DELIVERABLE SUBMISSION SHEET

To:	Susan Fraser	*(Project Officer)*
	EUROPEAN COMMISSION Directorate-General Information Society and Media EUFO 1165A L-2920 Luxembourg	

From:

Project acronym:	PHEME	Project number:	611233
Project manager:	Kalina Bontcheva		
Project coordinator	The University of Sheffield (USFD)		

The following deliverable:

Deliverable title:	Open-Source Visual Analytics Tools: Initial Prototype
Deliverable number:	D5.1.1
Deliverable date:	31 December 2014
Partners responsible:	MOD
Status:	☒ Public ☐ Restricted ☐ Confidential

is now complete. ☒ It is available for your inspection.
☒ Relevant descriptive documents are attached.

The deliverable is:

☐ a document
☐ a Website (URL: ....)
☐ software (...........................)
☐ an event
☒ other (..........Prototype..........)

Sent to Project Officer: *Susan.Fraser@ec.europa.eu*	Sent to functional mail box: *CNECT-ICT-*611233*@ec.europa.eu*	On date: 08 January 2015

FP7-ICT Strategic Targeted Research Project PHEME (No. 611233)
Computing Veracity Across Media, Languages, and Social Networks

D5.1.1 Open-Source Visual Analytics Tools: Initial Prototype

Arno Scharl, Tobi Schäfer, Ruslan Kamolov, Shu Zhu
(MODUL University Vienna)

Abstract

FP7-ICT Strategic Targeted Research Project PHEME (No. 611233)
Deliverable D5.1.1 (WP 5)

This deliverable summarizes the work conducted in WP5 over the first 12 months of the PHEME project. The research activities have focused on requirements analysis, a review of related work, experimental prototyping, and the development of a graph visualization library in conjunction with temporal controls to select appropriate timescales when investigating the evolution of a threaded discussion. The library is compatible with current Web browsers that support *Scalable Vector Graphics (SVG)*.

Keyword list: graph visualization, social network analysis, force-directed layout

Nature: Prototype
Contractual date of delivery: 31 Dec 2014
Actual date of delivery: 07 Jan 2015

Dissemination: PU
Reviewed By: Thierry Declerck

PHEME Consortium

This document is part of the PHEME research project (No. 611233), partially funded by the FP7-ICT Programme.

University of Sheffield
Department of Computer Science
Regent Court, 211 Portobello St.
Sheffield S1 4DP, UK
Tel: +44 114 222 1930
Fax: +44 114 222 1810
Contact person: Kalina Bontcheva
E-mail: K.Bontcheva@dcs.shef.ac.uk

Universitaet des Saarlandes
Language Technology Lab
Campus
D-66041 Saarbrücken
Germany
Contact person: Thierry Declerck
E-mail: declerck@dfki.de

MODUL University Vienna GMBH
Am Kahlenberg 1
1190 Wien
Austria
Contact person: Arno Scharl
E-mail: scharl@modul.ac.at

Ontotext AD
Polygraphia Office Center fl.4,
47A Tsarigradsko Shosse,
Sofia 1504, Bulgaria
Contact person: Georgi Georgiev
E-mail: georgiev@ontotext.com

ATOS Spain SA
Calle de Albarracin 25
28037 Madrid
Spain
Contact person: Tomás Pariente Lobo
E-mail: tomas.parientelobo@atos.net

King's College London
Strand
WC2R 2LS London
United Kingdom
Contact person: Robert Stewart
E-mail: robert.stewart@kcl.ac.uk

iHub Ltd.
NGONG, Road Bishop Magua
Building, 4th floor
00200 Nairobi
Kenya
Contact person: Rob Baker
E-mail: robbaker@ushahidi.com

SwissInfo.ch
Giacomettistrasse 3
3000 Bern
Switzerland
Contact person: Peter Schibli
E-mail: Peter.Schibli@swissinfo.ch

The University of Warwick
Kirby Corner Road
University House
CV4 8UW Coventry
United Kingdom
Contact person: Rob Procter
E-mail: Rob.Procter@warwick.ac.uk

Executive Summary

This deliverable summarizes the work conducted in WP5 over the first 12 months of the PHEME project. The research activities have focused on (i) requirements analysis and a review of related work, (ii) experimental prototyping based on the D3 JavaScript library to visualize clusters of documents and topics to represent emerging stories and threaded conversations (which are identified in T3.3) using a force-directed graph approach, and (iii) development of a flexible and scalable graph visualization library in conjunction with temporal controls to select appropriate timescales and limit the analysis to a particular interval when investigating the evolution of a threaded discussion.

