

select data contained on webpages and drag it into an *art'nfact*, which presents a visual summary of the collected data. This experience, and the system we implement to exemplify it, is aligned to Pousman et al.'s depiction of casual information visualization scenarios and systems [12]. In particular, we focus on simplicity rather than power since we set our target audience as general users without specific visualization training or expertise.

In this paper we make a number of contributions. Firstly, we present an integrated experience for rapid collection, visualization, and dissemination of heterogeneous information found in the Internet. While many of the individual components we use to achieve this are not novel, we feel their combination and application in the fields of sense making and information visualization constitutes a tangible novel contribution. Secondly, we make the case for *art'nfacts*, which encapsulate more than just a collection of URLs and become more than an arbitrary group of data sources. Thirdly, we introduce a lightweight way of enabling the collaboration among people using different web sources and data-schemas.

3 RELATED WORK

Our work builds upon and integrates concepts from four areas: visualization toolkits; end-user visualization systems; web related visualization systems; and entity extraction from web pages.

Many visualization toolkits have been created to help programmers create visualizations. The Infoviz Toolkit, provides a set of libraries that supports the creation rich Java application containing rich and interactive 2D information visualization components [7]. Similarly, the Prefuse toolkit [9] provides developers with a set of Java libraries that helps them create interactive information visualization in standalone applications or web applets. We have use a port of the Prefuse toolkit into Flash, the Flare toolkit [8], as the engine for the visualizations contained within *art'nfacts*.

Because of our focus on end-user construction of visualizations, we are inspired from interactive visualization systems such as Tableau [15] and Spotfire [13], in particular these systems' ability to allow users to the quickly transform the way data is visualized by selecting and applying different visual attributes.

Several recent systems focus on the delivery of visualizations on the web for the purpose of collaboration. Examples of these are the ManyEyes [16] and Sense.us [10] research systems along with commercial websites such as Swivel [14] and ChartAll [4]. These projects examine the annotation, commenting and sharing of visualizations hosted on a particular web page, and provide a compelling example of the power of collaboration applied to the analysis and interpretation of data. Unlike the visualization in these systems, *art'nfacts* exist locally on a user's browser and are created by gathering data from its original

web sources. This property of *art'nfacts* allows them to maintain a links back to them.

We aim to facilitate the rapid and easy collection of information from the web and build upon web page content-extracting methods such as Dontcheva et al.'s [6] and Huynh et al.'s [11]. These methods parse a web page's source code and are able to extract structured data and identify many attributes contained within, e.g., book titles, prices, product entries, etc. We use these extraction methods in the context of our work as the means to an end.

Closely related to our research are Cammarano et al.'s work [1] on visualizing heterogeneous data on the web and the Vispedia system [2]. While the former work [1] presents algorithms to map heterogeneous data sources into a common schema, the latter [2] applies these mapping techniques along with user interaction for the visualization of data contained within Wikipedia pages. A crucial component in Vispedia's functionality is the DBpedia database (<http://dbpedia.org/>), "... a community effort to extract structured information from Wikipedia and to make this information available on the Web". This design feature of Vispedia translates into their system only having access to data from tables and a handful of structured information in HTML pages, i.e., the information present in the DBpedia. Similarly, a Vispedia user can only access an attribute if and only if there is a path connecting it to a "seed", user-selected table.

Our work is differentiated from Vispedia in the following main aspects: a) it allows structured (e.g. tables) and unstructured (e.g., a word in a paragraph) information kept on diverse sources / separate web pages to be incorporated into a visualization; b) it is a local solution, i.e. a browser add-on, which is not hosted on a centralized site; and c) it does not focus on automatic methods for schema mapping or automatically deriving supplemental data types and leaves that work to manual specification by users.

Our work is important because it starts to examine how far one can address data integration challenges without having a structured DB backend. Furthermore, *art'nfacts* have the potential to become a supportive element for collaborating tasks. In particular, *art'nfacts* are designed to be shared and combined among peers and potentially published to a crowd.

