Maximilian Eibl, Martin Gaedke. (Hrsg.): INFORMATIK 2017,
Lecture Notes in Informatics (LNI), Gesellschaft für Informatik, Bonn 2017 1915

CubeViz.js: A Lightweight Framework for Discovering and Visualizing RDF Data Cubes

Konrad Abicht[1], Georges Alkhouri[2], Natanael Arndt[3], Roy Meissner[4] and Michael Martin[5]

Abstract: In this paper we present CubeViz.js, the successor of CubeViz, as an approach for lightweight visualization and exploration of statistical data using the RDF Data Cube vocabulary. In several use cases, such as the European Unions Open Data Portal, in which we deployed CubeViz, we were able to gather various requirements that eventually led to the decision of reimplementing CubeViz as JavaScript-only application. As part of this paper we showcase major functionalities of CubeViz.js and its improvements in comparison to the prior version.

Keywords: Statistics, Exploration, Linked Data, Visualization

1 Introduction

It is possible to encode and publish statistics originated in several knowledge domains as part of the Web of Data using the RDF Data Cube vocabulary [CRT13]. In order to visualize these statistics, the web application CubeViz [Ma15] can be used to generate user friendly and interactive charts. During the last six years of developing CubeViz[6] as an OntoWiki extension [Fr15], we have experienced some major limitations of CubeViz. These are:

1. IT-infrastructures often have strict requirements for integrating new applications. For example PHP applications are sometimes not allowed.
2. CubeViz can not be deployed separately from OntoWiki. Thus, integrating CubeViz results in setting up a large technology stack that might be challenging to set up.
3. CubeViz uses the RDF framework Erfurt that doesn't support blank nodes. As a consequence, CubeViz is not able to process all kinds of valid Data Cubes.

[1] AKSW, Leipzig University, Augustusplatz 10, 04109 Leipzig, Germany abicht@informatik.uni-leipzig.de
[2] HTWK Leipzig, Gustav-Freytag-Str. 42A, 04277 Leipzig, Germany georges.alkhouri@stud.htwk-leipzig.de
[3] AKSW, Leipzig University, Augustusplatz 10, 04109 Leipzig, Germany arndt@informatik.uni-leipzig.de
[4] AKSW, Institute for Applied Informatics (InfAI), e.V. Hainstraße 11, 04109 Leipzig, Germany meissner@informatik.uni-leipzig.de
[5] AKSW, Leipzig University, Augustusplatz 10, 04109 Leipzig, Germany martin@informatik.uni-leipzig.de
[6] CubeViz project at Github: `https://github.com/AKSW/cubeviz.ontowiki`

doi:10.18420/in2017_191

4. Erfurt has currently no stable implementation of remote SPARQL endpoints and files as data sources. Thus, a cumbersome workflow is necessary in order to publish RDF graphs, for CubeViz especially RDF Data Cubes.

In order to tackle these issues we decided to reimplement CubeViz from scratch as a JavaScript-only application. To showcase the differences between both, CubeViz and its successor CubeViz.js, we present a brief overview on CubeViz, its architecture, core functionality and its limitations first in Sect. 2. Based on the mentioned limitations requirements for the new implementation were derived, which is now applicable to almost any web project. Sect. 3 contains the resulting implementation and its implications. Tools that deal with visualization of RDF Data Cube encoded statistics are presented in section Sect. 4. Lastly, the whole paper is summarized in Sect. 5, where also future development tasks are outlined.

General information about the CubeViz project can be found at the project page cubeviz.aksw.org[7].

