Institute for Visualization and Interactive Systems

University of Stuttgart
Universitätsstraße 38
D–70569 Stuttgart

Master thesis

Filter (Dials | ...)

Jayaprashanthini Rajamani

Course of Study:	M.Sc.InformationTechnology(INFOTECH)
Examiner:	Prof.Dr. Thomas Ertl
Supervisor:	Dipl.-Inf. Florian Haag.
Commenced:	2014-09-15
Completed:	2015-03-17
CR-Classification:	H.2.3, H.2.4, H.3.3, H.5.2

Abstract

An increasing number of data providing organizations publish their unique information as Linked data which is available in the Semantic Web in RDF (Resource Distribution Framework) format. RDF is a standard model describing web information that is understandable to computer applications. For users without much technical knowledge, information visualization provides support to interpret the structure of underlying data in the web, understand their relationships, form queries and extract more interesting information. This thesis proposes a visual query technique focusing on the visualization of overall availability of data. The prototype visualization tool developed shows the available groups of data items and their size according to different filtering options. The thesis report starts with a clear description of the research problem and an analysis of the efforts made so far in the field for finding a solution. A survey of the the related work is presented to find the potential for the visualization. The Filter Dials, a novel visualization concept serves as the base work and a starting point and that the evolved new technique attempts to overcome some of the notable drawbacks of the Dials. The thesis also proposes a new visual query technique that aims to achieve the goal of visually presenting an overview of large and complex data available in the web. A prototypical implementation of the visualization tool is done and evaluated by users and conclusions are drawn with the applications, advantages, disadvantages and the possible future directions in the proposed method.

Acknowledgment

First and foremost I feel grateful for almighty for the immense blessings showered on me that made this two and half years journey of my master studies possible.

I am grateful to Prof.Dr. Thomas Ertl, who gave me the opportunity to work on this thesis at VIS institute. It gives me pleasure in acknowledging the motivation and guidance of my thesis supervisor Mr. Florian Haag through-out my thesis work. I thank you for being patient in answering my questions quickly even if you are away from work. The detailed and constructive suggestions you gave were crucial for the continuous progress of this thesis.

Personally I would like to thank my family for encouraging me to study abroad. I am sure this would be the greatest joy to them to see their prayers turning into reality now. Thanks to all my friends for the motivation and understanding.

I thank my friends in Germany who made this stay a very enriching experience even though I did not have much time to spend with them. I thank all the people who helped directly and indirectly to complete this thesis.

Contents

List of Figures

Listings

List of Abbreviations

W3C ..... World Wide Web Consortium

LOD ..... Linked Open Data

URL ..... Uniform Resource Locator

RDF ..... Resource Description Framework

SPARQL ..... SPARQL Protocol and RDF Query Language

DBLP ..... Digital Bibliographic Library Browser

NLD ..... Node-Link Diagram

GUI ..... Graphical User Interface

SQL ..... Structured Query Language

OQL ..... Object Query Language

RQL ..... Resource Query Language

NLI ..... Natural Language Interface

NZDL ..... New Zealand Digital Library

EPDM ..... Electronic Product Data Management

IDE ..... Integrated Development Environment

WPF ..... Windows Presentation Foundation

XAML ..... Extensible Application Markup Language

RDFS ..... Resource Description Framework Schema

OWL ..... Web Ontology Language

DOPE ..... Drug Ontology Project for Elsevier

Chapter 1

Introduction

The amount of information published in the internet is growing continuously. The Semantic Web [BLHL+01] is an initiative of the World-Wide Web Consortium (W3C)[1]. Its vision is to provide sufficient description about resources on the Web sometimes known as meta-data and also provide supporting tools to use the meta-data so that machines can readily carry out sophisticated tasks for users while exploring information on the web.

A simple example where the need of Semantic Web is demonstrated well is in finding documents on the Web written by a particular author. Conventional search engines looks for the textual content of the document and will respond with the web URL links to all the documents where the phrase appears as an author, reference or citation. Whereas with the Semantic Web, a description stating its author, time of creation, subject etc is annotated with all the documents and only appropriate documents created by that author are returned.

The Linked Open Data (LOD) cloud[2] [CJ14], also referred to as the Web of linked data, can be understood as a realization of the Semantic Web. As of August 2014, LOD diagram in Figure 1.1 contained more than 500 datasets from varied domains such as Government, Media, Life sciences, Geographic etc. One of the popular datasets within the LOD cloud is the DBpedia[3]. The Resource Description Framework (RDF) data model[4] is used to publish this structured information on the web. It links data from

[1] http://www.w3.org
[2] http://lod-cloud.net/
[3] http://dbpedia.org/
[4] http://www.w3.org/RDF/

different sources.

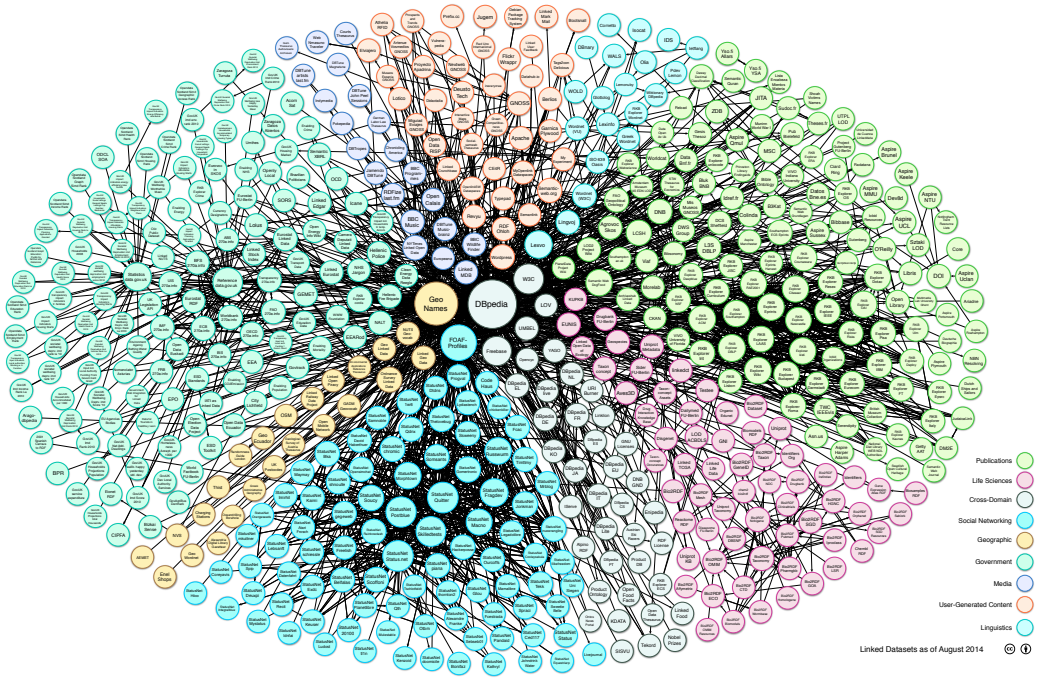


Figure 1.1: The Linking Open Data cloud diagram. Image source: http://lod-cloud.net/

Querying is the major interaction mode with the Semantic Web to make the best use of it. SPARQL Protocol and RDF Query Language (SPARQL) [PS+08] is a standard and a formal RDF querying language used to extract desired information from the RDF graphs and make the RDF data accessible via standard SPARQL end points.

The Web is already an important information source for students and researchers, so managing the information space is important in academia in order to reduce wastage of efforts in information organization and discovery. It is where the usefulness of a Semantic Web browser is felt. One such example is the Haystack Semantic Web-browser from MIT[5]. The Semantic Web technology also enhances the interaction of academic and research communities across the world in their collaborative research and has also made its impact on digital libraries and E-learning.

[5]http://haystack.csail.mit.edu/

Information visualization is becoming increasingly important in Semantic Web tools to cope with the information overload. Interactive visualizations transforms the Semantic Web content into meaningful visual representations to efficiently navigate and explore large data repositories which would otherwise be arduous in traditional textual environments. Not only that, it also makes extraction of information process user friendly for casual users and make the search a joyous experience without having to perform complex querying tasks. There are various popular information visualization tools and techniques such as IsaViz[6], mspace [SRO+05], Exhibit[7], Jambalaya [SMS+01]. Some of these techniques will be discussed in detail in the following chapters.

[6]http://www.w3.org/2001/11/IsaViz/
[7]http://simile-widgets.org/exhibit/

1.1 Problem Description

RDF data is available in the form of directed, labeled graphs that are large and complex. RDF graphs basically consist of a large number of triples with subject, object and predicate. RDF data is highly interconnected and as such understanding the RDF graphs or breaking them into desirable subsets and interpreting meaningful information is difficult. There are several tools and techniques like RDF-Gravity [GW04], GrapViz [EGK+02], Haystack [QHK03] etc. that help in seeking specific information from the RDF datasets and for navigating across data links.

Many of the available RDF querying methods aim to retrieve specific desired information from the large data resource in the web. For instance, when searching for a publication in Faceted DBLP[8], we get information like author, year, place etc about that specific publication item. It would be of much help for exploring if the users were able to grasp information like how many similar publications exist or what are the categories under which the publications can be grouped. It is not so easy to identify what and how much data is available. Getting an overall impression on the availability of the data is important when querying large unknown data groups in order to get meaningful results. This is also useful when they want to continue their search and narrow down the result set by choosing appropriate filtering attributes.

Obtaining an overview of how much data items are present based on combinations of filter criteria is still an active research problem [DKS06, YKSJ08]. There has to be some way to know what data is actually contained in the RDF graphs and also common filter combinations must be found so that the resulting subset is not immediately reduced to an empty set. The Filter Dials approach [HE14] was a first attempt to create a visualization that allows for dynamic combination of filter criteria to interactively get an idea of what data is contained in RDF graphs. Figure 1.2 shows that several filter dials are linked to each other.

[8] dblp.l3s.de/

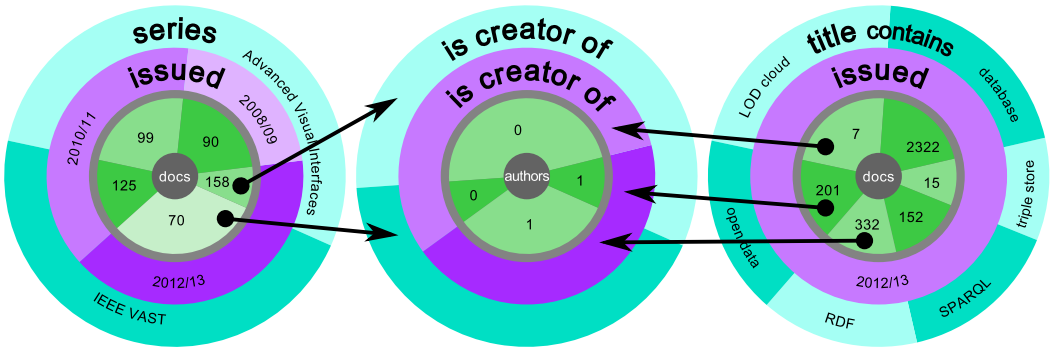


Figure 1.2: An Example of several linked Filter Dials. (Graphic courtesy of Florian Haag/VIS.)

The main aim of the Filter Dials is to see how much data is available in an unknown dataset. Applying combinations of filter criteria to the data reduces it into one or more subsets whose size can be easily recognized. These subsets can then be used in other filter dials for narrowing the obtained result set. In Filter Dials structure, there are concentric rings representing the filtering attributes, inner section as the resulting subset and the surrounding area of the rings as the whole data set. The users can sub-divide the rings into several sections for combining several filter criteria. In this way the Filter Dials visualization concept gives an overall impression of the availability of the data items.

The current Filter Dials technique is in a growing phase. With the solid foundation of the idea of filtering, there is a lot of scope for further enhancing the concept of the dials [HE14]. From the opinion of the fellow researchers and from the feedback obtained during a presentation of a poster on the Filter Dials concept, several important points that need improvement have been noticed and listed here. The following are considered to be the main limitations with the existing filter dials:

- Shape and arrangement of dials – The existing filter has circular rings for attribute selection. Because of this structure, it is difficult to display all the chosen attribute names in the respective rings. Hence it is also not easy for the user to recognize which ring has which attribute by just seeing the dial.
- Result preview – The result set displays only numbers in the ring sectors and does not provide a result preview.

- A visual representation of an overview of available datasets and possible attributes is missing. The non-expert or the casual users will have no idea what to query.
- Color coding techniques could be made use of in the tool for rings and attribute selection dialogue boxes.
- A separate dialogue box popup is used for configuring the ring attributes. The users will have quicker interaction with the tool if somehow it is possible to incorporate the configuration of the ring attributes within the ring itself and not as a separate dialogue box.
- It does not seem to include any support for the users for deciding among the available attributes for querying in order to refine the search and the query results.

1.2 Goal of the Project

The aim of the thesis work is to do a theoretical study of the important works done in the field to solve the above stated research problem and to design and implement a prototype of a visualization tool. The visualization query concept is intended to provide an intuitive visual representation of the web data which would make the further information exploration process easier through faster and easier comprehension of large datasets.

An overview of the available data showing their main types, properties and relationships between them is needed for the user to know what they can query and explore from the dataset and to make sure that the obtained result is meaningful (not a null set). By presenting an overview-visualization in an orderly and user-controlled way, the users can save time by avoiding trial and error querying and exploration of the complete dataset.

The Filter Dials concept sets the background of the project and serves to be a starting point. This project focuses on overcoming some of the limitations of the current filter dials. Of particular interests are the structure of the Filter Dials. As discussed in the previous section, one of the open issues to be dealt with in the Filter Dials concept is the shape and arrangement of the filters. Other possible better visual filters are to be designed.

There are also a few things to be taken care of while deciding the technology behind the visualization tool so that the efficiency of the tool is not disturbed. For example, the time taken for retrieving large amount of data and converting it into visual representation is usually high and it is also computationally expensive. The tool might become slow and as a consequence, the user's interactiveness with the tool might be disturbed. Similarly there is a possibility that the display screen gets cluttered and loses presentability and thus does not the serve the primary purpose of the visualization. These secondary aspects will also be considered in the thesis beside the core aspects of the visualization itself.

1.3 Remainder of the Thesis

The thesis is divided into the following chapters:

- Chapter 2 presents the related works done in this topic with a discussion of their short-comings in fulfilling the goal of this project.
- Chapter 3 elaborates on the major difficulties encountered so far in trying to solve this technical problem, few existing visualization tools and the proposed visualization technique.
- Chapter 4 describes the functionalities of the visualization tool and the Graphical User Interface (GUI) elements in detail.
- Chapter 5 contains the method and tools to implement the visualization prototype, the process architecture, Class diagrams and the SPARQL query structures.
- Chapter 6 provides evaluation and results from the user's experiment with the prototype.
- Chapter 7 is the conclusion which summarizes the work within this thesis along with any scope for future work.

1.4 Meta Information

This section briefs the software, tools and resources that is used for writing the thesis document. This technical document is created using LaTeX system. The list of references is given in BibTex format. The IDE used is

the TexMaker which is an open source cross-platform LaTeX editor that integrates many tools needed to develop documents with LaTeX . To input and edit entries in the .bib file, the JabRef reference manager tool is used. The graphics that are embedded in this document are designed and drawn using inkscape which is a freely available flexible drawing tool. The class diagram of the project in Chapter 5 is created based on the .cd file that is generated using the SharpDevelop IDE[9].

[9]http://www.icsharpcode.net/

Chapter 2

Related work

In the previous chapter, the background theoretical concepts have been introduced along with the problem statement. This chapter presents a survey of related work. Various interesting related work that deal with RDF data in a meaningful and competent way has been found to be published in the field of Semantic data visualization. Understanding those researches and literature would help to propose new visualization query techniques and visual structures and to augment existing designs and to check if the concept presented in this work has been done before in the same way. The following sections will provide an analysis of the state of the art work related to the overall data content visualization and navigation in large datasets along with discussion about how the proposed technique differs from the other similar works.

