Open Data Vis in Browser

Michael Krisper, Stephan Oberauer and Thomas Schlager
(G3)

A survey in
706.057 Information Visualisation SS 2012
Graz University of Technology

02 May 2012

Abstract

This survey explores the issue of *open data visualisation* in *web browsers*. Our findings are based on online field research and information gained from reverse engineering accessible projects. Several of these projects are described in detail.

Besides giving a short introduction to open data in general we try to find common patterns in the way developers achieve the move from gathering raw data to actually displaying it. This goal was only partly achieved by the creation of a *common transition model* that fits to the majority of available projects. But our sample of projects did not show any significant standard process. Right now we can only conclude that most open data based visualisation projects are unique creations in terms of how data is traversed from its origin to final display.

Contents

Chapter 1

Introduction

This survey examines several aspects of open data visualisation in web browsers. It provides a short theoretical introduction to the most important terms in this chapter. Chapter 2 is an attempt to explain why a common non obvious pattern in data transition occurs in most projects we analysed. Chapter 3 presents and describes technical aspects of selected example projects from our online field research. Most of our project insights are based on reverse engineering available program code. Chapter 4 concludes our findings and deals with several questions:

- Is there a standard in the way data is made available and retrieved?
- Are there common patterns in the way data is processed and visualised?
- Is it possible to retrieve and visualise open data by using only browser based (client side) processing?

1.1 Open Data

Public institutions collect and organize huge quantities of information regarding various sectors, e.g. population statistics, legislative issues, patent registrations and many more. It is possible to re-use and combine published information to conclude new findings. Sadly there is a lack of information on the masses of chaotically scattered data. [Kalampokis et al., 2011]

Open data can be defined through its characteristics:

- it is freely available
- it is structured and machine readable
- it can be re-used and re-published by everyone (free from legal restrictions like patents)

It should be published open with a documented interface and made available online on governmental infrastructure. Open data contains general public and anonymised data. [Wikipedia, 2012]

A lot of different file formats are used for making data available. Amongst them are plain text files (.txt), comma separated value lists (.csv), spreadsheet formats (e.g. .xls, or .ods), simplistic text based container formats (e.g. JSON) and well structured data formats (like RDF). [Matzat, 2011]

RDF (Resource Description Framework) is specified by the *World Wide Web Consortium (W3C)*. It is an XML based metadata model, which is typically used in combination with a query language like *SPARQL*. There is an official RDF[1] website by the W3C documenting the syntax. [W3C, 1999]

1.2 Data and Its Preparation

Before open data can be being visualised it needs to be prepared. Several activities are involved in delivering data from one endpoint (public organisation) to the other (browser, viewer). Chapter 2 explains some of these steps in detail. Whenever data is passed in a chain of actors or checkpoints we call it a transition. Common to all projects are three core activities: fetch, convert and display data. Data can be fetched from official sources like governmental or third party websites or be stored locally.

Few general assertions can be made about the data itself. It can be static data released and updated only once, frequently updated or real time data. It can carry statistics, geographical data or ultimately any type of data.

Some data even needs cleaning which means to get rid off unwanted data fragments or formats. There are tools in like *Data Wrangler*[2] or *Google Refine*[3], which support the process of data cleaning.

1.3 Data Visualisation in Browser

The idea of information visualisation combined with open data and browser technology delivers rich possibilities to express datasets and certain aspects of it. Today browser based visualisation can rely on technologies like *JavaScript*, *Adobe Flash* or *Oracle Java* and deliver visual presentations that compete with those of standalone applications. Therefore we are calling the browser rendering and data retrieval a browser application (that is bound to the constraints of web browser technologies and security aspects). With the uprising of JavaScript in the last few years modern libraries like *jQuery* deliver possibilities to create interactive and responsive applications without using proprietary containers.

[1]http://www.w3.org/TR/PR-rdf-syntax/
[2]http://vis.stanford.edu/wrangler/
[3]http://code.google.com/p/google-refine/

Chapter 2

The Transition of Data

Projects in *Open Data Vis in Browser* show common patterns of client server access structures and data transitions. This chapter illustrates the most common patterns based on our research. In the context of this report we bind the following terms (which represent actors) to its descriptions:

- **Open data provider**: a public institution (governmental or other) providing open data for public use. The term may reference the institution itself or its online server infrastructure providing the data. There is no declaration of type or format of provided data.

- **Vis developer**: an entity providing a browser based visualisation application (and possibly data to be displayed). It can be: a single person, a group of persons or a company. The term may reference the entity itself or its online infrastructure providing the application (and possibly data).

- **Browser**: a web browser running a visualisation application through built in technologies or third party plugins. It retrieves data and interprets it.

Open data provider and vis developer are a single unit if and only if the public institution develops, provides and hosts the visualisation application itself. In general this is not the case since most applications are developed and hosted by third parties.

Before and during a browser based visualisation several activities are involved, e.g. *provide data, fetch data, transform data, store intermediate data, do computations on data* or *display data*. While several actions are naturally bound to a single unit (like *display data* to *browser*) some are not. Figure 2.1 shows a trivial (and probably the most obvious) setup of actors and activities. In this case the vis

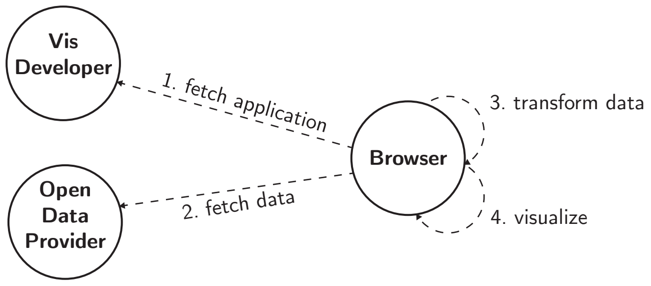


Figure 2.1: Trivial Transition Model

developer is providing an application which when run in browser directly connects to the open data provider, downloads the data, transforms and displays it. But in fact our research shows that this simple approach is not very common.

Figure 2.2 shows the data transition in a way found in many projects today. As pointed out

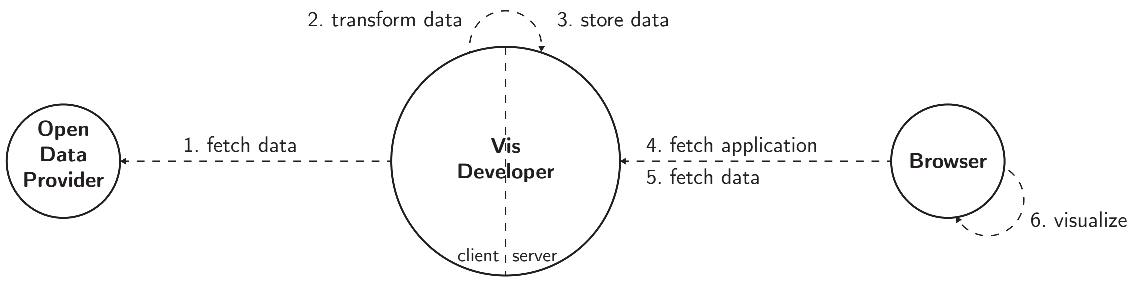


Figure 2.2: Common Transition Model

by the larger circle the vis developer plays a more significant role. It provides the application itself **and** the data necessary for visualisation. On one hand it has a server role (right side) to the browser and on the other hand (left side) it is a client on its own to the open data provider. Activities 1 and 2 (*fetch data*, *transform data*) are not necessarily bound to browser activities (requests) 4 and 5 (although they can be). This is heavily dependent on the type of application and data (*static data* or *live data*).

