FP7 Project ASAP
Adaptable Scalable Analytics Platform

D6.1 InfoViz Services Early Design

This report describes methods for collecting and visualizing statistical linked data, including metadata representations and mechanisms to select specific time intervals.

Arno Scharl, Albert Weichselbraun
Alexander Hubmann-Haidvogel, Walter Rafelsberger

webLyzard technology

WP 6 - Information Visualization

Nature: Report
Dissemination: Public
Date: 28 Feb 2015

Acknowledgement

This project has received funding from the European Union's 7th Framework Programme for research, technological development and demonstration under grant agreement number 619706.

Table of Contents

Executive Summary

The main goal of WP6 is the development of high-performance visual interfaces to help analysts navigate big data repositories, in real time and across multiple dimensions (temporal, spatial, or any semantic dimension). Special emphasis is placed on sub-second response times of the interface, and the ability to incorporate metadata as additional context information when analyzing complex relations in Web content (WP8) or telecommunications data (WP9).

In the first year of ASAP, WP6 has focused on methods to collect, represent and visualize statistical data. T6.1 developed methods for the rapid rendering of complex statistical data, using color coding to show metadata attributes as well as flexible mechanism to select appropriate timescales. This included an indexer component to collect, represent and index linked data, as well as a set of visualization components that can be linked together via the ASAP dashboard.

To ensure rapid prototyping of the desired visual methods, T6.1 used the D3 JavaScript library (Bostock et al. 2011).[1] D3 is perfectly suited for the purposes of ASAP, since it is not focused on a new grammar for graphics, but rather on how to integrate existing standards to create effective visualizations.

[1] www.d3js.org

Introduction

Visual Analytics methods are often tailored towards certain goals of users. Yet these goals might be very unspecific, like “finding out information about a certain topic”. To generate knowledge, it is necessary to generate more specific questions that later on can be answered with visual analytics methods. These questions can arise as follow-up questions from free insight generation with a suitable visualization. Consequently, there is a need for a visualization that supports free insight generation without prior modelling of a domain - encompassing both unstructured (Web intelligence) and structured (linked data) sources.

To address the increasing complexity of contextualized information spaces in big data research, a primary goal of information visualization is to show as much information as possible in a limited amount of space. The data density of pixel-based visualizations cannot be surpassed, but might not directly relate to the amount of information conveyed. Shneiderman et al. (1996) present a taxonomy of data types in the context of visualization, including temporal and multivariate data.

ASAP will dynamically combine such data types on the fly – taking into account the specifics of the use cases as well as the users’ current tasks. Time is a particularly important dimension to consider, and a focus of the work conducted in T6.1. We aim for interactive visualizations that (i) reveal complex patterns and evolving trends, (ii) provide flexible mechanism to select appropriate timescales, and (iii) are capable of rendering a considerable amount of data spanning significant timescales - without relying solely on aggregation, which might hide important facts.

Document Structure. The remainder of this deliverable summarizes the work of T6.1, starting with the representation of statistical data and a summary of the envisioned visualization workflow. This conceptual part is followed by a description of the visual analytics building blocks - interactive D3 modules for trend charts, pie charts, and bar charts, to be synchronized within the ASAP dashboard (T6.2), the *Granular Overview Overlay* (GROOVE), a novel way to display aggregated statistical data, and a slider mechanism to select time intervals for the analysis. D6.1 concludes with technical considerations in terms of rendering process, indexing strategy, and rapid deployment in distributed environments.

Representing Statistical Data

To represent the statistical data, we have adopted the RDF (Resource Description Framework) Data Cube Vocabulary (QB), which is the current standard for publishing statistical data. As a W3C Recommendation,[2] it is supported by both industry and

[2] www.w3.org/TR/vocab-data-cube

academia. QB has already gained widespread acceptance judged by the increasing number of statistical datasets published using this vocabulary. QB is based on a cube model that is compatible with the SDMX standard (Statistical Data and Metadata Exchange), designed to be generic and suitable for publishing various types of multi-dimensional datasets.

The basic building blocks of the cube model are measures, dimensions and attributes, collectively referred to as components:

- **Measure components** describe phenomena that are being observed; in most of the cases they are used for measurements - e.g. carbon emissions.
- **Dimension components** specify variables that are important when defining an individual observation for a measurement - e.g., time and space.
- **Attributes** help interpret the measured values by specifying the units of measurement, and additional metadata such as the status of the observation - e.g. estimated.
- **Observations** are the unit elements in a dataset that represent a concrete measurement value for a set of concrete dimension values. They correspond to a value in a statistical database. When the value of a dimension is the same in a large number of observations (for example, the geolocation), it is convenient to group them into a slice.
- The **Data Structure Document** (DSD) describes each dataset and contains all the required namespaces and components. A dataset that contains observations grouped into slices across dimensions constitutes a cube.

