D 4.1 - Semantic Descriptions for Visual Analytics Components

Summary: This deliverable reports on the development of semantic descriptions for user interface components to visualize Linked Data. The semantic descriptions in the form of a OWL Ontology serve as mapping between the data provider and the visual user interface components. The ontology aims to ease interface development on Linked Data as well as to ensure discoverability of visual components and their use. The ontology bridges between two standards, namely the RDF Data Cube Vocabulary for representing aggregated data sets and the SemanticScience Integrated Ontology for describing visual components. Besides bridging between those standards, we define a complete vocabulary to specify a visual analytics application as data sets, operations (e.g. filters) and a set of visualisations for displaying the data. The current, up-to-date version of the Visual Analytics Vocabulary will be available at http://code-research.eu/ontology/visual-analytics. In addition to the proposed vocabulary, this deliverable presents the context in which the semantic descriptions will be used as well as a corresponding related work analysis.

Key Words: Semantic Descriptions, RDF, OWL, Visualizations, Visual Analytics, Linked Data, Web

Project Acronym	CODE
Grant Agreement number	296150
Project Title	Commercially Empowered Linked Open Data Ecosystems in Research
Date	2012-11-02
Nature	O (Other)
Dissemination level	PU (Public)
WP Lead Partner	Know-Center
Revision	6
Authors	Belgin Mutlu, Patrick Hoefler, Michael Granitzer, Vedran Sabol

SEVENTH FRAMEWORK PROGRAMME

Project Officer & Project Coordinators

Project Officer	Stefano Bertolo	European Commission Directorate-General Communications Networks, Content and Technology (DG CONNECT) Unit Data Value Chain (G3) 10, Rue Robert Stümper 2920 Luxembourg stefano.bertolo@ec.europa.eu
Project Coordinator	Stefanie Lindstaedt	Know-Center GmbH Inffeldgasse 13, 8010 Graz, Austria +43 316 873 30800 slind@know-center.at
Scientific Coordinator	Michael Granitzer	University of Passau Innstrasse 33, 94032 Passau, Germany +49 851 509 3305 michael.granitzer@uni-passau.de

Document Revision History

Revision	Date	Author	Organization	Description
1	2012-09-01	Granitzer	University of Passau	Gathering of Ideas
2	2012-10-17	Mutlu, Hoefler	Know-Center	Draft
3	2012-10-21	Granitzer, Sabol	University of Passau	Review and Comments
4	2012-10-31	Mutlu, Hoefler	Know-Center	Revisions
5	2012-10-31	Granitzer	University of Passau	Review and minor adjustments
6	201-11-02	Mutlu, Hoefler, Sabol	Know-Center	Minor adjustments, formatting

Table of Contents

1 Introduction

In recent years, Linked Data has become the standard for expressing data in a semantic way. As of August 2011, the Linking Open Data Cloud Group at CKAN[1] had indexed 295 data sets totalling over 30 billion triples[2]. In October 2012, the number of data sets has increased to more than 330.

One goal of the CODE project is to make this huge amount of data easier accessible to people who are neither Semantic Web experts nor programmers. WP4 of CODE therefore deals with the development of web-based visual analytics user interfaces for Linked Data. This means that it should be possible for people who don't know anything about RDF or SPARQL to access information that is already available but hardly accessible in the Linked Open Data Cloud.

WP4 consists of two major components:

- The **Query Wizard** helps people select relevant data from the Linked Data Cloud — to be more exact, from a SPARQL 1.1 endpoint. The resulting data will be presented to the users in an easy-to-use web-based interface that looks and feels similar to current spreadsheet applications. For advanced data manipulation, e.g. aggregations, the tabular data will be converted into the RDF Data Cube Vocabulary[3].
- Once the user has selected the relevant data, the **Visualization Wizard** will support the user in creating one or more visualizations to make the data easier to understand. At a later stage, these combined visualizations should form a visual analytics interface. In order to facilitate this step, not only the data but also the visualizations need to have semantic descriptions.

The following figure combines a conceptual overview of this deliverable with its structure. It shows how the Query Wizard, the Visualisation Wizard, and the used vocabularies interact.

Our scientific main contributions in this deliverable are as follows:

- The definition of a Visual Analytics vocabulary that
  - maps data (using the RDF Data Cube Vocabulary) to visualizations (using the Semantic Science Integrated Ontology) and
  - adds semantic concepts and properties where necessary.
- As a result, we will be able to semantically define Visual Analytics Dashboards and allow the users the share them, including their state and the detected hypotheses. The Visual Analytics Vocabulary can therefore be used as a persistence format for visual analytics applications and their data. Such a persistence format allows sharing different states of a visual analytical process with standard web technology.

[1] http://thedatahub.org/group/lodcloud
[2] http://www4.wiwiss.fu-berlin.de/lodcloud/state/
[3] http://www.w3.org/TR/vocab-data-cube/#cubes-model

- A concept for an easy-to-use query wizard, focusing on creating RDF Data Cubes.
- A concept for an easy-to-use visualization wizard that creates visualizations from RDF Data Cubes.

All concepts are currently under development becoming full-fledged prototypes and a corresponding reference architecture for the defined format.

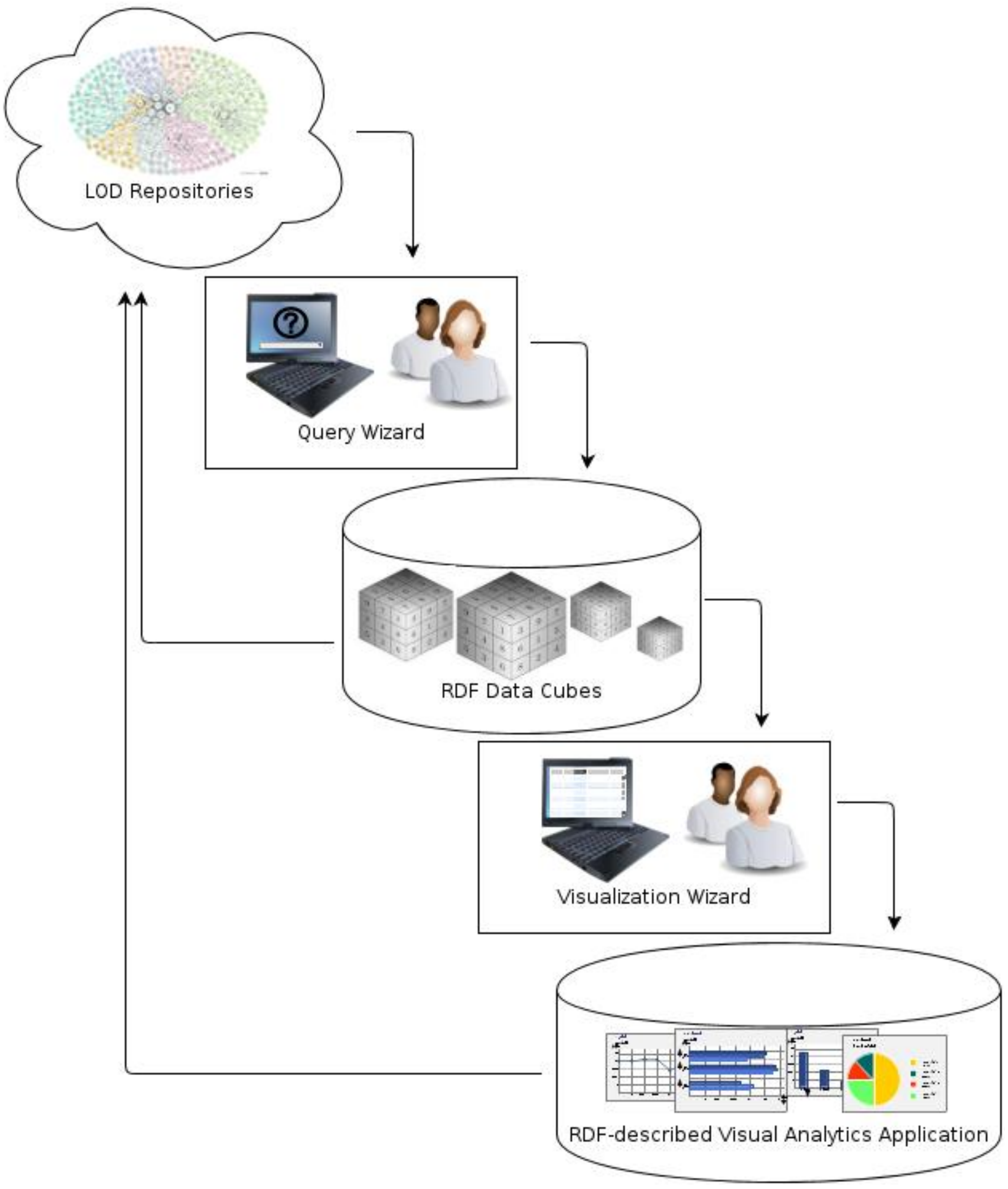


Figure 1: WP4 Data Flow

The focus of this deliverable is on the semantic descriptions of the visual analytics components. Leading up to this, the deliverable presents the context in which these semantic descriptions will be used. Therefore, the deliverable is structured as follows

- **Chapter 2** presents the concept of the **Query Wizard** and related work.
- **Chapter 3** presents the concept of the **Visualization Wizard** and related work.
- **Chapter 4** presents the **Visual Analytics Vocabulary** that will be used to describe the visual analytics components, the mapping between the data and the charts, as well the semantic descriptions of visual analytics applications. Related vocabularies (RDF Data Cube Vocabulary and SemanticScience Integrated Ontology) are also presented.
- **Chapter 5** provides some final **Conclusions**.
- **Chapter 6** presents the **Bibliography**.
- **Appendix A** presents the **Working Draft of the Visual Analytics Vocabulary** in the form of an OWL ontology written in Turtle.

2 Query Wizard

The Query Wizard should support users in querying data from the Linked Data Cloud. Usually, SPARQL[4] queries are used to query Linked Data. While SPARQL queries represent a powerful tool to query RDF and graph-based data, it also represents a huge barrier for users who don't speak SPARQL.

2.1 Related Work

Before we present the current prototype, we first take a look at a few examples of what has already been achieved in the field of query wizards.

2.1.1 Semantic Web Search Engines

One possibility to retrieve information from a Linked Data repository is the use of Semantic Web search engine. Both of the following search engines use interfaces similar to the well-known search paradigms employed by current "regular" search engines.

Falcons[5]

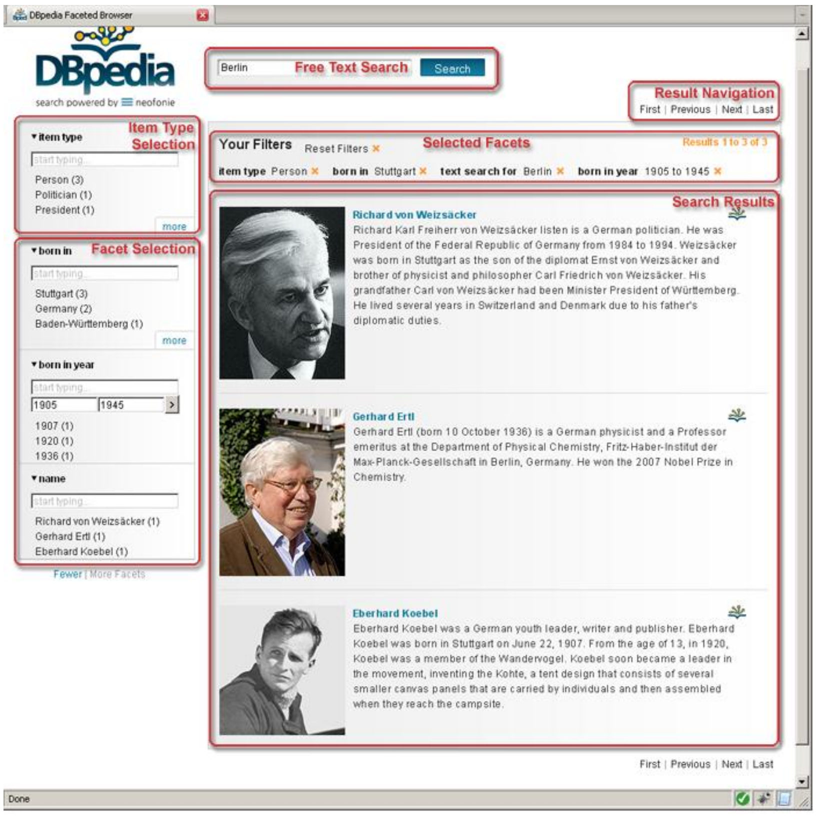
Falcons provides a keyword-based search for Semantic Web entities. Search is performed in two steps:

- concept search (in concepts, classes, properties)
- object search (individual resources or entities in resource): In this layer is it possible to filter the type, the result is a list of resources with individual properties.

Faceted Wikipedia Search[6]

Faceted Wikipedia Search is an entity-set based browser. It allows users to ask complex questions, like „Which rivers flow into the Rhine and are longer than 50 kilometres? ". Users can narrow down the result set along several criteria (Facets), e.g. the position of a place, the birthday of a person, or the height of a mountain.

[4] http://www.w3.org/TR/sparql11-query
[5] http://ws.nju.edu.cn/falcons
[6] http://wiki.dbpedia.org/FacetedSearch



Richard von Weizsäcker
Richard Karl Freiherr von Weizsäcker listen is a German politician. He was President of the Federal Republic of Germany from 1984 to 1994. Weizsäcker was born in Stuttgart as the son of the diplomat Ernst von Weizsäcker and brother of physicist and philosopher Carl Friedrich von Weizsäcker. His grandfather Carl von Weizsäcker had been Minister President of Württemberg. He lived several years in Switzerland and Denmark due to his father's diplomatic duties.

Gerhard Ertl
Gerhard Ertl (born 10 October 1936) is a German physicist and a Professor emeritus at the Department of Physical Chemistry, Fritz-Haber-Institut der Max-Planck-Gesellschaft in Berlin, Germany. He won the 2007 Nobel Prize in Chemistry.

Eberhard Koebel
Eberhard Koebel was a German youth leader, writer and publisher. Eberhard Koebel was born in Stuttgart on June 22, 1907. From the age of 13, in 1920, Koebel was a member of the Wandervogel. Koebel soon became a leader in the movement, inventing the Kohte, a tent design that consists of several smaller canvas panels that are carried by individuals and then assembled when they reach the campsite.

Figure 2: Faceted Search: User Interface[7]

2.1.2 Visual Query Builders

Another way to support the users in building the query is to use visual interfaces. Two examples are GoRelations and CQL.

GoRelations[8]

The main concept of GoRelations is to support the user graphically in generating SPARQL queries. GoRelations aims at enabling the users to easily generate a query over a question or problem description in order to retrieve an answer.

The user formulates the query by building a simple visual representation, called the Semantic Graph, which contains only a few blocks and relations between them. The visual representation will be converted to a SPARQL query.