The library is compatible with all current Web browsers that support *Scalable Vector Graphics* (SVG). It uses colour coding to show metadata elements such as the average sentiment towards a specific topic. Various layout optimizations such as a a collision detection component to avoid overlapping labels have considerably improved the usability and aesthetic appeal of the resulting diagrams.

Contents

Introduction

WP5 builds the visual analytics tools to explore the collected veracity intelligence within PHEME, including visualisations of geospatially and semantically referenced information across news media and social networks. The visual analytics tools will allow users to search and browse the automatically extracted veracity intelligence within an interactive dashboard based on multiple coordinated views technology.

The expertise gained from developing and maintaining the *ThreadVis*[1] open-source extension for the Thunderbird email client has provided important guidance for the development process. An adaptation of ThreadViz for the purposes of PHEME would have required a complete rewrite of the library. Given that the focus of ThreadVis on conversations among small groups was not ideal in light of the specific requirements of PHEME (flexible framework that encompasses everything from occasional discussions among a few individuals to large network structures where hundreds of social media users might participate in an active exchange), it has been decided to develop a new open source library from scratch. Work on this new graph visualization library has progressed well; a first public release is planned for the first or second quarter of 2015.

For the information visualization work conducted in Year 1 of PHEME, force-directed placement as outlined in the next section played a central role. It was applied to the on-the-fly identification of document clusters as well to the positioning of nodes in dynamic graph structures. Following an overview of related work, the “Prototype Development” section of this deliverable will describe both applications - including the methodological challenges encountered, and how they have been addressed.

Related Work

Force-Directed Layout Algorithms

Force-directed graphs visualize multidimensional data sets, where a graph consists of nodes connected by edges in two- or three-dimensional space. While such graphs are visually attractive, intuitive to interpret and therefore used in a wide area of applications, the underlying force-directed placement algorithms are resource-intensive compared to other visualization methods - especially in the case of dynamic graph representations (Kobourov 2012). Animated transitions, when appropriately designed, can significantly improve perception at both syntactic and semantic levels of analysis for statistical visualizations (Heer and Robertson 2007). More specific studies show the benefits of animated transitions between time slices for a dynamic graph series (Archambault et al. 2011).

In regards to the visualization of dynamic force-directed graphs, early literature dinstinguished between interactive interfaces and pre-rendered animations, not considering the usability of animated transitions between graph states in interactive environments. More recent work acknowledges the importance of animation and timelines, and their combination tailored to application scenarios (Beck et al. 2014).

[1] threadvis.mozdev.org

The JavaScript based library d3.js offers a customizable force-directed graph layout implementation and features a physical simulation and iterative constraint relaxation (Bostock et al. 2011). Nonetheless, it iterates on the principles of early contributions to the field. The strengths of the library lie in its simple configuration and extensibility, even allowing the adaptation of alternate graph layout algorithms like WebCoLa,[2] a library for arranging layouts using constraint-based optimization techniques.