By splitting the statistical data into cubes with a maximum of three dimensions - e.g., Gross Domestic Product (GDP), country and time, the QB vocabulary offers a simple structure that is shared by all datasets. To use data from multiple datasets, one needs to determine whether all the components of the respective datasets match.

Processing Workflow

Building on best-practice examples reported in the literature, we propose the following workflow for collecting, processing and visualizing linked data as part of big data applications:

- **Requirements** should include research questions to be explored, example data sources, and the type of visualizations that are appropriate to address the research questions.

- **Discovery and Indicator Selection.** Running SPARQL queries can yield an abundance of data, but to create real value the various dimensions of a dataset often need to be analyzed, aggregated or augmented in order to fit particular visualization scenarios.
- **Ontology Alignment** needs to take into account broken or missing DSDs, SPARQL endpoints failure, and broken dumps which result in parsing errors.
- **Indicator Storage and Retrieval.** Effective indexing strategies based on established platforms such as Elasticsearch[3] or Lucene[4] are essential building blocks of big data applications (see “Index Strategy and Deployment” below).
- **Transformation** includes the specification of data wranglers (scripts that transform data into formats suited for particular visualizations), queries and aggregations. The transformation step can be seen as a first part of the data and view specification (filter, derive) step of Heer and Shneiderman (2012) - although derive tasks can also appear in subsequent steps.
- **Visualization.** The basic building blocks such as line charts, bar charts and pie charts (see “Visualizing Statistical Data” below) tend to be reusable components, which resulted in visualization grammars that can be considered the second part of the data and view specification step (visualize, sort).
- **Interaction.** An interaction layer that includes select operations, zoom, pan, transitions, and synchronization mechanisms is usually built on top of the visualization layer. This level corresponds to the view manipulation step of Heer and Shneiderman (2012) and will be the focus of D1.3 ASAP Dashboard.
- **Reuse** can happen on multiple levels, from indicators to specific charts or the entire platform. It should be integral to the design process, corresponding to the process and provenance step of Heer and Shneiderman (2012).

Indexing of Statistical Linked Data

Any workflow for visualizing statistical Linked Data includes both linked data tasks (selection of indicators, ontology alignment, etc.) and visualization tasks (data wrangling, interaction, etc.). Due to the increased specialization of certain layers - e.g., alignment or interaction, visualizations represent collaborative processes. While visualization taxonomies are often straightforward to adapt to specific workflows, the linked data processing differs from case to case, as statistical linked data is still in the early stages of development.

[3] www.elasticsearch.org
[4] lucene.apache.org

We have experimented with several approaches for the data collection process, including Federated SPARQL (which often resulted in `queryTimeouts`), SPARQL queries or bash scripts (e.g., a combination of `cat` and `grep` commands yields all the URIs that respect a certain pattern) and run them against the dumps collected from the three services. We decided to index the data using ElasticSearch and create a Search API that obtains the data from linked data sources.

The resulting Linked Statistical Data Indexer (LSDI) component is a Python package that provides all the functionality for the three linked data layers - selection of indicators, ontology alignment, indexing and storage. It can parse RDF data dumps from various statistical data publishers such as Eurostat,[5] the European Environment Agency,[6] the World Bank,[7] the World Health Organization,[8] or the International Monetary Fund.[9]

Similar components, e.g. Elasticsearch RDF River[10] and Elasticsearch River OAI,[11] that are implemented as plugins do not perform alignment, focus on harvesting RDF as opposed to RDF Data Cubes, and have SPARQL `queryTimeouts` issues.

These RDF data dumps are generally not published by the institutions themselves, but by research groups from universities such as AKSW Leipzig and DERI Galway, or private companies such as Epimorphics and Eau de Web. Each data dump does contains all the data until the latest update. Updates are performed once per week. While most of the datasets can be queried via SPARQL endpoints, we have used RDF dumps to avoid `queryTimeouts`. As expected, the dumps that we usually ingest from a data source contain three types of files:

- **Data Structure Documents (DSDs).**[12] The documents that describe the schemas for the set of indicators published.
- **Dictionaries.**[13] Code lists for common entity types such as countries and units of measurements.
- **Data Dumps.**[14] RDF of Turtle files containing the statistical observations.

