computation frame- work for Linked Data co-reference resolution. In The Semantic Web: Research and Applications, pages 32?46. Springer, 2011.

Salvadores, M., Correndo, G., Szomszor, M., Yang, Y., Gibbins, N., Millard, I., Glaser, H. and Shadbolt, N. (2010) Domain-Specific Backlinking Services in the

Web of Data. In: Web Intelligence, 2010, Toronto, Canada.

Correndo, G., Salvadores, M., Yang, Y., Gibbins, N. and Shadbolt, N. (2010) Geographical Service: a compass for the Web of Data. In: Linked Data on the Web (LDOW2010), 27 April 2010, Raleigh, North Carolina, USA.

Omitola, T., Koumenides, C. L., Popov, I. O., Yang, Y., Salvadores, M., Correndo, G., Hall, W. and Shadbolt, N. (2010) Integrating Public Datasets Using Linked Data: Challenges and Design Principles. In: Future Internet Assembly, 16-17 December 2010, Ghent, Belgium.

Omitola, T., Koumenides, C. L., Popov, I. O., Yang, Y., Salvadores, M., Szomszor, M., Berners-Lee, T., Gibbins, N., Hall, W., schraefel, m. and Shadbolt, N. (2010) Put in your postcode, out comes the data: A Case Study. In: 7th Extended Semantic Web Conference, May 2010, Greece.

Chapter 2

The World Wide Web and Hypertext Link Services

What is a hyperlink on the Web?

The World Wide Web is essentially a hypertext system, and there are many pre-Web Hypertext Systems. This chapter briefly looks the hypertext concept, and then concentrates in more detail on a few pre-Web hypertext systems which were considered novel in terms of publishing mechanisms. Finally, the World Wide Web is studied in detail along with link services that seek to improve the linking of the Web of documents.

2.1 The Hypertext Concept

"Let me introduce the word 'hypertext'." Ted Nelson [110]

The origin of hypertext is usually considered to be an article entitled *As We May Think*, written by Vannevar Bush in 1945 and published in the Atlantic Monthly [40]. Prior to Vannevar Bush, the earlier vision of hypertext can be tracked back to Paul Otlet's work in 1903 [115] which is less frequently cited [1]. Otlet, a Belgian lawyer turned bibliographer worked on the idea of a structure of documents with indices and references between them. The problem he tried to tackle is information overload in sociological sciences, and he proposed "the creation of a kind of artificial brain by means of cards containing actual information or simply notes or references" [116]. The idea is to strip each article or each chapter in a book of whatever is a matter of fine language or repetition or padding and to collect separately on cards whatever is new and adds to knowledge. These cards are then annotated as to the genus and species of the information they contain and placed in a general alphabetical catalogue updated each day. This he called his Monographic Principle. Based on this, he developed the Universal Decimal Classification System (UDC) which is still widely used in libraries in many countries [141]. While the technology used was based on index cards and sheets of paper, the system he envisaged would effectively become a universal, dynamic Encyclopaedia, "formed by linking together materials and elements scattered in all relevant publications." In order to improve his system, he investigated many forms of mechanisation or automatization to enable data to be easily entered and

[1] According to the Googlescholar citation

classified. According to his papers, he proposed ideas to use a cyclops camera to record books, microfilm to save the data, a vocoder to transform voice into handwriting and vice versa, using television and telephone to read the text of book remotely, and selected machines to perform certain specialised tasks such as 'searching', as well as machines to enable a 'teletype' telegraphic encoder to enter data and text autocompletion to speed up the data entry process [116].

Bush was an American scientist and public figure, who coordinated researchers from many domains. In his famous article "As We May Think", Bush imagined a device called a Memex which could store personal information such as books and records, and could function as a supplement to ones memory. Similar to the Otlet, he tried to tackle the information overload problem in the research domain. He detailed and emphasized that the essential feature of the Memex is that information can be associatively indexed, and any item in the system can be immediately and automatically selected by another. The reason behind his design decision is that human memory operates by associating one item with another, due to the supposed intricate web of thoughts carried by the brain's cells. In the Memex, users can build trails to follow articles, and articles are connected by simply pressing a button. Trails are named and stored, so that later on, users can use the trails again. The idea of a trail is very similar to what we would now recognise as the hyperlinks on the Web. Certainly, there is some similarity between Otlet's work and Bush's, but Bush did not cite of Otlet's work in his paper. Some claim that Bush was aware of Otlet's work indirectly through Watson Davis' (who was a pioneer in the field of library and information science) visit in 1932 [9]. However, Bush wrote against an indexing system and

considered it artificial and cumbersome, and difficulty for retrieving information [39]. On the other hand, the structured information helps exchange information between systems. He stated that one may duplicate one's trails and give it to a friend to put into their Memex as their own trail. The structure of trails was not discussed.

Douglas Engelbart is another hypertext visionary who devoted his career to "augmenting the human intellect" [61]. In pursuit of this goal, he invented many computer devices which we still use today: the mouse, the window, the word processor, video conferencing, remote procedure calls and more [60]. In his 1962 report "Augmenting the Human Intellect" [61], he described a conceptual framework H-LAM/T system (Human using Language, Artefacts, Methodology, in which he/she is Trained). He defined four classes of possible augmentation means, through intertwining artefacts, language, methodology and training. He also discussed thought processes and repertoire hierarchies (e.g. how to use a tool, methods, strategies and rules), and how they influence problem-solving capabilities. He then defined severval structures of repertoire hierarchies such as mental structures, concept structures and symbol structures as well as their relationships with each other. He hypothesised that "better concept structures can be developed structures that when mapped into a human's mental structure will significantly improve his capability to comprehend and to find solutions within his complex problem situations." The report also referenced Bush's As We May Think, where Engelbart stated that "the Memex makes a very convincing case for the augmentation of the individual intellectual work" and fits his conceptual framework. He also wrote a personal email to Bush to ask his permission to quote

Bush's work in his report and express how much influence that paper had made on his personal life [1]. He considered that the 'associative trails' of Memex add a factor of speed and convenience to ordinary filing-system (symbol-structuring) processes; they can augment the human's process structuring and executing capability. Engelbart also considered that the classification is significant to the system and experimented with tags and links. However, he concluded that it is time-consuming to set up the nodes and links structure and "on the one hand that the links and nodes structures became much cleaner and required fewer members, and on the other hand that we could get considerably more sophisticated help from the computer in doing significant chores for us". He describes the ontology creation process of using category and relationships as well as a method for dealing with the semantic ambiguity [61].

The term hypertext, which conjoins 'hyper' and 'text', was coined in 1965 by Ted Nelson in his paper *Complex information processing: a file structure for the complex, the changing and the indeterminate*, based on studying the Bush's Memex [110]. 'Hyper' used as prefix, derives from the Greek hypér, originally meaning over, or above, but whose meaning typically implies excess or exaggeration [152]. Nelson [110] explains that hyper used in this context connotes extension and generality; the word 'hyperspace' and the criterion for this prefix means the inability of these objects to be comprised sensibly into linear media, like the text string, media and so on. He called the Executable and Linkable Format (ELF) a hyperfile. 'Text' has the original meaning of words woven together [152]. The word hypertext thus implies both a super text, a text that due to interlinking is

[1] http://sloan.stanford.edu/mousesite/EngelbartPapers/LetterToVBush.html

greater than the original texts, and a super weaving of words, creating new texts from old. In Nelson's original vision:

"hypertext means a body of written or pictorial material interconnected in such a complex way that it could not conveniently be presented or represented on paper. It may contain summaries, or maps of its contents and their interrelations; it may contain annotations, additions and footnotes from scholars who have examined it... such an object and system could have great potential for education, increasing the student's range of choices, his sense of freedom, his motivation, intellectual grasp. Such a system could grow indefinitely, gradually including more and more of the world's written knowledge." [110]

In his Literary machines [111], he gave a more concrete definition : *"hypertext means non sequential writing text that branches and allows choice to the reader, best read at an interactive screen."*

"hypermedia is used as a logical extension of the term hypertext, in which graphics, audio, video, plain text and hyperlinks intertwine to create a generally non-linear medium of information."

The link is used as a reference to data that the reader can directly follow, or that is followed automatically, and is referred to as a hyperlink.

2.2 Hypertext Systems

In the 1960s, Ted Nelson started the Xanadu project [111]. The idea was to create an interlinked universal document database (which he named the 'docuverse'). In his vision of Xanadu, users are able to form arbitrary links between documents via 'transclusion' an inclusion-by-reference mechanism which was designed to support copyright control. The file system in Xanadu supports subdivision of addressing across multiple dimensions: for example, it is possible to subdivide the concept of a movie by its director, actors and genre. It was designed to enable anyone to effectively edit any content by signing it using a new identity. Many of these proposals can be achieved in another form by using today's Semantic Web technology. The first attempt at implementation of Xanadu began in 1960, but it was not until 1998 that an implementation was released. [155]

Douglas Engelbart demonstrated the first working hypertext system named oN-line System(NLS) in 1968 [59]. As previously mentioned, he intended to augment human intellect, through collaboration in shared space [61]. Therefore, Engelbart's vision of hypertext mainly focused on human communication and collaboration through the computer. NLS was used for cross-referencing research papers for sharing among geographically distributed researchers. NLS was a sophisticated system that included the use of television images and the first appearance of the mouse. Files in NLS were structured as a hierarchy of statements. Links could exist between any two files or between statements.

The second working hypertext system was the Hypertext Editing System (HES), which was developed by researchers at Brown University, lead by Andries van Dam and Ted Nelson in 1969 [41]. The system was intended to serve two purposes: to produce print nicely and efficiently and to explore the hypertext concept. This system was later continued as File Retrieval and Editing System (FRESS) [57], in which bi-directional links were implemented and links could be stored in link databases separately from the documents. According to van Dam, FRESS was the first system to have an undo function. In 1985, Brown University developed another hypertext system named Intermedia [157]. The Intermedia was used in teaching Biology and English Literature at Brown. It was used both as a tool for professors preparing their lessons and course material, and by students for learning and creating reports. The distinctive feature of Intermedia was the separation of links and document data, where information about links was stored in link databases. Intermedia intended to ease link management where links could be shared by participating applications. Microcosm [65], a system developed by a research team led by Wendy Hall at the University of Southampton, took this vision further. The core of Microcosm is in essence a set of communication protocols that enable the integration of all types of information processing tools, including a hypermedia link service. Links were stored separated from the documents and link data were stored in link databases, which named link bases to support open hypertext functionality on diverse document formats.

Both the Microcosm and World Wide Web (WWW) hypertext systems all started around 1989; other systems in this period worthy of mention are the Hyper-G [93] and HyTime [112]. The Hyper-G was initially created as an individual hyper-

text system, and later developed into a system intended to improve the WWW in terms of hyperlink management, searching, dynamic content, maintenance of large datasets, authoring and scalability. An algorithm named p-flood was used propagate changes to links and nodes through the Hyper-G system. In Hyper-G, information about surface links (links from one server's resource to another server's resources) was passed to all interested parties with the aim of providing referential integrity. HyTime is a markup language which was based on the Standard Generalized Markup Language (SGML). It was intended to preserve information about the scheduling and interconnection of related components of a hypermedia document (e.g. audio). HyTime supports advanced linking features and provides five kinds of hyperlinks: independent (n-rary links), property (typed links), contextual (contextual links), aggregate (links handle co-reference links), and span (an implementation for specifically purpose dealing with SGML elements).

Due to the requirements of large-scale electronic publishing, XML [37] was a more flexible text format than HTML (Hypertext Markup Language), derived from SGML, became a W3C recommendation standard for publishing data on the Web. Similarly, a language related to HyTime named Xlink (the Extended Linking Language) [53] was designed to enable link creation in XML. Xlink provides a framework that allows XML documents to assert linking relationships among two or more resources, associate metadata with a link, and express links that reside in a location separate from the linked resources.

In the above-mentioned systems, hyperlinks play a variety of different roles. The

following section examines and summarises the types of hyperlink in hypertext systems.

2.3 Types of hyperlinks

Whitehead [153] describes five classes of links style in hypertext systems. The first is links as addresses - the address of the link destination is embedded within the code, e.g. the Web [18]). Next comes links as associations, where links are expressed as an association between entities, e.g. Intermedia [157]). The third class is links as structural elements - here links are used to represent hierarchical, or other organisation of material, e.g. NoteCards [149]. Next comes links and rhetorical representation - links represent the structure of an argument, e.g. gIBIS [47]). Finally, links as a semantic network - link types are used to represent semantic relationships between nodes, e.g. MacWeb [109]. For each type of linking, the links can be uni-directional (e.g. the Web), bi-directional - by following the link in reverse from its destination document, the source document can be found (e.g. NoteCards [148]) or n-ary links - links that can be anchored to either nodes (basic objects) or other links (e.g. Aquanet [100]).

Traditionally, the primary notion of a link has been of a navigable relationship between nodes, with link traversal causing a transition to one or more destination nodes. Different notions were also explored in other hypertext systems. For instance, the notion of linking in a spatial hypertext system is based on recog-

nising patterns and inference among physically proximate nodes, e.g. Viki [101] and VKB [140]. In the botanical taxonomy system [114], hypertext was used to model the taxonomies to structure the data.

Although links demonstrated their power in many computational systems, some researchers consider linking to be just a special case of a general philosophy of computing in which structure is more important than data [113]. Similar opinions were shared in the paper *Linking Considered Harmful* [52], whose author emphasised the analogy between the node-link model and the GOTO [55] statement in programming languages and consequently advocated the use of high-level structuring in hypermedia. Meyrowitz [108] (one of the main developers of the Intermedia system) in a keynotes address to Hypertext'89 supposed that hypertext should be an intelligent infrastructure to link together information at users' fingertips in an information enviroment. For example, when a user transfers data by a copy-paste operation, a temporary link can be formed between the source document and the destination document, but hypertext could make these links persistent -even across applications if the basic linking mechanism was a system hypertext services. Intermedia is in fact such a system, which was built on the principle of providing a standardised linking protocol for other applications to use. However, the link service is only internal to applications and not integrated with the complete computing environment. This vision of decoupling of link structures and data evolved into the open hypertext systems discussed in the following section.

2.4 Open Hypertext System

Open hypertext systems enable links to be freely used by decoupling links and nodes (nodes can be documents, data, cards). For example, links could be injected into a spreadsheet, word processor or on a webpage. Open hypertext aims to act as an underlying hypertext link service [119], which can be accessed from any available applications, thus acting as a service component of the user's environment. By comparison with a closed system which embeds link information into documents, open hypertext has a number of considerations [43]:

It reduces authoring and maintenance effort - for example, the use of generic links in the Microcosm enables common links to be created only once; wherever the source selection of the link occurs, the link is available. Storing links in a link service also reduce problems frequently encountered on the Web such as link fossilisation and decay [88] (As documents are moved, edited or deleted, any document that refers to it must also be altered to reflect this change).

It enhances user's experiences - open link services can provide underlying hypertext facilities to the user's whole environment, not only limited to a single application.

It can also provide an alternative views - open link services also enable the provision of multiple link databases for user to select from. It also enables

the separation between information provider and link provider. On the web, the hypertext links are bound to associated materials - links are embedded in the webpages.

Although the concept of open hypertext systems was invented about the same time as the Web, the Web system gained adoption more quickly. After it became the most widely used hypertext system, many hypertext systems started to either integrate with the Web or become extensions of the Web. Some researchers [151] consider that it is worthwhile to integrate the open hypertext system and the Web as they complement each other. The Web offers limited but scalable hypermedia services; in contrast, open hypertext services provide advanced hypermedia services across an open set of applications and data, but not as scalable as the Web.

2.5 The World Wide Web

The World Wide Web was invented at the International Laboratory for Particle Physics (CERN) in Geneva. It was designed for CERN remotely-located employees to share and exchange documents electronically via different computer system environments. Before the Web, Berners-Lee developed ENQUIRE for CERN in 1980, which was a simple hypertext program that enabled users to store snippets of information and links to related pieces. To retrieve information, one progressed via the links from one sheet to another. The ENQUIRE was very similar to Ap-

ple's Hypercard, but without the image rendering system. However, a difference was that the system ran on a multi-user system and allowed many people to access the same data. Starting from 1984, Berners-Lee extended the system to enable it to point to documents from other machines, using the existing Internet infrastructure. Based on this, he proposed a system called "Mesh", which incorporated technologies that have developed into today's Web. Berners-Lee also developed the first applications to use the Web: the first Web browser and server. In the original implementation, Web documents were both readable and editable. However, the "Mesh" was not well adopted, but it was an open and distributed architecture, features that allowed others to develop their own browsers and publish their own documents. Mosaic was the first widely available browser, which made the Web more popular.

The inventors of the Web,Tim Berners-Lee and Robert Cailliau, had two main criteria when designing the system: 1) an open design so that the system operates on different computer architectures and 2) network distribution so that the system can be shared over distributed communication systems. The Web is built on top of the Internet. The three major architectural bases of the Web are: 1) identification of resources; 2) interaction protocols for exchanging representation of resources; 3) data formats which represent data and metadata and transfer between agents [135]. The Web, as originally envisioned, embodies the three components above as URI (Uniform Resource Identifier) [23], HTTP (Hypertext Transfer Protocol) [64] and HTML [128] respectively. The URI is the global naming mechanism on the Web. As Berners-Lee elaborated, the two Universality rules of the Web are "any resource anywhere can be given a URI" and "any resource of significance should

be given a URI" [11]. If a resource is identified by URIs that use the http:// scheme, when dereferencing a URI in a web browser, the browser communicates with a server through HTTP and a resource represented as a document in HTML format is retrieved. Figure 2.1 illustrates the relationship between identifier, resource, and representation of a Web Resource Oaxaca Weather Report.

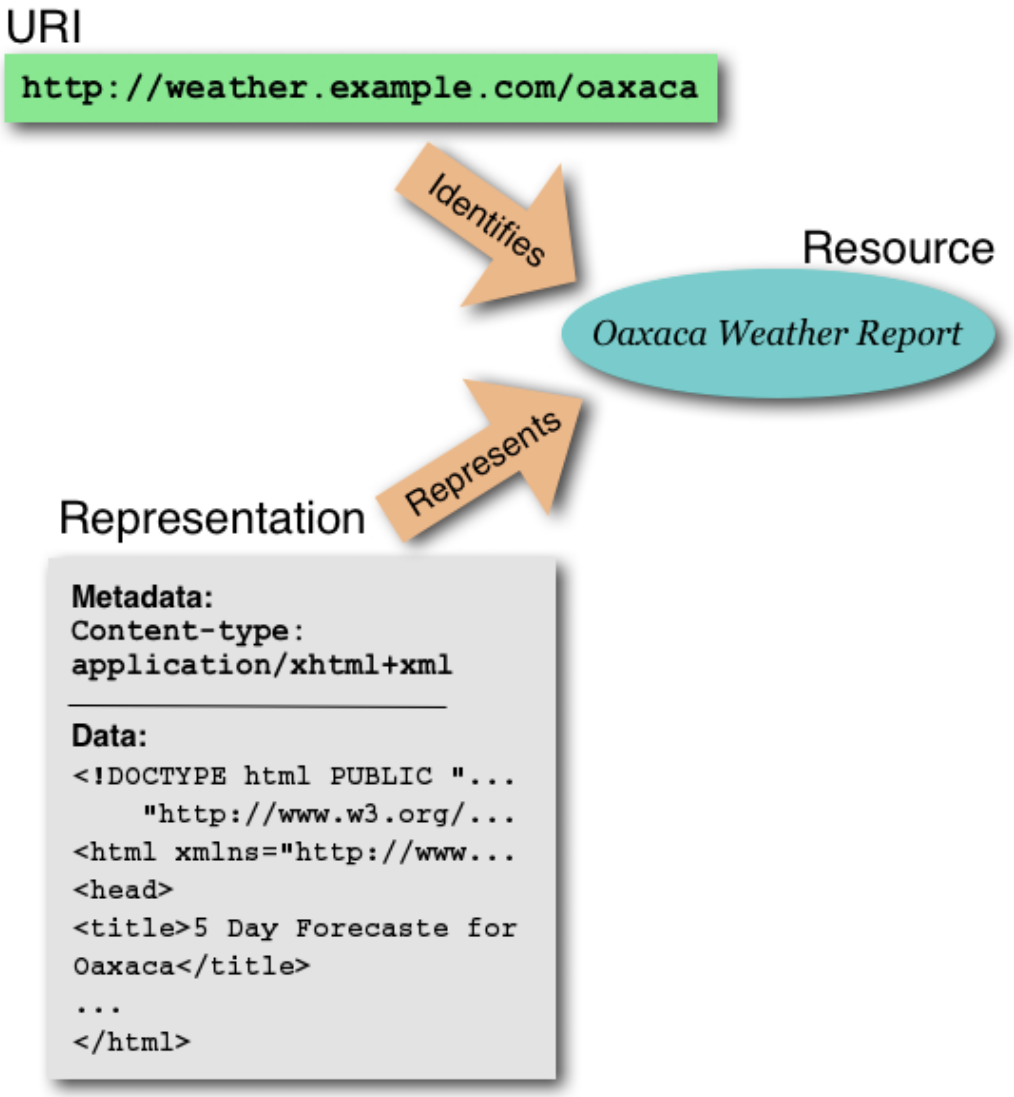


Figure 2.1: Identifier, resource and representation of a Web Resource Oaxaca Weather Report [90]

According to Galloway's definition - a protocol is a convention for transmitting information between two or more agents [67]. A protocol often specifies more than just the particular encoding, but also may attempt to specify the interpretation of this encoding and the meaningful behaviour that the sense of the information should engender in an agent. A payload is the information transmitted by a protocol. For instance, in a protocol like TCP/IP, the payload transmitted is bits in the body of the message, with the header being used by TCP to ensure lossless delivery of the bits. TCP/IP transmits strictly an encoding of data as bits and does not force any particular interpretation on the bits. The Web's HTTP is designed to transfer hypertext documents and is defined by Internet Engineering Task Force (IETF) Request for Comments RFC 2616 [64]. The following section look in detail at the resources, then at the Hypertext Transfer Protocol and finally at the definition of the Uniform Resource Identifiers.

2.5.1 Resources and Representation

"Someone, let's say a baby, is born; his parents call him by a certain name. They talk about him to their friends, other people meet him. Through various sorts of talk the name is spread from link to link as if by a chain. A speaker who is on the far end of this chain, who has heard about, say Richard Feynman, in the market place or elsewhere, may be referring to Richard Feynman even though he can't remember from whom he first heard of Feynman or from whom he ever heard of Feynman." Kripke [94]

A URI identifies a resource, as originally defined in the earliest specification [22]. However, itself was not defined in the specification. Berners-Lee later elaborated by saying that a resource can be "anything that has an identity" which is not only information that is accessible via the Web, but also real world objects, e.g. human beings, cars, etc. [24]. In the W3C recommendation *Architecture of the World Wide Web* [90], the term **resource** is used as a general term for whatever might be identified by a URI, and it does not limit the scope of what might be a resource. It is conventional on the hypertext Web to describe Web pages, images, etc. as resources. The distinguishing feature of these resources is that all of their essential characteristics can be conveyed in a message: such a set of resources are named as **information resources**. Hence, everything else, such as real world objects are defined to be **non-information resources**.

The representation on the Web is historically defined in HTML as "the encoding of information for interchange" [20]. W3C later defined it as "data that encodes information about resource state" [90]. A Web representation contains two components: a headers (e.g. a media type which describes how the information is encoded) and a payload (the encoding of the state of the resource at a particular time). A Web representation can only be given as a response to a request like HTTP GET.

When data publishers publish different web representations on their servers, the client can request their preferred or acceptable Internet media type for the representation. This can be achieved by using a Content Negotiation mechanism, which makes it possible to serve different versions of a document (formally a re-

source representation) under the same URI [64]. For example, a server can be configured to serve an image in both GIF and PNG format, and if a web browser cannot display the PNG images, the GIF version can be served instead.

It is also important to understand that the URI owner defines what a URI means. One can serve two different images (e.g. image of an apple and image of a dinosaur) under the same URI which is named as `http://www.example.com/apple` or do any other possible unusual things. Other agents/people understand the URI by dereferencing it. This behaves in a similar manner to Kripke's causal theory of proper names [94] - a name's referent is fixed by an original act of naming and later uses of the name succeed in referring to the referent by being linked to that original act via a causal chain. On the Web, one may create a URI (a name to identify an information resource) and webpage (a representation of an information resource), and bind this name and representation (as shown in previous section Figure 2.1) to define an information resource. Later, a user can re-use and reference this information resource on other webpages by embedding this URI. Consumers look up the URI and read the representation of the resource to understand what this information resource is about.

2.5.2 Hypertext Transfer Protocol

HTTP is the foundation of data communication for the Web. It functions as a request-response protocol in the client-server computing model. A web browser,

is typical example of a client and an application running on a computer hosting a website (a set of web documents), is typical example of a server. The client submits an HTTP request message to the server. The server returns a response message to the client. The response contains status information about the request and may also contain requested content in its message body. In the following thesis, we will use **agent** to refer to clients such as web browser, web crawlers, mobile apps and any other software that accesses, consumes the Web.

HTTP is a protocol follows REST(Representational State Transfer) [62] architectural style. A REST-style protocol conventionally consist of clients and servers. Clients initiate requests to servers; servers process requests and return appropriate responses. Requests and responses are built around the transfer of representations of resources. A resource can be essentially any coherent and meaningful concept that may be addressed. A representation of a resource is typically a document that captures the current or intended state of a resource. Fielding [62] summaries few characteristic of the REST style:

Client-Server : A client is a triggering process; a server is reactive process. Clients make requests that trigger reactions from servers.

Stateless : Each request from client to server must contain all of the information necessary to understand the request and cannot take advantage of any stored context on the server. The session state is therefore kept entirely on the client.

Cache : Data within a response to a request are implicitly or explicitly labeled as cacheable or non-cacheable. If a response is cacheable, then a client cache is given the right to reuse that response data for later.

Uniform Interface : REST is defined by four interface constraints: identification of resources (e.g. URIs), manipulation of resources through representations (e.g. modify the resource via HTML webpage); self-descriptive messages (enough information to describe how to process the message e.g. its encoding); and hypermedia as the engine of application state (HATEOAS) that is clients make state transitions only through actions that are dynamically identified within hypermedia by the server (e.g. by hyperlinks).

Layered System : Unless specialised, a client has no need to be aware of whether it is connected directly to the end server, or to an intermediary along the way. Intermediary servers may improve system scalability by enabling load-balancing and by providing shared caches. They may also enforce security policies.

Code-On-Demand : As an optional, it allows client functionality to be extended by downloading and executing code in the form of applets or scripts.

HTTP defines methods to indicate the desired action to be performed on the identified resource. It defines seven methods, namely, GET, HEAD, POST, PUT, DELETE, TRACE, and CONNECT. The discussion here is mainly about GET and POST and PUT, the three most frequently used methods on the Web. The

GET method retrieves whatever the representation of the resource identified by the Request-URI. The POST method is used to request that the destination server accept the entity enclosed in the request as a new subordinate of the resource identified by the Request-URI (e.g. when one fills a HTML form and it is used to transfer data to the server). The PUT method requests that the enclosed entity be stored under the supplied Request-URI (e.g. for uploading data directly to the server). Figure 2.2 top part shows a sample request to GET http://dbpedia.org/resource/Tim_Berners-Lee from a Mozilla 5.0 Web browser user agent.

An HTTP response from a server consists of an HTTP status code and an HTTP entity [67]. An HTTP status code is one of a finite number of codes which gives the user-agent information about the server's HTTP response itself. For example, HTTP 200 means that the request was successful, while 404 means that the user-agent requested data that was not found on the server. The HTTP entity is the information transferred as the payload of a request or response. Figure 2.2 b) shows an example of an HTTP response with status code 303 See Other. The content type is the formal language that can be explicitly given in a response or request in HTTP server. In the example, the content type is text/html, so the user agent interpreted the encoding of the HTTP entity body as HTML. The Content-types in HTTP is an 'Internet Media Types', which can be applied with any internet protocol [124], not restricted to HTTP or MIME (Multimedia Internet Message Extensions, an email protocol, the Internet Media Type originally named MIME) [66]. A media type, for example, text/html, consists of the type and subtype of encoding with a slash for the separation marker. IANA (Internet

Assigned Numbers Authority) [1] registers the Internet media types.

```
http://dbpedia.org/resource/Tim_Berners-Lee

GET /resource/Tim_Berners-Lee HTTP/1.1
Host: dbpedia.org
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10.7; rv:8.0.1) Gecko/20100101 Firefox/
8.0.1
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-us,en;q=0.5
Accept-Encoding: gzip, deflate
Accept-Charset: ISO-8859-1,utf-8;q=0.7,*;q=0.7
Connection: keep-alive

HTTP/1.1 303 See Other
Date: Mon, 13 Feb 2012 21:18:36 GMT
Content-Type: text/html; charset=UTF-8
Connection: keep-alive
Server: Virtuoso/06.03.3131 (Linux) x86_64-generic-linux-glibc25-64  VDB
Location: http://dbpedia.org/page/Tim_Berners-Lee
Content-Length: 0
```

Figure 2.2: An HTTP Request and Response from a server

2.5.3 Uniform Resource Identifiers

There are many other protocols built on top of TCP/IP, such as FTP (File Transfer Protocol) [125], SMTP (Simple Mail Transfer Protocol) [123] and Gopher [4] for the retrieval of documents. Since one computer can be used to provide many different kinds of information, IP addresses were not enough as they only identify the location of a computer on the network. Each protocol often creates its own naming scheme to enable it to refer to and access things at a more refined level than IP addresses. However, each protocol has its own naming conventions, and therefore, they cannot communicate with each other. The Uniform Resource Identifiers were designed to be used by other protocols as well. The URIs were

[1] http://www.iana.org/assignments/media-types/

invented to encode the name and addresses of objects on the internet. It accomplished universality over protocols by moving all information used by the protocol within the name itself. The notion of a URI can be understood as a 'meta-name', where it wraps a protocol of internet addresses as one name. This is similar to the reflection concept in programming languages (programming instructions are also stored as data), where how to access a resource identified by a URI is also defined in the name itself.

The full syntax of URI is defined in IETF RFC2396 [24], and Figure 2.3 shows the regular expression of URI. The scheme is the name of the protocol. The hierarchical component is the domain component of the URI which identifies an internet host. The question mark (?) denotes a query string which allows passing parameters to the HTTP request. The hash (#) is traditionally used to identify fragments of a hypertext document and it can now also identify a 'secondary resource' which is defined as 'some portion or subset of the primary resource, some view on representation of the primary resource, or some other resource defined or described by those representations'. [24]

URI = [scheme ":"] [hierarchical component]* ["?" query]? ["#" fragment]?

Figure 2.3: The regular expression of URI

When requesting a URI containing a hash, the process can be explained in three steps:

1. The client parsing the URI removes the fragment part and sends the request to the server

2. The server retrieves the requested resource representation and sends it back to the client

3. The client renders the the part of the representation of the resource identified by the fragment.

For example, when requesting a URI http://www.example.org/foo.html#bar with Internet media type text/html where the fragment refers to the element with id ='bar' , the typical web browser will render the HTML page and position page so that the top of the element identified by the fragment id is aligned with the top of the view port. For file type HTML, the fragment is the ID of an element within the HTML object. For file type 'application/xml', if it is just a word, then it is the XML ID of an element in the document.

Another important characteristic is that a URI has an owner (the creator). The owner is the agent that is accountable for originally determining what the URI identifies (e.g. the owner of a host who created URI in the first instance). The URI owner also defines the meaning of the URI by creating and altering the information accessible from the URI.

URIs are limited to a subset of the ASCII (American Standard Code for Information Interchange) character set, IRI (internationalized resource identifier) [58] is a generalisation of the URI, which may contain character from the Universal Character Set (Unicode/ISO 10646), including Chinese, Japanese kanji, Korean, Cyrillic and so on.

2.5.4 HyperText Markup Lanuage

The HyperText Markup Language is an application of the ISO [1] (International Organisation for Standardisation)'s SGML [71], which uses Document Type Definition (DTD) [49] to define a document markup language. Web browsers parse the markup to interpret and compose text, images and other media into a web page. The first version of HTML described 18 elements (named tags) that comprised the initial, relatively simple design of HTML. Apart from the hyperlink tag, many of the text elements[2] are found in the 1988 ISO technical report TR 9537 Techniques for using SGML, which in turn covers the features of the first text formatting languages RUNOFF [8] written in 1964 for the first time-sharing operating system CTSS [44]. The RUNOFF commands were derived from the commands used by typesetters to manually format documents. The SGML concept of generalized markup is based on elements with nested attributes which not only provide print effects, but also the separation of structure and processing. The CSS (Cascading Style Sheets) [36] language was designed for HTML to further enable the separation of document content from document representation, including aspects such as layout, colours and fonts. Default styles for every element of HTML markup are defined in the browser, and these characteristics can be altered or enhanced by CSS.

HTML2 [21] was developed under the aegis of the IETF to codify common practice in late 1994. HTML3 [127] proposed much richer versions of HTML. HTML4 im-

[1] http://www.iso.org/iso/home.html
[2] http://www.w3.org/History/19921103-hypertext/hypertext/WWW/MarkUp/Tags.html

proved richer tables, forms text formats and also improved accessibility for people with disabilities. In 1996, XML (Extensible Markup Language)[73] was developed as a language that is both human-readable and machine-readable and emphasises simplicity, generality and usability over the internet. XML is often used on the internet service as a data exchange format. In order to make the HTML more extensible and increase interoperability with other data formats, W3C decided to stop evolving HTML and instead work on an XML-based equivalent - XHTML [120]. The XHTML1.0 is simply a reformulation of HTML4 in XML. As the XHTML family evolves, documents conforming to XHTML will be more likely to interoperate within and among various XHTML environment. However, the XHTML was not well supported by the industry at the time, however W3C determined to continue developing XML-based replacements of HTML. In response to W3C's decision to abandon HTML in favourr of XML-based technologies and slowed development of HTML, Apple, Mozilla and Opera jointly formed a Web Hypertext Application Technology Working Group (WHATWG) [1] to continue development of HTML5 [87]. Later W3C decided to participate in the development and created the HTML5 Recommendation, which supports many fancy features such as 2D/3D graphics, local storage for the web applications and so on. The HTML5 improved the markup for documents and introduced many APIs (Application Programming Interfaces) that help in creating web applications, such as APIs for controlling video and audio playback, handling search providers, etc. HTML5 is considered to be a potential candidate for cross-platform mobile applications, as it is designed to be run on low-powered devices such as smartphones and tablets.

[1] http://www.whatwg.org/

2.6 Hypertext Link Services

The term 'link service' first appeared in Sun's Link Service paper [119]. The link service defines a protocol for an open hypertext system to enable a "closed" hypertext system to link to objects. The implementation consisted of a link database and a library which a developer could use to extend their systems. The service enabled the hypertext to be added via a series of popup windows without having a large impact on the existing system's interface.

Until the end of the 1980s, hypermedia systems were largely monolithic application. Intermedia was the very first system designed to have separate hyperlinks, with a link server which communicated with the rest of the hypermedia system using socket connections. This structure enables the easy maintenance of the hypertext system, as when nodes or links changed, they could be modified. In the system, in order to determine where to add a link within a node, the service performed an exhaustive search and assumed that the linked object was a single full line of text. As a deliberate design decision for, simplicity, the system did not support any specific link types, attributes or directionality. Although the authors designed a hypertext structure that separated links and data, they did not attempt to integrate the link service with other applications.

Multicard [133] was a large scale open hypertext system developed for an industrial environment. Multicard provides a hypermedia toolkit (e.g. editor, scripts language) that allows programmers to create and manipulate a distributed basic

hypermedia structure. One of the novelties of the system is that it does not itself handle the contents of the nodes: instead, it communicates with different editors. Links are annotated with scripts and communicate with each editor using event and message transmission. The default message is one to activate the destination object. Link endpoints could be anchors, nodes or groups of either. In Multicard, it was possible to attach scripts to any objects, in effect the endpoints of a link became handles to use in the scripts.

Finally, it is also worth mentioning is the Dexter Hypertext Reference Model [80] - a formal model of a hypertext system that was the result of a series of hypertext workshops in 1989. The Dexter model provides a systematic basis for comparing systems and developing interchange and interoperability standards. The model is divided in three layers, as shown in Figure 2.4 :

The runtime layer deals with the representation and interactive aspects of the system. The storage layer, as the focus of the system describes how the hypertext components and links are linked together, which essentially acts as a "database". The components contain the chunks of text, graphics, images, animations and so on. The within-component layer is responsible for the content selection of individual components through anchors and structure within the components of the hypertext network. The model considers the within-component structure as being outside the hypertext model. The function of the layer is to act as tools for the user to access, view and manipulated the network structure, as both the storage and within-component layers treat hypertext as a passive data structure. However, some researchers, such as Grnbk [75], consider that the Dexter model is

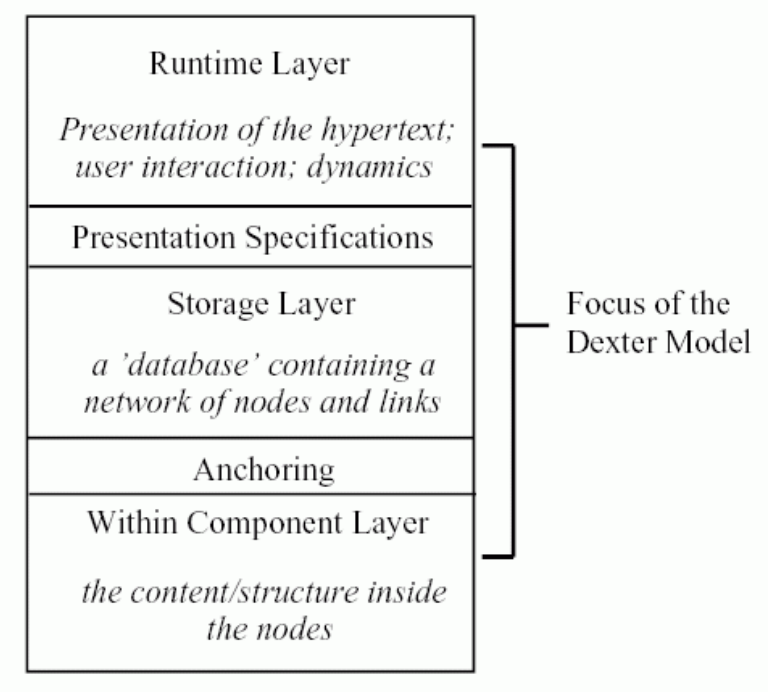


Figure 2.4: Dexter Hypertext Reference Model [80]

not adequate for systems whose linking is based on embedded references as in the Web system, nor for modelling the dynamic aspects of hypermedia systems as in Mircrocosm or DHM (Devise Hypermedia Model) at the time [74]. Grnbk [75] proposed a Dexter-based extensible object oriented model which introduces two new concepts, LocationSpecifiers (ability to specify a location in a body of electronic material) and ReferenceSpecifiers (when links appear in the components of hypermedia), which enable links as references to be embedded in document as well as links to be stored as objects in separate databases.