In summary, there is an important body of literature that touches different aspects of rapidly creating and sharing visualizations from web sources, yet each of these efforts only tackles a different aspect of the problem. Our work presents as a tangible contribution an entity that combines and applies the best aspects of these efforts in the fields of sense making and information visualization.

4 ART'NFACTS

Our implementation of *art'nfacts* aims to address usage scenarios where a person collects data from web pages while browsing the Internet, visually summarizes the data and later extends it, modifies it or shares it. Before

describing our interactive prototype in more detail, we first exemplify two *art'nfact* usage cases supported by our current implementation and that serve to emphasize the type of interactions and collaborations we envision.

4.1 Case 1: Collecting, Sharing, Checking & Extending

Alice surfs the web for her research about the richest people in China and decides to create an *art'nfact* about it to later share it with her friend, Bob. In her browser, Alice finds a web site (Forbes) containing billionaires from China. Alice drags the name of the first billionaire into one of the target boxes inside the *art'nfact* collection area (Figure 1). She then selects and drags the billionaire's net worth and age information from the web page into the second and third target boxes (Figure 5, top). The prototype reads the webpages' HTML source in order to name drop targets accordingly (e.g., Name, Net Worth, Age). The system is also able to detect that the collected data is part of a table and injects *GetXXX* buttons into the web page (Figure 1 – pop-out) to enable the collection of single rows or the whole table. After collecting her first row, Alice clicks the *GetTable* button to get all the billionaires in the current web page into the *art'nfact*. As Alice collects the data, the *art'nfact* updates a (default) Bubble Map visualization representing the information (Figure 1). Alice is satisfied with her *art'nfact* about China's wealthiest, saves it into a pack file and sends it to her friend Bob as an e-mail attachment.

Bob receives and loads Alice's pack file, effectively reproducing Alice's *art'nfact* in his browser. He doesn't find the Bubble Map view particularly informative and decides to switch to the Scatter Plot view using the adjacent combobox. Bob notices an odd data point: the richest person in China seems to be only 26 years-old. Bob decides to bring up the Forbes web page where Alice got her facts by clicking on this data point on the Scatter Plot. Navigating from this newly opened page, Bob learns that this young billionaire inherited her wealth from her father's construction company.

Having seen the information Alice shared with him, Bob decides to compare India's wealthiest people with China's and opens a Wikipedia web page containing information about India's wealthiest people. Using interactions techniques similar to the ones used by Alice, Bob collects the names, net worth, and age of the Indian millionaires listed in Wikipedia. To compare wealth by web source (i.e., country in this case) Bob creates a *country* column by interacting with one of the boxes in the *art'nfact* collection area, switches back to the Bubble Map view and maps the newly created *country* column to the bubbles' colour. He now sees that the wealth of the richest Indians is considerably larger than that of any Chinese billionaires, i.e., the largest Indian bubble can contain more than five times the largest Chinese bubble (Figure 2).

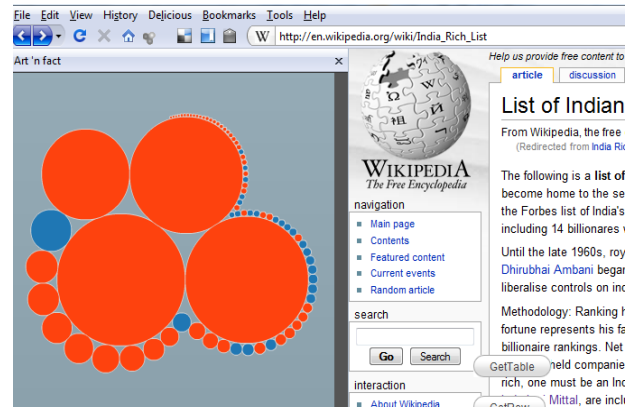


Figure 2: Detail of an artifact contrasting the wealth of China's (blue) and India's (orange) millionaires.