Social Network Analysis

Performing Social Network Analysis (SNA) on-the-fly, based on the documents returned by a certain search query such as “climate” and “barack obama”, supports the exploration of social relationships among entities - participants of an online conversation, authors of related documents, retweet patterns, etc. A graph can be constructed based on document similarity, for example, where the nodes represent the entities and the edge weights represent the document similarities between two nodes. Depending on the specific characteristics and quality of data, the following analyses could be applied to the graph:

- Measures of *centrality* at node level (betweenness, eigenvector, degree, etc.)
- Similarly, measures of *exceptionality* such as brokerage (Burt’s effective size)
- *Clustering* and *k-core analysis* for providing structural information.
- *Link centrality* for determining whether any ties, associations or channels represent key pathways for propagating information

In order to guide and facilitate the developing process, we conducted a survey of open source libraries for graph analytic libraries and graph visualization libraries. For the graph analytic libraries, we focused on libraries written in Java and R for reasons of compatibility with the existing platform (Java) and the libraries’ outstanding analytical and statistical abilities (R). The shortlist of third-party libraries based on their functionality and popularity in the developer forums included:

- Statnet (R and ANSI standard C) - statnet.csde.washington.edu
- Igraph (R) - cran.r-project.org/web/packages/igraph/igraph.pdf
- JGraphT (Java) - jgrapht.org
- JUNG (Java) - jung.sourceforge.net

A direct comparison of igraph and statnet shows that the former is more straightforward to deploy, but lacks some of statnet’s more complex functionality. igraph contains basic network analysis functionality, including the calculation of network properties, generating random graphs for simulations, etc. JGraphT and JUNG are the two most popular open source Java libraries for graphs. The portfolio of JUNG is more extensive as compared to JGraphT - yet both contain algorithms that the other library do not provide. JGraphT has a stronger focus on the data structure aspect for optimized representations (including automatic indexing via custom listeners), and the creation of sophisticated views into existing graphs - e.g. induced subgraphs. JUNG does not provide an explicit subgraph algorithm, but includes layout and centrality algorithms that do not exist in JGraphT.

[2] marvl.infotech.monash.edu/webcola

R includes plot functions to generate static graphs, including a package to create Graphics Interchange Format (GIF) animations. This is well suited to show the temporal dynamics of a network with igraph and statnet. JUNG has visualization capabilities for illustrating various features including painting, labeling, layout, zooming, and user interaction. The advantage of visualization capabilities on the server side is the generation of a graph preview as a blueprint for the more sophisticated designs on the client side. On the client side, the visualization can then be done based on Javascript, using on of the following graph visualization libraries:

- D3Network - christophergandrud.github.io/d3Network
- Visjs - visjs.org
- Sigmajs - sigmajs.org

In terms of graph visualization, none of the evaluated libraries fulfilled the requirements of PHEME, but nevertheless the comparison of various approaches yielded important insights and helped to guide the development activities in WP5. The assessment of the libraries' analytic functions in regards to modeling the structure and evolution of social networks is still ongoing and will be concluded in the first quarter of 2015.

Prototype Development

To provide the data required for developing and evaluating the visualization prototypes, we have used the Media Watch on Climate Change (MWCC)[3] – a continuously updated repository on climate change and related environmental issues. The system collects and filters online documents from a wide range of sources including news media, social networking platforms, the websites of Fortune 1000 companies and environmental organizations, etc.

To demonstrate the capabilities of the developed technologies and enable public access (complementing the testing interface outlined in the following sections), they have been integrated as "Cluster Map" and "Keyword Graph" components into the multiple coordinated view mechanism of the MWCC portal.

Document Cluster Visualization

The first implementation using the force-directed placement algorithm was a cluster visualization to group search results by topic, arranging documents by their semantic similarity using two alternative methods: (i) K-means, which divides the collection of documents into a fixed amount of clusters where each document belongs to the cluster with the nearest centroid, and (ii) agglomerative hierarchical clustering, a deterministic approach that pairs clusters into a tree-like structure.

The visual display shown in Figure 1 highlights groups of similar documents by a convex hull shape that visually holds its nodes together. The size of this shape is dynamic and depends on the number of contained nodes. Each node represents a document returned by the search function, shown as a circle shape of variable size and color. While node size is proportional to the reach of the document's media source, node color reflects normalized document sentiment. Three keywords per cluster are used as a label to describe its contents. The hull shapes of nodes and clusters are

[3] www.ecoresearch.net/climate

rendered with reduced opacity to decrease the visual load and increase the readability of cluster labels. These labels are extracted from the ordered list of all document keywords within the cluster, considering the reach of the documents' sources.