As the Web system became increasing popular, many hypertext research groups began to consider how to adapt and to integrate with the Web. The following section looks into link services for the Web.

2.6.1 Link Services for the World Wide Web

Although the Web is the most successful hypertext system built in history, researchers in the hypertext field consider that it has a number of limitations [88, 93]. For instance, its one-way links have no ability to determine which other documents refer to a particular document, therefore leading to inconsistencies when documents are moved or deleted. Some argue that to embed links within HTML, making the connections a part of the document node, restricts their reusability. In particular, researchers from the open hypertext community consider that links and documents (or nodes) should be managed separately, so that links can be freely used [88]. Based on these ideas, a number of services have been developed over time to improve the navigability of the Web (some of the services may not have been designed to solve the navigability problem, but improved navigability regardless). These services are summarised as below:

Linkbase Service : one of the limitations of the Web is that, once embedded URIs between document are created, others cannot contribute or add more links between documents. A linkbase service enables third parties to author links between documents by creating a database of links between documents (which originally may not have been linked). Given a URI, a list of links can be retrieved that relate to this URI, and therefore can navigate from one document to another. A backlink service [45], as an example, is a special linkbase service which provide a list of backlinks to enable users to discover a list of referral links of the browsing web page. This service enables the bi-

directional navigation of the Web. For example, Google backlink services, where users give search options for backlinks [1] of a website can retrieve a list of referral websites. Chakrabarti [45] designed a backlink system to enable the user to surf the Web backwards. The backlink metadata may be constructed from a variety of sources, including human editing, web crawling and automatic computation. There are proposals within the hypertext community for gathering backlinks. For instance, Bharat [28] proposed to build a server that provides linkage information for all pages indexed by search engines using the "link" option to index the backlinks. In addition, Atlas [122] was proposed as a distributed hyperlink database system [122] in which each of the Web servers creates a link base server called Atlas. The data rely on logging the HTTP referer header (see HTTP specification 14.36 [64]) and the Atlas server communicates amongst related servers.

Composition Service: there is much content on the Web that is related but not linked; this service aggregates a list of hyperlinks based on certain criteria or create links to enable navigation between all these webpages. Yahoo [95] and Yellow Pages [146] are among the early hypertext composition services, where services provide human-organised directory browser to facilitate the navigability of the Web. Pirolli et al. from Xerox [121] have also pointed out that the Web lacks much explicit structure and tried to automatically categorise web pages by using spreading activation techniques [3]. There are many other domain specific composition services, such as product rec-

[1] http://tiny.cc/gd5wlw

ommendation services, etc.; but an in-depth discussion is beyond the scope of this thesis.

Reasoning Service: based on given input, this service applies certain reasoning rules and generates dynamic links. For example, a travel booking system is a simple reasoning service, where the user can ask for a list available tickets for a particular time span. In 1990, Nanard brought into the hypertext field the concept of "semantic network" from Artificial Intelligence [143] into the hypertext field by creating a Conceptual Hypertext System [42], in which a hyperlink can be reasoned by using a domain model classification. The Conceptual Open Hypermedia Service (COHSE) project [42] later took this approach forward by providing ontological reasoning based on links of services to bridge the navigation gap between the Web and Linked Data, where the link services provided a mapping between concepts and the lexical labels on the web page. In addition, Rivlin [132] also designed a toolbox to facilitate structure-based navigation in hyperspace. The toolbox can answer users' questions such as 'where am I', 'how do I get to any destination' etc. and is another type of hypertext reasoning service.

Link Injection Service: this service advocate the idea of decoupling links and nodes and enabling the reuse of links on the Web by injecting links into Web resources where links were not originally available. Microcosm was one of the early hypertext systems to implement link injection. It introduced the concept of generic links - a link that may be followed from any occurrence of a particular object (e.g. a particular text string) with all links stored in

a linkbase. For instance, when there are three links in the linkbase about a concept, say "University of Southampton", it will inject all three links and a drop down list to any content containing the string of "University of Southampton". Magpie [56] is another system which supports link injection. The tool is designed to automatically associate an ontology-based semantic layer with Web resources. In other words, it injects ontology links into the Web contents. Magpie allows relevant services to be invoked as well as remotely triggered within a standard Web browser.

It is worth mentioning Distributed Link Services (DLS) [51]. This is a service that works in conjunction with existing Web resources to support an additional underlying link service. The system is developed based on the similar vision to Microcosm hypertext, which utilises a variety of link database processes to offer flexible hypertext functionality to a wide range of end-user applications. The system is composed of two parts: a link server which is accessed via the Web and the client interface that works in conjunction with a Web browser. The link server facilities of the DLS were implemented as CGI (Common Gateway Interface) scripts to runs on a different TCP/IP port number to an HTTP server. This server interacts with clients as if it were a normal Web server, clients access it using HTTP from the browser. The server does not store any returned documents. The system enables creation, traversal and editing of links which are stored in a number of link databases. The databases use style mark-up and record the source and destination attributes of the link, its type, creation time as well as a link description. Figure 2.5 is the user interface of the DLS, where a user requests a link from the link service. The client allows the user to select a predefined

context, or topic of interest and follow the link as it is displayed. Once the user has made a selection, the link server returns the results of the DLS request to the Web browser as it is shown in Figure 2.6.

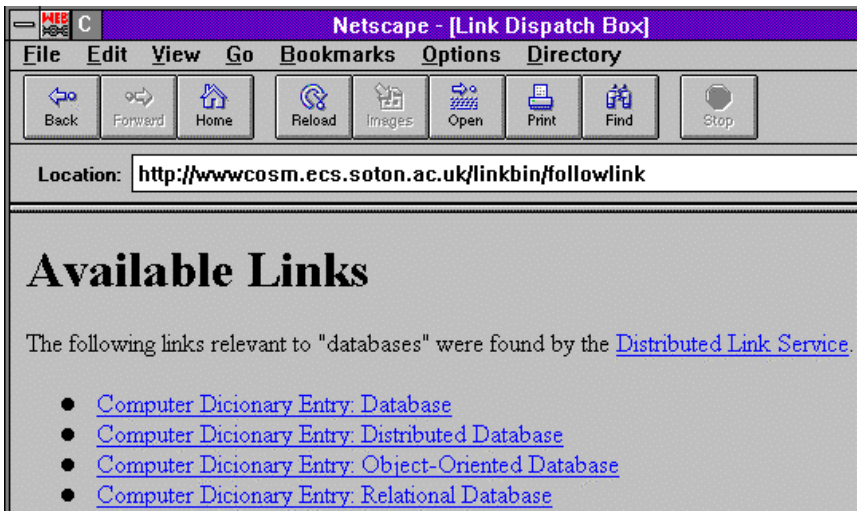


Figure 2.5: A user requests a link from the link service using the client interface [51]

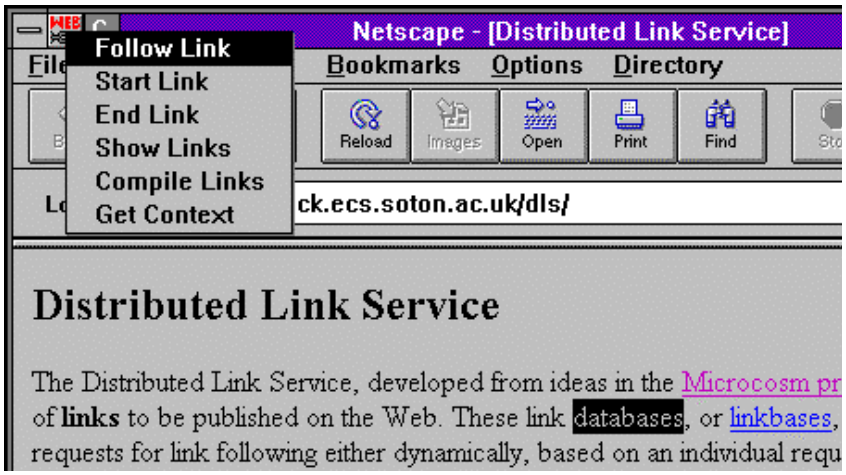


Figure 2.6: The link server responds with a page of available destinations [51]

The essence of DLS is to provide an interfaceless proxy link service which makes the link service transparent to its users by embedding it in the Web's document

transport system, compiling links into documents as they are delivered to the browser by a specially adapted Web proxy server. While the design decision was made for the practical benefit that no extra client software is needed on the user-side, the major disadvantage is that it limits the link can only be injected into the Web document (links are not reusable elsewhere in the operating system).

Figure 2.7 depicts the model of DLS, which extends the model originally described in [51]. A client accesses documents via the adapted Web proxy server described with the proxy accessing a local link base in order to resolve links. When the user makes queries interactively (e.g. following links or authoring a link), it is served by link server 1 which contains a local linkbase. The link server has two options: 1) It can pass the query on to another link server as is shown in Figure 2.7. The proxy server passes queries between link server 1 and 2. This is useful both for concurrency (obtaining links from multiple servers simultaneously) and for redundancy (working with multiple servers with the same linkbases); 2) Instead of sending the query towards the linkbase, the system can bring the linkbase towards the query. This is achieved by retrieving the linkbase data from an appropriate server, as shown for link server 2.

2.7 Conclusions

This Chapter studied the concept of hypertext from historical hypertext systems to different types of hyperlinks in different systems. It then discussed the World

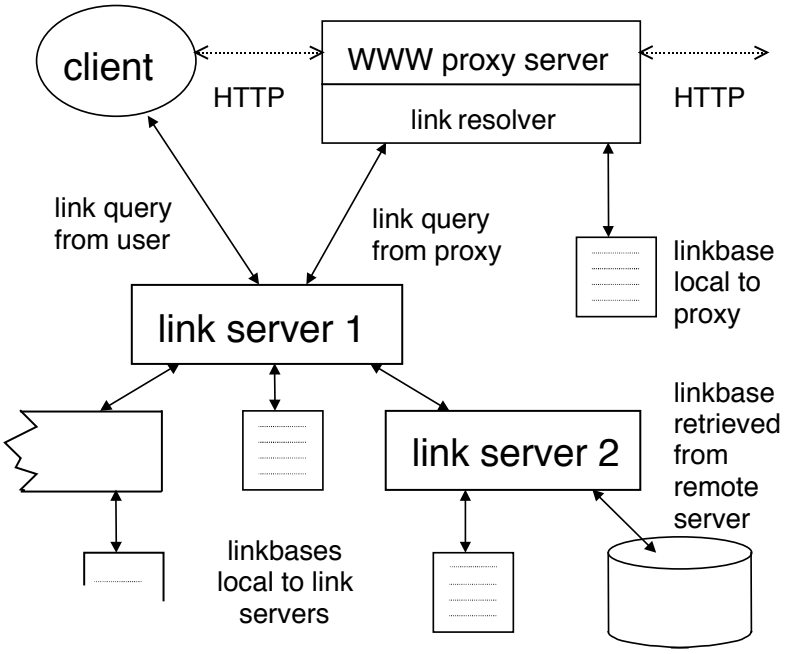


Figure 2.7: Distributed Link Service Network Model

Wide Web and the identifier URI in great detail. This was followed by a list of link services which were implemented to improve the navigability of the Web. All these services built on top of the Web enrich the Web hypertext ecosystem as a whole. In the next chapter, the nature of hyperlinks in the Semantic Web is addressed - an extension to be built on top of the Web of documents.

Chapter 3

The Semantic Web and URI

What is a hyperlink on the Semantic Web?

The Web was designed as an information space. On the Web, there is information such as novels and art, which can only be understood by humans and whose meaning is subjective; there is also information has clear meaning (e.g. the weather at a location), which can be processed and reused by computer programs for other pruposes. However, on today's Web, much data is not well structured in a form that can be easily processed by computer programs. Some data, even if it is structured, does not follow a standard (e.g. the weather data on BBC is different from that displayed on CNN). Therefore programmers have to write different

software to process them. Indeed, Web content is predominantly intended for human consumption. Certainly, one can write algorithms to process the contents of the Web, split them into parts, processing each words. However, when it comes to interpreting sentences and extracting useful information for users, the current algorithms and tools are still very limited. For example, an algorithm/tool is not capable of understanding that the string 'apple' is a fruit or the company easily without knowing the context. The major reason for this is that the Web contents are not machine-understandable. As already discussed, one solution to this is to use the content as it is represented today and to develop increasingly sophisticated techniques based on artificial intelligence and computational linguistics. However, at this juncture, in the author's opinion, such an approach still appears too ambitious for the multi-languaged and multi-cultured Web. An alternative approach is to represent Web contents in a form that is more machine processable. The Semantic Web follows this route. This chapter, first studies the Semantic Web by focusing on Linked Data. The nature of hyperlinks on the Semantic Web are then discussed and how they should be published.

3.1 The Semantic Web

In 1994, Berners-Lee presented the"future direction" of the Web, his initial vision of the Semantic Web at the WWW conference at CERN, in Geneva, Switzerland [10]. He described the Web as flat and devoid of meaning, but noted that the contents on the Web described both real objects (e.g. an human) and imaginary

concepts (e.g. a mythological god). It would be useful if a relationship can be established between these concepts. For example, if one document might describe a person, and another one might describe the organisation at which they work, then the relationship could be described by adding a link. Figure 3.1 is a diagram used by Berners-Lee to describe a semantic network layer of terms on the Web of documents. Berners-Lee explained his vision in Scientific American in 2001 [26], saying that "the Semantic Web is not a separate Web but an extension of the current one, in which information is given well-defined meaning, better enabling computers and people to work in cooperation."

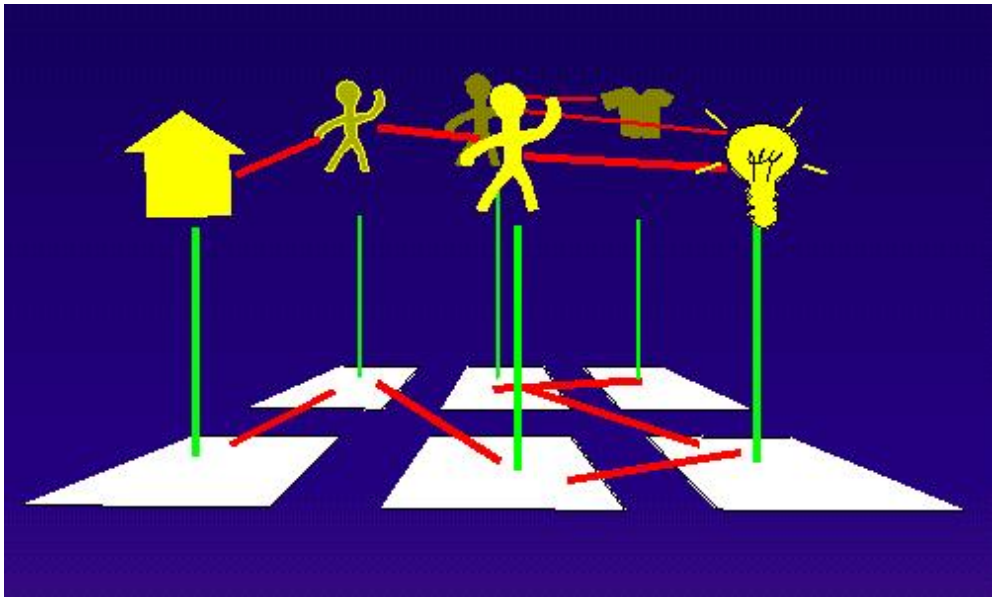


Figure 3.1: The Semantic Network Layer on the Web of documents [10]

The Web can be considered to be a global communication medium where people and machine meet. Although people can communicate freely on the Web, machines do not understand all the strings of text. It would be indeed useful to create a semantic layer to map the information of our reality. In other words, a language can be designed which supplements HTML with more structured infor-

mation. By doing so, better support to human's tasks can be provided by the machine and this also enable the machine to communicate (exchange information) with other machines. The major effort of the Semantic Web was to develop and standardize data structures, so that webmasters can augment their websites with more structured information. The Web is a technology innovation, so is the Semantic Web. The success of the Web is that Web consumers have adopted the HTML as way of publishing information, therefore the success of the Semantic Web is subject to adoption of its standards as well.

The technologies need for the realisation of the Semantic Web build upon work in the area of Artificial Intelligence (AI), and especially knowledge representation. In their 1963 paper of "some philosophical problems from the standpoint of artificial intelligence", John McCarthy and Patrick Hayes discussed the need for knowledge representation [104]. They considered that AI programs had been written to solve a class of problems that gave humans intellectual difficulties at the time, for example, playing chess or proving mathematical theorems. In the course of designing these programs, intellectual mechanisms whose generality is identified sometimes by introspection, by mathematical analysis or by experimentation, etc. An alternative approach is to start with the intellectual mechanisms (for example, memory, decision-making, learning) and make up problems that exercise these mechanisms. They call the latter approach a reasoning program, which involves both the epistemological and the heuristic parts of the artificial intelligence problem: the information in memory must be adequate to determine a strategy for achieving the goal and the reasoning program is capable of selecting a suitable strategy to draw a conclusion. The Knowledge Representation is the

very first step in the design of a Reasoning Program. A representation is called epistemologically adequate for a person or machine if it can be used practically to express the facts that one actually has about the aspect of the world [104]. The firstly attempted to formalise knowledge representation in first-order logic [105]. However, McDemott considered that, in practice, neither can all knowledge can be formalised since even given some fragment of formalised knowledge, the inferences are often trivial or irrelevant. Terry Winograd [154] pointed out that, in a procedural view of the intelligence, that the representation part is irrelevant if a program could successfully solve some task given some input and output, the symbol manipulation processes themselves are primary, and the rules of logic and mathematics are seen as an abstraction from a limited set of them. One prominent alternative in knowledge representation was semantic networks.

A semantic network is "a graphic notation for representing knowledge in patterns of interconnected nodes and arcs" [144]. The term 'semantic networks' was coined by Richard Richens [131] to describe a common knowledge-representation system for machine-translation systems. Semantic networks were used to represent the underlying structure of thesauri, such as Masterman's semantic lattices [103] and natural language systems such as Shapiro's Semantic Network Processing System [139]. However, some first-order logic researchers, like Hayes, pointed out that in semantic networks, only the author understood the exact meaning of the natural language, even when it was used as a formal language. Hayes [85] considered that, by providing formal semantics that defined 'meaning', first-order logic can enable knowledge representations to be transported across domains, and many other alternative knowledge representations could be re-expressed in first

order-logic. Therefore, ultimately, we would like to have a single knowledge representation system which is can be universally reused. This brought out another important concept 'ontology'. An ontology is an explicit and formal specification of a conceptualisation. Typically, an ontology consists of a finite list of terms and relationships between them. The terms denote important concepts of domain. For example, Figure 3.2 describes a movie ontology, in which a movie class has name, year, genre, actor and director etc.

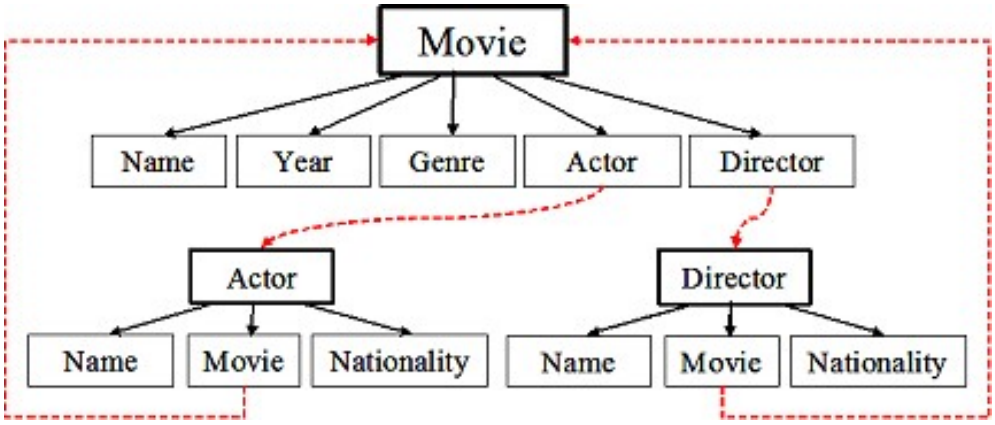


Figure 3.2: A movie ontology

The Cyc project [98] was to pursue the formalisation of all 'common-sense' knowledge in a single knowledge representation language, which is essentially engineering a common ontology [76]. The Cyc approach to structured ontology uses hyperlinkx structures, whereby the project took its direction by employing ideas from the hypertext field that "forget intelligence completely...constructing the world's largest hypertext system, with Cyc functioning as a radically improved counterpart for the Dewey decimal system. Such a system might facilitate what numerous projects are struggling to implement: reliable, content-based searching and indexing schemes for massive textual databases" [142].

In the context of the Web, ontologies provide a shared understanding of a domain which overcomes differences in terminology. In the same example of a movie ontology, one application may use the term 'film' to refer a movie. Another problem is that two applications may use the same term with different meaning. For example, the term 'film' may be referred as the physical material for exposure in a camera. Such differences can be overcome by mapping the particular terminology to a shared ontology. The ontology essentially enables interoperability between applications. In another words, the ontology acts as a shared 'knowledge and culture' base which enables machines to communicate with each other.

In a Scientific American, Berners-Lee describes the Semantic Web as for machine to communicate with machine. In his scenario, some way is needed to identify the information on the distributed Web, and its shared data structure and vocabulary, as well as the logic for processing retrieved information and drawing conclusions.

Berners-Lee also pictured the Semantic Web as being used by agents to perform intelligent tasks for humans. Agents are programs that work autonomously and proactively [5]. Figure 3.3 describes such a personal agent. An agent receives some tasks and preferences from its instructor and seeks information from Web sources. An agent can communicate with other agents and provide answers to its user. For the movie example, a user can ask the Semantic Web to retrieve a list of movies whoes genre is sci-fi and are both directed and written by Steven Spielberg.

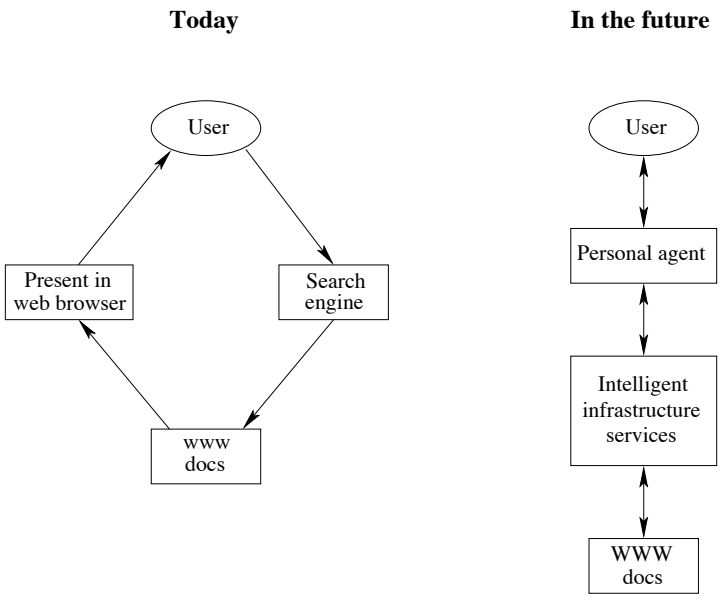


Figure 3.3: Intelligent personal agents [5]

The development of the Semantic Web proceeds in steps, each step building a layer on top of another. The pragmatic justification for this approach is that each step addresses a different issue and it is easier to achieve consensus on small steps. The W3C standardizes these consensuses for organisations and engineering to follow to avoid conflicts and achieve a common interoperability. Figure 3.4 is the latest version of the Semantic Web stack introduced in his keynote address to AAAI2006 [14]. Although the original version of the stack was introduced in a presentation in 2000 [13], the Semantic Web stack was never published in literature [68].

This thesis mainly focuses on the standards located at the bottom part of the Semantic Web stack highlighted in red in Figure 3.4. The URI and Unicode are technologies which already present in the WWW. Therefore, the Web extension

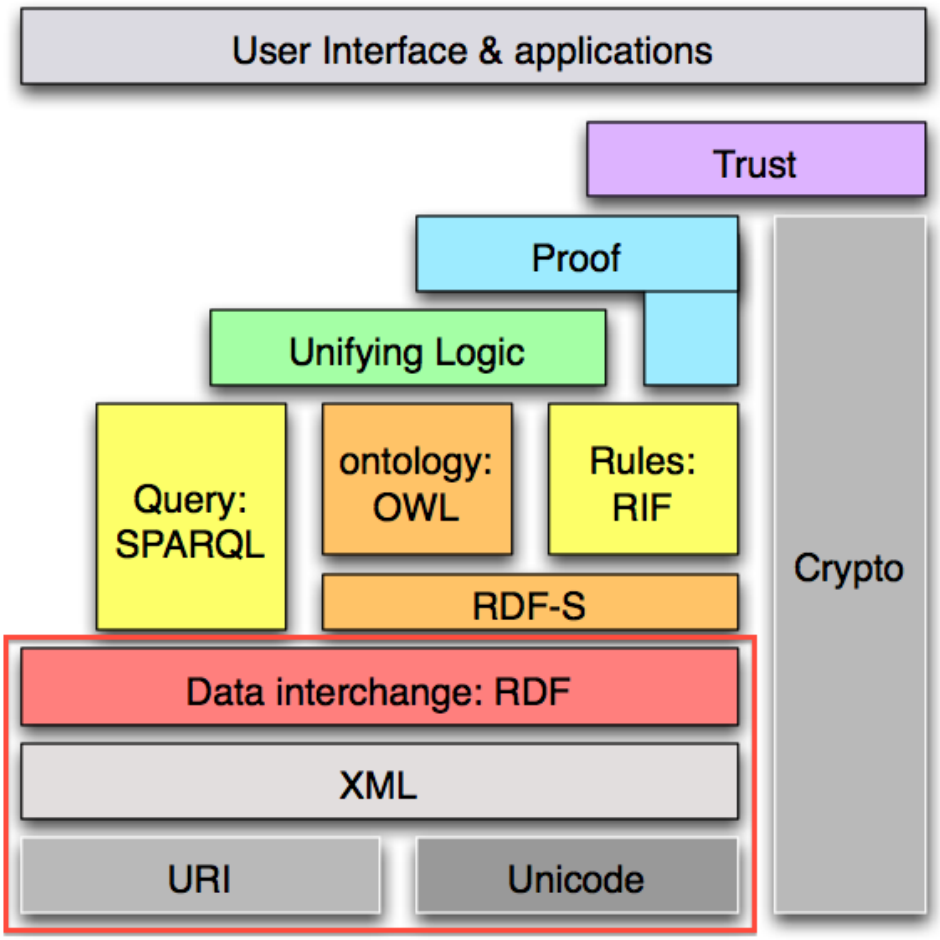


Figure 3.4: Semantic Web Stack

Semantic Web's fundamental layer is URI and Unicode. HTML is not sufficient to describe the Semantic Web data, therefore, a superset of HTML - XML can be used as the serialisation syntax for the Semantic Web. However, in the XML, data are not identified, and there is no mechanism to enable reuse of data. Namespace was added to XML to increase its modularisation and reusability. The Resource Description Framework(RDF) model that extends the XML is designed specifically for the Semantic Web. It uses the semantic network structure to represent

resources online. RDF was inspired by the work on Meta-Content Framework (MCF) by Ramanathan V. Guha who was previously worked as chief investigator on the Cyc project mentioned above [79]. The first use of RDF was in a light-weight knowledge representation system of subject-verb-object form as metadata for the 'Semantic Search' information extraction system [78].

In RDF, the fundamental unit of knowledge representation is the triple, a directed edge that represents a binary relationship between two nodes. The components of the triple are referred to as the subject, the predicate and the object. The nodes in the RDF graph denote resources, which are things in the domain of discourse. Nodes could be URIs, literals or blank nodes. URIs are used to denotes things with identity, such as webpages, or real world objects. Literals are values such as character strings; a (plain) literal is considered to denote itself. Blank nodes are used to indicate the existence of a thing, without needing to state the name of that thing. For example, a triple states a webpage (a subject) was created (a predicate) by John Smith (an object) can be expressed as below:

⟨http://www.example.org/index.html⟩⟨http://purl.org/dc/terms/creator⟩⟨http://www.example.org/staffid/85740⟩

RDF is designed as a 'machine readable' language, and the following section discusses how agents can achieve a universal meaning of a URI after they discover its RDF.

3.2 Issues on achieving an universal meaning of a URI in knowledge representations

"A rough statement of causal theory of reference might be the following: an initial 'baptism' takes place. Here the object may be named by ostension, or the reference of the name may be fixed by a description. When the name is 'passed from link to link', the receiver of the name must, I think, intend when he learns it to use it with the same reference as the man from whom he heard it." Kripke [94]

For different agents to exchange knowledge representations on the Semantic Web, they need to share the meaning of a URI - **what a URI refers to**. This is the core to enable interoperability between applications. On the hypertext Web, URIs identify documents. It is relatively straightforward. As discussed in section 2.5.1, the URI resolution connect to the presentation of the resource itself that is, when a URI is dereferenced, the document which it identifies is obtained. On the Semantic Web, when we use URI to identify a real world object, surely this URI cannot be dereferenced to obtain a real world object. The URI can only be used as a reference (a name) and to provide relevant information about the object. In this context, URI is used as a reference instead of access as a document locator. Therefore, how this identity is used on the Web is absolutely crucial for the Semantic Web to provide machine readable information.

What URIs exactly identify has been an ongoing discussion on the W3C Technical Architecture Group mailing list, known as httpRange-14 and subsequent TAG

issues [63]. There are generally two positions to this problem, as Halpin [82] summarieses: the direct reference position and logicist position.

The direct reference position: the meaning of a URI is whatever was intended by the owner. In this case, it is like a webmaster giving a name (URI) to a thing, and from then on, others make use the same URI on the Web to refer to whatever he intended to reference in the first place. Figure 3.5 shows a such scenario, where the URI owner named the Eiffel Tower as `http://www.example.org/EiffelTower`. This view is similar to the Kripke's causal theory of proper names [94] as we quoted at the beginning of this section.

The logicist position: the meaning of a URI is given by whatever things satisfy the model(s) given by the formal semantics of the Semantic Web. In this context, the referent of a URI is ambiguous, as there can be many different things satisfying whatever model is given by the interpretation of sentences using the URI. Some believe a URI has no meaning, but only in the context of its use in other triples, while others hold that one should be able to access logical descriptions from the URI itself. Figure 3.5 shows a scenario, where a URI `http://www.example.org/EiffelTower` is mentioned in a RDF triples, after formally being interpreted by a model, it concludes that the URI identifies the Eiffel Tower.

Direct reference views are held by the Web architects such as Berners-Lee. Berners-Lee consider that this rule should not only apply on the Semantic Web, but also on the Web as a whole. As discussed in the section2.5.1, this view is consistent with the architecture of the Web. The problem of the logicist position is that

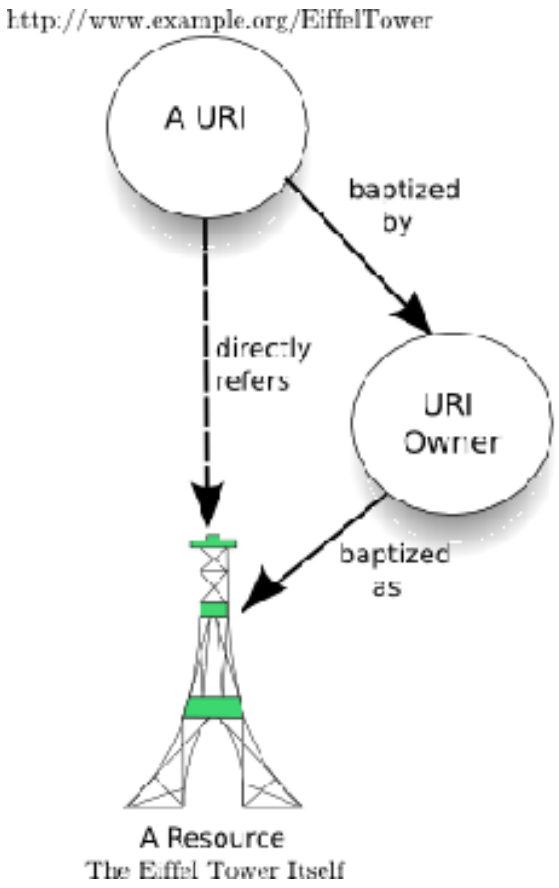


Figure 3.5: Direct reference by URIs [82]

it is only valid if the Semantic Web is considered as a new system instead of extension of the Web. As on the existing Web, URIs are already used to identify documents (information resources) and access to documents. Regarding the debate between these two positions, in the TAG mailing list, Berners-Lee emphases that "a URI identify one thing...the URI ownership system makes statements by owners authoritative weight, despite what other documents may say...the use of a URI in RDF implies a commitment to its ontology, and if there is doubt as to what ontology that is, the Web may be used to resolve it...the Web is not the final arbiter of meaning, because URI ownership is primary, and the lookup system of

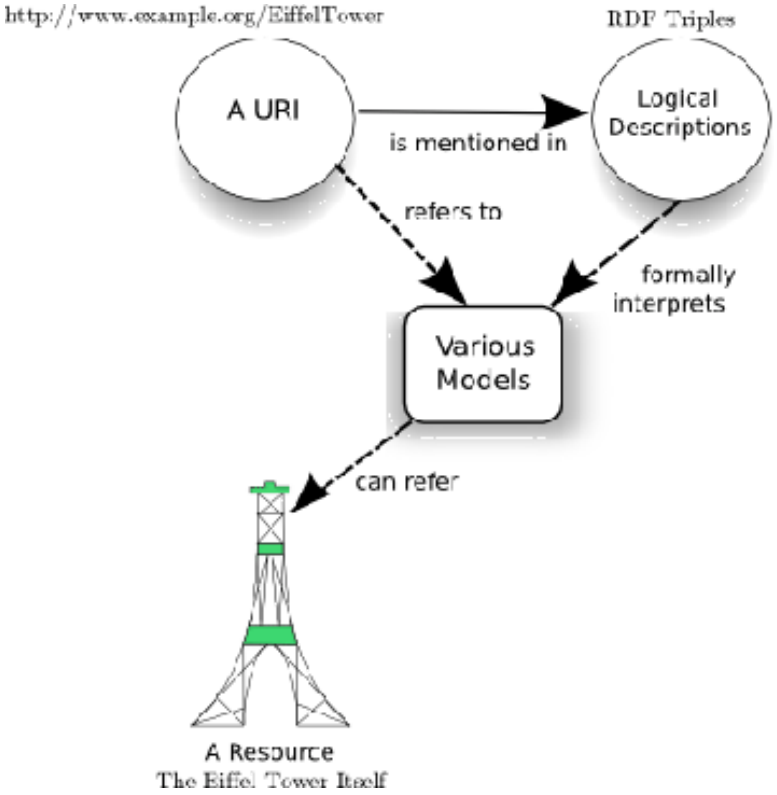


Figure 3.6: The logicist approach of reference of URIs [82]

HTTP is though important secondary" [1].

Hayes and Halpin [85] discussed the possibility of extending the logicist position of the Web, however there no clear solution has been published which can avoid inconsistency of the existing architecture of the Web (where URIs identify web documents). Clearly, there are obvious advantages to building the Semantic Web on top of the Web, instead of creating a separate new system. In this way, the technology infrastructure available can be re-used as well as billions of information documents on the Web. Other than this, in my opinion, since the URIs and RDFs are published by a crowd of people, direct reference seems to be intuitive and easy

[1] http://lists.w3.org/Archives/Public/www-tag/2003Jul/0022.html

to understand. In contrast, the logicist approach can be overly restrictive to be applied easily in practice. The idea of "URIs can refer to many different things as long as they satisfy the model" can introduce more complications for agents that consume the data. Based on these understandings, the direct reference position will be used as the approach in this thesis.

One of the problems caused by the early adopter of the Semantic Web is that many research project neglected to host accessible links of the data [1]. One of the reasons is that many ontologists follows the logicist position therefore there was no clear incentive to publish RDF on the Web. Some recognised that the purely logicist view of Semantic Web of ontologies is a failure, the Semantic Web is taking off under the new name "Linked Data" [17].

3.3 Linked Data

"Data is a precious thing and will last longer than the systems themselves." [2]

Tim Berners-Lee

"Linked Data" [17] arose as a term referring to a set of best practices for publishing the connecting structured data on the Web. It is an idea which uses the technology available in the infrastructure of the World Wide Web: URIs and the HTTP and is supplemented by RDF for publishing and retrieving data on the

[1] http://www.w3.org/DesignIssues/TermResource.html
[2] http://www.bcs.org/content/conWebDoc/3337

Web.

Berners-Lee [15] outlined the idea of Linked Data as four principles that should govern the publication of data on the Web:

1. Use URIs as names for things

2. Use HTTP URIs so that people can look up those names

3. When someone looks up a URI, provide useful information, using the standards (RDF, SPARQL [126])

4. Include links to other URIs, so that they can discover more things.

The first rule defines a fundamental change from the Web to the Semantic Web. The second rule means that the URI needs to be published online, so that people can look it up via HTTP. The third rule means standards should be used by publisher to provide relevant information about the URIs. The fourth rule goes back to the hypertext idea, that a network of 'things' should be created, so that more resources can be discovered.

What are the 'things' Berners-Lee mentions here and how can URIs be published which identify real world objects on the Web which is used to publish documents? The following section discusses these.

3.4 URI resolution: a solution to an universal meaning of a URI on the Web

In order to understand how to publish URIs identifying real world objects, a few terms first need to be defined to discuss this subject more clearly. Following the W3C recommendation *Architecture of the World Wide Web* and the discussion in section 2.5.1 Resources and Representation of the Web, resource, information resource and non-information resource are defined as below:

Definition 1. ***Resource****: anything can be identified by a URI.*

Definition 2. ***Information resource****: the essential characteristic of the resource can be conveyed in a message. A web document or an image is an example of an information resource.*

Definition 3. ***Non-information resource****: anything else which is not an information resource is a non-information resource. A* ***real world object*** *such as a person is a non-information resource.*

Definition 4. *The URI which identifies an information resource is named* ***a document URI*** *and the URI which identifies a non-information resource is named* ***a real world object URI*** *for the purpose of distinguishing the nature of these URIs.*

As discussed in section 2.5 Figure 2.1, when dereferencing a URI via HTTP on the Web, a web presentation of the resource is returned (a document). On the Semantic Web, when dereferencing a real world object URI, a RDF document description about it will be retrieved. For instance, we can give a URI to the Eiffel Tower in Paris, and when resolving it, the Eiffel Tower itself will not be retrieved, but a set of RDF triples describing it. However, in this scenario, to be consistent with the existing Web Architecture, the document description in turn requires another URI to reside on the Web. This means in oder to publish a URI which identifies a real world object on the Web, a URI is also needed to publish its Web document description. Figure 3.7 shows this scenario.