2.1 Dependency Comparison

To identify advantages for choosing the approach shown in figure 2.2 over the one in figure 2.1 we elaborate the bonds resulting from those two models. If two actors or activities between them are dependent in order to work we call it a bond. Dependencies related to the underlying technology like JavaScript, Flash or other plug ins are not of concern in this consideration - we assume that full browser side support for application execution is given.

At first glance the trivial model involves a strong bond between the open data provider and the browser because in order for the application to work the available data must remain in a consistent format or the application needs updating. Since the application is provided by the vis developer (and is always up to date) the bond is actually passed on to it although there are no direct activities between the vis developer and the open data provider. Still there can be a (weaker) bond between browser and open data provider. Some applications require server responses to meet certain constraints like *response time*.

The common model also involves a strong bond between the vis developer and the open data provider for the intermediate data processing to work. Both the application and final data is provided by the vis developer which means there is no strong bond between vis developer and browser. The latter constraint related bond in the preceding paragraph still applies - only this time between vis developer and browser.

Both models suffer from the same strong dependence between vis developer and open data provider. A major difference as shown in figures 2.3 and 2.4 is the relation between browser and vis developer or browser and open data provider. A benefit on choosing the common approach is a gain in control by the vis developer. Another advantage can be identified: in case the open data

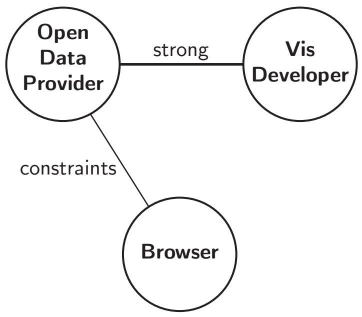


Figure 2.3: Trivial Approach Bonds

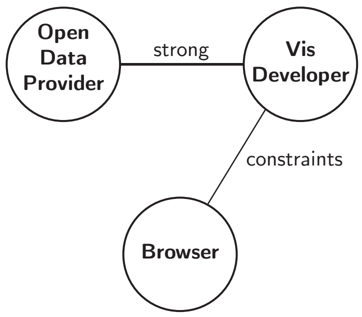


Figure 2.4: Common Approach Bonds

provider changes the format or availability of data there might remain a working set of stored intermediate data by the vis developer.

2.2 Other Benefits

Section 2.1 points out that there are benefits in terms of dependency in the common approach of figure 2.2. We can identify more reasons why current projects avoid the direct link between open data provider and browser:

- In almost any case available open data needs conversion in order to work with available toolkits like *JIT* or *jQuery*. While repeatedly transmitting data to browsers cannot be avoided it has a certain appeal to convert data only once and save it (if possible).
- Downloading application and data from the same source improves speed.
- Providing application and data the vis developer can decide on its own what to make available or when to remove content. Since e.g. JavaScript code is always visible, the application itself can be copied by anyone and executed independent of the vis developer. If an application directly accesses the open data provider the vis developer has no means to restrict its own creation after release.

Open data has got some drive in last few years. But the situation of available data and applications visualising it can still be seen as on a premature development stage. Even though we see the pattern of figure 2.2, there is a lack of standard approaches to accomplish the move from raw data to visualisation. Chapter 3 shows that projects following a similar transition pattern can be quite different from one another on the inside.

Chapter 3

Analysis of Existing Projects

During our research we took a wider look at over 40 projects to find a way of grouping them. Grouping could base on:

1. the way data is processed and traversed
2. the type or origin of data that is covered
3. the type of visualisation used.

Most projects follow the same common data transition approach, but are unique creations with a lot of individually written code and the way libraries are used. Therefore the first suggestion does not fit to form groups. Neither does the second one because the projects we describe in detail are quite different from one another. While we could form two major groups named geographical data and non geographical data we chose to use the type of visualisation which slightly connected to it. A lot of projects use maps to display some kind of open data. It seams reasonable to group into map based visualisations and non map based visualisations.

3.1 Map Based Visualisation

Map based visualisations process some sort of geographically bound information on maps. They do not use any specific software related to information visualisation. They rather rely on proprietary or open map libraries and APIs like *Google Maps*.

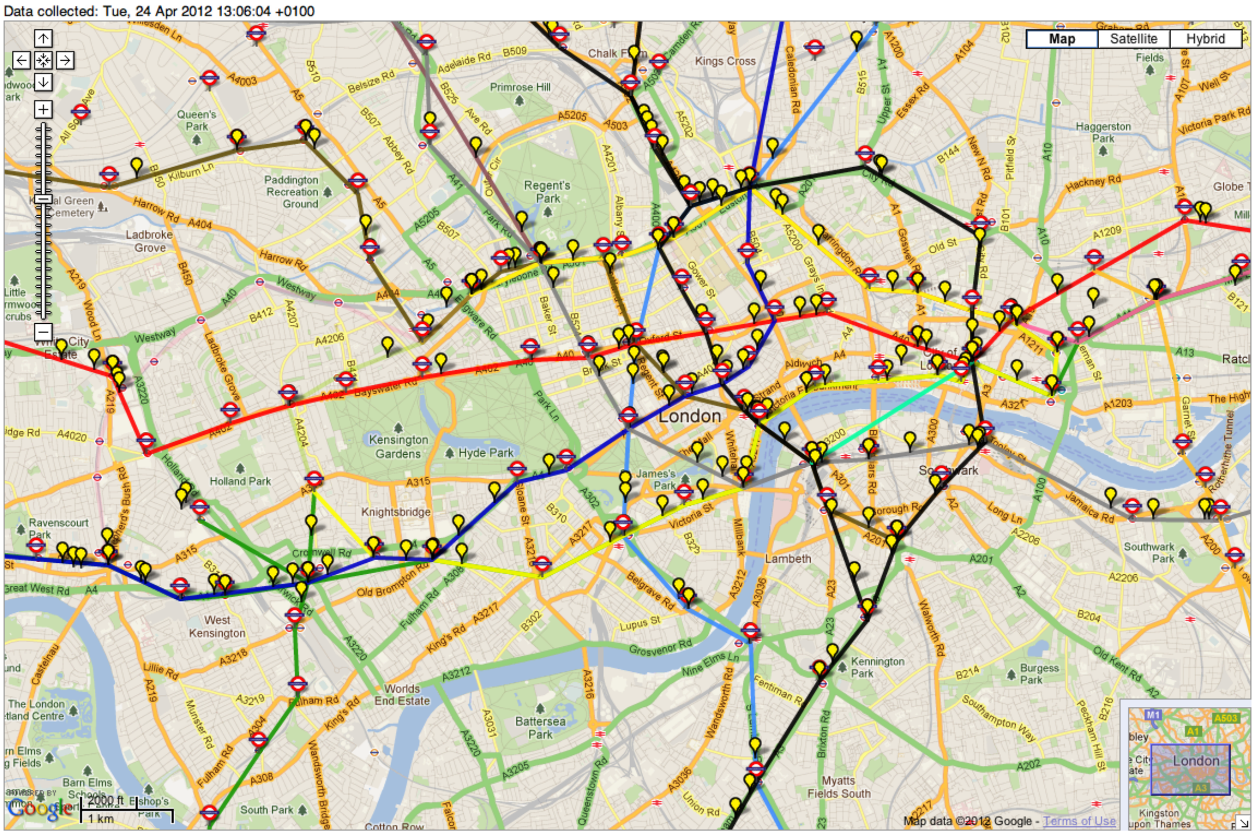
In order to satisfy the map API data needs to be prepared. We can see a common factor which is the use of JSON - JavaScript based and suitable for using the free Google Maps service [Büttiger, 2006]. Map visualisations are widespread and show different types of data from crime events to live display of tube train positions.