[5] eurostat.linked-statistics.org
[6] semantic.eea.europa.eu
[7] worldbank.270a.info
[8] gho.aksw.org
[9] imf.270a.info
[10] www.github.com/eea/eea.elasticsearch.river.rdf
[11] github.com/jprante/elasticsearch-river-oai
[12] eurostat.linked-statistics.org/dsd
[13] eurostat.linked-statistics.org/dic
[14] eurostat.linked-statistics.org/data

A typical RDF dump for statistical indicators contains all the observations related to that indicator. Our indexer API contains several types of API functions:

- **KB Helper Methods.** Interact with the large knowledge bases that are typically used for interlinking datasets (DBpedia, GeoNames, etc.).
- **Dictionary Methods.** Deal with the dictionaries from particular data sources.
- **Observation Converters.** Convert and align new data sources.

Typically only a single converter is required for each new data publisher (the dumps related to a publisher tend to come from the same source - all World Bank RDF data dumps we ingested, for example, originate from the World Bank 270a dataspace). This approach eliminates the need for SPARQL queries or templates to get the data for various indicators. The URL of the indicator dump to be ingested suffices. In order to ingest multiple indicators from the same source, the API requires a list of these indicators.

Since the URIs from Eurostat and World Bank published in the 270 Linked Spaces are well-defined, the discovery and selection of indicators are straightforward processes. If the indicator name and URI are already known, the indexer will harvest all triples from that location that match the specified criteria - e.g., only data for indicators that correspond to real geographic entities, as compared to derived entities, or only data for the last five years.

A simple process of harvesting triples that match certain criteria would not offer enough information for advanced visualizations. Additional tasks that might need to be performed relate to ontology alignment, for example geospatial alignment where Geonames[15] URIs are used instead of the names of actual locations, avoiding issues such as spelling mistakes, wrong encoding, and unknown name variants. Another example is the alignment of units of measurements, which we have done using the DSDs whenever available.

We have not performed any alignment based on granularity of the temporal data (month, quarter, years), but instead used the following convention: each observation corresponds to a data point in a graphic; the granularity information is added to each observation, and it can be used whenever it is needed - e.g. at query time for complex aggregations. When indexing the data, we have kept all information including the links from the RDF dumps so that any observation or slice can be recreated if needed. Added information like granularity or geonames URI is only used for visualization purposes.

[15] www.geonames.org

The transformation layer contains queries and aggregations needed to retrieve the data for particular visualizations. The indexer described in the previous section already performs many of the tasks usually found in the transformation layer. The visualizations were written in JavaScript using the jQuery[16] and d3.js libraries, following the conventions of the D3 reusable chart pattern. All visualizations are suited for a combined interface to be synchronized using a multiple coordinated views design pattern, which will form the basis for the ASAP dashboard (T6.3).

The rest following section focuses on the visualization layer and the interaction layer. There are strong interdependencies between these two layers, as some interaction types are easier to implement embedded into a specific visualization, as opposed to external modules. An integrated approach also helps us to present the workflow to construct particular visualizations to be embedded into the ASAP dashboard (D6.3).

Visual Analytics Components

This section summarizes the various visualization components of D6.1. While the *trend chart* and *pie chart* represent previous work that has been adapted and applied for the purposes of ASAP, the *bar chart* and *time interval slider* components are new developments. The GROOVE visualization for showing the temporal distribution of large datasets is based on previous work, but has been ported to D3.js and will be made compatible with the ASAP dashboard (D6.2) in the second year of the project - with a focus on the *interaction layer*, including real-time synchronization capabilities with various other dashboard components.

Trend and Pie Charts

These two chart types represent central components of the webLyzard dashboard, already developed in previous projects (see Figure 1). The charts use D3.js to render dynamic transitions. The line chart shows the evolution of topics. Its vertical axis rescales automatically. This feature is particularly useful if a high value obscures the distributions of other variables. Hovering above data points displays tooltips with context-specific metadata. For *Web content analytics* (WP8), the frequency of a topic as well as the sentiment expressed towards this topic will serve as key indicators. Derived metrics such as disagreement (= the standard deviation of sentiment) can show how contested a topic is. The term ‘tsunami’, for example, typically has a low standard deviation since everyone agrees on its negative connotation. For the WP9 use case on *telecommunications data analysis*, the trend chart will enable us to show the number of bookings or enquiries, for example, and to summarize the temporal distribution in conjunction with specific events.