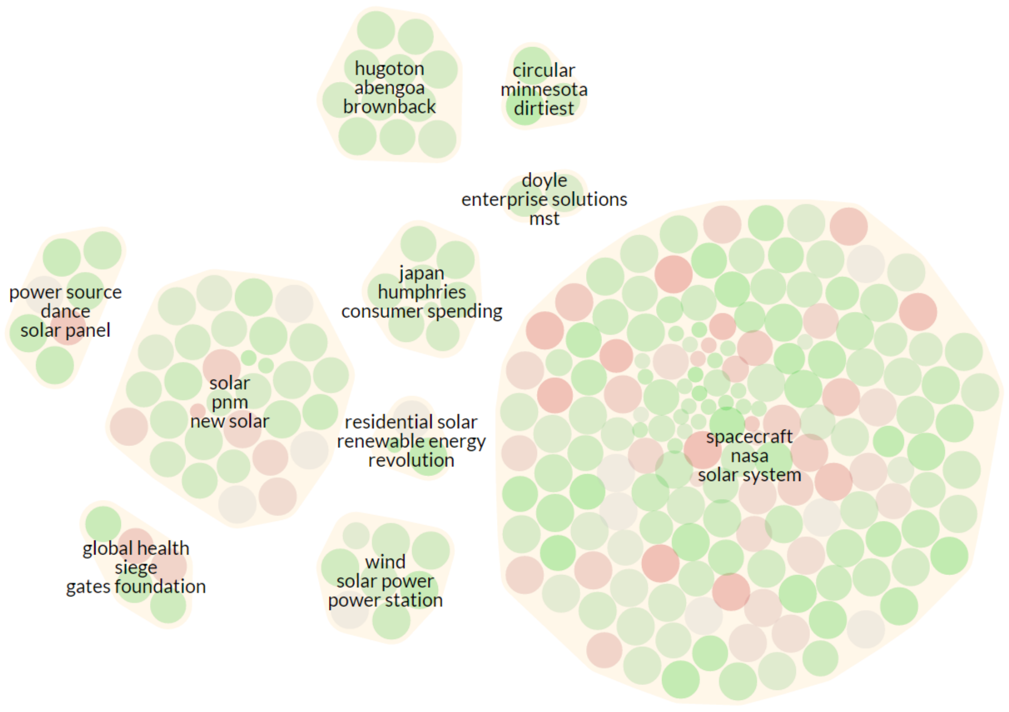


Figure 1. Document clusters based on the search query "solar energy"

Graph Visualization

Moving beyond simple document clusters, the second application of the force-directed layout algorithm focused on the visualization of graphs as a flexible method to show not only semantic associations among extracted topics, but also shed light on the structure of a story or conversation on both the micro and macro level.

To visualize the relations between the strongest associations (keywords) detected in a search query, the MWCC portal outlined above provides an undirected graph. This graph consists of vertices (nodes) – representing the keywords – and edges (connections) – representing the relations between the nodes. At the graph's center resides the original term of the query, the root node, which the associations were evaluated for.

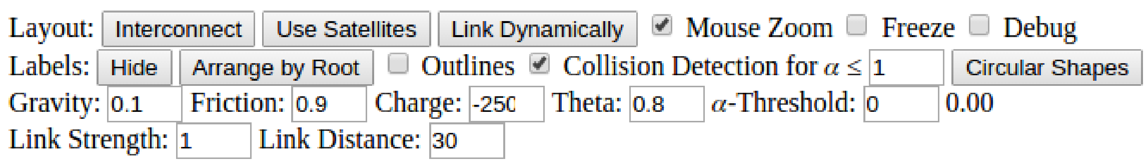
Node size reflects the term's frequency of occurrence, while the color represents its average sentiment. Each node is textually labelled with its corresponding term. The lengths of the edges (i.e., the distances between nodes) are dynamically calculated by multiple algorithms:

- Force-directed layout
- Collision detection between the text labels of the nodes, based on quadtrees
- Automatic scaling for fitting the graph to the available window size
- Radial positioning of the nodes according to their distance from the root node

Interactive Controls

To develop and test the various configuration settings and experimental modifications, the interactive controls shown in Figure 2 allow modifying the following parameters:

- Switch between single (tree) or multiple paths (interconnected) between nodes
- Switch between using non-semantic helper nodes on the window edges ("satellites") and automatic scaling of the graph's bounding box to fit the graph in the visible area
- Switch between static and dynamic edge lengths
- Toggle zooming in and out using the mouse wheel
- Toggle the application of the force-directed algorithm ("freeze")
- Toggle debug mode - i.e., log messages and visual helpers
- Switch between visible and hidden labels
- Switch between arranging the labels by parent or by root node
- Toggle the label outlines (i.e. the white background between label and graph)
- Toggle collision detection with a custom alpha value as threshold (i.e. delay)
- Switch between the various shapes available for collision detection
- Set the general values for the force-directed layout as defined by d3.js (gravity, friction, charge, theta and alpha, link strength and link distance)



Layout: ✓ Mouse Zoom ☐ Freeze ☐ Debug
Labels: ☐ Outlines ✓ Collision Detection for $\alpha \leq$ 1
Gravity: 0.1 Friction: 0.9 Charge: -250 Theta: 0.8 α-Threshold: 0 0.00
Link Strength: 1 Link Distance: 30

Figure 2. Control settings to access the API features of the graph visualization library

Node and Label Overlap Removal

Overlap in graphs heavily affects readability and clarity of the visualization. One approach to mitigate such overlap is the use of collision detection algorithms. The PHEME component demonstrated in Figure 3 supports overlap removal based on different bounding shapes including rectangles, circles and custom polygons based on the size of the corresponding node and label. The collision detection based on custom polygons uses Sat-js,[4] a JavaScript library using the separating axis theorem.

The calculation of polygons and their potential collisions results in a bottleneck during an already costly computation of the visualization and requires additional performance improvements in Y2. Additionally, support for ellipsoid bounding shapes is planned.

[4] jriecken.github.io/sat-js

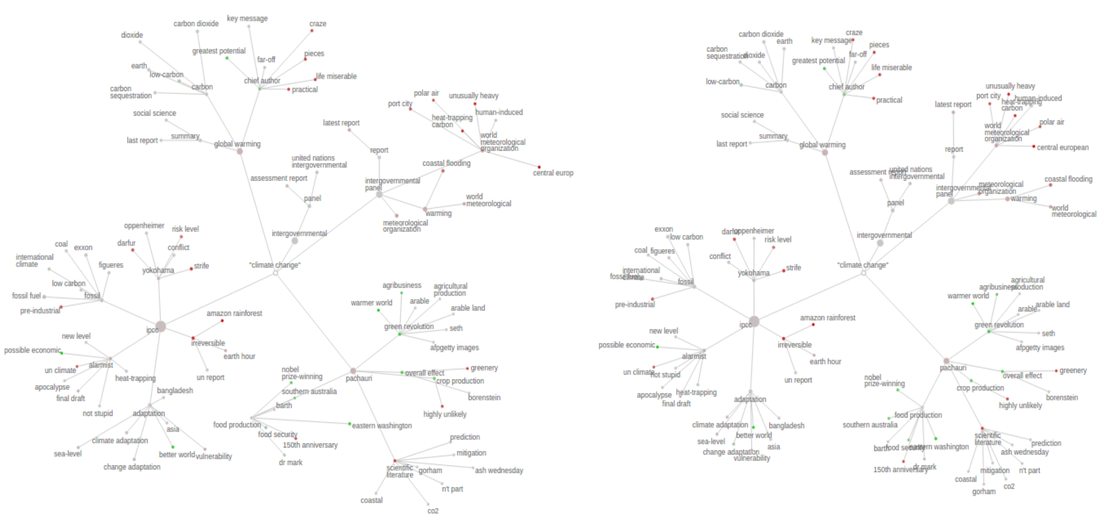


Figure 3. Comparison of initial keyword graph with overlapping labels (left) and the optimized layout using collision detection (right)