4.2 Case 2: Joining Data Tables from Two Art'nFacts

Charlie and Diane are friends with different interests: Politics and Economics. Using similar interactive procedures as the previous scenario, Charlie creates a red-blue visualization (Figure 3, left) of election results by state from the Wikipedia US 2004 election web page while Diane creates a median US household income map (Figure 3-right) by states from the US census website. Charlie posts a link to his *art'nfact*'s pack file in his blog and Diane (who subscribes to Charlie's RSS feed) downloads it.

Looking at Charlie's map Diane suspects there is a relationship between election results and household income and combines their two data sources to further explore her hypothesis. She selects the *merge* view in the data collection area (Figure 6), selects Charlie's and her datasets to merge and indicates that the state field in Charlie's and her *art'nfact* correspond to the same entity (Figure 4, a).

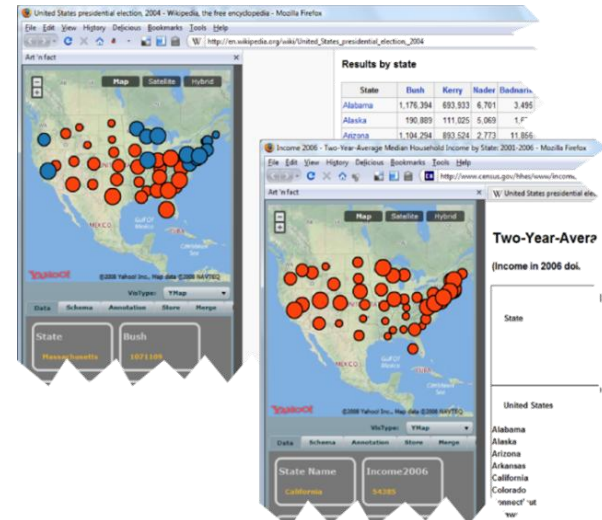


Figure 3: Art'nfacts reflecting (left) election results by state and (right) median household income by state.

After the merge operation is complete Diane chooses to display this newly joined data table as scatter plot showing income vs. differences between Bush's and Kerry's votes (Figure 4, b) with different colours assigned to each

candidate. The resulting chart supports Diane’s intuition of a relationship between the income and election results, i.e., it would appear that there are no democratic wins for states where the average income is less than \$43,000.

5 ART’NFACTS SYSTEM DESCRIPTION

We implemented an interactive prototype demonstrating the creation, viewing and modification of *art’nfacts* as an add-on to the Firefox Internet browser. Architecturally, our system consists of three parts: a Flash component, a browser add-on or extension, and the user’s active webpage. The Flash component hosts an *art’nfact* data table and most of our system’s UI and logic. The browser extension acts both as a container for the Flash component and provides access to the user’s webpage DOM for data extraction and HTML code injection. The Flash component takes advantage of the Flare visualization toolkit [7] and the Yahoo Maps Services API [17]. A low level description of how these components interact with each other falls outside the scope of this paper.

An *art’nfact*’s UI is divided in two main areas: a top visualization area and a bottom data collection area.

The visualization area (Figure 1) hosts a visual representation of the *art’nfact* data table. Our current implementation supports three visualization types: Scatter Plot, Bubble Map, and Map with the potential to include many more. Each of these visualizations has a number of visual attributes, e.g., size, x, y, location, color, label, that need to be mapped to a particular column of the data table. We use a simple rule-based mapping to match data types to columns with attributes to create a default visualization. Users can also re-map attributes using the data collection area, as we describe in the next paragraphs.

An *art’nfact*’s visualization area allows for a number of interactions. When a user places the mouse’s cursor over a data point, a tooltip displaying information about the data point (attributes + data source) appears. In addition to this tooltip, the *art’nfact* highlights the browser tab (if opened), containing the page from which the data was originally collected. If a user clicks on a data point, the system opens the browser tab if it’s not already opened and switches the browser focus to it.