3.1.1 Project "London Train Live Map"

This visualisation created by Matthew Somerville during the ScienceHackday in June 2010 overlays the London Tube Map over a Google Maps Plugin and displays the realtime position of the trains. The live data is fetched from the Transport for London Departure Webservice every 30 seconds, and the position is interpolated between live updates.

URL:	`http://traintimes.org.uk/map/tube/`
Created By:	Matthew Somerville, 2010
Data Source:	*Original Source:*Transport for London Webservice `http://cloud.tfl.gov.uk/TrackerNet/PredictionSummary/` *Intermediate Source:*`http://traintimes.org.uk/map/tube/data/london.js`
Data Formats:	XML, JSON
Data Retrieval:	Query the original data every 30 seconds by a server script Live fetch of the temporary stored data by the browser
Visualisation:	Geographical Map
Frameworks/Toolkits:	Google Maps, Python Scripts

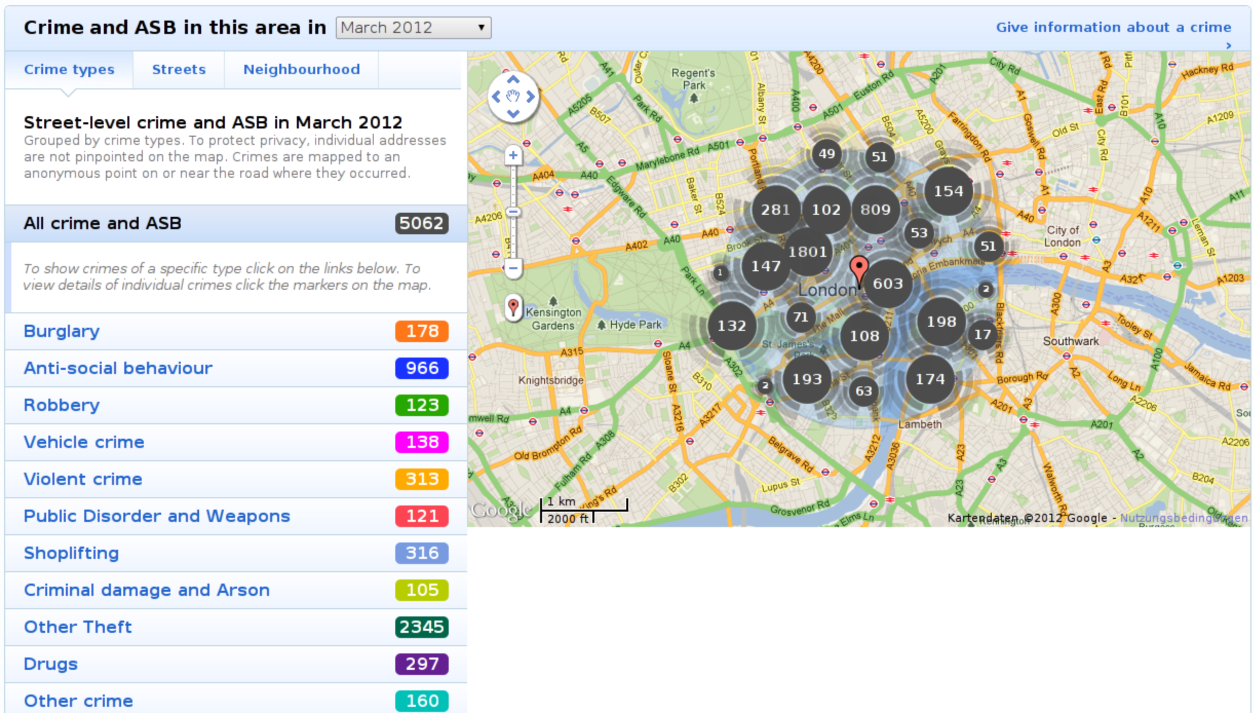
Data collected: Tue, 24 Apr 2012 13:06:04 +0100



3.1.2 Project "Crime Map"

The Crime Map of the Police UK (Project CrimeMapper) is a web application which shows the crime statistics and distribution for cities in the UK on a map.

URL:	`http://www.police.uk/`
Created By:	Rock Kitchen Harris on behalf of Police UK, Jan 2011
Data Source:	*Web Service:* `http://www.police.uk/crime/radius/all-crime`
Data Formats:	JSON
Data Retrieval:	Webservice-query (URL encoded) by webbrowser on opening the page
Visualisation:	Geographical Map
Frameworks/Toolkits:	Google Maps, jQuery



Crime and ASB in this area in March 2012

Street-level crime and ASB in March 2012
Grouped by crime types. To protect privacy, individual addresses are not pinpointed on the map. Crimes are mapped to an anonymous point on or near the road where they occurred.

To show crimes of a specific type click on the links below. To view details of individual crimes click the markers on the map.

All crime and ASB	5062
Burglary	178
Anti-social behaviour	966
Robbery	123
Vehicle crime	138
Violent crime	313
Public Disorder and Weapons	121
Shoplifting	316
Criminal damage and Arson	105
Other Theft	2345
Drugs	297
Other crime	160

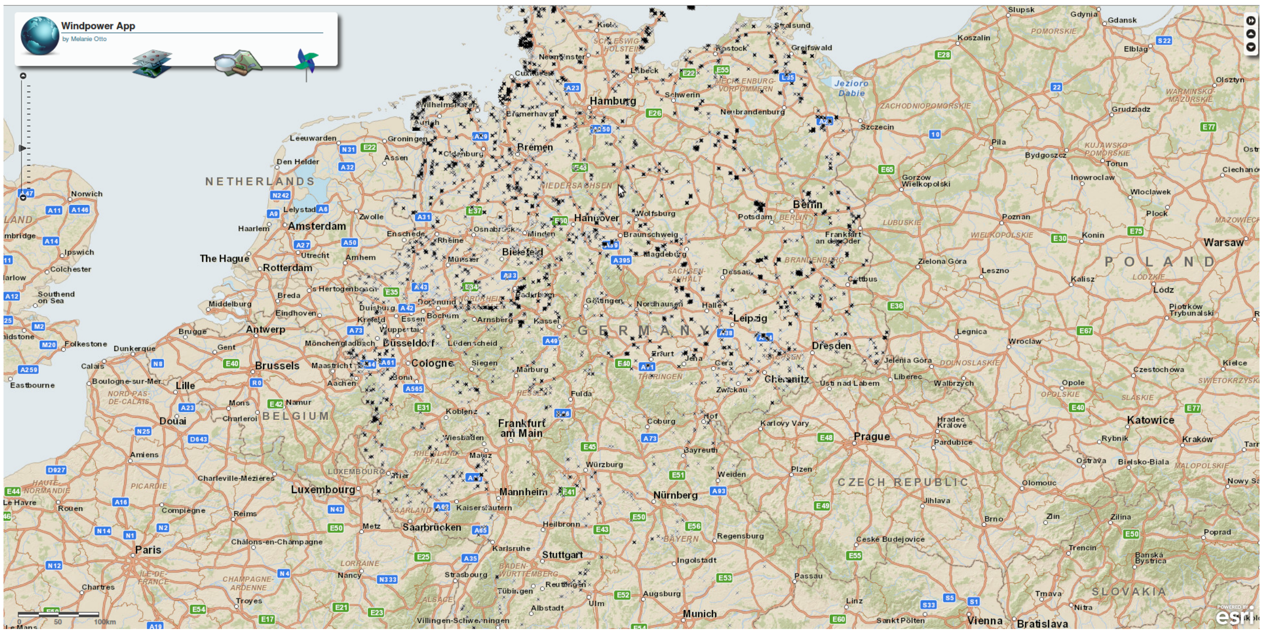
3.1.3 Project "Windpower App"

This project was made in the context of the german Open-Data contest *Apps für Deutschland* from the student Melanie Otto. The idea of this project was to use public open data and visualize it for the usage of the citizens of germany. The project displays a map with the wind energy plants of germany and their calculated shadow over the year. Otto [2012a]

The data about the wind energy plants she uses get exported from the open-street-map project and are stored in her own database. To fetch the data from the database she uses the Dojo-Framework, which is a JavaScript Framework with the possibility to access a rdf server with SPARQL.