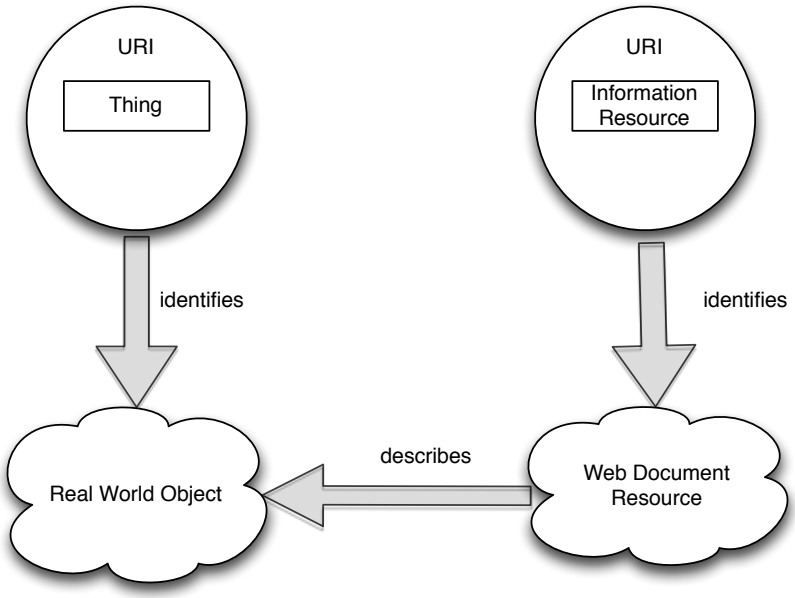


Figure 3.7: URIs on the Semantic Web

Certainly, this problem can be solved in various ways. Many approaches attempted to solve this problem over the years. First to be studied is Berners-Lee's original solution to the above problem.

3.5 Linked Data Publishing: a technical solution to a URI refers to a real world object

W3C proposed two solutions to the above problem over the time in the note of Cool URIs for the Semantic Web [138] known as hash URIs and 303 URIs. The hash URI is the original Berners-Lee's solution to publish URI describe things on the Web [1].

3.5.1 Berners-Lee's original solution: Hash URIs publishing approach

Hash URIs contain a fragment which separates URIs into two parts using a "#" symbol. When dereferencing a hash URI in a browser, the HTTP protocol requires the fragment part to be stripped off before the request; this process is referred to as "unhash". Figure 3.8 shows such a publishing mechanism. In this approach, a URI with a hash fragment (`http://www.w3.org/People/Berners-Lee/card#i`) identifies a person (real world object) and the URI without the hash fragment (`http://www.w3.org/People/Berners-Lee/card`) identifies the RDF document resource which describes him. This is a new way of using hash fragment, as it is traditionally used to identify fragments of a hyper-

[1] http://www.w3.org/DesignIssues/Fragment.html

text document. It is now also used to identify a 'secondary resource' which is defined as "some portion or subset of the primary resource, some view on presentation of primary resource, or some other resource defined or described by those representations". [24] In the following thesis, the term **hash URI** will be used to refer to new usage of URI.

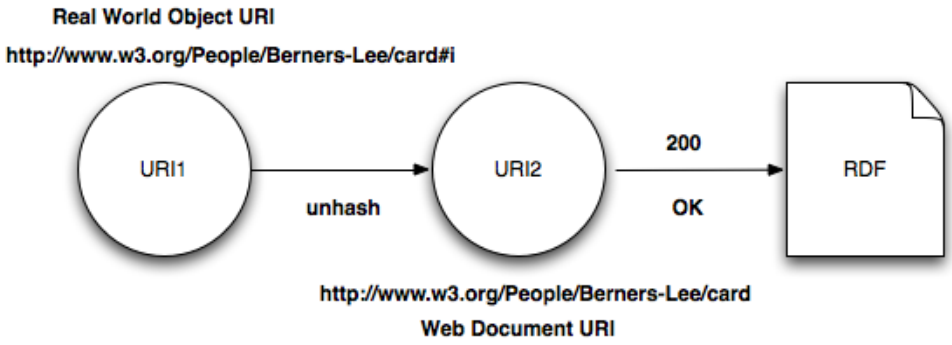


Figure 3.8: Hash URIs Publishing Approach

In Berners-Lee's original view, the hash fragment was designed to enable the extensibility of the Web [12]. The idea is that the hash fragment part can be used as a different function that enables the creation of new applications (such as the Semantic Web) on top of the Web, without the need to modify the core architecture components of the Web (the URI and the HTTP). As shown in Figure 3.9, an application can parse a URI, access its web presentation via the HTTP restful service. Once it obtained the Web presentation, it can use the hash fragment part as a function to reference a part of the web presentation object as globally available variables. In this way, there will be no need to change the access part - displayed in blue - in any other application requirements.

Early in the Semantic Web era, most RDF users recognised the value of using

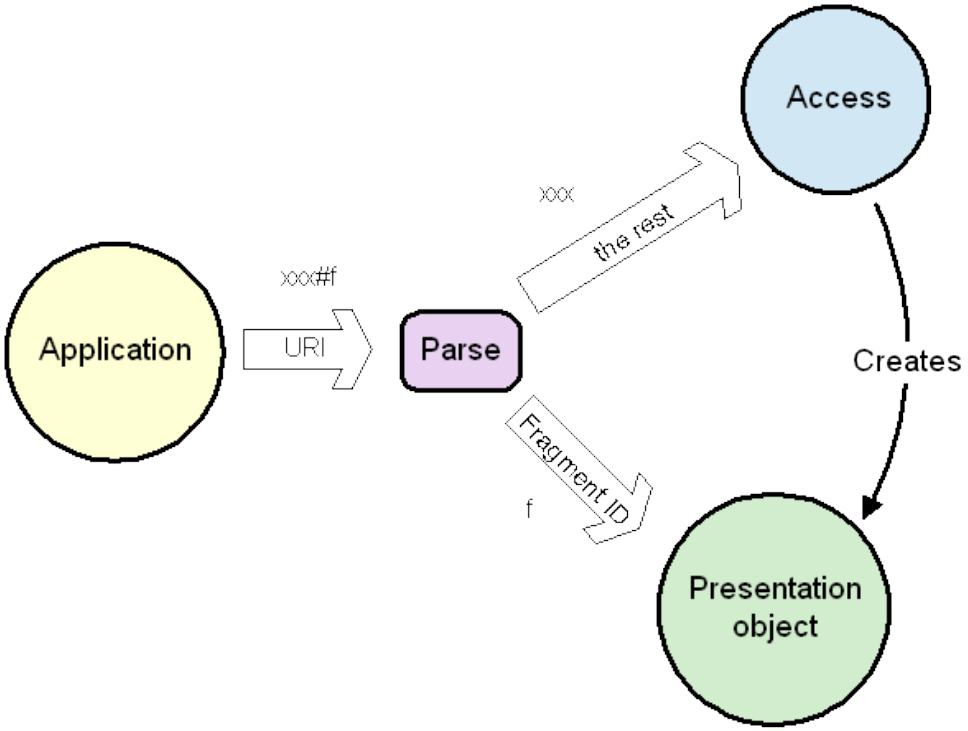


Figure 3.9: Hash Fragment and Presentation Object [12]

HTTP URI in the RDF, however there were problems with adopting hash URIs and publishing RDF online to make URIs deferenceable [16]. Some users simply neglected the importance of the hash URIs, and some building large systems considered the hash URIs as identifier were not practical for very large documents, as it is difficult to divide up the information into middle-sized chunks. Researchers such as Hayes also demanded the right to use a regular slash URI to identify a real world object. These resulted in a large number of RDFs being built with URIs which are not dereferenceable or have identify ambiguity problems (as slash URIs on the Web were only used to identify documents). In response to this, W3C enabled the use of regular slash URIs to identify real world objects and introduced another publishing approach which used HTTP 303 redirection.

Enabling URIs to identify real world objects in fact changed the existing Web architecture, therefore, the HTTP protocols are need to be modified or interpreted differently to cope with the change. In the following, the 303 URIs publishing approach is first grasped, then the unaddressed publishing issues are discussed in the later sections.

3.5.2 A social adoption fix: 303 URIs publishing approach

The 303 URIs solution is to uses the HTTP Status Code 303 See Other to indicate that the requested resource is not a regular Web document, and to direct the request to a document URI which contains a description of the requested object. Figure 3.10 shows such a publishing mechanism. The URI (http://dbpedia.org/resource/Tim_Berners-Lee) identifies a famous researcher (real world object) and the URI (http://dbpedia.org/data/Tim_Berners-Lee.xml) identifies a RDF document located on the DBpedia server.

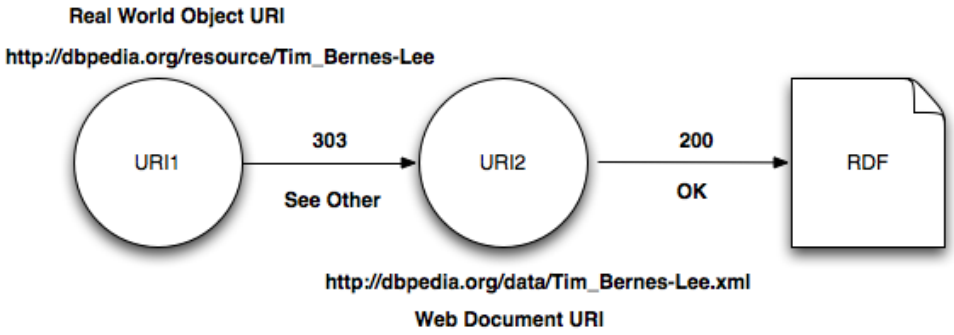


Figure 3.10: HTTP 303 Redirect URIs Publishing Approach

The 303 is essentially an adoption fix. This also shows that popular adoption

drives system design decisions, even though the original design decision may considered more elegant. Over the years, there were many publishing approaches tried to address the URI identity problem. The following discusses why these approaches were not applicable or not considered good solutions.

3.6 Other publishing approaches and why they donot work

There are many approaches that have attempted to address the URI identity problem. They tend to fall into two categories 1) by creating a new naming scheme, and 2) modify the HTTP protocol. In a nutshell, most of approaches do not work because they only partially address the problem or they require to change/modify the existing Web architecture. The following section examine a few of these approaches.

3.6.1 Why a new naming scheme approach is not desirable

URN (Uniform Resource Name) is a scheme created for names that can refer to location-independent resources, such as things outside of the causal reach of the Internet. The syntax of URNs is

<URN> ::=“urn:” <NID> “:” <NSS>

<NID> is the namespace identifier, which determines the syntactic interpretation of <NSS>, the namespace-specific string. For example, urn:isbn:0451450523 corresponds to “the 1968 book The Last Unicorn which is identified by its book number.” A client may find a “resolver” for a URN. A resolver is a database that can provide information about the resource identified by a URN, such as the resource’s location, a bibliographic description, or even the resource itself. A typical URN resolver will translate a URN name into a URL and obtain its description.

One major advantage claimed for the URNs design is that it is persistent and location independent. Mealling and Daniel [107] explains that the URN can “be globally unique forever, and may well be used as a reference to a resource well beyond the life of the resource it identifies or of any naming authority involved in the assignment of its name.” Thompson [147] has summerised a W3C notes to explain that URI is sufficient as an identifier for many requirements of systems and there is no need for new approaches to naming information resources. The main argument is that the URI is standardised, supports persistence, is protocol and location independent, where the URN is essentially a subset of URIs. They also observed that the HTTP protocol is rather generic and can be extended and integrated, while offering substantial benefits, in terms of existing Web architecture, scalability and, if required, security, at very low cost. These reasons also applies to other naming schemes such as Magnet [1], the info: URI [150] scheme,

[1] http://magnet-uri.sourceforge.net/

Extensible Resource Identifier (XRI) [129], etc.

3.6.2 Why extending the HTTP approach is not desirable

There are also approaches which extend the HTTP, for example, in the URIQA project [145], an extra HTTP method is introduced called MGET. It returns a concise bounded description of the resource denoted by the request URI. Although this enables to publish a URI describes the real world object with a description, it only partially solves the problem. As on the Web, if the metadata of an information resource are published (for example, RDF metadata of bbc.co.uk), it is also a description of the resource. In this scenario, this metadata can simply be published and given a URI, it would function as a regular document on the Web. The URI can be dereferenced by using the HTTP GET method to obtain the description. For this particular case, MGET overlaps with the GET method. Other than this, to add an extra method in HTTP specification will require many elements of the existing infrastructure to be updated.

In contrast, the hash URI and 303 redirection approaches have less impact on changing of the existing infrastructure, and therefore are considered better solutions. However, as mentioned previously, the 303 redirection is a fixed solution based on adoption, therefore it also require the some change or reinterpretation of the existing HTTP specification. The following section we discusses some

unaddressed issues.

3.7 Unaddressed Linked Data publishing issues

Although the above mentioned hash URI and 303 redirection solves the URI identity issues, it introduces other issues into the existing HTTP protocol. A function of a protocol is essentially a convention for transmitting information between agents. The HTTP of the Web is only used to deliver document, but it hows transmitting information about URI which refers to the real world objects. We certainly need to ensure that what URI refers to are consistent in this transmitting information process.

There are many types of HTTP redirections on the Web (for example the temporary and permanent redirection between URIs were often used on the Web), but how these URIs interact witha 303 redirect has not been discussed in detail, nor how to dereference these URIs. Booth [35] briefly described an algorithm for locating authoritative descriptive information - known as the Follow Your Nose algorithm. However, it only covered the simplest scenario of the HTTP dereferencing process which did not include all the possibilities of URI dereferencing processes, such as when there are chains of redirection in the middle of the dereferencing processes. Lewis [99] from W3C TAG also tried to draft Dereferencing HTTP URIs guidance in 2007, however, this work remains incomplete to date. There is also discussion that the 303 URIs Publishing mechanism has

performance problems as it requires redirections. An alternative approach of 200 status response but with a header "Document" link to the document presentation has been suggested [1]. However, this approach means that many existing agents (e.g. broswers) have to be updated to support such a change. To date 29 April 2013, no official decision has been made on this topic.

Other than this, some research experiments concluded that the Linked Data publishing mechanism is not straight forward and can be error-prone [89]. A Linked Data validator named Vapour [27] was implemented, which can be used to facilitate Linked Data publishing. However, the validator focuses on the perspective of validating the content negotiation process when dereferencing a URI. Although it also checks whether a URI is a non-information resource or not, it is hard coded to checks against the Best Practice Recipes for publishing RDF guidance. It would not function properly when dealing with other cases or when it encounters chains of the direction (e.g. when validating `http://www.geonames.org/ontology`, it produces an error). In other words, it is incapable of distinguishing a document URI from a real world object URI, which we consider this as a crucial missing component for facilitating the publication of URIs correctly.

[1] http://www.w3.org/wiki/HTML/ChangeProposal25

3.8 Conclusions

This chapter have studied the Semantic Web and its technology. It discussed fundamental change to the hyperlink - the URI, from the Web to Linked Data - links are only used as an address of webpages, but also can identify data (the real world object). This makes Linked Data a different hypertext system from the Web. Linked Data publishing mechanism was then studied to address the URI identity problem. W3C proposed two approaches which use either hash URI or 303 HTTP redirection. Quite a few issues of Linked Data publishing are not addressed, especially with regard to URI resolution. The next chapter studies the dereferencing of the Semantic Web URIs.

Chapter 4

URI Resolution on the Semantic Web

How can an agent understand what a URI refers to by dereferencing it?

A 'paramount requirement' of the Semantic Web is that an agent can follow a URI to understand what a URI refers to. As previously discussed, URIs traditionally only accessed the Web document resource but can now also identify real world objects. When dereferencing a real world object URI, a RDF description of the requested object will be retrieved. Using a URI to identify a real world object, essentially changes the existing Web architecture, since it was designed

for delivering web documents to the clients. In order to publish the real world object URIs, W3C proposed two technical solutions without the need to modify the existing Web architecture, called hash URI and 303 redirection, to make a distinction between a real world object (non-information resource) URI and a web document (information resource) URIs. This has indeed changed the Web architecture and requires its HTTP protocol to be reinterpreted. So far, not much research has been done in this area. And we discussed a few unaddressed issues in Chapter 3 section 3.7.

A protocol is typically a convention for transmitting information between agents. The very fundamental requirement of the HTTP is that in this transmitting information process, the identity of the URIs must be consistent throughout. As HTTP is simply a client-server protocol, this problem can also be examined from a client side point of view. On the traditional Web, a URI in a browser address bar simply identifies the web page; in contrast, on the Web of Data, it requires agents to track the requested URI between redirections or hash URI to understand what a URI refers to. When dereferencing URIs, will there be a problem with identity inconsistencies that make the Semantic Web architecture fail to achieve its fundamental requirement? In order to study this process, the client-server HTTP conversation messages need to be looked into, and how URIs interact with HTTP. Ultimately, we aim to define a URI dereference algorithm that can be used by an agent to understand what the URI refers to.

This Chapter will first looks into the semantics of HTTP. It then presents a generic dereferencing algorithm, which is essentially for an agent to understand

what a URI refers to. This algorithm is very basics for building a Linked Data agents such as a browser. Based on this algorithm, a Linked Data URI validator has been implemented, which visualises the dereferencing process and facilitates the publication of Linked Data URIs.

4.1 Dereferencing URI on the Web and Semantic Web: the base cases

Chapter 2 section 2.5.1 discussed the architecture of the Web, the resource and its representation. Figure 4.1 shows the base case of dereferencing a URI on the Web. In this example, the URI identifies an information resource - Berners-Lee's professional homepage. When dereferencing this URI via HTTP, the server returned a 200OK status code and HTML document which is a representation of this information resource. This is intuitive, as it is similar to the case in the real world, when you call someone's name and the person comes to you physically. This dereferencing process is straightforwards, as there is no extra stage in it. Although there may be some redirections within this process, however there is no need to discuss them, as on the traditional Web, everything is information resource (document), therefore, unless dereferencing failed, a representation of the resource that was requested is always obtained.

On the Semantic Web, it becomes more complicated. When dereferencing a URI

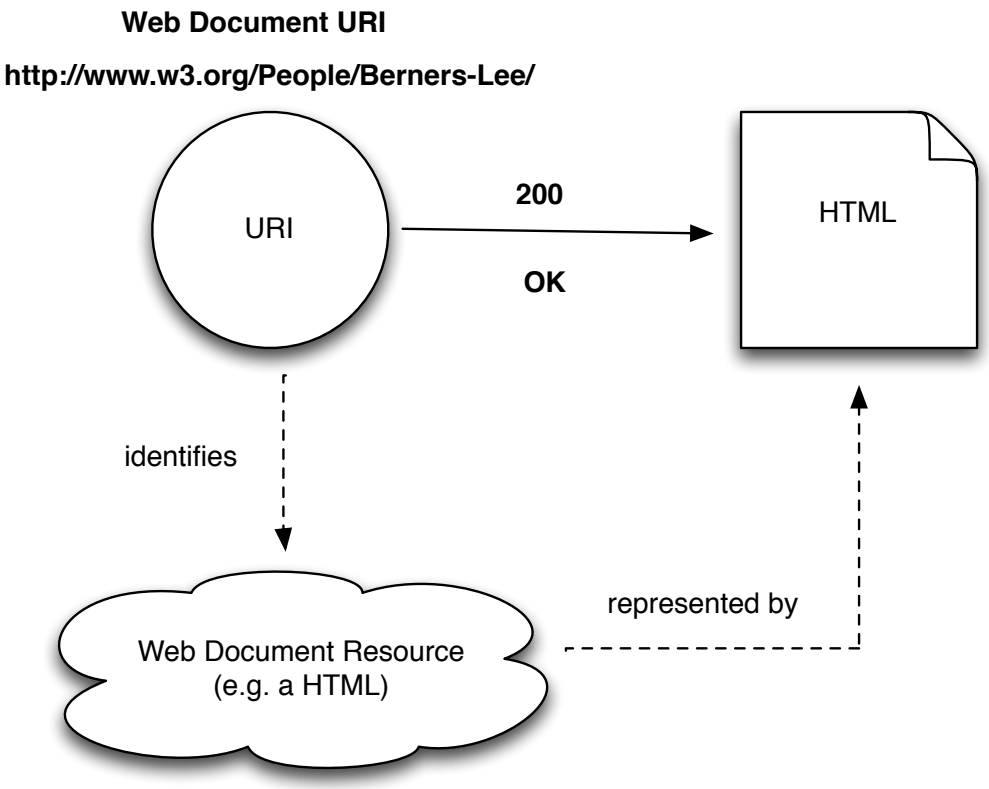


Figure 4.1: Dereferencing a URI on the Web: The base case

identifying a real world object, its representation cannot be obtained directly, instead obtaining an information resource (a RDF document) which describes it. This has actually already changed the existing architecture by adding the case that, when dereferencing a URI, a representation of a resource is not always obtained, but instead, it can be described by another information resource such as a RDF document.

In the Chapter 3 section 3.5.1, we have showed how hash URIs published. Figure 4.2 further explains what each URIs identifies. In this case, dereferencing a URI1 which identifies a person, after the 'unhash' process, the client dereferencing the

URI2 (another URI without hash fragment), which identifies a RDF document that describes the URI1.

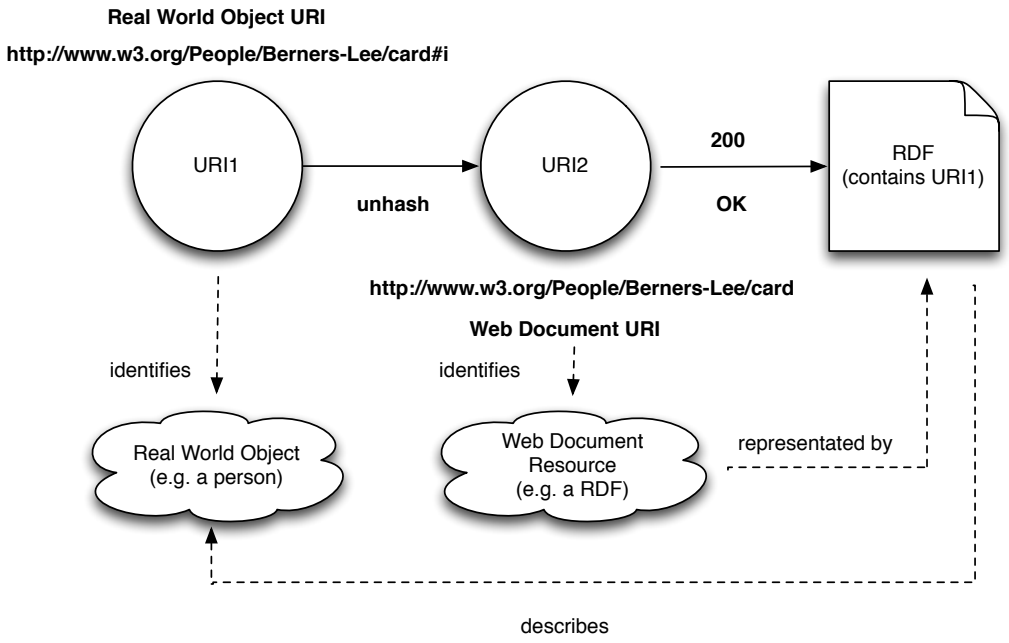


Figure 4.2: Dereferencing a Hash URI on the Semantic Web

Similarly, the 303 URIs publishing approach was discussed in Chapter 3 section 3.5.2. Figure 4.3 shows what each URI identifies in this scenario, which follows exactly the same pattern as the hash URI previously described.

After the dereferencing process, the client will obtain a RDF and the client will also need to know which URI to look for in the RDF descriptions. In the successful case, RDF should describe the requested URI, otherwise, it failed to retrieve its description. In that case, nothing is done about the requested URI whether it is a web document or a real world object.

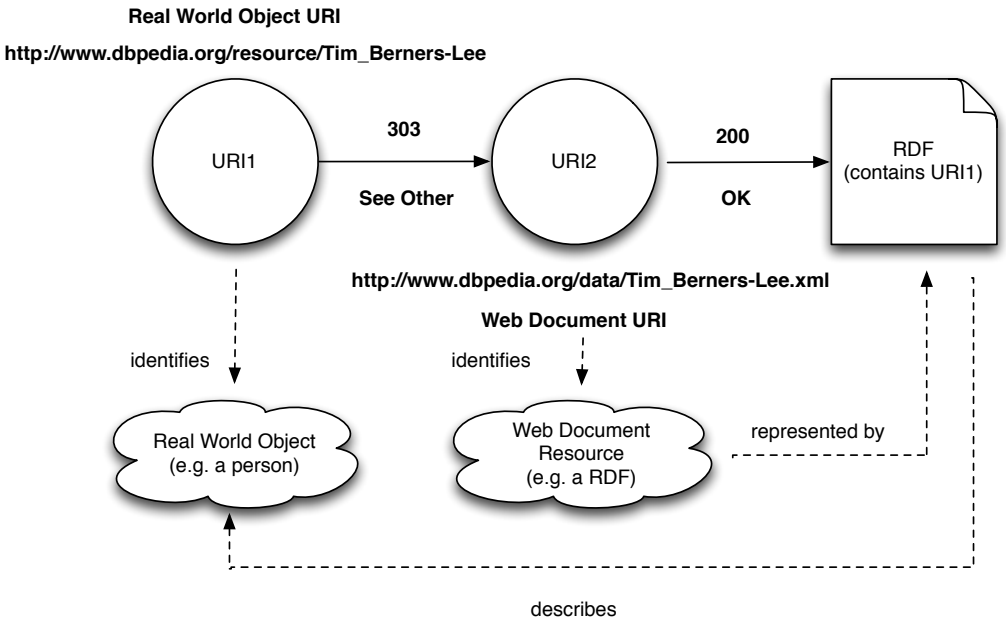


Figure 4.3: Dereferencing a 303 URI on the Semantic Web

In order to satisfy the above scenario, the client is required to track the requested URI and its redirected URI. This introduces more complications in implementing the Semantic Web agent.

The base case seems straightforward; as long as these cases can be handled and the requested URI kept track of, then the URIs should be able to be dereferenced. Booth [35] briefly described an algorithm for locating authoritative descriptive information, known as the Follow Your Nose algorithm simply track URIs in this case. However, on the traditional Web, there are already more HTTP redirection cases for the web document URIs. And these will make the above scenario much more complicated. Both hash URI and 303 URI approaches are essentially implementation of 'redirections' between real world object URI and web document URI. There are already many HTTP redirections in use on the Web, and

redirection between real world object URIs have not been discussed.

To build the Semantic Web on top of the Web will require the dereferencing URI algorithm not only to satisfy the new requirements, but be compatible with the existing deployed Web. Therefore, in order to design a URI dereferencing algorithm which can be used to implement generic agents (e.g. a browser) to customise the Semantic Web, the existing HTTP redirections and its semantics need to be studied.

4.2 Motivation for studying the HTTP status code - HTTP Semantics

When a client interacts with a server, HTTP returns a response. Chapter 2 section 6.1 discussed the technical details of HTTP. The first line of the HTTP response is called the status line, which includes a numeric status code (e.g. 404) and a textual reason phrase (e.g. Not Found). These are very important messages for a protocol, as they indicate if a request was successful or resulted errors.

The Web tolerates a vast amount of diversity from millions of human inputs. The HTTP status codes are designated not only to be a technical protocol to serve the Webpages, but also to handles many important problems we encountered in daily life. For example, what should be done with a website, when the server shuts

down or when the domain no longer valid, and so on. HTTP designated some status codes for web users to implement on their HTTP servers to handle these problems. For example, the HTTP permanent redirect on the Web means that the requested document has been assigned a new URI and future reference to the document should use the newly assigned URI. It will become more complicated when a permanent redirect of a URI, which refers to a real world object, to a newly assigned URI, as the URI should at least have the same nature as the requested URI which also refers to a real world object. Although this issue is important, to date, the W3C has not issued any recommendation of the HTTP semantics for the Semantic Web.

The HTTP status code on the Web are first examined. The following list summaries a list of HTTP status codes:

1xx Informational indicates request received, continuing process. It is a provisional response, consisting only of the status-line and optional headers, and is terminated by an empty line.

2xx Success indicates the action requested by the client was successfully received, understood, and accepted.

3xx Redirection indicates that further action needs to be taken by the user agent in order to fulfil the request.

4xx Client Error is intended for cases in which the clients seems to have erred.

5xx Server Error indicates that the server is incapable of performing the request or the server is aware that it has erred.

The 1xx, 4xx and 5xx codes are straightforward as it only provides information or error messages. 200OK is the most frequently used codes in the 2xx class. It simply indicate the request has succeeded. If the request is a HTTP GET method and server returned 200OK, it means the agent has successfully obtained the representation of the resource. The following mainly studies the 3xx code, as it involves the redirection between resource identifiers. The redirection behave on the Web will be first examined, and then its impact on the Semantic Web.

4.2.1 HTTP redirect for the Web

As mentioned the 3xx code requires an agent to take further actions to fulfil the request. Previously discussion showed how the 303 See Other code could be used on the Web to publish the real world object URIs. Other frequently used status codes fall into two categories: *permanent redirect* and *temporary redirect*.

Note that the purpose of this study is to examine some of the status codes and identify potential issues when dealing with real world object URIs. Ultimately, the aim is to identifies generic approaches that can be extended or reused to comprehensively to handle with all HTTP status codes.

Permanent Redirect: The HTTP Status Code 301 Permanent Redirect means that the requested resource has been assigned a new permanent URI and any future references to this resource should use returned URIs. For example, a person changed their homepage host due to a change of job, and therefore permanently redirected the homepage URI to a new location. A URI shortening service such as Bitly [1] frequently used on Twitter [2], implements a such scenario, basically a shortened version of the URI Permanent Redirect to a longer version of the URI. Figure 4.4 shows an example of 301redirection, where the requested URI is permanently redirected to the BBC news website.

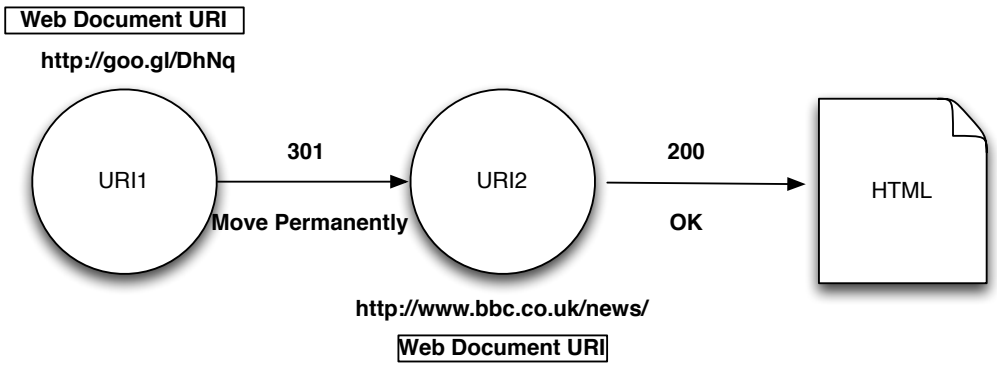


Figure 4.4: HTTP 301 Permanent Redirect on the Web

Temporary Redirect: There are other two common HTTP Redirect Status Codes used on the Web: 302 Found and 307 Temporary Redirect. Semantically, there is no difference between 302 and 307. In both cases are the requested resource resides temporarily under a different URI. The 307 code was introduced because many user agent implementations at the time treated 302 as if it were a 303 response, performing a HTTP GET on the Location field-value regardless of

[1] https://bitly.com/shorten/
[2] twitter.com

the original request method [64]. Figure 4.5 shows an example of 302 redirection, where the official Xbox[1] website URI is temporarily redirected to a different URI.

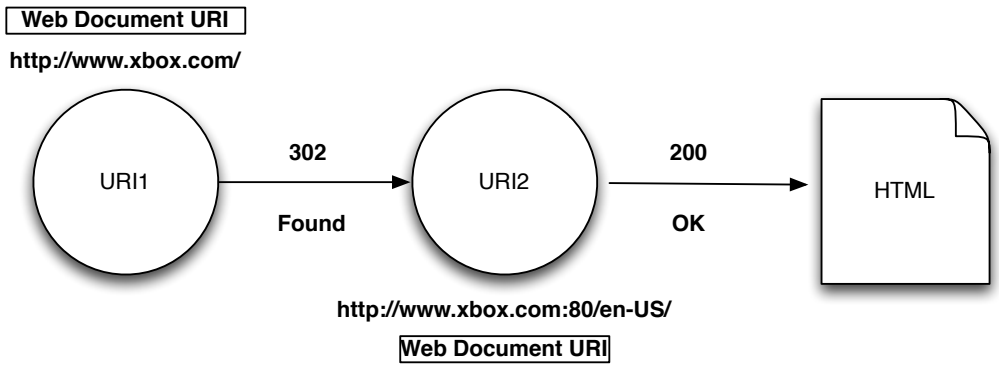


Figure 4.5: HTTP 302 Found on the Web

4.2.2 HTTP redirect for the Semantic Web

The Web was not invented to handle the real world objects in the first place, therefore, how real world objects interact with other existing status codes has not been studied before. However, these seems to complicate the URI dereferencing process.

Permanent Redirect: Similar scenarios could apply to the Real World Object URI: for instance, a person wants to have a new URI for himself with a better domain name, but his old URI has been referenced elsewhere, for instance, in others' FOAF files. Such a person might decide to permanently redirect the

[1] Xbox.com

old URI to the new. The Geoname Ontology implemented a similar scenario as illustrated in Figure 4.6. When dereferencing the Geoname Ontology URI http://www.geonames.org/ontology, it is followed by a 301 Move Permanently which redirects to a new URI (with a slash at the end) http://www.geonames.org/ontology/. This URI implemented a HTTP 303 redirection to the a RDF URI http://www.geonames.org/ontology/ontology_v2.2.1.rdf.

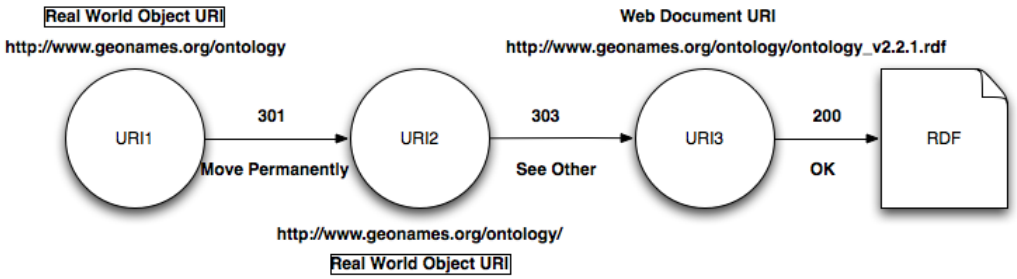


Figure 4.6: HTTP 301 Permanent Redirect between Real World Object URIs

When a real world object URI is permanently redirected to another URI, it means these two resources are refer to the same thing. Semantically, it is as if we are applying 'owl:sameas' relationship between these two URIs.

Temporary Redirect: In the HTTP Temporary Redirect, when dereferencing a URI, the requested URI is expected in the returned RDF description as well. Therefore, both URI Permanent Redirect and Temporary Redirect are the same from the semantic perspective, which signifies that the requested URI is the same as the redirected URI (URI alias). The difference between these two types is that in the Permanent Redirect scenario, the data consumers are expected to reuse the redirected URI as their future reference in their RDFs. In contrast, in the Temporary Redirect scenario the data consumers should keep on referencing the

originally requested URI. Figure 4.7 shows a chain of HTTP Redirects which can happen at any stage of the dereferencing process, either between real world object URIs or document URIs.

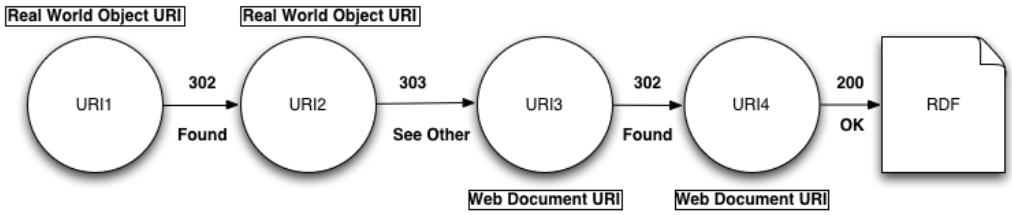


Figure 4.7: Chains of HTTP Redirect with 302 Found Status Code

4.2.3 HTTP redirect between real world object URIs and document URIs

Section 4.1 discussed redirections between real world object URIs and document URIs, which use "303 redirect" and "unhash" between them. Sections 4.2.1 and 4.2.2 discussed the temporary and permanent redirection that can happen on the Web and the Semantic Web. The conclusion was that permanent redirection and temporary redirection can only be meaningful between URIs of the same nature - either between real world object URIs or information resource URIs.

To sum up, three are three common redirection patterns on the Web, see Figure 4.8.

1. Redirections between Web Document URIs, including commonly used codes 301/302/303/307/unhash;

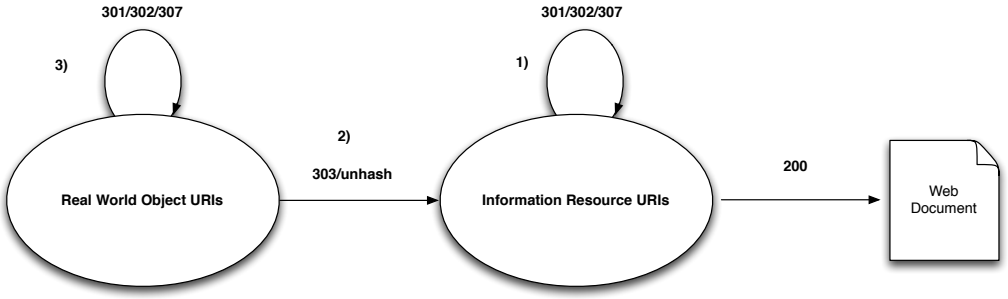


Figure 4.8: Redirection between Real World Object URIs and Web Document URIs

2. Redirections between Real World Object URIs, including codes 301/302/307;

3. Redirections from Real World Object URIs to Web Document URIs only using 303/unhash.

This thesis refers to the Real World Object URIs as **ObjURIs** and to Web Document URIs as **DocURIs** from this this point for the ease of discussion purpose. For the redirection of the URIs that have the same type of resource, i.e., redirection only between ObjURIs or between DocURIs, all types of HTTP redirection remain valid. However, the redirection from ObjURIs to DocURIs can only be carried out by using either a 303 Status Code or by 'unhash'. From this point onwards, 'unhash' is treated as a special code to achieve a "redirections" between URIs from the client side. It is defined below

Definition 5. ***Unhash****: A special client-side status code indicates that the Information about the requested hash URI can be found under the 'unhashed' URI.*

In the following section, a URI dereferencing algorithm is designed that is based

on the understanding of HTTP redirection discussed above.