The data collection area allows users to explore and manipulate the collected data in an *art’nfact* in different ways. This area contains a series of tabs that let users switch to a particular data operation mode.

The *data* tab contains a number of target boxes that (when assigned) correspond to particular attributes of the collected data, i.e., they collectively represent a data row. When a user selects data from a webpage and drags it into a target box, we create an RDF triplet mapping the page’s content and its metadata to a row /column in the *art’nfact*’s data table. For collected items, we extract XPath data (as in Dontcheva et al. [6] or Huynh [11]) that allows us to a) collect other data points with the same schema in the same or similar pages, and b) automatically label data columns derived directly from the web page. Our implementation recognizes three types of data primitives: strings, real numbers and locations (which are interpreted as such by the Yahoo Map Services). After the first row has been added, our system injects *GetXXX* buttons into the web page, so that a user can collect one or many rows at a time. Once collected, this content is stored in an internal data table that fuels the visualization component. Once the *art’nfact* detects a non-empty data table, it presents its default visualization (a bubble map) with an arbitrary mapping

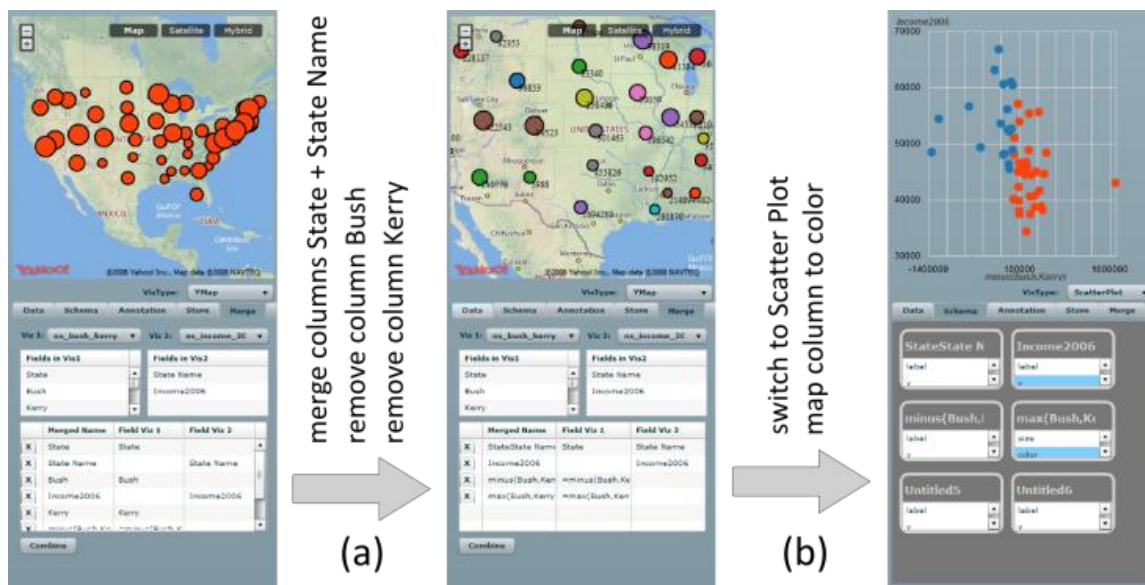


Figure 4: Joining two *art’nfacts* into a new one. (left) selecting two *art’nfacts* fills the merge table with the participating columns. (center) a user selects the joining columns and discard superfluous ones. (right) after joining and remapping color to the candidate’s column, the scatter plot reveals two distinct clusters.

between the visualization attributes (e.g., bubble size, colour) and each target box.

The *schema* tab provides users with the ability to remap target boxes to any of the current visualization attributes, e.g., a scatterplot's x, y, size and color attributes. This tab presents the same target boxes as in the *data* tab, but with their content replaced with a list of visualization attributes to map to (Figure 5, bottom).