2.1 Faceted approach

tFacet [BH11] and gFacet [HZL08] are tools for visualization and faceted exploration of linked web data sources via SPARQL endpoints. tFacet is a faceted hierarchical exploration tool for seeking information in the semantic web with a notion of superfacet and subfacets. With tFacet, it becomes difficult to switch the view from the current facet class to different classes and to retrieve instances.

gFacet 2.1 is a tool that combines pivoting and graph based visualization. With gFacet, first a class is selected using key words search and then it is pivoted to a related class and thus forming faceted relationship structure between data items of different related classes. The instances in the classes formed through pivoted relation are filtered according to

the selected instance in the first class. Similar to gFacet is the Jambalaya project [SMS+01] which also uses a graph representation of ontologies.

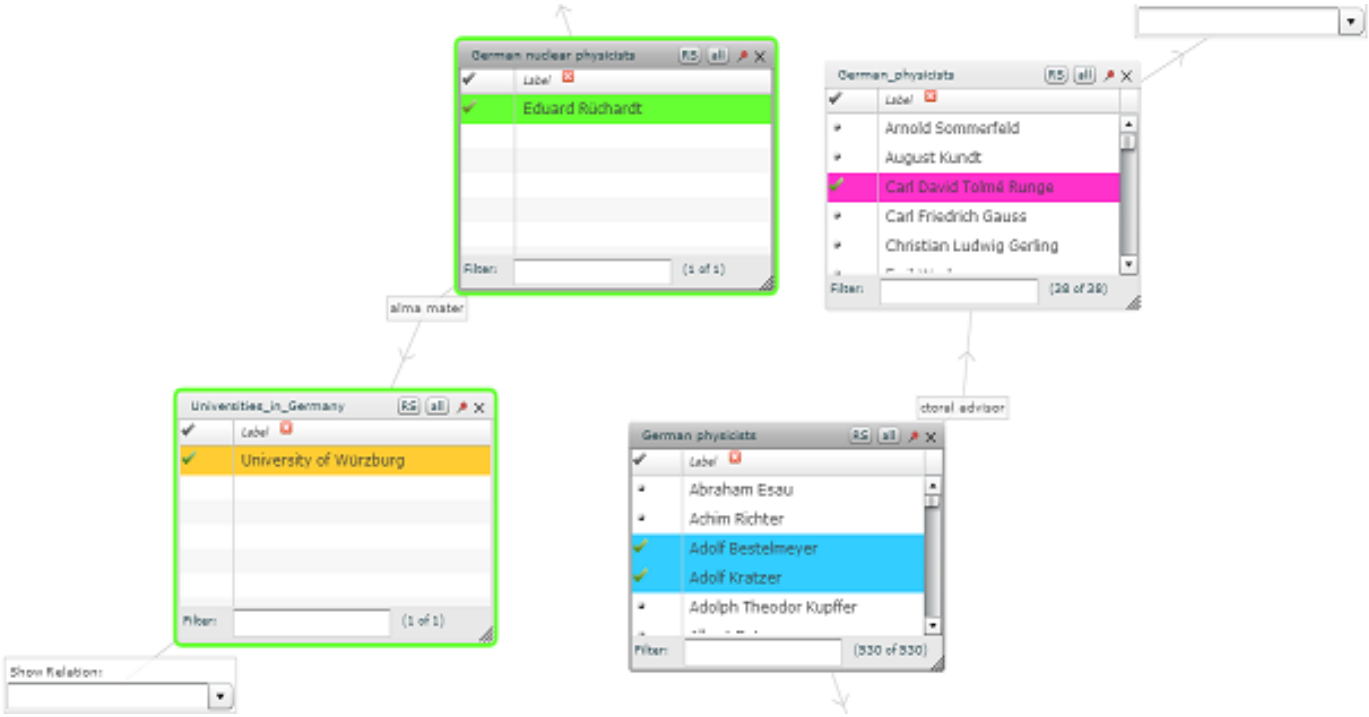


Figure 2.1: gFacet screenshot

Another interesting example of faceted exploration approach is EXHIBIT[1]. EXHIBIT an open source tool, part of the SMILE project, which helps website developers to create web pages with advanced text search and filtering functionalities for faceted browsing. It makes use of interactive maps, timelines, thumbnails, tiles and other visualizations appropriate to the type of database.

[1]http://simile-widgets.org/exhibit/

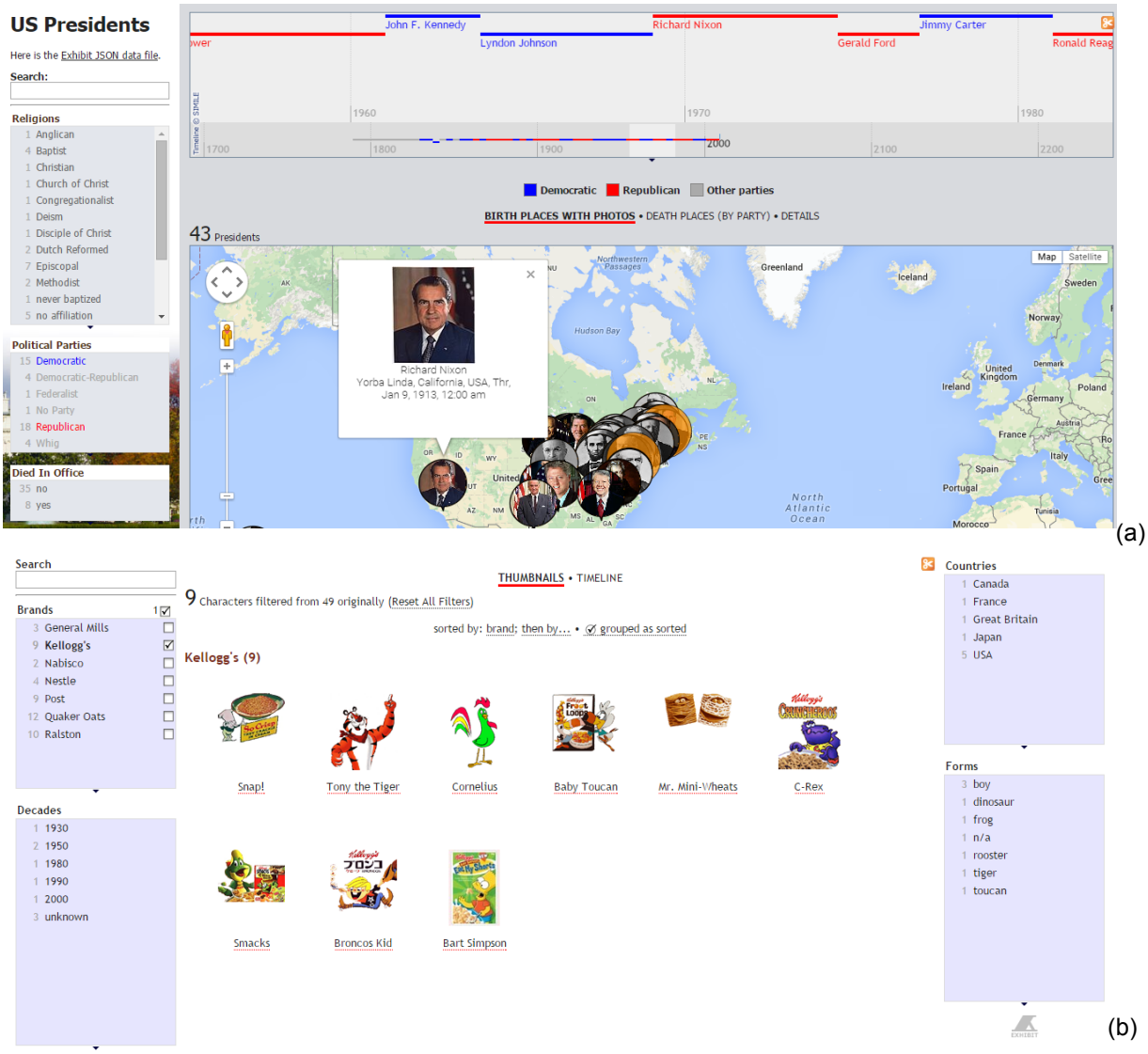


Figure 2.2: Sreenshot of online tool EXHIBIT with its different views: (a) TimeLine and map view. (b) Thumbnail view

Many semantic web browsers that are available help users to search and retrieve specific things from the web using keywords. The mSpace framework [SOR+05] goes a step further and allows knowledge building with the semantic web by associating one data item to others. It provides a solution to overcome the limitations of browsing systems. mSpace helps people build knowledge from exploring relationships by organizing an

information space in the notion of Slices visualized in two dimensional multi-panel display. Slices are directional column-facets represented as a list of values. Selecting a value in a slice will update the values displayed in the next slice that are related to each other.

the Sculpture project [SGL+05] uses the mSpace framework to develop an image search interface to navigate a large collection using hierarchical faceted metadata. Similarly, the web based mSpace Classical Music Explorer [SRO+05] combines the features of an iTunes Music Store with Google search for accessing the classical music domain.

2.2 Tabulator

The Tabulator project [BLCC+06] is a generic semantic web browser and editor for linked data on the web. It is helpful for browsing RDF data on the web. This semantic web browser presents different types of views such as tree, table, map, calendar, timeline to provide an outline of the RDF data from different domains. To see what information is available, a table of predicate/object pairs is used where the user may recursively open a nested view of the properties of an object. This mode of operation of the Tabulator is known as the outliner mode (Figure: 2.4). The outliner mode view is a straightforward metaphor of a nested hierarchical tree view.

The Map view (Figure 2.3:a) displays query results on a geographical view in order to get meaningful location specific information about the data items by making use of the latitudes and longitudes values from a query result. Calendar view (Figure:2.3:b) helps the user to visually access information about events and duration. By using the RDF iCalendar-equivalent ontology[2] it shows the duration of the events at different scales (year, month, or day). With these modes, Tabulator provides a smooth interoperability between a generic views and an application-specific one. However the users should have knowledge about the ontologies while working with data from previously unknown domains. Another similar experimental FireFox extension browser is Piggy Bank [HMK05].

[2]http://www.w3.org/2002/12/cal/icaltzd

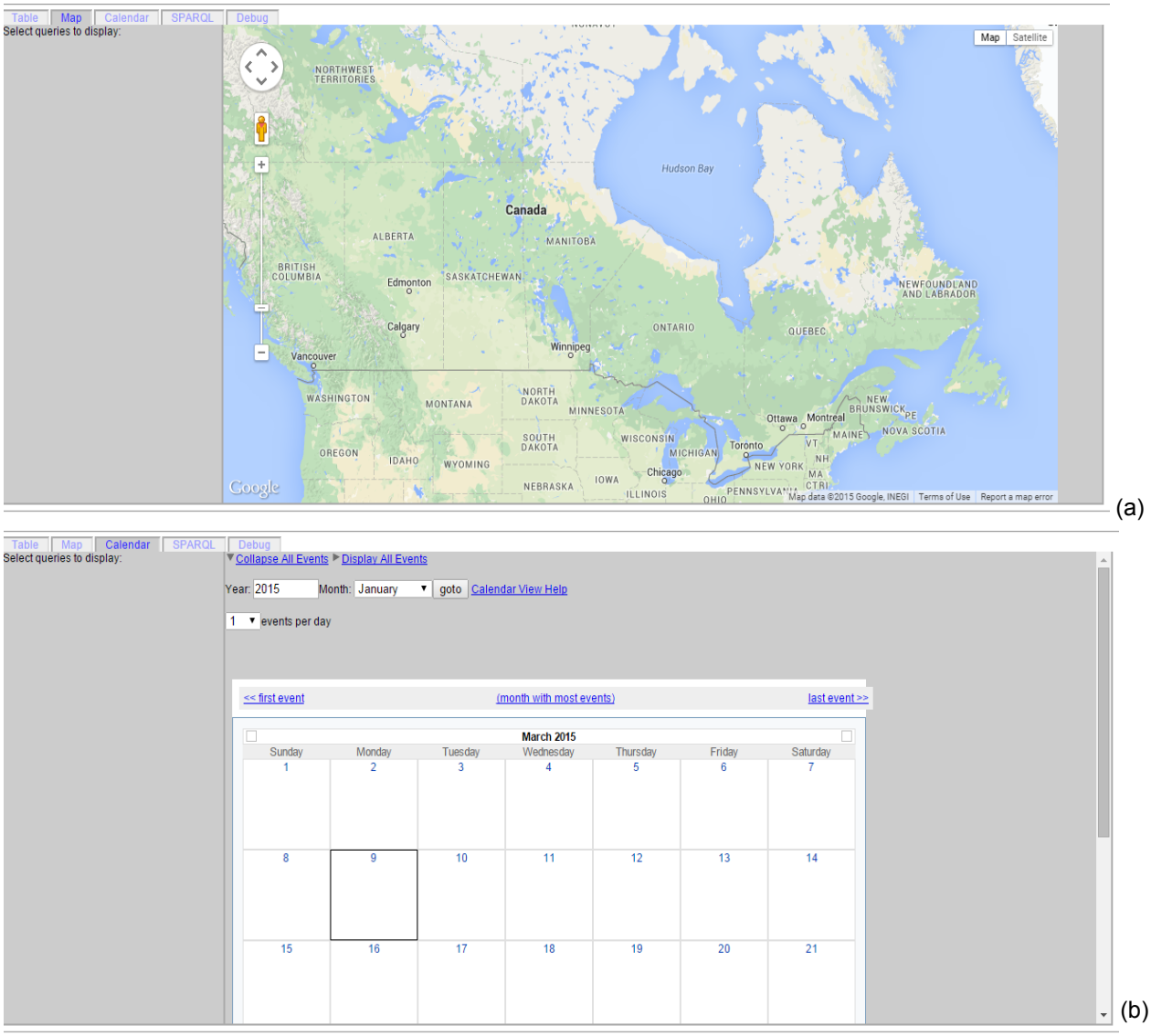


Figure 2.3: Tabulator (a) Map View and (b) Calendar View

Figure 2.4: Tabulator Outliner mode of View.

2.3 RelFinder

RelFinder [LHSZ10] is also a Linked data browser with visualization options that allows for a very large amount of data to be visualized and filtered. It provides a graphical structure showing links between selected individual resources by extracting complex relationships between them. It follows the simple subject-object relationship schema in the graph visualization. It is helpful to visualize and explore parts of the content of an RDF dataset.

In RelFilder (Figure: 2.5) the overview of data is provided by graph visualization and information about the selected nodes are maintained in a table view with filtering features. RelFinder helps users in finding possible relationships between selected objects of interest from different Linked Data repositories. The results are updated dynamically in each iteration.

The graph can be reshaped by the user and the entities in the graph can be clicked to know more information about them. However this is a computationally intensive activity. This type of graphical representation will be practically difficult for the users to comprehend the information when data space becomes larger and complex relationships exists between the resources.

OntoRelFinder [SKL+11] is also a similar Relationship path Finder that visualizes all possible paths between two or more instances of ontology database. OntoRelFinder has been said to overcome the system response time issue in finding all relationship among the data items that was a problem in RelFinder. Other similar tools are LodLive [CMA12], a graph-based linked open data live browser, RDF Gravity [GW04] (RDF Graph Visualization Tool), a visual browser for graphically visualizing RDF and OWL[3] data.