Scalability and Optimization

To improve performance, the component makes use of the d3.js *quadtree* implementation to reduce the amount label-to-label checks required for detecting collisions. The screenshot in Figure 4 shows part of an evolving graph with circular boundaries of node-label pairs (red), quadtree elements used for collision detection (blue), and the graph's maximum bounding box (green). The various elements are displayed dynamically within the evaluation interface to support "visual debugging".

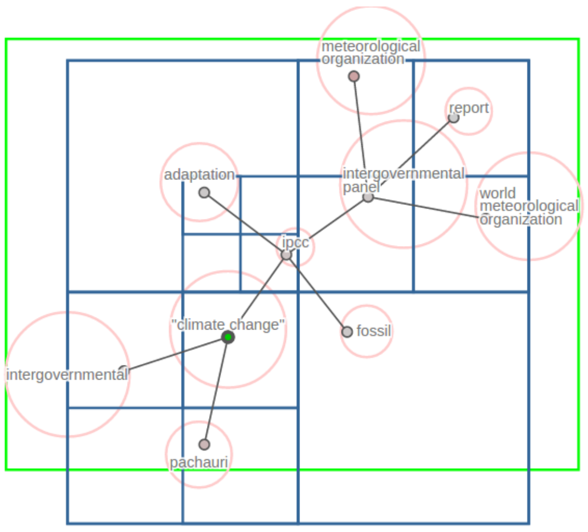


Figure 4. Collision detection using the d3.js quadtree implementation

Another way to address the performance issue is to move the calculation of the force-directed layout into a (non-shared) web worker component[5] - i.e., making it run in an extra thread of the browser. Thereby, the main thread has access to more resources to render the actual graph. While the layout calculation is running in the thread, the visualization requests the layout data periodically and updates itself accordingly.

[5] www.w3.org/TR/workers

Layout Optimization

Jittering may occur when competing forces - e.g. force directed layout vs. collision detection - work against each other. To reduce such jittering in the layout process, the library supports customizable delays and durations for a fine-grained control of the various layout steps. Using dynamically growing bounding shapes, for example, can prevent sudden movements when initializing the graph layout.

The latest version of the component, using a color design with dark edges and lighter labels with a white outline, can be accessed using the "Keyword Graph" feature of the MWCC portal. The current design tries to find a balance between putting the visual focus on the graph's main feature being the nodes and corresponding edges while retaining label readability.

In the example shown in Figure 5, the color of the nodes reflect normalized term sentiment, ranging from red (negative) to grey (neutral) and green (positive). Sentiment is shown with variable saturation, depending on the degree of polarity - vivid colors indicate emotional articles, lower saturation a more factual coverage.