4.3 HTTP URI Dereferencing Model

The general goal of our model is to provide a formal basis for analysing chains of HTTP requests and responses, and determine the URI and its corresponding RDF description. The HTTP dereferencing process is defined as a function (4.1), which takes a Status and a Response and returns a new Status. The reason the a Status and a Response are kept track of is that the Semantic Web agents are required to track the requested URI as we discussed in section 4.1. This function can be considered analogous to the relation found in an abstract rewriting system [54]. The Status is composed of a requested URI and a Status Code from the previous response (if any), as well as a Temporary URI from the previous response (if any). A response is composed of a HTTP Status Code and URI returned in the HTTP response header named "Location" (if any). In the following, each HTTP Status Code is first interpreted according to the HTTP specification, then the nature of the resource which the URI identifies is inferred and summarized, see Table 4.1. Based on this, a set of Dereferencing URI rules is created to describe each HTTP request and response pattern from a client/agent perspective.

URI Rewriting Rules:

$$\textbf{Deref:Status} \times \textbf{Response} \rightarrow \textbf{Status} \tag{4.1}$$

Code	Meaning on the Web	Semantic Web Interpretation	Response Type	Resource Type	Formula
200	OK	OK	DOC (document)	Information Resource	(4.2) (4.7)
301	Moved Permanently	The requested URI is permanently equivalent to the redirected URI	URI	Unknown	(4.3) (4.8)
302	Found	The requested URI is temporarily equivalent to the redirected URI	URI	Unknown	(4.3) (4.8)
303	See Other	The information about the requested URI can be found under the redirected URI	URI	Unknown	(4.4) (4.8)
307	Temporary Redirect	The requested URI is temporarily equivalent to the redirected URI	URI	Unknown	(4.3) (4.8)
4XX	Client Error	Client Error	BAD	Unknown	(4.5)
5XX	Server Error	Server Error	BAD	Unknown	(4.5)
unhash (added)	See Fragment Identifier	The information about the requested URI can be found under the 'unhashed' URI	-	Unknown	(4.6)

Table 4.1: Interpretations of HTTP Status Codes and the nature of the resource request that the URI identifies

---

Where Status and Reponse are :

Status : $(RequestURI, StatusCode, TemporaryURI)$

Response : $(StatusCode, URI)$

$StatusCode$: $200, 301/2/3/7, 4XX, 5XX, unhash, BAD, ANY, \_(denote\ empty)$

4.3.1 HTTP status codes 200-299

The most commonly used HTTP response code in the range 200-299 is 200 OK, which indicates that the retrieval has been successful and a representation of the resource has been returned as part of the response. Since the resource identified by the URI has a representation, the resource is therefore identifies as an information resource. This axiom is described in Formula (4.2). We started with a single request and nothing else from previous states, then we obtained a response 200 OK and a document (therefore no URI is returned in this case).

$$Deref : (URI_r, \_, ANY)(200, \_) \rightarrow (URI_r, 200, ANY) \tag{4.2}$$

4.3.2 HTTP status codes 300-399

When a URI is inaccessible, the HTTP indicates the need for a redirection with response codes in the range 300-399. The 301 is a popular codes which indicates that the requested URI is assigned to a new permanent URI. The Status Code 302 Found and 307 Temporary Redirect behave in a similar fashion, where the original request is temporarily redirected to a new URI. As we summarised earlier, 301/302/307 implies the original URI is the same as the redirected URI. Therefore, when clients request a URI, the server will issue a 301/302/307 Status Code with a new URI_t to replace the original request URI_r. So, the dereferencing result in the Formula (4.3) is that the requested URI is overwritten by the newly URI_t returned one. The Status Code 303 See Other indicates that othe response to the request can be found under a different URI and should be retrieved using an HTTP GET method on that resource. This method exists primarily to allow the output of a post-activated script to redirect the user agent to a selected resource. The new URI is not a substitute reference for the originally requested resource.Therefore, in Formula (4.4) the original request URI_r will not be rewritten by the See Other URI_t.

$$Deref : (URI_r, \_, ANY)(301/302/307, URI_t) \rightarrow (URI_t, \_, URI_t) \quad (4.3)$$

$$Deref : (URI_r, \_, ANY)(303, URI_t) \rightarrow (URI_r, 303, URI_t) \quad (4.4)$$

4.3.3 HTTP status codes 400-599

The HTTP indicates errors in the processing of a request with response codes in the range 400-599. Such behaviours are modelled in Formula (4.5), where no matter what URI is requested and what Code is returned from previous dereferencing, it results in a BAD Status Code.

$$Deref : (URI_r, ANY, ANY)(4XX/5XX, \_) \rightarrow (URI_r, BAD, \_) \qquad (4.5)$$

4.3.4 HTTP special case for the hash URI and the 303 redirection

Formula (4.6) demonstrates a case of dereferencing a Hash URI. When requesting a Hash URI, it will unhash the hash fragment from the client-side. Since the URI dereferencing is being modelled from the client side, such a behaviour can be tracked. In the result, the request $URI_r\#frag$ is not replaced by the URI_r (the unhashed URI), because the URI with the hash fragment is the primary interest for a client when using it as an ObjURI.

$$Deref : (URI_r\#frag, ANY, ANY)(unhash, URI_r) \rightarrow (URI_r\#frag, unhash, URI_r) \quad (4.6)$$

Formula (4.7) and (4.8) are two rules that deal with the previous Status Code, which may include a 303 or an unhash. The reason there are two more different formula is that 303 and unhash codes are an indicators of whether there has been a redirection between a real world object URIs and a document URIs. Existence of either of these two Status Codes indicates the requested URI may identify a Real World Object, and it should not be substituted by any other type of redirected URI in the dereferencing process.

$$Deref : (URI_r, 303/unhash, URI_t)(200, \_) \rightarrow (URI_r, 303/unhash, URI_t) \quad (4.7)$$

$$Deref : (URI_r, 303/unhash, ANY)(301/302/303/307/unhash, URI_t) \rightarrow (URI_r, 303/unhash, URI_t) \quad (4.8)$$

Due to the publishing mechanism restriction, as mentioned earlier that it is not possible to use the hash URI and 303 redirection to publish RDF for information resources, we can therefore determine whether a requested URI identifies an information resource or a Real World Object. In the final returned Status tuple (RequestURI, Status Code, TemporaryURI), if the Status Code is 200, then the requested URI identifies a document, thus an **Information Resource URI** (DocURI). If the Status Code is 303/unhash and the requested URI can be found in the Web document which is identified by the returned TemporaryURI, then a Resource Description has been found. The requested URI identifies an object, thus is a **Real World Object URI** (ObjURI).

The URI dereferencing rules proposed here can be extended and updated according to the changes in HTTP and URI specifications. For instance, the rarely used status code 300 Multiple Choices has not been discussed. According to the specification, one can understand its behaviour in a similar way to that of temporary redirection, and implement the rules as such.

4.4 The Hyperthing Validator for Linked Data URIs

To demonstrate our model, a Semantic Web URI validator named Hyperthing [1] has been implemented which aims to help users to publish Linked Data better.

[1] http://www.hyperthing.org

This validator has three purposes: 1) It determines whether the requested URI identifies a Real World Object or a document; 2) It validates the URI publishing mechanism against the W3C 303 URIs and Hash URIs guidance; 3) It can be used to check the validity of the chains of the redirection between the Real World Object URI and document URI to prevent the data publisher mistakenly redirecting between these two kinds. For example, it is would be a wrong implementation if it permanently redirected a Real World Object URI to a document URI, since they refer two different things. The validator is implemented in Python [1] under a web framework named Django [2].

Figure 4.9 shows the Hyperthing validator - when validating the URI, http://dbpedia.org/resource/Tim_Berners-Lee. The Results Section shows that the requested URI identifies a Real World Object, and the Summary Section shows the validated results against the W3C guidance from three perspectives: 1) it checks whether the return document is RDF by requesting with content type 'application/rdf+xml' header, 2) it validates whether the dereferencing process includes 303 or unhash code, 3) it parses the RDF to see if relevant triples can be found for the requested URI.

The validator also represents the whole Dereferencing Process as a diagram and illustrates 'What URI identifies'. When users click the 'show more' link, it will explain the URI rewriting rules applied to dereference the URI. The dereferencing process when request URI1: http://dbpedia.org/resource/Tim_Berners-Lee

[1] http://www.python.org/
[2] https://www.djangoproject.com/

(as the same case, it was depicted in the section 4.1 Figure 4.3) can be described by the model as follows:

$$Deref : ((URI_1, \_, ANY)(303, URI_2))(200, \_) \tag{4.9}$$
$$\rightarrow (URI_1, 303, URI_2)(200, \_) \qquad applying formula(4.4)$$
$$\rightarrow (URI_1, 200, URI_2) \qquad applying formula(4.7)$$

In the returned final Status tuple, the URI1 description can be found in the RDF document which is located under URI2, where a total of 172 triples have been found. Therefore, the request URI1 identifies a Real World Object.

4.5 Empirical study of Linked Data URIs

To evaluate our model and the validator, we carried out an empirical study of Linked Data URIs.

4.5.1 Methodology

The most visible adoption of Linked Data is the Linking Open Data project [50] [1]. This project summaries a number of datasets published, their linkage, and

[1] http://www.w3.org/wiki/SweoIG/TaskForces/CommunityProjects/LinkingOpenData

the most popular ontologies used. An ontology is an important piece data as it defines the schema of the data. URIs in ontology are used to identify concepts and relations, therefore, they are required follow the Linked Data publishing practice. Ontology URIs are important links for the navigation and discovery. When consuming Linked Data, if a popular ontology URI is referenced in many datasets, it is likely to be dereferenced frequently when consuming. For example, the owl:sameas URI is heavily used by publishers to state their instances are the same as others, and consumers from anywhere on the Web of Data will need to dereference this URI frequently to interpret any triples contains such a URI.

Based on the above assumption, 25 of the most referenced ontology URIs[1] on the Linked Data Cloud were selected as a case study. A breadth-first web crawler was designed, based on the dereferencing algorithm developed in the previous chapter, to dereference all ontology URIs as well as their properties. The crawler begins at the root ontology URI and explores all the neighbouring URIs. Each of the crawler's requests was send out with the header Internet Media Type RDF/XML. The entire data crawling, data processing and analysis were performed on a 4GB RAM virtual machine (similar to a workstation capability) at the University of Southampton. The crawler and data analysis were implemented in Python under an Ubuntu Linux environment and data were simply stored as a text document. The following section presents the dereferencing results.

[1] http://richard.cyganiak.de/2007/10/lod/

4.5.2 Results

The experiments were carried on 10 December 2011 and the results are shown in Table 4.2. The total indicates the number of URIs in an ontology (including the ontology itself), where h denotes the Hash URIs, and s denotes slash URIs. The *succ* means when dereferencing URIs, the requested URI has been found in the returned RDF document (however, it does not imply that it follows the Hash URIs and slash URIs practice).

The results found were categorised as "bad, wrongImp, noRDF, notFound" as follows:

(1) **bad**: Some URIs cannot be resolved at all (e.g. http://www.w3.org/TR/NOTE-datetime);

(2) **wrongImp**: The implementation did not follow either Slash URIs or Hash URIs practice (e.g. the most referenced ontology http://purl.org/dc/elements/1.1/publisher and http://www.w3.org/2004/02/skos/core#note)

(3) **noRDF**: No RDF returned (e.g. http://dublincore.org/usage/terms/history/#BibliographicResource-001)

(4) **notFound**: The requested URI is not found in the returned RDF (e.g.

ontology	total		succ		bad		noRDF		notFound		wrongImpl	
	h	s	h	s	h	s	h	s	h	s	h	s
http://purl.org/dc/elements/1.1/	17	18	2	17	0	0	15	1	0	0	0	18
http://dbpedia.org/resource/	0	1	0	0	0	0	0	0	0	1	0	0
http://purl.org/stuff/rev#	30	22	30	9	0	1	0	12	0	0	18	19
http://www.openarchives.org/ore/terms/	2	19	2	18	0	1	0	0	0	0	0	19
http://usefulinc.com/ns/doap#	60	7	60	4	0	1	0	1	0	1	0	3
http://www.w3.org/2006/time#	81	2	74	1	6	0	0	1	1	0	6	2
http://rdf.geospecies.org/ont/geospecies#	410	21	404	16	0	2	5	3	1	0	5	16
http://xmlns.com/foaf/0.1/	16	84	16	82	0	1	0	1	0	0	1	12
http://rdfs.org/sioc/ns#	104	12	103	11	1	0	0	1	0	0	1	8
http://swrc.ontoware.org/ontology#	135	1	6	0	0	0	129	1	1	1	0	1
http://www.geonames.org/ontology#	731	4	728	2	1	0	1	1	1	1	718	2
http://creativecommons.org/ns#	1	0	0	0	0	0	1	0	0	0	0	0
http://purl.org/NET/c4dm/event.owl#	33	8	32	8	0	0	0	0	1	0	22	5
http://umbel.org/umbel#	61	5	59	4	1	0	0	0	1	1	2	3
http://www.w3.org/2001/XMLSchema#	1	0	0	0	1	0	0	0	0	0	1	0
http://purl.org/goodrelations/v1#	180	9	173	7	6	0	0	2	1	0	170	8
http://purl.org/vocab/bio/0.1/	14	103	15	86	0	1	0	15	0	0	1	23
http://purl.org/ontology/mo/	37	262	31	223	5	36	0	2	1	1	20	47
http://purl.org/ontology/bibo/	29	180	28	32	1	148	0	0	0	0	14	170
http://purl.org/vocab/frbr/core#	84	42	83	22	1	2	0	18	0	0	75	34
http://purl.org/dc/terms/	103	139	5	119	0	1	98	19	0	0	0	137
http://www.w3.org/2004/02/skos/core#	45	2	44	1	0	0	0	1	1	0	33	1
http://www.aktors.org/ontology/portal#	357	7	7	4	1	0	349	3	1	0	1	7
http://www.w3.org/2003/01/geo/wgs84_pos#	11	1	11	1	0	0	0	0	0	0	0	0
http://www.w3.org/2006/vcard/ns#	84	6	79	6	3	0	1	0	1	0	4	6
Total:	**3581**		**2665**		**221**		**681**		**17**		**1633**	

Table 4.2: Results of dereferencing the 25 most frequently used Ontologies on the Web

http://www.w3.org/2006/time#).

In case (1), most of the URIs can be dereferenced successfully when requested with content-type: text/html instead of rdf+xml. This means that webmasters may have difficulties in their implementation of the content-type in the HTTP header or in implementation of the content negotiation.

In case (2), some slash URIs used the wrong redirection status code - 302 Found status code instead of 303 See Other. It seems that many ontologies with domain name http://purl.org have such a problem. We found out that, in 2006, there was no way to configure 303 response codes on PURL servers [1]. Therefore, several ontologies were configured that misused the 302 redirect. However, after many years, the implementation of these URIs remains unchanged. This indicates that these ontologies were rarely updated or poorly maintained. In a rare case, the http://www.w3.org/2004/02/skos/core#note ontology used both the hash URI and 303 See Other redirection at the same time. This is controversial as it raises the question about the meaning of the 'unhash'. Since the unhash mechanism should have the same meaning as 303 in this scenario, it acts as if the URI with a hash tag was redirected the URI with hash tag to the unhashed URI (a document URI); this document URI then 303 redirected to a document URI. Since the URI on the Semantic Web is no longer simply an address, the Linked Data publishing mechanism would also require the HTTP specification to have a clear definition of the semantics of the HTTP status code. For example, the 301 Move Permanently would assert two URIs are semantically equivalent.

In case (3), some of the ontologies were not returned due to the move of the server or technical problems. This raises another interesting issue - when a 404 on the Web is encountered, it simply means that this webpage does not exist; in contrast, on the Semantic Web, if an ontology is not found. It also means that it is does not merely mean the resource description not found; it also means that it is not able to determine the meaning of all RDF triples containing such a

[1] http://www.w3.org/TR/swbp-vocab-pub/

	Hash URIs	**Slash URIs**
succ (total)	2626	955
succ (wrongImp)	76%(55%)	71%(80%)
bad	1%	20%
noRDF	23%	9%
notFound	0.4%	0.6%

Table 4.3: Comparison of the dereferencing result of Hash URIs and 303 URIs

ontology URI on the cloud.

In case (4), when a RDF is returned with 200OK, if the requested URI in a returned RDF cannot be found, the description for this resource is considered as not found. In this case, no link mentioned in the RDF should not be followed to determine its meaning.

Table 4.3 shows the summarised results of the dereferencing Hash URIs and Slash URIs. For both types of URI, more than 70% of them can be dereferenced successfully (found RDF description about requested URIs). However, 55% of Hash URIs and 80% of Slash URIs out of *succ* cases did not follow the publishing practice correctly. This may indicate that the publishing approach of the Hash URIs has been better adopted by the community than the Slash URIs. However, one may argue that the Slash URIs implementation might also include those data publishers who neglected the practice at all. Therefore, we cannot conclude Hash URI is better adopted than the Slash URIs. However, the overall results indicate that there is relatively high error rate in publishing URIs. These results are consistent with the experiments mentioned in the background section, where 45.5% of URIs were successfully dereferenced with content type application/rdf+xml.

From the cases discussed previously, it shows that data publishers are not only having technical difficulties in deployment of Semantic Web URIs, but are also misusing and misunderstanding the URI identities, thus making mistakes in publishing.

4.6 Conclusions

This Chapter studied Linked Data publishing mechanism and discussed publishing issues on the Web which have not previously been discussed by the W3C practice. In particular, all types of the HTTP redirection of the ObjURIs and DocURIs were summarised. In order to study it systematically, the HTTP URI dereferencing process was formally modelled according to its specification and summarised in a set of URI Rewriting Rules. This is essentially a generic dereferencing algorithm that can be used by agents to dereference a URI and understand what the URI refers to. Based on the algorithm, a Linked Data validator called Hyperthing was implemented which can be used to help data publishers to publish Semantic Web URIs as well as to differentiate whether a URI identity is a real world object or web document. To evaluate the validator and the current deployment of the Semantic Web URIs, an empirical study was carried into dereferencing the most frequently referenced 25 ontology URIs and their properties. The conclusion was that, on average, more than half of Linked Data ontology URIs did not follow the published guidance to make the URI identity consistent. The Hyperthing validator can be used to avoid these publishing mistakes.

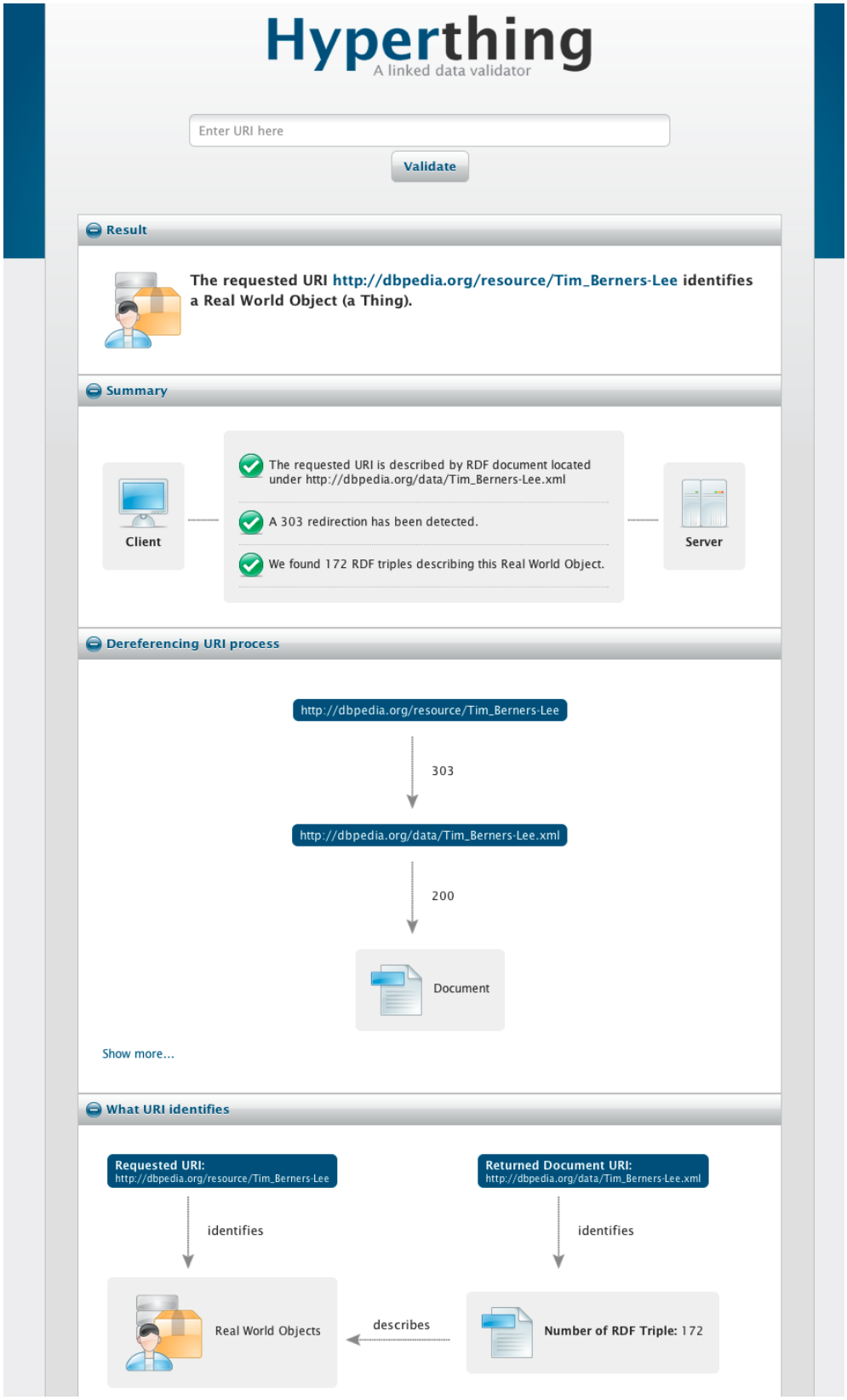


Figure 4.9: Hyperthing the Semantic Web URI Validator

Chapter 5

Analysis of Linkages Between Linked Datasets

How are Linked Data interlinked by URIs?

Chapter 4 studied the dereferencing of a single URI, which is essentially the first three principles of Linked Data (using a URI to identify data, HTTP to look up a URI, and RDF to describe data). The fourth principle of the Linked Data is that, when publishing an RDF, one should include other URIs, so that the agent can follow those URIs to discover more things. Indeed, if a URI is the name of real world object, the more RDF description that is provided about this URI,

the more information will become elaborate. The agent can know more about what a URI refers to. If these RDF descriptions are provided with links to other datasets, the URI will have 'global' context instead of the 'local' context. Agents can then follow links discover more information which has been created in other datasets. In this way, others' resource are being re-used, and more and better services can be provided, interoperability is also being achieved. The Web of Linked Data also have the network effect [134] - similar to the telephone network, the more people who own a telephone, the more valuable the telephone is to each owner. Also there have been many studies of the Web which believe that the more URIs referenced the better it is [86]. The Pagerank algorithm [117] is an example which applies academic citation network analysis on the Web to rank the importance of a page. This also applies to Linked Data.

Hartig [83] discussed the importance of interlinked URIs when querying over the Web of data. Unlike the traditional database executed over a finite datasets, his approach investigated the idea that querying on the "infinite" Web of data by dereferencing URIs and generating databases: he named it traversal based query execution. There are many cases, which the way data is interlinked will determine how it is discovered and used. This Chapter studies the connectivity of Linked Data. It focuses on analysing how datasets are linked together - whether an agent can follow links to discover other datasets, in other words, its 'global context' information.

5.1 The Connectivity of Linked Data

The Linking Open Data community project [1] summarises the current deployment of Linked Data as a cloud diagram (the 19/09/2011 version) [50] as shown in Figure 5.1. In total, 295 datasets with more than 5million triples have been published and the cloud diagram shows linkages between datasets. The direction of arrows indicates that following the link navigates from one dataset to another. For example, an arrow from A to B means that dataset A contains RDF triples that reference URI from B and we can navigate from dataset A to dataset B, but not vice versa.

Figure 5.1 shows that there are three clusters of datasets are densely linked. Although, overall, the diagram indicates there are links between the majority of datasets, only some datasets have reciprocal links. The majority of the datasets are only sparsely linked to other datasets. The report, State of the LOD Cloud [29], summarises more detailed link statistics of the diagram. Figure 5.2 shows the out-links of the Web of Linked Data, where about 10% of the datasets have no outgoing links at all and 30.5% of the datasets have 1 to 1,000 outgoing links. Figure 5.3 shows linkages between the Web of Linked Data, where about 33% of the datasets link to only 1 dataset and 21% of datasets link to 2 datasets. Therefore, more than 60% of datasets have fewer than 10,000 out links, and the majority of datasets link to 1 or 2 datasets. However, this does not provide us with a detailed insight of how two datasets are linked.

[1] http://lod-cloud.net

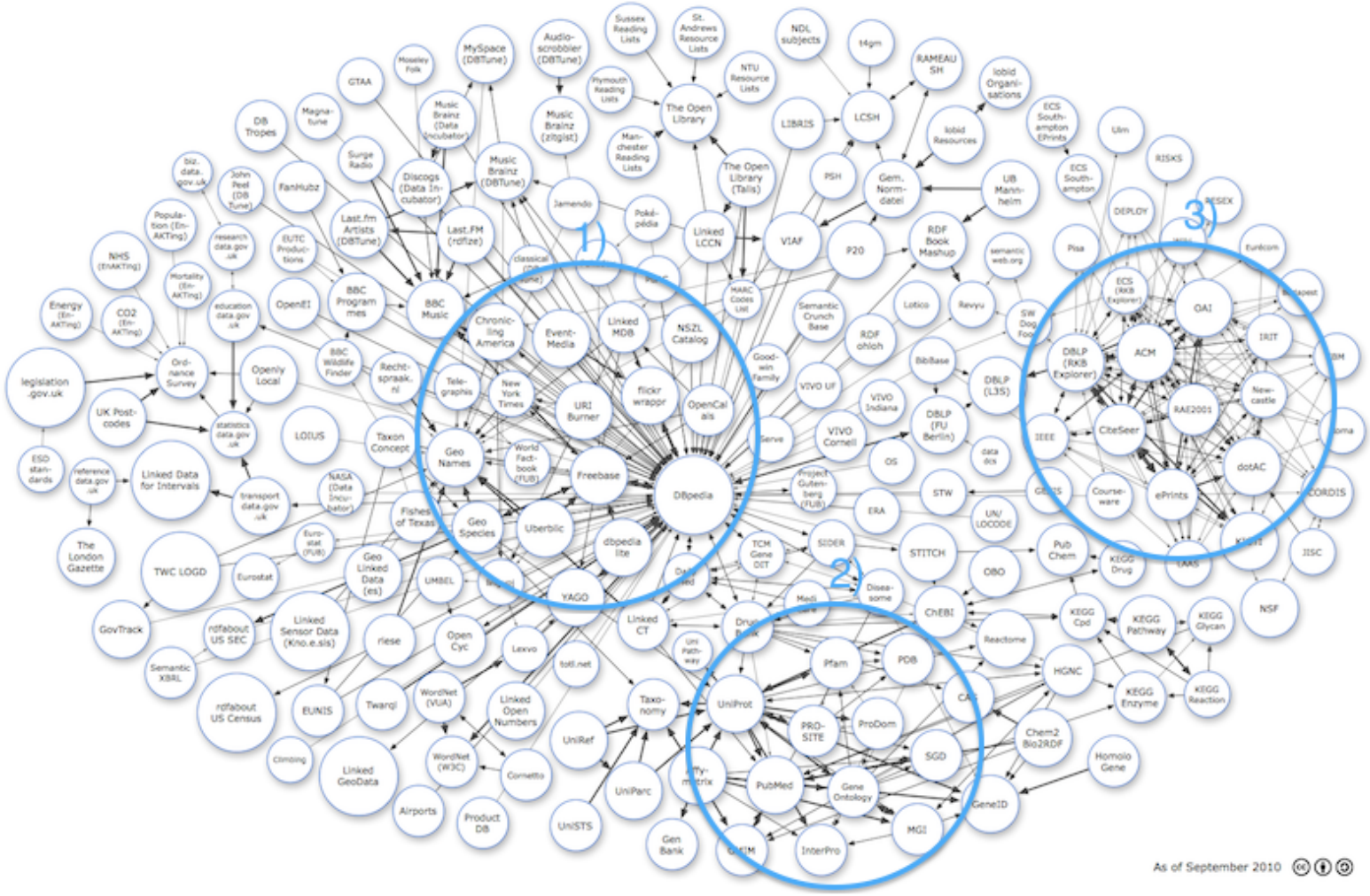


Figure 5.1: Linked Data Cloud (08/10/2011) [50]

(Out-)Links	Number of datasets
0 outgoing links	30 (10.17 %)
up to 1,000	90 (30.51 %)
1,000 to 10,000	58 (19.66 %)
10,000 to 100,000	45 (15.25 %)
100,000 to 1,000,000	43 (14.58 %)
more than 1,000,000	29 (9.83 %)
	237

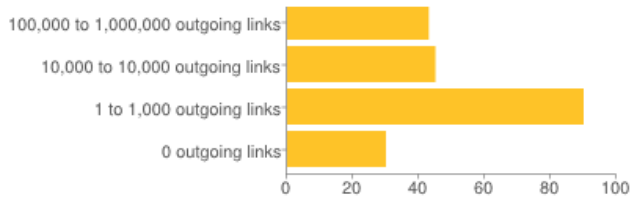


Figure 5.2: Out-links of the Web of Linked Data [29]

Number of linked datasets	Number of datasets
more than 10	27 (9.15 %)
6 to 10	17 (5.76 %)
5	5 (1.69 %)
4	19 (6.44 %)
3	38 (12.88 %)
2	62 (21.02 %)
1	98 (33.22 %)
	266

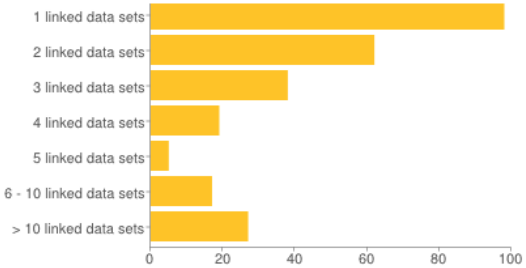


Figure 5.3: Linkages of the Web of Linked Data [29]

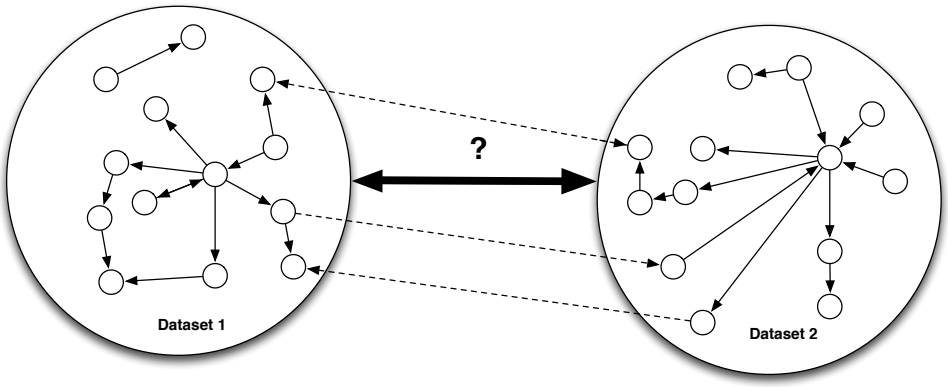


Figure 5.4: How two datasets are interlinked?

The New York Times dataset, as an example, is linked to the Freebase dataset. However, it certainly does not mean that there are foreign URIs from every single piece of data in the New York Times to the Freebase. The bidirectional link only indicates that there are some data has bidirectional links. For datasets that are already linked, we can still question the foreign URIs coverage - and ask what portion of the data have foreign URIs to navigate from one dataset to another. As shown in Figure 5.4, in order to understand how the Web of Linked Data are connected, they need to be analysed in more details. The following section, first models the connectivity of the Web of Linked Data, then a case study is carried out to see how datasets are interlinked.

5.1.1 Modelling the Navigability of Linked Data

The Web of Linked Data are commonly modelled as a giant interlinked directed RDF graphs. Here it is modelled from an agent point of view. Chapter 4 defined

a dereference function (4.1) which takes a 'Status' and 'Response' to generate a new 'Status'. This section only studies the dereference URI 'successful', that is when dereferencing a URI, we obtained a HTTP 200 OK and a RDF document. The dereference function defined previously is used for agents to track the dereferencing process, here the focus is on the returned document RDF, hence we introduce a Semantic Web URI resolution function as below:

Definition 6. ***Semantic Web URI Resolution****: Given a uri, its resolution gives an RDF document, which describes the uri. This process is denote as* $\mathbf{resov}(uri) \rightarrow RDF$.

In addition, uris(RDF) denotes a function which returns the set of URIs in a given RDF documents.

If resolving a URI in a RDF document leads to another RDF document, one RDF document is said to be 'navigable' (linked) to another. We introduce the notion of navigability between RDFs as below:

Definition 7. ***Navigability of RDF documents****:* RDF_1 *can navigate to* RDF_2*, if* $RDF_2 = resov(uri)$*, where* $uri \in uirs(RDF_1)$*. This navigability is denoted as* $RDF_1 \Longrightarrow RDF_2$*. This uri is called a* foreign URI *in* RDF_1.

For the case, $RDF_1 \Longrightarrow RDF_2$ where RDF_1 and RDF_2 belong to different datasets, the above foreign URI is called as an *across-datasets foreign URI*. This often also reflects in the domain name of the URIs in a RDF - an across-datasets

foreign URI often has a different namespace than other URIs in a RDF.

On the Linked Data Cloud diagram, the bidirectional arrows indicate that there are reciprocal links between a pair of datasets, where users/agents can follow links to navigate between the dataset from either directions. Bidirectional navigability is defined as below:

Definition 8. ***Bidirectional navigability of RDF documents****: Two RDF documents can mutually navigate from one to another, if* $RDF_1 \Longrightarrow RDF_2$ *and* $RDF_2 \Longrightarrow RDF_1$*. This bidirectional navigability is denoted as* $RDF_1 \Longleftrightarrow RDF_2$*.*

With this basis, we carried out studies to analyse the navigability of Linked Data.

5.2 An overview analysis of the navigability of Linked Data

In an extreme picture, the Web of Linked Data can be viewed as a single giant RDF graph, where all RDF in all datasets can be composed into a single RDF. In order to create an overview of the navigability, some metadata was collected that had been submitted by contributors to the CKAN [1] data portal (which the Linked Data cloud diagram was plotted based on). A typical piece of metadata consists

[1] http://datahub.io/

	Datasets	Number of triples across Linked Data RDF graph	Number of across datasets foreign URIs	Foreign triples coverage	Entity triples coverage	The across datasets foreign URIs coverage
Total	285	31.5 billion	425 million	1.3 %	17.9% (5,639 million)	Max. 7.5%

The number of entity triples coverage is an estimation based on the data provided by LODStats project [1].

Table 5.1: An overview of the navigability of the Web of Linked Data as a giant RDF graph

of a number of triples and links to particular datasets. A simple web crawler was written which parsed the metadata of these datasets. Table 5.1 summarises the results. A total of 285 pieces of valid metadata on datasets was collected from the CKAN database (10 of the metadata pages resulted in HTTP error 404). A total of 31.5 billion RDF triples published were found and a total of 425 million foreign URIs. Therefore, only around 1.3% of the triples in the datasets are foreign RDF triples. The foreign URIs simply links two resources from different datasets. This is relatively a low coverage. It should be noted that these 31.5 billion triples could contain duplicated resources triples, likewise with the foreign triples; therefore, only a rough estimation of interlinked the Web of Linked Data resources.

According to the latest record [2] of the LODstats project [3], on average there are about 2,180,652 triples, where there are about 390,326 entities. Assuming these entities are all resources, about 18% triples are the triples which contains a URI

[2]http://stats.lod2.eu/stats last accessed on 11th Sep 2013
[3]http://aksw.org/Projects/LODStats.html

identify a resource. Based on this estimated ratio, there is about 5,639 million triples contains a URI identifying a resource. Regardless of the fact that there would be many duplicated foreign URIs, at most the 7.5 % of resources are linked to other across-datasets URIs (has a 'global context').

The across datasets foreign URIs coverage is relatively low, and as we mentioned in section 5.1 Figure 5.4, for those datasets that are linked, their linking patterns can still be questioned. A case study is carried out below to identify linking patterns between these datasets.

5.3 Case Study: the linking patterns of the Web of Linked Data

Clarify methodology: choice of NYTimes in ??5.3

Three linked clusters can be identified visually on the Linked Data graph, shown in Figure 5.1 highlighted by blue cycles. The first cluster is datasets which are densely linked to the DBpedia dataset, the second is PubMed related datasets, and the third is ACM related datasets. As the DBpedia is often described as the nucleus of a web of data [7], this cluster was chosen to analyze the linking patterns of the Web of data. In particular, we will chose NYTimes dataset from this cluster as a starting point to explore the linking patterns of the Web of data.

The reason the NYTimes dataset was chosen is that it is a good representative dataset from this cluster which links to both big datasets such as Freebase [34] and DBpedia, as well as smaller datasets such as LODE(Linked Open Data Ecology) [130].

END

5.3.1 Methodology

This study focused on analysing linkages of across-datasets foreign URIs, therefore, the first step was to collect the across-datasets foreign URIs. For example, a triple in the NYTimes dataset contains a foreign URI to Freebase shown as below:

<http://data.NYTimes.com/43397021109288751180> <http://www.w3.org/2002/07/owl#sameAs> <http://rdf.freebase.com/ns/en.wildfire>

We wrote a web crawler that went through all the datasets mentioned above and collected all across-datasets foreign URIs.

These foreign URIs triples was then modelled as a directed bipartite graph. A bipartite graph is one whose vertices can be divided into two disjoint sets, U and V, such that every edge connects a vertex in U to one in V; that is, U and V are in-

	URIs	**in-links**	**out-links**	**in- and out- links**
Total	1,037,667	28,035 (2.7%)	23,700 (2.2%)	51,735 (5.0%)

Table 5.2: New York Time datasets link summary

dependent sets. For instance, in the above-mentioned foreign URIs triple, the resource <http://data.NYTimes.com/43397021109288751180> which belongs to U (a set of NYTimes nodes) has a directed edge <http://www.w3.org/2002/07/owl#sameAs> that points to the resource <http://rdf.freebase.com/ns/en.wildfire> which belongs to V (a set of Freebase nodes).