2 Limitations of CubeViz and derived requirements for CubeViz.js

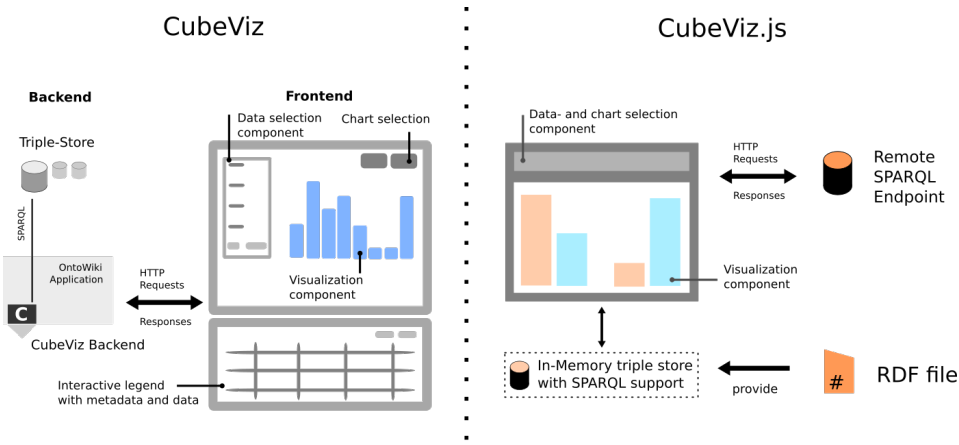

Fig. 1: Architectures of CubeViz and CubeViz.js

CubeViz [Ma15] is an OntoWiki extension for exploring Data Cubes expressed in RDF. In the following paragraphs, we briefly summarize the architecture and the three major components of CubeViz: *integrity analysis*, *faceted data selection* and *chart generation*. CubeViz is based on the architecture of the OntoWiki Application Framework (OAF) as illustrated in Fig. 1. The presentation layer (written in HTML and JavaScript) is used as an interface for human computer interaction tasks, while the application layer (implemented in

[7] CubeViz project website: `http://cubeviz.aksw.org/`

PHP) handles the communication with a configured SPARQL endpoint and processes and enriches the result sets of the queried RDF Data Cubes. The architecture is well supported through OAF regarding the back-end communication, message handling and GUI generation, but has some drawbacks, regarding optimization of the data flow in order to speed-up the application. Furthermore we discovered that it is not necessary to implement the application logic server-side. Implementing CubeViz as a client-side application moves major tasks (e.g. rendering or data processing) to the client, therefore reducing server load and HTTP request handling a lot. This speeds-up the application and reduces dependencies to various runtime environment libraries.

CubeViz is a client-server application that heavily depends on the availability of the server-side part. In case the server-side part crashes for some reason, the whole application isn't usable anymore. CubeViz.js in contrast is a client-side only application that connects to a server-side part and is thus resistant against server-side failures. Exchanging the used SPARQL endpoint is enough to get CubeViz.js back to work.

The GUI of CubeViz is designed in a modular way as illustrated on the left side of Fig. 1 and consists of three major components: data selection, visualization and the interactive chart legend. The data selection component provides a selection box for resources that are referenced by component properties (Dimensions, Attributes, Measures, Slices). By providing a desired selection to the respective server-side component, a result set containing all selected statistical observations is computed and sent to the visualization and chart legend components. The visualization component analyses the set of resulting observations to offer the selection of meaningful charts. Offered charts (e.g. bar chart and scatter plots) are selectable, visualized using different chart APIs (such as HighCharts[8] and d3.js[9]), and configured according to the users needs (e.g. the selection of a specific scale). The chart legend component contains all given observations, represented as a data table and provides the option to refine given values and to update the chart accordingly. Due to the fact that every (revised) data selection and its corresponding (configured) visualization is represented by a unique URL, generated charts are easily shareable.

Based on our own experiences with CubeViz and supported by the user community we derived the following requirements for CubeViz.js:

- *Client-side only Application:* To be fitted into the majority of IT-infrastructures and web applications, the server-side part has to be replaced by respective JavaScript components that are communicating to a SPARQL endpoint via HTTP(S).
- *Extend the support of further data sources:* CubeViz is able to process Data Cubes provided by a self-maintained SPARQL endpoint. Data Cubes that are published differently (e.g. Turtle or JSON files) are only processable if they are imported into

[8] HighCharts Website: `http://www.highcharts.com/`
[9] D3.js Website: `https://d3js.org/`

a SPARQL endpoint first. Apart from this, Data Cubes provided as JSON-LD will improve the integration into third party systems.

- *Improve handling:* Removal of unnecessary abstraction layers and improve handling of huge data sets (50+ dimension elements).
- *Prohibition of a zero selection:* If RDF Data Cubes are sparse, it is possible to receive an empty result set after using the data selection component of CubeViz. Every selection must result into a set of observations.
- *Data flow:* There has to be support for unidirectional data flow and loosely coupled web components. An unidirectional data flow improves, for instance, UI update conditions and debugging. Loose coupling helps to integrate CubeViz into different software environments.
- *Message bus:* Supporting the usage of a subscription based message bus for error and message handling.
- *Responsiveness:* The application needs to support common devices, such as smartphones, tablets and desktop computers.

3 Architecture and Implementation of CubeViz.js

As a result of the requirements and limitations presented in Sect. 2, we designed CubeViz.js as a JavaScript-only application. The following subsections outlines an overview of CubeViz.js's architectural details.

To integrate CubeViz.js in almost every web project we decided to use Flux[10] and React[11] as frameworks. Both frameworks complement each other perfectly and offer the ability to integrate with different design patterns and support loose coupling. Furthermore, we used the ecc-messagebus[12] that extends the Flux event dispatcher by a subscription based message bus and provides an improved error and message handling. CubeViz.js is implemented according to the ECMAScript 6 [Ec15] specification. We applied for example *Promises*, that is a convenient alternative to nested callbacks [KBDM13].

