



Deliverable D6.4.1

Demonstrator Front-End

Editor:	Zhixing Li, THU
Author(s):	Yixin Cao, THU; Zhixing Li, THU; Ruibing Yang, THU, Peng Zhang, THU.
Deliverable Nature:	Report
Dissemination Level: (Confidentiality)	Public (PU)
Contractual Delivery Date:	M24
Actual Delivery Date:	M24
Suggested Readers:	Developers creating software components to be integrated, developers creating case study prototypes, software developers.
Version:	1.0
Keywords:	toolkit; visualization; components;

Disclaimer

This document contains material, which is the copyright of certain XLike consortium parties, and may not be reproduced or copied without permission.

All XLike consortium parties have agreed to full publication of this document.

The commercial use of any information contained in this document may require a license from the proprietor of that information.

Neither the XLike consortium as a whole, nor a certain party of the XLike consortium warrant that the information contained in this document is capable of use, or that use of the information is free from risk, and accept no liability for loss or damage suffered by any person using this information.

Full Project Title:	Cross-lingual Knowledge Extraction
Short Project Title:	XLike
Number and Title of Work package:	WP6 – Integration and Toolkit
Document Title:	D6.4.1 – Demonstrator Front-End
Editor (Name, Affiliation)	
Work package Leader (Name, affiliation)	Esteban García-Cuesta, iSOCO
Estimation of PM spent on the deliverable:	9PM

Copyright notice

© 2012-2014 Participants in project XLike

Executive Summary

This deliverable documents the toolkit for visualization of diversified data in news articles, social media and other resources. Four components, independent of each other, are integrated in the toolkit. By providing a formalized interface, the goal of toolkit is to help people understand abstract statistical data more efficient in a visualized way.

According to the diversity of data, the toolkit provides four kinds of different visualization scheme. First, Entity graph, one of the toolkit components, is designed for displaying instances and the complex relationships among them. Secondly, Radar chart focuses on different features of instances, which could be celebrities or devices. Finally, the other two, Timeline and Time-Geo visualization, is designed to visualize dynamic data.

All the components are programmed in JavaScript, which allows for high flexibility and scalability.

Table of Contents

Executive Summary	3
Table of Contents	4
List of Figures.....	5
List of Tables.....	6
Abbreviations.....	7
Definitions	8
1 Introduction	9
2 Visualization Toolkit	10
2.1 Entity Graph	10
2.1.1 Functionality.....	10
2.1.2 Data Format	11
2.1.3 Interactions	12
2.2 Radar Chart	15
2.2.1 Functionality.....	15
2.2.2 Data format	16
2.2.3 Interactions	17
2.3 Timeline.....	17
2.3.1 Functionality.....	17
2.3.2 Data format	19
2.3.3 Interactions	19
2.4 Time-Geo Visualization	19
2.4.1 Functionality.....	20
2.4.2 Data Format	20
2.4.3 Interactions	21
3 Library and Services.....	22
3.1 Node.js	22
3.2 D3.js.....	22
3.3 Google Map.....	22
4 Conclusions	23
5 References.....	24

List of Figures

Figure 1. A Demo of Entity Graph.....	10
Figure 2. Data Format for Entity Graph.....	11
Figure 3. Data Format for Relations.	12
Figure 4. A Demo of Entity Graph.....	13
Figure 5. Layout Configurations.....	13
Figure 6. Weights Scale.....	14
Figure 7. Color (Entity Category) Filter.....	14
Figure 8. Highlight in Entity Graph	15
Figure 9. A Demo of Radar Chart.....	16
Figure 10. Data Format for Radar Chart.....	16
Figure 11. Highlight in Radar Chart.....	17
Figure 12. A Demo of Timeline.	18
Figure 13. Data Format for Timeline.	19
Figure 14. A Demo of Time-Geo Control.	20
Figure 15. Example Data for Time-Geo Control.	21

List of Tables

Table 1. Data Fields for Entity Graph.....	12
Table 2. Data Fields of Relation Data.	12
Table 3. Meaning of Data Fields for Radar Chart.	16
Table 4. Meaning of Data Fields for Timeline.....	19
Table 5. Meaning of Data Fields for Time-Geo Control.....	21

Abbreviations

API	Application Programming Interface
D	Deliverable
SOA	Service Oriented Architecture
T	Task
URL	Uniform Resource Locator
WP	Work Package

Definitions

Pipeline Refers to the flux of different processes which are applied to a set of raw data in order to analyse it and interpret it. In XLike project. It covers the following phases: gathering data, pre-processing data, application of Natural Language Processing Tools, semantic interpretation, visualization, and finally domain interpretation

1 Introduction

This report documents the toolkit for visualization of diversified data in news articles, social media and other resources. Four components, independent of each other, are integrated in the toolkit. Each component provides a formalized interface to structured data, and displays them in an efficient visualized way. This may reveal the internal connection of instances included in news articles or other texts online, which is difficult to identify from data itself. So the goal of toolkit is to help people understand those articles more efficiently and directly.

According to the diversity of data, the toolkit provides four kinds of different visualization scheme. First, Entity graph, one of the toolkit components, is designed for displaying instances and the complex relationships among them. Also, the length and thickness of edges could be justified automatically w.r.t the strength of relationship based on Potential Field Method. Secondly, Radar chart focuses on the comparing between different instances in various dimensions. Finally, the other two, Timeline and Time-Geo control, present news in different distribution of time and space respectively.

Using D3 JavaScript library and google chart, all the components are programmed in JavaScript, which allows for high flexibility and scalability. For the convenience of easy integration of these controls in target systems, formalized interfaces are defined in all the controls.