[16] www.jquery.com

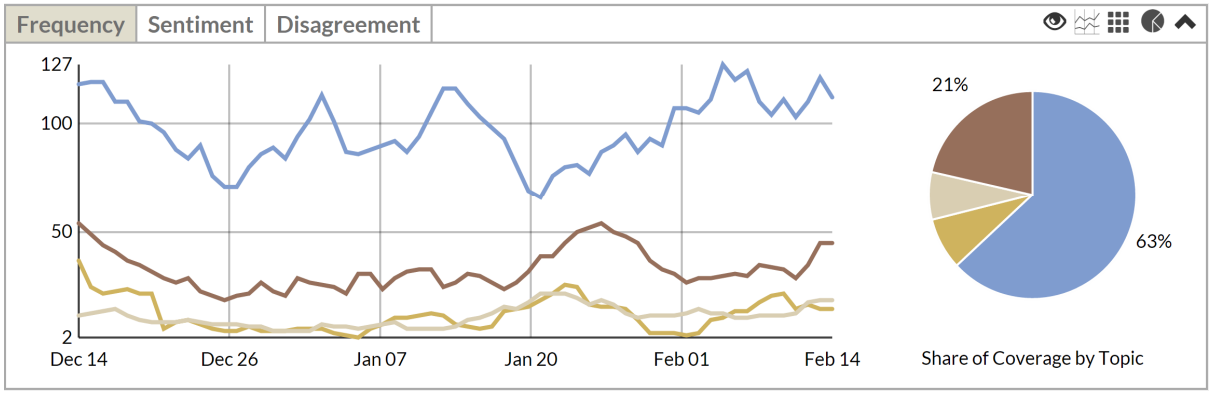

Figure 1. Trend and pie charts of the Media Watch on Climate Change[17]

The data export function of the webLyzard dashboard provides time series data from the trend chart module in XLSX format or as a comma-separated text file (CSV) encoded in UTF-8 format. Specific ASAP extensions to this component include (i) the export of larger datasets with a progress bar, (ii) the ability to start several downloads in parallel, and (iii) the option to cancel an extensive download while the progress bar is being shown. This will allow the processing of large datasets in statistical packages such as R and SPSS, and a range of other external applications.

Bar Chart

To increase the versatility of the display, T6.1 has added the bar chart shown in the left part of Figure 2 to complement the trend chart. This can be seen as a generic extension that allows visualizing any type of grouped datasets – e.g., static information sources with a limited number of daily or weekly updates in the case of the *Web content analytics* of WP8, or different tourist segments in the telecommunications data analysis of WP9.

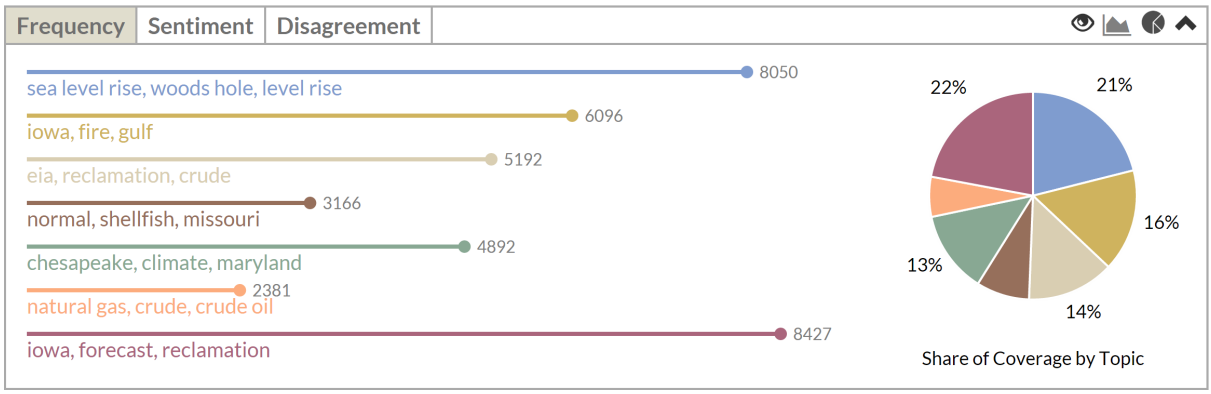

Figure 2. Bar chart to present grouped datasets

[17] www.ecoresearch.net/climate

Recursive Pattern Arrangement for Showing Temporal Views

Keim et al. (1995) propose a recursive pattern arrangement for pixel-based visualizations. One important information source for this arrangement is the calendar aspect of time, i.e. time being composed of multiple granularities such as day, weeks, or months. For pixel-based visualizations of time-oriented data, the method has become the standard arrangement. Two important shortcomings are a lack of user-orientation and the need for focus and context capabilities (Card et al. 1999). For the latter, Shimabukuro et al. (2004) propose the multi-scale visualization. It provides focus and context, but the views are detached and the lack of orientation becomes an even bigger problem due to frequent view shifting that users need to perform. To resolve those issues, Lammarsch et al. (2009) present the GROOVE visualization that provides integrated focus and context based on granularities and the recursive pattern arrangement. D6.1 has adopted the GROOVE approach, implemented it using the D3.js library, and extended the available set of options to show additional metadata dimension – e.g., document sentiment as shown in Figure 3.