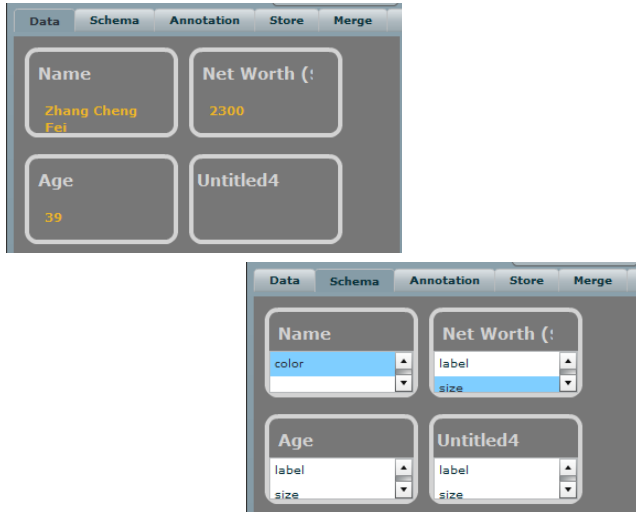


Figure 5: (top) Collection area's *data* view. Three of the four target boxes show data that was dragged into them from a web page. (bottom) Collection area's *schema* view. Two of the four target boxes show the visual attribute they are mapped to (the current visualization is a bubble map).

The *merge* tab allows a user to merge the data tables from two *art'nfacts*. A user performs a merge operation by first choosing the *art'nfacts* she wants to combine from two drop-down selection boxes. After this selection, a merging table gets populated with the fields of the participating data tables, each row having candidate names chosen by our system. The user then has the choice of selecting what pair of rows corresponds to the merging attribute by dragging a row into another. Finally the user can remove rows that should not participate in the new table (Figure 5).

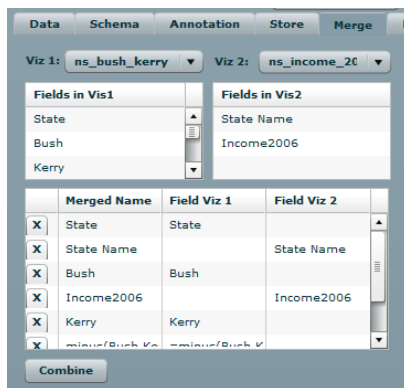


Figure 6: Collection area's merge view. The top two lists show the fields corresponding to the selected *art'nfacts*. The lower table presents a collection of rows that users can drag in order to select the merging column.

The annotation tab lets users store text comments within an *art'nfact*. In our current implementation, this tab consists of a simple writing pad where users can write notes for self-reference, documentation or commenting for 3rd parties. The *store* tab present users with a list of previously loaded *art'nfacts* from previous sessions as well as with the capability to load or import a new *art'nfact*'s pack file.

5.1 *Art'nfact* Serialization and Sharing

All of the information that constitutes an *art'nfact* can be represented as plain text, making *art'nfacts* capable of being encoded in a straightforward fashion, e.g., text, xml, etc. Encoding an *art'nfact* into bits results in what we call a pack file. This serialization capacity enables a number of collaboration and sharing scenarios, e.g., extending a table or combining it with data from other web pages or *art'nfacts*. In the case where a user extends or adds data points to a given *art'nfact*, she can use simple drag-and-drop interactions (as described in Case 1) and add further data with the injected *getRow* buttons. In the case where a user wants to merge the data tables from two *art'nfacts*, she uses the merge section of the data collection area (Figure 6) where she selects data sources and specifies the column(s) to merge.

Our current implementation supports a one-to-one asynchronous collaboration in model, where a user transfers a pack file using e-mail, an instant messenger client, by uploading it into a well know Internet location, etc. This model is a current limitation of our prototype and we plan to incorporate mechanisms to enable sharing *art'nfacts* in a synchronous way.