This deliverable is structured as follows. Section two details these controls with the aspects of function, data format and interactions. Section three provides the depended libraries or services such as D3.js, Google chart and Google map.

2 Visualization Toolkit

This toolkit contains four independent components developed with Google V8 JavaScript engine [1]. This allows for easy entry level from programming point of view, high flexibility (since it's a scripting language) and scalability. Moreover, Node.js [2], a software platform that allows for running a web server without the use of external software, is used when develop this prototype. Therefore, all these controls could be deployed simply through a JavaScript API with any popular browser.

In this section, each control will be introduced in detail with aspects of functionalities, data format and interactions.

2.1 Entity Graph

2.1.1 Functionality

Entity graph is designed for displaying instances and the complex relationships among them. Particularly, nodes and edges of Entity graph denote instances and their relationships respectively. In addition, the length and thickness of edges could be justified automatically with respect to its strength of relationship based on Potential Field Method. That is to say, by setting appropriate weights scale, instances with strong connections will cluster together, those have weak or no connections will depart away from each other, which is helpful to understand the internal relation of instances directly.

There are four major functionalities that should be noticed as follows.

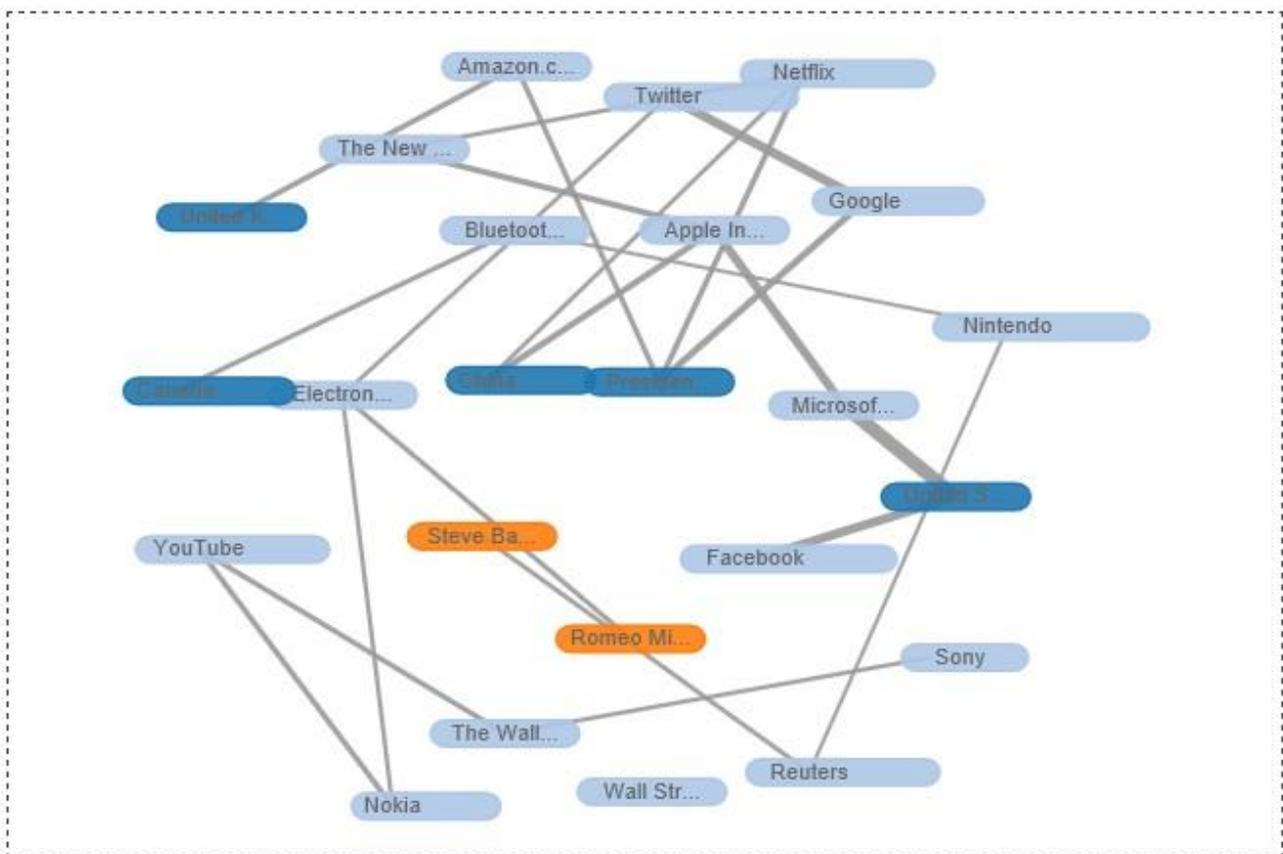


Figure 1. A Demo of Entity Graph

➤ Classification of instances

As is shown in Figure 1. A Demo, each node of Entity graph is filled with one kind of color. In this case, both nodes with label of *Google* and *Twitter* are filled with color of light blue, which means that they belong to

the same category in this sample. On the contrary, the *China* node has the different color of dark blue, for it's not a company but a location.

Also, user can increase/decrease the number of color for more/less categories of instances, and the color can be set as you wish.

➤ Weights of edges

To visualize the connections between entities, Entity graph use edges with weights to link relevant nodes. Particularly, short/long edges with more/less weights signifying strong/weak connections, which can be adjusted automatically according to the current weight scale based on Potential Field Method. Thus, nodes can be clustered into groups to imply their relationship.

➤ Filter

The Entity Graph also provides functions to filter entities/relations/weights to remove irrelevant node pairs or unimportant nodes.