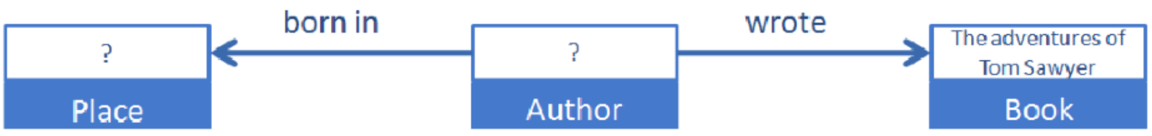


Figure 3: Example for a Semantic Graph in GoRelations

[7] http://wiki.dbpedia.org/FacetedSearch
[8] http://ebiquity.umbc.edu/paper/html/id/557/GoRelations-An-Intuitive-Query-System-for-DBpedia

The above visual representational has the following meaning:

- The name of the book should be “The Adventures of Tom Sawyer”
- The name of the author is unknown
- The place of birth is also unknown

This graph will be translated in this SPARQL query:

```
select ?place ?author WHERE { ?author :wrote „Tom Sawyer“ . ?author :born ?place }
```

Since the non-expert user is not familiar with the SPARQL syntax, i.e. what the triples do mean, he is not able to correctly construct the SPARQL query. To still allow such a user group to query the ontology, the framework provides several steps which assist the user to optimize his results. As first, the entered user keyword is directly compared with each of the triple entries (with the subject, predicate and object) in the ontology.

1. *Statement of the semantic similarities*: There are more results for place and this is why the framework gives the similarity value between 0 and 1 (e.g. author and writer have the similarity of 0.76)
2. *Disambiguation*: This step is used to filter out the meaningful interpretations.
3. *Polish:* User can remove properties they have are mapped incorrectly (e.g. @name property from wrote property, see Figure 4).

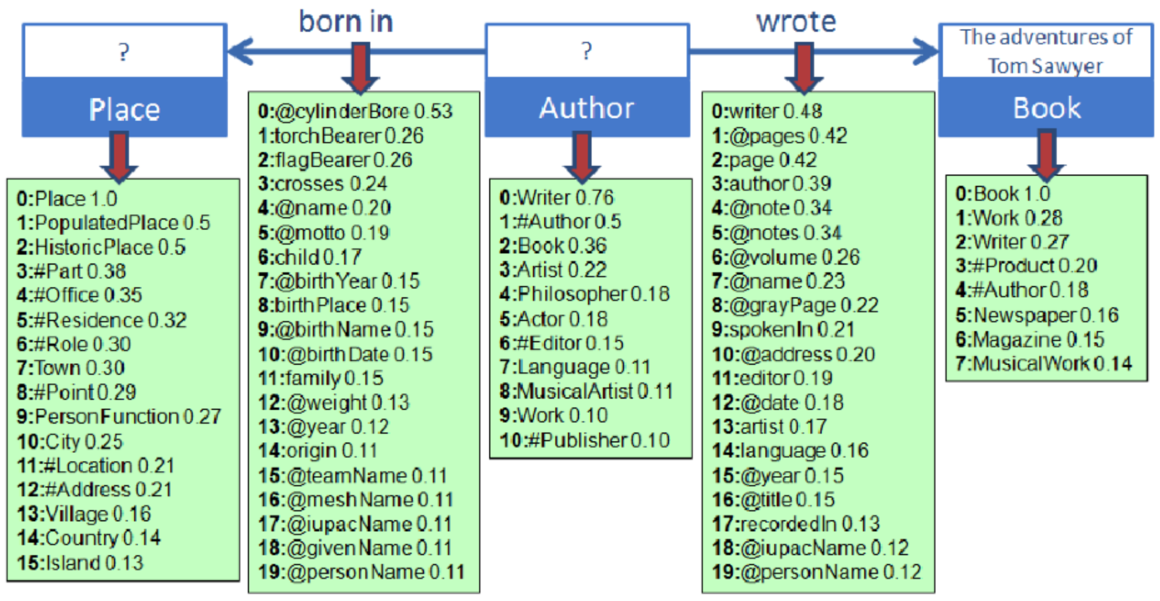
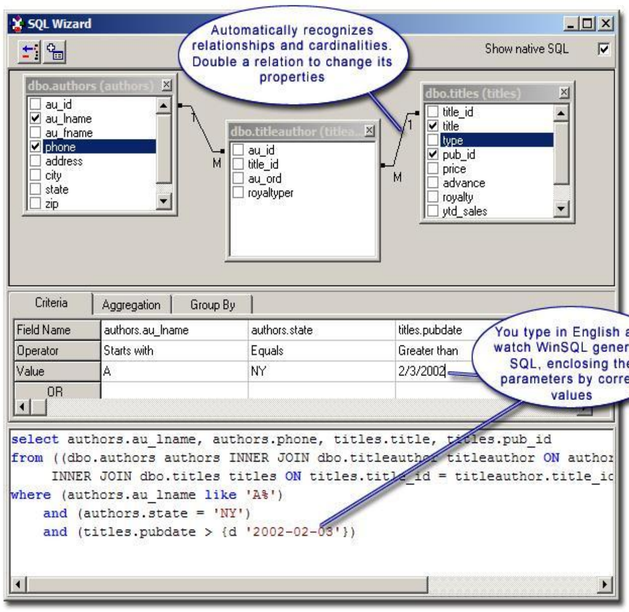


Figure 4: Possible terms of the ontology

CQL

The *Continuous Query Language* is a query language for SQL. It supports users in building SQL Queries without any knowledge about the syntax of SQL.

The system provides *skeletons* which are filled by users with exemplary elements, comparison elements, and commands.



Field Name	authors.au_lname	authors.state	titles.pubdate
Operator	Starts with	Equals	Greater than
Value	A	NY	2/3/2002
OR			

```
select authors.au_lname, authors.phone, titles.title, titles.pub_id
from ((dbo.authors authors INNER JOIN dbo.titleauthor titleauthor ON author
     INNER JOIN dbo.titles titles ON titles.title_id = titleauthor.title_ic
where (authors.au_lname like 'A%')
    and (authors.state = 'NY')
    and (titles.pubdate > {d '2002-02-03'})
```

Figure 5: SQL Wizard [9]

Method

- User gets a skeleton and selects the required attributes from it
- *FieldName* contains which attribute have been selected
- *Operator* contains the fixed operators, which can be selected
- When the skeleton is ready, the SQL query will be generated

2.1.3 Conclusions

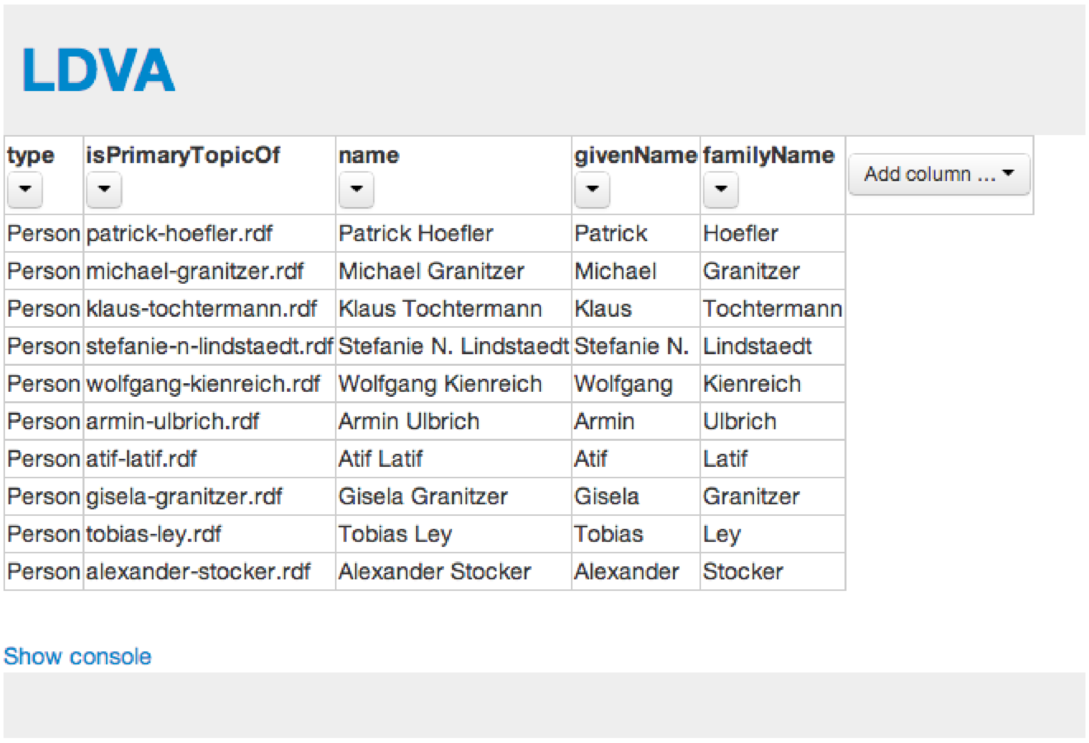
All previous approaches aim to provide a general query builder for non-IT experts. Without knowledge on the domain of the query, building a highly usable wizard becomes challenging. Since CODE envisions the data-warehousing-like queries and tabular result sets with aggregated results, we can:

1. Restrict the target user group to people capable working with spread sheet like data and
2. Use the well-known table metaphor for realizing a query wizard.

[9] http://synametrics.com/SynametricsWebApp/ISQLWiz.jsp

2.2 Current CODE Prototype

The current prototypical implementation of CODE's Query Wizard is displayed in Figure 7. Different to generic Wizards, it emphasizes the creation of tabular, data warehouse like data and assumes that users are familiar with spread sheet like applications.



type	isPrimaryTopicOf	name	givenName	familyName
Person	patrick-hoefler.rdf	Patrick Hoefler	Patrick	Hoefler
Person	michael-granitzer.rdf	Michael Granitzer	Michael	Granitzer
Person	klaus-tochtermann.rdf	Klaus Tochtermann	Klaus	Tochtermann
Person	stefanie-n-lindstaedt.rdf	Stefanie N. Lindstaedt	Stefanie N.	Lindstaedt
Person	wolfgang-kienreich.rdf	Wolfgang Kienreich	Wolfgang	Kienreich
Person	armin-ulbrich.rdf	Armin Ulbrich	Armin	Ulbrich
Person	atif-latif.rdf	Atif Latif	Atif	Latif
Person	gisela-granitzer.rdf	Gisela Granitzer	Gisela	Granitzer
Person	tobias-ley.rdf	Tobias Ley	Tobias	Ley
Person	alexander-stocker.rdf	Alexander Stocker	Alexander	Stocker

Figure 6: Current prototypical implementation of the Query Wizard

After the user has entered a search term, the matching results are displayed in a table. If the results table contains numeric attributes (e.g. population, age, height, …), it is already suitable for visualization. For more advanced visualizations, e.g. based on aggregations, it first needs to be converted in a more suitable format. In the context of CODE, this format is the RDF Data Cube Vocabulary, which will be described in more detail later. Also, for expressing statistical data extracted from tables, the RDF Data Cube Vocabulary is much better suited than "generic" RDF. The interfaces and methods of how to turn "generic" RDF into RDF Data Cubes with the help of the user are currently being investigated, especially considering that the typical users of this prototype are non-experts with no or a very limited knowledge of semantic technologies.

The interface will also provide easy-to-use mechanisms to filter the data in order to provide the user with a table that contain exactly the data the user is interested in. Looking at the example above, this might include filters to only show people whose name contains certain letters, or to filter the results to only contain entities of the type "Person" in the first place.
Additionally, the interface will provide ways to add new columns with related data or expand columns with their super/sub-classes if available. The nature of Linked Data — namely the fact that is already linked on a semantic level — opens some nice opportunities, which will be explored and researched during the development of the prototype.

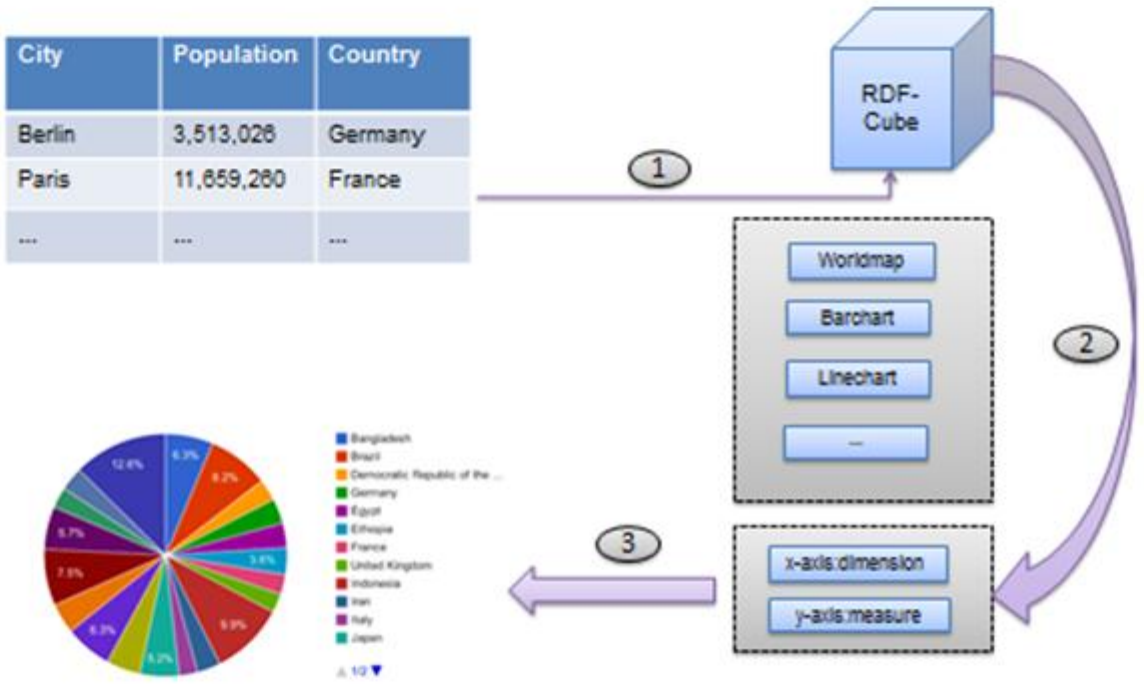
3 Visualization Wizard

The aim of the Visualization Wizard is to visualize the result of the Query Wizard, represented as a RDF Data Cube. The wizard will support a variety of chart types for the visualization.