Transforming data is often a necessary task for creating and combining visualizations. For example, one might be interested in finding the difference between two columns, multiply a column by a scalar for comparison purposes, apply a logarithmic transformation, retrieve a column name, lookup a value in another table, etc. Because of this need, we provide basic functions such as addition, subtraction, division, maximum, and minimum that help creating meaningful visualizations. To leverage most of current information workers' expertise, we use syntax similar to Microsoft Excel. For example, in the Bush vs. Kerry visualization from Figure 3, right, the column $=max(Bush, Kerry)$ will yield *Bush* if and only if the cell named *Bush* has a higher value than the cell named *Kerry*. Users use these data transformations by defining new columns in the *data* view by entering the appropriate transformation in the title field of an empty target box. Table 1 presents a list of the transformation we support in our current prototype's implementation.

Table 1: List of data transformations supported by our current prototype.

Transformation	Description
max(Ca, Cb)	Given two columns, returns the name of the column with the largest value.
min(Ca, Cb)	Given two columns, returns the name of the column with the smallest value.
diff(Ca, Cb)	Given two columns, subtracts the value in Cb from the value in Ca.
sum(Ca, Cb)	Given two columns, adds the value in Cb and Ca.
url()	Returns the URL of the website the value in a cell originates from.

5.2 Scope and Limitations

In its current implementation, our system is capable of successfully parse and capture structured and unstructured information from a number of well-known Internet sites such as Wikipedia, Amazon, Yelp, Forbes and TripAdvisor. While this set of websites represents an important cross-section of informational sources in the Internet, it is not an exhaustive one. Also, our system only captures webpage tables that are arranged with their column labels at the top row and is unable to interpret tables where columns appear as rows, i.e. as the first column.

Most of this inability to capture some sources of information is a limitation that stems from an implementation issue. There are a number of existing techniques and research, such as end-user web programming [1] that we did not have time to incorporate and that can substantially open and increase the number web sources available for entities such as *art'nfacts*.

6 FIRST IMPRESSIONS

We demonstrated our prototype to seven people who use data from the web daily. Our users belong to our research lab and are unrelated to our project. We showed users videos demonstrating Cases 1 and 2. After each video we solicited user feedback and first impressions regarding the overall system’s functions and usage, as well as allowing them to interact directly with the prototype themselves. This last part of our sessions consisted in us guiding participants through the tasks used as examples in the paper

Users commented on concerns they had about our prototype. While drag-and-drop was praised, people realized that selecting (and dragging) a small piece of text is sometimes difficult. Also, a user pointed out that the tool did not help people to agree on, or prepare a schema for collaboration scenarios. Opinions were divided in terms of the default visualizations an *art'nfact* supported, e.g., while half our users rapidly understood Bubble Maps, the rest did not. Finally, most users commented that it took them a brief time to understand how to use the system.

Despite some of these critiques, users found that the *art'nfact*’s visualizations were extremely useful for investigating and sharing patterns of information. They particularly liked summarizing many different links in a single visualization for both revisitation and sharing scenarios.

The drag-and-drop interaction style was praised as quick, “simple” and as the “way people work”. While most people felt that using drag-and-drop was intuitive, they also found that selecting and dragging elements from web pages could be difficult. In particular, words with very few characters.

A user indicated that she liked the ability to quickly switch between different types of visualizations. Another area of praise among interviewees was how an *art'nfact* was linked to its data sources: “connecting back is key”, it is “more than just a snapshot” one of our users said.

All users voiced that sharing an *art'nfact* could be really useful in a number of collaboration scenarios. For example, a user felt that *art'nfacts* would be a powerful tool for teams of students doing assignments at school and another user saw *art'nfacts* helping conscious consumers doing product research. All our users saw the ability to combine and extend *art'nfacts* as a great, useful feature. For example, a user was pleased to have the ability to give to another person “the same experience” she just conjured.