The visualisation of the data is made with the map from the ArcGIS Server, which is a server based geoinformationsystem with different maps, which can be accessed over the browser.

URL:	`http://46.137.102.73/windpower_app/web/`
Documentation:	`http://46.137.102.73/windpower_app/windpower_app_doku.pdf`
Created By:	Melanie Otto, 2012
Data Source:	*Original Source:* Open-Street-Map export, `http://www.openstreetmap.org/`
Data Formats:	JavaScript
Data Retrieval:	fetching data with Dojo-Framework from own server
Visualisation:	Map (ArcGIS Server)
Frameworks/Toolkits:	jquery, Dojo-Framework



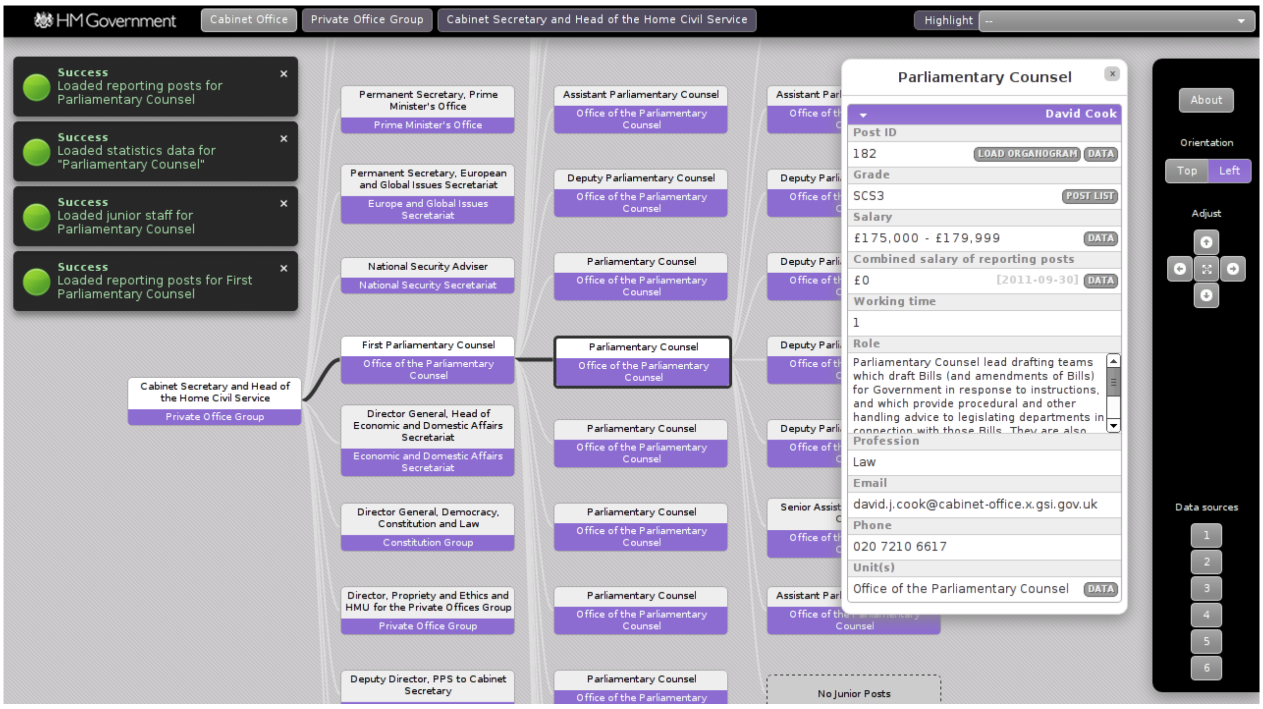
3.2 Non Map Based Visualisation

Non map based open data visualisations rely on the same technologies normal information visualisation applications do. They use JavaScript libraries like JIT or ProtoVis in combination with browser supported features like HTML, CSS and DOM rendering capabilities.

3.2.1 Project "Cabinet Office Organogram"

The Cabinet Office of the UK created this project in order to increase transparency about the members and civil servants working in the government. It displays the hierarchy and organizational structures of the government and their respective data.

URL:	`http://reference.data.gov.uk/gov-structure/organogram/?dept=co&post=1`
Created By:	CabinetOffice UK
Data Source:	CabinetOffice `http://reference.data.gov.uk/doc/department/co/post/`
Data Formats:	JSON
Data Retrieval:	Live fetching JSON Data with from their own server (filtered by URL parameters).
Visualisation:	Spacetree visualisation
Frameworks/Toolkits:	JQuery, JIT, jGrowl (User notifications)


HM Government
Cabinet Office
Private Office Group
Cabinet Secretary and Head of the Home Civil Service
Highlight
--
Cabinet Secretary and Head of the Home Civil Service
Private Office Group
Permanent Secretary, Prime Minister's Office
Prime Minister's Office
Permanent Secretary, European and Global Issues Secretariat
Europe and Global Issues Secretariat
National Security Adviser
National Security Secretariat
First Parliamentary Counsel
Office of the Parliamentary Counsel
Director General, Head of Economic and Domestic Affairs Secretariat
Economic and Domestic Affairs Secretariat
Director General, Democracy, Constitution and Law
Constitution Group
Director, Propriety and Ethics and HMU for the Private Offices Group
Private Office Group
Deputy Director, PPS to Cabinet Secretary
Assistant Parliamentary Counsel
Deputy Parliamentary Counsel
Parliamentary Counsel
Office of the Parliamentary Counsel
No Junior Posts
About
Orientation
Top
Left
Adjust
Data sources
1
2
3
4
5
6

Success
Loaded reporting posts for Parliamentary Counsel

Success
Loaded statistics data for "Parliamentary Counsel"

Success
Loaded junior staff for Parliamentary Counsel

Success
Loaded reporting posts for First Parliamentary Counsel

Parliamentary Counsel

David Cook

Post ID	
182	LOAD ORGANOGRAM DATA
Grade	
SCS3	POST LIST
Salary	
£175,000 - £179,999	DATA
Combined salary of reporting posts	
£0	[2011-09-30] DATA
Working time	
1	
Role	
Parliamentary Counsel lead drafting teams which draft Bills (and amendments of Bills) for Government in response to instructions, and which provide procedural and other handling advice to legislating departments in connection with those Bills. They are also	
Profession	
Law	
Email	
david.j.cook@cabinet-office.x.gsi.gov.uk	
Phone	
020 7210 6617	
Unit(s)	
Office of the Parliamentary Counsel	DATA

Data Representation

The data is represented as JSON for the visualisation. There are several fields which are used to display specific information about the government.

Data Retrieval

The Data is fetched by applying an http-get request in a special format (parameters are url encoded). The return value is in JSON data format an is processed with javascript for the visualisation in the webbrowser.