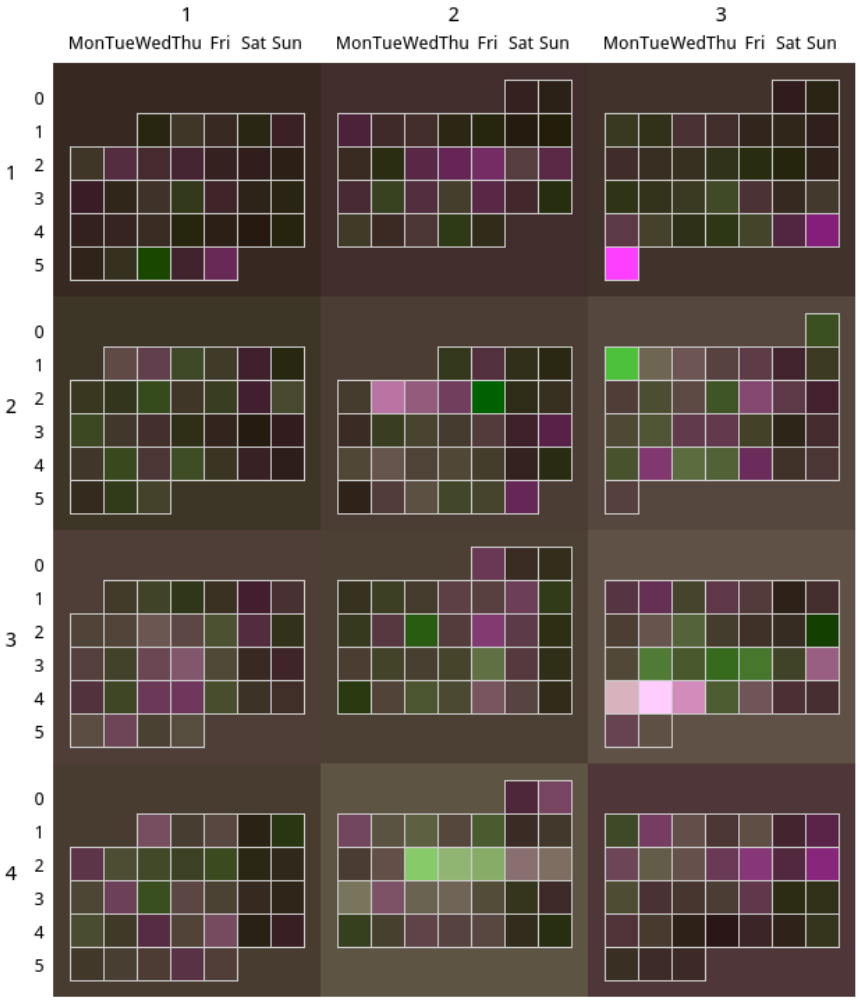

Figure 3. GROOVE representation based on occurrences of the term "climate change" in 2014 Anglo-American news media coverage

Pixel-based visualization originally used one pixel per data element (e.g., the data for one calendar date), using color coding to represent the data value. Newer implementations such as GROOVE represent each data element by a square that might be only the size of one pixel, but automatically expands if more space is available. The pixels, or squares, are arranged according to the days of a year, similar to a calendar. The following example shows the year 2014, positioning the days of each month (each day with a white border) inside a larger rectangle (the larger areas without border). The rectangles are arranged quarterly, so each section labelled with 1-4 on the left side represents one quarter. At the top, the labels of 1-3 represent the first, second, and third month of each of those quarters.

The shown data reflects the frequency of the term “climate change” in Anglo-American news media between January and December 2014. The goal is to show both frequency and sentiment data, which exceeds the possibilities of the original GROOVE implementation. Therefore, we designed a new color coding scheme where lightness refers to the number of daily occurrences, while sentiment is mapped to the hue and chroma of each pixel:

- Pink and magenta colors represent negative bias.
- Green colors represent positive bias.
- Colors closer to gray represent balanced sentiment.

Very strong biases are represented by more vibrant colors, weaker biases are closer to gray, and thus, more pale. This mapping of hues is based on work by Choudhury et al. (2011), which has been extended by a hybrid computation of hue, chroma, and lightness. Monthly averages are mapped to the colors of the rectangles that surround the days and represent the months.

Date Selection

To investigate longitudinal data, a time slider for selecting a date range for the analysis has been developed (see Figure 4), in a joint effort with researchers from the FP7 project Pheme.[18] We plan to activate the new component as part of the webLyzard dashboard in the second quarter of 2015.