3.1 Work Flow

The visualization process will be performed in the following steps:

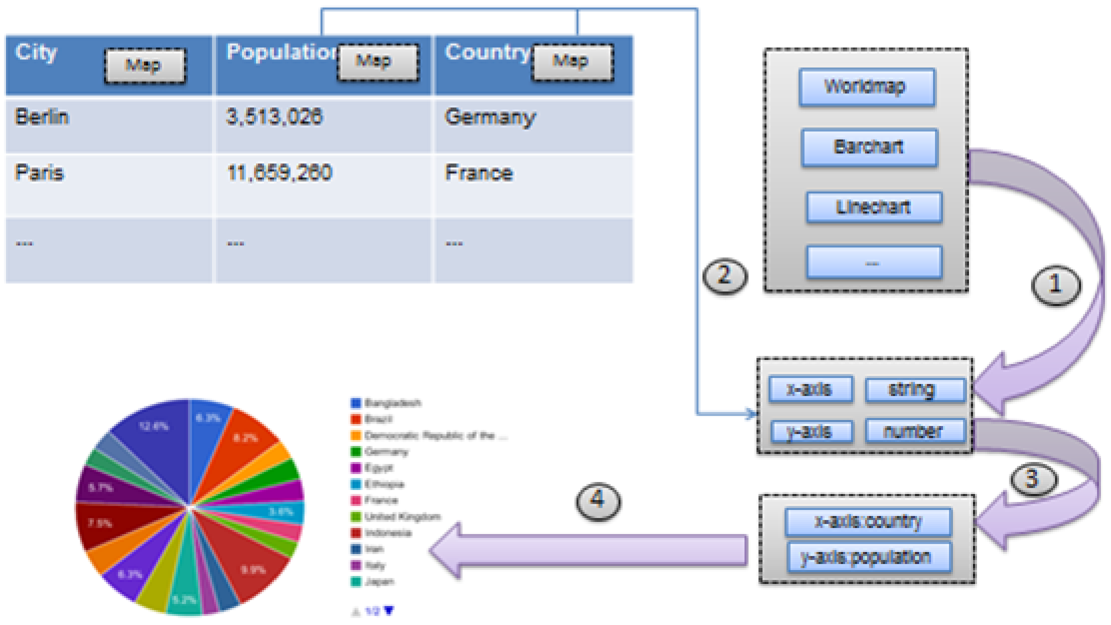
- After the relevant data has been selected using the SPARQL Wizard, the semantically annotated HTML table is converted to a RDF Data Cube. (The exact mechanisms and user interfaces are currently being researched.)
- The Visualization Wizard then suggests one or more chart types to the user. The user can choose one of these chart types for the visualization.
- Next, the values of the data need to be mapped to the different visual channels of the respective chart (more on that later).
- Finally, the chart is created.



City	Population	Country
Berlin	3,513,026	Germany
Paris	11,659,260	France
...	...	...

Figure 7: Conceptual work flow of the Visualization Wizard visualizing an RDF Data Cube

Optionally, instead of using an RDF Data Cube, it might also be possible to use “generic” RDF as input for the Visualization Wizard. The work flow would be similar to the one described above and is depicted in the following figure.



City	Population	Country
Berlin	3,513,026	Germany
Paris	11,659,260	France
...	...	...

Figure 8: Conceptual work flow of the Visualization Wizard visualizing “generic” RDF

One research questions that will be addressed during the development of the Visualization Wizard (and the Query Wizard, for that matter) will be how the “semantic lifting” of generic RDF into RDF Data Cubes can help in the visualization process and the (semi-)automatic support of the user when creating visualizations from Linked Data.

The primary approach will be to turn generic RDF into RDF Data Cubes and visualize those. Should this approach turn out unfeasible for certain use cases, the visualization of generic RDF might offer an alternative solution.

3.2 Conceptual Architecture

This following figure describes the technical architecture of the Visualization Wizard from a developer's point of view.

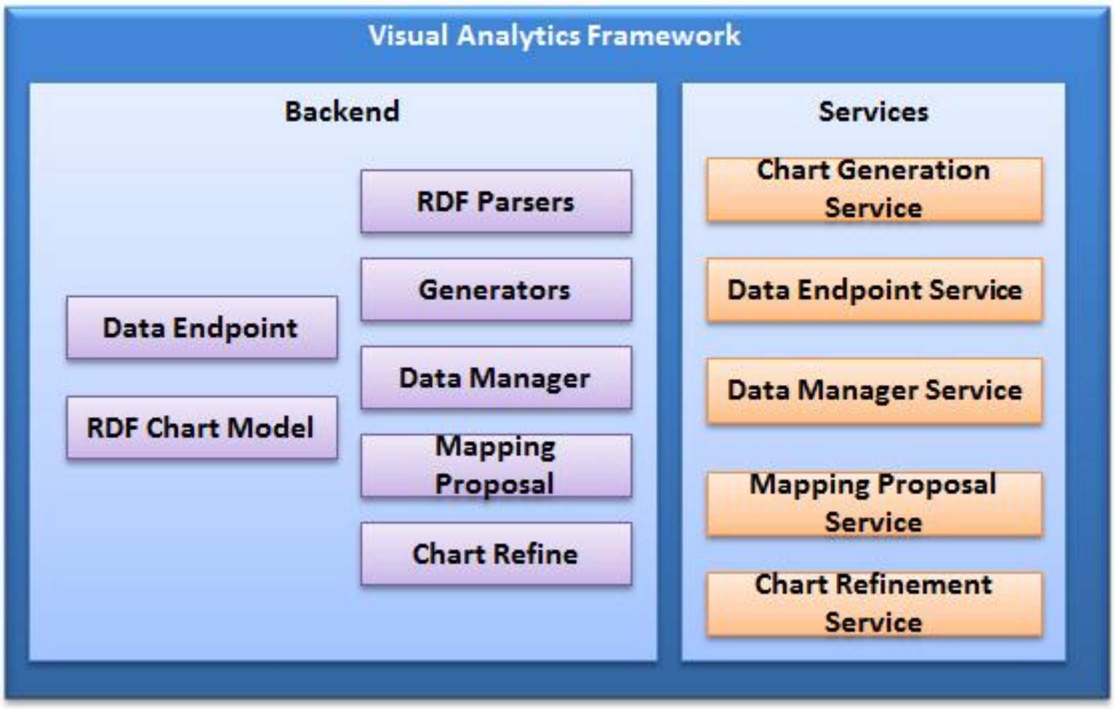


Figure 9: Conceptual architecture of the Visualization Wizard

Backend
The backend consists of the following components:

<u>RDF Parser</u>
Since the data as well as the ontologies / vocabularies that we work with are defined in RDF, we need an RDF parser. We well be able to make use of readily available open source RDF libraries for Python.

<u>Generators</u>
The source code of a chart will be generated in generators. Therefore, the results of a SPARQL query — e.g. the content of the RDF Data Cube Vocabulary — should be merged to the source code of the respective chart. The finished chart will be displayed to the client in the browser.

<u>Data Manager</u>
The data manager is responsible for setting up the content of the database (components, source code etc. of a chart - based on a specific technology such as D3). Additionally, it

executes the methods that interact with the initialized data and are necessary to generate a chart.

RDF Chart Model
For the description of the charts and their visual elements as well as description of the mapping from RDF Data Cube data points to the chart a new Visual Analytics Vocabulary will be defined (see Chapter 4.2.1).

Data Endpoint
The endpoint will be provided by the University of Passau. It will provide “generic” RDF data from the Linked Open Data cloud as well as RDF data conforming to the RDF Data Cube Vocabulary.

Mapping Proposal
The Visualization Wizard might suggest mapping options based on data and chart properties or mappings done by previous users.

Chart Refine
The purpose of the *Chart Refine* is the configuration of the charts, i.e. the variability of the dynamic components of a chart (height, interval selection, title, etc.).

Services
The framework will be implemented as a service-oriented architecture. Thus the individual tasks like mapping definitions, generators, and automatic chart suggestion can be realized as independent services.

3.2 Web-Based Charting Solutions

The technical implementation of the web-based charts will be based on existing solutions. The charts can be generated using the following charting frameworks, which are explained in the following:

- SIMILE Widgets
- Raphaël
- Google Chart Tools
- Protovis
- D3

3.2.1 SIMILE Widgets

The SIMILE Widgets are an open-source JavaScript-based framework, developed by the MIT (Massachusetts Institute of Technology) as a part of the research project SIMILE (Semantic Interoperability of Metadata and Information in unLike Environments). The framework is based on semantic web technologies with the purpose to visualize statistical data.

This framework offers four different groups of visualization types, so called widgets: two for highly interactive timeline charts (e.g. Roadmap, XY-charts, see Figure 10), one for interactive maps, and one for the interactive visualization of the pictures in the form of an iTunes Cover Flow.

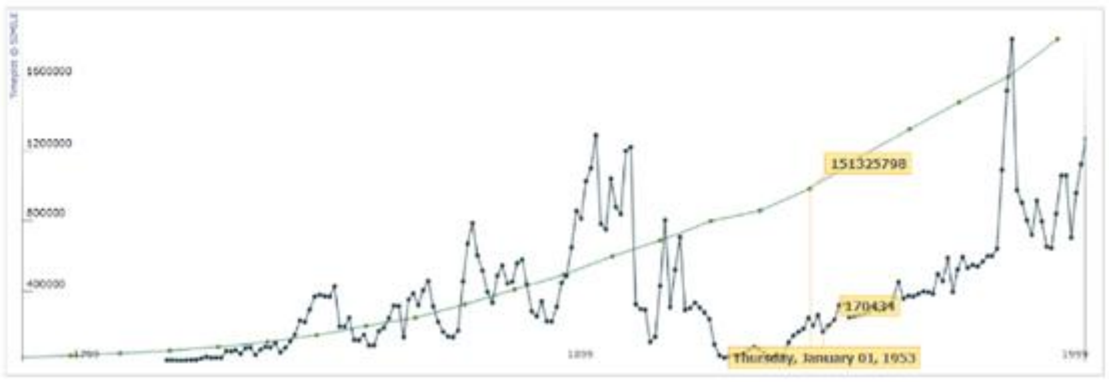


Figure 10: Example for a Timeline, created with SIMILE API[10]

3.2.2 Raphaël

Raphaël[11] is a highly flexible JavaScript-based framework with the aim to create vector graphics. It uses the Vector Graphic Standard SVG, which is interpreted by most modern browsers. It also provides an interface for the Internet Explorer through Microsoft VML (Vector Markup Language).

3.2.3 Google Chart Tools

Google Chart Tools[12] is a JavaScript-based API to create charts. The statistical data is interpreted by the back end (Google) and the resulting chart is displayed in the browser using HTML5 technologies.

3.2.4 Protovis

Protovis[13] is a JavaScript-based framework which has been developed at the University of Stanford (Stanford Visualization Group) primarily to visualize statistical data. In 2001 the development of Protovis stopped due to the focus on its successor D3.

3.2.5 D3

D3[14] (Data Driven Documents) is a JavaScript library for data visualization and is the official successor of the Protovis framework. It produces SVG, provides the possibility to easily create transitions, and allows the representation of large data sets.

[10] http://simile-widgets.org/timeplot/
[11] http://raphaeljs.com/
[12] https://developers.google.com/chart/interactive/docs/gallery
[13] http://vis.stanford.edu/papers/protovis
[14] http://d3js.org/

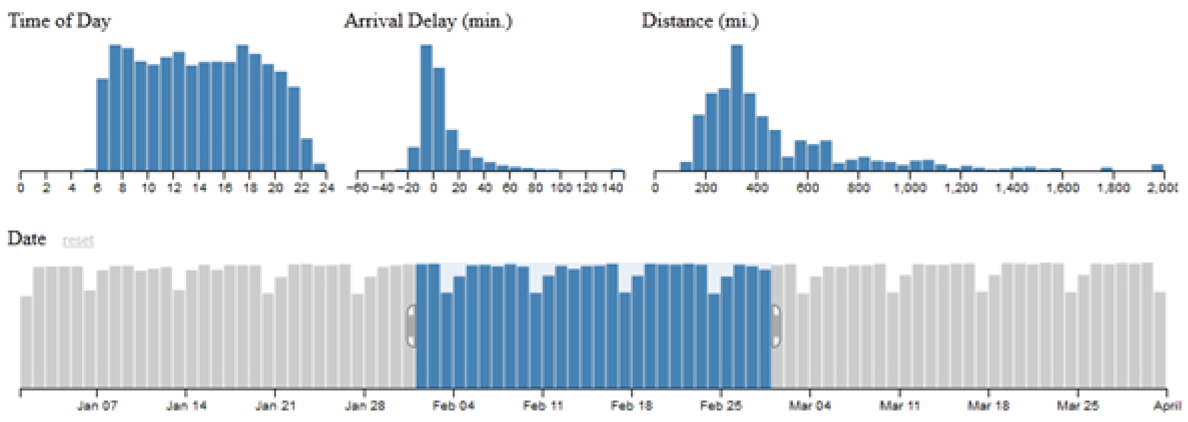


Figure 11: An example for D3: Airline on-time performance

3.2.6 Conclusions

After examining the web-based charting technologies available today, the plan for the development of the visualization wizard is the following:

- Google Charts will be used in the first iteration of the prototype due to the easy integration into the prototype.
- For the following iterations, open-source solutions will be looked into, especially Raphaël and D3. Depending on the experiences during the development of the prototype, one or both of these technologies will be used.
- For special or advanced visualizations, SIMILE widgets or other, not mentioned technologies might offer better solutions. Those will be investigated in more detail should the need arise.

4 Visual Analytics Vocabulary

Semantic description of charts using RDF is a new research topic and the literature, up to now, offers only few related works. In the following chapter we describe some selected works that have dealt with it and afterwards introduce the developed vocabulary.

4.1 Related Work

4.1.1 iGraph-Lite

The authors, Leo Ferres, Verkhogliad Petro and Lindgaard Gitte, of the paper *Improving Accessibility to Statistical Graphs: The iGraph-Lite System* [2] have developed a system, named iGRAPH-Lite, that provides short verbal description of the information contained in statistical graphs and a method to also interact with this information.

The framework is tailored to a special user group, but the most relevant feature for our work is the graph model behind it. In short, the framework maintains a chart knowledge base (KB), which allows the user to navigate, thought the content of the charts. As an input, the KB takes the formal description of the visual representation. Further, the results of queries are provided to the last component of the framework, which generates spoken text based on these results.

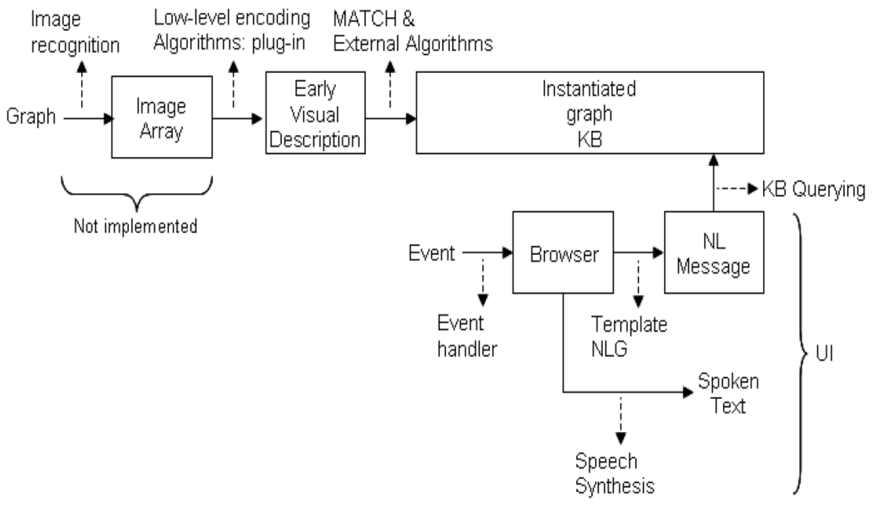


Figure 12: Architecture of iGraph-Lite [2]

Workflow

The process starts with *Early Visual Description* where statistic data, stored in an Excel file, is translated into an XML-based chart model (see Figure 13). There exist generic mechanisms for querying the charts, which can be also applied in this case. Therefore, in the next step this XML-based chart model will be instantiated in the KB.

For querying, the framework provides a set of algorithms for different purposes, like interval querying, shape matching, etc. This means that for each query type an appropriate algorithm is required.