There are three filters to simplify this graph.

- Entity filter: Each entity (node of Entity graph) has a score to imply its significance, which will be introduced in detail in 2.1.2 Data Format. A threshold can be set to filter out any entity with a score below it, and all the entities will be shown by default.
- Relation filter: Each relation (edge of Entity graph) has a score to imply its significance, which will be introduced in detail in 2.1.2 Data Format. A threshold can be set to filter out any relation with a score below it, and all the relations will be shown by default.
- Classification filter: Each instance (node of Entity graph) belongs to a classification in the initial setting, which will be introduced in detail in 2.1.2 Data Format, either. As states in the previous section, the colors of nodes denote different classifications of instances. By cancelling a color, the matching nodes will disappear correspondingly.

➤ Movability of nodes

User can change the position of node. However, it should be noticed that other nodes will be motivated to move simultaneously until the forces in virtual potential field converge back to balance. This might help user to analyze data as groups more conveniently.

2.1.2 Data Format

Using Google V8 JavaScript engine, this component is programmed based on Node.js, a platform that allows for running a web server without the use of external software. Besides, D3.js [3], as a dependent library, provides support through a JavaScript API to draw the Entity graph.

All the necessary libraries and web services will be detailed in Section 3 .

```
{
  uri: "pm_org:http://en.wikipedia.org/wiki/Microsoft",
  id: "23605",
  type: "pm_org",
  label: "Microsoft",
  label_eng: "Microsoft",
  label_deu: "Microsoft",
  label_spa: "Microsoft",
  label_zho: "微软",
  label_slv: "Microsoft",
  score: 79
},
```

Figure 2. Data Format for Entity Graph.

To deploy entity graph in target system, user must provide a list of entities which satisfy the predefined format. **Error! Reference source not found.** shows an example entity, of which the data fields' meaning can be found in Table 1.

Table 1. Data Fields for Entity Graph.

url	Source url of entity
id	entity id
Type	Classification of entity
label	Node's text shown in the graph
Label_eng	Node's label in English
Label_deu	Node's label in German
Label_spa	Node's label in Spanish
Label_zho	Node's label in Chinese
Label_slv	Node's label in Slovenian
score	Weight of entity

Except an entity list, each relation, edge of the graph, requires a standard input data format. Figure shows an example of relation data structure that links two nodes: id of 233 and 23605.

```
{
  Id1: "233",
  Id2: "23605",
  score: 2605
},
```

Figure 3. Data Format for Relations.

Table 2 details the meanings of parameters of relation.

Table 2. Data Fields of Relation Data.

Id1	Id of the head of the edge
Id2	Id of the tail node of the edge
score	Weight of edge

2.1.3 Interactions

The main user interface is shown in Figure . The entire page is split into two area vertically. The left part shows the entity graph with the current configuration, and the right part is the functionality panel with filters and settings of necessary.

To satisfy the primary requirements of general users, there are several kinds of interactions designed in this prototype. In this subsection, they will be introduced by the method of giving examples.

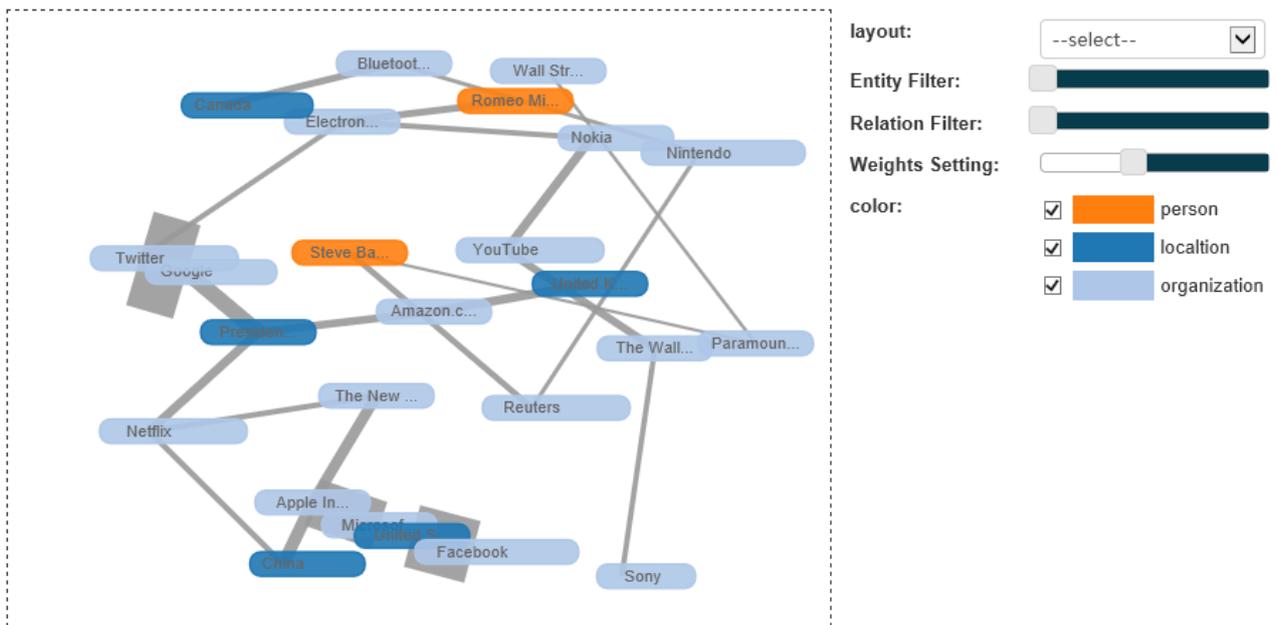


Figure 4. A Demo of Entity Graph.