The slider element consists of a timespan bar in conjunction with a sliding window. The variation of the point heights forms a line chart which is transformed to a horizon chart (shown with a light grey color) to emphasize minimum and maximum values. The screenshot of Figure 4 illustrates the timespan bar based on randomly generated data. The grid ticks on the horizontal axis provide the temporal context, with labels for years and months.

[18] www.pheme.eu

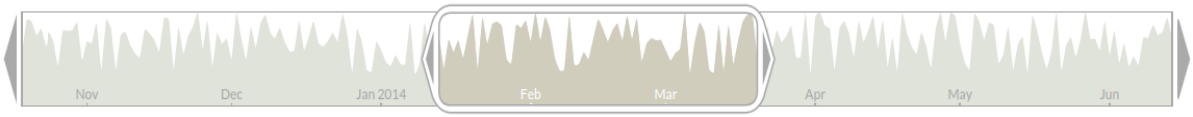

Figure 4. Interactive date range selector

User can drag and drop the sliding window; the current date range will then be updated according to the new position. The two triangular grey-colored handles situated on each side of the window resize the window and adjust the date range. Finally, users can move the selected time span using the triangular arrow buttons located on both sides of the timespan bar - clicking on a button will shift the sliding window by one week towards the side that the arrow is facing.

Technical Considerations

Rendering Process

To assess and improve the rendering and animation performance of the visualization methods, we compared different rendering techniques (SVG, HTML5 Canvas). Both techniques have specific strengths and weaknesses.[19] While *Scalable Vector Graphics* (SVG) is known to have better performance for less elements and larger rendering areas, *Canvas* is superior in the case of more elements and smaller rendering areas.

Since rendering text in SVG is known to be prone to performance issues, we developed a simple test program to render and animate several thousand text elements, using the following techniques:

- D3 (SVG)
- D3 (Canvas)
- D3 (WebGL, Canvas)
- KineticJS (Canvas)[20]
- PixiJS (WebGL, Canvas)[21]

In terms of relative rendering speed we observed the best performance using PixiJS, which is a 2D rendering library utilizing the graphics card by using WebGL and rendering the output on a Canvas element. However, although slight performance improvements could be achieved using PixiJS, we assessed the various implications and decided to keep using D3 and SVG for the following reasons:

[19] www.smus.com/canvas-vs-svg-performance
[20] www.kineticjs.com
[21] www.pixijs.com

- Since SVG works with vector graphics, the visual quality of the output is more appealing compared to pixel-based graphics;
- WebGL is a fairly new technology and therefore might not be supported by all available (mobile) browsers;
- ASAP visualizations are highly interactive, where the major advantage lies with D3 and SVG, due to its data-binding and event handling capabilities;
- Straightforward implementation of animations - when the attributes of the elements to be moved are known, it is straightforward to directly select and animate them in D3. In Canvas, by contrast, all elements need to be redrawn, and one needs to keep track of the elements to be moved, and interpolate their new position (D3 manages these processes automatically).
- Text rendering in GL is not done directly, since a texture element needs to be generated for each text element; this requirement complicates the animation of font size changes as compared to SVG (with respect to text quality and correct positioning);

In light of the potential drawbacks outlined above, we consider the gains in raw performance not significant enough to justify the efforts required to further test and potentially adopt Canvas-based approaches.

Indexing Strategy and Deployment

The rendering of ASAP visualization components requires high-performance queries on unstructured and structured data repositories. In 2014, the Media Watch on Climate Change (which is being used as the knowledge repository for developing the ASAP visualization components) has been migrated from Apache Lucene[22] to Elasticsearch,[23] a distributed search and analytics engine made available under an Apache 2 open source license. It provides a RESTful API using JSON over HTTP. Built for the cloud, Elasticsearch ensures the required scalability for real-time queries that provide the data for through multitenancy and sharded indexing.

Elasticsearch not only speeds up accessing domain-specific repositories of unstructured content, but also facilitates the process of slicing statistical linked data and their integration with other data sources. Elasticsearch aggregations yield additional information for user queries or API calls:

[22] lucene.apache.org
[23] www.elasticsearch.org

- ***top k answers***. *T*op search results based on a simple or advanced query;
- ***data slices***. *M*ost often across time, but in the case of complex datasets other dimensions can be included as well - when analyzing tourism transactions, for example, one can obtain all the flights originating from a specific airport to multiple destinations;
- ***data summaries***. *I*ndicator performance is highlighted not just by color coding, but also through a data summary.

For increasing the flexibility of deploying ASAP visualization components, we are in the process of replacing Kernel-based Virtual Machines (KVMs)[24] with Docker instances.[25] Since the Docker engine container comprises just the application and its dependencies – in contrast to KVM, which also includes the guest operating system, this will increase the portability and efficiency of the developed components. We expect the migration to a distributed Docker architecture to be completed in the second quarter of 2015.