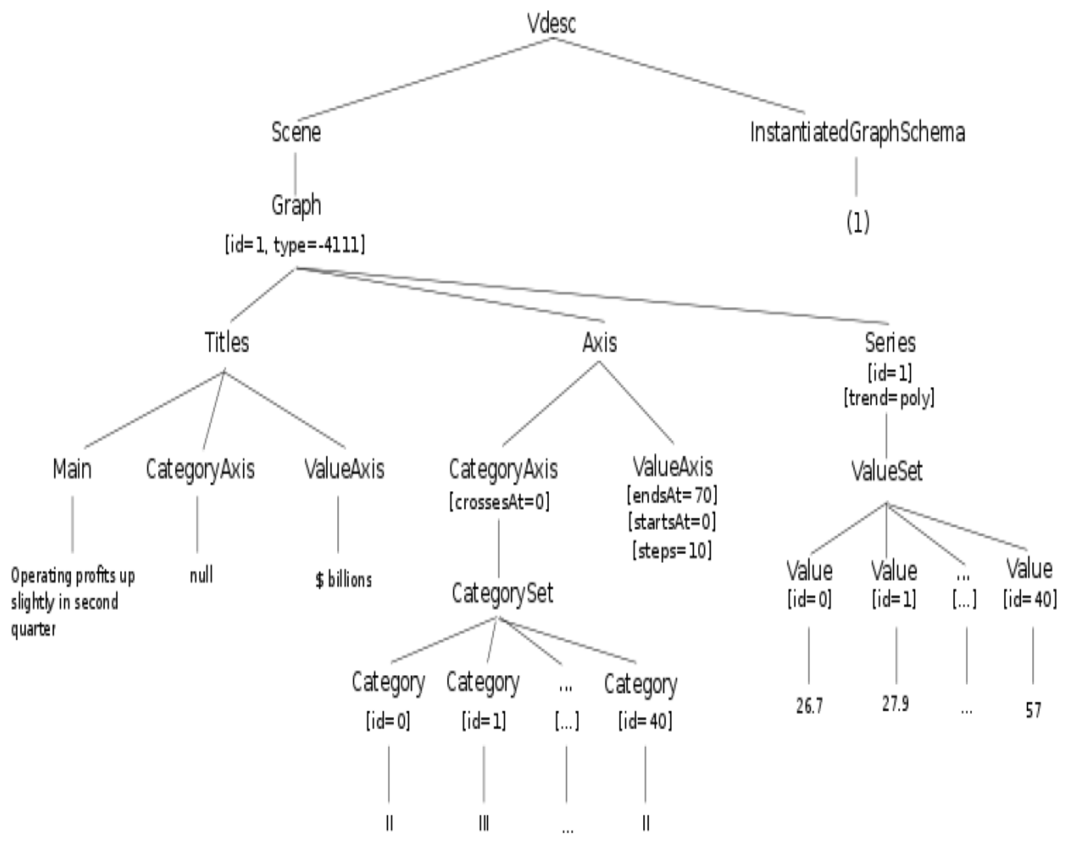


Figure 13: Graph model of a line chart

4.1.2 XML-SVG

The focus of the paper *Search and inference with Diagrams* [3] is the presentation of diagrams (bar, line, and pie charts) on the *World Wide Web* so that the information can be accessed and queried by inference engines The developed process stores chart information in XML documents and a server-side rendering engine present this information to the user graphically in SVG.

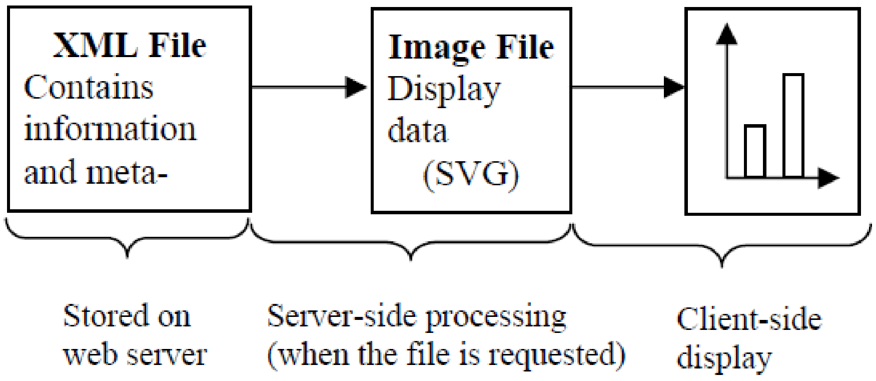


Figure 14: The primary components of the system [3]

The XML Schema of the system can be used for information from Microsoft Excel Spreadsheets to convert these into the XML format. An excerpt of an XML-based chart is shown below.

```
<?xml version="1.0" ?>
<chart type="bar">
  <title>4-Year GPA by Gender</title>
  <legend />
  <plotarea>
    <axis variable=”x1”>Year</axis>
    <axis variable=”x2” min="0" max="4.0"
         step="0.5">GPA</axis>
  </plotarea>
  <dataset label="Men">
    <datapoint x1="2000" x2="2.6" />
    <datapoint x1="2001" x2="2.8" />
    <datapoint x1="2002" x2="2.9" />
    <datapoint x1="2003" x2="3.0" />
  </dataset>
```

Figure 15: XML file describing a bar chart [3]

4.1.3 XBrain

The paper *A Framework for XML-based Integration of Data, Visualization and Analysis in a Biomedical Domain* [4] describes a framework for managing, sharing, integrating, and visualizing complex, heterogeneous, and multi-modality data about the human brain.

The interesting part of the contribution in this work is the following:

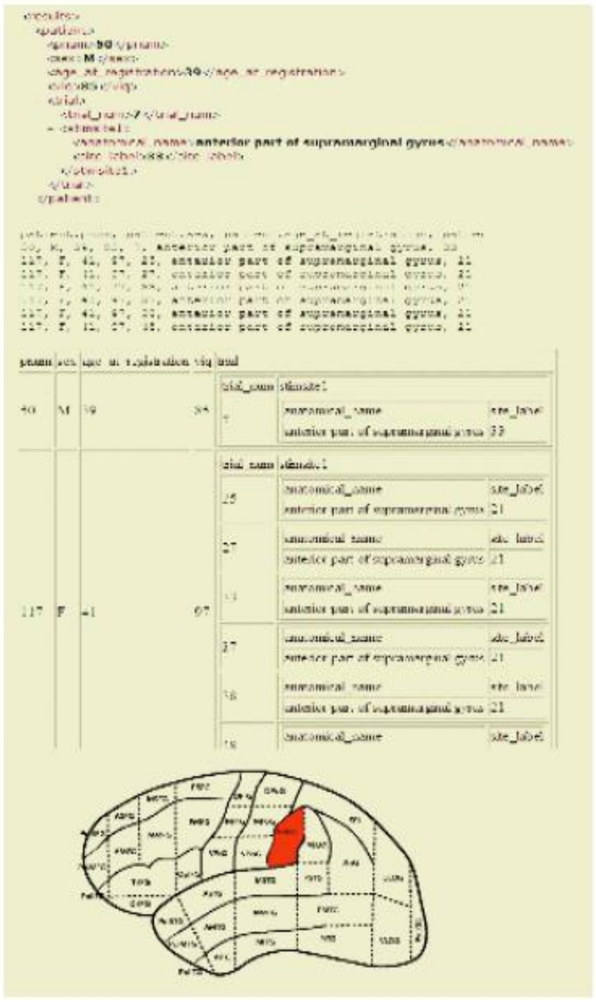
- a simple extension to XQuery, RXQuery, to define the complex mapping, i.e mapping relational data to XML data
- a simple interface to map the data into a variety of formats for direct visualization
- developing of multiple web-based data visualization tools

Mapping to XML

To define the map, it is necessary to write an XQuery program that maps the entire relational database to a virtual XML document, called the *public view*. Users write queries in XQuery language, which will be internally translated to SQL in order to correctly map the entire relational database into XML representation. This process is, however, hidden to the user. The answer will be provided in form of an XML document.

Integration of Heterogeneous Visualization Tools

This work also provides a set of independent tools that can run as web services for a 2-D image visualization tool that accepts the above mentioned XML output and generates an image showing the locations on a 2-D sketch of the brain where specific types of language processing occur.



(a)

```
<results>
  { for $trial in PublicView("Scrubbed.pv")
      /patient/surgery/csmstudy/trial
    where $trial/trialcode/term/abbrev/text()="2"
    return
      <answer>
        <name>
          { $trial/stimsite/name() }
        </name>
        <image-anchor>
          <uri>
            http://csm.biostr.washington.edu/vbm
          </uri>
          <param>
            { $trial/stimsite/name() }
          </param>
          <param>
            gray
          </param>
        </image-anchor>
      </answer>
  }
</results>
```

(b)

Figure 16: Different query output formats (XML, CSV, HTML and Image) (a) [4]. Query for embedding images in XML files (b).

One of the most relevant features in this framework is the integration of heterogeneous data sources in a common XML-based format. Furthermore, the framework allows to generate such a common data representation in various formats and hence to attach different visualization tools (e.g. HTML, CSV, etc.). However, the authors do not go deep into details about the common data format, which is here the reason for the heterogeneity. There is also no discussion about techniques (i.e. chart models) to display patient data.

4.1.4 Statistical Graph Ontology

The authors, Michel Dumontier, Leo Ferres and Natalia Villanueva-Rosales, of the paper *Modeling and querying graphical representations of statistical data* [5] present a new approach to manage statistical graph knowledge by semantic annotation of graphs. The aim of this work is to make statistical graphs available as highly structured representations that can be queried, exchange, integrated, and whose structure can be extended using Semantic Web technology [5]. The focus of this work lies on the multi-variable line graphs, which appear in Statistics Canada’s online publication, “The Daily”. The final vocabulary is presented as statistical graph ontology in[15].

A statistical graph in SGO (Statistical Graph Ontology, developed by *SemanticScience*[16]*)* is defined with following objects and properties:

- A *StatisticalGraph* **hasTitle** (one PrimaryTitle, some SecondaryTitle)
- *SecondaryTitle* **isPartOf** some *Graph* that **hasPart** some *PrimaryTitle*
- A *StatisticalGraph* **hasPart** exactly one *Plot*
- A *Plot* **hasPart** some *Series* and **hasPart** some *CoordinateAxis* (XAxis, YAxis and ZAxis).
- *CoordinateAxis* **hasPart** one or more *CategoryAxis* (PrimaryCategoryAxis and SecondaryCategoryAxis) and one or more *ValueAxis* (LeftValueAxis or RightValueAxis).
- *CategoryAxis* **hasPart** CategoryData and, more specifically, *PrimarCategoryAxis* **hasPart** *PrimaryCategoryData* and *SecondaryCategoryAxis* **hasPart** *SecondaryCategoryData*
- *ValueAxis* is usually used to represent continuously varying numerical data (*ValueData*, which **isPartOf** some *ValueAxis*) and may be scaled
- A Series **hasPart** two or more *DataPoint*, that **hasPart** Data to be plotted from the set of *CoordinateAxis*
- *ValueData* **hasValue** a concrete numerical data type (e.g., int, real, float, double)
- **hasTag** is applied to existing ontology in order to allow users to annotate their charts with additional semantics using keywords. Based on these keywords, the search for specific information is simplified, since **hasTag** can refer to any resource.

[15] http://www.inf.udec.cl/~leo/ontologies/
[16] http://code.google.com/p/semanticscience/

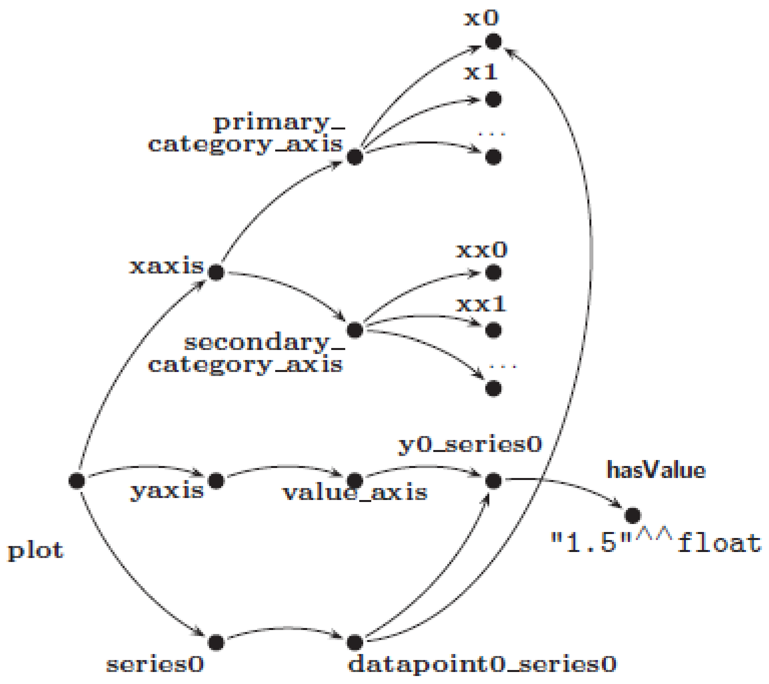


Figure 17: Diagrammatic representation of some of the relations between the different object instances of the ontology [5]

Graph querying

There exist two general types of queries to ask the graphs:

1. Queries about the objects composing a single graph (intra-graphical queries)
2. Queries that are asked to a graph (inter-graphical queries)

The authors use queries, which are specified in Manchester OWL Syntax with some new keyword, like “and” and “that”, which denote the logical operator *and,* “or”, which denotes the logical inclusive *or* and “some”, which stands for “at least one”.

An example for the first observation is shown below,

```
ss:ValueData that ss:isPartOf some (ss:DataPoint
that ss:hasTag value ss:May_2006)
```

This query asks for ValueData that isPartOf some (Datapoint that hasTag value May_2006). Query returns the value for the month of May 2006.

Result explanation:
Individual **y28_Series0** is of type **ValueData**

```
<dl:ValueData rdf:about="#y28_Series0">
   <dl:hasValue rdf:datatype="http://www.w3.org/2001/XMLSchema#float">1.8</dl:hasValue>
 </dl:ValueData>
```

and **datapoint28_Series0 hasPart y28_Series0**.

```
<dl:DataPoint rdf:about="#datapoint28_Series0">
   <dl:hasPart rdf:resource="#x28"/>
   <dl:hasPart rdf:resource="#y28_Series0"/>
 </dl:DataPoint>
```

Since the relation **isPartOf** is the inverse of **hasPart**, therefore **y28_Series0 isPartOf datapoint28_Series0**. Hence, **datapoint28_Series0 hasPart x28,** which **hasTag May_2006.**

```
<dl:CategoryData rdf:about="#x28">
    <dl:hasTag rdf:resource="http://semanticscience.org/ontology/May_2006"/>
  </dl:CategoryData>
```

<u>An example for the second observation is shown below</u>

```
 DataPoint that (isPartOf some (Series that hasTag
some Sales)) and (hasTag some (Month and (precededBy
value January 2005) and (precedes value May 2008)))
and (hasPart some ((hasValue some float[<41.0]) or
(hasValue some float[>48.0]))))
```

This query is an example of an inter-graphical query using the information given in tags. It asks to return the data points that belong to a graph that contains information about "Sales" after January 2005 and before May 2008 and have a value lower than 41 or greater than 48. The result includes data points from two graphs, since these graphs have series that contain information about "Sales" by using the **hasTag** property.