7 CONCLUSIONS AND FUTURE WORK

While we have not tested *art'nfact* concept extensively; we have successfully used our prototype to rapidly create and later share visualizations from a wide variety of sites such as cia.gov, census.gov, wikipedia.org, cnn.com and forbes.com for sites with HTML tables; and amazon.com, yelp.com and tripadvisor.com for sites where information was less structured.

Future work includes both straightforward extensions, such as providing more and better visualization types, adding smarter data extraction / normalization mechanisms, and considering other interaction metaphors for capturing data, e.g., using contextual pop-ups instead of drag-and-drop. We would also like to refine the interaction for both creating and collaborating on visualizations.

We would like to extend the ways in which we automatically extract information from web pages to include traversal and loading of linked pages. Some sites accumulate tables of information, but other sites have the necessary information at the ‘leaf’ pages (similar to Dontcheva et al’s ‘Add Linked Pages’ [6]).

We have not focused on schema matching nor on automatically finding the information necessary for creating a visualization in the style of Vispedia [7]. It could greatly ease visualization creation as well as increase the power of the system to use some of the techniques discussed in [7].

Finally, while we designed our system with collaboration in mind, we currently only support the basic collaboration

pattern of “mailing information back and forth”. We intend to investigate ways to post *art'nfacts* at well known locations for people to review and refine (in the style of ManyEyes [16]) as well as mechanisms for more than one person to interact simultaneously with the same *art'nfact*.

8 REFERENCES

1. Bolin, M. and Rha, P. and Miller, R.C. (2005) Automation and customization of rendered web pages. *UIST*, 163-172
2. Cammarano, M., Dong, X.L., Chan, B., et al. (2007) Visualization of Heterogeneous Data. *IEEE TOV & CG*, 1200-1207.
3. Chan B., Wu L., Talbot J., Cammarano M., Hanrahan P. (2008) Vispedia: Interactive Visual Exploration of Wikipedia Data via Search-Based Integration. *IEEE TOV & CG*, 14(6), 1213-1220
4. ChartAll. <http://www.chartall.com>
5. DBpedia. <http://dbpedia.org>
6. Dontcheva, M., Drucker, S.M., Wade, G., Salesin, D., and Cohen, M.F. (2006) Summarizing *personal web* browsing sessions. *UIST*, 115-124.
7. Fekete, J.D. (2004) The InfoVis Toolkit. *Proceedings of the IEEE INFOVIS'04*, 167-174.
8. Flare Data Visualization Library <http://flare.prefuse.org/>
9. Heer, J., Card, S.K., and Landay, J.A. (2005) Prefuse: a Toolkit for Interactive Information Visualization. *CHI'05*, 421-430.
10. Heer, J., Viégas, F.B., and Wattenberg, M. (2007) Voyagers and voyeurs: supporting asynchronous collaborative information visualization. *CHI'07*, 1029-1038.
11. Huynh, D. F., Miller, R. C., and Karger, D. R. (2006) Enabling web browsers to augment web sites' filtering and sorting functionalities. *UIST '06*.
12. Pousman, Z., Stasko J.T., Mateas, M. (2007) Casual Information Visualization: Depictions of Data in Everyday Life. *IEEE TOV&CG*, 13(6), 1145-1152.
13. Spotfire. <http://spotfire.tibco.com/>
14. Swivel. <http://www.swivel.com/>
15. Tableau Software. <http://www.tableausoftware.com>
16. Viégas, F.B., Wattenberg, M., van Ham, F., Kriss, J., and McKeon, M. (2007) Many Eyes: A Site for Visualization at Internet Scale. *IEEE TOV & CG*, 1121-1128.
17. Wikipedia. <http://en.wikipedia.org/>
18. Yahoo Map Services, <http://developer.yahoo.com/maps/>