Figure 5.5 shows how the NYTimes datasets are linked with their neighbouring datasets and their size (number of RDF triples). This is created based on the CKAN[1] records. The edge of the graph represents the number of directed links from one dataset to another. Table 5.2 summarises the linkages displayed. There is a total of more than 1 million links in the NYTimes datasets, where about a total of 28,035 URIs are linked to other datasets. Therefore, 2.7% of URIs can be discovered by following links from other datasets on the Linked Data Cloud. About 2.2% of URIs are linked to external datasets' URIs. In total, around 5.0% of URIs have a relationship with other datasets, which is again a low coverage. The following section analyses in detail how the NYTimes are linked with Freebase, DBpedia and LODE.

The entire data crawling, data processing and analysis were performed on a 4GB RAM virtual machine (similar to a workstation capability) at the University of Southampton. The data processing was implemented in Python under an Ubuntu

[1]http://thedatahub.org/

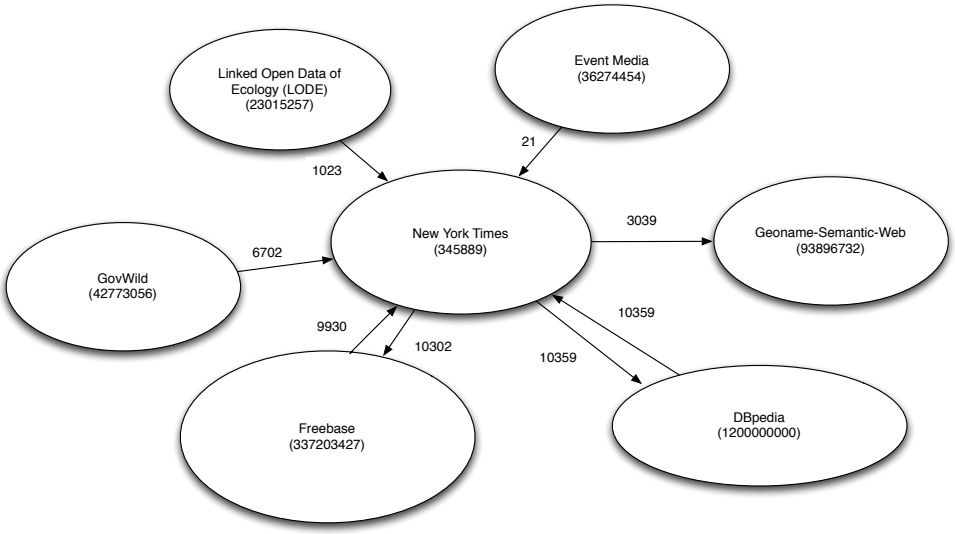


Figure 5.5: Linkages between New York Times and other datasets

Linux environment and data were simply stored in a GraphML document. The graph was analysed by using the Network Workbench [1] to determine the linking patterns of the graph: degree distribution, the properties (typed links) which links two datasets together and reciprocity of the foreign dataset links. The results are now presented.

5.3.2 Analysis of the linkage between NYTimes and Freebase

Table 5.3 shows the results of the analysis of the linkage between NYTimes and Freebase. According to the CKAN records[2], there are 10,302 links from NYTimes

[1] http://nwb.cns.iu.edu/

[2] http://thedatahub.org/dataset/NYTimes-linked-open-data

	NYTimes-Freebase	**Freebase-NYTimes**
CKAN foreign URIs Records	10302	9930
Collected foreign URIs	10424	9932
Linking Patterns (in-degree distribution)	0 (0.500144E+00) 1 (0.499568E+00) 2 (0.287880E-03)	0 (0.500933E+00) 1 (0.497251E+00) 2 (0.176527E-02) 3 (0.504363E-04)
Typed Links (Properties)	owl:sameas	owl:sameas
Reciprocal Links	9889	

Table 5.3: Linkages between NYTimes and Freebase

to Freebase, and we collected 10,424 foreign URIs online.

In the directed graph from NYTimes to Freebase as shown in the left column of the table, there are about 50% (0.500144E+00) of the resources (nodes of the graph) have no link point to Freebase. Around 49% (0.499568E+00) of resources have one link point to Freebase resources. A typical example of such link is a geographical data link between NYTimes and Freebase data. Fewer than 1% (0.287880E-03) of links have 2 links pointing to the Freebase resources. In the directed graph from Freebase to NYTimes, the 49% (0.497251E+00) of resources have 1 link pointing to NYTimes resources, and fewer than 1% of 2 and 3 Freebase links pointing to NYTimes resources. Both of them were linked by owl:sameas properties. In total, there were 9,889 reciprocity links between the two datasets.

	NYTimes-DBpedia	DBpedia-NYTimes
CKAN foreign URIs Records	10359	10359
Collected foreign URIs	10486	0
Linking Patterns (in-degree distribution)	0 (0.525193E+00) 1 (0.421855E+00) 2 (0.529018E-01) 3 (0.503347E-04)	Nil
Typed Links (Properties)	owl:sameas	Nil
Reciprocal Links	0	

Table 5.4: Linkages between NYTimes and DBpedia

5.3.3 Analysis of the linkage between NYTimes and DBpedia

Table 5.4 shows the linkage between NYTimes and DBpedia. Similar to the above NYTimes and Freebase case, 42% (0.421855E+00) of the NYTimes resources have a single link point to the DBpedia resources, while fewer than 1% of them have 2 and 3 linkage patterns. Although on the DBpedia official website, it claims that it has external links to NYTimes, no NYTimes foreign URIs were found in the DBpedia dataset [1] (links provided here were the NYTimes to DBpedia foreign URIs). This was also verified this from the DBpedia sparql endpoint. Therefore, there is only uni-directional linking to DBpedia datasets, so users/agents can follow the link to discover the DBpedia website, but not the reverse. All links were established by using the owl:sameas.

[1] http://wiki.dbpedia.org/Downloads37#externallinks

	NYTimes-LODE	LODE-NYTimes
CKAN foreign URIs Records	0	1023
Collected foreign URIs	0	4525
Linking Patterns (in-degree distribution)	Nil	0 (0.994129E+00) 1 (0.469704E-03) 2 (0.140911E-02) 3 (0.939408E-03) 4 (0.704556E-03) 5 (0.234852E-03) 10 (0.234852E-03) 11 (0.234852E-03) 15 (0.234852E-03) 28 (0.234852E-03) 31 (0.234852E-03) 2072 (0.234852E-03) 2086 (0.234852E-03)
Typed Links (Properties)	Nil	geospecies:isExpectedIn foaf:based_near eco:county eco:location geo:location
Reciprocal Links	0	

Table 5.5: Linkages between NYTimes and LODE

5.3.4 Analysis of the linkage between NYTimes and LODE

Table 5.5 shows the linkages between NYTimes and LODE datasets. Although it claims to have only 1023 links to NYTimes, 4525 links were collected. The CKAN record may be out of date. The majority of nodes in LODE are linked to a few URIs in NYTimes. For instance, 0.023% (0.234852E-03) of URIs have been linked 2086 times. Since many of these ecological data are generated based

on the location Taipei(Taiwan), the Taipei URI in NYTimes has been heavily referenced. This is rather different from the previous two cases.

5.4 Discussion

Interestingly, we found that NYTimes also hosted non-authoritative data about other datasets. For instance, in a location description RDF http://data.NYTimes.com/N41755357763759651471.rdf, it also hosted triples about the Geoname datasets. For example, it made the following statement:

<http://sws.geonames.org/4568929/> <http://www.geonames.org/ontology#inCountry> <http://www.geonames.org/countries/#PR>

Clearly, anyone on the Web can express anything on the Web. However, they should create their own URI and description of others data. In this scenario, the data publisher is expected to create their own location URI and assert that it is the same as the Geoname URI, and then add extra metadata about it. It is ineffective to hijack statements about other people's URIs as a subject directly in a RDF description, because that description is provided elsewhere and retrievable via its original URI. Theoretically, dereferencing this URI would not be able to dereference and obtain the triple mentioned. When browsing a URI node in a graph, we should only obtain all statements where the node is a subject or object and describing all blank nodes attached to the node by one arc. Therefore, when

dereferencing the NYTimes URI, the extra triples about Geoname published by NYTimes would not be parsed, thus making it redundant. This is considered as an AntiPatterns (the common mistakes in software practice) [38] in Linked Data publishing.

Another interesting case we identified is that the same URI is linked via different ontologies within one RDF. For example, http://ecowlim.tfri.gov.tw/lode/page/taif/CollectingEvent/SyunichiSasaki19200301Ta ipei19200301 is linked to http://data.NYTimes.com/N38414174717121564171 via four differently typed links (ontology property) - eco:county; eco:location; geo:location and foaf:based_near. The benefit of such publishing approaches is that it enables this data to be used by agents which are based on different ontologies.

Overall, although more than 6 datasets (all have more than 20 million RDF triples) claim to have links with NYTimes, the in- and out- link coverage is low. In other words, the navigability from one dataset to another is low. The linkages between datasets are context-based, and cases analysed exhibit two extreme patterns - either the majority of nodes are one-to-one parallel linked from one dataset to another (as shown in Figure 5.6 pattern (1)) or the majority of the nodes from one dataset only link to a few nodes in another dataset (as shown in Figure 5.6 pattern (2)). A typical example of the patter (1) is geographical data, for all the location instance is parallel linked to the same location data in another dataset. The pattern (2) is the case, where many of data instance in one dataset is belong to a single geographical location. This pattern indicates that by following the link we will always end up with a single instance in another dataset.

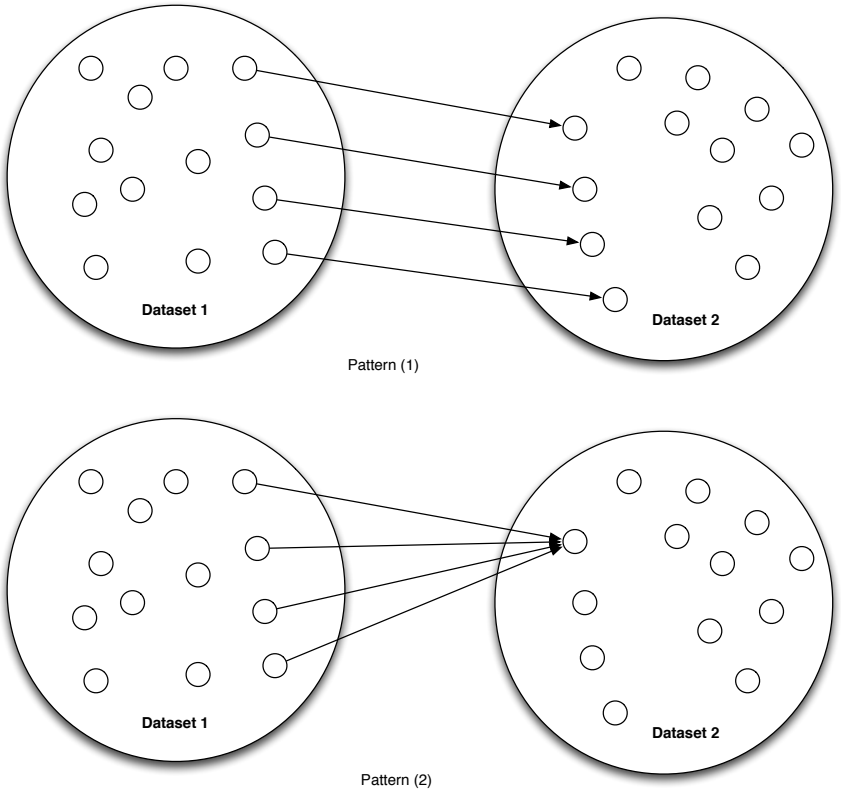


Figure 5.6: The linking patterns between two datasets

Only 1 out of the 6 datasets have reciprocal links. In all, the navigability of these datasets is rather limited.

We suppose these datasets were linked following two patterns mentioned above is because it is not so easy for data publisher to link their own data to other datasets (to create 'global' context of data). First of all, data publisher will need to search and identity what data to link to; after finding the target data, they need to either write their write own algorithm link to the data or link the data manually.

5.5 Conclusions

This Chapter studied the connectivity of the Web of Linked Data. We first modelled the navigability of the Linked Data. Two studies were then carried out to analyse how Linked Data are linked. We found that only 7.5% resources on the Linked Data cloud has across datasets foreign URIs, which is relatively low. The case study demonstrated that the linkage between datasets exhibited two extreme patterns: either the majority of data instances are one-to-one parallel linked from one dataset to another, or the majority of the data instances from one dataset only link to a few data instances of another dataset. This means that, for those linked datasets, an agent can only follow some of the resources in a datasets to discover other resources and their navigability could be rather limited. According to the updated state of the LOD cloud in 2014 [30], this still remains to be a problem on the largely connected Web of Linked Data. The linking patterns of these across-datasets URIs seems monotonous. More diversity linking patterns are desired to enable the discovering of data. The next Chapter investigates approaches to improve the navigability of the Linked Data.

Chapter 6

Link Services for Linked Data

How can Linked Data be made better linked?

This chapter examines approaches for improving the navigability of Linked Data by making it better linked, and proposes the notion of a link service for Linked Data.

There are many approaches to improving the linking of Linked Data: we may put the onus on the data publisher to publish more data and establish more linkages between datasets; we may use crowdsourcing to encourage data users to create links (please see author's paper on the distributed human computation approach

[156] investigated in the EnAKTing project); we may encourage data publishers to publish the metadata of datasets. For example the VoiD ontology [2] can be used to describe how a dataset is linked to another. However, it only facilitates navigation on the dataset level, which does not provide fine-grained granularity. Instead of focusing on the link creation, it is proposed to build link services on the Linked Data to facilitate its navigability.

As discussed in the introduction, the Web and link services compose a hypertext ecosystem (as shown in introduction Figure1.1), where link services facilitate the navigability of the Web. A similar idea for Linked Data of building link services on the Linked Data hypertext system is now investigated. Chapter 5 concluded that only 7.5% of the resources on the Web of Linked Data have across-datasets foreign URIs, and linking patterns of these datasets are monotonous follow one of two patterns: resources are one-to-one parallel linked or many-to-one linked. We believe that by building link services as a part of the Linked Data hypertext system can enrich the linkage and introduce more linking patterns to facilitate links. This Chapter first proposes a set of possible link services, implements exemplars of each type and measures their effectiveness.

6.1 Link Services

As discussed Chapter 2 section 2.6, a link service (originally used by Sun's Link Service) was defined as a protocol for an open hypertext system to enable a

"closed" hypertext system to be able to link to objects. This was a rather limited definition for their own system. However, this vision enabled the monolithic hypertext system to have a more open architecture, where "link features", for example, a link search engine, can be decoupled from the system). After the invention of the Web, its REST-based architecture (as discussed in section) enables many client-server styles service to be built on top of the Web [1]. There are many modalities of web based services [2] with different purposes: those that manipulated URIs have resulted in facilitating the navigability of the Web. Here we define a Web based link service from a hypertext system perspective as:

Definition 9. ***Link Service****: A restful service that interact with the hypertext components of a system and function as an extension of the hypertext system.*

Definition 10. ***Link Service of the Web****: A restful service that interacts with the hypertext components of the Web (here is meat the URI and its HTML representation, because the hypertext on the Web are embedded links) and functions as an extension of the Web.*

There are many popular link services that have resulted in facilitating the navigability of the Web. They have been discussed in details in section 2.6.1 and are summarised below:

[1] Although strictly speaking that Fielding defines a resource was a time-varying mapping between URI and representation, this definition is not applicable to the Semantic Web, we will still call these services as restful services, as they are based on the Web architecture and many other restful characteristic remains

[2] Please note that the service we referred here is not directly related to the Web services with, for example, SOAP messages for machines to communicate over the internet. As those Web services has a broad definition, and many of them are essentially an "internet service".

Linkbase Service : a linkbase service that enables third parties to author links between documents by creating a database of links between document (which originally may not have been linked). Given a URI, one can retrieve a list of links that relate to this URI, and therefore can navigate from one document to another. For example, in blog communities such as WordPress, a pingback service [1] can be used for authors to request notification when someone links to one of their documents. This enables authors to keep track of who is linking to, or referring to, their articles.

Composition Service: a service that aggregates a list of hyperlinks based on certain criteria or creates links to enable navigation between all these web pages. For example, the Amazon recommendation service and Google Ads are services that aggregate lists of relevant hyperlinks and by following these links, users can navigate from one site to another.

Reasoning Service: based on given input, this service applies certain reasoning rules and generates dynamic links. For example, a travel booking system is a simple reasoning service, where a user can ask for a list of available tickets for a particular time interval.

Link Injection Service: this service advocates the idea of decoupling links and nodes and enabling the reuse of links on the Web by injecting links to Web resources where links were not originally available. Liquid information

[1] http://goo.gl/Hn1KI

link injection [1] as an example, injects links into the Web contents.

As we discussed in the introduction, all these services interact with the Web to facilitate its navigability, forms a hypertext ecosystem, as it is shown in Figure 6.1.

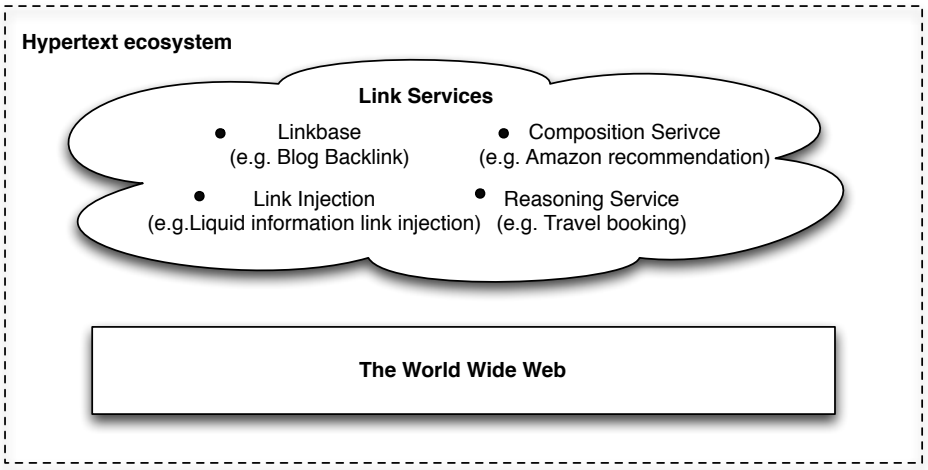


Figure 6.1: The Hypertext Ecosystem of the Web

From the Web to Linked Data, one of the fundamental changes is that a URI identifies data and data are published in RDF documents. All this structured information will enable improved reusability of links to build better services. A typical link service for the Web manipulates URIs and their representation the HTML documents.

Here we propose a list of link services, inspired by the Link service for the Web summarised above. These link services were purposely designed to manipulate URIs and their RDF descriptions and so improve the navigability of the Web of

[1] http://liquid.info/

Linked Data. As displayed in Figure 6.2 those link services interact with the Web of Linked Data and to form a hypertext ecosystem.

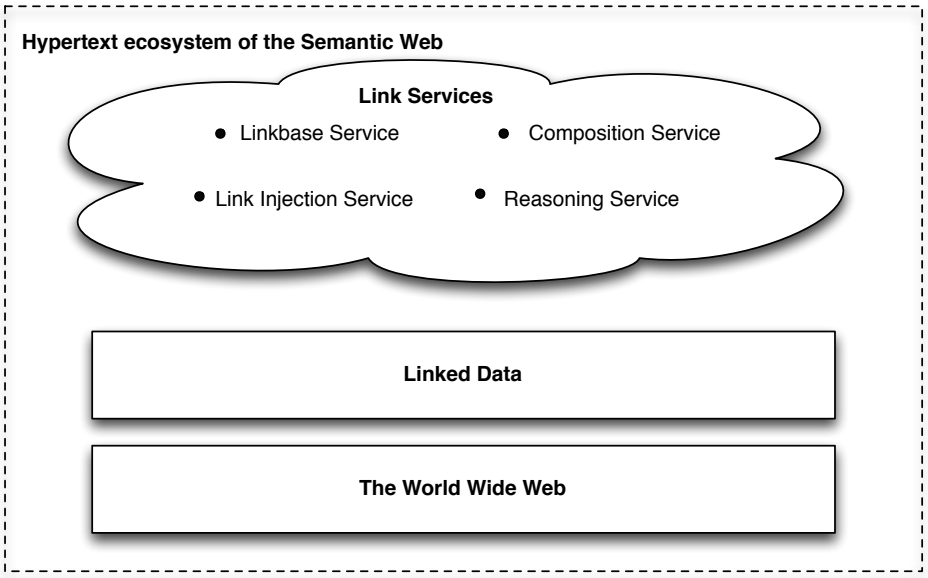


Figure 6.2: The Hypertext Ecosystem of the Semantic Web

It worth mentioning that the restful services can be easily programmed. A popular online service called IFTTT(IfThisThenThat) [1] enables end users to program to RESTful services. For example, a user can program the Web API provided by a weather service and gmail service that if weather API says it is a rainy day, then send the client an email to remind of taking umbrella. This idea can apply to all RESTful link services as well. A service that helps end user to navigate the Web can be created. For example, programming a back link service provide by Google and a specialised link injection service that we can inject all back links of a embedded links on the Web page. In this way, the Web is actually made bidirectionally navigable. Chapter 3 Figure 3.3 demonstrate that one major dif-

[1] https://ifttt.com/

ference between the Web and the Semantic Web is that the latter will be used by agents to perform intelligent tasks for humans. With a programable link service in mind, two primitive link services (utility services) were purposely designed that enable the navigability of the Web of Linked Data. This set of link services that can be built to improve the navigability of the Web of Linked Data is now presented.

6.1.1 Two basic link service enable the navigability of the Web of Linked Data

Chapter 4 studied URI dereferencing processes. In general, two simple steps are required for an agent to navigate along the Web of Linked Data: 1) dereferencing the URI and obtaining the RDF; 2) processing the RDF to extract URIs and dereferencing these URIs. This section first defines two basic services that can be used to navigate the Web of Linked Data.

6.1.1.1 Resolution Service

We presented a HTTP URI dereferencing model in Chapter 4 section 4.3. As discussed, it is a rather generic algorithm, which requires all semantic web agents to implement when dereferencing a URI. A service that avoid this process is proposed here. An agent provides an input URI to the service, and obtain the

RDF about it straight away without need to track the dereferencing process. Figure 6.3 describes this resolution service - for a given URI, the service simply provides its RDF description.

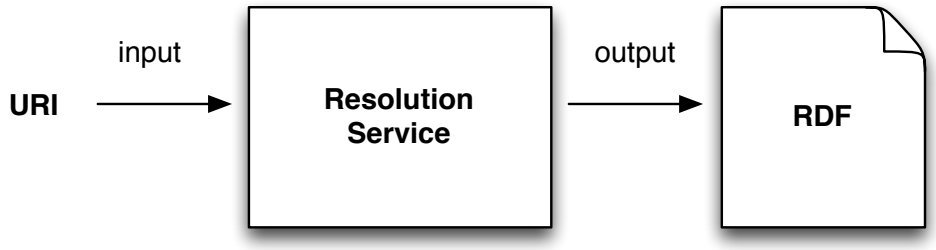


Figure 6.3: The resolution service

The resolution service is an implementation of the dereferencing algorithm and is provided as a service. Figure 6.4 shows an example of such service, the agent can requests `http://www.geonames.org/ontology` to the service and obtains a RDF straight away without need to track the dereferencing process. The Hyperthing validator presented in section 4.4 implements a dereferencing algorithm, and can be extended as a resolution service by returning a RDF document. For the current example, the client requests the service by giving the URI as an input to the service `http://www.hyperthing.org/service/http://www.geonames.org/ontology`, and the service send the request with the HTTP header "rdf+xml" on behalf of the agent directly to the requested server to obtain the relevant RDF document and send it back to the agent.

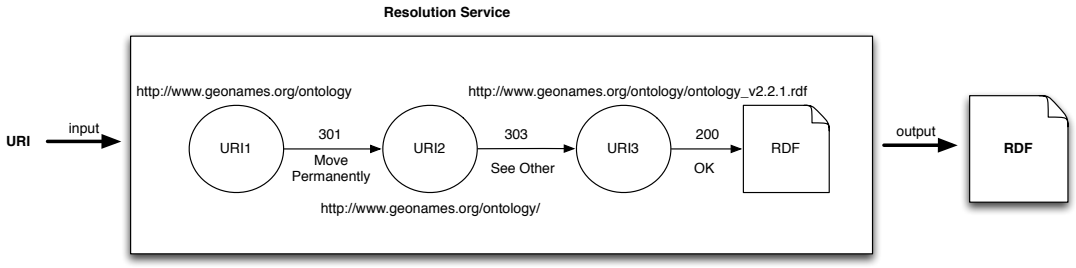


Figure 6.4: Example of the resolution service

6.1.1.2 Link Extraction Service

The link extraction service, as it is shown in Figure 6.5, is a simple service that take a RDF document as an input and extracts a list of URIs. An agent can simply call (program) this services together with the resolution service one after another to navigate along the Web of Linked Data. By default the link extract service will only extract the URIs which are either a subject or object in RDF, as in most of cases they are the client's primary interests. Unless specified, the predicate URI is filtered by this service.

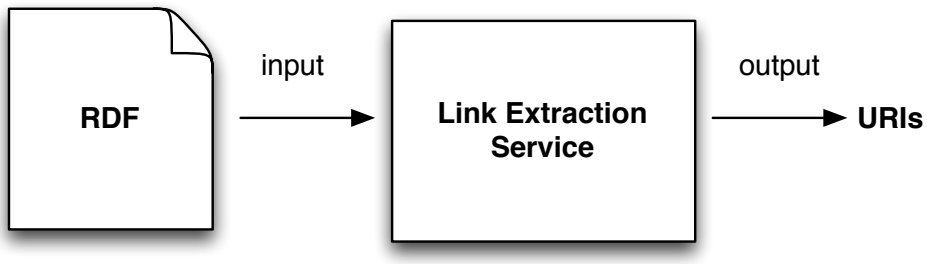


Figure 6.5: The link extraction service

As mentioned in Chapter 4, the dereference algorithm is capable of differentiating a URI that identifies a real word object and a URI that identifies a document.

With this capability, when browsing the data, these URIs can be processed differently, as only those real world object URIs need to be sent to the link services. Therefore, a more sophisticated service can provide options to enable the return two type of URIs separately and all together (e.g. with option parameter in the service: objURIs, docURIs, allURIs). Agents can utilise them based on their needs.

The above two services are basic utility services that enable the agent to navigate along the Web of Linked Data. The following describes in detail the Link services that improve the navigability of the Web of Linked Data as shown in Figure 6.2 with exemplars.

6.1.2 Linkbase Service

Similar to the linkbase service for the Web, linkbase services can be built to host link relations between datasets, especially for those data which were not originally linked. Figure 6.6 shows such a service: when a client requests a URI, the service generates an RDF document which contains links to the requested resource. For example, on the Linked Data cloud, there are no links between any instance of New York Time datasets and BBC datasets. When a user dereferences a URI about a celebrity in a New York Time datasets, they simply only obtain a local description of this person. A third party discovers that there are many data about celebrities in both the New York Times datasets and the BBC datasets and these

are related, hence, creates links between all these instances and provides them as a service. When users use the service by providing the requested URI, they discovered additional information on this about celebrity was described in the BBC datasets by following the RDF triples returned by the service.

Two examples are presented of a linkbase service: the backlink service and coreference service.

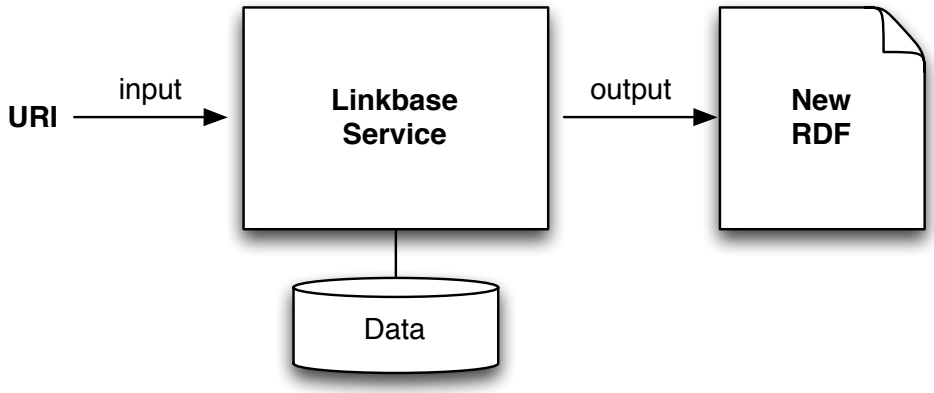


Figure 6.6: Linkbase services

6.1.2.1 Backlink Service

In the Chapter 5 section 5.1.1, we defined the meaning of the foreign URI. For example, a link enabling navigability between two RDF document, $RDF_1 \Longrightarrow RDF_2$, is a foreign URI. We call the reverse link that enable the navigability $RDF_2 \Longrightarrow RDF_1$ a backlink. A backlink will enable the bidirectional navigability between two RDF documents.

Figure 6.7 shows a problem of a lack of backlinks. When dereferencing URI nhs:TrustBarnsley from the NHS dataset, an RDF graph is obtained which states that nhs:TrustBarnsley is based near os: Barnsley, which is a URI from the Ordinance Survey (OS) dataset. When the os:Barnsley URI is dereferenced, the RDF graph obtained has no information about nhs:TrustBarnsley. Therefore, the nhs:TrustBarnsley link can be followed to navigate to the OS dataset but not vice versa.

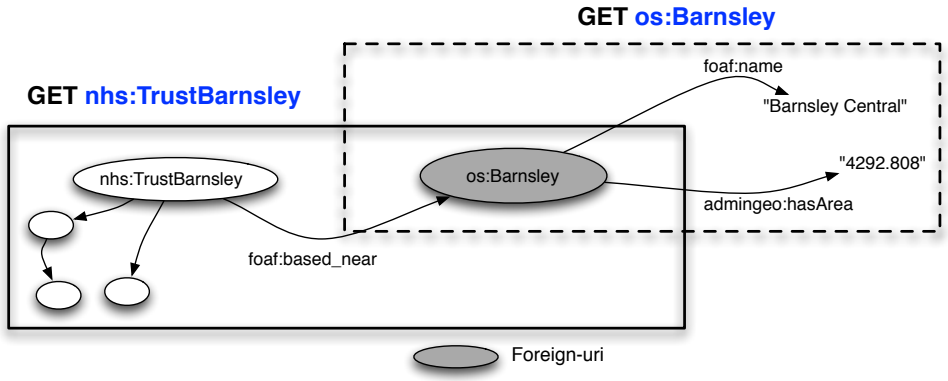


Figure 6.7: Navigation Problem when missing backlinks

For the above case, if there were mechanisms to notify the OS dataset when someone was making a reference to their RDF, data curators would be able to update their records and create a triple with a reverse predicate, enabling the reciprocal navigability of OS and NHS. However, in reality the dataset curator may also decide not to host other datasets information for many reasons, such as maintenance costs, copyrights, the reliability of the data source or other social and political issues.

In the enAKTing project, we developed a lightweight backlink service [1] for Public

[1] http://backlinks.psi.enAKTing.org/resource/rdf/

Sector Information (PSI) datasets [137]. The services monitor all PSI datasets, identify foreign URIs and generate backlinks triples. The service provides a RESTful API which accepts requests using simple HTTP GETs with an input URI, and provides a list of backlinked URIs in RDF.

For instance, for the case described in Figure 6.7, the backlink service analysis the datasets and identifies the foreign URIs `os:Barnsley`: `http:/data.ordnancesurvey.co.uk/id/7000000000024753`. Then the service generate the backlink triples using an rdfs:seeAlso predicate in the knowledge base. The client request the backlink service via HTTP by giving the requested URI, such as `http://backlinks.psi.enakting.org/resource/http://doc/http:/data.ordnancesurvey.co.uk/id/7000000000024753`, and the following RDF document is returned from the service as below:

```
os:Barnsley
    rdfs:seeAlso nsh:TrustBarnsley .
```

Figure 6.8 shows the backlinking service output for `os:Barnsley`. As it is highlighted in red, theclient can now follow the URI to discover the NHS trust dataset by following the URI of `nhs:TrustBarnsley`. Other than this, it also provides backlinks to datasets such as populations, educations and so on. The illustrated backlink services also integrated with the coreference service sameas to provide more useful information, which we will be discussed later. More information about the architecture of the backlink service can be found in Appendix 7.1.

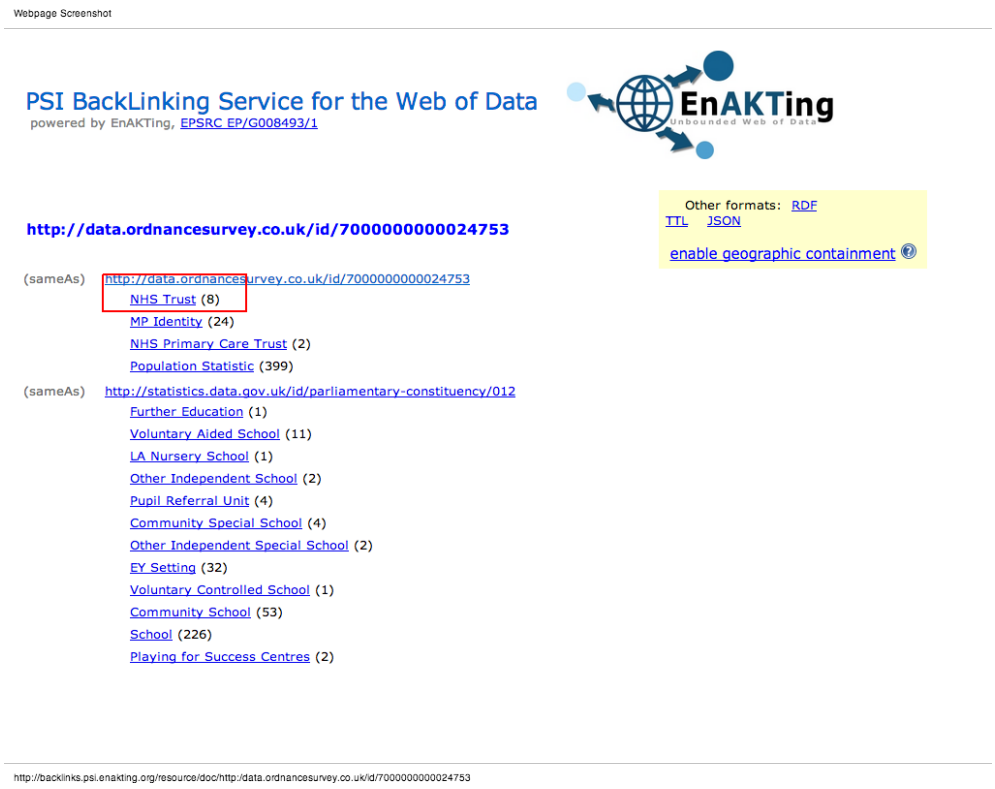


Figure 6.8: Backlinking Service for dbpedia:Barnsley

When implementing the backlink service, ideally, the service curator should use a 'reverse predicate' for each foreign URI triples identified to ensure the completeness of information. For example, if a triple says that URI 'A owns B', we should create a triple says that 'B is owned by A' instead of simply saying 'B is linked to A'. However this increases the difficulties in implementation, as each case needs to be analysed individually. A simplified backlink service can be built by creating a backlink ontology and can make a simple backlink triple assertion without worrying about reverse predicates. For example, URI 'B has a backlink to A' using the 'hasBacklink' predicate. In some cases, the backlink services may be considered too expensive to be deployed, especially when a single URI has been widely referenced by other datasets. An example, when building a GPS track of

a person, millions of latitude and longitude data will be generated as this person moves along, so it is reasonable to have a pointer from the track of each point to the person, but it is impractical to append all these data to this person's FOAF file.

6.1.2.2 Co-reference Service

The co-reference service sameas.org [69] is essentially a linkbase service which hosts third party owl:sameAs relationships of other datasets. Figure 6.9 describes a co-reference scenario in which there is no link between the two resources. Two RDF graphs - a mortality data instance and DBpedia data instance - both have a link to a location, Portsmouth. The mortality:Portsmouth and dbpedia:Portsmouth instances are essentially the same concepts, but defined by different datasets. Although these data have relevant content, this cannot be discovered because there is no link between two datasets. The sameas

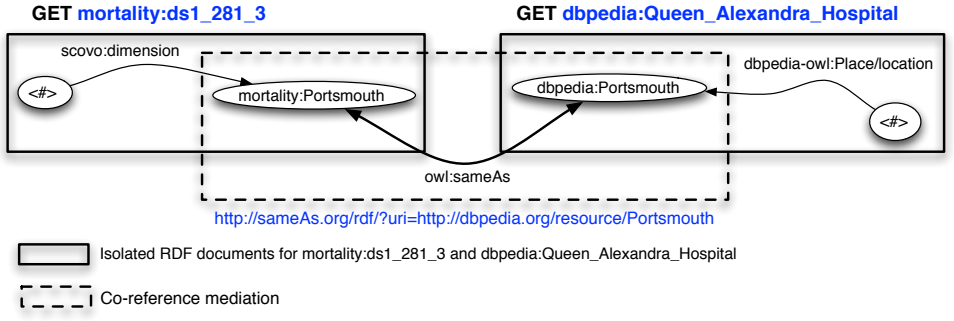


Figure 6.9: Co-reference Navigation Scenario

service acts as a third-party database which hosts link relationships between

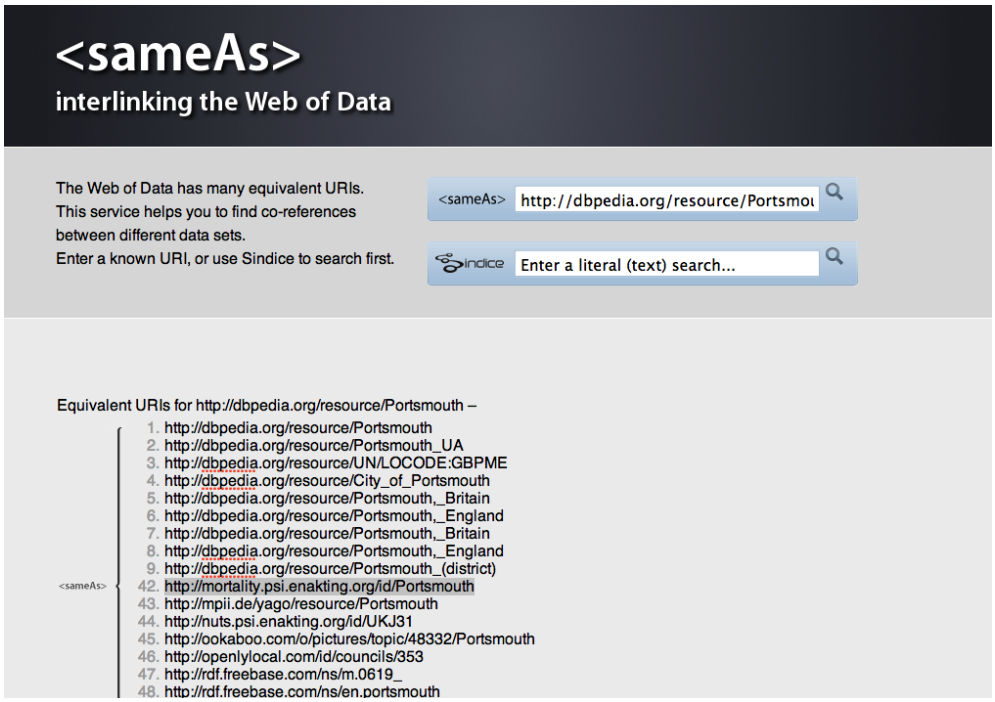


Figure 6.10: Co-reference Service sameas.org

other datasets. For the example above, sameas.org makes an assertion that these two instances are the same. Therefore, when navigating the RDF, using the sameas service enables navigation between the two graphs as well as the discovery of other resources pertaining to the same concepts. Figure 6.10 shows the results returned from the sameas service, where the user can follow `http://mortality.psi.enakting.org/id/Portsmouth` to discover the PSI mortality dataset.