<u>Visual queries</u>

To query graphs using a language based on the visual properties of the lines in the graphs, the authors have implemented two kinds of visual tags:

1 querying by slope event (visual slop)
2 querying by slope quality

The first kind of visual queries based on the direction of slopes, i.e whether they go up, down or stays unchanged. The predicates, such *AcceleratingSlope*, *PlummetingSlope*, *AdvancingSlope*, *EdgingUpSlope*, *IncreasingLine*, *DescreasingLine* and *PlateauingLine* are implemented to identify these visual properties.
The second kind has to do with the angle of the slope (predicates which can be used are *ModerateSlope*, *SharpSlope, SubtleSlope* or *SlightSlope*).

4.1.5 Conclusions

For qualified evaluation of the outlined frameworks we use a simple metric that includes following elements: (1) querying, (2) supported diagrams, (3) model prototype and (4) format. Querying is very important feature for our work, since it allows making queries not only on diagrammatic information that is related to chart components, but also on data represented on the plot. Moreover, reuse of existing statistical information is possible (e.g. already queried results from a publication that is stored in a PDF file). Another important metric for framework characterization are supported charts. Ideally, the model should support any chart required in the project. The remaining two metrics are model format and prototype. The format denotes kind of the persistence model, i.e. whether the stored chart is e.g. in an XML, RDF, etc., whereby the prototype stands for a proof-of-concept provided in the publication, i.e. if a kind of a model prototype is available. Table 1 summarizes related work with respect to the introduced metrics.

	Querying	Diagrams	Prototype	Format
iGraph-Lite [2]	explicit framework algorithm for each query type	user-defined	not available anymore	XML
XML-SVG [3]	explicit parser function for each query	bar, line, pie only	not available	XML
XBrain [4]	custom XML-based query language	user-defined	available	XML
SGO [5]	SPARQL	all statistical; scheme is extensible	ontology available; chart examples available	RDF/OWL

Table 1: Evaluation of graph ontologies

Currently, to our knowledge, the Statistical Graph Ontology (SGO) and its successor, the SemanticScience Integrated Ontology (SIO), are the only existing approaches for describing statistical charts in RDF. The approach has been applied to biomedical knowledge to model a broad range of heterogeneous data and, fortunately, it offers a rich set of examples from this domain, which can help to progress more effectively with our project. Other XML-based approaches are able to describe chart models, but their common disadvantage is the inability to effectively query the existing data. Therefore, to realize the queries for such models, their structure has to be known beforehand.

It is possible to use the formal language, like XML, for the semantic description of charts. However, this standard offers no way of knowing simple facts such that both the x-axis and the y-axis of a chart are axes of the chart, or other relations such that if x-axis has entries that are in years that represents some time interval[17]. This is why we can use RDF for the semantic description of the charts. Such a representation can be easily queried and offers the ability to retrieve charts and included data on a standard way. For the querying of already existing charts including their data points (certain range of values, data series, chart configuration, etc.), the use of RDF models is more efficient [2].

To sum it up, the RDF-based approach has following advantages: (1) it allows to query existing charts enriched with data — therefore, charts are created only once; (2) it is possible to interlink chart data with each other and to represent them in a combined chart and (3) it

[17] http://dumontierlab.com

provides a common persistence model for representing char that can be used by various visualization technologies to visualize such data.

4.2 Semantic Descriptions for Visual Analytics Components

Based on our analysis of related work and existing standards and on the choice of using the RDF Data Cube Vocabulary for representing the underlying data sets, the Visual Analytics Vocabulary consists of the following parts:

1 Description of data points and data sources taken from the RDF Data Cube Vocabulary (http://purl.org/linked-data/cube#) as well as description of charts and their visual elements taken from the SemanticScience Integrated Ontology (SIO, http://semanticscience.org/ontology/sio.owl)
2 A mapping between RDF Data Cubes and the Statistical Graph Ontology as part of our own Visual Analytics Vocabulary
3 Extensions to SIO to define typical visual analytics properties, namely
    a that an Visual Analytics Dashboard consists of a set of graphs arranged in a specific way
    b that this set of graph is synchronized over certain data properties (thereby providing an multiple synchronized view on the data)
    c that user interactions change the state of the view port of the graphs and the displayed data points (filter data points, zoom level of a graph etc.)

The predecessor of the SemanticScience Integrated Ontology (SIO), namely the Statistical Graph Ontology (SGO), has been previously introduced. The RDF Data Cube Vocabulary will be introduced in more detail later.

The following three sub-chapters depict:

1 Our extensions to SIO in order to refine their definition of statistical graphs
2 Our RDF Data Cube / SIO mappings
3 Our extensions specific to the semantic description of a Visual Analytics Dashboard

The following abbreviations and namespaces will be used:

- qb (RDF Data Cube Vocabulary, http://purl.org/linked-data/cube#)
- sio (SemanticScience Integrated Ontology, http://semanticscience.org/ontology/sio.owl)
- va (Visual Analytics Vocabulary, http://code-research.eu/ontology/visual-analytics#)
- rdf, rdfs (Core RDF namespaces)
- dc (Dublin Core Metadata Initiative, http://purl.org/dc/terms/)

4.2.1 Extending the SIO Vocabulary

While the SIO ontology provides a sophisticated ground for describing statistical graphs (e.g. pie charts, bar charts), some key issues for our applications are missing. Our extensions are listed below. For every concept, we define the intensional properties through a short, verbal description and by listing the relevant relationships. Further, we define the extensional properties by giving corresponding examples. Note that the SIO namespace does not provide human readable concepts. In order to ease reading, we deviate from standard triple formats by giving the human readable name of a concept in round brackets (e.g. sio:SIO_000904 refers to the SIO concept while (Chart) depicts the human readable label).

The concept Chart is the graphical representation of numerical or qualitative data. The representation will be done by using symbols, like slices in a pie chart or lines in a line charts. A chart has the following properties

- *VisualChannel,*
  - `sio:SIO_000904 (Chart) sio:SIO_000028 (hasPart) va:VisualChannel`
- *Title,*
  - `sio:SIO_000904 (Chart) sio:SIO_000028 (hasPart) sio:SIO_000469 (GrapTitle)`
- *DataSeries* and
  - `sio:SIO_000904 (Chart) sio:SIO_000028 (hasPart) sio:SIO_000464 (DataSeries)`
- *Geometry*
  - `sio:SIO_000904 (Chart) sio:SIO_000028 (hasPart) va:Geometry`

The concept *VisualChannel* represents a visual dimension of a chart, like:

- *Axis,*
  - `sio:SIO_000450 (Axis) is_a  va:VisualChannel`
- *Color,*
  - `va:Color is_a va:VisualChannel`
- *Size,*
  - `va:Size is_a va:VisualChannel`
- *Symbol,*
  - `va:Symbol is_a va:VisualChannel`
- …

The concept Axis is a line segment that is part of a chart in which the position along the line corresponds to a numeric or categorical value. Axes can be categorized based on chart type, such as *CategoryAxis and ValueAxis* for Line, Bar, Pie and Geo Map charts etc. For instance, Line chart has for x-Axis items, which has as data type string (i.e. *CategoryAxis*) and for y-Axis integer (i.e. *ValueAxis*). For other charts there will be other divisions.
The concept CategoryAxis is an axis in which the position along the line is partitioned into categories

- `sio: SIO_000455` *`(CategoryAxis)`* `is_a  sio: SIO_000450` *`(Axis)`*

The concept ValueAxis is an axis in which the position along the line is partitioned into numeric values

- `sio: SIO_000458` *`(ValueAxis)`* `is_a  sio: SIO_000450` *`(Axis)`*

A chart's title is essential for the user because it indicates what the represented statistics semantically means.

- `sio:SIO_000904` *`(Chart)`* `sio:SIO_000028` *`(hasPart)`* `sio:SIO_000469` *`(GraphTitle)`*

The individual visual channels can also have a ***title**,* e.g.

- `va:VisualChannel sio:SIO_000028` *`(hasPart)`* `va:VisualChannelTitle`

Independent of chart axes and related data, two additional constructs are used to add more semantics to statistics: data points and data series. A data point is a n-tuple that consists of a single observation point, e.g. x and y points in a line graph.
The concept *DataSeries* are in turn used to collect ***DataPoints*** that belong to a single observation (e.g. a single line in a line chart). This allows to separate multiple statistics and to query them independently.

- `sio:SIO_000464` *`(DataSeries)`* `sio:SIO_000028` *`(hasPart)`* `sio:SIO_000465` *`(DataPoint)`*

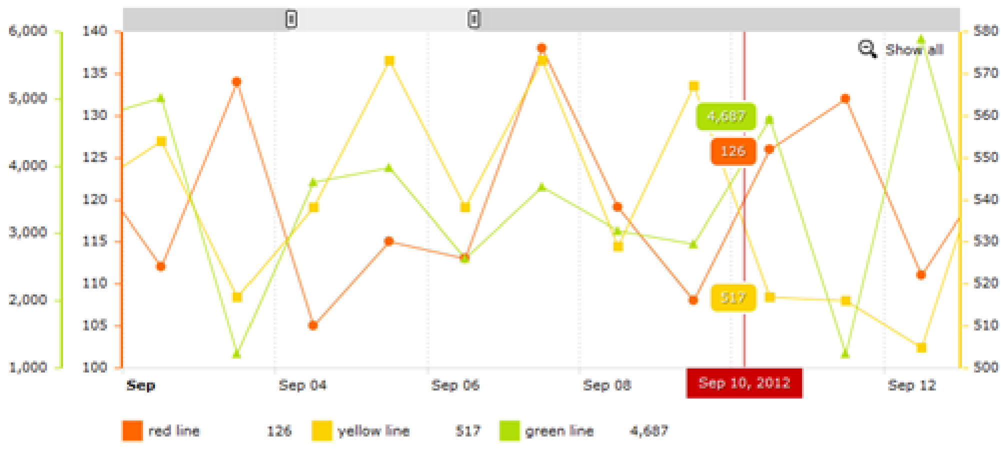


Figure 18: An Example for a Line Chart with multiple data series[18]

The concept *PlotArea* is used to express the geometrical boundary of a chart. It is defined by the *Geometry* element.

- `va:Geometry sio:SIO_000028 (hasPart) va:PlotArea`

The concept *PlotArea* has the following components:

- *Width*, which defines the width of the chart
  - `va:PlotArea sio:SIO_000028 (hasPart) sio:SIO_000042 (Width)`
- *Height*, which defines the height of the chart
  - `va:PlotArea sio:SIO_000028 (hasPart) sio:SIO_000040 (Height)`
- *BackgroundColor*, which defines the background color of the chart
  - `va:PlotArea sio:SIO_000028 (hasPart) va:BackgroundColor`
- *SeriesColor*, which defines the color of the *DataSeries* of the chart
  - `va:PlotArea sio:SIO_000028 (hasPart) va:SeriesColor`
- *BackgroundImage*, which defines the background image of the chart
  - `va:PlotArea sio:SIO_000028 (hasPart) va:BackgroundImage`

Other features can be defined when needed.

In addition, the following attributes of the chart can be defined for their semantic description:

- Data type of the visual channels. The data type identifies a type (i.e. real-values, integer, boolean, etc.) of data represented in chart virtual channels

[18] http://www.amcharts.com/javascript/line-chart-with-multiple-value-axes/

- `va: VisualChannel sio:SIO_000028` *(hasPart)* `va:DataType`

- Persistence of the visual channels: Persistence denotes whether a visual channel is permanently present in the chart and must be specified or it might be defined if needed, i.e. it is optional. In the current SIO model, it is not possible to express such a capability. For our domain, it is important to express the persistence of the visual channel, since some charts will expose optional dimensions. Therefore, following extension regarding the persistence model of axes is proposed:

1 – Fix	
0-* -> none until infinity many:	UnboundPersistence
1-* -> at least one :	MinOnePersistence
0-1 -> optional (none or one):	OptionalPersistence

- `va: VisualChannel sio:SIO_000028` *(hasPart)* `va:Persistence`

It is important to mention that the chart model above lies in a generic form that can be refined based on chart type which has to be developed. Therefore, such a generic form is applicable to any type of the statistical graph.

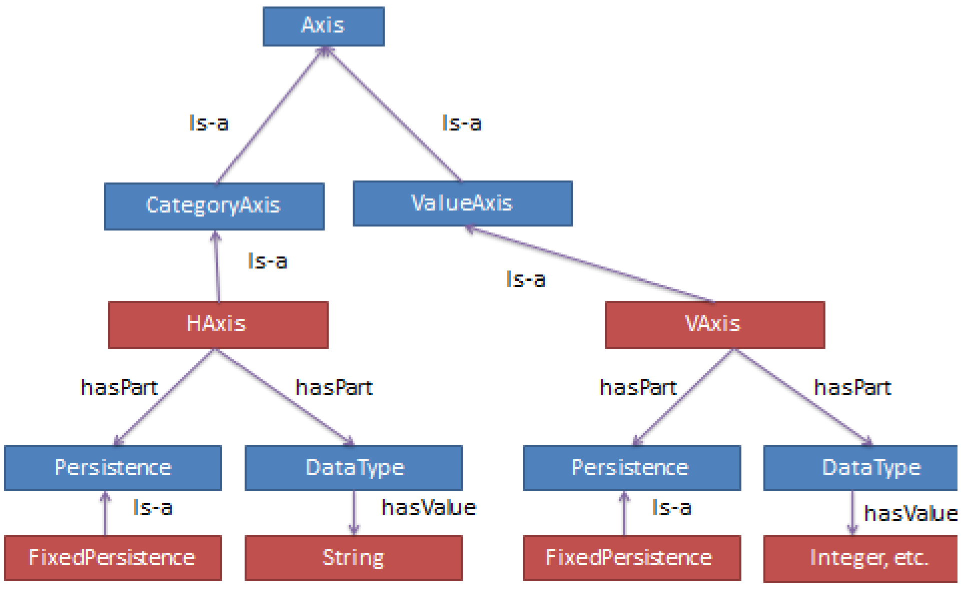


Figure 19: Concrete Description of the axes of the Bar Chart

Definitions of general classes for the Visual Analytics Vocabulary

The general classes, which we have defined for our visual analytics vocabulary, can be obtained from the following table.

Class	Description
va:VisualChannel	Represents a visual dimension of a chart.
va:Geometry	Represents the plot area information of a chart.
va:Color	Represents a concrete dimension of a chart.
va:Size	Represents a concrete dimension of a chart.
va:Symbol	Represents a concrete dimension of a chart.
va:VisualChannelTitle	Represent the title of a visual channel of a chart.
va:PlotArea	Is used to express the geometrical boundary of a chart.
va:BackgroundColor	Represents the background color of the chart.
va:SeriesColor	Represents the color of the data series of the chart.
va:DataType	Data type identifies a type of data represented in chart's visual channel.
va:Persistence	Persistence denotes whether a visual channel is permanently present in the chart and must be specified or it might be defined if needed.
va:BackgroundImage	Represents the image of the background of a chart.