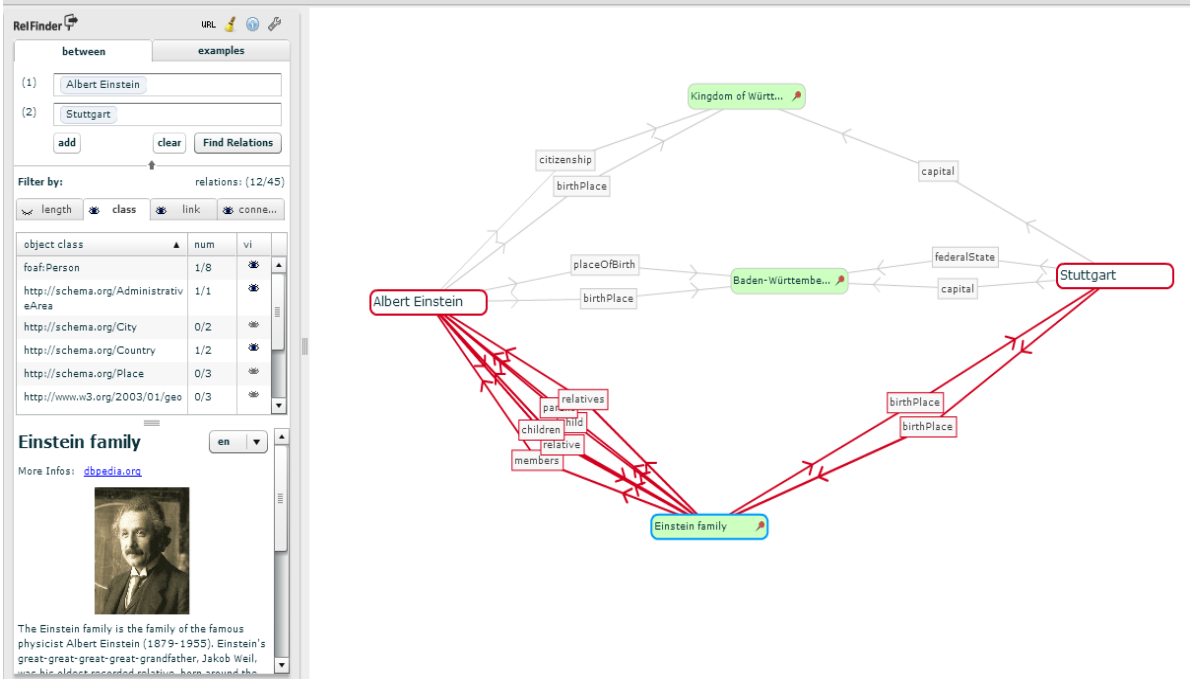


Figure 2.5: RelFinder screenshot showing relationships in a RDF dataset

[3]http://www.w3.org/2001/sw/wiki/OWL

2.4 Clustering Technique

Clustering is an important technique for understanding of large multi-dimensional datasets [FSVH06]. Clustering refers to the grouping of items according to some measure of similarity. For example, collection of text documents are clustered typically using words and phrases [CKPT92]. The DOPE project [SVHDW+04], Clusty.com[4], Yippy.com[5] are few of the interesting implementation of clustering.

One of the advantages of clustering is that it is automatable. It is useful for refining vague and ambiguous searches by presenting the users with the dominant categories of the result. Nonetheless, the algorithms used in the clustering techniques are not always often generate groups which are hard to label in ways that are meaningful for users [Hea06].

2.5 Discussion

The use of a graphical representation in tools like gFacet makes the user interface difficult to manage. It is because common users may not be aware of the underlying graph structure. Moreover, by using these boxes and links (Figure: 2.1) more screen space is wasted. Also, these these visualization structures users are not really aware of what queries they are actually asking for from the database. An initial overview of the structure of the dataset to help users know from where they can start their graph construction is missing here.

The Faceted meta-data browsing is an appealing replacement to traditional keyword searching and browsing in text based search engines. These faceted interfaces display the data in hierarchical categories. Facets can be single or multi-level hierarchies. It provides better support for exploration, discovery and iterative query refinement and navigation in multi-dimensional data spaces. The information search process in a faceted browsing basically includes two things: When an item is selected from a facet, it is a disjunction (OR) of all the other items selected from the same facet and at the same time a conjunction (AND) with all the items selected from other facets. This way it provides possible options for different filter criteria. However, the main drawback of the faceted meta-data approach

[4]http://clusty.com/
[5]http://yippy.com/

is that it is bound to a hierarchical organization whereas different users have varied perspectives in organizing data in real world.

The predominant task of the RelFinder browser-based system is to detect unknown relationship or patterns among RDF elements and visualize them as a network for searching and exploring individual data elements in the Web of Data. In several semantic browsing applications like RelFinder the visualization rely on the explicit semantic relations in the linked data structure without exploring any additional knowledge for contextualization.

In Tabulator-like semantic web browsers, because of the absence of knowledge about the domains from where the information is retrieved, the browsing support for users will not be effective especially if the information source is heterogeneous, containing information in several ontologies.

The approaches like Relfinder and Clustering techniques have a limited scalability in terms of the number of resources in the dataset. They do not show which combinations of filters will result in desirable results. These type of interfaces provide support for querying metadata that describes the information in the semantic web. However, the user must be aware of the data space, what to ask for and where to look for before starting the exploration process.

Chapter 3

Background

The Semantic Web Data Visualization is a growing research topic and various attempts have been made to solve the problem of presenting an efficient overview technique. Some of these techniques could also be considered for reuse with modification in the new visual query technique. Starting from the basic formal query languages to the advanced graphical presentation tools, several methods have been employed in these query techniques to present the overall content of the dataset. Naturally most of these tools and techniques although having been useful in information visualization with various domains also have few limitations in fulfilling this overview task of visualization. Understanding these difficulties would also definitely help us to gain an insight of the research problem in depth and to come up with a convincing solution. How this analysis was helpful in designing the new visual query concept, how far those concepts were felt useful, as a consequence how it lead to the evolution of the new implemented technique have also been explained in detail in the following sections. A new visual design structure called Filter Layers is also proposed here.

3.1 Difficulties Concerning the Overview Visualization Concepts

The concept of Information visualization [YaKSJ07, CMS09, DKS07] has proved that it is faster to recognize the amount of data available and understand the relationship between them in a visual structure than by scanning the textual information. The problem with the traditional formal textual query languages like SQL, OQL [Clu98], RQL [KAC+02], SPARQL,

SquishQL [MSR02] is that it is difficult for the common user to construct meaningful queries without any knowledge of the database classes, attributes and their relation structure [BR92, DK09]. Moreover this kind of a typed or declarative text based search is more useful for fine grained and in depth exploration of given data whereas, we are now concerned only about the overview visualization.

Natural Language Interfaces (NLI) [KBZ06, KBF07, DTB08], prove to be superior to formal query languages in terms of their usability making it easier for the casual end users who lack expertise in complex query constructs [KB07]. There are several NLIs that work upon different types of query languages and available in various natural languages like English, German, French etc. These interfaces are simple to use with free and unrestricted query languages and are similar to common search engines. However for using an NLI for successful querying, users first need to know what is possible to ask for. The formulation of the underlying query statements in these systems still need further development to achieve this goal.

With the textual query languages there is a possibility that users make syntactic and semantic errors while forming queries and thus take a longer time to formulate and refine queries. This problem of text based query languages can be overcome by using Venn diagrams which serves to be a convenient alternative medium for boolean query specification. One such graphical user interface that makes use of Venn diagram like structures is VQuery [Jon98]. VQuery interface is designed specifically for New Zealand Digital Library (NZDL) database. Venn diagrams are used to dynamically form visual queries with boolean operations such as AND and OR on the whole dataset and also to preview and refine the queries. However there is not yet any appropriate method identified to represent the universal set of data in the Venn diagram structure in order to include all the boolean expressions in visual form.

Form based query interfaces [Zlo77, AMH10, BE06] are mainly used for easy accessing of databases. These form based query interfaces provide tools to assist the users with limited technical knowledge to build complex queries in an intuitive way. Query expressions are formulated as entries in a table. The disadvantage here is that without a clear understanding of the underlying data it is frustrating for the users to determine the availability of the data with a constrained expressiveness of the query in a table skeleton.

In graph based query language interfaces [ACS90, PK95, HGR95, FH06], the schema of the dataset is presented graphically based on their entity relationships. To start the information exploration, a query is constructed on only interesting subsets selected from the schema or classes. This type of graphical representation is useful for understanding the database schema. It also simplifies the extraction of the subschema of interest by means of a graphical friendly interaction with the system. This method is however is not supportive for users to comprehend the information in less time if the graph becomes large and complex when dealing with bigger datasets.

After careful analysis of the several difficulties that the so far existing techniques have to face for meeting the goal of gaining an observation of the overall data content, few things were made clear. Browsing is not a suitable method for accessing large datasets since it is arduous for users to stroll through long list of results. In formal querying methods as seen above such as SPARQL which includes forming queries with specific properties and sending them to databases does not guarantee meaningful results. The visualization strategies presented above (NLIs, form based interfaces, graph representation) does not help the users to know what to ask for or to ask in a better way. In large information spaces like the World Wide Web, the information is abstract in many cases cannot be naturally mapped to the obvious physical world representation (e.g. geographical space). A key research problem in information visualization is to discover new visual metaphors for representing the huge quantities of information to make sense of and reach decisions in short time based on the meta data describing the data items.

3.2 Available Solutions to specific domains

There are variety of information visualization structures and proposed designs and techniques. It is appropriate to analyze the differences in these techniques to come up with new possibilities. It is notable that as there are varied domains and different types of data, there are appropriate visualization methods matching to specific domains. For instance, Google maps are solely utilized for exploring geographical information, temporal data is well presented in timeline format, quantitative data is depicted in charts (pie, bar etc) for easy comparison. In the following sub-sections, few interesting graphical interfaces and their visualization strategies are described. A glimpse of the type of graphical attributes that these visualization techniques use and how they are used for depicting the usability

and availability of the data in the dataset and the relationship between the data items are discussed.

3.2.1 EPDM Domain

Many companies today adapt an EPDM (Electronic Product Data Management) system for handling their product information effectively for various industrial and management tasks. These product items are often visualized in a hierarchical data space. A special visualization method proposed in [GKR08] for an EPDM system allows the users to identify the data items of different types and to interact with them.

Figure 3.1 shows an interactive user interface of the EPDM visualization prototype that shows a pixel display of about 1500 data objects. The pixel representation technique used here is a popular approach for visualizing a very large amount of data and making best possible use of the provided screen space by mapping each data item to a single pixel. Hence it allows for an overview visualization of the data and to identify clusters and relationship between data items.

The technique adapted here for visualizing the EPDM data can also be applied in other domains like social networks and company organization structures. However there are certain limitations with the pixel oriented representation. In representing each data value as a pixel and arranging these pixels a number of optimization problems have to be solved. Moreover the users are rarely interested to identify single data items or explore specific data items for which other better exploration and browsing techniques exists.

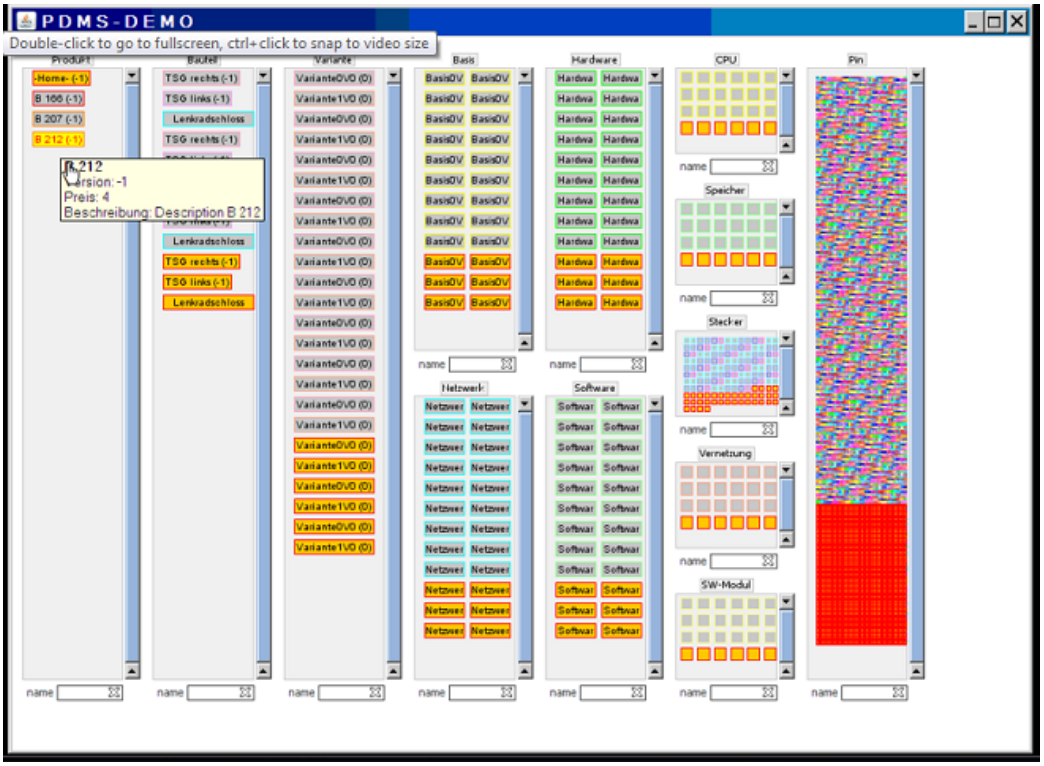


Figure 3.1: Interactive EPDM visualization prototype with pixel representation technique. Image source: HCI Konstanz(http://hci.uni-konstanz.de/epdm/)

3.2.2 TimeLine Data

Visualization of timeline data has its application in wide areas such as logging the medical record, historical events etc. An important aspect in visualization of temporal events is the navigation between different timelines. TimeSlice [ZDFB12] is one such timeline visualization tool where users can explore the temporal events and compare them easily by faceted attributes. Figure 3.2 shows a sreenshot of TimeSlice, an interactive visualization system for faceted temporal events data. In TimeSlice, a faceted

tree structure is used for constructing queries, filtering and visualizing the results. It is an interactive faceted browser and an exploration tool. The tree nodes represent the attributes and the levels of the tree indicate the different facets of data. This dynamic filtering tree structure proves to be an efficient technique for representing different timelines of data categorized on different sets of attributes.

Figure 3.2: Timeslice screenshot Image source: Jian Zhao Homepage(http://www.cs.toronto.edu/ jianzhao/)

3.2.3 Mambo:Musical Browser

User interface techniques like FacetZoom [DFW08], FacetFolders [WD08] and Mambo [DF07] tool are developed for the purpose of faceted media browsing. Objects are categorized according to types such as artist, genre, time, publisher etc. to gain an overview about a large collection of media data and to further browse, search or filter specific items of interest. The Mambo musical browser overcomes the problem of scrolling down long lists of items in the musical database with the help of FacetZoom widget.

The FacetZoom widget consists of multiple stacks of one dimensional Tree-map visualization structures which represent a few subset levels of the whole hierarchy. The widgets allow to traverse across different hierarchical levels of data items based on the filtering dimension. As a result of its scalability to various display sizes, users can browse data from mobile devices using pan and zoom navigation and tap-centre interaction.

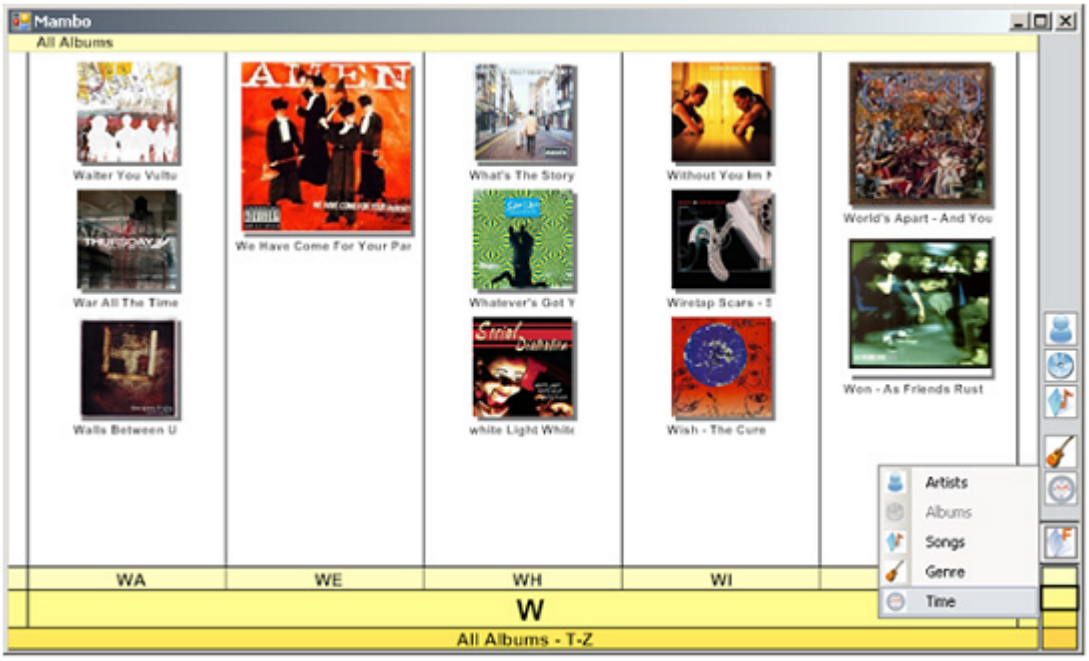


Figure 3.3: Mambo screenshot. Image source: MMT(http://www.mmt.inf.tu-dresden.de/Forschung/Projekte/INPERIC/)

3.3 The idea of Filter Layers

To emphasize the fact again, the major issues that are to be addressed in this thesis are the radial layout of the Filter Dials, filter combinations, guidance on which attributes to select and support for remembering the found results. With an intention to overcome these issues and find a solution to enhance the filter dials concepts, the idea of filter layers was considered. At first it appeared to be a convincing method however it does not prove itself to be a good solution after careful analysis and comparison with the related works. The idea is presented in detail in this section and also its limitations with this concept in coming close enough to address the main issues of this thesis discussed in section 1.2.