➤ Layout

Layout item lists on the top of the functionality panel. When user clicks it (the blank with a text “select”), a drop-down menu pops up. For convenience, four common kinds of configuration for different browsing modes are provided here. Thus, user can select the best one.

As shown in Figure , in the drop-down menu, user can choose:

- Select: displaying all the nodes and edges, which is the default option when initialize.
- Default: a recommended choice that balance the weights of both entities and relations.
- Tight: High relation entities will be pulled closely.
- Normal: Low rank edges and entities will be filtered
- Simple: Only the most important nodes and edges will be shown.

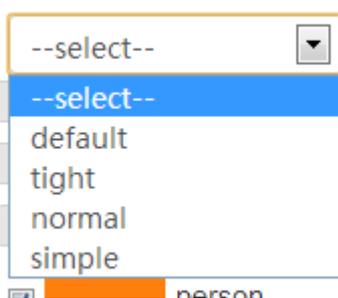


Figure 5. Layout Configurations

➤ Entity filter

This filter utilizes a slide to control the entity importance threshold which can be changed simply by dragging the button. At default, the minimum of this parameter is 0 (nothing will be filtered), and the maximum is set up automatically according to all the scores of entities when initialize.

➤ Relation filter

This filter utilizes a slide to control the threshold of edges. At default, the minimum of this parameter is 0 (nothing will be filtered), and the maximum is set up automatically according to all the scores of relations when initialize.

➤ Weights setting

The slide controls the influence of relations on the length/width of edges. With higher weights, the influence of relationship between nodes on the length/width will be multiplied. For example, by dragging the button, we increase the weight scale a little bit. Consequently, all the nodes cluster into two groups distinctly, and inside the upper group there are six subgroups that connect each other with different edges according to their strength of relations. Figure 6 shows the difference.

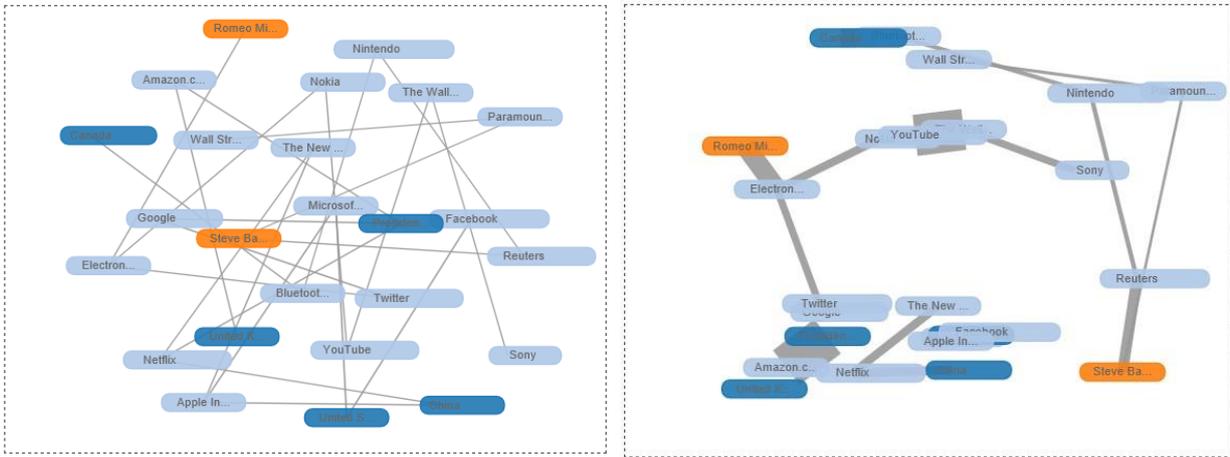


Figure 6. Weights Scale.

➤ Color (Entity Category)

In Entity graph, nodes are filled with colors to identify its Category. So, user can check/uncheck the corresponding color, at the bottom of the right-side panel, to decide which category will be displayed.