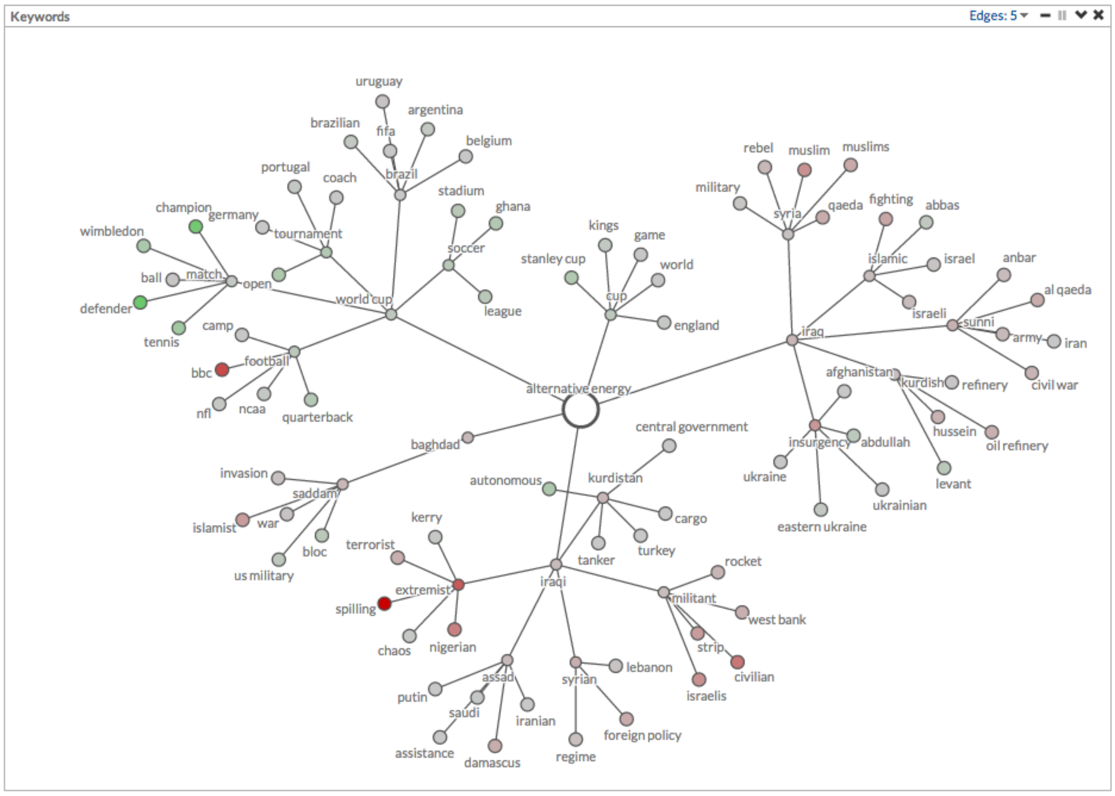


Figure 5. Keyword Graph for a query on "alternative energy"

Evolving Stories and Conversations

For performance reasons the current approach to model dynamic graph transitions includes the following three processing steps:

1. Obsolete nodes are removed from the graph
2. New nodes are added to the graph
3. Remaining nodes are updated

From a visual perspective, new nodes are added by initially placing them nearby and around their parent, i.e. the originating node, then – together with the remaining nodes – arranging them again according to the force-directed placement and collision detection algorithms.

Combined with the date range selector outline below, this provides an intuitive way to depict stories and conversations – i.e. to visualize the communication flows in a network of actors. Just as new nodes appear and disappear over time, the participants of the conversation join and leave. A new edge connecting two nodes shows how a dialog starts. On an aggregated level, the dynamic pattern of new and reinforced edges identifies opinion leaders and shows how the focus of activity moves among the various participants of the conversation.

Temporal Controls

To investigate evolving stories and conversations, an interactive controller has been developed to facilitate the selection of data. It consists of a timespan bar in conjunction with a sliding window. The timespan bar displays daily amounts of documents from the entire dataset, starting from the first day on the left and incrementing to the right side until the current day. Each data point represents a weekly average. The amount of documents is shown with the height of the point on the vertical axis. 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 peaks and lows in the daily document frequency. Figure 6 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.

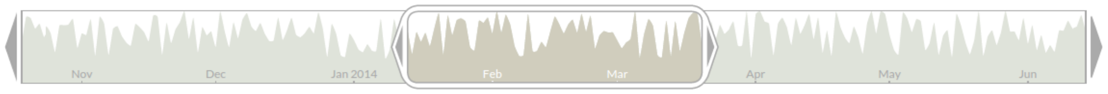


Figure 6. Interactive date range selector

The sliding window consists of a white shape with a grey outline. It is movable along the bar and defines the current date range: the left boundary represents the starting date, the right boundary the ending date. The selected range from the timespan bar is highlighted with a pale brown color. The grid ticks that fall into the selected range are also contrasted.