As seen in the previous sections the shape and the arrangement of the dials of the filter needs an alternative. It is because with the circular rings and the sections of the dial, it is not possible to graphically convey necessary details of different parts of the dials **??**. That is, details like which attribute(s) are selected by the user for forming filtering combinations with rings, the range or the value of the property in each section of a ring, num-

ber of elements obtained in the result set were not included in the dial structure.

Instead of circular rings, forming a horizontal layer will allow enough space to clearly view the different sections of filter layer and recognize how much data is available in each section by looking at the size of the section and also display any details on the result set like number of objects. Moreover horizontal layers are better than circular ones for timeline databases. Most often users are interested to know time related information of the data. For example, when a scientific paper is published or which are the conferences that took place in a particular period of time. For this kind of classification of data in timeline vertical layers can also be thought of as in Timeslice 3.2.2. A conceptual diagram of the filter layers is shown in Figure 3.4.

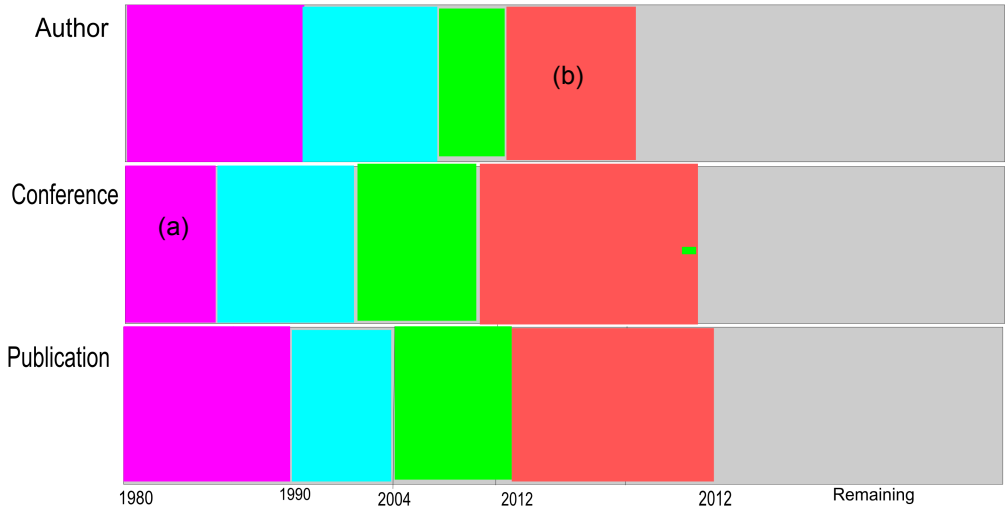


Figure 3.4: An instance of Filter layer structure

In the filter dial, rings are added in the direction of inner core to the outer periphery of the circular dial to introduce filtering attributes to the dataset. This method is extended here in the new filter structure too as layers can be added one above the other to construct the filter combinations so that layers represents the chosen filter attributes, it may be a type, class or numerical or object property. However, the filtering process is done gradually layer by layer. The layers added upon can be sectioned depend-

ing on the sections formed in the first bottom-most layer which will be the first chosen filtering property. Second layer depending on the first, third layer sectioned based on the first and the second layer and so on. In this way a combination of filter criteria is possible.

For example in Figure 3.4 there are three layers of filter types: *Publication*, *Conference* and *Author*. The bottom-most layer representing the data objects of type publication is sectioned based on the numerical attribute year/time period between a range or a single value. This initial selection of different range of the attributes made by the user could be applied to the filter layers added above the first layer. Here in this example when *Conference* is chosen as the second filter layer by the user this layer gets subdivided into horizontal sections according to the sections formed in the previous layer. The magenta color section 3.4(a) of the second layer indicates the result set obtained upon the application of two filtering criterias: *Year* and *Publication*. That is it represents all the conferences that constitutes all the publications belonging between the time period 1980-1990.

Similarly sections of the third layer are considered to have applied with the first and second layers as a filtering criteria. The orange color section 3.4(b) of the third or the top layer represents the authors who participated in the conference which includes the publications created in the year 2012. Different colors are used for each section of a layer to distinguish the categories of that type attribute. And sections of different layers that relate to each other are same in color to denote the corresponding filter criteria and the result. This explains the possible way that is found for creating filter combinations and result sets. Now here arises a question whether this type of result preview as well as filter combination in a single view supports dynamic filtering and interactive visualization.

The sizes of the sections depend on the amount of objects present in that category, however the size is not relative to the size of the whole dataset. It is because the data items present in one section and the other are not exclusive to themselves. The Remaining portion does not necessarily represent the rest of the dataset. A data item may appear to be present more than once in several sections. For example the possibilities of authors having published their scientific papers in multiple time periods/year range has not been taken into consideration. The union of two or more sections is difficult to be depicted in this type of layer structure. Hence, there arises a need for a separate supporting visual structure along

with this filter layers for visually presenting the overview of how much data is present in the database. A simple bar or pie chart should be made as a complementary part of this visualization structure for this purpose.

One of the important goals to be achieved in this thesis is the overview feature. The new visualization technique is expected to fulfill the criteria of visually presenting an overview of the available dataset, the amount of data available, how the data is organized and what are the possible queries that can be made on an unknown dataset. Although this idea of filter layers helps to overcome the inherent problems in the filter dial structure for filter combinations and result preview, unfortunately it does not seem to provide an overview of the entire collection of data. Hence, the idea of filter layers will not be taken further for implementation and a new visual concept is introduced and discussed in the rest of the thesis.

Chapter 4

The Visualization Concept

In the previous chapters 3, 2 it has been discussed in detail about the approaches used in different visualization tools in various applications and their limitations. The difficulties in accomplishing the goal has been analyzed in section 3.1. It has also been concluded in section 3.3 how the Filter layers design concept did not meet the requirements stated in section 1.1. This chapter will introduce and explain the new visual query concept which has been implemented in this project along with a demonstration.

4.1 The Visual Query Technique

In order to manage the information overload and decrease the anxiety in the information exploration [Wur89] several information visualization methods were proposed. Shneiderman et all [Shn96] have presented a description of the important tasks constituting the visual-information seeking process. Though it is difficult to generalize those tasks that can be performed in all available different information visualization methods, it has been given in an abstract way in the form of a Visual Information Seeking Mantra: overview first, zoom and filter, then details on demand. This way it can be perceived that a common goal of all the graphical interface design of the visualization tools is to ease the information exploration for wide range of users by them an providing an overview of the increasingly growing large volume of data and to locate and browse items in a short time.

Normally when using standard querying languages against endpoints to access information from a database the exploration process starts with

empty result set. That is the users have to write queries from scratch to get some data results. Then they form simple or advanced queries and send them to the database and get back results in the form of traditional lists. Then after analyzing the output, the users may reformulate the query to refine their search process. They might want to filter out certain data or may alternate the search criteria by applying different attributes. The process continues till the valuable information is obtained and it is time consuming.

With the SPAQRL query language facilities it is difficult to gain an overview of a vast collection of data items in the database. An important and useful criteria to be considered in the graphical user interface design for the information visualization systems is the support for overview facilities. The importance of gaining an overview of the complete collection of data for the users has been stated earlier in this paper 1.2. To know the attributes of the data to query on, a prior knowledge of the content of the dataset is required. Hence, it is an important criteria for visualization techniques to provide facilities to present the overview when the ordinary querying languages fail to do so.

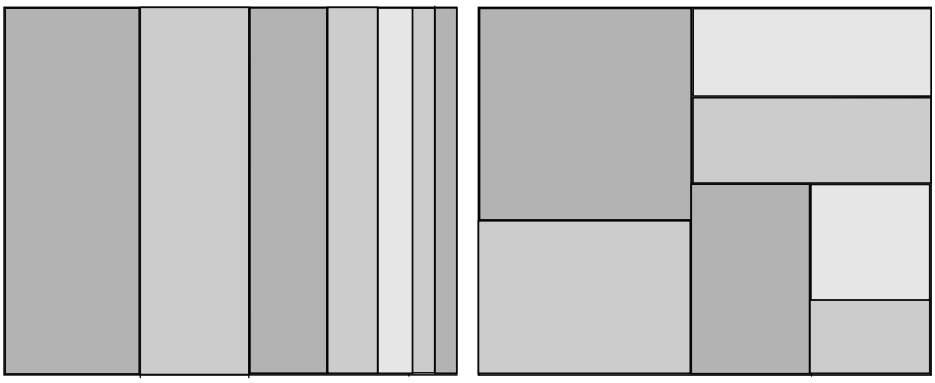
The proposed visual query aims to solve the difficulties in visualization of the RDF datasets where the intended users of this visualization tool are experts as well as a amateurs who are interested in retrieval of information from the Semantic Web. There are several domains where the information is in RDF format. For example Dbpedia[1], linkedmovie database [HC09], DBLP[2]. A paradigm for effective and flexible access to these datsets is always required. The visualization method presents the overview of the data model in a database to the end users avoiding the complexity of the underlying complex structure of meta-data.

The technique uses the approach of visualization of the available data items grouped into several categories according to possible grouping criteria. The amount of data items in each category is represented visually. It then allows for further filtering of data from the presented visualization and easier result analysis. This approach does not focus on any specific schema or ontology structure visualization that can be applied only to ontology based semantic web information. The strategy is to visualize the classification of the data items in all available categories according to the

[1]http://dbpedia.org/
[2]dblp.l3s.de/

grouping properties. This approach reduces the cognitive load to the users caused by large and complex datasets.

4.2 The Visualization Structure



(a) Slice-and-dice. (b) Squarified.

Figure 4.1: Different Tree-map layouts

Unlike scientific visualization, where the data usually has an intrinsic visual representation, the power of information visualization relies mainly on the technique used to transform the data types into forms that can be visualized [TGM83]. This 2-D representation of Tree-map information visualization technique is capable of handling high volume of data items by managing the screen space effectively. It also takes care that the data items are visible to the users. This space-filling technique uses special layout algorithms to fill up pixels of the screen space. When the parameters of the visualization changes, then the algorithm redraws those pixels. The time taken to compute and draw the layout is a limitation with its implementation of Tree-map.

Tree-map method introduced by Shneiderman and colleagues [JS91] helps in the efficient space utilization for presentation of large information structure. The Tree-map visualization algorithm make the visualization of large information space easier by emphasizing the visual representation of the overall patterns in the data. Initially displaying the entire information structure at once allows the user to rapidly navigate to the desired data set. This way the complete overall view is available to the users and it serves a vital starting point in the process of information exploration.

The initial "Slice-and-dice" layout (Figure: 4.1a) of the Tree-map method creates rectangles with extremely large aspect ratios making them hard to see and compare in size. It is also hindrance for applying zooming feature to these structures. The squarification [BHVW00] scheme solves this problem. It is an extension to the standard Tree-map method with improved subdivision technique and is best suited for visualization of larger hierarchical structures. Here the rectangular area is subdivided into rectangles that are approximated to squares so that it is easy to spot and compare the data in the structural form with respect to size (Figure: 4.1b).

The perception of the visual attributes by the users is also an important criterion for effective data visualization design [CM84]. Experiments were conducted on rectangular area perception [HB10] and the findings suggest that as the squarified Tree-map algorithm does not perfectly optimize the rectangles to 1:1 aspect ratios, the visual perception is not hampered. This structure provides a comprehensive overview of a complex set of data at a glance [Wat99]

4.3 Features and Functionalities

4.3.1 Overview

Overview is the primary task for the users when dealing with a dataset. The visualization prototype allows the users to initially select a database to obtain an overview of RDF datasets and understand what constitutes the datasets, the main types and properties of the data, etc. In this visualization prototype, the user starts his search by entering a simple search criterion to make a first selection or filtering of the objects. The results of this simple search could be used as a good starting point for constructing the visualization space. It would be an overwhelming task for the users to identify at first without any visual guidance what part of the whole dataset they would prefer to explore more. Therefore List of available major classes or meta-types of the dataset is presented for the user to select along with the related attributes of the class (Figure 4.2. And the resulting visualization is as shown in Figure 4.3

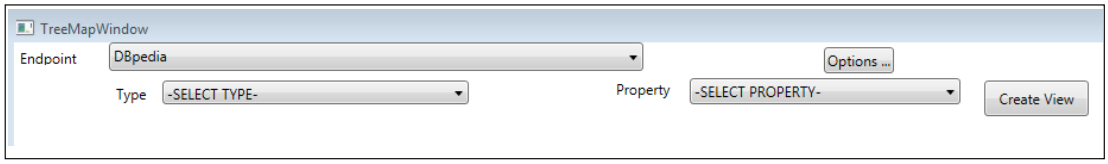


Figure 4.2: Initial selection options for choosing the dataset and type and grouping property for construction of the visualization structure in the work space.

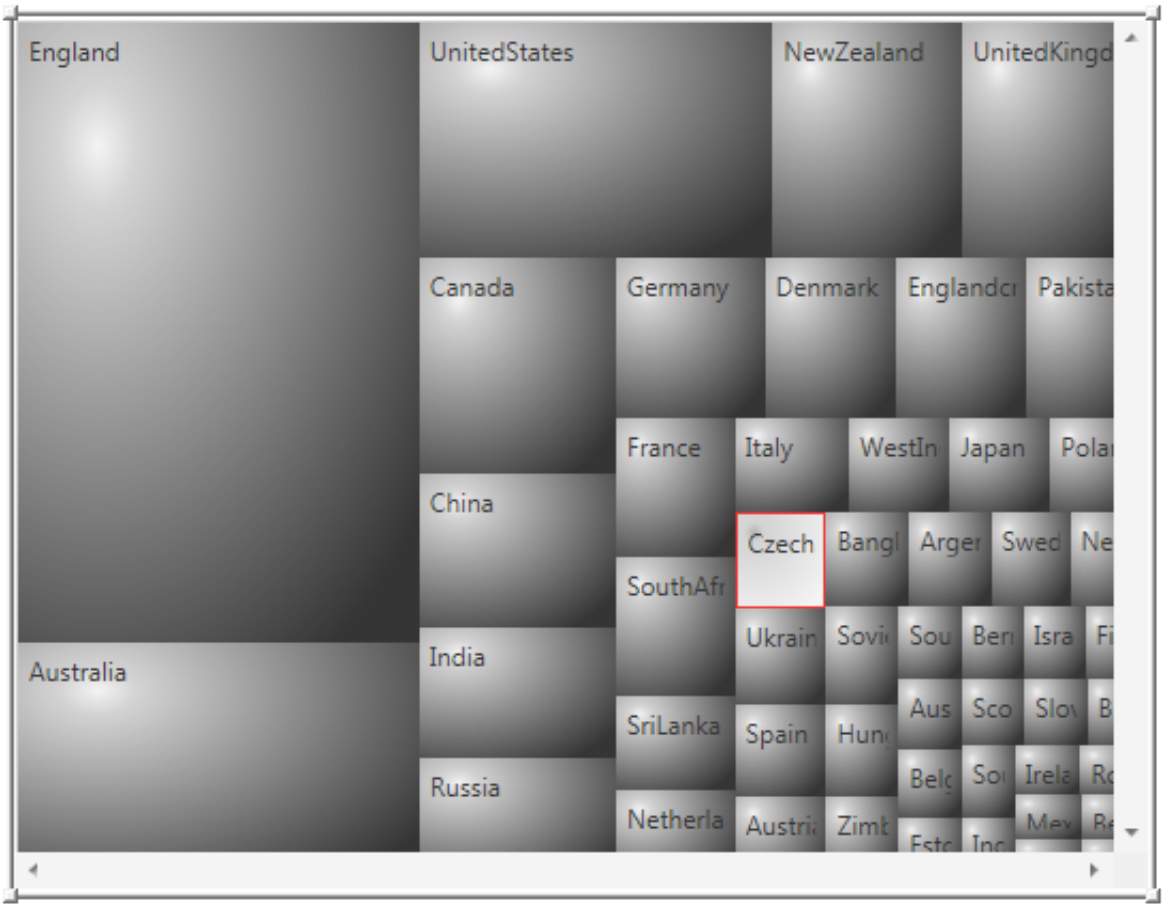


Figure 4.3: Overview of a dataset in Dbpedia, with data items categorized in major groups.