The sameas.org data are generated manually or by simple algorithms. For example, for explicit relationships between two pieces of data in different datasets that both refer to the city name 'London', a string matching algorithm can be used to create links between two identical entities. However, for complicated cases,

such as to determine if 'City of London' is the same as the 'Greater London', or if 'London Airport' is in Canada or UK need to be examined by administrators.

There has been some debates that the many 'sameas' entities are not exactly identical [81]. Halpin summarised and categorised them as follows:

Identical but referentially opaque: things are identical, and the two names do identify the same thing, but all the properties ascribed to one name are not necessarily appropriate for the other, so their names can not be substituted.

Identity as Claims: One could attempt to avoid the entire problem by simply treating all statements of identity as claims, where the statement of identity is not necessarily true, but only stated by a particular agent.

Matching: A strong similarity relationship called matching where different things share enough properties enough to substitute for each other, at least for some purposes.

Similar: Two different things share some but not all properties in their given incomplete descriptions.

Related: Two different things share no properties in common in a given description but are nonetheless closely aligned in some fashion.

Indeed, different data publishers may have different understanding of the data, and hence create different relations between data. Therefore, a coreference service essentially only hosts the third party understanding of the relationship between data and the quality of the service is based on the quality of data created.

6.1.3 Reasoning Service

The reasoning service, as shown in Figure 6.21, is a service that takes an RDF document as an input, applies any reasoning rules and generates a new RDF document as an output. The reasoning service can help to generate links between resources which are 'logically' related. Taking the example as shown in Figure 6.12, it assumes that a user would like to discover data about the county of Hampshire. Although Hampshire contains Winchester, Eastleigh and Fareham, due to the missing link, the user would not be able to access these data.

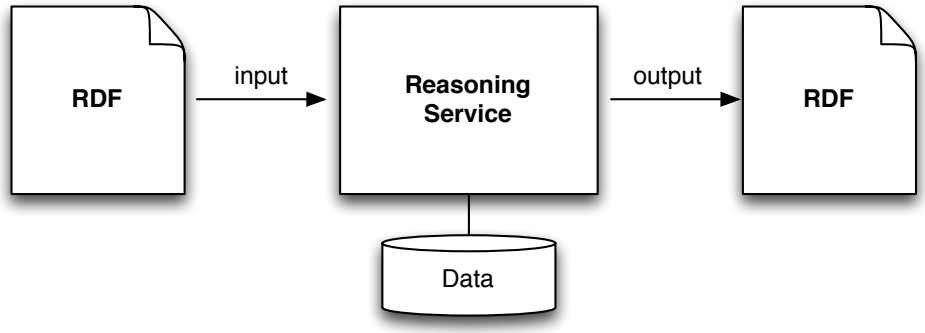


Figure 6.11: The reasoning service

We developed a geo-reasoning service [48] as an example to facilitate navigation

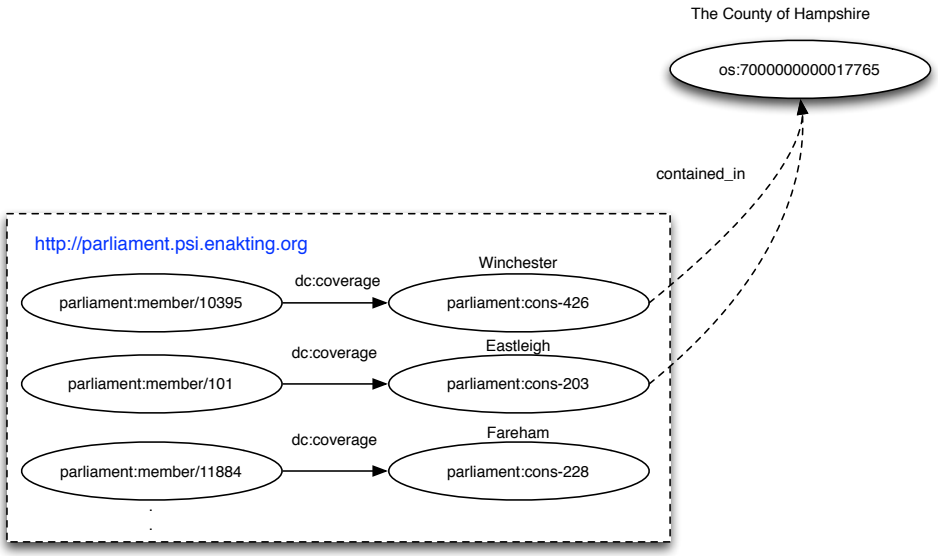


Figure 6.12: Resource irretrievable via geographical gap

between location-based data. The service is designed for querying the UK territory structure. The Ordnance Survey (OS) provides an ontology[1] and an RDF dump about spatial relations between UK regions. The triples from OS have been parsed and only the relation of physical containments have been retained, normalised and completed with the inverse relations in a separate knowledge base. The service presented here manages the spatial representation and reasoning by using the region connection calculus (RCC) as it is shown in Figure 6.13. For the sake of simplicity and efficiency[2], the service only manages the non-tangential proper part (NTPP), and the relative inverse, NTPPi relations. The knowledge extracted from the OS data set has been then normalised in terms of an internal ontology that represent qualitative spatial relations.

[1] http://www.ordnancesurvey.co.uk/oswebsite/ontology/SpatialRelations/v0.2/SpatialRelations.owl

[2] In this way the data to manage has been drastically reduced in order to provide a very focused service.

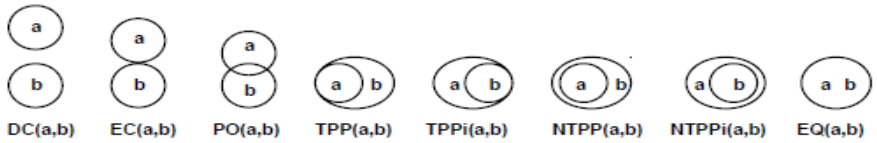


Figure 6.13: RCC Eight Jointly Exhaustive and Pariwise Disjoint Relations

A simple example of how the normalised triples from the OS ontology are used in coupling with a co-reference service for bridging the navigational gap for different data sets is depicted in Figure 6.14. In the figure it is possible to see that a single statement from OS describing the fact that the County of Hampshire **contains** Fareham and Winchester[1]:

```
os:7...17765 os:contains os:7...25157.
os:7...17765 os:contains os:7...25128.
```

This has been translated into an internal representation containing both relations: part, and part_of, as shown below:

```
os:7...17765 geoservice:part    os:7...25157.
os:7...25157 geoservice:part_of os:7...17765 .
os:7...17765 geoservice:part    os:7...25128.
os:7...25128 geoservice:part_of os:7...17765 .
```

The containment relations so normalised (see central part of Figure 6.14) are then internally stored in the system and queried for serving users' requests.

[1]OS URIs are shortened, the trail of '0' are replaced by '...'.

The normalised containment relations are integrated with the information provided by the co-reference system that allows bridging different data sources, both in the input phase (i.e. where the input URI must be translated into its OS equivalent, see top part of Figure 6.14) and the output phase (i.e. when the results must be translated into a target data set provided by the user, see bottom part of Figure 6.14). The co-reference service used is the http://sameas.org service as mentioned previously. The relevant bundles have been retrieved from the service and cached for performance. It is important to note that, in order to choose the required quality of service, one could opt for using one co-reference service instead of another. The functionality provided is transparent from the provenance of the co-reference bundles.

Exploiting co-reference services and OS ontology, it is therefore possible to infer containment relations between resources from different data sets. For example:

```
dbpedia:Hampshire owl:sameAs os:7...17765
AND
os:7...17765 geoservice:part os:7...25128
AND
os:7...25128 owl:sameAs dbpedia:Winchester
```

$\Longrightarrow$

```
dbpedia:Hampshire geoservice:part dbpedia:Winchester
```

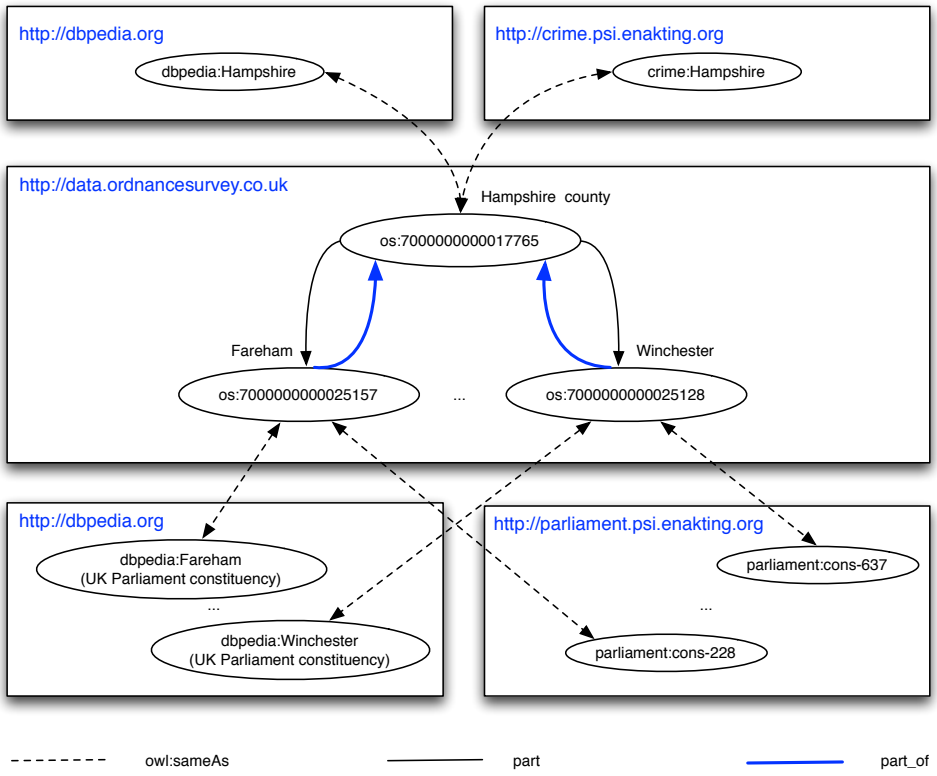


Figure 6.14: The architecture of reasoning service and interaction with co-reference system

For a location URI, the services will provide a list of geographical regions either containing the location or contained by the location. For example, when querying a URI about the city Winchester, the service will return a list of location URIs such as its county (Hampshire), or a list of constituent locations contained in Winchester. This service will enable the user to group or slice location-based data at different ways, thus enabling navigation at different granularities. The service can be accessed via `http://geoservice.psi.enAKTing.org/{command}/{dictionary}/{format}/{URI}`. The command option is either 'contains' or 'container' of the location, where the dictionary option is essentially a filter that enables the user to select only results from certain datasets. The

service supports reasoning from DBpedia, Ordnance Survey, UK National Statistics, Geonames, PSI, OpenCYC, and the Openly Local. More information about the reasoning service can be found in Appendix 7.2.

The reasoning service can be implemented on precomputed data, or it can be processed on the fly. In order to ensure its extensibility, the reasoning component can be implemented by using the Semantic Web Rule Languages, so that the reasoning components can be reused and enhanced easily.

6.1.4 Composition Service

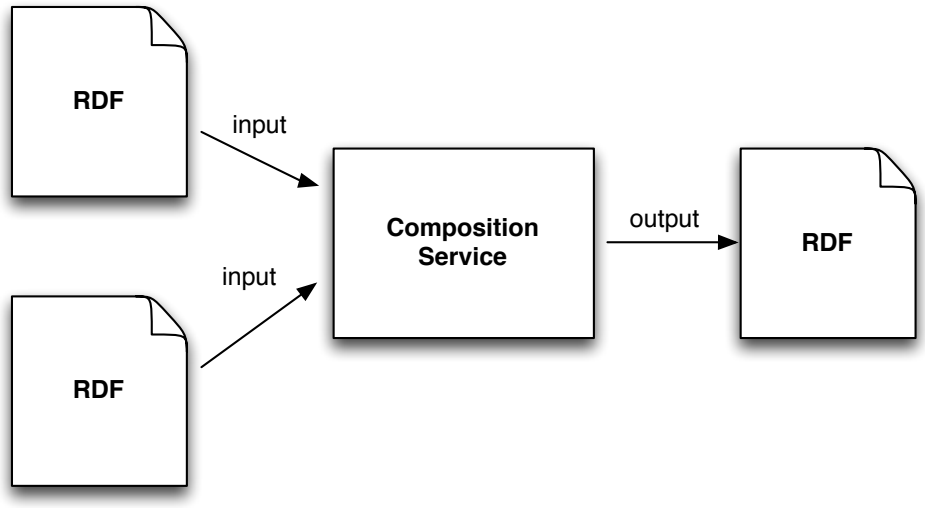


Figure 6.15: The Composition Service

On the Linked Data cloud, there could be many related RDF documents that describe a given resource. In terms of navigability, it can be assumed that each RDF would contains a unique across dataset foreign URI link to another datasets;

following the links in each RDF, will only navigate to one of the foreign datasets referenced. However, if all RDFs can be aggregated into one file, the user can then navigate to all three datasets instantly. For example, there is a URI about a city called Southampton in the New York Times datasets and its RDF description links to the Geo Name datasets. This URI is also referenced in the DBpedia datasets in the RDF description about the city of Southampton. Following the links in the original RDF document in the New York Times datasets will not discover the DBpedia datasets. However, a composition service, which discovers such a relationship and compose them into a single and bigger RDF document, will help to discover all these datasets. We can then discover all these datasets. By doing so, a RDF is also provided with (most likely) more elaborated ('global context') information about the city of Southampton.

Figure 6.15 describes the RDF composition service that was implemented (as a component of the Link injection service will be described later). The service takes multiple RDF documents from different service and aggregate them into a single RDF. For the example about the city Southampton, looking up the backlink service, co-reference services and reasoning service, we can obtains more RDF about the requested resources, and this service then combines these RDF documents together with the original RDF document from the New York Times dataset into a single RDF documents and returns to the clients.

A composition service, we should avoid duplicated triples and redundant triples. The drawback of a composition service is that the size of RDF can be built up very quickly. The more triples, the more ontologies are needed to be processed

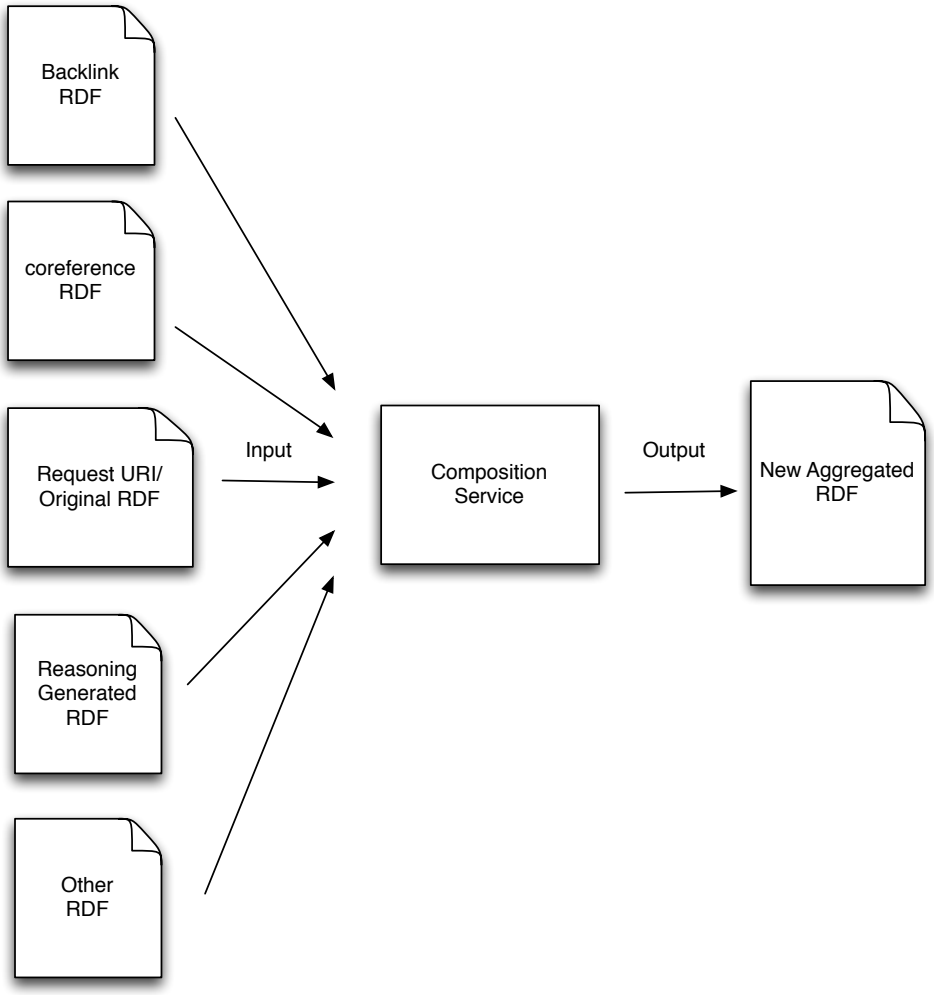


Figure 6.16: Link Composition Service

to utilise the data.

6.1.5 Link Injection Service

The Link Injection Service on the Web, as discussed in section 2.6.1, is a service that injects the links into Web documents where links were not originally available. On the Linked Data, all URIs are grouped in a triple form in the RDF documents. Here, a similar idea is used to augmenting existing resources by building a link injection service to inject triples into RDF documents. Figure 6.17 describes a link injection service which essentially orchestrates three services: a URI extraction service, a linkbase service and a composition service. The service functions as follows:

1. The service takes the RDF document as an input, and extract all URIs from it.

2. Linkbase takes all these URIs as input and retrieves the third party RDF documents about these URIs.

3. The composition service takes the resulting RDF documents and the original RDF document as inputs, combines them into a big RDF document and returns them to the client. ...

For example, for the following input RDF document:

```
vendor1:productX     vendor1:sellBy  vendor1: John.
```

```
vendor1:productX dc:title "iPhone 5s" .
vendor1:productX retail:price  "549.00" .
vendor1:productX geo:City     geo:London
```

the service will extract three URIs from the RDF: vendor1:productX; vendor1:John; and geo:London. The service then sends a request to the Linkbase service to get more information about these URIs. It provides additional RDF documents about John (see below new triple 1), latitude and longitude of the city (see below new triple 2 and 3). These are then combined them into a bigger RDF document and returned to the client as below:

```
vendor1:productX     vendor1:sellBy  vendor1: John.
vendor1:productX dc:title "iPhone 5s" .
vendor1:productX retail:price  "549.00" .
vendor1:productX geo:City     geo:London .
(new triple 1) vendor1: John         foaf:email    john@gmail.com .
(new triple 2) geo:London          geo:lat         "50.5072" .
(new triple 3) geo:London        geo:long       "0.1275" .
```

Note that the above link service is a orchestrated service and its essential components are the URI extraction service and composition service, while the Linkbase service can be replaced by other services such as reasoning service or resolution service. The URI extraction service and composition service can be recursively reused. Figure 6.18 illustrates a link injection service with resolution service components, the system recursively invoking itself. The system will halt when there

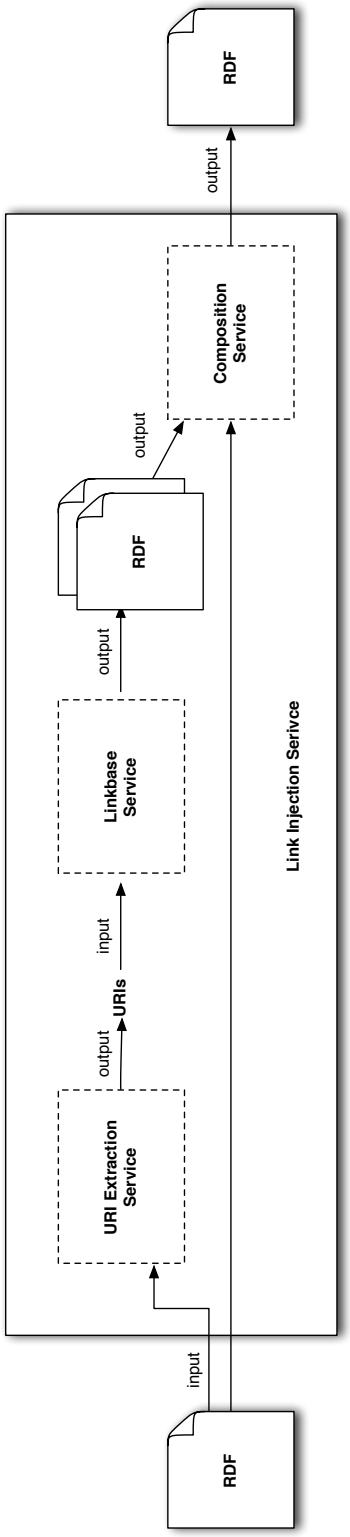


Figure 6.17: Link Injection Service for Linked Data

is no links are found. The service assembles a breadth-first search on the RDF graph.

6.2 Link Service Proxy

Using link services, as discussed, can enhance the navigability of the resource. However, in order to use the link service, it is almost always requires multiple requests from the clients (e.g. a web browser). The reason for this is that the clients will request a RDF description of a resource, then want to obtain more information by using the link service. To make the service transparent to the clients and to enable clients to orchestrate link services easily from the client side, we propose to build a link service proxy which can be used from the client side together with any Linked Data applications.We call this proxy a Hyperthing Proxy. This idea is inspired by the Distributed Link Service (DLS) for the Web. As it is discussed in Chapter 2 section 2.6.1, the DSL is a service that works in conjunction with existing Web resources to support an additional underlying link service. It provide an interfaceless proxy link service which makes the link service transparent to its users by embedding it in the Web's document transport system, compiling links into documents as they are delivered to the browser by a specially adapted Web proxy server. In a similar way, the Hyperthing Proxy designed specifically for the link services for Linked Data will acts as a transparent proxy sitting in the between clients and servers. When users request a URI, link services are consolidated, obtain relevant information and then combine it with

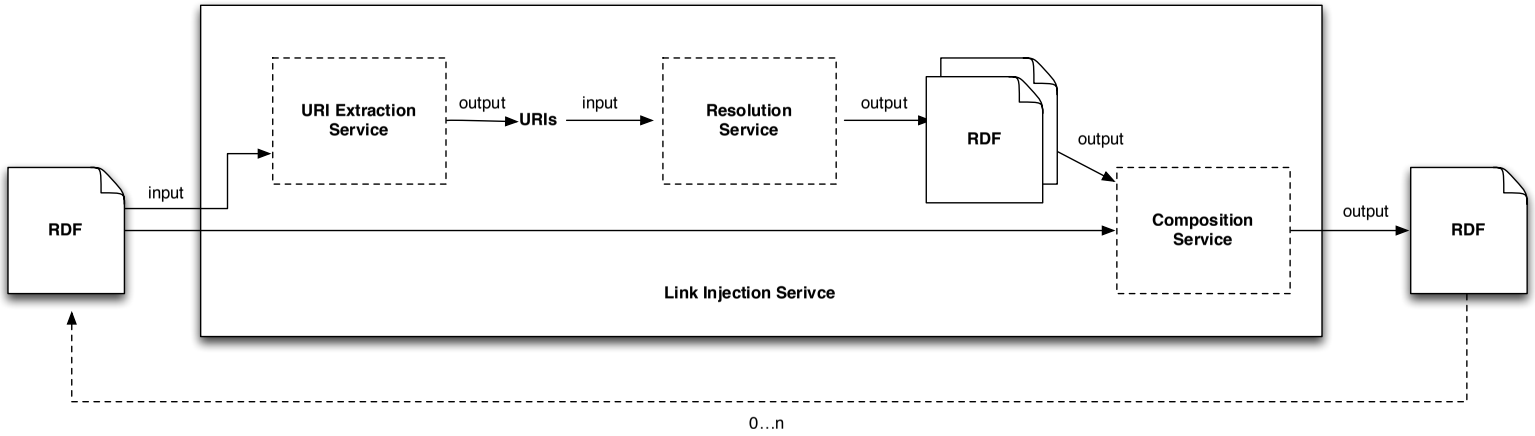


Figure 6.18: Link Injection Service with resolution service components recursively invoke itself to generate a new RDF

the original resource and send everything to the clients. The clients can follow the link and navigate to the other dataset instantly.

Figure 6.19 describes the architecture of the Hyperthing Proxy. The architecture of the system is similar to that of the DLS previously mentioned. The Proxy itself is a customisable. A settings text file is provided where clients can program and orchestrate a list of link services by providing the link service URI in an order. As depicted in the Figure 6.19 that by default, there is a local resolution service which performs the client original request.

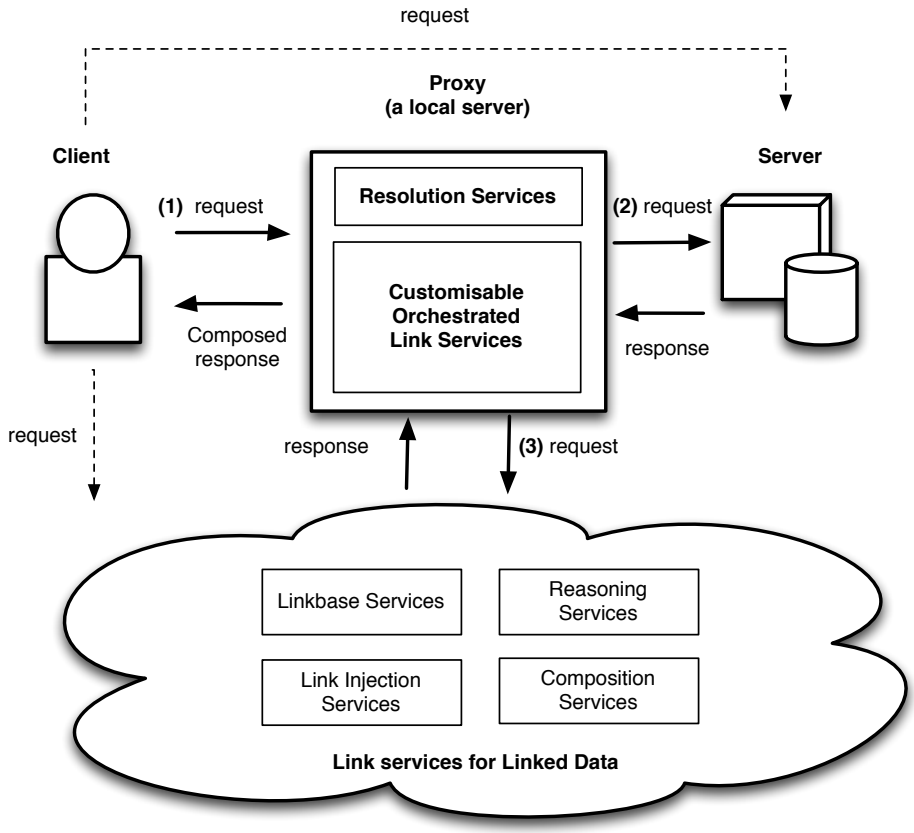


Figure 6.19: Hyperthing Link Service Proxy

The client sends the request for a desired resource URI to the proxy instead of the server directly (see request one). The link proxy then perform a multiple requests, first it sends the client's original request to the server (see request two), then it sends multiple requests to the link services specified in the proxy setting files.

For example, for the services such as co-reference service (http://sameas .org/rdf?uri=), geo-reasoning services (http://geoservice.psi.enAKTing.org/contains/os/) and a composition service, one can program the setting files by giving the URIs of the service in the following order:

```
resolution service URI; (the default link service);
co-reference service URI;
geo-reasoning service URI;
composition service URI;
```

This means that the proxy will send the client's original request to the resolution service. The proxy will also send request to the co-reference service geo-reasoning service, then compose them together into a bigger RDF and return to the clients.

To demonstrate the link proxy, we implemented a simple orchestration mechanism, which sends requests to a list of services one after another. However this component can be extended by defining some simple syntax to enable the link services to be fully programable. For example, in the previous section, Figure

6.18 described a link injection service with resolution service components that recessively invoked itself to generate a new RDF. The syntax can simply be defined in a similar way to the Lisp programming language [106], where parentheses define the order of the service and the syntax '*' means recursive.

```
resolution service URI;
(((URI extraction service URI) resolution service URI)
 composition service URI)* ;
```

As mentioned before, this service setting assembles a breadth-first search on the RDF graph and the program will halt when there is no more links are retrieved. Doing so also enables the clients to easily program a Linked Data crawler.

One important default components of Hyperthing proxy is the resolution service components (as depicted in section 6.1.1.1 Figure 6.3). To demonstrate the idea of customisable orchestrated link service, we implemented a local Link Injection service, exactly as depicted in section 6.1.5 Figure 6.17. Please note that the proxy essentially function as a local server, where client request the local server in the similar way of a remote service server. Here a co-reference services is used as Linkbase services to demonstrate how the system operates. The proxy is customised by the following list of services:

```
resolution service;
URI extraction service;
co-reference service;
```

```
composition service;
```

For an given input URI, the proxy orchestrates the service in a following order:

1. The service takes a URI as input, calls the resolution service, and obtains a RDF description.

2. The URI extraction service extracts all URIs to be processed. As the service is able to differentiate objURI and docURI, all docURI are filtered out, only processing the objURIs.

3. A list of objURIs is sent the co-reference service, obtaining their co-reference RDF documents.

4. The composition service takes all the resulting RDF documents and the original RDF document as inputs, combines them into a big RDF document, and returns it to the client. ...

The following evaluation section demonstrates how the service improves a URI in the New York Times Datasets and also evaluates how services improve the navigability of the Web of Linked Data.

6.3 Evaluation of Link Service for improving the navigability of Linked Data

Chapter 5, we concluded that there 7.5% resources on the Linked Data cloud has across datasets foreign URIs, which is relatively low and the datasets exhibit two extreme linking patterns; either the majority of data instances are one-to-one parallel linked from one dataset to another or the majority of the data instances from one dataset only link to a few data instances of another dataset. This Chapter has explored the Link Service as a feature extension to built on top of Linked Data to improve its navigability. This section will evaluate its effectiveness.

6.3.1 Methodology

The Linked Service Proxy will be evaluated using four services: PSI backlink service (http://backlinks.psi.enAKTing.org/resource/rdf/), co-reference service (http://sameas .org/rdf?uri=) and geo-reasoning services (http://geoservice.psi.enAKTing.org/contains/os/) and a local link injection service that injects all these returned RDF into the original requested RDF document. In order to measure the improvement of the navigability of the Web of Linked Data as a whole, we firstly measured how well these services improved navigability of these linked datasets individually (e.g. how much across dataset foreign URIs are added to the Web of Data). We then analyzed its overall improvement.

To demonstrate how to utilise the proxy to improve the navigability of the Web of Linked Data, a Web browser as an application to test the functionality of the Link Service Proxy. The reason a Web browser was chosen as an agent is that it is a generic application to the Web hypertext system. To date, most of Linked Data browser is merely an extension to the Web browser (e.g. Tabulator [19]) and the Web browser is sufficient to be used together with the Hyperthing proxy to browse the Web of Linked Data. In particular, the system will be tested on a personal laptop, with Mac OS X (version 10.8.5) and Google's Chrome Web browser (version 30.0.1599.101). The first demonstration is of browse the Web of Linked Data in a Web browser without the Hyperthing proxy, and then its improvement by sending the HTTP request via the proxy.

6.3.2 Results

This section first present the evaluation of the above mentioned first three services individually. The link injection service is only used locally to inject links which does not generate any accoss-datasets foreign URIs, therefore excluded in this section and will be discussed in the next section. We then present the overall improvement and how it interacts and functions with the Web browser.

6.3.2.1 The across-datasets foreign URIs generated by the Backlink Service

The Backlink service, as discussed in section 6.1.2.1, was implemented with the aim of improving the linkage in the Public Sector Information in the enAKTing project,t therefore the following datasets were chosen to generate backlinks.

- http://co2emission.psi.enakting.org: Statistical data about CO2 in the UK, the emissions are hierarchically typed by cause of emission and linked to the region where it was measure.

- http://energy.psi.enakting.org: This dataset represents the energy consumption for the road network in the UK between 2002 and 2007.

- http://population.psi.enakting.org: Statistical data about population in the UK segmented by, age, gender and location between 2001 and 2007.

- nhs.psi.enakting.org: Statistical data about the number of patients waiting for a first outpatient appointment following a GP referral. NHS names that can be connected to its locations, year of validity and weeks waited.

- mortality.psi.enakting.org: Statistical data about mortality. This data set is geographically segmented by UK regions and gender.

- crime.psi.enakting.org: Crime statistics provided by the UK Home Office[1]. These statistics are divided by region and type of crime.

- parliament.psi.enakting.org: Information about the UK Members of Parliament, their expenses, affiliations and votes. The original sources of this data are `theyworkforyou.com` and `publicwhip.org.uk`.

- data.ordnancesurvey.co.uk: The Ordnance Survey is the authoritative publisher of geographic information in the UK. This data set contains several interconnected hierarchies of types of regions in the UK such as Constituencies, Wards and Boroughs. These hierarchies are interlinked with spatial relations.

- *.data.gov.uk: Under `data.gov.uk` the UK is publishing different data sets using Semantic Web technologies. Two datasets were identified that are ready to be part of this study: education and statistics. The rest of the datasets, at the time of writing this thesis, did not contain links to other datasets.

Table 6.4 describes the linkage improvement of the backlink service, the number of across datasets foreign URIs added to these datasets. The first column of the table represents datasets from where a backlink exists. The datasets in the header represent the origin of the backlinks. The numbers are the occurrences of backlinks for each pair of datasets. As an example, we can say that from

[1] `http://www.homeoffice.gov.uk/` last accessed 20/12/2009

`sws.geonames.org`, it is possible to navigate to an extra 9,345 resources from the `education` dataset. Overall, there are 3,995,178 across datasets foreign URIs are generated in the service.

6.3.2.2 The across-datasets foreign URIs generated by the Geo Reasoning Service

The data of the geo-reasoning service, as mentioned in section 6.1.3, are generated by using the spatial relation ontology provided by Ordnance Survey and its extensions to other datasets via co-references. There are 60 millions of statements about geo graphical containments in the OS datasets that were used as data seed.Table 6.2 represents the number of triples generated by geo-reasoning services in terms of the number of triples having a predicate of either `geoservice:part` or `geoservice:part_of` (the NTPP and NTPPi relations as mentioned in section 6.1.3). This table shows the numbers of triples linking every pair of datasets in the system. For example, the service produced 30,995 geographic containments between *dbpedia* and *mortality.psi.enakting.org*. Overall, there are 242,533,051 across-datasets foreign URIs were added to these datasets.

6.3.2.3 The across-datasets foreign URIs generated by the sameAs co-reference Service

According to an informal interview with the creator of the sameAs co-reference

Table 6.1: Backlink Service: Datasets linkage improvement statistics

	Backlink to							
Backlink from	**crime**	**mortality**	**education**	**parliament**	**nhs**	**energy**	**co2**	**population**
statistics.data.gov.uk	1280	4389	1444196	22830	240	38850	117487	418152
data.ordnancesurvey.co.uk	888	2376	226936	24398	256	20780	62422	421344
linkedgeodata.org	19	54	7036	0	0	540	1580	0
sw.cyc.com	11	9	475	0	0	60	172	0
unlocode.rkbexplorer.com	11	15	950	0	0	120	344	0
rdf.freebase.com	337	279	83094	11337	120	2190	6617	207480
airports.dataincubator.org	1	3	505	0	0	30	90	0
dbpedia.org	511	426	97945	11721	122	3480	10480	213864
www.twine.com	0	3	252	0	0	20780	87	0
www.okkam.org	0	6	475	0	0	30	172	0
guardian.dataincubator.org	0	0	63520	11768	120	0	0	208677
revyu.com	0	3	252	0	0	60	0	0
umbel.org	205	183	16841	360	0	1650	4840	5187
os.rkbexplorer.com	367	324	21613	0	0	2610	8035	0
sws.geonames.org	32	93	9345	0	0	930	2759	0
sw.opencyc.org	1635	1170	75720	30	0	8400	25480	1995
mpii.de	250	255	19775	102	0	2250	6887	798

Table 6.2: Geo Reasoning Service: Datasets linkage improvement statistics

	OS	**dbpedia**	**statistics**	**mortality**	**parliament**	**crime**	**geonames**	**openlylocal**	**opencyc**
OS	**60469910**	1757760	45354078	1035901	1338214	235906	94072	18559453	1106900
dbpedia	1757760	59640	1393322	30995	46077	9570	3035	540619	35250
statistics	45354078	1393322	36179867	813660	1056892	206217	71232	14430773	819965
mortality	1035901	30995	813660	19109	23929	4607	1488	344436	17415
parliament	1338214	46077	1056892	23929	37631	7654	2410	436883	28070
crime	235906	9570	206217	4607	7654	2249	334	82559	4160
geonames	94072	3035	71232	1488	2410	334	224	26427	2475
openlylocal	18559453	540619	14430773	344436	436883	82559	26427	6498462	312120
opencyc	1106900	35250	819965	17415	28070	4160	2475	312120	27975

service, the sameAs data were mostly generated mannually by the administrator or simply provided by other datasets hosting owl:sameas relations.The improvement of linkage between datasets of sameas.org is assessed via http://www.sameas.org/sameAs-domains.csv. The service covers a wide range of datasets, with 476 datasets having more than 10 foreign links in the service. Table 6.4 summarises the improvement in navigability by using these three services. There are 502 datasets covered in total (note that citeseer.rkbexplorer.com and eprints.rkbexplorer.com are treated as different datasets) and it generated a total of 373,323,542 across-datasets foreign URIs.

6.3.2.4 Summary of across-datasets foreign URIs contributed to the Web of Linked Data

Table 6.4 - what is the overlaps of links produced by the different services?