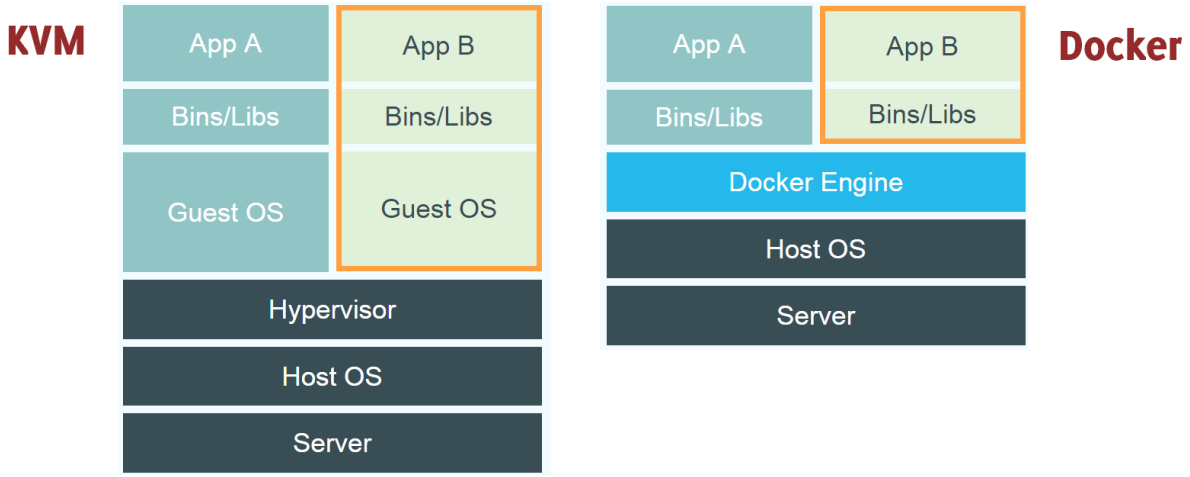

Figure 5. Comparison of the layered architectures of Kernel-based Virtual Machines (left) and Docker containers (right); Source: www.docker.com

Outlook and Next Steps

This deliverable summarizes the work conducted in T6.1, with a focus on extracting statistical indicators from linked data sources (enterprise as well as open), preprocessing and indexing the collected data, and providing real-time query capabilities to generate the data for rendering interactive visualizations.

A first working prototype with a testable dashboard is planned for the second quarter of 2015, allowing users to query for specific indicators and exploring the data with the visual components described in this deliverable. Based on the *Media Watch on*

[24] www.linux-kvm.org
[25] www.docker.com

Climate Change,[26] the primary public showcase of the webLyzard dashboard, the mockup in Figure 6 outlines how the ASAP dashboard of D6.3 will look like (while this example uses only statistics extracted from unstructured document collections, D6.3 will integrate structured indicators from linked data sources as well).