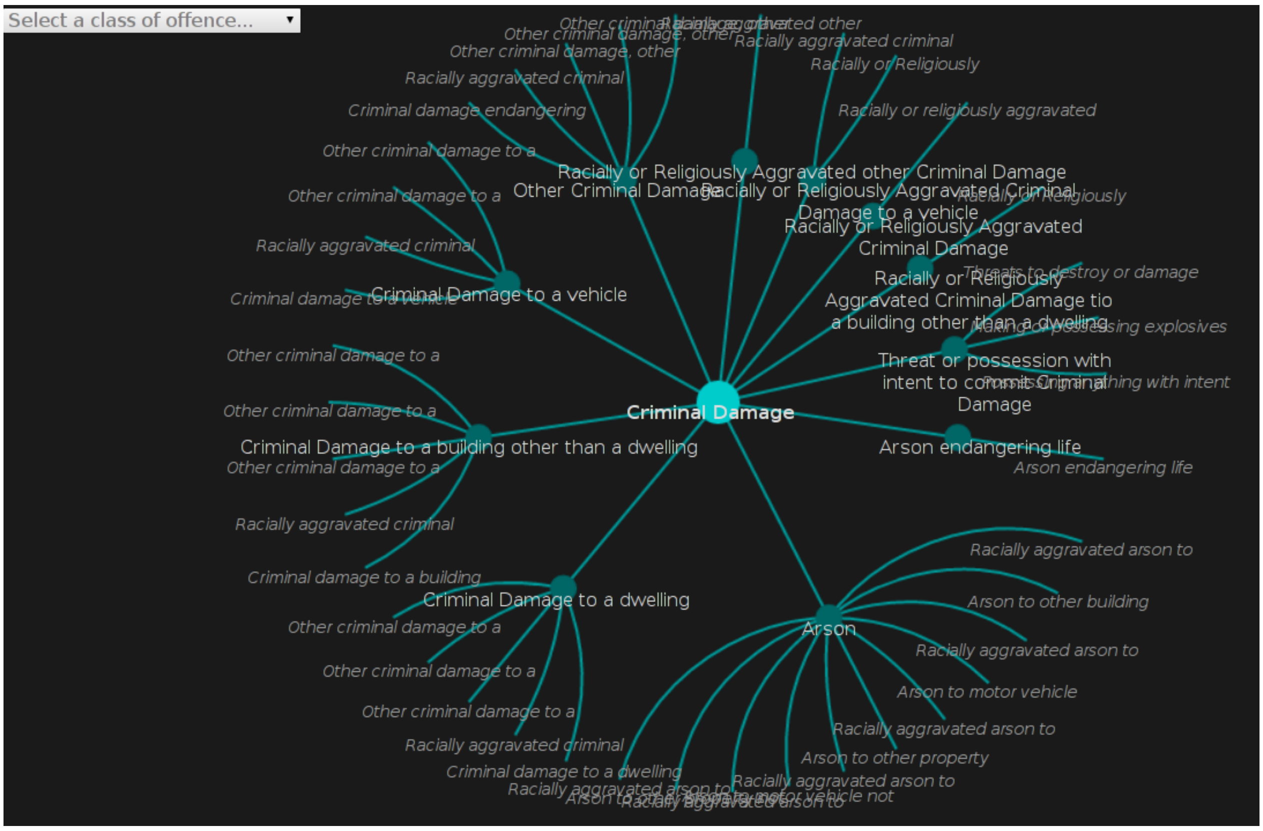
Data Visualization

The data is visualized as a Spacetree visualisation with the JIT toolkit. Further Information about JIT: `http://thejit.org`. A detail box displays the detailed data to one selected node. Notifications about every data request are also displayed as overlay to the visualisation.

3.2.2 Project "Offence Hierarchy Visualisation with rdfQuery and JIT"

This Project was created by Jeni Tennison in 2009 as a demonstration of hierarchy visualisation in combination with semantic web data. The Website displays a hierarchy of crimes (according to law) in a circular hyperbolic tree visualisation. The original data came from the UK Homeoffice as a spreadsheet and was initially cleaned and transformed into SKOS RDF (Simple Knowledge Organization System). The RDF Data is queried live by the webbrowser with SPARQL and then converted to JSON for the visualisation.

URL:	`http://www.jenitennison.com/visualisation/offences.html`
Created By:	Jeni Tennison, 2009
Data Source:	*Original Source:* Home Office UK, `http://www.homeoffice.gov.uk/rds/pdfs09/countnotif09.xls` *Intermediate Host:* http://www.jenitennison.com
Data Formats:	RDF (SKOS), JSON, XLS
Data Retrieval:	Initial conversion of source-data (XLS to RDF SKOS) Live querying RDF SKOS data from own server (with SPARQL) Converting to JSON on the fly in the webbrowser
Visualisation:	Circular Hierarchy Tree
Frameworks/Toolkits:	JQuery, JQuery RDF, JIT



Select a class of offence...
Criminal Damage
Criminal Damage to a vehicle
Criminal Damage to a building other than a dwelling
Criminal Damage to a dwelling
Arson
Arson endangering life
Threat or possession with intent to commit Criminal Damage
Racially or Religiously Aggravated other Criminal Damage
Other Criminal Damage
Racially or Religiously Aggravated Criminal Damage to a vehicle
Racially or Religiously Aggravated Criminal Damage
Racially or Religiously Aggravated Criminal Damage tio a building other than a dwelling
Other criminal damage, other
Racially aggravated criminal
Criminal damage endangering
Other criminal damage to a
Racially or religiously aggravated
Threats to destroy or damage
Having or possessing explosives
Possessing anything with intent
Arson endangering life
Racially aggravated arson to
Arson to other building
Arson to motor vehicle
Arson to other property
Criminal damage to a building
Criminal damage to a dwelling
Criminal damage to a vehicle

Data Representation

The Data was fetched from homeoffice.gov.uk as a spreadsheet (`http://www.homeoffice.gov.uk/rds/pdfs09/countnotif09.xls`) and converted into SKOS RDF format by Jeni Tennison (it is not exactly known how she converted it). The data was translated into CONCEPTS-Entities of the SKOS format.

SKOS (RDF) is a data format to represent RDF data. In SKOS the data is splitted into conceptual ressources (CONCEPT-Entities) which are related to each other and described by several types of fields. SKOS is very handy for classification schemes, taxonomies, or thesauri. Further information: `http://www.w3.org/TR/skos-primer/`

Data Retrieval

The data is hosted on www.jenitennison.com and is fetched by an http-get request (with javascript in the browser). The query language SPARQL is used to query the retrieved RDF-SKOS data for specific elements.

SPARQL is a query language for RDF data. A sparql query consists of several conditions in the form: ¡subject¿ ¡predicate¿ ¡object¿. It has two main parts: a SELECT part where the output is defined and a WHERE part which defines the conditions. The query is applied on a existing set of xml-rdf data (which has to be fetched separately). Further information: `http://www.w3.org/TR/rdf-sparql-query/`