Based on the data generated by all three services, Table 6.4 summarises the total number of across-datasets foreign URIs generated. All link services together contribute a total of 373 million across-datasets foreign URIs to the Web of Linked Data. It worth to clarifying that, although backlink services used the co-reference service to discover more resources to generate more blacklinks, there is no overlaps of links generated between these two services. For example, for URI_1 rdf:sameAs URI_2, it will generate both URIs backlinks: URI_1 rdf:seeAlso URI_x, and URI_2 rdf:seeAlso URI_y. Similarly, the geo-reasoning service used co-reference and back-

domain	No. of links	domain	No. of links
rdf.freebase.com	53960446	rae2008.rkbexplorer.com	63020
dbpedia.org	15778333	ookaboo.com	62247
dbpedialite.org	9888286	stitch.cs.vu.nl	55963
viaf.org	6776523	www.discogs.com	54749
oai.rkbexplorer.com	4606413	southampton.rkbexplorer.com	51187
dblp.rkbexplorer.com	4309171	dewey.info	50732
d-nb.info	3702983	eprints.ecs.soton.ac.uk	49925
mpii.de	3405449	eprints.gla.ac.uk	47568
yago-knowledge.org	2812691	data.open.ac.uk	41171
dotac.rkbexplorer.com	2160523	bibsonomy.org	39722
bnb.data.bl.uk	1997823	chem2bio2rdf.org	39490
data.ordnancesurvey.co.uk	1717841	oro.open.ac.uk	35073
www.uk-postcodes.com	1696035	opendatacommunities.org	34377
acm.rkbexplorer.com	1287235	opus.bath.ac.uk	32122
dblp.l3s.de	1196763	orca.cf.ac.uk	31071
citeseer.rkbexplorer.com	1120803	en.wikipedia.org	30192
eprints.rkbexplorer.com	980531	openei.org	29275
umbel.org	964326	sw.cyc.com	27116
sws.geonames.org	921631	kar.kent.ac.uk	25848
linkedgeodata.org	831809	eprints.lse.ac.uk	25397
sw.opencyc.org	723661	zitgist.com	25387
bio2rdf.org	610093	wrap.warwick.ac.uk	24863
www.w3.org	464803	dbtropes.org	24819
wordnet.rkbexplorer.com	464783	purl.org	24522
dbtune.org	274454	eprints.qut.edu.au	23127
id.loc.gov	254095	data.nytimes.com	20958
ol.dataincubator.org	240235	eprints.lancs.ac.uk	20946
openlibrary.org	240235	www.rdfabout.com	19887
eprints.soton.ac.uk	220836	centaur.reading.ac.uk	19179
data.bibsys.no	213730	data.linkedmdb.org	18444
kisti.rkbexplorer.com	179424	publications.eng.cam.ac.uk	18357
nsf.rkbexplorer.com	145228	species.geospecies.org	17523
www4.wiwiss.fu-berlin.de	118227	id.ndl.go.jp	16964
www.bbc.co.uk	116320	www.uniprot.org	16145
data.europeana.eu	100622	eprints.bournemouth.ac.uk	14746
kulturarvsdata.se	100489	www.zora.uzh.ch	14573
lod.geospecies.org	87804	www.myspace.com	14333
lod.taxonconcept.org	86111	purl.uniprot.org	13260
data.bnf.fr	78775	eprints.kingston.ac.uk	12641
data.linkedct.org	77259	airports.dataincubator.org	12106
periodicals.dataincubator.org	74500	dilettantes.code4lib.org	11375
statistics.data.gov.uk	74136	eprints.hud.ac.uk	11333
rae2001.rkbexplorer.com	69467	nektar.oszk.hu	10027
cordis.rkbexplorer.com	67656		

Table 6.3: Sameas Service: Datasets linkage improvement statistics

links information to discover the data and also used the co-reference service for reasoning. The data seed that triggered of generating geo-reasoning data is the Ordnance Survey to Ordnance Survey data containments, some 60 million of statements. The total number of triples generated was 242 million and these are partially interlinking every pair of data sets. Other than this, there are no duplicated foreign URIs generated from these services, and they all use different semantic predicates. In some cases two datasets may already be linked by their own existing foreign URIs, but regardless of this, the services generating foreign URIs will add more "elaborated" links between resources across these dataset regardless.

As discussed in Chapter 5 section 5.2 that there are a total of 425 million across-datasets foreign URIs and 5,639 million triples (17.9%) contains a URI which identifies a resource. The estimated across-datasets foreign URIs coverage is that 7.5% at most (ignoring the fraction of potential duplicated triples and foreign URIs) are linked to other datasets. Building the Link service on top of Linked Data dramatically increases this rate to the 14% coverage, almost doubling the previous rate.

END

Table 6.4: Navigability Improvement Statistics of Services

	No. of across-datasets foreign URIs
PSI Backlink service	3,995,178
Linkbase co-reference service	126,795,313
Geo reasoning service	242,533,051
All link services	373 million
The Web of Linked Data	425 million

6.3.2.5 Demonstration of the use of Link Service Proxy via the Web browser

The link service proxy is applicable to all Linked Data applications. Any Linked Data application can be enhanced immediately by linking to the link service proxy. All previously mentioned, 373million across-datasets foreign URIs can be utilised easily by on-the-fly request. To demonstrate the use of link service proxy, the behaviour of the Web browser application is compared with and without the link service proxy.

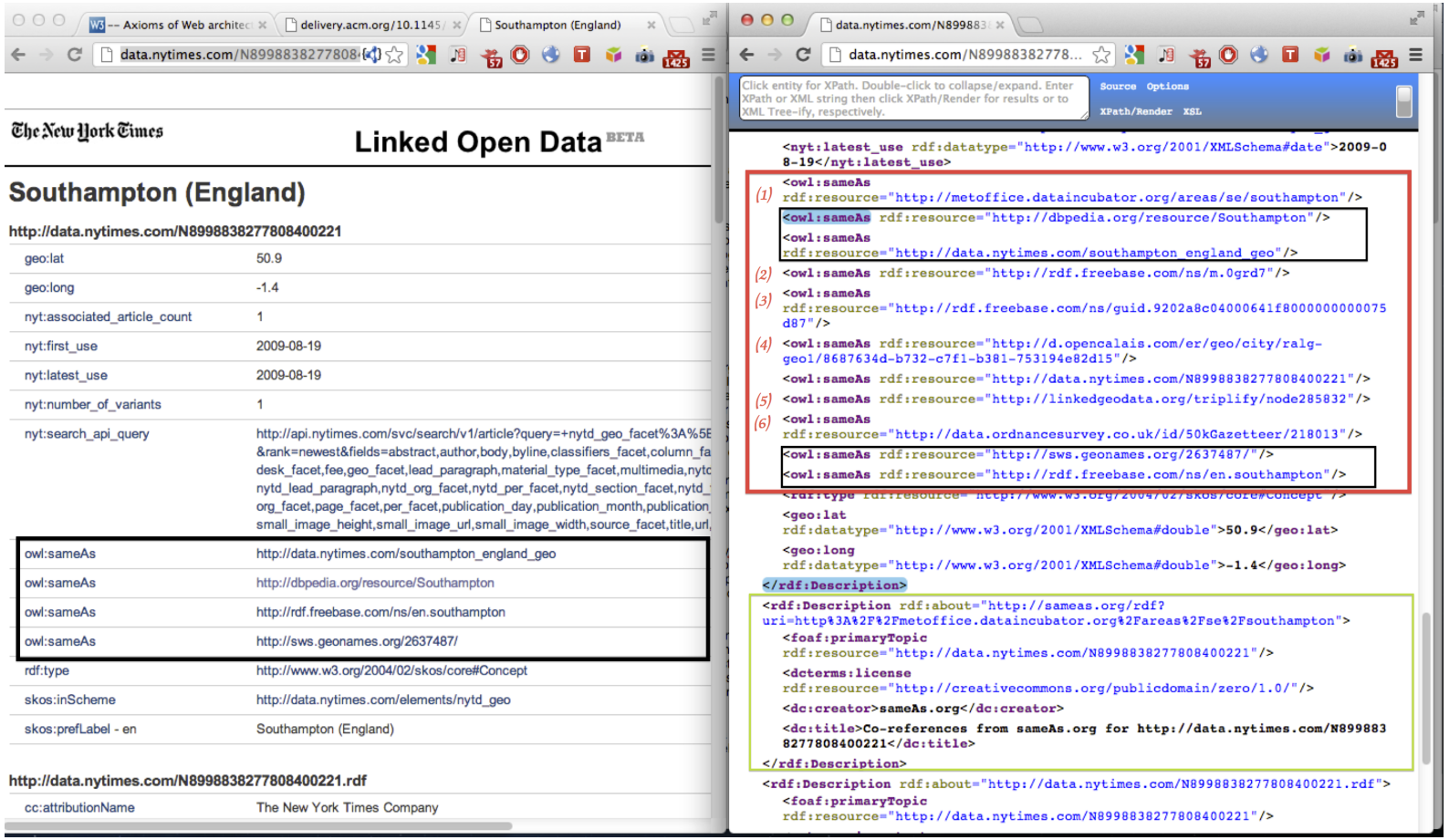
Figure 6.20 illustrates a case when browsing the New York Times datasets for the city of Southampton. In the left browser window, when requesting a URI http://data.nytimes.com/N89988382778084002221, a HTML page is rendered to display the RDF file. In the right browser window, the same request was send via the proxy, a RDF document is rendered in the browser. This is because our proxy sends an http request with header "rdf+xm" instead of the 'text/html'.

The left browser window shows the Southampton RDF description in HTML,

which has four sameas links and three foreign links pointing to DBpedia, Freebase, and Geonames, respectively (as highlighted in black). In the right browser window, when browsing the same URI in the same browser with the Hyperthing proxy turned on - a few more sameas links were injected (as highlighted in red). A total of 6 of triples were injected with the new sameas relations, for example, the very first triple link is to the Met Office data about Southampton. Similarly, the triples four, five and six link to OpenCalais, Linked Geo Data and Ordnance Survey data. Although in the original RDF document, it has relations between Freebase datasets, however there is only a single link, while in the newly injected RDF document, two new relations with instance in the Freebase dataset are created (see triples two and three). As it mentioned in Chapter 5 section 5.3.1 that the New York Time dataset only links to another six datasets, but this particular case, we already enabled the navigability to another four more datasets, and more links to the already related datasets.

In this use of the link service proxy, all triples returned from the sameas.org service were injected. Their original co-reference RDF document contained the service description, creators and copyright information. These triples were injected to the original New York Times Southampton RDF document as well (highlighted in green in Figure 6.20), as it can be used to track the provenance of the data, which can help the client to select and filter the information based on their needs.

In order to for end-users to use the link services more easily, we also implemented an application: A link injection Mac OS service plugin, which calls a link service with a link proxy and inject link to the local applications. For example,



Black: original sameas relations; Red: newly injected sameas relations; Green: service description

Figure 6.20: A web browser function without and with link service proxy

when users select text from local documents or highlight text, the service plugin communicates with link services and inject links. This fulfils the open hypertext vision that links can be used anywhere in any type of document or application. Hence, it is also bridges the navigability between local document, the Web and Linked Data. For more detailed information please see Appendix 7.3.

6.3.3 Discussion

The link services are shown to be useful in improving the navigability of the Web of Linked Data. To compare to the link services of the Web, RDF provides a standard way to model the data and enable the links to be easily reused. Unlike the link services of the Web which does not have a structured data model of the links, the generic RDF data structure enable the programmable link services to be easily built. However, when modelling the links, it often requires extra ontologies to describe the link relations. For example, as we discussed in section 6.1.2.1, the backlink service that it requires an simple backlink ontology to describe the reverse linking relationship between URIs. In order to achieve a higher interoperability between services, it worth to standardising some URI relations ontologies, so that services can reuse these ontologies to describe their link relations.

The link services can be built by reusing and orchestrating existing link services. As discussed, a Linked Data crawler can be built using the resolution service and URIs extraction service. Hence, these services can be used to crawl the data and

analyse it, adding more relations and publish them as a link services. All of these can be achieve by only programming just the link services.

The data provided by link service can be precomputed or to be processed on the fly. The reasoning service for example can be based on the reasoning algorithm which crawls and precomputes the data, or the request can be reasoned on the fly. In order to ensure the extensibility, the reasoning component can be implemented by using the Semantic Web Rule Languages [33], so that the reasoning components can be reused and enhanced easily.

The quality of links provided by the link services is dependent to how links are generated by the service provided. Generally, there are three ways of generating links: the algorithmic approach, the human computation approach, and the hybrid approach which combines these two [156]. For definite link relations between data, such as the backlink and the reasoning service provided, the algorithmic approach can be used to generate links. For tentative link relations between data, such as the coreference service, human computation or the hybrid approach can be used. The link relations are simply a type of data, thus the quality of data, and the trust of the service are beyond the scope of this paper, which we will not be discussed further. When dealing with the information, often the more elaborate information we have, the better. However, it may be argued that there will be balance between a deficiency and an excess of data. The reality is that when information is deficient, nothing can be done about it. In contrast, when there are really so-called excessed data, it can always be filtered it out.

In the following, we discuss how link quality in link services could be evaluated practically.

6.4 Link quality in link services

discussion of how quality could be evaluated practically. You already have quantity, let's have quality. Give some indication of the quality of the links being created. How do different link services differ in terms of the quality of links they produce? - Does a service increase measure X where X is something you choose that is desirable?

The quality of links between two resources, depends on what the link can be used for, is a subjective question. In many research fields, data quality assessment is carried out by measuring different dimensions, such as accessibility, appropriate volume of data, believability, completeness, freedom-from-error, interoperability, ease of manipulation, understandability, timeliness, and so on [92]. Many of these dimensions cannot be assessed in a context-free manner. Data quality is generally conceived as "fitness for use" [91], which is the capability of data to fit the requirements of a specific user given a certain use case.

A number of studies have examined characteristics of the data that are considered to define quality. Four targets of study have been identified and are summarised here as link quality in terms of "technical infrastructure", "semantics", "trust",

and “network connectivity”.

6.4.1 Link quality as technical infrastructure

One way to measure the link quality is to examine data accessibility from the perspective of technical infrastructure. These studies often look into questions such as data availability upon requesting [89], whether the data provider follows best practice [89], and will failures of a Linked Data server cause network connectivity [77] to be broken. Also in this category is section 4.5, where an empirical study was carried of the Linked Data URI to see if the data publisher followed the Linked Data practices.

Guéret et. al. [77] measured the robustness of the Linked Data infrastructure as how link data is still connected after a node server is removed. He sampled Linked Data by using the 2010 billion triple challenge, and found that almost all infrastructural connectivity on the Web of Data is mediated by 3 servers: xmlns.org, dbpedia.org and purl.org. He identified the extreme brittleness of the infrastructure underlying the Web of Data: taking out only a handful of servers would completely cripple the entire network. He pointed out this can be solved by deploying mirror and cached data. However, another possible cost-effective method is by adding links to the graph to improve its connectivity.

Berners-Lee [15] proposed a five star scheme for describing Linked data quality

shown in Figure 6.21, which has been widely referenced in the Linked Data community. The first to the fourth stars can be considered as technical infrastructure, while the last point could be understood as about measuring the link quality from the semantic perspective.

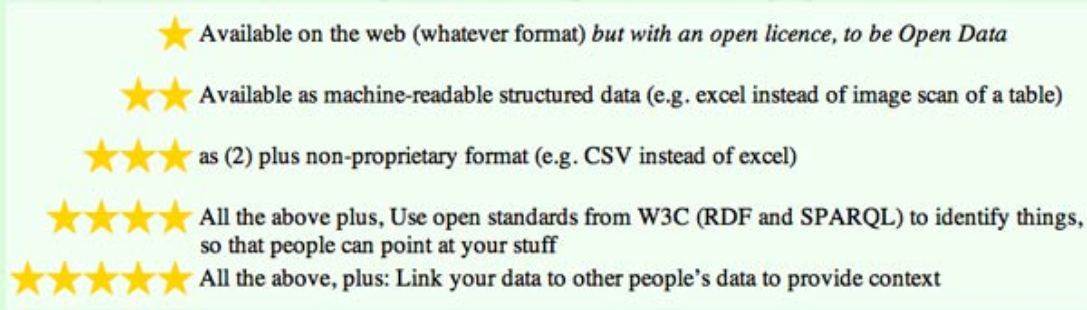


Figure 6.21: Five Star Linked Data [15]

6.4.2 Link quality as semantics

Similar to measuring the completeness of data within in a context, link quality from the semantic perspective is an important factor to measure. Since the Linked Data focus on lightweight semantics, studies such as [81], which measures if the semantic meaning of a link, will not discussed. Likewise, Bizer et al [118] discussed in great detail the semantics of Linked Data when published from structured sources such as a relational database [31] or a semi-structured source, such as Wikipedia [96, 118] . This is also outside of the scope of this work. Heath [86], in his widely cited article "Evolving the Web into a global data space", emphasised the importance of the fourth Linked Data principle - to set RDF links pointing to other data sources on the Web. He considers this is fundamental for the Web of Data, as these links are the glue that connects data islands into a global,

interconnected data space. Indeed, as mentioned in several places here, to include a foreign URI will provide a global context for the data. Likewise with the naming process, when a name is used and referenced elsewhere, it makes the name socially useful and meaningful. Therefore, links could be classified as "local links" and "global links" (across datasets foreign links). These foreign links can thus be considered a measure of semantically quality as they provide a global context of the Web of data.

6.4.3 Link quality as trust

Trustworthiness is often used in artificial intelligence [136] and on the Web [70] as a way of measuring quality. Quality on the Web is discussed in detail in Ciolek et al [46], who highlighted that massive amounts of Web content is becoming dated with the rapid change of the Web. Massa and Hayes, in their trust-related study [102], discussed classifying links as positive, negative or neither.

Artz and Gil [6] surveyed trust in the Semantic Web. Bizer and Oldakowski [32] considered that any statements contained in the Semantic Web must be considered as claims rather than facts until trust can be established. They also referenced Tim Berners-Lee's "Oh yeah?" button [25], which enable the user can click on every piece of information within an application and get explanations of why they should trust that information.

One of the important characteristics of trust is the provenance of the data. Quite a few approaches suggested [32] implementing trust on top of the linked data by attaching its provenance metadata to it. Since this would require additional implementation, the focus can be on the basic information naturally provided by its domain name. Whatever data is published under its description domain, can be considered to be from an original source, and anything else can be classified as of third party origin. Depending on the level of trust required, one can decide to use only the original data resources or a third party provider.

6.4.4 Link quality as network connectivity

Link quality measurement can also be based on network connectivity [77]. The method is to use statistical summaries of the network along different dimensions, for example, by detecting how interlinked a node is within a network. This method follows many classic network analysis studies. One of the common characteristics of these studies is the robustness of the network, which is its degree of tolerance against errors. For example, Barabasi shows scale-free networks are robust against random failure, but not against targeted attacks [1]. The way he measured this was to address the error tolerance of the network by changing the diameter when a few nodes were removed.

Guéret [77] applied this network analysis strategy to the Web of Data. One way to measure the robustness of a graph is to examine its diameter (the maximum of

the shortest paths in the graph), which provides information about connectivity. A smaller diameter implies that there are a large number of connections within the network while a larger diameter means that the network is less connected. This approach provides a global summary of connectivity.

Another approach is to investigate the graph node by node, measuring the betweenness centrality, or how often a node occurs on a shortest path linking any pair of nodes. In another words, betweenness measures the importance of a node for connectivity between other nodes - the removal of this node will directly influence the cost of connectivity between other nodes. Based on this, a completely connected network has the maximal robustness and correspondingly the lowest betweenness, and removing one node does not impact the overall connectivity of the network greatly.

Guéret considered that the robustness of the Web of Data could be improved by reducing the number of nodes that have high betweenness, as they are potential points of failure. As with our study mentioned in section 6.4.1, Guéret carried out a network analysis on the RDF namespace graph of the web of data, and concluded that the network relied heavily on the hubs. He then investigated several algorithms which "added links" to improve the robustness of the network. He showed that the centrality of the namespace graph could be improved by factor of 2 by adding just 4 edges to the graph. He also showed that the use of an evolution algorithm was successful in identifying a small number of links that substantially increase the robustness of the graph. His study also found that the improvement can only be seen for fewer than 100 million edges added and had

no impact for fewer than 10,000 edges.

Although Guéret proposed effective solutions, he only discussed the algorithmic approach to "adding links" to the graph. His study did not cover how the links would be published and used. The link services proposed here enrich connectivity not only by "adding links" from the data level, but also improve connectivity on the fly. For example, the composition service, as implemented, composes RDF data from various resources, enriches the connectivity by "folding the graph" rather than only "adding edges". From a high-level point view, these methods present two different linking qualities, since "folding the graph" to enrich connectivity is based on navigating the existing data, whereas, "adding edges" is about publishing the third party data to permanently enrich the web of data graph. This links back to the trust quality as well.

6.4.5 Discussion Link quality in link services

This section briefly examines the quality of link services based on the four concepts treated above. The difference of link service is discussed in terms of the quality of links they produce, as well as providing guidance on selecting the link services based on one's need.

In the following table 6.5, we summarise the five types of link services.

- To improve the technical infrastructure, the proposed resolution service can be used to avoid the servers that are permanently or temporarily unavailable, and follow the redirected links to bridge the navigation. The Linkbase and Reasoning services can also be used to mitigate server failures, as both these services can provide linking information for the data originally requested and discover alternative resources.

- To improve the global semantic context is the situation when one would like to discover more (non-local) semantic data of the requested resources. The Linkbase and Reasoning services produce the additional links, hence can be used to add more global semantic context links. The Composition service will append information from other resources, and therefore also enriches the semantic context.

- From the trust perspective, the services are only classified based on the data source: original data or a third party. As mentioned in section 6.4.3, the determination of whether data is from an original source is from its domain name. For instance, if data was obtained from dbpedia.org through `http://dbpedia.org/resource/United_Kingdom`, then it can be said that the data is from its original source. If an RDF triple about this URI is obtained from some other domain, e.g. from the sameas.org, it is said to come from a third party. The Resolution service, the Link extraction service, and the Composition service are based on the original data source; they do not generate or make assertions about the data. Whereas the Linkbase service, such as backlink service and co-reference reference service, makes

Table 6.5: Link quality in link services

	Improve Technical Infrastructure	**Adds Global Semantic Context**	**Trust**	**Enriches the Web of Data Connectivity**
Resolution service	Y	N	Original Data	On the fly
Link extraction service	N	N	Original Data	On the fly
Linkbase service	Y	Y	Third Party	Adding links
Reasoning service	Y	Y	Original and third party	One the fly or Adding links
Composition service	N	Y	Original	On the fly

new statements about the data. The Reasoning service can be implemented to provide reasoning on the fly, or based on pre-computed data. Therefore, based on the level of trust needed, a group of services can be selected to be used together.

- Do these services enrich the web of data connectivity based on adding links or on the fly? As discussed in section 6.4.4, "adding links" would require re-publishing the pre-computed third party data, and Linkbase is just such a service. The Resolution service, Link extraction, and Composition services enrich connectivity by either navigating the web of data or "folding" the graphs of the Web of data on the fly. The Reasoning service provides either on-the-fly reasoning navigation or republishing pre-computed reasoning data. The on-the-fly approach can be used by any URI, whereas the "adding link" approach is only useful for third party data created.

This section discussed the link quality different services can provide for different contexts. The link service was found to be useful for mitigating technical infrastructure failure, adding global semantic context to the data, and enriching the connectivity of the Web of data. The services also provide different levels of trust depending on whether they were based on the original source of data or a third party provider. Two different ways of enriching the connectivity of the Web of data were discussed: by navigating the Web of data or by adding links to the Web of data. Possible future work would be a more in depth link quality study, similar to Guéret's study mentioned in section 6.4.4, to look into how much the Linkbase service and the Reasoning service improve the robustness of connectivity of the data. For the services which follow an on-the-fly approach, network statistical analysis could also be applied to dynamically evaluate how much connectivity improves for each individual linking case.

In one of the enAKTing project meetings, Berners-Lee expressed his personal view on the link service by recommending adding links to the Web of data. Although the Web follows a "scale-free" network characteristic, he believes the growth of the web of data should be "free scale", that is, grown naturally by the data publisher. We also share this vision, and consider these linking services to be complementary services to the existing web of data than deliberately engineering the completely linked web of data.

END

6.5 Conclusions

This Chapter presented two basic link services that enable the navigability of the Web of Linked Data; the resolution service and the URI extraction service. Link services that improve the navigability of the Web of Linked Data were then proposed and implemented. These services are: linkbase service, composition service, reasoning service and link injection service. In order for users to better utilise link services, a link service proxy named Hyperthing proxy was implemented. This proxy acts as a local server with a resolution service and customisable orchestrated link services, which aggregates the list of services and enriches the Web of Linked Data dynamically. To evaluate the system, four services were used in the proxy to test the proposed system. They are the PSI backlink service, co-reference server sameas.org, geo-reasoning service and link injection service. These services contribute a total of 373 million across-datasets foreign URIs, from a total of 425 million across-datasets foreign URIs in the Linked Data cloud. It is estimated that at most the 7.5% resources across-datasets foreign URIs coverage are linked to other datasets. Building the Link service on top of Linked Data dramatically increases this rate to the 14% coverage, which almost doubles previous rates.

Chapter 7

Conclusions and Future Works

This thesis studied Linked Data as a hypertext system and investigated idea of building link services as an extension of Linked Data to improve its navigability. The study was carried out by answering three research questions: 1) How can an agent understand what a URI refers to by dereferencing it; 2) How is the Web of Linked Data linked; 3) What services we can be built to extend Linked Data to make data better linked.

First the Semantic Web and its URI were studied. A fundamental changes from the Web to the Semantic Web is that the URI can be used to identify real world objects instead of only used to identify Web documents. When building Linked Data on the Web, these identities need to be kept consistent. W3C has proposed two publishing approaches to differentiate these two types of URIs, however how an agent can understand what a URI refers to by dereferencing it was not studied in details. In this thesis, the HTTP URI dereferencing process was formally

modelled according to its specification and summarised in a set of URI rewriting rules. Based on these rules, a Linked Data validator was implemented which can be used to help data publisher to publish Semantic Web URIs as well as differentiate whether a URI identifies a real world object or a web document. To evaluate the validator and the current deployment of the Semantic Web URIs, we carried an empirical study was carried out into dereferencing the most frequently referenced 25 ontology URIs and their properties. It concluded that, on average, more than half of Linked Data ontology URIs did not follow the published guidance to make the URI identity consistent. The Hyperthing validator can be used to avoid these publishing mistakes.

To answer the second research question, two studies were then carried out to analyse how Linked Data are linked. An overall analysis on the metadata of Linked Data cloud provided by the CKAN was first carried out. This determined that there are 425 million across datasets foreign URIs on the Web of Linked Data, of which only 7.5% resources had across datasets foreign, which is relatively low. A detailed analysis of the linking pattern of the New York Times datasets and its neighbouring datasets determined that these datasets were linked by two extreme patterns: either the majority of data instances are one-to-one parallel linked from one dataset to another, or the majority of the data instances from one dataset only link to a few data instances of another dataset. This means that, for those linked datasets, an agent can only follow some of the resources in a dataset to discover other resources, and their navigability are rather limited. More diverse linking patterns are desired to enable the discovery of data.

In order to improve the navigability of the Web of Linked Data, a category of link service built for the Web was first summarised and defined, and a few link services for Linked Data then built. They are: resolution service (retrieves the RDF description of the requested URI for agents), link extraction service (extracts URIs from a RDF), linkbase service (third party hosting link relations between datasets, especially for those data which were not originally linked), reasoning service (applying rules of reasoning to generate a new RDF), composition service (compose multiple RDF documents into one documents), and link injection service (inject extra links relations into the client requested RDF document). To use the link service, it is almost always requires multiple requests from the clients: thus, to make the service transparent to the clients and to enable clients to orchestrate link services easily, we built a link service proxy which can be used from the client side with any Linked Data application. By doing so, when clients request a URI via HTTP, the proxy injects link relations to the requested RDF documents on the fly, hence augmenting Linked Data. We evaluated the link service proxy using four services we built in the enAKTing project: PSI backlink service, sameAs co-reference service, geo-reasoning services, and a link injection service. It was demonstrated that these these services alone added 373 million across datasets foreign URIs, which almost doubles the previously mentioned 7.5% across-datasets foreign URIs coverage to the 14%. We also demonstrated how the linked service proxy works with the Web browser to enrich Linked Data dynamically. As all link services can be easily reused, and programmed to navigate the Web of Linked Data as well as generating new link services, we believe this provides a basis for agents to consume Linked Data. Following this trend, the Linked Data consumers will only need to orchestrate or create the link services

to consume Linked Data. Any other web-based Linked Data applications can be understood as specialised services to be built on top of the link services.

This work showed the great advantages of building link services based on the REST architecture. Decoupling each component enables the reusability and extensibility of services. One future work direction is that the link services can be standardised by designing commonly used ontologies. Linked Data hypothesised that an agent can dereference a URI and obtain a RDF description about the URI, and RDF is sufficient for an agent to understand what a URI refers to. Based on these, we can hypothesised that links can be established automatically between data by agents to create a self-organised the Web of Data (automated Linked Data). Especially, with the link service ontologies in place, link services can be programmed to create more link relations on the Web of Linked Data. For example, one can program a link services that crawls the Web of Linked Data and constantly updates backlink triples for the whole Web of Linked Data.

This thesis recognize the link services as extensions of a hypertext system, since the Web is a hypertext, another future work direction is to decouple the Web components and rethink the Web as a link service.

7.0.1 Rethink the Web as a link service

From a simple view, a hypertext system deals with three elements (data): information, link and combined link and information (a new type of data). The Web as hypertext system decoupled many hypertext system features, provided a basic hypertext data structure, HTML document and URI links. Its powerful REST architecture enables many services to be build on top, to extend the system easily. If the hypertext system is rethought from a service point view, we can design a hypertext service ecosystem with three minimal services (as shown in Figure 7.1): a service that serves information, a service to serves link and a service to serves combined information and link.

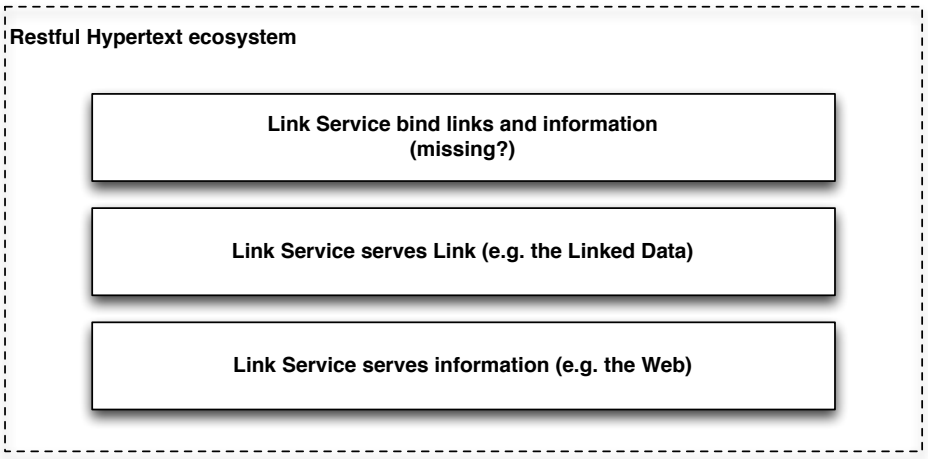


Figure 7.1: Restful Hypertext Ecosystem

Decoupling each elements in the hypertext enables the reusability of the all components (data). For example, we can design a hypertext ecosystem by implementing three services:

1. A service can be designed that serves a human readable string (e.g. a plain text document);

2. A service that serves link identity which can be used to identify any resources mentioned in the sentence (e.g. a document with link identity);

3. A service that can combine the link and words in the sentence based on any interpretation or order (e.g. annotated document with link identity).

These three services, as architectural components, are the very fundamental basis of a hypertext service system. Finally, in order to achieve interoperability the output of each service needs to be standardised. Users can reuse the output from the service in any possible way for republishing the new data. So far, with the existing technology of the Web and Linked Data, we can see they map the above architecture in the following way:

1. A simplified version of the Web as a service that serves information as HTML document (we mean a document without any URI links);

2. Linked Data as a service that serves link identity as URI which is described by RDF document;

3. A service that combines the link and information as a new type of document (e.g. an annotated HTML with URIs);

Note that the third service is the idea of republish the annotated HTML with URIs as a new type of document and it should be served under a new URI. For example, one may achieve this using RDFa to annotate the HTML and republish this ocument under a new URI, so that this data can be reused. Therefore, another piece of future work would be carried out is to build a such system to identify potential issues.

At the beginning of the thesis, we quoted Berners-Lee's point of view that data is a precious thing and will last longer than the systems themselves. We believe this marked up information with links as data will last for very long time, can always be reused by machines and enables a vast range of possibilities.

Appendix A

7.1 PSI Backlinking Service

Figure 7.2 is a visual overview of the Backlinking Service architecture, and is composed of three parts:

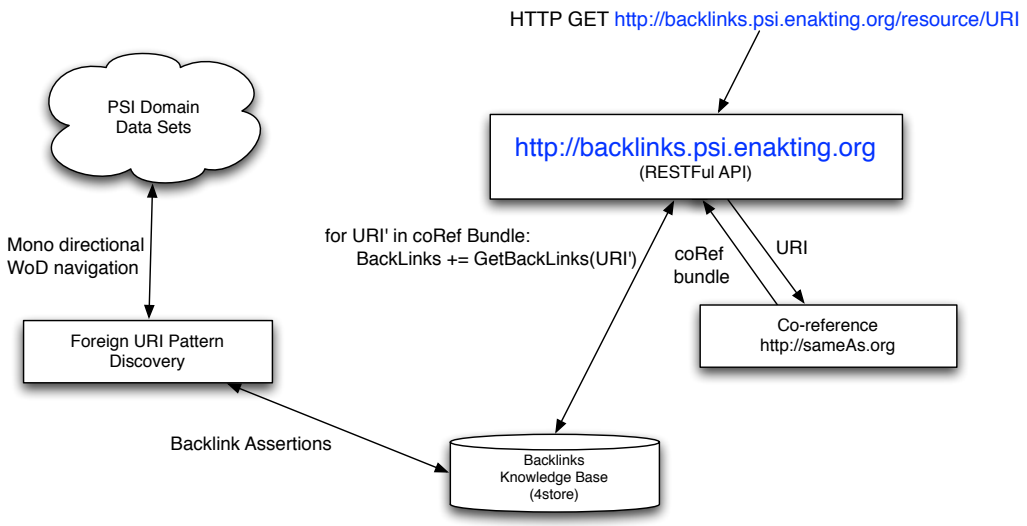


Figure 7.2: Back Link Service Architecture

Foreign URI Pattern Discovery Component: This is the component responsible for automatically navigating the PSI data sets from the WoD and identifying *foreign URIs*. This component navigates the WoD retrieving all the *foreign URIs* found in the data sets under the study. It resolves all the RDF documents from a starting (or input) list of URIs and inspects each document returned to identify triples in which the object is a *foreign URI*. For every foreign URI found we assert a *rdfs:seeAlso* statement into the knowledge base. The *seeAlso* statement is a triple that points to the original URI in case of backward navigation. For instance, if the service was analysing `nsh:TrustBarnsley` then we would discover that the document returned by resolving that URI contains a triple in which `os:Barnsley` is in the object position, i.e. `os:Barnsley` is a foreign URI in this context. If a client were seeking information about `os:Barnsley` then it may wish to discover the information about that concept contained within the document retrieved when resolving `nsh:TrustBarnsley`. As a result, the corresponding assertion into the backlinks knowledge base is:

Which follows the pattern:

```
<FOREIGN-URI>

        rdfs:seeAlso <ORIGIN-URI>
```

RESTFul API as Front-end Service: Access to the service is provided by a REST-Ful API that accepts requests by simple HTTP GETs. The interface is:

http://backlinks.psi.enakting.org/resource/URI

Where *URI* is the resource for which we want to discover the backlinks. The service queries the knowledge base where the *seeAlso* statements were asserted and returns a document with all the backlinks. The output of the service can be obtained in JSON, RDF+XML, TURTLE or HTML, either through the Service URL or by specifying the accept header of the HTTP request[1].

7.2 Geo Reasoning Service

In the Enakting project, we implemented a service called Geo Services [48], which takes input a city URI as input, and provides a country RDF which contains the city. Taking as an example the PSI data sets published recently[2], we adopted the Ordnance Survey administrative ontology in order to provide context to our data items (i.e. SCOVO [84] items instances[3] and local governmental data). The SCOVO ontology allows us to describe statistical data as a collection of *Items* where each item describes a statistical value (i.e. a single cell in a multidimensional table) along with all the dimensions that characterise it. In the case of UK PSI statistics, many data sets collected were related to geographical regions (counties, districts, etc.)

[1] More detailed documentation about the API can be found at http://backlinks.psi.enakting.org

[2] http://browser.psi.enakting.org

[3] http://purl.org/NET/scovo

In this case, users who wished to discover useful information about their own region (e.g. the County of Hampshire, top Figure 7.3) would start their searching activity by browsing one of its available URIs. The OS URI for such geographical entity would be `os:7000000000017765`[1], but any equivalent URI provided by a co-reference system will provide the same results as will be described in the following. Using a backlinking service for resolving the entities linking to the given URI for Hampshire, we are able to retrieve links to mortality statistics (`mortality:ds1_299_[1...3]`[2]) and crime statistics (`crime:ds1_37_[1...11]`[3]). In Figure 7.3 those URIs are contained in boxes labelled as *"accessible"*, meaning that those URIs are retrievable following back already existent arcs. Those SCOVO data sets' items address in fact Hampshire county as one of their dimensions. What is missing is the further data collected that reports valuable information about regions contained in Hampshire. In particular, within the EnAKTing project, we published linked data about the singular constituencies too. In detail we published, for each of constituency, an historical record of the MP in charge for that constituency, his/her voting records and expenses. In Figure 7.3 those resources are contained in dotted boxes labelled as *"inaccessible"*, meaning that they cannot be retrieved with the existent knowledge.