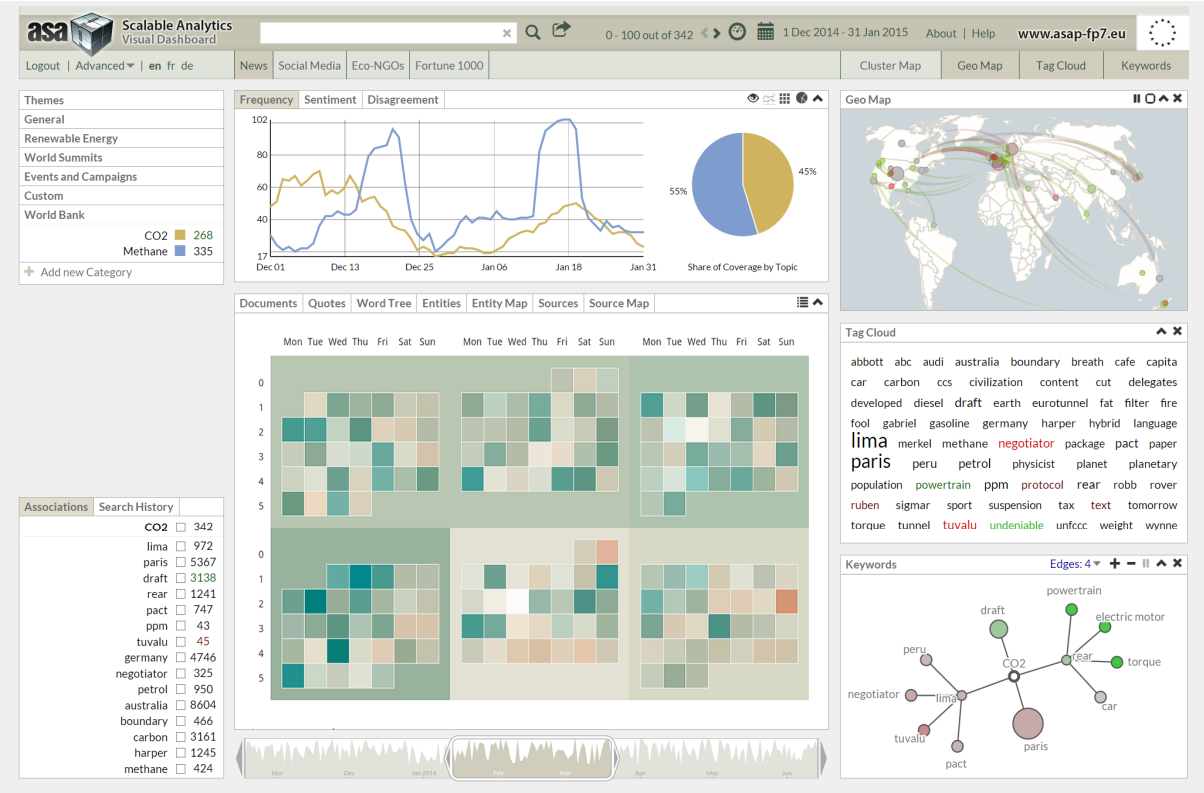

Figure 6. Mockup of integrated dashboard with new D6.1 components

In Year 2 of the ASAP project, D6.2 will focus on geospatial visualizations and data overlays. The existing map component shown in the upper right corner of the above mockup will be extended with the ability to show topographic information underneath the interactive SVG layer, choropleth functionality to render per-country statistical data (e.g., the number of users or the average sentiment expressed towards a country in a tourism context), increased rendering performance using a Web Workers-based approach,[27] additional interactive features to highlight data points and connections, and a revised tooltip to provide per-county metadata.

To prepare for a seamless integration, we will also continue our work on the processing of statistical linked data. The current version of the indexer allows adding multiple indicators from the same source without writing any new mappings, queries or templates.

[26] www.ecoresearch.net/climate
[27] www.w3.org/TR/workers

In the spirit of rapid prototyping, the initial focus was on the general ability to ingest large quantities of data, whereas the next versions will explore two new features: (i) *additional datasets and a simplified configuration* - currently, the indexer requires a list of URIs; we will add the possibility to index all the indicators related to a certain topic from a single source, or across multiple sources; (ii) *export formats* - currently, data can be exported to CSV, MS Excel or diagrams in PNG or SVG format (using the Web interface). Future versions will include the possibility to reconstruct the RDF fragments ingested into Elasticsearch.

References

Bostock, M., Ogievetsky, V. and Heer, J. (2011). “D3: Data-Driven Documents”, *IEEE Transactions on Visualization and Computer Graphics,* 17(12): 2301-2309.

Card, S.K., Mackinlay, J.D. and Shneiderman, B. (1999). *Readings in Information Visualization: Using Vision to Think.* San Francisco: Morgan Kaufmann.

Choudhury, A., Potter, K., Rhyne, T., Livnat, Y., Johnson, C., and Alter, O. (2011). “Visualizing Global Correlation in Large-Scale Molecular Biological Data.”, 1st IEEE Symposium on Biological Data Visualization (BioVis-2011). Providence, USA.

Heer, J. and Shneiderman, B. (2012). “Interactive Dynamics for Visual Analysis”, *Communications of the ACM,* 55(4): 45-54.

Keim, D., Ankerst, M., and Kriegel, H.-P. (1995). "Recursive pattern: A technique for visualizing very large amounts of data", *6th Conference on Visualization* (Vis-95). Atlanta, USA: 279-287.

Lammarsch, T., Aigner, W., Bertone, A., Mayr, E., Gartner, J., Smuc, M. and Miksch, S. (2009). Hierarchical Temporal Patterns and Interactive Aggregated Views for Pixel-based Visualizations. *13th International Conference Information Visualisation.* Barcelona, Spain: 44-50.

Shimabukuro, M., Flores, E., de Oliveira, M., and Levkowitz, H. (2004). Coordinated Views to Assist Exploration of Spatio-Temporal Data: A Case Study. *2nd International Conference on Coordinated and Multiple Views in Exploratory Visualization*, São Paulo, Brazil: 107-117.

Shneiderman, B. (1996). The Eyes Have It: A Task by Data Type Taxonomy for Information Visualizations. *IEEE Symposium on Visual Languages.* Boulder, USA: IEEE Press: 336-343.