4.3.2 Highlight

The coordination of the thinking process, visual perception and motion of the hands is smooth while interacting with the visualization displays. Highlighting technique is used for rapid selection of certain data items from the visual display of large number of items with immediate feedback. Highlighting draws the attention of the viewer by visually pointing specific items in the visualization. It is also possible to highlight which section of the dataset has a specific data item by entering search term in text box and the items in the Tree-map that matches the filter are highlighted in green yellow (Figure 4.4).

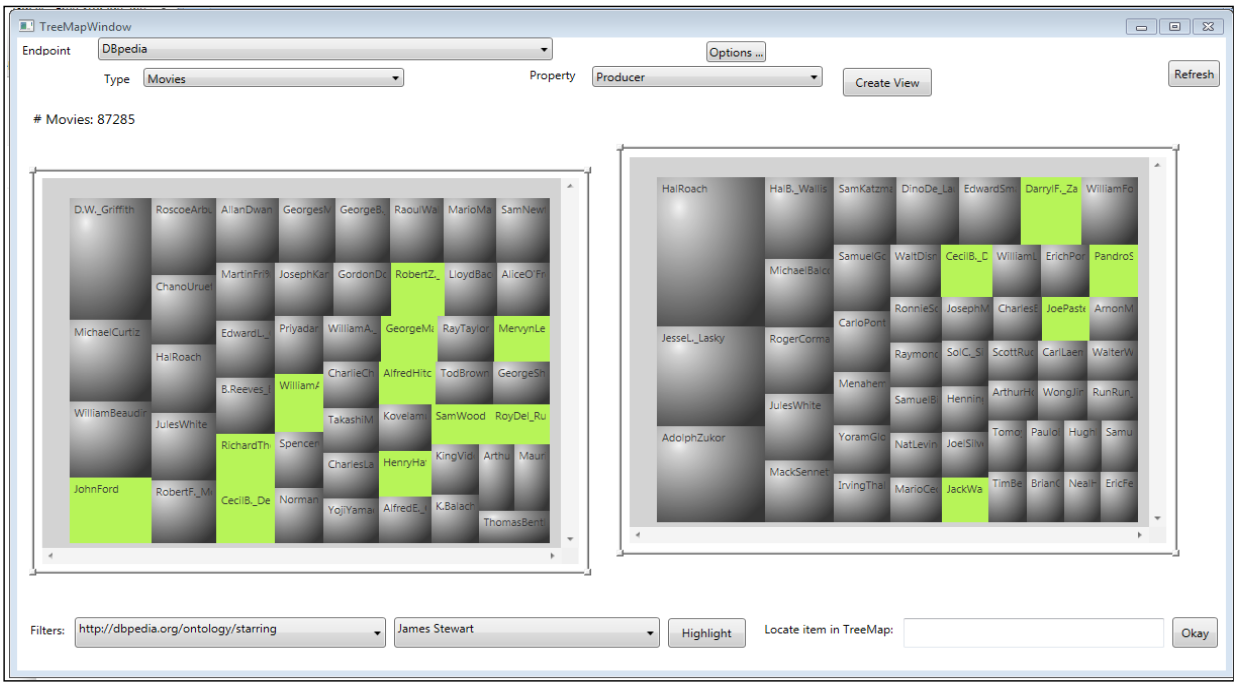


Figure 4.4: Items of the both the Tree-map visualization structures that are matching to the chosen filtering criteria are highlighted.

4.3.3 Details-on-Demand

In the so far discussed visualization, a preview of the result set is presented showing what data is present, how it can be filtered and how much data is available. As the visualization technique focus primarily on the overview task, it is not expected for the tool to show specific information

about the data items. Hence navigation, browsing and further exploration is not a feature of this visualization tool. However it provides few facilitites Tooltips provide basic information about the data subsets: identifier and number of objects available in that group are shown. Tooltips allow the user to get more details on demand when the the user hovers over a data group.

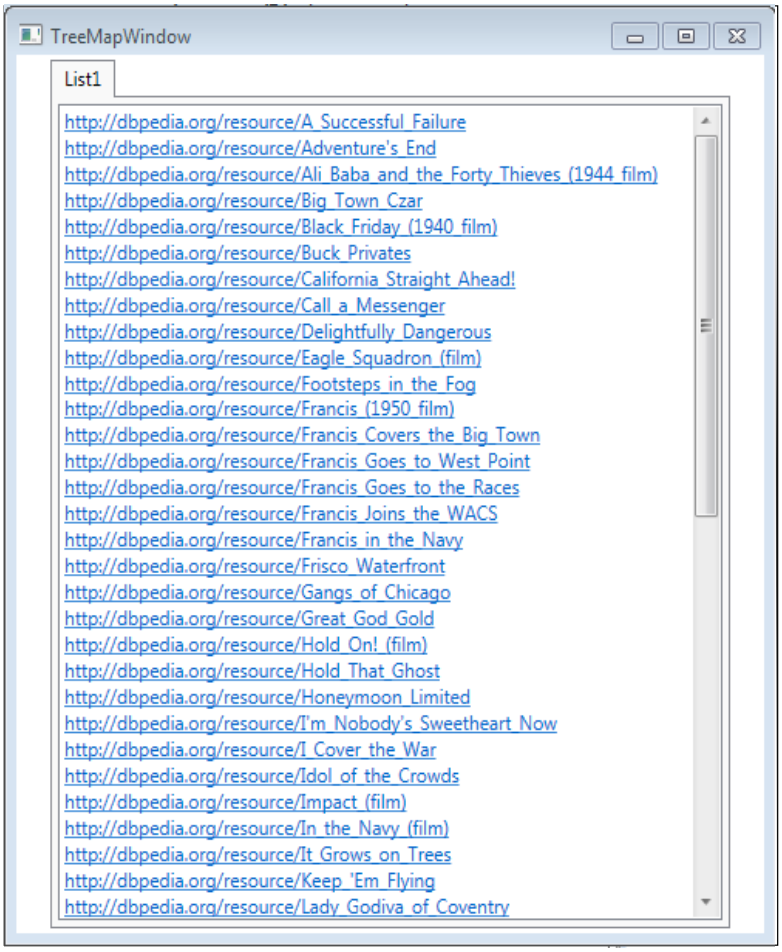


Figure 4.5: List View scrollable window displaying the hyperlink URIs of the data items belonging to a particular group.

While exploring the data set, users would be interested to see the details of a particular set of data items of interest. This is accomplished by clicking on one or more items in the Tree-map as a result of which having the details are displayed in an ”List View” auxiliary window (Figure 4.5.

Not to interrupt with the main visualization space, the details-on-demand or the list view appears as a separate text window. Right–clicking any group and choosing /emphshow list option provokes this action. Querying details of a group here returns the list of all the data instances/objects of that group. This large list of linear data is organized in a sequential manner. The ”List view” window displays the list of URI or literals of the items of a selected group. The data items are organized in a sequential manner by alphabetical ordering. The Scroll bar frame used in the window allows to accommodate thousands of items in the fixed sized space. The items are hyperlinked and clicking on an item re-directs to the web-page resource of that item.

Chapter 5

Implementation

The previous chapter described the design concepts and the algorithm used for the visualization. The main functionality of the user interface tool has also been explained with the help of a demonstration. This chapter introduces several technical details regarding the practical realization of the prototype. The most considerable technical scenarios implemented in the thesis, including the SPARQL querying method, the data retrieval model and the tools and libraries used and more are discussed in the following sections.

5.1 Implementation Process Overview

The RDF data visualization design space here could be related to the general Data State Reference Model (DSRM) [CR98] which is a conceptual framework for several information visualization techniques. The Linked Data visualization Model (LDVM)[BAG+13] that adopts the Data State Model for RDF data has been implemented in LOD visualization which serves as a proof-of-concept. This framework describes the different stages involved in the visualization process for transforming raw RDF data into a visual form.

The raw RDF data has to pass through the following four distinct stages of the data visualization pipeline. An overview of the Data State Model is shown in Figure 5.1. The pipeline is broken into four distinct Data Stages: Value, Analytical Abstraction, Visualization Abstraction, and View and the Data Transformation operators involved are Data Transformation, Visualization Transformation, and Visual Mapping Transformation.

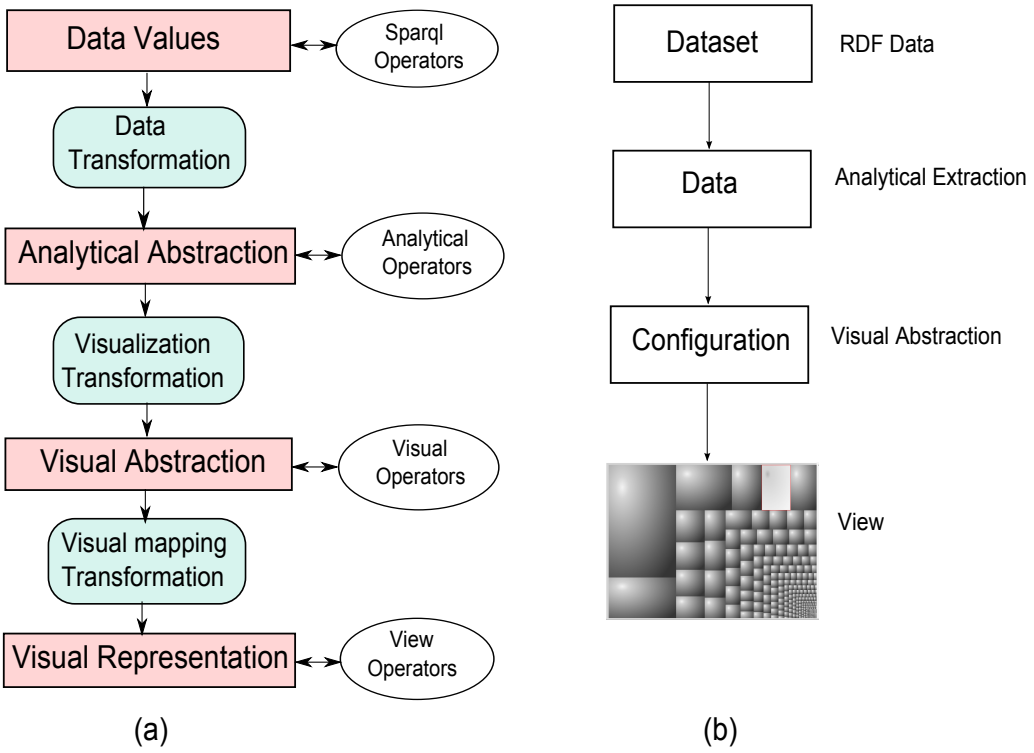


Figure 5.1: (a).High level overview of the data visualization model. (b).Data model applied to the RDF Data set

Data Visualization stages:

1. RDF Data: The raw data (value) from the datasets.
2. Analytical extraction: Information obtained from raw meta-data stating details about the name of the class, number of objects etc.
3. Visual abstraction: Information that is visualizable on the screen using a visualization technique.
4. View: The visual output presented to the user.

Data Transformation Operators:

1. Data transformation: Transforms raw data values into analytical extractions using SPARQL query templates.

2. Visualization transformation: Takes analytical extractions and transforms them into a visualization abstraction. The goal of this transformation is to condense the data into a displayable size and create a suitable data structure for particular visualizations.

3. Visual mapping transformation: Processes the visualization abstractions in order to obtain a visual representation. It can also be called as picture translation where textual information is converted to two or three dimentional information .

5.2 Tools

The prototype has been implemented as a stand-alone graphical query designer application for multivariate data setin C# using the .NET framework. The application has been created in SharpDevelop 4.4 an open source Intergrated Developemt Environmnt (IDE) for .NET. SharpDevelop is written in C# and runs on Windows. The application uses the Microsoft Windows Presentation Foundation (WPF) toolkit for the graphical user interface.

Windows Presentation Foundation (WPF) is a Microsoft's modern UI technology for building Windows client applications. WPF is a user interface framework that facilitates UI design to create rich, intuitive and interactive applications in a shorter amount of time. Using Extensible Application Markup Language (XAML), a declarative language for developing interactive user interfaces, WPF provides developers with the tools that allow them to decouple the visual layout design and the underlying program logic. XAML is used to implement the appearance of an application and code-behind approach (programming in C#) to implement its behavior or the functionality. Most of the WPF base classes are found in the *System.Windows* namespace

5.3 Class Diagram

In order to illustrate the important modules that were implemented a class diagram is presented in Figure 5.2 defining the overall static view of the classes with their components and the dependencies among them.

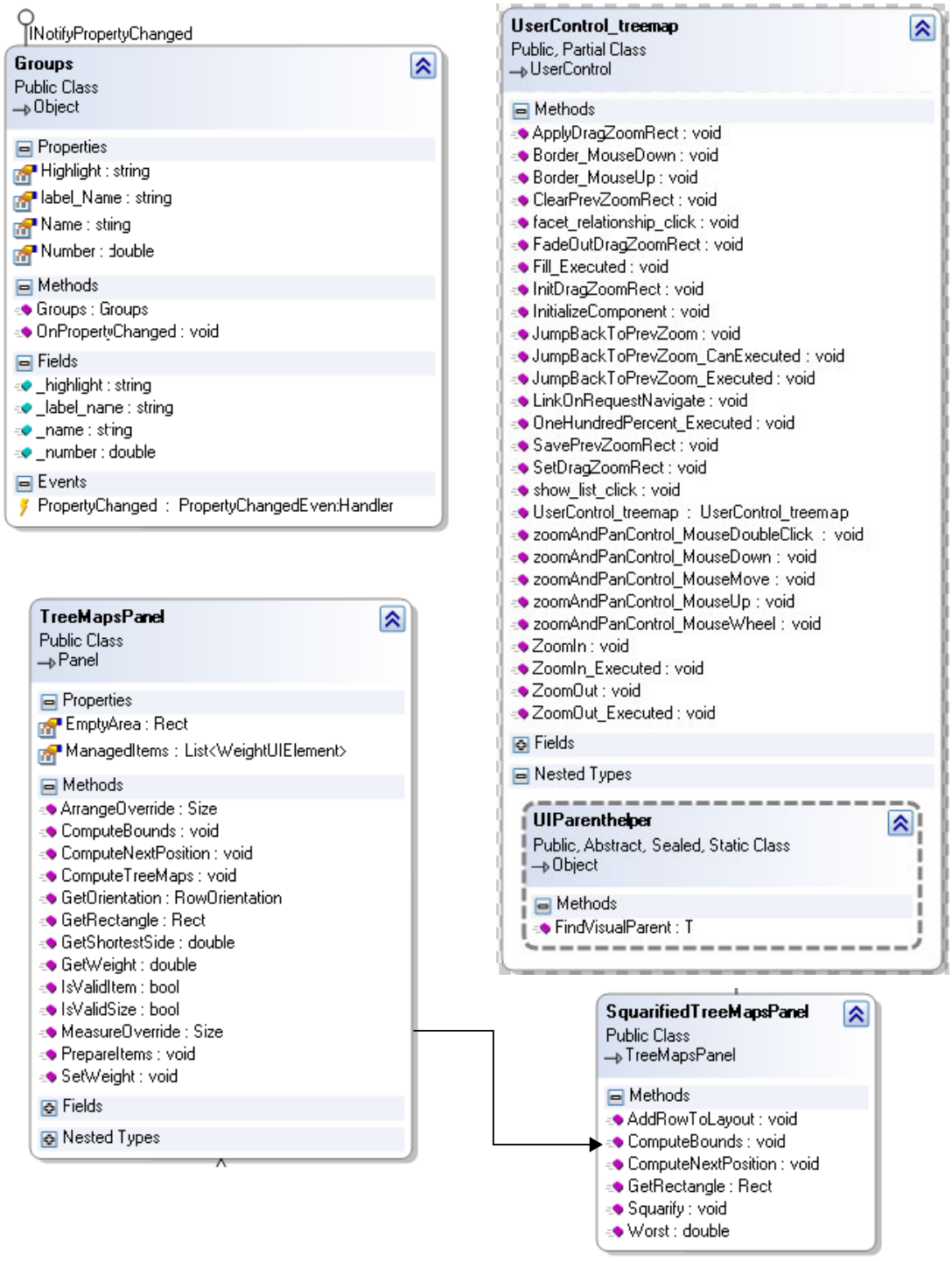


Figure 5.2: Class Diagram

5.4 Tree-map Implementation

In WPF the visualization structure is realized as bindable TreeMaps control. The control was introduced at CodePlex, Microsoft's free open source project hosting site under the Microsoft Public License (MS-PL)[1].