User can move the sliding window by pointing the cursor over it and performing the dragging action. When the dragging ends, the current date range will be updated accordingly to the new position of the window. The two triangular grey-coloured handles situated on each side of the window resize the window and therefore adjust the boundaries of the date range. This action can be performed by pointing the cursor on a handle and dragging it to the desired position. Once the handle is released, the date range is updated accordingly. 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 to the side the arrow is facing.

Advanced Graph State Transitions

A challenge for time-based graph visualization is how to implement transitions from one graph state to the next. A common technique to shorten the wait for the user is to show the layout process and update the appearance of the graph on every calculated iteration. While in its organic look watching the nodes' movement may be fascinating by itself, the process adds little value to display and might result in temporal visual clutter (Tufte 1983). This gets more problematic (i) in the case of large graph, (ii) if attributes of the graph change through updates in interactive environments, or (iii) when visualizing evolving graphs. When movement of layout iterations are mixed with the removal or addition of nodes and edges in complex graphs, these changes cannot be perceived appropriately by the user because of the simultaneously happening events (Tversky et al. 2002).

The PHEME prototype aims to mitigate these problems when using animated transitions in force-directed graphs to enhance understanding of network evolutions. It is an approach to overcome the often contradictory results of graph readability and mental map preservation as described by Kobourov (2012). To minimize visual temporal visual clutter introduced by the display of the layout algorithm, the animation uses the following workflow:

- Internally, the states of the initial graph layout and the next are compared.
- The graph retains its overall position, but nodes and edges that are not part of the next graph are faded out and removed.
- Without showing the iterations of the layout algorithm, the updated node positions are calculated in the background.
- The shared nodes from both graphs move to their new locations using animated transitions.
- Once this process is complete, the new nodes fade in.

The consecutive character of these steps gives the user a better way to recognize and understand changes in the structure of the dynamic graph.

Summary and Outlook

This deliverable summarizes the work in WP5 of PHEME in the first 12 months, with a special emphasis on experimental prototyping using the D3 JavaScript library (Bostock et al., 2011) to visualize clusters of documents and topics to represent emerging stories and threaded conversations using a force-directed graph approach. Work on a new graph visualization library has progressed well. A showcase of its capabilities is available through the "Keyword Graph" component of the Media Watch on Climate Change.[6] The release of the open source library itself is planned for the first or second quarter of 2015, integrating various components that have been developed separately. To encourage uptake and reuse, special emphasis has been placed upon compatibility of the library with all current Web browsers that support *Scalable Vector Graphics* (SVG).

The further evolution of the component will increase the scope of its applicability - moving beyond keyword relations based on co-occurrence patterns in large text corpora. Specifically, we aim to visualize (i) relations among referenced named entities including people, organizations and locations, and (ii) relations among the authors of Web documents and social media postings including descriptive metrics from Social Network Analysis (SNA).

[6] www.ecoresearch.net/climate

References

Tufte, E.R. (1983). *The Visual Display of Quantitative Information.* Cheshire: Graphics Press.

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

Kobourov S.G. (2012), “Spring embedders and force directed graph drawing algorithms”, *Computing Research Repository* (CoRR). http://arxiv.org/abs/1201.3011

Beck F., Burch M., Diehl S. et al. (2014), “The State of the Art in Visualizing Dynamic Graphs”, *Eurographics Conference on Visualization* (EuroVis-2014). Swansea, UK.

Tversky B., Julie B.M.Y, Betrancourt M. (2002) “Animation: Can it facilitate”, *International Journal of Human-Computer Studies,* 57, 247–262.

Heer J., Robertson G. (2007), “Animated transitions in statistical data graphics”, *IEEE Transactions on Visualization and Computer Graphics,* 13(6), 1240–1247.

Archambault D., Purchase H.C., Pinaud B. (2011), “Animation, small multiples, and the effect of mental map preservation in dynamic graphs”, *IEEE Transactions on Visualization and Computer Graphics*, 17(4), 539–552.

Green I. (2012), *Web Workers - Multithreaded Programs in JavaScript.* Sebastopol: O’Reilly Media.