Table 2: Definition of the generic classes of the Visual Analytics Vocabulary

Definition of the general classes for Visual Analytics Vocabulary (OWL)

```
va:VisualChannel a rdfs:Class, owl:Class;
    rdfs:label "Visual channel"@en;
    rdfs:comment "Represents a visual dimension of a chart."@en;
    rdfs:isDefinedBy <http://code-research.eu/ontology/visual-analytics>;
    .

va:Geometry a rdfs:Class, owl:Class;
    rdfs:label "Geometry"@en;
    rdfs:comment "Represents the plot area information of a chart."@en;
    rdfs:isDefinedBy <http://code-research.eu/ontology/visual-analytics>;
    .

va:Color a rdfs:Class, owl:Class;
    rdfs:label "Color"@en;
    rdfs:comment "Represents a visual dimension of a chart."@en;
    rdfs:isDefinedBy <http://code-research.eu/ontology/visual-analytics>;
    .
```

```
va:Size a rdfs:Class, owl:Class;
    rdfs:label "Size"@en;
    rdfs:comment "Represents a visual dimension of a chart."@en;
    rdfs:isDefinedBy <http://code-research.eu/ontology/visual-analytics>;
    .

va:Symbol a rdfs:Class, owl:Class;
    rdfs:label "Symbol"@en;
    rdfs:comment "Represents a visual dimension of a chart."@en;
    rdfs:isDefinedBy <http://code-research.eu/ontology/visual-analytics>;
    .

va:VisualChannelTitle a rdfs:Class, owl:Class;
    rdfs:label "Visual Channel Title"@en;
    rdfs:comment "Represent the title of a visual channel of a chart."@en;
    rdfs:isDefinedBy <http://code-research.eu/ontology/visual-analytics>;
    .

va:PlotArea a rdfs:Class, owl:Class;
    rdfs:label "Plot area"@en;
    rdfs:comment "Is used to express the geometrical boundary of a chart."@en;
    rdfs:isDefinedBy <http://code-research.eu/ontology/visual-analytics>;
    .

va:BackgroundColor a rdfs:Class, owl:Class;
    rdfs:label "Background color"@en;
    rdfs:comment "Represents the background color of the chart."@en;
    rdfs:isDefinedBy <http://code-research.eu/ontology/visual-analytics>;
    .

va:SeriesColor a rdfs:Class, owl:Class;
    rdfs:label "Series color"@en;
    rdfs:comment "Represents the color of the data series of the chart."@en;
    rdfs:isDefinedBy <http://code-research.eu/ontology/visual-analytics>;
    .

va:DataType a rdfs:Class, owl:Class;
    rdfs:label "Data type"@en;
    rdfs:comment "Data type identifies a type of data represented in chart’s visual
channel."@en;
    rdfs:isDefinedBy <http://code-research.eu/ontology/visual-analytics>;
    .

va:Persistence a rdfs:Class, owl:Class;
    rdfs:label "Persistence"@en;
    rdfs:comment "Persistence denotes whether a visual channel is permanently
present in the chart and must be specified or it might be defined if needed."@en;
    rdfs:isDefinedBy <http://code-research.eu/ontology/visual-analytics>;
    .
```

```
va:BackgroundImage a rdfs:Class, owl:Class;
    rdfs:label "Background image"@en;
    rdfs:comment " Represents the image of the background of a chart."@en;
    rdfs:isDefinedBy <http://code-research.eu/ontology/visual-analytics>;
    .
```

Definitions of general properties for the Visual Analytics Vocabulary

The general properties, which we have defined for our visual analytics vocabulary, can be obtained from the following table.

Property	Domain → Range	Cardinality	Comment
va:supportsType	va:VisualChannel → qb:DimensionProperty	one or more	Defines which kind of Cube Dimensions this chart supports.
va:unit	va:VisualChannel → rdfs:Resource	exactly one	Defines the unit measure of a visual channel.

Table 3: Property Definitons of the Visual Analytics Vocabulary

Definitions of general properties for the Visual Analytics Vocabulary (OWL)

```
va:supportsType a rdf:Property, owl:ObjectProperty;
    rdfs:label "supports type"@en;
    rdfs:comment "Defines which kind of Cube Dimensions this chart supports"@en;
    rdfs:domain sio:SIO_000904;
    rdfs:range  qb:DimensionProperty;
    rdfs:isDefinedBy <http://code-research.eu/ontology/visual-analytics>;
    .

owl:Restriction owl:onProperty va:supportsType;
    owl:minCardinality 1;
    .

va:unit a rdf:Property, owl:ObjectProperty;
    rdfs:label "unit"@en;
    rdfs:comment "Defines the unit measure of a visual channel."@en;
    rdfs:domain va:VisualChannel;
    rdfs:range  rdfs:Resource;
    rdfs:isDefinedBy <http://code-research.eu/ontology/visual-analytics>;
    .
owl:Restriction owl:onProperty va:unit;
    owl:minCardinality 1;
    owl:maxCardinality 1;
  .
```

Classes defined by the SIO

Some relevant classes, which have already been defined by the SIO, can be obtained from the following table.

Class	Description
sio:SIO_000904 (Chart)	A chart is a figure that displays the relationship among tabular numeric data, functions or some kinds of qualitative structures.
sio:SIO_000469 (GraphTitle)	A graph title is a title that describes a graph.
sio:SIO_000464 (DataSeries)	A data series is a data set composed of related values displayed in a statistical graph.
sio:SIO_000450 (Axis)	An axis is a line segment that is part of a chart in which the position along the line corresponds to a numeric or categorical value
sio:SIO_000455 (CategoryAxis)	The horizontal axis of the chart.
sio:SIO_000458 (ValueAxis)	A value axis is an axis in which the position along the line is partitioned into numeric values.
sio:SIO_000465 (DataPoint)	A data point is a point that which corresponds to the projection of the values of measurement data against the visual channels of a statistical graph.
sio:SIO_000042 (Width)	Represents the width of the chart.
sio:SIO_000040 (Height)	Represents the height of the chart.

Table 4: Classes defined by the SemanticScience Integrated Ontology

Classes defined by the SIO (OWL)

For definition of SIO classes in OWL can be obtained from http://semanticscience.org/ontology/sio.owl

Properties defined by the SIO

Some relevant properties, which have already been defined by the SIO, can be obtained from the following table.

Property	Domain → Range	Cardinality	Comment
sio:SIO_000028 (hasPart)	sio:SIO_000904 (Chart) → va:VisualChannel sio:SIO_000904 (Chart) → sio:SIO_000469 (Graph Title) sio:SIO_000904 (Chart) → sio:SIO_000464 (DataSeries) sio:SIO_000904 (Chart) → va:Geometry sio:SIO_000455 (CategoryAxis)→ va:CategoryAxisTitle sio:SIO_000458 (ValueAxis)→ va:ValueAxisTitle sio:SIO_000464 (DataSeries)→ sio:SIO_000465 (DataPoint) va:Geometry → va:PlotArea va:PlotArea → sio.SIO_000042 (Width) va:PlotArea → sio.SIO_000040 (Height) va:PlotArea → va:BackgroundColor va:PlotArea → va:SeriesColor va:PlotArea → va:BackgroundImage	exactly one	Is a transitive, reflexive and antisymmetric relation between a whole and itself or a whole and its part.

Table 5: Property Definition of the SemanticScience Integrated Ontology

Properties defined by the SIO (OWL)

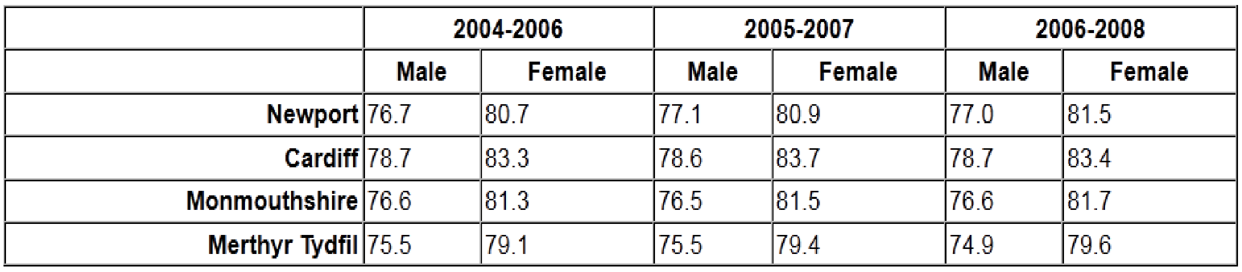
The definitions of the SIO object properties can be obtained from http://semanticscience.org/ontology/sio.owl

4.2.2 Linking RDF Data Cubes and Charts

After extending the SemanticScience Integrated Ontology (SIO), we link them with the RDF Data Cube Vocabulary (QB).

The RDF Data Cube Vocabulary is a W3C Standard and is developed to translate statistical data into a format in which the data can be easily linked and combined with related information.
This data set represents a collection of observations representing some statistics. It consist of dimensions that identify the observation (e.g. time, area, sex), measures related to concrete values and attributes that add semantics to them (e.g. life expediency). It is possible to model any kind of statistical data with these components. The statistical data set is a cube organized as a set of dimensions, attributes and measures and the Data Cube vocabulary such data as RDF properties: *qb:DimensionProperty*, *qb:AttributeProperty* and *qb:MeasurePropety* derive from generic observation class *qb:ComponentProperty*. This cube model is compatible with the SDMX (Statistical Data and Metadata eXchange), an ISO standard for exchanging and sharing statistical data and metadata among organizations. One of the major components of this model is Content-Oriented Guidelines (COGs) to ensure the interoperability between statistical data sets.

For instance, in the table below, *time period*, *region* and *sex* are the dimensions, *life expectancy* the measure, and *years* is the attribute (the unit of the measures).



	2004-2006		2005-2007		2006-2008	
	Male	Female	Male	Female	Male	Female
Newport	76.7	80.7	77.1	80.9	77.0	81.5
Cardiff	78.7	83.3	78.6	83.7	78.7	83.4
Monmouthshire	76.6	81.3	76.5	81.5	76.6	81.7
Merthyr Tydfil	75.5	79.1	75.5	79.4	74.9	79.6

Table 6: Data set extracted from StateWales[19]

[19] http://www.w3.org/TR/vocab-data-cube/

Some of the components of the example table above will be defined as follows:

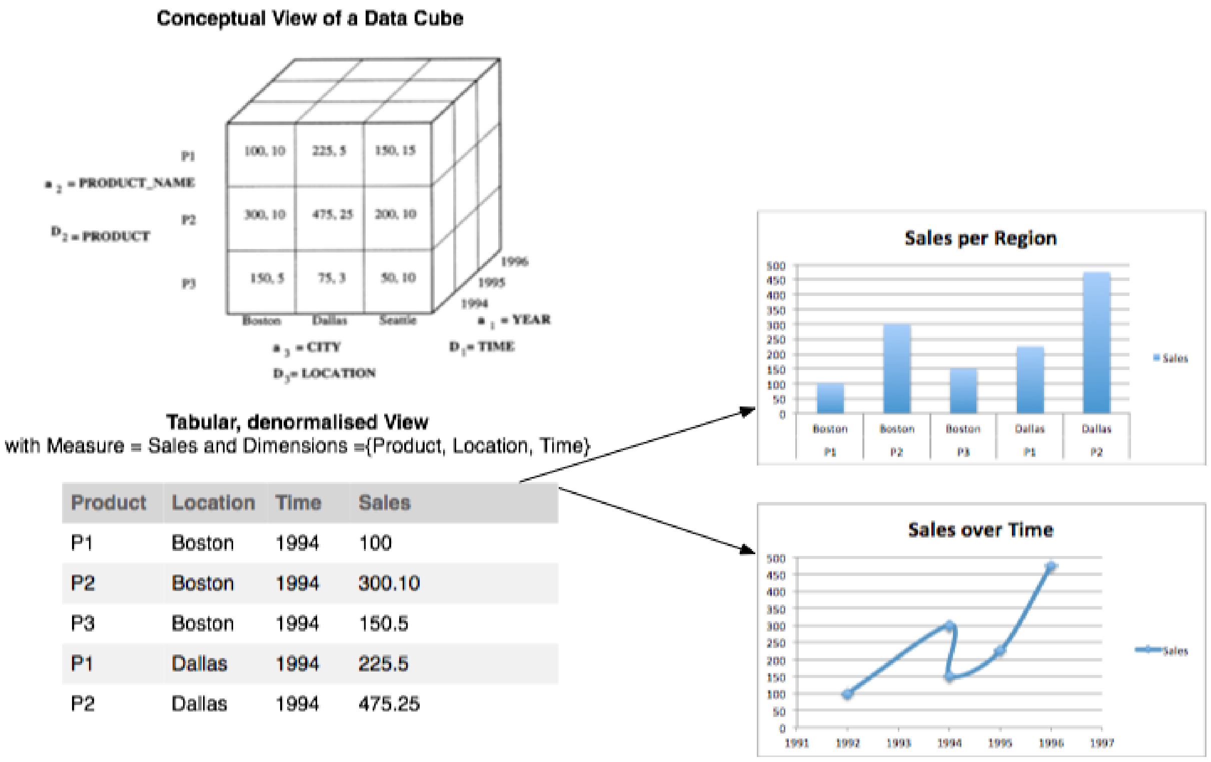
- Time: For time there is a predefined concept and corresponding property in the SDMX-COG, `sdmx-dimension:refPeriod`

```
eg:refPeriod  a rdf:Property, qb:DimensionProperty;
    rdfs:label "reference period"@en;
    rdfs:subPropertyOf sdmx-dimension:refPeriod;
    rdfs:range interval:Interval;
    qb:concept sdmx-concept:refPeriod .
```

- Measure: For measure we can use the default `sdmx-measure:obsValue`

```
eg:lifeExpectancy  a rdf:Property, qb:MeasureProperty;
    rdfs:label "life expectancy"@en;
    rdfs:subPropertyOf sdmx-measure:obsValue;
    rdfs:range xsd:decimal .
```

Conceptually, the source of a data cube is a table, where columns form either dimensions or measures. For the visualization of these components of the Data cube, there are many charts integrated on the visualization framework. The following figure depicts a simple example and how it converts into charts.



Product	Location	Time	Sales
P1	Boston	1994	100
P2	Boston	1994	300.10
P3	Boston	1994	150.5
P1	Dallas	1994	225.5
P2	Dallas	1994	475.25

Figure 20: Example of a data cube in its conceptually view (top-left), its de-normalised table form (bottom-left) and two possible visualisations on the right. Note that time can be used either as dimension or as measure.

The qb-Vocabulary allows to specify which column belongs to a dimension (`qb:DimensionProperty`) and which column belongs to a measure (`qb:MeasureProperty`) (via as part of the `qb:ComponentSpecification` (as part of the data structure definition)).
On the other hand, the SIO vocabulary refer to an axis of a chart (which can be thought of dimensions) and to data points as value along those axis. Figure 21 shows the Data Cube Vocabulary.