Data-binding is a useful concept in WPF that helps to separate data from the layout. We use templates to define items that cannot be embedded in the XAML description. Templates tell how the items that are instances of a class defined in the code-behind should be visually presented to the user. In WPF, a *DataTemplate* defines how the data looks visually in the rendered application and a *ControlTemplate* defines the structure and appearance of the control. The *ItemsControl* is meant to support data-binding. A data template is specified inside the *ItemsContro* for this purpose. This list of items generated is assigned to the *ItemsSource* property of the *ItemsControl*. The *ItemsControl* allows us to select various useful WPF panel type such as *WrapPanel*, *UniformGrid* panel etc by declaring one in the *ItemsPanelTemplate* property.

The TreeMaps controls inherits from *ItemsControls*[2]. *ItemsControl* is a type of control in WPF that is able to represent a collection of multiple types of items such as string or any UI element. To generate the content of the *ItemsControl*, *ItemSource* property is used. *ItemSource* property renders different types of items that implements *IEnumerable*[3] and binds the collection to the *ItemsControl*. With the help of this property we can add and remove items and it can also be set to null to empty the items. The control is bindable and styling and templating can be applied.

The TreeMaps control supports grouping over homogeneous collection of data items and also supports in creating hierarchy levels in depending on the relationship between the data items. The algorithm that serves as a base to this implementation of Tree-maps is derived from the Squarified Tree-maps concepts that is clearly presented in [JS91]. The decision of choosing the Squarified Tree-maps among various other structure has already been described in the previous chapter. The main methods used in the algorithm has been described in short in the previous chapter.

[1] http://opensource.org/licenses/ms-pl.html
[2] System.Windows.Controls.ItemsControl
[3] System.Collections.IEnumerable

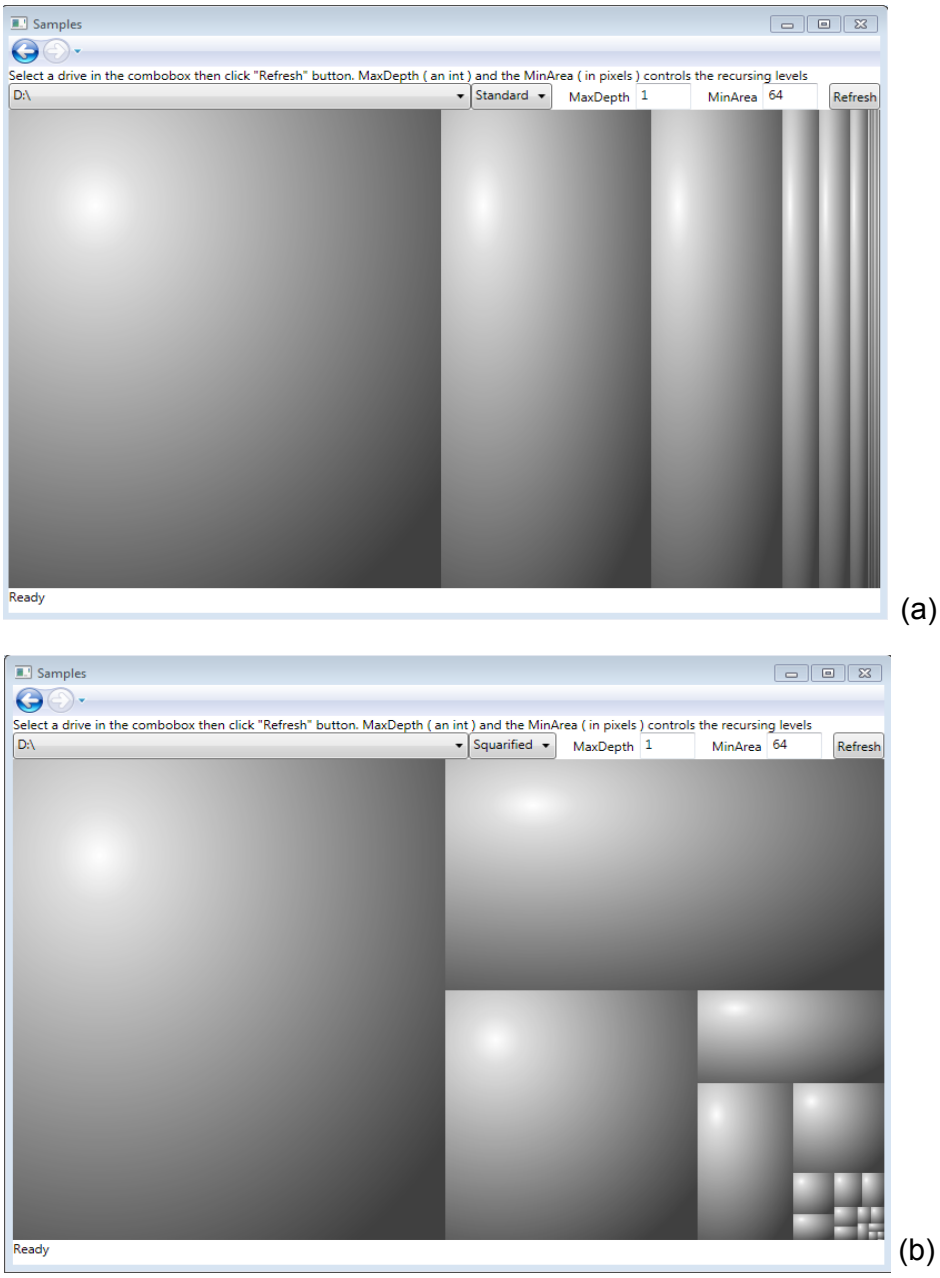


Figure 5.3: Disk analyzer sample: (a) Using TreeMapsPanel, (b) Using SquarifiesTreeMapsPanel

Here the SquarifiedTreeMapsPanel layout logic is implemented for the TreeMaps control. SquarifiedTreeMapsPanel arrange the elements in a way that the resulting layout constitutes of almost square like rectangles.

Figure 5.3 depicts a snapshot of the Sample demo implementation of the WPF TreeMaps control. The sub figures (a) and (b) show the Disk analyzer sample that uses a HierarchicalDataTemplate to manage recursing child with TreeMapsPanel and SquarifiedTreeMapsPanel respectively. The figure clearly depicts the difference in the resulting layout of same dataset with the two different panels.

In general the goal of this Tree-map visual structure is to display the items of the Tree-map with size relative to the weight of those items. In this project all the object instances of a class in the dataset are categorized into groups. The groups are formed based on any of the possible properties that all instances share. Hence a group is considered to be a data item in the Tree-map. Each group is related with the number of data items i.e. the number of the object instances that it constitutes as its weight.

5.5 SPARQL Query Design

The Semantic Web architecture is based on RDF technology. Information about data resources is represented as Resource Description Framework (RDF) and RDF Schema (RDFS)[4] provides the taxonomical organization. Web Ontology Language (OWL) extends RDFS and describes the semantics of the elements. SPARQL [PS+08] is a standard RDF query language that is used to retrieve information from the Semantic web data sources.

To extract information from RDF datasets using SPARQL languages, URIs of the class name, relationships should be needed to start with. There are tools and techniques that makes this work easier by providing a visual representation of the RDF schemas. For example: RelFinder [LHSZ10], tfacet [BH11], SparqlFilterFlow [HLE12], ViziQuer [ZB11] are examples of visual interfaces for the composition of SPARQL queries there were quite intuitive and help to learn sparql querying quickly without requiring in depth knowledge of Semantic web concepts and the RDF syntax.

Several simple SPARQL queries are formulated and applied to the RDF knowledgebase to acquire and store results. The query results are checked if they are satisfactory and then the visual representation is built based on the data obtained. The SPARLQL queries are executed against one SPARQL end point at a time. For loading, displaying and storing RDF

[4]http://www.w3.org/TR/rdf-schema/

data dotNetRDF[5] API [VZA+13] is used. dotNetRDF is a powerful open source API under the permissive MIT license[6] and it is a .Net library that is written in C#. It supports any data store that supports SPARQL Graph Store HTTP Protocol[7]. The *Store Manager* utility provides a simple GUI interface where we can make SPARQL queries against the chosen store and the results can be displayed in the window. The GUI was useful in this project for forming querying against the DBLP endpoint and viewing the results.

Remote Querying concept in dotNetRDF helps to make queries against a remote SPARQL endpoint. Queries to an endpoint are sent using a the wrapper class *SparqlRemoteEndpoint*. This class provides the following special methods for creating queries. *QueryWithResultSet(String sparqlQuery)* method has been used to make the SELECT query. *QueryWithResultGraph(String sparqlQuery)* method has been used to make the CONSTRUCT query. The *SparqlResultSet* class is used to represent the results of the SELECT query in the form of an enumerable list and the *IGraph* class is used to represent the results of the query in the triples in graph structure. *GetUriNode* method is used to create a URI node by specifying an URI string, *GetTriplesWithSubject* is used to select a node from the triples that were created. Listings 5.1 and 5.2 show the corresponding C# code written for this project to implement all of these methods.

```
//Make a SELECT query against the Endpoint
string query = "SELECT DISTINCT ?obj (COUNT(?a) AS
   ?Count)
WHERE{ ?a a <"+typeURI+">. ?a <"+groupURI+"> ?obj.}
GROUP BY ?obj
ORDER BY DESC (?Count)
LIMIT 60";

try
{
 SparqlResultSet results =
    endpoint.QueryWithResultSet(query);
```

[5]http://dotnetrdf.org/
[6]http://opensource.org/licenses/MIT
[7]http://www.w3.org/TR/sparql11-http-rdf-update/

```
 foreach(SparqlResult result in results)
 {
  foreach(String var in result.Variables)
  {
   if (result[var] != null)
   {
    INode n = result[var];
    String output;
    switch (n.NodeType)
    {
     case NodeType.Literal:
     output = ((ILiteralNode)n).Value;
     list.Add(output.ToString());
     break;

     case NodeType.Uri:
     output = ((IUriNode)n).ToString();
     list.Add(output.ToString());
     break;

     default:
     output = n.ToString();
     list.Add(output.ToString());
     break;
    }
   }
  }
 }
}
catch (RdfQueryException ex)
{
 MessageBox.Show(ex.Message + "\n\n" +
    ex.StackTrace);
}
```

Listing 5.1: C# code snippet showing the implementation of the *QueryWithResultSet* and other related methods to make SELECT query and add result items to Tree-map.

```
//Define remote endpoint
SparqlRemoteEndpoint endpoint=new
   SparqlRemoteEndpoint(new Uri(URI_string));
//Create CONSTRCUT query
string query="CONSTRUCT{<"+rdfslabel+">
   <"+typeURI+"> ?c.}WHERE{<"+rdfslabel+">
   <"+groupstring+"> ?c.}";
//Get the result as graph
IGraph g;
try
{ g=endpoint.QueryWithResultGraph(query); }
catch (Exception ex)
{
System.Diagnostics.Debug.WriteLine(ex.ToString());
throw ex;
}
//Specify a node with URI
IUriNode select=g.GetUriNode(new Uri(rdfslabel));
HashSet<string> highlight_group=new
   HashSet<string>();
//Find the triple with that node
foreach(VDS.RDF.Triple t in
   g.GetTriplesWithSubject(select))
{
 System.Diagnostics.Debug.WriteLine(t.ToString());
 highlight_group.Add(t.Object.ToString());
}
//Highlight the appropriate item in Tree-map
foreach(Groups gr in found_tree._tree.ItemsSource)
{
 gr.Highlight="Transparent";
 if(highlight_group.Contains(gr.Name))
 gr.Highlight = "GreenYellow";
}
```

Listing 5.2: C# code snippet showing the implementation of the *QueryWithResultGraph* and other related methods to highlight an item in Tree-map.

```
1.SELECT DISTINCT ?obj (COUNT(?a) AS ?articleCount)
2.WHERE {
3.?a a <http://swrc.ontoware.org/ontology#Article>.
4.?a <http://swrc.ontoware.org/ontology#journal>
   ?obj.}
5.GROUP BY ?obj
6.ORDER BY DESC (?articleCount)
```

Listing 5.3: An example SPARQL query on DBLP dataset.

Some examples of SPARQL used that are used in this implementation are described in short as follows. The SPARQL query in Listing 5.3 uses the SPARQL clause GROUP BY to categorize the articles that are found in the DBLP datset according to the journal in which they appear. The purpose of the GROUP BY clause allows to form a collection of data items over one or more properties. The properties of interest are used on the variables in the SELECT clause. The results are divided into several groups with their aggregate values. The COUNT clause is used to list how many data items belong to each group. The LIMIT clause puts an upper bound on the number of results returned. The ORDER BY DESC clause orders the results in the list as descending. The SELECT query form returns all, or a subset of, the variables matching the query pattern. The CONSTRUCT query form returns an RDF graph whose triples are formed according to the graph template provided in the query.

```
1.CONSTRUCT {
2.<http://dbpedia.org/resource/MovieMaker>
3.<http://dbpedia.org/class/yago/Magazine106595351>
   ?c.}
4.WHERE{
5.<http://dbpedia.org/resource/MovieMaker>
6.<http://dbpedia.org/property/category> ?c.}
```

Listing 5.4: A CONSTRUCT query example 1 on Dbpedia.

```
1.CONSTRUCT {
2.<http://dbpedia.org/resource/Victor_Mature>
3.<http://dbpedia.org/ontology/Film>  ?c.}
4.WHERE {
5.?s a <http://dbpedia.org/ontology/Film>.
6.?s <http://dbpedia.org/ontology/starring>
 <http://dbpedia.org/resource/Victor_Mature>.
7.?s <http://dbpedia.org/ontology/director> ?c.}
```

Listing 5.5: A CONSTRUCT query example 2 on Dbpedia.

The SPARQL query examples depicted here are made on the Dbpedia endpoint. The CONSTRUCT query example in Listing 5.4 is formed to find out the category to which the magazine named *MovieMaker* belongs to. The query in Listing5.5 is used for finding the director(s) of the films where the actor names Victor Mature has been a star. The result obtained from the SELECT query in Listing 5.6 is the list of companies in alphabetical order that produces Electric generators.

```
1.SELECT DISTINCT ?a  WHERE{
2.?a a
   <http://dbpedia.org/class/yago/Company108058098>.
3.?a <http://dbpedia.org/property/products>
   <http://dbpedia.org/resource/Electric_generator>.}
4.ORDER BY(?a)
```

Listing 5.6: A SPARQL query example on Dbpedia dataset.

Chapter 6

Experimental User Evaluation

Visualization tools cannot be successfully developed without comments and suggestions from the users. The prototype developed here is experimented with two RDF knowledge bases (Dbpedia, DBLP) and small user studies have been conducted where the users were given a set of tasks to be performed with the prototype. The basic functionality of the tool is tested along with other subjective factors. Experienced and novice users have different level of expertise in performing visualization tasks hence some amount of initial training is also given for the users to conduct experiment with the visualization prototype tool. This chapter explains in detail on what the user evaluation study consist of, how it was conducted, what tasks were performed by the users and results of the study.