To provide RDF Data Cube resources from different sources (remote SPARQL endpoint files), we integrated rdfstore-js[13], a JavaScript-based triple store. Rdfstore-js supports SPARQL 1.0 and 1.1 standards for communication with web accessible triple stores. Furthermore rdfstore-js supports dynamic import of files containing triples (encoded with common notations) and provides them as an in memory model that may be queried via SPARQL. Architectural changes between CubeViz and CubeViz.js are depicted in Figure 1.

[10] Website of Flux: `https://facebook.github.io/flux/`
[11] Website of React: `https://facebook.github.io/react/`
[12] Github project of ecc-messagebus: `https://github.com/elds/ecc-messagebus`
[13] Github project of rdfstore-js: `https://github.com/antoniogarrote/rdfstore-js`

CubeViz provides a comprehensive set of features to select and visualize observations. In CubeViz.js, we adopted most of the concepts of CubeViz but improved them regarding the performance/handling using a configurable set of restrictions (i.e. the amount of observations to be visualized). Furthermore, CubeViz.js is being developed using the *mobile first* approach in order to create a responsive web application. Thus, CubeViz.js offers an enhanced user experience and may be used on almost any commonly used device.

The repository for CubeViz.js is hosted on Github at `https://github.com/AKSW/cubevizjs/`. Alongside the publicly available repository, we provide a web demo of the current development state of CubeViz.js[14] as depicted in Fig. 2.

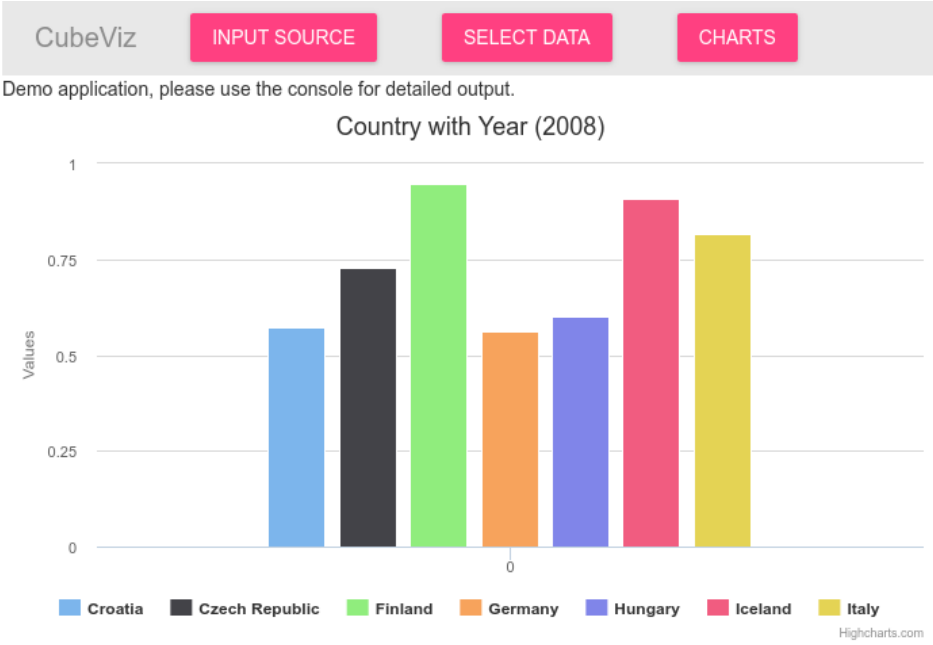

Fig. 2: Screenshot of CubeViz.js's user interface

4 Related Work

The W3C aggregated a comprehensive overview of tools in this research area at the W3C Wiki[15]. Three major tools from the W3C list are explained in the following paragraphs.

[14] CubeViz.js Demo: `http://cubevizjs.demo.aksw.org`

[15] W3C Wiki about RDF Data Cube visualization tools: `https://www.w3.org/2011/gld/wiki/Data_Cube_Implementations`

OpenCube Toolkit The OpenCube Toolkit [Ka14] enables to publish and to explore RDF Data Cubes using charts. It supports aggregation functions like `AVG` and `SUM` and integrates `R` scripts [Ri01, Da08]. These provide comprehensive support for statistical analyses. CubeViz.js is, in contrast, focused on data discovery tasks and is operating on materialized RDF Data Cubes. One of the advantages of CubeViz.js is the comprehensive faceted based search (i.e. to search and select dimension elements), that is as a basic version also available for OpenCube, but way less detailed.