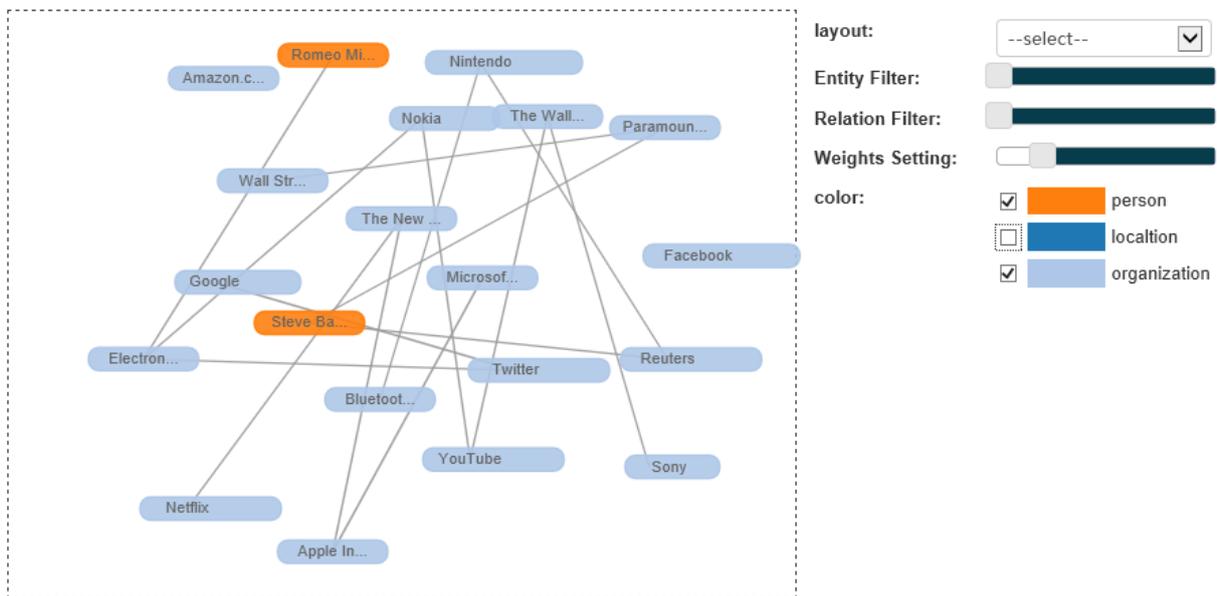


Figure 7. Color (Entity Category) Filter.

Figure shows an example that the location entities with the color of dark blue are removed in the chart, and it should be noticed that the relevant edges disappear either.

The last interaction that facility user’s operation should be mention: highlight.

As shown in Figure , when user move mouse to any node, the node and all the relevant edges will be highlighted with red color. Also, the value of url field of the selected entity will be popped out in a small text box.

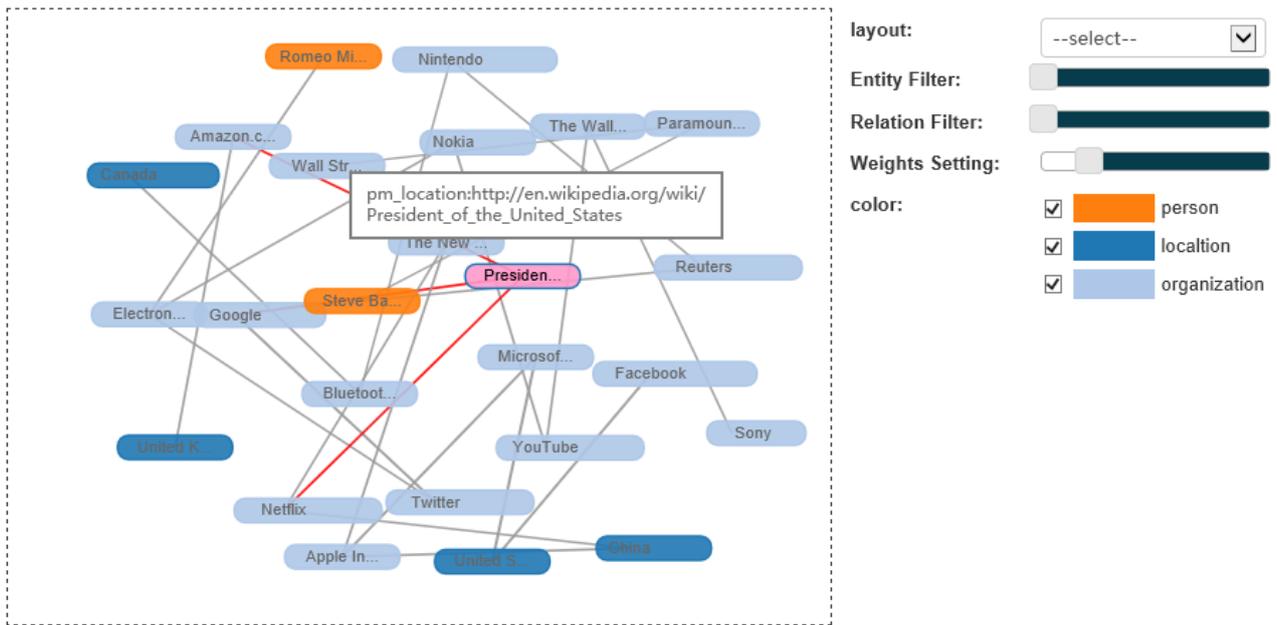


Figure 8. Highlight in Entity Graph

2.2 Radar Chart

2.2.1 Functionality

Radar chart is visualization control which provides graphical method of displaying multivariate data in the form of a plate-like chart. It automatically generates a polygon for an instance in a multi-axes coordinate system where each axis represents a component. Usually, the axes (components) are considered as independent with each other.

The radar chart is a useful way to display multivariate observations with an arbitrary number of variables. In general, it split the surface into several equi-angular areas with a star-like coordinate system with each axis representing one of the variables. In this coordinate system, the polygon of an instance is draw by:

- a) Plotting one point in each axis according to the magnitude of corresponding variable of the instance;
- b) Connecting points in nearby axes.

Figure gives a typical example of radar chart with six variables, for each represents the percentage of a specific service that user use in a week. There are three instances shown in this chart, and three kinds of colors are used to distinguish them. The blue one represents the usage rate of smartphones distributing among these services, yellow for tablets, and green for books.

To compare these observations, the most obvious feature shown in this chart is that they have something in common on the distribution. They cluster in the left side of this chart, which means that people hardly use a mobile device to pay online. Except for this, it’s not hard to see that people use books more frequently than the other two, especially for Internet Banking and News Sportsites. Smartphone and tablet distribute similarly and evenly among Email, Social Networks, Internet Banking, News Sportsites, Search Engine, and view shopping sites.

In conclusion, radar charts are primarily suited for strikingly showing bias and commonality. One possible application of radar charts in news data visualization is bias visualization. By assigning an entity/keyword in each axis, the radar chart will give user an intuitive image that how two or more event/news report differ from each other.