The service is accessed via HTTP GET requests and provide two essential information: the list of entities **contained** the input URI, and the list of entities that **contains** the input URI. The interface is then accessible via the following

[1] `PREFIX os:<http://data.ordnancesurvey.co.uk>`

[2] `PREFIX mortality:<http://mortality.psi.enakting.org/id/>`

[3] `PREFIX crime:<http://crime.psi.enakting.org/id/>`

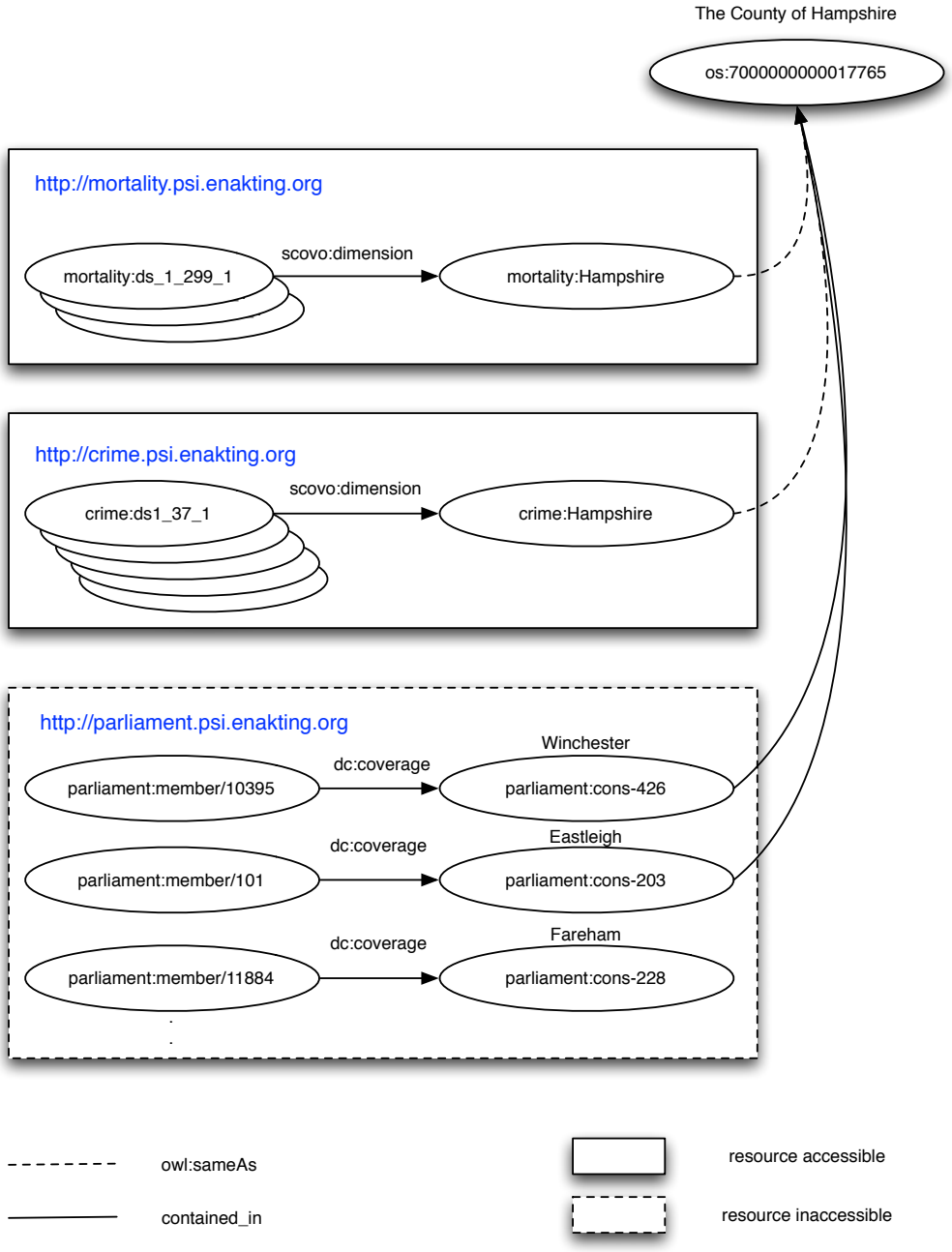


Figure 7.3: Resource irretrievable via geographical gap

URIs:

```
http://geoservice.psi.enakting.org/{command}/

  {dictionary}/{format}/{URI}
```

In the above API description, the parameters are enclosed in brackets and their meaning is the following:

command: can be either **contains** or **container**: in the first case it returns the URIs of the entities contained by the input URI; in the second case it returns the URIs of the entities that contains the input URI.

dictionary: can be one of the followings (**dbpedia**, **os**, **statistics**, **geonames**, **enakting**, **opencyc**, **openlylocal**, or **none**) and instructs the service to use the co-reference system in order to retrieve the equivalent URIs in the respective data sets (i.e. DBpedia [7], Ordnance Survey [72], UK National Statistics[1], Geonames[2], PSI enAKTing[3], OpenCYC [97], Openly Local project[4]). The value **none** is used for not applying any filter. In this case the URIs returned will be the ones from the Ordnance Survey plus the ones returned from the co-reference service.

[1] http://statistics.data.gov.uk last accessed 10/02/10
[2] http://geonames.org last accessed 10/02/10
[3] http://browser.psi.enakting.org last accessed 10/02/10
[4] Community devoted to provide linked data access for local government data, see http://openlylocal.com last accessed 10/02/10

format: the format parameter is optional and can be one of the followings (**rdf**, **text**, **ttl**, or **json**). The value of the **format** parameter decide then the format of the returned content: RDF/XML for **rdf**; list of URIs separated by new lines for **text**; RDF/Turtle for **turtle**; and finally JSON[1] for **json**. If the parameter is not given the right content is decided using the 303 HTTP redirection. Even for the content requests `Accept:text/html` done using the browser, the client is redirected to the HTML page of the service initialised with the input URI.

URI: is the URI of the input entity to query using the service. The service uses a co-reference system in order to find the equivalent URI for the Ordnance Survey and the Geonames data set. This means that the user can use one of the data set of preference (e.g. DBpedia or Geonames) and ask for contained, or container, entities in one of the desired target data set (e.g. again DBpedia, Geonames, or enAKTing published information).

The service returns a list of URIs if the content type is **text** or **json**. The RDF content, for both **rdf** and **turtle**, describes the containment relations between the input URI and the resulting resources. In both cases the returned URIs are translated into the desired address space.

The procedure followed by the service, and an overall architecture, is depicted in Figure 7.4, and can be describe as follows:

[1] `http://json.org` last accessed 10/02/10

1. user generated request (HTTP GET request)

2. normalisation of the input URI to OS

3. computation of the property closure (i.e. **part** or **part-of**) over the normalised URI

4. optional phase of translation and filtering of the resulting URIs to the target URI space

5. formatted content, as per user request, returned to the user (HTTP Response)

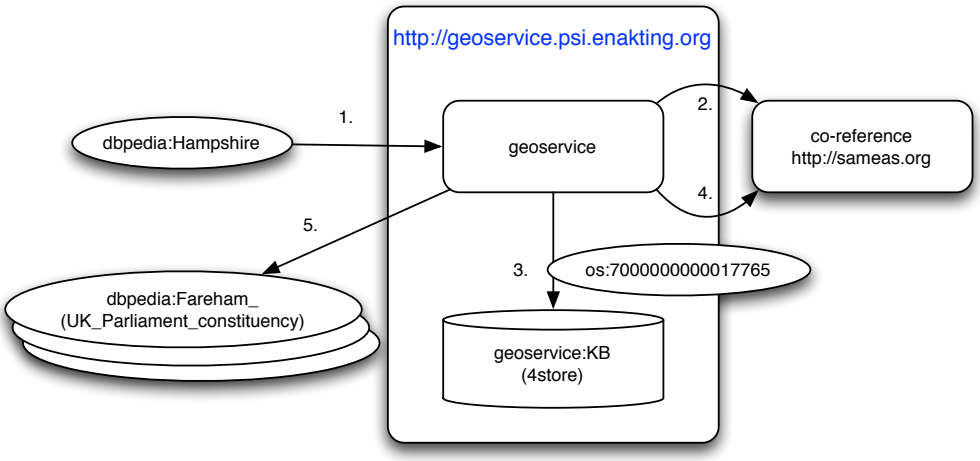


Figure 7.4: Overall architecture and interaction with co-reference system

7.3 A Mac OS service plugin use link service proxy to bridge the navigation between local documents, the Web and Linked Data

The open hypertext vision is that links can be used anywhere in any type of document or application. To fulfil this, we implemented a link injection services plugin for Mac OS. This can be used to bridge the navigation between personal local documents, the Web and Linked Data. When users select text from local documents or highlight text, our services communicate with search engines Sindice and Freebase (customiseable) to retrieve a list of URIs and display them in the browser; the user can then follow the link to discover more RDF data. Figure 7.5 shows a use case of the link injection service, wherein a user highlights text in an email in the Apple Mail ('Dr Nicholas Gibbins') and from the right-click menu selects the Hyperthing service, a list of URIs is retrieved from the search engine and are listed in the browser on the right. All these URIs refer to a 'person' rather than a webpage. A keyboard shortcut can be created to call the service. This plugin turns the whole Mac OS into an open hypertext system, where links can be injected anywhere.

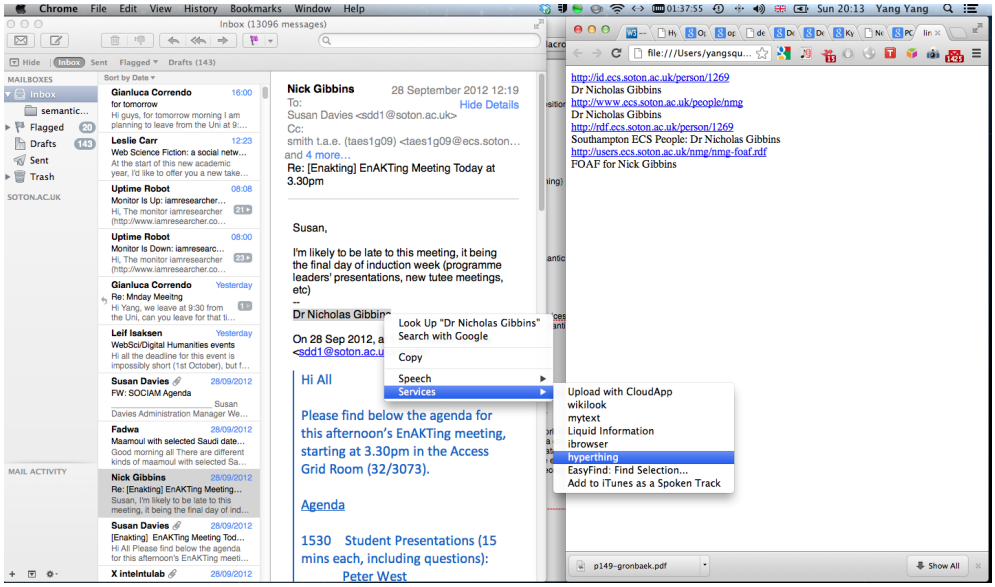


Figure 7.5: Hyperthing Mac OS Service Extension Bridge Navigation between local application and Linked Data

References

[1] Réka Albert, Hawoong Jeong, and Albert-László Barabási. Error and attack tolerance of complex networks. *Nature*, Nature Publishing Group, **406**[6794]:378–382, 2000. 177

[2] K. Alexander and M. Hausenblas. Describing Linked Datasets on the design and usage of VoID, the vocabulary of interLinked Datasets. In *Linked Data on the Web Workshop (LDOW 09), in conjunction with 18th International World Wide Web Conference (WWW 09)*. Citeseer, 2009. 125

[3] J. R. Anderson and P. L. Pirolli. Spread of activation. *Journal of Experimental Psychology: Learning, Memory, and Cognition*, American Psychological Association, **10**[4]:791, 1984. 40

[4] F. Anklesaria, M. McCahill, P. Lindner, D. Johnson, D. Torrey, and B. Alberti. The Internet Gopher Protocol: (a distributed document search and retrieval protocol). Technical report, RFC 1436, 1993. 31

[5] G. Antoniou and F. Van Harmelen. *Semantic Web Primer*. MIT Press, 2004. vii, 52, 53

[6] Donovan Artz and Yolanda Gil. A survey of trust in computer science and the semantic web. *Web Semantics: Science, Services and Agents on the World Wide Web*, Elsevier, **5**[2]:58–71, 2007. 176

[7] S. Auer, Christian B., G. Kobilarov, J. Lehmann, and Z. Ives. DBpedia: A nucleus for a Web of Open Data. *In* 6^{th} *International Semantic Web Conference, Busan, Korea*, pages 11–15, 2007. 113, 196

[8] L. Barnes. RUNOFF: A program for the preparation of documents. Technical report, DTIC Document, 1969. 34

[9] D. Bawden. Bush, Goldberg, Memex and the revision of history. *Journal of Documentation*, Emerald Group Publishing Limited, **66**[4], 2010. 12

[10] T. Berners-Lee. W3 future directions. *Accessed 26/09/2013*, WWW Geneva 94. vii, 47, 48

[11] T. Berners-Lee. Universal Resource Identifiers. *W3C Design Issues*, [http://www.w3.org/DesignIssues/Axioms.html], 1996. 24

[12] T. Berners-Lee. The Web Model. *W3C Design Issues*, 1998. vii, 65, 66

[13] T. Berners-Lee. Semantic Web-xml2000, 2000. *W3C Talk*, 2000. 53

[14] T. Berners-Lee. Artificial Intelligence and the Semantic Web: AAAI2006 Keynote. *W3C Web site. URL: http://www. w3. org/2006/Talks/0718-aaai-tbl/Overview. html. Accessed 12/10/2013*, **12**[10], 2006. 53

[15] T. Berners-Lee. Linked Data. *W3C Design Issues*, 2006. ix, 2, 61, 174, 175

[16] T. Berners-Lee. A Short History of the term "Resource". *W3C Design Issues*, 2009. 66

[17] T. Berners-Lee. Linked Data - the story so far. *International Journal on Semantic Web and Information Systems*, **5**[3]:1–22, 2009. 60

[18] T. Berners-Lee, R. Cailliau, J.F. Groff, and B. Pollermann. World Wide Web: The information universe. *Internet Research*, MCB UP Ltd, **2**[1]:52–58, 1992. 1, 19

[19] T. Berners-Lee, Y. Chen, L. Chilton, D. Connolly, R. Dhanaraj, J. Hollenbach, A. Lerer, and D. Sheets. Tabulator: Exploring and analyzing linked data on the semantic web. In *Proceedings of the 3rd International Semantic Web User Interaction Workshop*, **2006**, 2006. 159

[20] T. Berners-Lee and D. Connolly. Hypertext markup language: A representation of textual information and metainformation for retrieval and interchange. *CERN Internal Draft*, 1993. 26

[21] T. Berners-Lee and D. Connolly. HTML 2.0 specification. *W3C: http://www. w3. org/MarkUp/html-spec*, 1995. 34

[22] T. Berners-Lee, R. Fielding, and L. Masinter. Uniform Resource Identifiers (uri): Generic syntax. Technical report, RFC 1630, 1994. 26

[23] T. Berners-Lee, R. Fielding, and L. Masinter. Uniform Resource Identifiers (uri): Generic syntax. Technical report, RFC 2396, 1998. 23

[24] T. Berners-Lee, R. Fielding, and L. Masinter. Uniform Resource Identifiers (uri): Generic syntax. Technical report, RFC 2396, 1998. 26, 32, 65

[25] T. Berners-Lee and M. Fischetti. Weaving the web. Orion Business Books United Kingdom, 1999. 176

[26] T. Berners-Lee, J. Hendler, and O. Lassila. The Semantic Web. *Scientific American*, **284**[5]:34–43, 2001. 2, 48

[27] D. Berrueta, S. Fernández, and I. Frade. Cooking http content negotiation with vapour. In *Proceedings of 4th Workshop on Scripting for the Semantic Web (SFSW2008)*, 2008. 72

[28] K. Bharat, A. Broder, M. Henzinger, P. Kumar, and S. Venkatasubramanian. The connectivity server: Fast access to linkage information on the web. *Computer Networks and ISDN Systems*, Elsevier, **30**[1]:469–477, 1998. 40

[29] C. Bizer, A. Jentzsch, and R. Cyganiak. State of the LOD Cloud. *Version 0.3 (September 2011)*, 2011. viii, 3, 106, 108

[30] C Bizer, A Jentzsch, and R Cyganiak. State of the lod cloud: Version 0.3, 2011. 123

[31] Christian Bizer and Richard Cyganiak. D2r server-publishing relational databases on the semantic web. In *Poster at the 5th International Semantic Web Conference*, 2006. 175

[32] Christian Bizer and Radoslaw Oldakowski. Using context-and content-based trust policies on the semantic web. In *Proceedings of the 13th international World Wide Web conference on Alternate track papers & posters*, pages 228–229. ACM, 2004. 176, 177

[33] H. Boley, S. Tabet, and G. Wagner. Design rationale for RuleMl: a markup language for Semantic Web Rules. In *SWWS*, **1**, pages 381–401, 2001. 172

[34] K. Bollacker, C. Evans, P. Paritosh, T. Sturge, and J. Taylor. Freebase: a collaboratively created graph database for structuring human knowledge. In *Proceedings of the 2008 ACM SIGMOD international conference on Management of data*, pages 1247–1250. ACM, 2008. 114

[35] D. Booth. URIs and the myth of resource identity. In *Proceedings of Identity, Reference, and the Web (IRW 2006) Workshop at the World Wide Web Conference*, 2006. 71, 79

[36] B. Bos, T. Çelik, I. Hickson, and H.W. Lie. Cascading style sheets level 2 revision 1 (css 2.1) specification. *W3C Recommendation, W3C, June*, 2005. 34

[37] T. Bray, J. Paoli, C.M. Sperberg-McQueen, E. Maler, and F. Yergeau. Extensible Markup Language (xml). *World Wide Web Journal*, **2**[4]:27–66, 1997. 18

[38] W. H. Brown, R. C. Malveau, and T. J. Mowbray. AntiPatterns: refactoring software, architectures, and projects in crisis. Wiley, 1998. 121

[39] M. Buckland. Emanuel Goldberg, electronic document retrieval, and Vannevar Bush's Memex. *Journal of the American Society for Information Science*, Wiley Online Library, **43**[4]:284–294, 1992. 13

[40] V. Bush. As we may think. *ACM SIGPC Notes*, ACM, **1**[4]:36–44, 1979. 11

[41] S. Carmody, W. Gross, T.H. Nelson, D. Rice, and A. van Dam. A hypertext editing system for the/360. *Pertinent concepts in computer graphics*, University of Illinois Press, pages 291–330, 1969. 17

[42] L. Carr, W. Hall, S. Bechhofer, and C. Goble. Conceptual linking: ontology-based open hypermedia. In *WWW '01: Proceedings of the 10th international conference on World Wide Web*, pages 334–342. ACM, 2001. 41

[43] L.A. Carr, D. De Roure, W. Hall, and G. Hill. Implementing an open link service for the World Wide Web. *World Wide Web Journal*, Springer, **1**[2]:61–71, 1998. 21

[44] E. Castro. *HTML for the World Wide Web*. Peachpit Press, 2003. 34

[45] S. Chakrabarti, D.A. Gibson, and K.S. McCurley. Surfing the Web backwards. *Computer Networks*, Elsevier, **31**[11]:1679–1693, 1999. 39, 40

[46] T Matthew Ciolek. The six quests for the electronic grail: Current approaches to information quality in www resources. *Revue Informatique et Statistique dans les Sciences Humaines*, **32**:1–4, 1996. 176

[47] J. Conklin and M.L. Begeman. gIBIS: A hypertext tool for team design deliberation. In *Proceedings of the ACM conference on Hypertext*, pages 247–251. ACM, 1987. 19

[48] G. Correndo, M. Salvadores, Y. Yang, N. Gibbins, and N. Shadbolt. Geographical service: a compass for the Web of Data. In *Linked Data on the Web (LDOW2010)*, April 2010. Event Dates: 27 April 2010. 141, 193

[49] D. Curry, H. Debar, and B. Feinstein. Intrusion detection message exchange format data model and extensible markup language (xml) document type definition. *Intrusion Detection Working Group*, 2001. 34

[50] R. Cyganiak and A. Jentzsch. Linking open data cloud diagram. *Online. http://lod-cloud. net/, Accessed 26/09/2013*, 2011. viii, 96, 106, 107

[51] D. De Roure, R. Carr, W. Hall, and G. Hill. A distributed hypermedia link service. In *Proceedings of the 3rd International Workshop on services in Distributed and Networked Environments, 1996.,*, pages 156–161. IEEE, 1996. vii, 42, 43, 44

[52] L. De Young. Linking considered harmful. In *Hypertext: Concepts, Systems and Applications*, **90**, pages 238–249, 1990. 20

[53] S. DeRose, E. Maler, D. Orchard, and B. Trafford. XML Linking language (xlink). *W3C Recommendations*, 2000. 18

[54] N. Dershowitz and J. Jouannaud. *Rewrite systems*. Citeseer, 1989. 88

[55] E. W. Dijkstra. Letters to the editor: go to statement considered harmful. *Communications of the ACM*, ACM, **11**[3]:147–148, 1968. 20

[56] J. Domingue and M. Dzbor. Magpie: supporting browsing and navigation on the semantic web. In *Proceedings of the 9th international conference on Intelligent user interfaces*, pages 191–197. ACM, 2004. 42

[57] G. Durand and S. J. DeRose. FRESS hypertext system. In *Proceedings of the 5th ACM conference on Hypertext*, HYPERTEXT '93, page 240, New York, NY, USA, 1993. ACM. 17

[58] M. Dürst and M. Suignard. Internationalized resource identifiers (iris). Technical report, RFC 3987, 2005. 33

[59] D. Engelbart. A conceptual framework for the augmentation of man's intellect. *Computer-supported cooperative work: a book of readings*, Morgan Kaufmann Pub, pages 35–66, 1988. 16

[60] D. Engelbart and W. English. A research center for augmenting human intellect. In *Proceedings of the fall joint computer conference, part I*, pages 395–410. ACM, 1968. 13

[61] D. C. Engelbart. Augmenting human intellect: a conceptual framework (1962). *PACKER, Randall and JORDAN, Ken. Multimedia. From Wagner*

to Virtual Reality. New York: WW Norton & Company, pages 64–90, 2001. 13, 14, 16

[62] R. Fielding. *Architectural styles and the design of network-based software architectures*. PhD thesis, University of California, 2000. 28

[63] R. Fielding. httpRange-14. *W3C Technical Architecture Group Issue*, [http://www.w3.org/2001/tag/issues.html#httpRange-14,], 2003. 57

[64] R. Fielding, J. Gettys, J. Mogul, H. Frystyk, L. Masinter, P. Leach, and T. Berners-Lee. Hypertext Transfer Protocol - HTTP/1.1. Technical report, RFC 2616, 1999. 23, 25, 27, 40, 84

[65] A.M. Fountain, W. Hall, I. Heath, and H.C. Davis. Microcosm: An open model for hypermedia with dynamic linking. In *The Proceedings of The European Conference on Hypertext: Concepts, Systems and Applications. INRIA, France*, 1990. 17

[66] N. Freed and N. Borenstein. Multipurpose internet mail extensions (MIME) part one: Format of internet message bodies, 1996. 30

[67] A. R. Galloway. *Protocol: how control exists after decentralization*. The MIT Press, 2004. 25, 30

[68] A. Gerber, A. Van der Merwe, and A. Barnard. A functional Semantic Web architecture. *The Semantic Web: Research and Applications*, Springer, pages 273–287, 2008. 53

[69] H. Glaser, A. Jaffri, and I. Millard. Managing co-reference on the Semantic Web. In *WWW2009 Workshop: Linked Data on the Web*, April 2009. 138

[70] Jennifer Golbeck. Trust on the world wide web: a survey. *Foundations and Trends in Web Science*, Now Publishers Inc., **1**[2]:131–197, 2006. 176

[71] C.F. Goldfarb and Y. Rubinsky. *The SGML handbook*. Oxford University Press, 1990. 34

[72] J. Goodwin, C. Dolbear, and G. Hart. Geographical Linked Data: The administrative geography of Great Britain on the Semantic Web. *Transaction in GIS*, **12**[1]:19–30, February 2009. 196

[73] I. Graham and L. Quin. *XML specification guide*. John Wiley & Sons, Inc., 1999. 35

[74] K. Grønbæk and R. H. Trigg. Design issues for a Dexter-based hypermedia system. In *Proceedings of the ACM conference on Hypertext 92*, pages 191–200. ACM, 1992. 38

[75] K. Gronbaek and R. H. Trigg. Toward a Dexter-based model for open hypermedia: unifying embedded references and link objects. In *Proceedings of the the 7th ACM conference on Hypertext*, HYPERTEXT '96, pages 149–160. ACM, 1996. 37, 38

[76] T. Gruber. A translation approach to portable ontology specifications. *Knowledge acquisition*, Academic Press, **5**[2]:199–220, 1993. 51

[77] Christophe Guéret, Paul Groth, Frank Van Harmelen, and Stefan Schlobach. Finding the achilles heel of the web of data: using network analysis for link-recommendation. In *The Semantic Web–ISWC 2010*, pages 289–304. Springer, 2010. 174, 177

[78] R. Guha, R. McCool, and E. Miller. Semantic search. In *Proceedings of the 12th international conference on World Wide Web*, pages 700–709. ACM, 2003. 55

[79] R. V. Guha. Meta content framework: A white paper. Technical report, Apple Technical Report, 1997. 55

[80] F. Halasz, M. Schwartz, K. Grønbæk, and R. H. Trigg. The Dexter hypertext reference model. *Communications of the ACM*, ACM, **37**[2]:30–39, 1994. vii, 37, 38

[81] H. Halpin, P. Hayes, J. McCusker, D. McGuinness, and H. Thompson. When owl: sameas isn't the same: an analysis of identity in Linked Data. In *The Semantic Web-ISWC 2010*, pages 305–320. Springer, 2010. 140, 175

[82] H. Halpin and V. Presutti. An ontology of resources: Solving the identity crisis. *The Semantic Web: Research and Applications*, Springer, pages 521–534, 2009. vii, 57, 58, 59

[83] O. Hartig and J. Freytag. Foundations of traversal based query execution over linked data. In *Proceedings of the 23rd ACM conference on Hypertext and social media*, pages 43–52. ACM, 2012. 105

[84] M. Hausenblas, W. Halb, Y. Raimond, L. Feigenbaum, and D. Ayers. Scovo: Using statistics on the web of data. *The Semantic Web: Research and Applications*, Springer, pages 708–722, 2009. 193

[85] P.J. Hayes and H. Halpin. In defense of ambiguity. *International Journal on Semantic Web and Information Systems*, IGI Publishing, 701 E. Chocolate Ave, Suite 200, Hershey, PA, 17033-1240, USA,, **4**[2]:1–18, 2008. 50, 59

[86] Tom Heath and Christian Bizer. Linked data: Evolving the web into a global data space. *Synthesis lectures on the semantic web: theory and technology*, Morgan & Claypool Publishers, **1**[1]:1–136, 2011. 105, 175

[87] I. Hickson and D. Hyatt. HTML5 specification. *W3C Working Draft,(March 10, 2012). Available online at: http://www. w3. org/TR/html5*, 2011. 35

[88] G. J. Hill, W. Hall, D.C. DeRoure, and L. A. Carr. Applying open hypertext principles to the WWW. *Hypermedia Design*, Springer, 1995. 1, 21, 39

[89] A. Hogan, A. Harth, A. Passant, S. Decker, and A. Polleres. Weaving the pedantic web. CEUR, 2010. 72, 174

[90] I. Jacobs and N. Walsh. Architecture of the World Wide Web, volume two. W3C Recommendations, 2004. vii, 24, 26

[91] J. JURAN. The quality control handbook. *McGraw-Hill*, 1974. 173

[92] Beverly K Kahn, Diane M Strong, and Richard Y Wang. Information quality benchmarks: product and service performance. *Communications of the ACM*, ACM, **45**[4]:184–192, 2002. 173

[93] F. Kappe. Hyper-G: A universal hypermedia system. *Journal of Educational Multimedia and Hypermedia*, **2**[1]:39–66, 1993. 1, 17, 39

[94] S. A. Kripke. *Naming and necessity*. Wiley-Blackwell, 1981. 25, 27, 56, 57

[95] Y. Labrou and T. Finin. Yahoo! as an ontology: using yahoo! categories to describe documents. In *Proceedings of the eighth international conference on Information and knowledge management*, pages 180–187. ACM, 1999. 40

[96] Jens Lehmann, Robert Isele, Max Jakob, Anja Jentzsch, Dimitris Kontokostas, Pablo N Mendes, Sebastian Hellmann, Mohamed Morsey, Patrick van Kleef, Sören Auer, et al. Dbpedia–a large-scale, multilingual knowledge base extracted from wikipedia. *Semantic Web*, IOS Press, 2014. 175

[97] D. B. Lenat. Cyc: a large-scale investment in knowledge infrastructure. *Communications of the ACM*, ACM, **38**[11]:33–38, 1995. 196

[98] D. B. Lenat, R. V. Guha, K. Pittman, D. Pratt, and M. Shepherd. Cyc: toward programs with common sense. *Communications of the ACM*, ACM, **33**[8]:30–49, 1990. 51

[99] R. Lews. Dereferencing HTTP URIs. *W3C obsolete working draft, Accessed 26/09/2013*, 2007. 71

[100] C. C. Marshall, F. G. Halasz, R. A. Rogers, and W. C. Janssen Jr. Aquanet: a hypertext tool to hold your knowledge in place. In *Proceedings of the 3rd annual ACM conference on Hypertext*, pages 261–275. ACM, 1991. 19

[101] C. C. Marshall, F. M. Shipman III, and J. H. Coombs. VIKI: spatial hypertext supporting emergent structure. In *Proceedings of the 1994 ACM European conference on Hypermedia technology*, pages 13–23. ACM, 1994. 20

[102] Paolo Massa and Conor Hayes. Page-rerank: using trusted links to re-rank authority. In *Web Intelligence, 2005. Proceedings. The 2005 IEEE/WIC/ACM International Conference on*, pages 614–617. IEEE, 2005. 176

[103] M. Masterman. Semantic message detection for machine translation, using an interlingua. In *International Conference on Machine Translation of Languages and Applied Language Analysis*, pages 437–475, 1961. 50

[104] J. McCarthy. *Programs with common sense*. Defense Technical Information Center, 1963. 49, 50

[105] J. McCarthy and P. Hayes. *Some philosophical problems from the standpoint of artificial intelligence*. Stanford University, 1968. 50

[106] John J McCarthy. *Lisp one five programmer's manual*. The MIT Press, 1965. 156

[107] M Mealling and R Daniel. URI resolution services necessary for URN resolution., 2008. 69

[108] N. Meyrowitz. Hypertext - does it reduce cholesterol, too? In *From Memex to hypertext*, pages 287–318. Academic Press Professional, Inc., 1991. 20

[109] J. Nanard and M. Nanard. Using structured types to incorporate knowledge in hypertext. In *Proceedings of the 3rd annual ACM conference on Hypertext*, pages 329–343. ACM, 1991. 19

[110] T. H. Nelson. Complex information processing: a file structure for the complex, the changing and the indeterminate. In *Proceedings of the 1965 20th national conference*, pages 84–100. ACM, 1965. 11, 14, 15

[111] T. H. Nelson. *Literary Machines 93*. Mindful Press, 1993. 15, 16

[112] S. R. Newcomb, N. A. Kipp, and V. T. Newcomb. The 'HyTime': hypermedia/time-based document structuring language. *Communications of the ACM*, ACM, **34**[11]:67–83, 1991. 17

[113] P. J. Nürnberg, J. J. Leggett, and E. R. Schneider. As we should have thought. In *Proceedings of Hypertext '97*, pages 96–101, 1997. 20

[114] P. J. Nürnberg, J. J. Leggett, E. R. Schneider, and J. L. Schnase. Hypermedia operating systems: a new paradigm for computing. In *Proceedings of the the 7th ACM conference on Hypertext*, pages 194–202. ACM, 1996. 20

[115] P. Otlet. The science of bibliography and documentation. *Journal of the American Society for Information*, 1903. 11

[116] P. Otlet. Something about bibliography. *Rayward, W.B. (Trans. and Ed.) The International Organization and Dissemination of Knowledge: Selected Essays of Paul Otlet. Amsterdam: Elsevier*, 1990. 11, 12

[117] L. Page, S. Brin, R. Motwani, and T. Winograd. The pagerank citation ranking: bringing order to the web. Stanford InfoLab, 1999. 1, 105

[118] Heiko Paulheim and Christian Bizer. Improving the quality of linked data using statistical distributions. *International Journal on Semantic Web and Information Systems (IJSWIS)*, IGI Global, **10**[2]:63–86, 2014. 175

[119] A. Pearl. Sun's Link Service: a protocol for open linking. In *Proceedings of the 2nd annual ACM conference on Hypertext*, pages 137–146. ACM, 1989. 21, 36

[120] S. Pemberton. XHTML1.0 the extensible hypertext markup language. *W3C Recommendations*, W3C, page 111, 2000. 35

[121] P. Pirelli, J. Pitkow, and R. Rao. Silk from a sow's ear: extracting usable structures from the web. In *Proceedings of the SIGCHI conference on Human factors in computing systems: common ground*, pages 118–125. ACM, 1996. 40

[122] J. E. Pitkow and R. K. Jones. Supporting the web: a distributed hyperlink database system. *Computer Networks and ISDN Systems*, Elsevier, **28**[7]:981–991, 1996. 40

[123] J. Postel. Simple mail transfer protocol. *Information Sciences*, 1982. 31

[124] J. Postel. Media type registration procedure. Internet Engineering Task Force, 1994. 30

[125] J. Postel and J. Reynolds. File transfer protocol. Internet Engineering Task Force, 1985. 31

[126] E. Prud 'Hommeaux, A. Seaborne, et al. SPARQL query language for RDF. *W3C recommendation*, **15**, 2008. 61

[127] D. Raggett. HTML 3.0 reference specification. *W3C Recommendation*, 1997. 34

[128] D. Raggett, A. Le Hors, and I. Jacobs. HTML 4.01 specification. *W3C recommendation*, 1999. 23

[129] D. Reed, D. McAlpin, P. Davis, N. Sakimura, M. Lindelsee, and G. Wachob. Extensible resource identifier (XRI) syntax v2. 0. *OASIS Standard*, 2005. 70

[130] O. J. Reichman, M. B. Jones, and M.P. Schildhauer. Challenges and opportunities of open data in ecology. *Science(Washington)*, American Association for the Advancement of Science, 1200 New York Avenue, NW Washington DC 20005 USA, **331**[6018]:703–705, 2011. 114

[131] R. H. Richens. Preprogramming for mechanical translation. *Mechanical Translation*, **3**[1]:20–28, 1956. 50

[132] E. Rivlin, R. Botafogo, and B. Shneiderman. Navigating in hyperspace: designing a structure-based toolbox. *Communications of the ACM*, ACM, **37**[2], 1994. 41

[133] A. Rizk and L. Sauter. Multicard: An open hypermedia system. In *Proceedings of the ACM conference on Hypertext*, pages 4–10. ACM, 1992. 36

[134] J. Rohlfs. A theory of interdependent demand for a communications service. *The Bell Journal of Economics and Management Science*, JSTOR, pages 16–37, 1974. 1, 105

[135] L. Rosenfeld and P. Morville. *Information architecture for the World Wide Web*. O'Reilly Media, Inc., 2002. 23

[136] Jordi Sabater and Carles Sierra. Review on computational trust and reputation models. *Artificial intelligence review*, Springer, **24**[1]:33–60, 2005. 176

[137] M. Salvadores, G. Correndo, M. Szomszor, Y. Yang, N. Gibbins, I. Millard, H. Glaser, and N. Shadbolt. Domain-Specific Backlinking Services in the Web of Data. *IEEE/WIC/ACM International Cinference on Web Intelligence and Intelligent Agent Technology(WI-IAT)*, September 2010. 136

[138] L. Sauermann and R. Cyganiak. Cool URIs for the Semantic Web. *W3C Interest Group Note*, 2008. 64

[139] C. Shapiro. *The SNePS semantic network processing system*. State University of New York at Buffalo, Department of Computer Science, 1978. 50

[140] F. M. Shipman III, H. Hsieh, P. Maloor, and J. M. Moore. The visual knowledge builder: a second generation spatial hypertext. In *Proceedings of the 12th ACM conference on Hypertext and Hypermedia*, pages 113–122. ACM, 2001. 20

[141] A. Slavic. Use of the universal decimal classification: A world-wide survey. *Journal of documentation*, Emerald Group Publishing Limited, **64**[2]:211–228, 2008. 11

[142] B. C. Smith. The owl and the electric encyclopedia. *Artificial Intelligence*, Elsevier, **47**[1-3]:251–288, 1991. 51

[143] J. Sowa and A. Borgida. *Principles of semantic networks : explorations in the representation of knowledge*. Morgan Kaufmann, 1991. 41

[144] J. F. Sowa. Semantic networks. *Encyclopedia of Cognitive Science*, Wiley Online Library, 1992. 50

[145] P. Stickler. URIQAI:The URI query agent model-a Semantic Web enabler. Technical report, Technical report, NOKIA, 2004. 70

[146] G. Taubes. Indexing the internet. *Science (New York, NY)*, **269**[5229]:1354, 1995. 40

[147] H. Thompson and D. Orchard. URNs, Namespaces and Registries. *W3C TAG Editor draft*, pages 08–17, 2006. 69

[148] R. H. Trigg and P. M. Irish. Hypertext habitats: experiences of writers in NoteCards. In *Proceedings of the ACM conference on Hypertext*, pages 89–108. ACM, 1987. 19

[149] R. H. Trigg, L. A. Suchman, and F. G. Halasz. Supporting collaboration in notecards. In *Proceedings of the 1986 ACM conference on Computer-supported cooperative work*, pages 153–162. ACM, 1986. 19

[150] H. Van de Sompel, T. Hammond, E. Neylon, and S. L. Weibel. The 'info'URI scheme for information assets with identifiers in public namespaces. *Internet Engineering Task Force*, 2003. 69

[151] J. Van Ossenbruggen, A. Eliëns, and L. Rutledge. The role of XML in open hypermedia systems. In *Technical Report CS-98-01 of Aalborg University, Denmark*, **176**, 1998. 22

[152] N. Webster and J.L. McKechnie. *Webster's new universal unabridged dictionary*. Dorset & Baber, 1983. 14

[153] J. Whitehead. As we do write: hyper-terms for hypertext. *ACM SIGWEB Newsletter*, ACM, **9**[2-3]:8–18, 2000. 19

[154] T.A. Winograd. Towards a procedural understanding of semantics. Stanford University, 1976. 50

[155] G. Wolf. The curse of xanadu. *Wired*, **3**[6]:137, 1995. 16

[156] Y. Yang, P. Singh, J. Yao, C. Yeung, A. Zareian, X. Wang, Z. Cai, M. Salvadores, N. Gibbins, and W. Hall. Distributed human computation framework for Linked Data co-reference resolution. In *The*

Semantic Web: Research and Applications, pages 32–46. Springer, 2011. 125, 172

[157] N. Yankelovich, B.J. Haan, N.K. Meyrowitz, and S.M. Drucker. Intermedia: The concept and the construction of a seamless information environment. *Computer*, IEEE, **21**[1]:81–96, 1988. 17, 19