Linked Data Cube Explorer LDCX The Linked Data Cube Explorer[16] (LDCX) is part of the OLAP4LD framework [KH14]. This framework enables to develop linked data web applications facilitating actions on RDF Data Cubes. LDCX provides a detailed data selection via dimension elements, measures, and a chart visualization, similar to CubeViz.

Linked Statistical Data Analysis The Linked Statistical Data Analysis[17] analyses and visualizes statistics encoded as RDF Data Cubes [CAR13]. The authors of the project summarized, that their work offers a Web based user-interface for a wide community (researchers, journalists, or interested people in general) which enables to compare statistical data from different sources without requiring any knowledge about the underlying technology. Furthermore, this tool provides a huge set of analysing functionalities due to the integration of R[18].

5 Conclusion and Future Work

We presented CubeViz.js, a lightweight Web-application for statistical data discovery and exploration tasks. The first version of CubeViz was implemented as an OntoWiki extension using PHP/HTML/JS. During various projects, in which CubeViz was used and had to be deployed to differing IT-infrastructures, we recognized huge limitations that arose due to architectural design decisions. As a consequence, we started to implement CubeViz.js from scratch, as a JavaScript-only client that can be integrated in almost every Web based infrastructure. The CubeViz project was started in 2011 and is still ongoing. Future development iterations will focus on:

- Improving the error tolerance while processing not perfectly designed RDF Data Cubes,
- Integration of additional charts for an improved visualization of multi-dimensional selections.

[16] Project website of Linked Data Cube Explorer: `http://km.aifb.kit.edu/projects/ldcx/`
[17] Website of Linked Statistical Data Analysis: `http://stats.270a.info`
[18] `https://www.r-project.org/`

Modules, integrated into CubeViz, for (a) quality analysis and (b) simultaneous processing of several RDF Data Cubes are removed from CubeViz.js and will be pursued in separate projects.

Acknowledgement

This work was partly supported by the following grants from the German Federal Ministry of Education and Research (BMBF) for the LEDS Project under grant agreement No 03WKCG11C and the European Union Horizon 2020 research and innovation programme for the SlideWiki Project under grant agreement No 688095.

References

[CAR13] Capadisli, Sarven; Auer, Sören; Riedl, Reinhard: Linked statistical data analysis. Semantic Web Challenge, 2013.

[CRT13] Cyganiak, Richard; Reynolds, Dave; Tennison, Jeni: The RDF Data Cube vocabulary. Technical report, W3C, 2013. http://www.w3.org/TR/vocab-data-cube/.

[Da08] Dalgaard, Peter: Introductory statistics with R. Springer Science & Business Media, 2008.

[Ec15] Ecma International: ECMAScript 2015 Language Specification. Geneva, 6th edition, June 2015.

[Fr15] Frischmuth, Philipp; Martin, Michael; Tramp, Sebastian; Riechert, Thomas; Auer, Sören: OntoWiki—An Authoring, Publication and Visualization Interface for the Data Web. Semantic Web Journal, 6(3):215–240, 2015.

[Ka14] Kalampokis, Evangelos; Nikolov, Andriy; Haase, Peter; Cyganiak, Richard; Stasiewicz, Arkadiusz; Karamanou, Areti; Zotou, Maria; Zeginis, Dimitris; Tambouris, Efthimios; Tarabanis, Konstantinos A.: Exploiting Linked Data Cubes with OpenCube Toolkit. In: International Semantic Web Conference (Posters & Demos). volume 1272 of CEUR Workshop Proceedings. CEUR-WS.org, pp. 137–140, 2014.

[KBDM13] Kambona, Kennedy; Boix, Elisa Gonzalez; De Meuter, Wolfgang: An evaluation of reactive programming and promises for structuring collaborative web applications. In: Proceedings of the 7th Workshop on Dynamic Languages and Applications. ACM, p. 3, 2013.

[KH14] Kämpgen, Benedikt; Harth, Andreas: OLAP4LD–A framework for building analysis applications over governmental statistics. In: European Semantic Web Conference. Springer, pp. 389–394, 2014.

[Ma15] Martin, Michael; Abicht, Konrad; Stadler, Claus; Auer, Sören; Ngomo, Axel-C. Ngonga; Soru, Tommaso: CubeViz – Exploration and Visualization of Statistical Linked Data. In: Proceedings of the 24th International Conference on World Wide Web, WWW 2015. 2015.

[Ri01] Ripley, Brian D: The R project in statistical computing. MSOR Connections. The newsletter of the LTSN Maths, Stats & OR Network, 1(1):23–25, 2001.