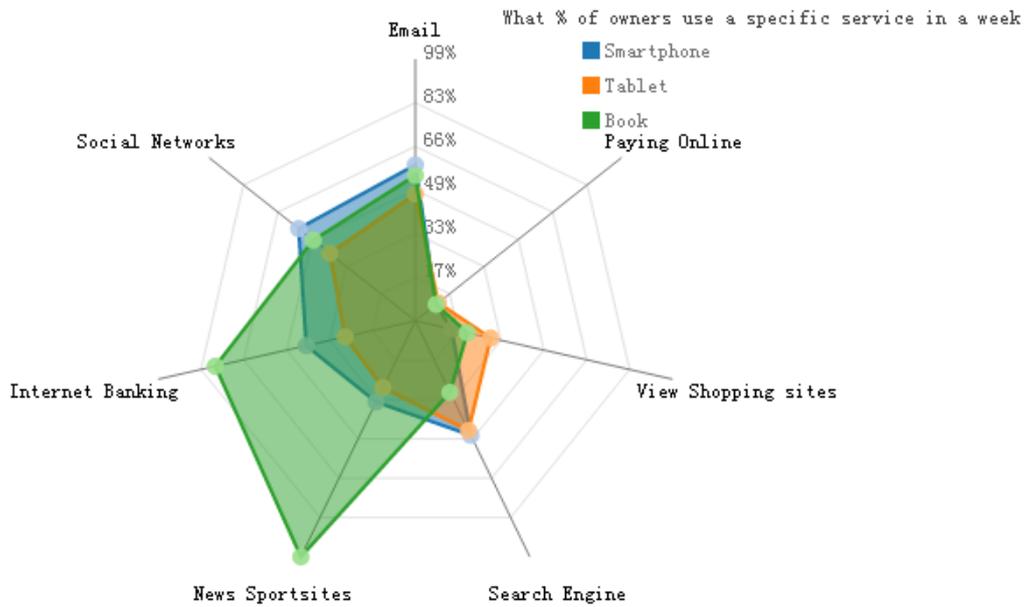


Figure 9. A Demo of Radar Chart.

2.2.2 Data format

This control is also programmed based on Node.js, a platform that allows for running a web server without the use of external software. Besides, D3.js, as a dependent library, provides support through a JavaScript API to draw this control.

```
[
  {axis:"Email",value:[0.59,0.48,0.55]},
  {axis:"Social Networks",value:[0.56,0.41,0.49]},
  {axis:"Internet Banking",value:[0.42,0.27,0.77]},
  {axis:"News Sportsites",value:[0.34,0.28,0.99]},
  {axis:"Search Engine",value:[0.48,0.46,0.30]},
  {axis:"View Shopping sites",value:[0.14,0.29,0.20]},
  {axis:"Paying Online",value:[0.1,0.11,0.10]},
]
```

Figure 10. Data Format for Radar Chart.

Error! Reference source not found. shows a sample data t of the same radar chart in the last subsection. Each line represents a variable/axis, where:

Table 3. Meaning of Data Fields for Radar Chart.

Axis	The label of the variable, a specific service in this sample
value	The corresponding value array of the observations

It should be noticed that there are three observations in this chart, each one corresponds to a number in the value field by the order of colors listed in the upper right corner. For example, all the first numbers of each line will be connected to generate a star plot representing one observation, and so does the second and third numbers.

2.2.3 Interactions

Figure shows an example of the main user interface, which contains the star plot of three kinds of devices. The variables list for the sample star plot is:

1. Email
2. Paying online
3. View shopping sites
4. Search engine
5. News sport sites
6. Internet banking
7. Social networks

We can look at these plots individually or we can use them to identify clusters of devices with similar features. For example, we can look at the star plot of the book device (the plot filled with green) and see that it is used approximately as frequently as the other two devices, little for paying online, but has a high usage rate in Internet banking and News sport sites. We can then compare the smartphone (the plot filled with blue) with the smartphone (the plot filled with orange). This comparison shows similar patterns. Both of these two devices are used evenly in all the services except paying online.

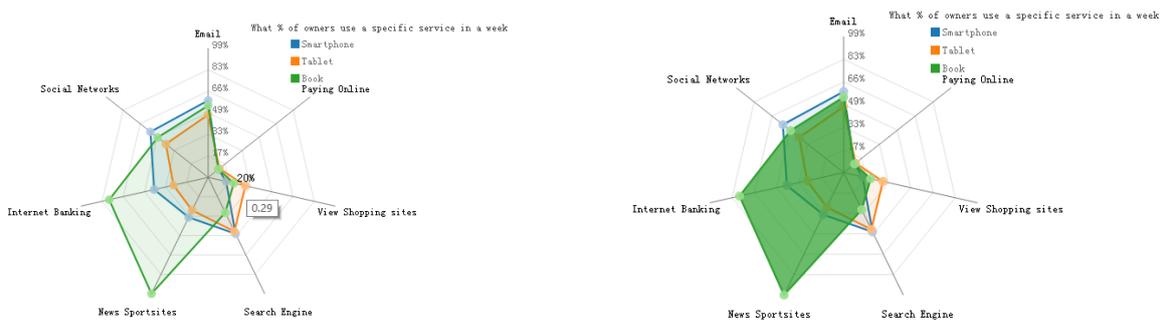


Figure 11. Highlight in Radar Chart.

Figure shows two examples of the highlight interaction. When user moves mouse to the data point, the chart will change to the left part of Figure . All the star plot become light, and the specific value displays floating.

The right part presents another way of highlight interaction to help user concentrate on one observation. When user move mouse in the area of one observation, the corresponding plot will becomes dark, and meanwhile the other plots become light.