6.1 Hypothesis

One of the important feature of any Semantic Web information visualization tool is its capability to allow the users to extract more relevant and useful information from the wide range of data available from various RDF databases. There have been difficulties for the users in getting to know about an unknown dataset before they can query something and extract more interesting information. In an attempt to overcome this problem, a visual query technique has been proposed and a prototype has been practically implemented in this thesis focusing on the visualization of overall availability of data.

The prototype is primarily designed to provide a graphical visual representation of the large categories of data available as RDF in the web so as to help the users to comprehend the contents of a dataset. It is useful

especially for those users who have little or no knowledge about the underlying structure of the dataset and the relationships among those data items. The visualization prototype tool shows the available groups of data items and their size according to different filtering options. The users after gaining a meaningful insight of the aggregates of data available can then proceed their information exploration process with a wider perspective and good grasping of the overall content of the dataset. The implementation of the visualization structure is achieved with the help of the space filling Tree-map structure and is based on the concept of Squarified Tree-maps.

As seen earlier in previous chapters, in Filter Dials the shape and arrangement of dials creates difficulty in displaying details about the chosen properties or types or any such information in its circular ring structure. Hence it is not easy for the users to get a preview of the result through the dials. It also lacks support to provide the users with any idea of which attribute could be chosen for querying so that the result set is not null or overwhelmingly more. It is conjectured that using this visualization prototype proposed here users would be able to comprehend the available amount data categorized in possible subsets easily.

6.2 Subjects

To evaluate the prototype interface, the usability study as been promoted by invitations via mailing lists of the department and the university. This initial evaluation of the prototype did not attempt to focus on a specific target user community or only those users which have knowledge and/or experience in SPARQL but also on casual end-users. The study was conducted in a small way with 6 participants composed of 5 males and 1 female of age group 24-29 years. All subjects were students from Software background with sufficient English language skills. Three subjects claimed to have knowledge in Semantic web technologies like RDF, SPARQL etc and the three other subjects do not have any knowledge in this subject area. The users were made aware that the query interface were being tested and not the users which will influence the test results. The subject's anonymity and confidential data storing has also been ensured. The subjects participated in the evaluation were given a small reward at the end of the overall experiment.

6.3 Training and Method

The specific areas of focus for the evaluation of the prototype include the general usability of the visualization tool, the ability of the tool to support the users in gaining an overview of the large dataset and comparative analysis of the tool with other visualization techniques in trying to get a grasp of the overall content information of a dataset. There are several such useful tools like gfacet [HZL08], tfacet [BH11] and Tabulator [BLCC+06], Relfinder [LHSZ10] etc. In order to keep the user study evaluation for the prototype simple and short with limited number of users, only one tool was chosen for the comparative evaluation. gFacet has been chosen here and it is available online as a demo tool[1].

Users were given similar kind of tasks to be performed in both the tool. Half of the users started using the prototype first and then the gFacet, the other half did vice-versa. Through this comparative test analysis it can be found how much the users are able to see the advantage in using this visualization prototype preferring it to the other techniques and what disadvantages they see in the current prototype. In the context of user evaluation studies there are a few dependent variables that could possibly be considered. These include:

- Accuracy: the extent to which the prototype visually presents the right information.
- Comprehensibility: the level of comprehension attained by the users.
- User Satisfaction: subjective ratings of the user's satisfaction with the visualization tool.
- Efficiency: the amount of time taken to perform the tasks.

Before the test, the users were given information on the purpose of the test, the goal of the project and basic introduction to the visualization concept used here. Instructions on how to use the prototype interface and the gFacet tool were given. All the information concerning the experiment such as the introductory notes, instructions were given on paper and also explained orally to all the users (The complete study material can be found at the appendix A of this document). At the beginning the experiment, users were requested to answer a few demographic questions such as age, gender, profession, knowledge of semantic web, query languages,

[1] http://www.visualdataweb.org/gfacet.php

information visualization and knowledge of English were to answered by participants. Then the task sheet was handed over.

Users were made sure that they could interrupt or end the experiment anytime. The overall experiment took about 40–50 min for each subject:

1. 5 mins: Introduction to the concept.
2. 10 mins: Training on the tool.
3. 20 mins: Actual evaluation tasks.
4. 5 mins: To fill the interview sheet.

After testing the interfaces, users were explicitly asked to fill in a questionnaire in which they were questions on advantages, disadvantages of the tool , subjective satisfaction of using the interface, if the prototype is suitable for any real world applications to the user. They were also asked about the motivation of choice between the two interfaces and open suggestions and comments were welcomed.

6.4 Tasks

The experiment consists of the following three tasks to be carried out by the users. Each task question is framed to assess a particular feature and functionality of the prototype. The tasks are also to be performed with the other available tools taken for comparison one by one. Time taken, correctness of result and ease of use is measured and compared between the different methods.

Task 1: This task is given to the users to assess how easily the users are able to get to learn the tool and to allow the users get acquainted with the interface. This is done by giving them a simple task through which they are asked to make initial attribute and type selection. As a result of this, the users will be lead to successfully construct the visualization structure which will serve as the starting point to perform for further tasks and to analyze the visualization structure to get the expected information. The task is:

- To categorize *Movies* that are found in the Dbpedia dataset into several available smaller subsets according to two attributes: one visual instance consists of the objects of type *Director* grouped based on the

country they belong to and the other visual instance consists of subgroups categorized according to the *Producer*.

- Identify the total number of movies available in the dataset.

Tasks 2 and 3: The second task motive is to check whether the basic functionality is achieved. The users would construct the Tree-map visualization structure as done in the previous task. By performing these tasks users should be able to gain an overview of the what and how much data is available and what are the possible attribute filter that can be applied to the chosen dataset of interest. The task consists of the questions as follows:

- The users are asked to identify those countries in which the number of *Athletes* are dominant and what are their numbers.
- Find out which country has a greater number of athletes *Canada* or *Ukraine*.

Tasks 4 and 5: The motive of these tasks is to apply a filter and identify the interesting section of the data. The successfulness of these tasks will be an assessment criteria to show that the visualization technique proved to be a good starting point to the more detailed and deep information exploration process of the information seekers in the web. These tasks were performed to exemplify two different features of the prototype: The highlight feature and the list view. The questions are:

- Identify those directors and producers who worked with the actor *Victor Mature*.
- Find out to which country does the athlete *Arina Martinova* belong to.
- List a few films directed by *John Ford*.

The highlight feature helps the user to identify the result of the specific filter(here the name of the athlete) and the list view window is considered to be the detailed view of each subsets. It provides a list of all the objects belonging to that group as URL links through which the users could navigate the actual page of the resource in the web.

Tasks with DBLP internal server for scientific publications:

1. Note down the names of some of the major *Conferences* in which the scientific publications have been published so far.
2. Find out the conferences in which the professor *Thomas Ertl* has presented his scientific papers.
3. Does the number of *Preceedings* published in the year 2004 and 2014 differ too much?

Tasks to be performed with the gFacet demo tool: As discussed in gFacet is a visualization tool based on graph and facet approach. In the process of finding information from the toll users perform the following operations:

- Initially the tool provides a key word search. Entering a term or a phrase, the tool presents a list of classes with number of instances matching the text.
- The user could select the interesting class and as a result facets with the list of objects belonging to that class is constructed in the visualization space.
- The users can now continue this process of faceted exploration in the facet relation graph by seeing relationships among the objects in various facets.

The following tasks were given to the users to work with the gFacet tool:

1. Find out how many people are computer scientists from Germany?
2. How many scientists among them have studied in *University of Bonn*?
3. In which University in Germany there are highest number of German computer scientists produced that are known to Dbpedia?
4. To which university does *Susanne Albers* relate to?
5. List a few more information about *Susanne Albers* that you could collect from this tool.

6.5 Results

Participants were able to quickly understand all the tasks. No one reported any problem in understanding the basic idea of the tool and they said that the concepts were clear. All the participants learned to use the tool in a short time and they found it quite easy to use. They tried to solve all the tasks that were to be done with the prototype tool. The users pointed out that with help of tree view it is easier to spot and find the desired items from among the Tree-map items. It was noted that the users initially did not know how to query the database for getting the results but in the first iteration itself they were confident in the data domain they would work with and a detailed investigation was possible once they constructed the visualization. Also, they realized that depicting massive data in smaller blocks/sets is preferable for easier investigation with reduced amount of data.

While working with gFacet tool, many of the participants were struck with the task 3 where they have to find the university in Germany with the highest number of German computer scientists listed on DBpedia. As opposed to that, with the prototype, similar kinds of tasks were completed successfully; for example, one of the tasks was to find out which country has a greater number of athletes. They also reported problems in the difficulty to choose meaningful attributes for constructing the facets in the graph because of inconsistent language usage in the tool. Also sometimes the chosen attribute resulted in empty facets.

As an advantage with the implemented prototype tool, it has been quoted that the tool encouraged the users to discover the datasets by providing a glance of the main categories of the large dataset in the Tree-map view which would otherwise be not manageable. Users said that it was good to see the size of the datasets visually and and easier to understand and is more intuitive when performing queries. The results proved that this approach of visualization an overall picture of the main categories helped in better understanding of how the data is structured in the database.

When asked for their ideas of possible real world applications, one user said that it would be useful if the tool is completely developed to big data and social media analyses and yet another user said she would find it useful in statistical analysis. As a further improvement step it was suggested that the possible filtering options can be made powerful by pro-

viding more filter options to choose from.

Chapter 7

Summary and Conclusion

In the beginning of the thesis, the importance of the visualization tools in coping with information overload in the Semantic Web linked RDF data has been emphasized. The Filter Dials, a novel visualization concept served as the base work and a starting point for the thesis research. Some of the notable limitations with the current Filter Dials such as the structure of the Filter Dials, the shape and arrangement of the filters has been discussed in short. Chapter 2 presented some interesting related works done in this topic such as Faceted visualization approach, clustering technique, Tabulator like semantic web browsers, etc. The chapter also included a discussion of their short-comings in fulfilling the goal of this project.

The difficulties concerning the overview-visualization concept has been elaborated in Chapter 3. The major difficulties encountered in trying to solve this technical problem with the similar existing visualization tools were investigated. The idea of Filter layers has been proposed in this chapter as a solution to enhance the Filter Dials and overcome some of its issues. However, it has been identified that the Filter Layers concept has some inherent limitations and does not meet the requirements of the thesis goal. Chapter 4 described the new proposed visual query technique. The functionalities of the visualization tool and the Graphical User Interface (GUI) elements are also explained. Chapter 5 presented the tools and methods used to implement the visualization prototype along with the data visualization process model, Class diagram, examples of SPARQL queries used and C# code snippets implemented in WPF. Finally, user evaluation has been conducted with the implemented prototype. The experiment procedure and the results obtained from the user study have been discussed in Chapter 6 .

Through the survey, it was clear that the process of exploring the increasingly growing Semantic Web data would become arduous for the casual end users without any support from effective information visualization techniques. This problem has been addressed in the proposed visualization tool, where it is possible to obtain an overview of the dataset in an easily understandable way to common users. The Squarified Treemap algorithm has been used for the realization of the prototype. The technique uses the approach of visualization of the available data items grouped into several categories according to possible grouping criteria. The amount of data items in each category is represented visually in the Tree-map structure. It then allows for further filtering of data from the presented visualization and easier result analysis.

The visualization of the overall data content would be a good starting point for users to explore data in an unknown data domain. This functionality is usually not available in text-based semantic web browsers. The prototype developed here does not require users to form complex queries which would be difficult without any knowledge on the structure of datasets. Also it reduces the time for exploration of data since the users can avoid visiting several datasets to know where the interesting data is.

The initial prototype developed here provides limited functionality and there is scope for further enhancements. For example, better filtering options, color coding to distinguish objects, refinement of results are a few key features for improvement of the prototype. When this prototype is fully developed, it would find its application both in industry and academia where finding relevant information from ever growing online databases can be simplified with the help of the visualization tool to save time.

Appendices

Appendix A

Appendix

A.1 User Study Material

User study Material

This is a short introductory note to the users who are participating in the user evaluation studies with the visualization prototype that is developed as a part of the undergoing Master thesis work titled “Filter Dials and more”.

Increasingly growing large amount of data in the internet is available as semantic web of data in RDF format. The visualization prototype is trying to find out generally what data is contained in a given RDF graph. The tasks performed by the users are aimed in understanding what and how much data is available in a dataset and how they can be meaningfully grouped into categories. As a result of making use of this visualization tool, the users will gain some knowledge of the underlying dataset and get an overall impression of the large data items. It is expected that this type of visualization makes their further data exploration process easy.

The gfacet visualization demo tool is given to the users to perform similar tasks and to compare with the new prototype. In order of get a prior knowledge of the major features and functionalities of the tools, the following example tasks can be tried out by the user before the actual evaluation. Please not that the evaluation does not aim to test the user skills but the prototype.

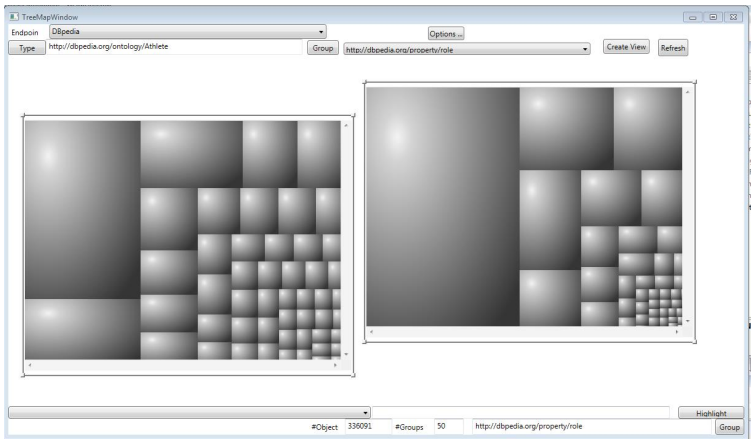
A.2 Tasks for Practice

Tasks for practice

Snapshot of the prototype and some sample tasks for practice:

Few instructions for the user:

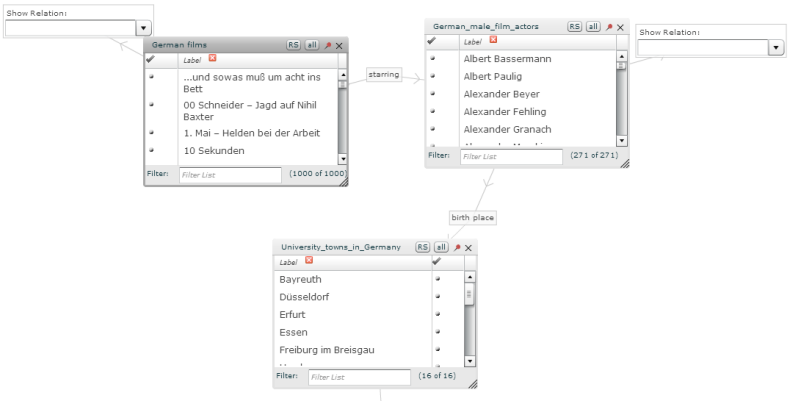
1. In the top region of the main-window, the endpoint, type and property can be chosen from the dropdown list.
2. Create view button constructs the visualization structure.
3. The bottom most region of the window consists the options for choosing a filter to highlight.
4. Search text box at the bottom can be used for locating an item in the treemap.



1. Create an instance of the visualization structure in the initial work space by selecting a type and a property from the drop down list.
2. Drag, resize, move the treemap visualization instances thus formed. Zooming can be done by mouse scrolling.
3. Try mouse over the visual treemap structure and see what details are shown in the tooltip.

4. Select a filter options and hit the highlight button to see the subgroup that matches with the filter criteria.
5. Right click any group to see a list of all the data items belonging to that subset.

Snapshot of the gfacet demo tool and few practice tasks:



1. Enter in the search term box the word “India” and see what classes are available matching to that term.
2. Select a class of your interest for example “Research institutes in India” from the list of displayed classes and try to construct the gfacet graph.
3. In the facet that is created check how many objects are listed. (look for numbers).
4. Choose an attribute from the “show relation” menu list to form another facet. See how those facets are related to each other. Similarly you can create and relate to many other facets.
5. Click on any particular data item in a facet and see how the matching data items in other facets are filtered.