Data Visualization

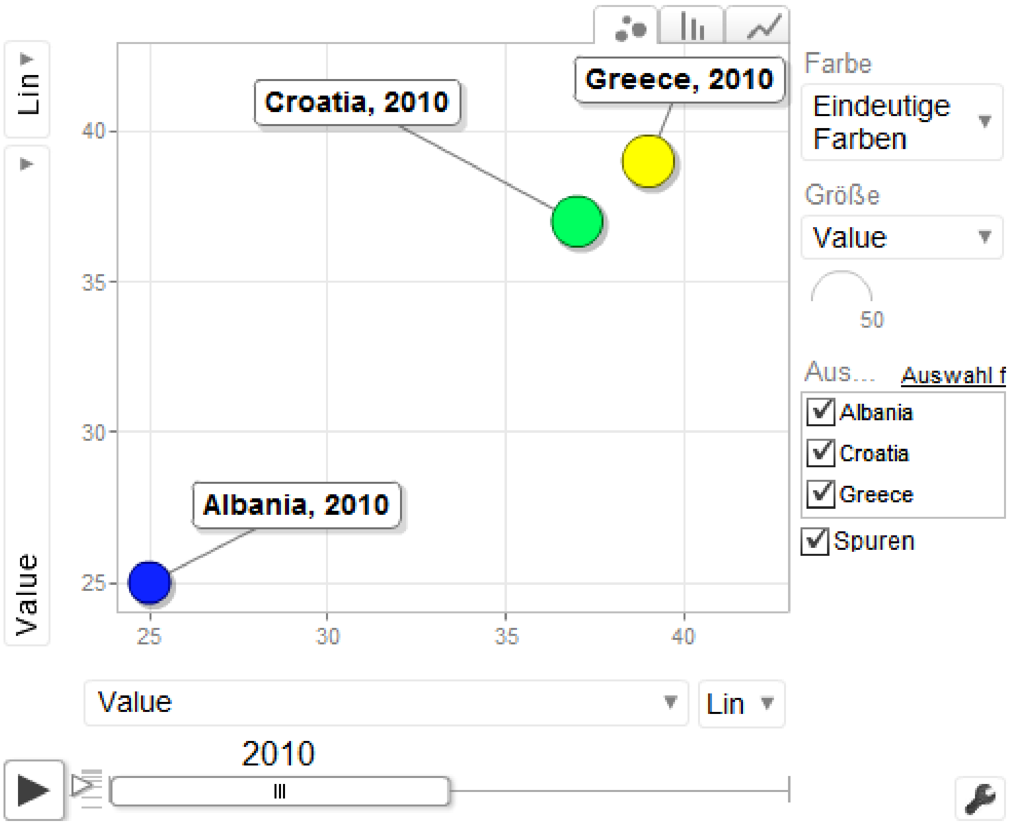
The data is visualized with a circular hyperbolic tree, implemented in JIT - a javascript toolkit for visualisation. Further Information about JIT: `http://thejit.org`

3.2.3 Project "Open Governmental Data on the Web: A Semantic Approach"

The goal of the project "Open Governmental Data on the Web: A Semantic Approach" is to semantically and formally represent governmental data. Large datasets can be queried and visualized in an understandable way and are open to everyone. In the paper Julia Hoxha and Armand Brahaj discuss their semantic approach to the problem of heterogeneous formats in open data projects.

Currently this project is only of theoretical nature written down on the paper of Julia Hoxha and Armand Brahaj. Hoxha and Brahaj [2011] They split their paper in two stages, the first stage is the data aggregation and data processing. For this project the sources would come from different governments as raw datasets, which should get converted to RDF triples and stored in the "Knowledge Base". The data is stored in rdf/XML format in CKAN Repositories. To access the pulic rdf triples a CKAN web interface and the CKAN API is needed. The second stage should consist of a set of tools which provide e.g. different sets of visualisation. The data is also stored in the ODA repository and can be accessed via the Open Data API. The Visualisation is done with the Google Visualisation API through a SPARQL query.

Created By:	Julia Hoxha, Armand Brahaj, 2011
Data Source:	*Original Source:* open governmental homepages,
Data Formats:	RDF
Data Retrieval:	Live querying RDF SKOS data from own server (with SPARQL)
Visualisation:	Google Visualisation API (Motion Charts)
Frameworks/Toolkits:	CKAN, ODA Repository, Google Visualisation



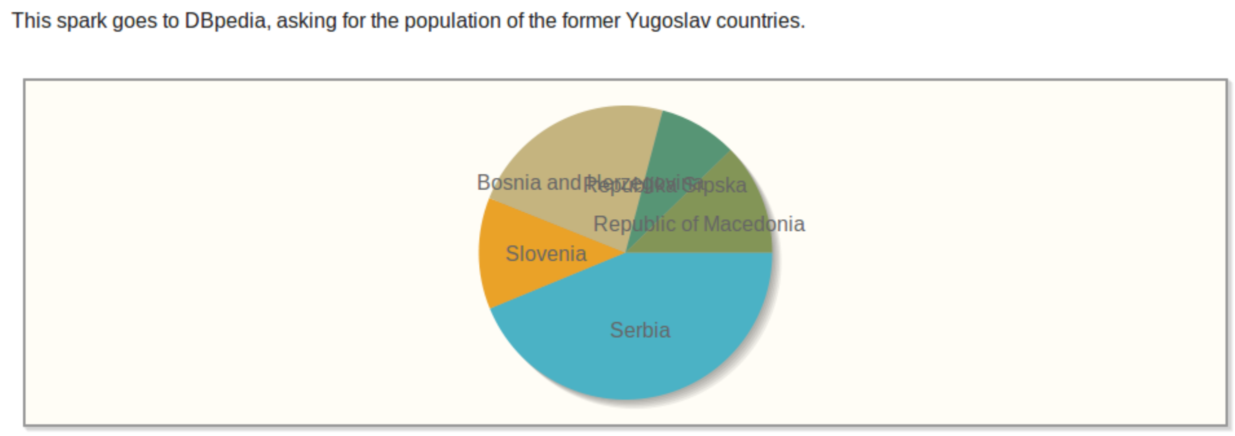
3.2.4 Project ”Spark”

The project Spark consists of a JavaScript library which allows to fetch RDF data, query it with SPARQL query language and visualize it in any HTML page. The toolkit can be simply included on a website or downloaded and locally included.

The data which is used in the examples on the project homepage get fetched from the open linked database DBpedia, which uses by default the RDF data type. With the query language SPARQL the datasets any datasets can be selected and formatted that they can be visualised as a pie chart, date chart, table or some other visualisation types.

Spark requires jquery to work correctly. It works either as an additional markup language or it can be used directly from JavaScript with the call of the function *spark*.

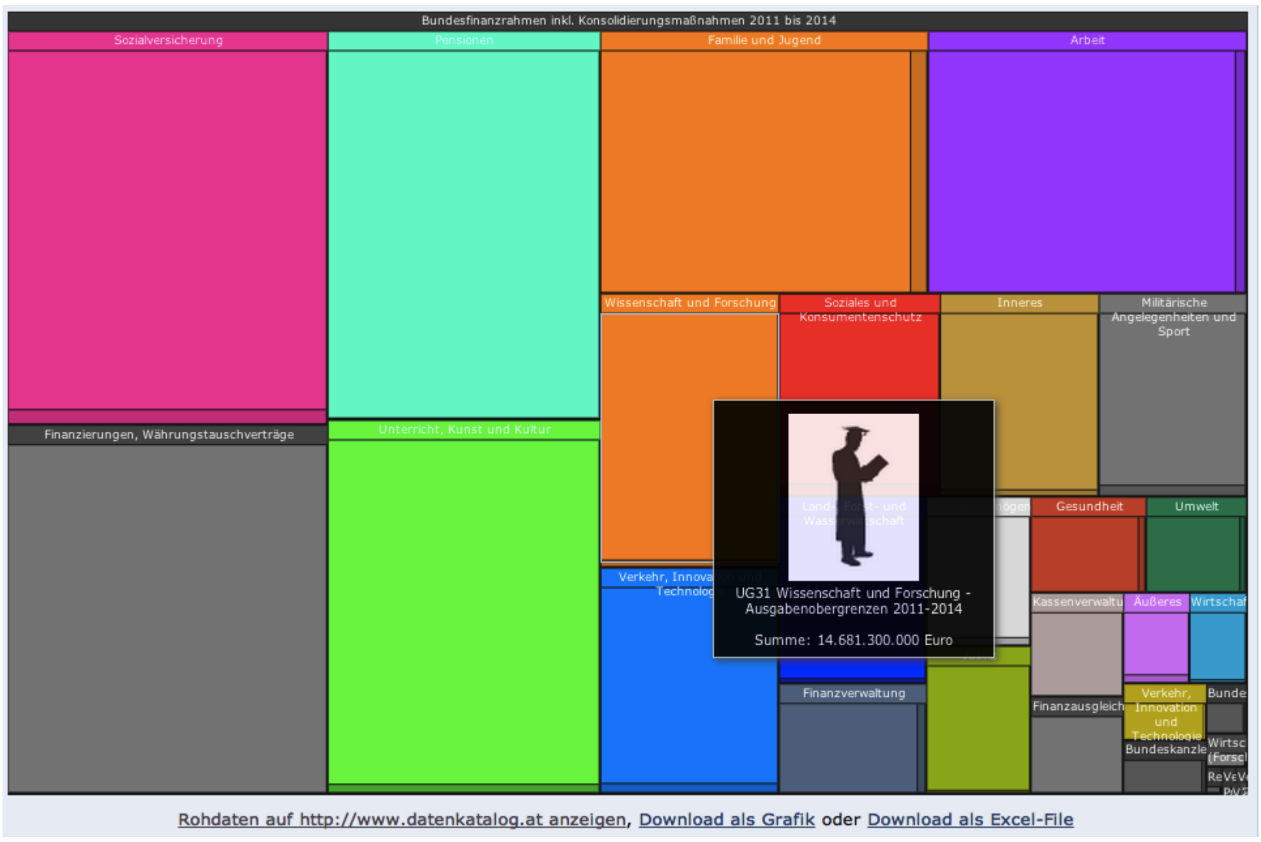
URL:	`http://km.aifb.kit.edu/sites/spark`
Created By:	Denny Vrandečić and Andreas Harth, 2011
Data Source:	*Original Source:* DBpedia, `http://dbpedia.org/sparql`
Data Formats:	RDF
Data Retrieval:	SPARQL
Visualisation:	Simple Table, DateChart, PieChart, PivotChart (by OAT)
Frameworks/Toolkits:	spark, jquery, jqplot, oat