2.3 Timeline

2.3.1 Functionality

A timeline is a method of displaying a list of news or events in chronological order, which is often used to visualize time lapses between events, durations, and the simultaneity or overlap of spans and events.

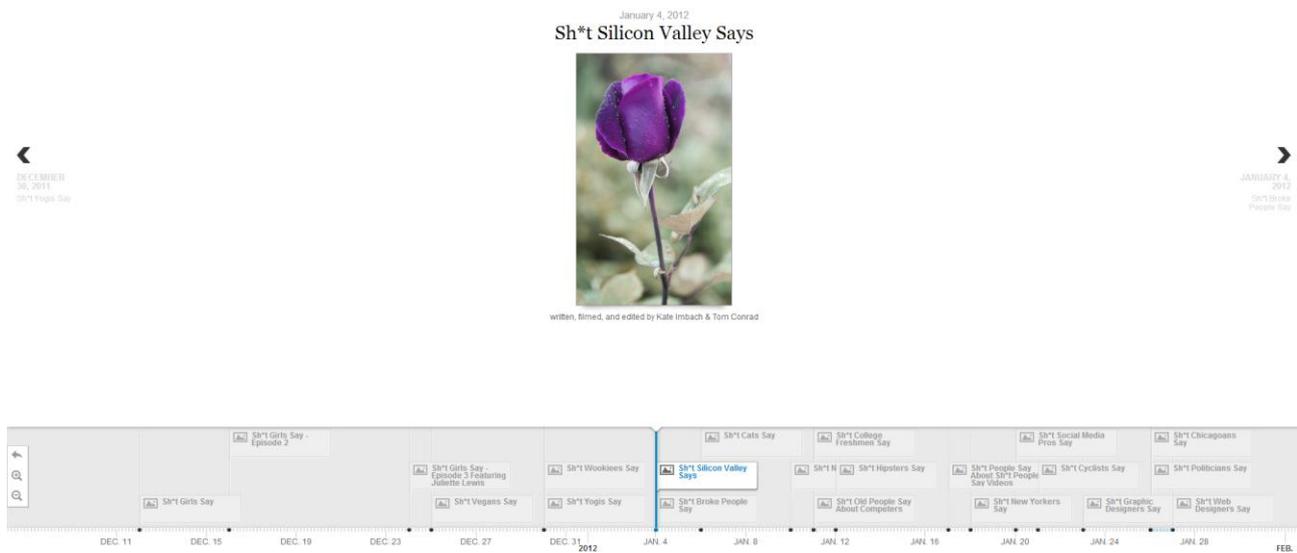


Figure 12. A Demo of Timeline.

This component provides a formalized interface to generate a timeline. The main user interface is shown in Figure . The entire page is split into two area laterally, of which the upper part shows the main body of news or events, and the lower part is a long bar labelled with dates alongside itself and events and news labelled on points where they would have happened.

As mentioned above, there are two functionality areas laterally:

➤ Events body

This area shows the main body of events, which could be a picture, a piece of news article or even a video. It should be noticed that the video is embedded in web page, so that the supported video formats depend on web browser.

Through a JavaScript API, all the events are displayed with a readable structure automatically. Figure 12 shows a type of structure including its data at the top, a title below the data, a picture in the middle, and the relevant introduction at the bottom.

On both sides of this page, there are also two arrowheaded buttons, labelled with a brief introduction, providing a functionality of switching to the next event or the previous one conveniently.

➤ Time bar

All the events list as a rectangle span begins at their start time and ends at the end time in this area. The label of each span gives a brief information about its title and whether it contains a picture (a small icon suggests that there is a picture). Except for browsing events and news one by one through the buttons on both sides of the upper area, clicking a rectangle span is another way of replacing the current event with the new one.

Time scale can be easily changed through the toolbar at the right side of this area, or the roller of the mouse. There are three buttons, of which the uppermost one provides the functionality of back to the begin date, the other two represent zoom up and out respectively.

Due to the chronological distribution of events, it is clearly to see the life circle of a news, what time is the breaking out point, and so on.

2.3.2 Data format

As a dependent library, Timeline.js gives support to this component, which requires a specific data format for every event or news shown in timeline. **Error! Reference source not found.** shows an example of the data structure for one event, and in table 4, each domain will be detailed.

```

{
  "startDate": "2011,12,25",
  "headline": "Vegans Say",
  "text": "text to be shown",
  "asset":
    {
      "media": "http://g.iphotos.baidu.com/pic/w%3D230/sign=1f6136aa38dbb6fd255be2253925aba6/b8014a90f603738de7cc8e73b21bb051f919ecd9.jpg",
      "credit": "",
      "caption": ""
    }
}
    
```

Figure 13. Data Format for Timeline.

The JavaScript library Timeline.js will be introduced in section 3 Library and Services.

Table 4. Meaning of Data Fields for Timeline.

startDate		Start date of event
endDate		End date of event
Headline		Title of event
Text		Body of event
asset	Media	Media embedded in the event
	Credit	
	Caption	Caption of the media

2.3.3 Interactions

There are two main ways to browse the events.

- Browsing in sequence

On both sides of the page, there are two arrowheaded buttons, labelled with a brief introduction, providing a functionality of switching to the next event (the right button) or the previous one (the left button) conveniently.

- Switching quickly

Each rectangle span listed at the time bar indicate an event. User can browse this event by clicking the matching span.