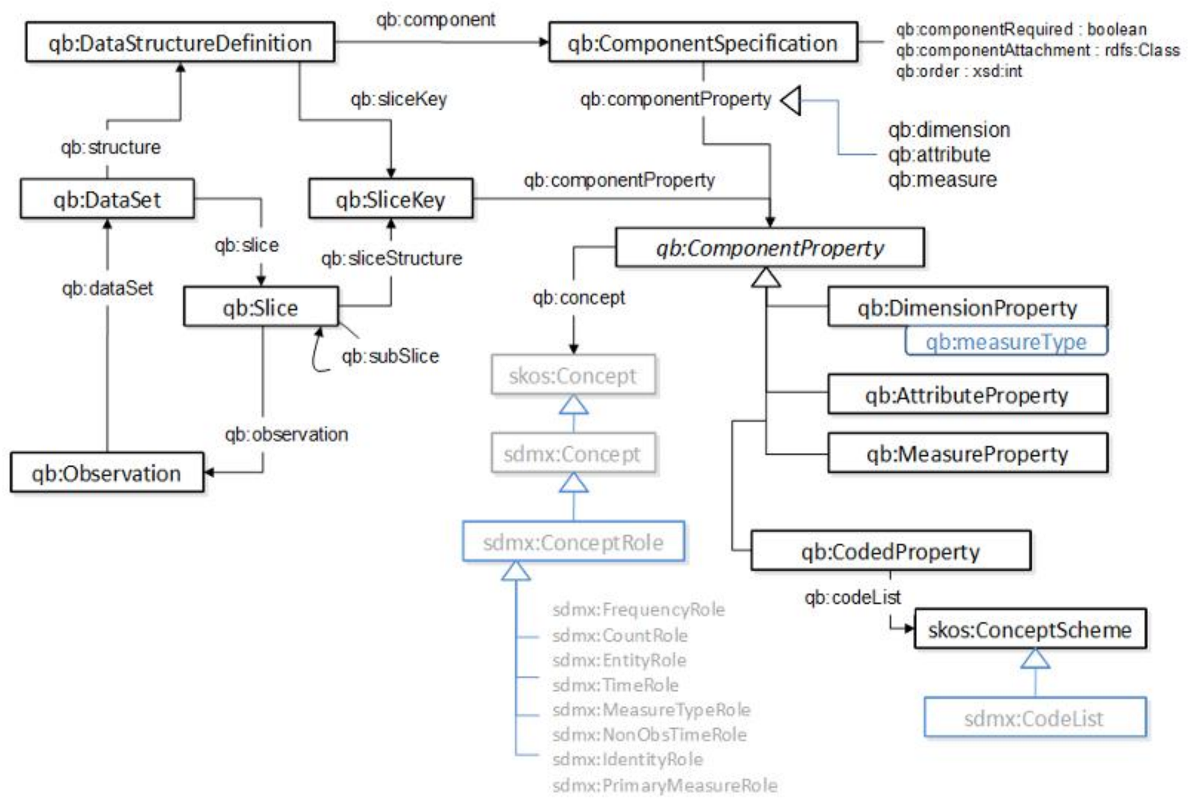


Figure 21: Diagram of the Data Cube Vocabulary[20]

For introducing the mapping between those two vocabularies, **we assume that a data cube consists of *`1-d dimensions`* and *`1-m measures.`***

The definition of a data cube is specified via the `qb:DataStructureDefinition` while instantiated data sets are defined via `qb:Observation`, `qb:DataSet` and / or `qb:Slices`.
Given such a data cube, the following observations can be made (we will give a formal definition below):

- The axis of a chart refers to exactly one `qb:ComponentProperty`, which can be either a dimension or a measure

[20] http://www.w3.org/TR/vocab-data-cube/

- ○ Since qb:DimensionProperty and `qb:MeasureProperty` can be either nominal, ordinal or rational scaled (since the type can be from an arbitrary vocabulary), we cannot make a more specific assignment. Hence, mappings on the different possible value ranges (which are defined via the `qb:concept` property of the `qb:ComponentProperty`) have to be introduced on demand and as needed. The user interface will take care of ease mappings by providing dynamic recommendations to the user.
- ○ Metadata for an axis (e.g. axis title, categorization axis, description of the unit scale etc.) can be obtained from the properties of a `qb:ComponentProperty`, namely
  - ■ Title of an axis is either the `rdfs:label`, or the `rdfs:label` of one of the associated concepts (range of the `qb:concept` property). There is no need for an explicit mapping, since the chart axis and the cube dimension can already be linked via `va:representsDimension`
  - ■ Unit measures may also be obtained from the cube dimension metadata. Again, there is no need for a separate mapping, since the dimensions are already mapped to the chart elements.
  - ■ The user will be able to select the corresponding label, which has to be stored afterwards.
- ● Metadata of the chart comes from the data cube structure description and potential sub-slices.
- ● Data and data points in the SIO vocabulary are provided by `qb:Observations` from the cube vocabulary. Note that we do not model constraints on types, since the cube type system is open.

Property Definitions for Mappping from Chart to RDF Data Cube

Property	Domain → Range	Cardinality	Comment
va:representsDimension	va:VisualChannel → qb:DimensionProperty	exactly one	The mapping from a visual channel of a chart to a Cube Dimension.
va:representsObservati on	sio:SIO_000465 (DataPoint) → qb:Observation	exactly one	The mapping from a data point to a Cube Observation. Observations are automatically included in data series.
va:representsDatasets	sio:SIO_000464 (DataSeries) → qb:DataSet	exactly one	The mapping from a data series to a Cube data set.
va:displaysMeasure	va:VisualChannel → qb:MeasureProperty	one or more	Associates a visual channel with Cube Measures.

Table 7: Extension properties for mapping between RDF Cube and SIO vocabularies

Property Definitions for Mapping from Chart to RDF Data Cube (OWL)

```
va:representsDimension a rdf:Property, owl:ObjectProperty;
    rdfs:label "represents dimension"@en;
    rdfs:comment "The mapping from a visual channel  of a chart to a Cube
Dimension."@en;
    rdfs:domain sio:SIO_000450;
    rdfs:range  qb:DimensionProperty;
    rdfs:isDefinedBy <http://code-research.eu/ontology/visual-analytics>;
    .

owl:Restriction owl:onProperty va:representsAxis;
    owl:minCardinality 1;
    owl:maxCardinality 1;
    .

va:representsObservation a rdf:Property, owl:ObjectProperty;
    rdfs:label "represents observation"@en;
    rdfs:comment "The mapping from a data point to a Cube  Observation."@en;
    rdfs:domain SIO_000465 ;
    rdfs:range  qb:Observation;
    rdfs:isDefinedBy <http://code-research.eu/ontology/visual-analytics>;
    .

owl:Restriction owl:onProperty va:representsAxis;
    owl:minCardinality 1;
    owl:maxCardinality 1;
    .

va:representsDatasets a rdf:Property, owl:ObjectProperty;
    rdfs:label "represents observation"@en;
    rdfs:comment "The mapping from a data series to a Cube data set to."@en;
    rdfs:domain SIO_000464  ;
    rdfs:range  qb:DataSet;
    rdfs:isDefinedBy <http://code-research.eu/ontology/visual-analytics>;
    .

owl:Restriction owl:onProperty va:representsAxis;
    owl:minCardinality 1;
    owl:maxCardinality 1;
    .

va:displayMeasure a rdf:Property, owl:ObjectProperty;
    rdfs:label "represents observation"@en;
    rdfs:comment "Associates a chart with Cube Measures."@en;
    rdfs:domain SIO_000904   ;
    rdfs:range  qb:MeasureProperty;
    rdfs:isDefinedBy <http://code-research.eu/ontology/visual-analytics>;
    .

owl:Restriction owl:onProperty va:representsAxis;
    owl:minCardinality 1;
    .
```

<u>Example for a mapping from a RDF Cube to a line chart</u>

This example illustrates a population distribution of *Cardiff_North* and *Cardiff_West* for period 2008-2009 (see Figure 22). In constructed RDF Cube model, period and area are represented as dimensions, and population is measure.

Definition of the data structure of the RDF Data Cube

```
va:Total_population_DSD  rdf:type  cube:DataStructureDefinition,  :NamedIndividual  ;

    rdfs:label "Total Population DSD"@en ;
    rdfs:comment "Structure of Total population"@en ;
    cube:slicekey <http://statistics.data.wales.gov.uk/slice/population-region>;
    cube:component [ cube:measure va:population ],
                   [ cube:dimension va:region ],
                   [ cube:dimension cube:refPeriod ].
```

Definition of the data set of the RDF Data Cube

```
va:Total_Population rdf:type cube:DataSet, :NamedIndividual;
    rdfs:label "Total population"@en;
    dc:date "2011-03-07"^^xsd:dateTime;
    rdfs:comment "Total Population";
    cube:struct va:Total_population_DSD;
    dc:publisher <http://www.wales.gov.uk/>.
```

Having only such data in the RDF Data Cube, it is first important to identify which chart is suitable to visualize statistics. The abstract class chart has the property *supportedTypes* that indicates which *ComponentProperty* of the Cube (i.e. dimension or measure) can be visualized. Based on this information, it is possible to identify candidate charts (here is assumed that the user has already selected columns for visualization). In this example we have instantiated the line chart (as mentioned, this can be performed automatically).

Definition of the instance of a line chart

```
va:linechart1 rdf:type va:LineChart, :NamedIndividual;
    ontology:hasPart va:geometry1, va:series1,
                     va:x-axis, va:y-axis.
```

After the definitions of the data sets and the data structure of the RDF Data Cube and the initialization of the line chart, the mapping can be started.

Mapping of a data series of the line chart to the date set of the RDF Data Cube

```
va:series1 rdf:type ontology:Series, :NamedIndividual;
   va:representsDatasets va:Total_Population;
         ontology:hasPart va:datapointx1y1,
                          va:datapointx2y2,
                          va:datapointx3y3,
                          va:datapointx4y4.


_:NAWAC28 rdf:type cube:Observation, :NamedIndividual;
   va:population "88059"^^xsd:long;
   va:region va:Cardiff_Nord;
   va:representsDatasets va:Total_Population;
   cube:dataset va:Total_Population;
   cube:refPeriod <http://reference.data.gov.uk/id/year/2008>.


_:NAWAC63 rdf:type cube:Observation, :NamedIndividual;
   va:population "87511"^^xsd:long;
   va:region va:Cardiff_West;
   va:representsDatasets va:Total_Population;
   cube:dataset visual-analytics:Total_Population;
   cube:refPeriod <http://reference.data.gov.uk/id/year/2008> .

_:NAWAC28 rdf:type cube:Observation, :NamedIndividual;
   va:population "89333"^^xsd:long;
   va:region va:Cardiff_Nord;
   va:representsDatasets va:Total_Population;
   cube:dataset visual-analytics:Total_Population;
   cube:refPeriod <http://reference.data.gov.uk/id/year/2009>.

_:NAWAC63 rdf:type cube:Observation, :NamedIndividual;
   va:population "88388"^^xsd:long;
   va:region va:Cardiff_West;
   va:representsDatasets va:Total_Population;
   cube:dataset visual-analytics:Total_Population;
   cube:refPeriod <http://reference.data.gov.uk/id/year/2009>.
```

Mapping from a data point of the line chart to a RDF Data Cube Observation

```
_:DataPointx1y1 rdf:type ontology:DataPoint, :NamedIndividual;
    va:representsObservation va:NAWAC28.

_:DataPointx2y2 rdf:type ontology:DataPoint, :NamedIndividual;
    va:representsObservation va:NAWAC63.

_:DataPointx3y3 rdf:type ontology:DataPoint, :NamedIndividual;
    va:representsObservation va:NAWAC28.

_:DataPointx4y4 rdf:type ontology:DataPoint, :NamedIndividual;
    va:representsObservation va:NAWAC63.
```

As shown in extended SIO model, dimensions and measures are mapped to the visual channels. Moreover, additional dimensions complicate the situation since they produce multiple variants of visualizing the same observation set. Therefore, the user has to assist this process. However, it is also important to define the boundary where this should happen. In this example the axes of the chart are mapped to dimensions and measure with following schema: x-axis is mapped to either period property or area property, and y-axis is mapped to population. This map process cannot be automated (at least not always). However, there is also an option to automate mapping to axes by reusing previously stored mapping definitions. So, for instance, for the next time the property population would be automatically detected and mapping could be suggested.

In the worst case, there is no cached mapping and the user has to define it herself. The aim is to allow the user to navigate through data structures of involved dimensions and measures, on the one side, and data structures of charts on the other side, to find the proper mapping. Both vocabularies have to comply on primitive data types, regardless of how the Cube components are constructed.

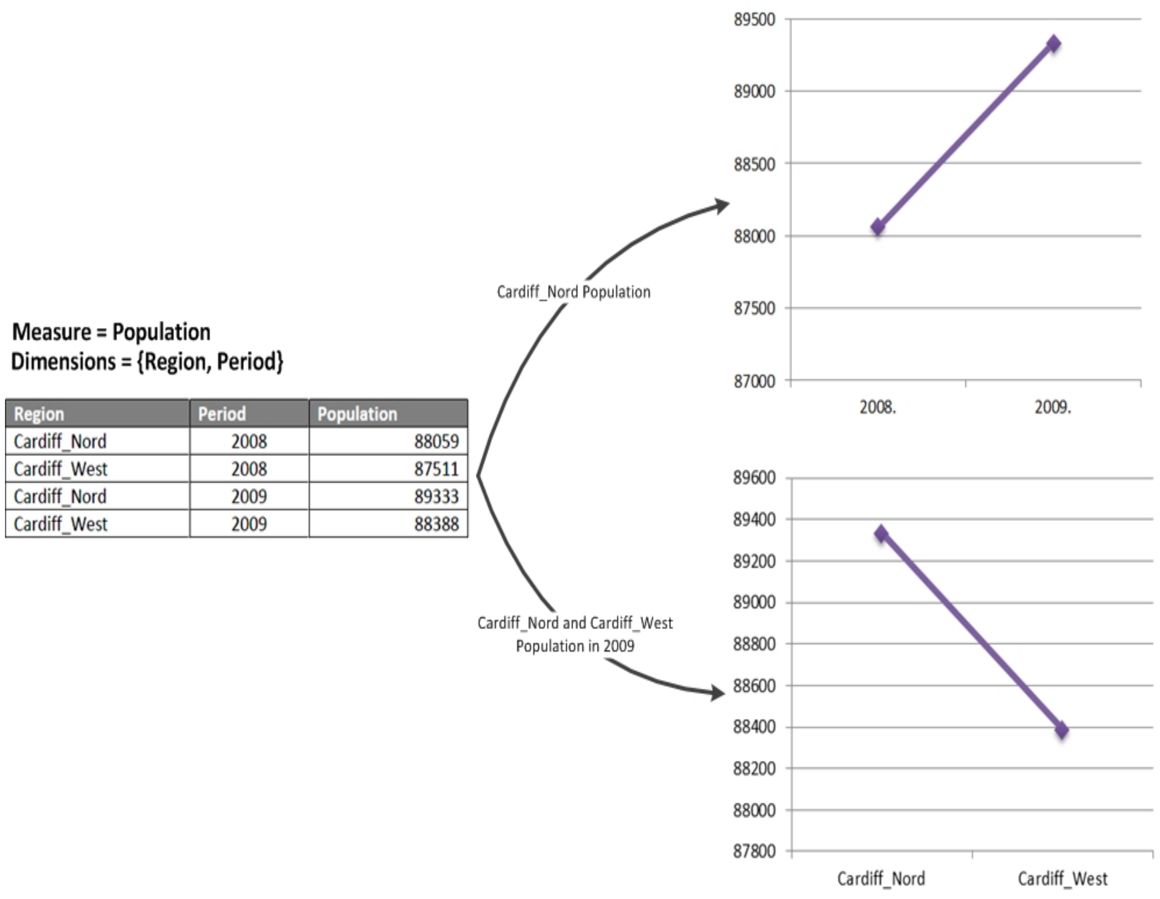
Mapping from a visual channel (in our example the classes x-axis and y-axis are visual channels) to a cube dimension

```
va:x-axis rdf:type ontology:CategoryAxis, :NamedIndividual;
    va:representsDimension cube:region.
```

Mapping from a visual channel (in our example the classes x-axis and y-axis are visual channels) to a cube measure

```
va:y-axis rdf:type ontology:ValueAxis, :NamedIndividual;
    va:displaysMeasures va:population.
```

From the proposed extended model above, we are able to access any Data Cube from the chart model. This standard interface to the Data Cube will also be a foundation for seamless integration of visualization generators.