This spark goes to DBpedia, asking for the population of the former Yugoslav countries.

3.2.5 Project "Sparpaket"

Open3.at provides an online visualisation of the Austrian *Bundesfinanzrahmen 2011-2014* (governmental spending plans). The visualisation is a JIT based rectangular tree map with hover information and the possibility to "zoom in" to certain areas. Data is hosted statically on open3.at's infrastructure. It originates from an official scanned PDF document of the governmental plans. In order to make it machine readable the PDF file was hand processed into spreadsheet software and most likely automatically exported to a JavaScript executable file with object/matrix definition.

URL:	`http://open3.at/sparpaket`
Created By:	Open3.at
Data Source:	Website of *Der Standard* (`http://images.derstandard.at/2010/10/27/papier.pdf`)
Data Formats:	Scanned PDF file.
Data Retrieval:	Manual extraction from scanned PDF file, spreadsheet export to JavaScript object code, evaluation of downloaded JavaScript file
Visualisation:	Treemap
Frameworks/Toolkits:	JIT-Treemap

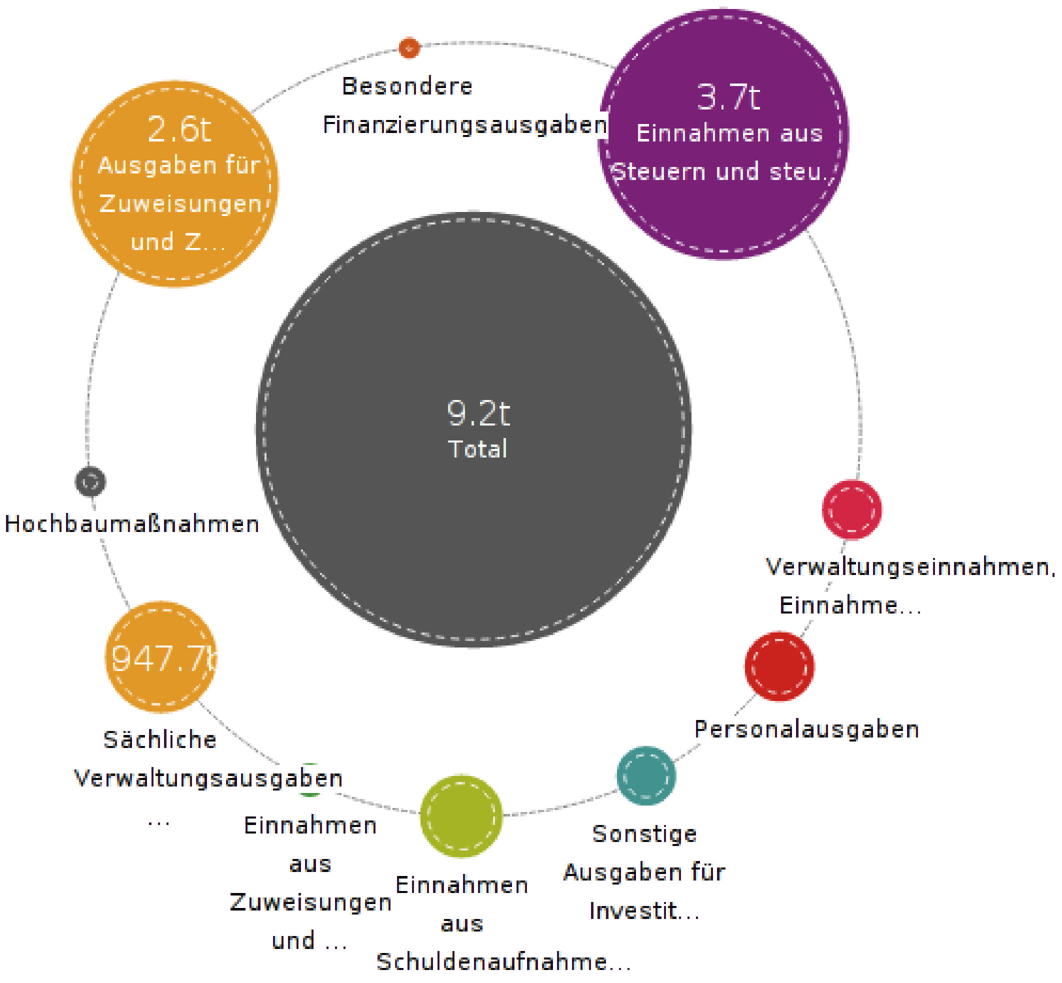


Bundesfinanzrahmen inkl. Konsolidierungsmaßnahmen 2011 bis 2014
Sozialversicherung
Pensionen
Familie und Jugend
Arbeit
Wissenschaft und Forschung
Soziales und Konsumentenschutz
Inneres
Militärische Angelegenheiten und Sport
Finanzierungen, Währungstauschverträge
Unterricht, Kunst und Kultur
Land- , Forst- und Wasserwirtschaft
Gesundheit
Umwelt
Verkehr, Innovation und Technologie
Kassenverwaltung
Äußeres
Wirtschaft
Finanzverwaltung
Finanzausgleich
Verkehr, Innovation und Technologie
Bundeskanzleramt
Bunde
Wirtsc (Forsc
ReVeV
UG31 Wissenschaft und Forschung - Ausgabenobergrenzen 2011-2014
Summe: 14.681.300.000 Euro
Rohdaten auf http://www.datenkatalog.at anzeigen, Download als Grafik oder Download als Excel-File

3.2.6 Project "Openspending"

OpenSpending.org is a webpage dedicated to storing financial open data of all kind on their webpage. They also implemented some charts to display the stored data. The possible chart types are: Treemap, Circular Bubble Map and 'Pivot Table'Pivot Table. The User can choose which columns to be displayed, and add filters to the data.