2.4 Time-Geo Visualization

Time and Location are two very important elements of news articles, or events. In the visualization prototype (D5.2.2) and news event registry, we have designed techniques to visualize news articles on the time dimension (Time Trend) and the geographic distribution (Map [4]). It is useful for user to learn about how an event develops as the passing of time and the where the event has spread. However, in the map, users can only get a static image of the geographic distribution of news articles. The Time-Geo Visualization controls aims to showing the development of an event dynamically.

2.4.1 Functionality

The Time-Geo Visualization control contains two controls. One is the map showing the locations of articles and another is a Time-Trend chart showing the number of reports on a sequence of time stamps. Different from static visualization, the map can show the geographic distribution of news articles from one time stamp to another. In this way, users will know where the event is firstly reported and how it spreads over the world.



Figure 14. A Demo of Time-Geo Control.

2.4.2 Data Format

The data inputted to the Time-Geo control should be a list of articles with publish time and publish location.

```
[
  {
    "id":1334938,
    "url":"http://www.eluniversal.com.mx/finanzas-cartera/2013/abren-con-resultados-
    mixtos-principales-bolsas-europeas-972749.html",
    "title":"Abren con resultados mixtos principales bolsas europeas",
    "language":"spa",
    "date":"2013-12-13T09:23",
    "source":"www.eluniversal.com.mx",
    "location":[
      23,
      -102
    ]
  },
  ...
]
```

Figure 15. Example Data for Time-Geo Control.**Table 5. Meaning of Data Fields for Time-Geo Control.**

Id	Id of the article
url	url of the article
Title	Title of the article, will be shown in the article list
Date	The publish time of the article
Source	The publisher of the article
Location	The location of the publisher, or where the news happens

2.4.3 Interactions

The Time-Geo control is designed to visualize the spreading or diffusion of information on geographic aspect. To help user tracing the trend, this control provides two interactions.

➤ Manual interaction

Users can click the nodes in the time trend chart to see the geographic distribution of news reports published at the selected time point. Or,

➤ Automatic interaction

The control can automatically generate a short animation which shows the spreading of news reports according to the given data. This will show the diffusion of information though over the world vividly.

3 Library and Services

This section describes the dependent JavaScript libraries and web services that support the controls in the front-end toolkit. Currently Node.js, D3.js and Google Map are included.

3.1 Node.js

Node.js is a platform built on Chrome's JavaScript runtime for easily building fast, scalable network applications. Node.js uses an event-driven, non-blocking I/O model that makes it lightweight and efficient, perfect for data-intensive real-time applications that run across distributed devices.

3.2 D3.js

D3.js is a JavaScript library for manipulating documents based on data. D3 helps you bring data to life using HTML, SVG and CSS. D3's emphasis on web standards gives you the full capabilities of modern browsers without tying yourself to a proprietary framework, combining powerful visualization components and a data-driven approach to DOM manipulation.

D3 allows you to bind arbitrary data to a Document Object Model (DOM), and then apply data-driven transformations to the document. For example, you can use D3 to generate an HTML table from an array of numbers. Or, use the same data to create an interactive SVG bar chart with smooth transitions and interaction.

D3 is not a monolithic framework that seeks to provide every conceivable feature. Instead, D3 solves the crux of the problem: efficient manipulation of documents based on data. This avoids proprietary representation and affords extraordinary flexibility, exposing the full capabilities of web standards such as CSS3, HTML5 and SVG. With minimal overhead, D3 is extremely fast, supporting large datasets and dynamic behaviours for interaction and animation. D3's functional style allows code reuse through a diverse collection of components and plugins.

3.3 Google Map

Google Maps is a set of services provided by Google. In the visualization component, the Google Maps Javascript API and Google Map API Geocoding web service are employed.

The Google Maps Javascript API lets developers embed Google Maps in your own web pages. Version 3 of this API is especially designed to be faster and more applicable to mobile devices, as well as traditional desktop browser applications.

The Google Map API Geocoding API provides interface to convert addresses (like "1600 Amphitheatre Parkway, Mountain View, CA") into geographic coordinates (like latitude 37.423021 and longitude -122.083739), which you can use to place markers or position the map. The Google Geocoding API provides a direct way to access a geocoder via an HTTP request. Additionally, the service allows you to perform the converse operation (turning coordinates into addresses); this process is known as "reverse geocoding."

Google provides a free license to these APIs but with a condition that our service must be freely and publicly accessible to end users.

4 Conclusions

In this document, we have presented the front-end toolkit which contains four controls currently. The toolkit is supposed to be used in industrial show case thus one important requirements is to be easily deployed. In this toolkit, we have integrated visualization controls in different sources but packaged them with well-defined interfaces. One do not need to know how the controls work but only need to feeding formatted data to the interfaces. In addition, all the controls in the toolkit are developed using Java Script therefore are independent with operating system.

In this deliverable, four controls with distinguish functionalities are presented. The Entity Graph is suitable for the visualization of relations between different instances. The instance here do not need to be named entity such as Persons, Locations and Organizations, but can be any concepts as long as we can find a way to measure the relationships or affinities between them. The Radar Chart is suitable to show different dimensions of an instance or visualize the difference of a small set of instances in the same dimensions. The Timeline control is suitable for the visualization of a sequence of news/events. At last the Time-Geo control is suitable to visualize data with both time stamp and location information.

This deliverable have introduced the functionalities, data formats and interactions of these four controls in detail and the codes have been uploaded in github (except the Time-Geo control. It will be uploaded in a short time).

This is the early version of the toolkits and in the following year, we will develop more controls and revise existing controls according to the feedback from industrial showcase.

5 References

- [1] <https://code.google.com/p/v8/>
- [2] <http://nodejs.org/>
- [3] <http://d3js.org/>
- [4] <https://developers.google.com/maps/>