Region	Period	Population
Cardiff_Nord	2008	88059
Cardiff_West	2008	87511
Cardiff_Nord	2009	89333
Cardiff_West	2009	88388

Figure 22: Mapping from a RDF Cube to a Line Chart

4.2.3 Defining the Visual Analytics Dashboard

Given that we now can represent charts and back them with data, we also want to group multiple charts into one single Visual Analytics (VA) Dashboard. The following observations can be made:

- Every VA Dashboard has a title and a description. The description might be used to communicate the interpretation of the visualizations.
  - `va:Dashboard  dc:title        rdfs:Literal`
  - `va:Dashboard  dc:description  rdfs:Literal`
- Every VA Dashboard consists of one or more charts
  - `va:Dashboard  sio:SIO_000028` *(has part)* `sio:SIO_000904` *(Chart)*
- Every Chart in the VA Dashboard has size and position properties as well as a description.
  - `sio:SIO_000904` *(Chart)* `va:chartSize      rdfs:Literal`
  - `sio:SIO_000904` *(Chart)* `va:chartPosition  rdfs:Literal`
  - `sio:SIO_000904` *(Chart)* `dc:description    rdfs:Literal`

SEVENTH FRAMEWORK PROGRAMME

- Every VA Dashboard operates on a single data cube. This assumption is valid, since data cubes cannot be merged automatically. (Consider a cube representing different data points and sharing only one common dimension like location — the data points remain completely incomparable.)
  - `va:Dashboard  va:visualizes  qb:DataSet`

- Only a subset of a data cube, a `qb:Slice`, might be visualised. Slices could be the result of
  a. data warehousing operations like roll-up or drill-down on concept hierarchies defined by `qb:Concept` (and sub-classes therefrom).
  b. filter operations like filtering data points within a certain range

  As a result of these operations, the data set in all visualisation changes.
  - `va:Dashboard  va:visualizes  qb:Slice`

- Axis of different charts that share qb:DimensionProperties may be coordinated. Coordination means that when the viewport of an axis changes due to a user interaction such as selection or zooming, all viewports of charts having these axes change too. This empowers the user to interactively analyze complex data over different dimensions using multiple charts.
  - `sio:SIO_000450` *`(Axis)`* `va:representsDimension qb:DimensionProperty`

- More classes and properties will be added as the need arises during development of the prototype. This is expected in year two of the CODE project.

Class Definitions

Class	Comment
va:Dashboard	Represents a collection of one or more charts that visualize the same Data Cube or Data Slice.

Table 8: Definition of the generic classes for the Visual Analytics Dashboard

Class Definitions (OWL)

```
va:Dashboard a rdfs:Class, owl:Class;
    rdfs:label "Dashboard"@en;
    rdfs:comment "Represents a collection of one or more charts that visualize the
same Data Cube or Data Slice."@en;
    rdfs:isDefinedBy <http://code-research.eu/ontology/visual-analytics>;
    .
```

Property Definitions

Property	Domain → Range	Cardinality	Comment
va:visualizes	va:Application → {qb:DataSet, qb:Slice}	exactly one	Links a Visual Analytics Dashboard with a Data Cube or Data Slice.

Table 9: Definition of the properties for the Visual Analytics Dashboard

Property Definitions (OWL)

```
va:visualizes a rdf:Property, owl:ObjectProperty;
    rdfs:label "visualizes"@en;
    rdfs:comment "Links a Visual Analytics Dashboard with a Data Cube or Data
Slice"@en;
    rdfs:domain  va:Dashboard;
    rdfs:range qb:DataSet, qb:Slice;
    rdfs:isDefinedBy <http://code-research.eu/ontology/visual-analytics>;
    .

owl:Restriction owl:onProperty va:visualizes;
    owl:minCardinality 1;
    owl:maxCardinality 1;
    .
```

5 Conclusions

In this deliverable, we presented a proposal for semantically describing data and visualizations, mapping from one to the other, and semantically describing web-based visual analytics applications that make use of these semantic descriptions. We also provide a first working draft of the resulting Visual Analytics Vocabulary in the form of an OWL ontology (see Appendix A).

The prototypical implementation of a web-based visual analytics application based on these semantic descriptions is underway. However, due to the early development stage, the Visual Analytics Vocabulary will evolve over time. The version presented in this deliverable should illustrate the core principles. The current, up-to-date version of the Visual Analytics Vocabulary will be available at http://code-research.eu/ontology/visual-analytics

6 Bibliography

[1] Lehmann, J. and Bühmann, L., 2011: Springer Berlin / Heidelberg. http://jens-lehmann.org/files/2011/autosparql_eswc.pdf

[2] Ferres Leo, Petro Verkhogliad and Gitte Lindgaard, 2007: Improving Accessibility to Statistical Graphs: The iGraph-Lite System

[3] Wollowski Michael, 2004: Search and inference with Diagrams. http://www.rose-hulman.edu/~wollowsk/Multi-modal-final-submission.pdf

[4] Bales Nathan, Brinkley James, Lee E.Sally, Mathur Shobhit, Re Christopher, and Suciu Dan, 2005: A Framework for XML-based Integration of Data, Visualization and Analysis in a Biomedical Domain.
http://homes.cs.washington.edu/~suciu/file16_paper.pdf

[5] Dumontier Michel, Ferres Leo and Villanueva-Rosales Natalia, 2010: Modeling and querying graphical representations of statistical data.

[6] Arasu Arvind, Babu Shivnath and Widow Jennifer, 2003: The CQL Continuous Query Language. Semantic Foundations and Query Execution.
http://ilpubs.stanford.edu:8090/758/1/2003-67.pdf

[7] Ferres Leo, Dumontier Michel, Villanueva-Rosales Natalia, EDOC Conference Workshop, 2007: EDOC '07. Eleventh International IEEE: Semantic Query Answering with Time-Series Graphs

[8] Cadag Eithon, Tarcy-Hornoch Peter, Journal of Biomedical Informatics, 2009: Supporting retrieval of diverse biomedical data using evidence-aware queries

Appendix A: Visual Analytics Vocabulary (Working Draft)

The current version of the Visual Analytics Vocabulary is available at http://code-research.eu/ontology/visual-analytics

```
@prefix rdf:      <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix rdfs:     <http://www.w3.org/2000/01/rdf-schema#> .
@prefix owl:      <http://www.w3.org/2002/07/owl#> .
@prefix xsd:      <http://www.w3.org/2001/XMLSchema#> .
@prefix dc:       <http://purl.org/dc/terms/> .
@prefix va:       <http://code-research.eu/ontology/visual-analytics#>
@prefix qb:       <http://purl.org/linked-data/cube#> .
@prefix sio:      <http://semanticscience.org/ontology/sio.owl#> .

<http://code-research.eu/ontology/visual-analytics>
    a owl:Ontology;
    owl:versionInfo "0.1";
    rdfs:label "The Visual Analytics Vocabulary";
    rdfs:comment "This vocabulary allows the semantic description of visual
analytics applications. It is based on the RDF Data Cube Vocabulary and the
Semanticscience Integrated Ontology.";
    dcterms:created "2012-10-31"^^xsd:date;
    dcterms:modified "2010-10-31"^^xsd:date;
    .


# --- General Classes ---------------------------

va:VisualChannel a rdfs:Class, owl:Class;
    rdfs:label "Visual channel"@en;
    rdfs:comment "Represents a visual dimension of a chart."@en;
    rdfs:isDefinedBy <http://code-research.eu/ontology/visual-analytics>;
    .

va:Geometry a rdfs:Class, owl:Class;
    rdfs:label "Geometry"@en;
    rdfs:comment "Represents the plot area information of a chart."@en;
    rdfs:isDefinedBy <http://code-research.eu/ontology/visual-analytics>;
    .

va:Color a rdfs:Class, owl:Class;
    rdfs:label "Color"@en;
    rdfs:comment "Represents a visual dimension of a chart."@en;
    rdfs:isDefinedBy <http://code-research.eu/ontology/visual-analytics>;
    .

va:Size a rdfs:Class, owl:Class;
    rdfs:label "Size"@en;
    rdfs:comment "Represents a visual dimension of a chart."@en;
    rdfs:isDefinedBy <http://code-research.eu/ontology/visual-analytics>;
    .

va:Symbol a rdfs:Class, owl:Class;
    rdfs:label "Symbol"@en;
    rdfs:comment "Represents a visual dimension of a chart."@en;
    rdfs:isDefinedBy <http://code-research.eu/ontology/visual-analytics>;
    .

va:VisualChannelTitle a rdfs:Class, owl:Class;
    rdfs:label "Visual Channel Title"@en;
    rdfs:comment "Represent the title of a visual channel of a chart."@en;
    rdfs:isDefinedBy <http://code-research.eu/ontology/visual-analytics>;
    .

va:PlotArea a rdfs:Class, owl:Class;
    rdfs:label "Plot area"@en;
    rdfs:comment "Is used to express the geometrical boundary of a chart."@en;
    rdfs:isDefinedBy <http://code-research.eu/ontology/visual-analytics>;
    .

va:BackgroundColor a rdfs:Class, owl:Class;
    rdfs:label "Background color"@en;
    rdfs:comment "Represents the background color of the chart."@en;
    rdfs:isDefinedBy <http://code-research.eu/ontology/visual-analytics>;
    .

va:SeriesColor a rdfs:Class, owl:Class;
    rdfs:label "Series color"@en;
    rdfs:comment "Represents the color of the series of the chart."@en;
    rdfs:isDefinedBy <http://code-research.eu/ontology/visual-analytics>;
    .

va:DataType a rdfs:Class, owl:Class;
    rdfs:label "Data type"@en;
    rdfs:comment "Data type identifies a type of data represented in chart’s visual
channel."@en;
    rdfs:isDefinedBy <http://code-research.eu/ontology/visual-analytics>;
    .

va:Persistence a rdfs:Class, owl:Class;
    rdfs:label "Persistence"@en;
    rdfs:comment "Persistence denotes whether a visual channel is permanently
present in the chart and must be specified or it might be defined if needed."@en;
    rdfs:isDefinedBy <http://code-research.eu/ontology/visual-analytics>;
    .

va:BackgroundImage a rdfs:Class, owl:Class;
    rdfs:label "Background image"@en;
    rdfs:comment " Represents the image/texture for the background of a chart."@en;
    rdfs:isDefinedBy <http://code-research.eu/ontology/visual-analytics>;
    .
```

```
# --- General Properties ---------------------------

va:supportsType a rdf:Property, owl:ObjectProperty;
    rdfs:label "supports type"@en;
    rdfs:comment "Defines which kind of Cube Dimensions this chart supports"@en;
    rdfs:domain sio:SIO_000904;
    rdfs:range  qb:DimensionProperty;
    rdfs:isDefinedBy <http://code-research.eu/ontology/visual-analytics>;
    .

owl:Restriction owl:onProperty va:supportsType;
    owl:minCardinality 1;
    .

va:unit a rdf:Property, owl:ObjectProperty;
    rdfs:label "unit"@en;
    rdfs:comment "Defines the unit measure of a visual channel."@en;
    rdfs:domain va:VisualChannel;
    rdfs:range  rdfs:Resource;
    rdfs:isDefinedBy <http://code-research.eu/ontology/visual-analytics>;
    .

owl:Restriction owl:onProperty va:unit;
    owl:minCardinality 1;
    owl:maxCardinality 1;
  .


# --- Mapping-Specific Properties ---------------------------

va:representsDimension a rdf:Property, owl:ObjectProperty;
    rdfs:label "represents dimension"@en;
    rdfs:comment "The mapping from a chart’s visual channel to a Cube 
Dimension."@en;
    rdfs:domain sio:SIO_000450;
    rdfs:range  qb:DimensionProperty;
    rdfs:isDefinedBy <http://code-research.eu/ontology/visual-analytics>;
    .

owl:Restriction owl:onProperty va:representsAxis;
    owl:minCardinality 1;
    owl:maxCardinality 1;
    .

va:representsObservation a rdf:Property, owl:ObjectProperty;
    rdfs:label "represents observation"@en;
    rdfs:comment "The mapping from a data point to a Cube  Observation."@en;
    rdfs:domain SIO_000465 ;
    rdfs:range  qb:Observation;
    rdfs:isDefinedBy <http://code-research.eu/ontology/visual-analytics>;
    .

owl:Restriction owl:onProperty va:representsAxis;
    owl:minCardinality 1;
    owl:maxCardinality 1;
    .
```

```
va:representsDatasets a rdf:Property, owl:ObjectProperty;
    rdfs:label "represents observation"@en;
    rdfs:comment "The mapping from a data series to a Cube data set to."@en;
    rdfs:domain SIO_000464  ;
    rdfs:range  qb:DataSet;
    rdfs:isDefinedBy <http://code-research.eu/ontology/visual-analytics>;
    .

owl:Restriction owl:onProperty va:representsAxis;
    owl:minCardinality 1;
    owl:maxCardinality 1;
    .

va:displayMeasure a rdf:Property, owl:ObjectProperty;
    rdfs:label "represents observation"@en;
    rdfs:comment "Associates a chart with Cube Measures."@en;
    rdfs:domain SIO_000904   ;
    rdfs:range  qb:MeasureProperty;
    rdfs:isDefinedBy <http://code-research.eu/ontology/visual-analytics>;
    .

owl:Restriction owl:onProperty va:representsAxis;
    owl:minCardinality 1;
    .


# --- Dashboard-Specific Classes ---------------------------

va:Dashboard a rdfs:Class, owl:Class;
    rdfs:label "Dashboard"@en;
    rdfs:comment "Represents a collection of one or more charts that visualize the
same Data Cube."@en;
    rdfs:isDefinedBy <http://code-research.eu/ontology/visual-analytics>;
    .


# --- Dashboard-Specific Properties ---------------------------

va:visualizes a rdf:Property, owl:ObjectProperty;
    rdfs:label "visualizes"@en;
    rdfs:comment "Links a Visual Analytics Dashboard with a Data Cube or Data
Slice"@en;
    rdfs:domain  va:Dashboard;
    rdfs:range qb:DataSet, qb:Slice;
    rdfs:isDefinedBy <http://code-research.eu/ontology/visual-analytics>;
    .

owl:Restriction owl:onProperty va:visualizes;
    owl:minCardinality 1;
    owl:maxCardinality 1;    .
```