URL:	`http://openspending.org/`
Created By:	OpenSpending
Data Source:	Many financial datasets (OpenData) of different countries and institutions
Data Formats:	JSON
Data Retrieval:	Imported Data is stored permanently on their servers. Live querying the stored data by webbrowser
Visualisation:	Treemap, Circular Bubble Map, Pivot Table
Frameworks/Toolkits:	jQuery, JIT

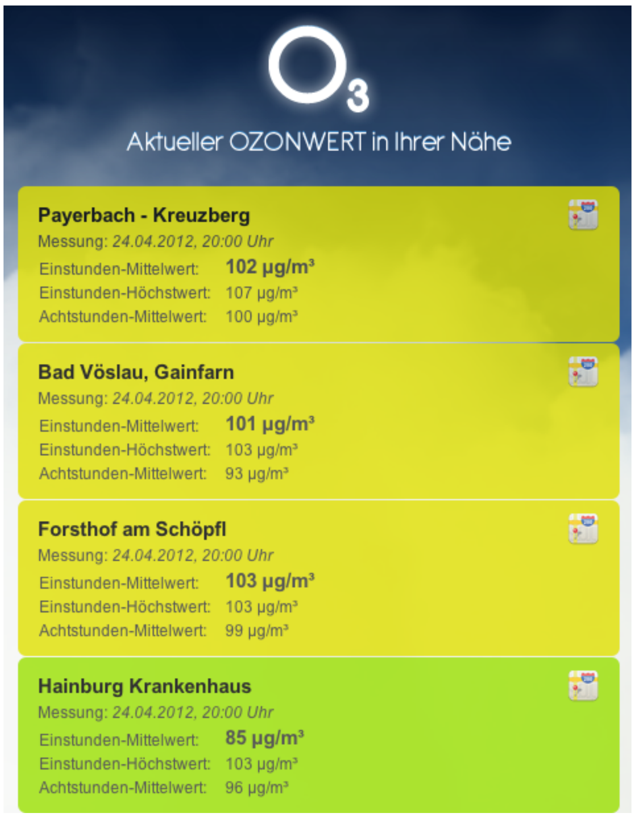


3.2.7 Project "Ozon-Info"

Ozon-Info is a small application developed for regular web browsers and mobile devices. The visual presentation of data is a simple css formatted box (or list of boxes) showing information of ozone levels of your current position (if available).

Ozone values are provided by the *Umweltbundesamt* of Austria directly in a *JSON*-formatted text file on their server. This makes he application one of the simplest we found because the data needs no transformation in order to work with libraries (jQuery in this case). jQuery is used to parse the JSON file.

URL:	`http://ozon-info.at`
Created By:	@sindrewimberger in cooperation with `open3.at`
Data Source:	Umweltbundesamt Österreich
Data Formats:	JSON provided by Umweltbundesamt
Data Retrieval:	Download JSON file
Visualisation:	CSS formatted HTML
Frameworks/Toolkits:	jQuery, Leaflet (map library)



Aktueller OZONWERT in Ihrer Nähe

Payerbach - Kreuzberg
Messung: *24.04.2012, 20:00 Uhr*
Einstunden-Mittelwert: **102 µg/m³**
Einstunden-Höchstwert: 107 µg/m³
Achtstunden-Mittelwert: 100 µg/m³

Bad Vöslau, Gainfarn
Messung: *24.04.2012, 20:00 Uhr*
Einstunden-Mittelwert: **101 µg/m³**
Einstunden-Höchstwert: 103 µg/m³
Achtstunden-Mittelwert: 93 µg/m³

Forsthof am Schöpfl
Messung: *24.04.2012, 20:00 Uhr*
Einstunden-Mittelwert: **103 µg/m³**
Einstunden-Höchstwert: 103 µg/m³
Achtstunden-Mittelwert: 99 µg/m³

Hainburg Krankenhaus
Messung: *24.04.2012, 20:00 Uhr*
Einstunden-Mittelwert: **85 µg/m³**
Einstunden-Höchstwert: 103 µg/m³
Achtstunden-Mittelwert: 96 µg/m³

Chapter 4

Conclusion

It is hard to find a simple verdict in this survey. One of our goals, to find a common pattern in data transition, could only be partially answered. While the we see a pattern in the greater picture of data transition ever single application has its own way of handling the data in detail.

Further on it is clear that the lack of standardised offering of open data implies a lot of thought when developing an application. Even if documented data formats like *RDF* are used there is no guarantee that the gathering, extraction and presentation of data can be accomplished by browser based technologies alone. Certain constraints like request times and parsing speed make trivial approaches difficult.

For that reason a lot of developers choose to implement an intermediate or alternative data pool in addition to official open data providers. This data is preprocessed to better fulfil the needs of today's libraries and toolkits. Those applications are quite independent from their original data source.

During the research we saw that several linked or praised projects were taken offline or malfunctioned. This fact can relate to several facts: vanishing of vis developer, modification of open data offerings, changes in libraries and technologies, incompatibilities with new browser releases and many more. It is hard to foresee the upcoming developments in this research field.

One observation is the heavy use of the JSON data format. We saw an involvement during data transition of more than 50% of the projects we analysed.

References

Büttiger, Leo [2006]. *Faster google maps with JSON*. `http://leo.freeflux.net/blog/archiv/faster-google-maps-with-json.html`. Access: 2012-05-02.

Denny Vrandecic, Andreas Harth [2011]. *Spark*. `http://km.aifb.kit.edu/sites/spark/`. Access: 2012-05-01.

Harris, Rock Kitchen [2011a]. *Crime Map*. `http://www.police.uk/`. Access: 2012-05-01.

Harris, Rock Kitchen [2011b]. *Crime Map*. `http://openspending.org/`. Access: 2012-05-01.

Hoxha, Julia and Armand Brahaj [2011]. *Open Government Data on the Web: A Semantic Approach*. In *2-nd International Conference on on Emerging Intelligent Data and Web Technologies (EIDWT-2011)*. Springer, Tirana.

Kalampokis, Evangelos, Efthimios Tambouris, and Konstantinos Tarabanis [2011]. *Open Government Data: A Stage Model*. *Lecture Notes in Computer Science*, 6846/2011.

Matzat, Lorenz [2011]. *Ein Glossar rund um Open Data*. *Zeit Online*. `http://blog.zeit.de/open-data/2011/05/13/begriff-definition-opengov/`. Access: 2012-05-02.

Open3.at [2012a]. *Austrian Ozon-Info Service*. `http://ozon.sonar1.mobi/`. Access: 2012-05-01.

Open3.at [2012b]. *Bundesfinanzrahmen 2011-2014*. `http://open3.at/sparpaket`. Access: 2012-05-01.

Otto, Melanie [2012a]. *Windpower App*. `http://46.137.102.73/windpower_app/windpower_app_doku.pdf`. Access: 2012-05-01.

Otto, Melanie [2012b]. *Windpower App*. `http://46.137.102.73/windpower_app/web`. Access: 2012-05-01.

Somerville, Mattew [2010]. *London Train Live Map*. `http://traintimes.org.uk/map/tube/`. Access: 2012-05-02.

Tennison, Jeni [2009]. *Offence Hierarchy Visualisation with rdfQuery and JIT*. `http://www.jenitennison.com/visualisation/offences.html`. Access: 2012-05-01.

W3C [1999]. *Resource Description Framework (RDF) Model and Syntax Specification*. W3C website. `http://www.w3.org/TR/PR-rdf-syntax/`. Accessed: 2012-05-02.

Wikipedia [2012]. *Open Data*. Online Encyclopedia. `http://en.wikipedia.org/wiki/Open_data`. Accessed: 2012-04-29.