A.3 Evaluation Tasks

Tasks to be performed with the new visualization prototype:

1. Categorize athletes into several available smaller subsets according to two attributes: one visual instance consists of the objects of type athletes grouped based on the country they belong to and the other visual instance consists of subgroups of the athletes according to the role they play.
2. Identify the total number of athletes. Also identify those countries in which there are maximum number of athletes.
3. Find out which country has more number of athletes Canada or Ukraine.
4. Identify to which country does the athlete Chris Cowdrey belong to and what role does he play as an athlete.
5. List a few athletes who are All rounders.

DBLP internal server for scientific publications:

1. Note down the names of some of the major conferences in which the scientific publications have been published so far.
2. Does the number of proceedings published in the year 2004 and 2014 differ too much?
3. Can you find out the conferences in which the professor Thomas Ertl has presented his scientific papers?

Tasks to be performed with the gfacet demo tool:

1. How many people belong to the German computer scientists' category?
2. How many scientists among them have studied in Universities in Germany.
3. In which University in Germany there are highest number of german computer scientists produced that are known to DBpedia?
4. To which university does Susanne Albers relate to?
5. List a few more information about Susanne Albers that you could collect from this tool.

Tasks to be performed with the new visualization prototype:

1. Categorize all the Movies that are found in the Dbpedia dataset according to Directors and Producers. Identify the total number of movies available in the dataset.
2. Identify those directors and producers who worked with the actor Victor Mature.
3. List a few films directed by JohnFord.
4. Identify the country in which there are maximum number of athletes.
5. Could you find out to which country does the athlete Arina Martinova belong to?
6. Which country has more number of athletes Canada or Ukraine?

DBLP internal server for scientific publications:

1. Note down the names of some of the major conferences in which the scientific publications have been published so far.
2. Can you find out the conferences in which the professor Thomas Ertl has presented his scientific papers?
3. Does the number of proceedings published in the year 2004 and 2014 differ too much?

Tasks to be performed with the gfacet demo tool:

1. Could you find out how many people are computer scientists from Germany?
2. How many scientists among them have studied in University of Bonn?
3. In which University in Germany there are highest number of German computer scientists produced that are known to Dbpedia?
4. To which university does Susanne Albers relate to?
5. List a few more information about Susanne Albers that you could collect from this tool.

A.4 Interview Questions Sheet for the Users

Interview with the users

1. How old are you? …….
2. Gender: Male/Female ……
3. Profession: …….
4. Highest Education: ……
5. Field of study: ……
6. English language skill:
   (On scale of 1-4, 1=no knowledge 4=fluent in speaking, reading, writing):
   …….
7. Do you have knowledge about the Semantic web technology and SPARQL ? (Basic, Medium, Expert)
   ……
8. Did you have any difficulty in understanding the basic concept of the visualization prototype? Please explain.
   ……
   ……
   ……
9. Did you have any difficulty in performing the given tasks with the visualization tool? Please explain

……
……
……

10. Ease of use and subjective satisfaction with the interface: Easy/Medium/Difficult……..
11. Do you see any advantages or disadvantages of using this visualization technique compared to the other tools for performing similar tasks:
……
……..……
…….
12. Do you think this prototype will be of use in the real world applications if developed into a complete tool? If yes could you say for what purpose you would use it?
.....
.....
.....

13. Open comments and suggestions about the visualization tool:
……
……..…
…….

Bibliography

[ACS90] Michele Angelaccio, Tiziana Catarci, and Giuseppe Santucci. Qbd*: A graphical query language with recursion. *IEEE Transactions on Software Engineering*, 16(10):1150–1163, 1990.

[AMH10] Oszkár Ambrus, Knud Möller, and Siegfried Handschuh. Konduit VQB: a visual query builder for sparql on the social semantic desktop. In *Proc. VISSW2010*, 2010.

[BAG+13] Josep Maria Brunetti, Sören Auer, Roberto García, Jakub Klímek, and Martin Nečaský. Formal linked data visualization model. In *Proc. IIWAS '13*, pages 309–309, 2013.

[BE06] Jethro Borsje and Hanno Embregts. Graphical query composition and natural language processing in an rdf visualization interface. *Erasmus School of Economics and Business Economics, Vol. Bachelor. Erasmus University, Rotterdam*, 2006.

[BH11] Sören Brunk and Philipp Heim. tfacet: Hierarchical faceted exploration of semantic data using well-known interaction concepts. In *Proc. DCI 2011*, volume 817 of *CEUR-WS.org*, pages 31–36, 2011.

[BHVW00] Mark Bruls, Kees Huizing, and Jarke J Van Wijk. *Squarified treemaps*. Springer, 2000.

[BLCC+06] Tim Berners-Lee, Yuhsin Chen, Lydia Chilton, Dan Connolly, Ruth Dhanaraj, James Hollenbach, Adam Lerer, and David Sheets. Tabulator: Exploring and analyzing linked data on the semantic web. In *Proc. SWUI 2006*, volume 2006, 2006.

[BLHL+01] Tim Berners-Lee, James Hendler, Ora Lassila, et al. The semantic web. *Scientific american*, 284(5):28–37, 2001.

[BR92] John E Bell and Lawrence A Rowe. An exploratory study of ad hoc query languages to databases. In *Proc. ICDE 1992*, pages 606–613, 1992.

[CJ14] Richard Cyganiak and Anja Jentzsch. Linking open data cloud diagram. *LOD Community (http://lod-cloud. net/)*, 2014.

[CKPT92] Douglass R Cutting, David R Karger, Jan O Pedersen, and John W Tukey. Scatter/gather: A cluster-based approach to browsing large document collections. In *Proc. SIGIR '92*, pages 318–329, 1992.

[Clu98] Sophie Cluet. Designing OQL: Allowing objects to be queried. *Information systems*, 23(5):279–305, 1998.

[CM84] William S Cleveland and Robert McGill. Graphical perception: Theory, experimentation, and application to the development of graphical methods. *Journal of the American statistical association*, 79(387):531–554, 1984.

[CMA12] Diego Valerio Camarda, Silvia Mazzini, and Alessandro Antonuccio. Lodlive, exploring the web of data. In *Proc. I–SEMANTICS '12*, pages 197–200, 2012.

[CMS09] Stuart Card, JD Mackinlay, and B Shneiderman. Information visualization. *Human-computer interaction: design issues, solutions, and applications*, 181, 2009.

[CR98] Ed Huai-hsin Chi and John T Riedl. An operator interaction framework for visualization systems. In *Proc. IEEE VIS 1998*, pages 63–70, 1998.

[DF07] Raimund Dachselt and Mathias Frisch. Mambo: A facet-based zoomable music browser. In *Proc. MUM '07*, MUM '07, pages 110–117, 2007.

[DFW08] Raimund Dachselt, Mathias Frisch, and Markus Weiland. Facetzoom: a continuous multi-scale widget for navigating hierarchical metadata. In *Proc. CHI '89*, pages 1353–1356, 2008.

[DK09] Jiri Dokulil and Jana Katreniakova. Rdf visualization-thinking big. In *Proc. DEXA'09*, pages 459–463, 2009.

[DKS06] Leonidas Deligiannidis, Krzysztof J Kochut, and Amit P Sheth. User-centered incremental data exploration and visualization. Technical report, Tech. rep., LSDIS Lab and Computer Science, University of Georgia, Athens, USA, 2006.

[DKS07] Leonidas Deligiannidis, Krys J Kochut, and Amit P Sheth. Rdf data exploration and visualization. In *Proc. CIKM '07*, pages 39–46, 2007.

[DTB08] Danica Damljanovic, Valentin Tablan, and Kalina Bontcheva. A text-based query interface to owl ontologies. In *LREC*, 2008.

[EGK+02] John Ellson, Emden Gansner, Lefteris Koutsofios, Stephen C North, and Gordon Woodhull. Graphviz-open source graph drawing tools. In *Graph Drawing*, pages 483–484, 2002.

[FH06] Amineh Fadhil and Volker Haarslev. Gloo: A graphical query language for owl ontologies. In *OWLED*, volume 216, 2006.

[FSVH06] Christiaan Fluit, Marta Sabou, and Frank Van Harmelen. Ontology-based information visualization: toward semantic web applications. In *Visualizing the semantic web*, pages 45–58. Springer, 2006.

[GKR08] Fredrik Gundelsweiler, Robert Konstanzer, and Harald Reiterer. An innovative user interface concept for large hierarchical data spaces by example of the epdm domain. In *Proc. IUI '08*, pages 421–422, 2008.

[GW04] Sunil Goyal and Rupert Westenthaler. Rdf gravity (rdf graph visualization tool). *Salzburg Research, Austria*, 2004.

[HB10] Jeffrey Heer and Michael Bostock. Crowdsourcing graphical perception: using mechanical turk to assess visualization design. In *Proc. CHI '13*, pages 203–212, 2010.

[HC09] Oktie Hassanzadeh and Mariano P Consens. Linked movie data base. In *LDOW*, 2009.

[HE14] Florian Haag and Thomas Ertl. Filter dials: Combine filter criteria, see how much data is available. In *Proc. AVI '14*, AVI '14, pages 369–370, 2014.

[Hea06] Marti A Hearst. Clustering versus faceted categories for information exploration. *Communications of the ACM*, 49(4):59–61, 2006.

[HGR95] David Haw, Carole Goble, and Alan Rector. Guidance: Making it easy for the user to be an expert. In *Interfaces to Database Systems (IDS94)*, pages 25–48. Springer, 1995.

[HLE12] Florian Haag, Steffen Lohmann, and Thomas Ertl. Simplifying filter/flow graphs by subgraph substitution. In *Proc. VL/HCC 2012*, pages 145–148, 2012.

[HMK05] David Huynh, Stefano Mazzocchi, and David Karger. Piggy bank: Experience the semantic web inside your web browser. In *The Semantic Web–ISWC 2005*, pages 413–430. Springer, 2005.

[HZL08] Philipp Heim, Jürgen Ziegler, and Steffen Lohmann. gFacet: A browser for the web of data. In *Proc. IMC-SSW 2008*, volume 417 of *CEUR-WS*, pages 49–58, 2008.

[Jon98] Steve Jones. Vquery: a graphical user interface for boolean query specification and dynamic result preview. 1998.

[JS91] Brian Johnson and Ben Shneiderman. Tree-maps: A space-filling approach to the visualization of hierarchical information structures. In *Proc. Visualization'91*, pages 284–291, 1991.

[KAC+02] Gregory Karvounarakis, Sofia Alexaki, Vassilis Christophides, Dimitris Plexousakis, and Michel Scholl. Rql: a declarative query language for rdf. In *Proc. WWW'02*, pages 592–603, 2002.

[KB07] Esther Kaufmann and Abraham Bernstein. *How useful are natural language interfaces to the semantic web for casual end-users?* Springer, 2007.

[KBF07] Esther Kaufmann, Abraham Bernstein, and Lorenz Fischer. Nlp-reduce: A "naıve" but domain-independent natural language interface for querying ontologies. 2007.

[KBZ06] Esther Kaufmann, Abraham Bernstein, and Renato Zumstein. Querix: A natural language interface to query ontologies based on clarification dialogs. In *Proc. ISWC 2006*, pages 980–981, 2006.

[LHSZ10] Steffen Lohmann, Philipp Heim, Timo Stegemann, and Jürgen Ziegler. The relfinder user interface: Interactive exploration of relationships between objects of interest. In *Proc. IUI 2010*, pages 421–422, 2010.

[MSR02] Libby Miller, Andy Seaborne, and Alberto Reggiori. Three implementations of squishql, a simple rdf query language. In *The Semantic Web-ISWC 2002*, pages 423–435. Springer, 2002.

[PK95] Anthony Papantonakis and Peter J. H. King. Syntax and semantics of gql, a graphical query language. *Journal of Visual Languages and Computing*, 6(1):3–25, 1995.

[PS+08] Eric Prud'Hommeaux, Andy Seaborne, et al. Sparql query language for rdf. *W3C recommendation*, 15, 2008.

[QHK03] Dennis Quan, David Huynh, and David R Karger. Haystack: A platform for authoring end user semantic web applications. In *The semantic web-ISWC 2003*, pages 738–753. Springer, 2003.

[SGL+05] PAS Sinclair, S Goodall, PH Lewis, K Martinez, and MJ Addis. Concept browsing for multimedia retrieval in the sculpteur project. 2005.

[Shn96] Ben Shneiderman. The eyes have it: A task by data type taxonomy for information visualizations. In *Proc. Visual Languages 1996*, pages 336–343, 1996.

[SKL+11] Dongmin Seo, Hee Kwan Koo, Seungwoo Lee, Pyung Kim, Hanmin Jung, and Won-Kyung Sung. Efficient finding relationship between individuals in a mass ontology database. In *U-and E-Service, Science and Technology*, pages 281–286. Springer, 2011.

[SMS+01] Margaret-Anne Storey, Mark Musen, John Silva, Casey Best, Neil Ernst, Ray Fergerson, and Natasha Noy. Jambalaya: Interactive visualization to enhance ontology authoring and knowledge acquisition in protégé. In *Proc. K-CAP-2001*, 2001.

[SOR+05] Daniel A Smith, Alisdair Owens, Alistair Russell, Craig Harris, Max Wilson, et al. The evolving mspace platform: leveraging the semantic web on the trail of the memex. In *Proc. HYPERTEXT '05*, pages 174–183, 2005.

[SRO+05] Daniel Alexander Smith, Alistair Russel, Alisdair Owens, Craig Harris, Max L Wilson, et al. The mspace classical music explorer: Improving access to classical music for real people. 2005.

[SVHDW+04] Heiner Stuckenschmidt, Frank Van Harmelen, Anita De Waard, Tony Scerri, Ravinder Bhogal, Jan Van Buel, Ian Crowlesmith, Christiaan Fluit, Arjohn Kampman, Jeen Broekstra, et al. Exploring large document repositories with rdf technology: the dope project. *Intelligent Systems, IEEE*, 19(3):34–40, 2004.

[TGM83] Edward R Tufte and PR Graves-Morris. *The visual display of quantitative information*, volume 2. Graphics press Cheshire, CT, 1983.

[VZA+13] R Vesse, RM Zettlemoyer, K Ahmed, G Moore, and T Pluskiewicz. dotnetrdf an open source c#. net library for rdf, 2013.

[Wat99] Martin Wattenberg. Visualizing the stock market. In *Proc. CHI'99*, pages 188–189, 1999.

[WD08] Markus Weiland and Raimund Dachselt. Facet folders: flexible filter hierarchies with faceted metadata. In *Proc. CHI'08*, pages 3735–3740, 2008.

[Wur89] Richard Saul Wurman. *Information anxiety*, volume 1. Doubleday New York, 1989.

[YaKSJ07] Ji Soo Yi, Youn ah Kang, John T Stasko, and Julie A Jacko. Toward a deeper understanding of the role of interaction

in information visualization. *Visualization and Computer Graphics, IEEE Transactions on*, 13(6):1224–1231, 2007.

[YKSJ08] Ji Soo Yi, Youn-ah Kang, John T Stasko, and Julie A Jacko. Understanding and characterizing insights: how do people gain insights using information visualization? In *Proc. CHI '08*, page 4, 2008.

[ZB11] Martins Zviedris and Guntis Barzdins. Viziquer: a tool to explore and query sparql endpoints. In *The Semanic Web: Research and Applications*, pages 441–445. Springer, 2011.

[ZDFB12] Jian Zhao, Steven M. Drucker, Danyel Fisher, and Donald Brinkman. Timeslice: Interactive faceted browsing of timeline data. In *Proc. AVI'12*, AVI '12, pages 433–436, 2012.

[Zlo77] Moshe M. Zloof. Query-by-example: A data base language. *IBM systems Journal*, 16(4):324–343, 1977.

All links were last followed on March 16, 2015.

Declaration

I hereby declare that the work presented in this thesis is entirely my own and that I did not use any other sources and references than the listed ones. I have marked all direct or indirect statements from other sources contained therein as quotations. Neither this work nor significant parts of it were part of another examination procedure. I have not published this work in whole or in part before. The electronic copy is consistent with all submitted copies.

Stuttgart, March 17, 2015.