Visualization to Support Augmented Web Browsing

Dinh Quyen Nguyen
Faculty of Computer Science
University of Information Technology, VNU-HCM
Ho Chi Minh City, Vietnam
ndquyen@uit.edu.vn

Heidrun Schumann
Institute of Computer Science
University of Rostock
Rostock, Germany
schumann@informatik.uni-rostock.de

***Abstract*—In conventional Web browsing, to explore contents of interest while browsing a webpage, in many cases users need to leave that webpage and switch to external applications or other task-specific websites. To avoid switching between webpages, we present in this paper an approach to support the handling of further data and tasks added onto any browsed webpage in terms of augmented Web browsing. Without changing the existing Web contents and page-layouts, two levels of visual supports addable onto any existing webpage are suggested: (1) at *micro* level, visual cues encoding additional information are directly bound to particular hypermedia items, and (2) at *macro* level, additional visual layers are flexibly attached onto the browsed webpage. To illustrate the ideas, the Firefox add-on VizMe is developed with examples mimicking Google as a browser's extension, and to browse Web snippets with geospatial and temporal references.**

***Index Terms*—Augmented Web Browsing, Casual Information Visualization, Web Visualization.**

I. Introduction

A webpage is a mixture of hypermedia items presenting Web contents arranged and hyperlinked together, which are displayable and browsable on the window of a Web browser. With the advancement of nowadays hi-tech devices, computational intelligence, and Web services, it is requested to investigate and develop forward-looking visual interfaces and techniques that can support Web users in effectively browsing and completing many tasks with the diverse Web contents.

Using a search engine, reading news, or looking for information of interest are amongst the most common everyday activities on the Internet [1]. There are a lot of tools and services provided online to support the users in performing those tasks. Unfortunately, many existing tools and services are typically provided on specific stand-alone applications or websites (such as searching on `www.google.com`). That means, if a user wants to google further contents of interest while browsing a webpage, she has to switch to the Google site to get the needed information. In various cases, this is not always sufficient for the user to switch in between different webpages, as she might loose the context or connections due to the matters of cross-memorizing, locating, and relating the tasks in between the many windows. Therefore, we need interfaces to support Web users in reducing the switching of webpages and performing useful browsing tasks.

In multimedia browsing [2], data presented on the screen are content-centric – which means text, images, videos, and other raw media are the central objects. However, media contents alone are not always enough. Rather, visual cues are added as hints for data navigation and linking of contents.

With this paper, we pick up that direction and present a novel approach for augmented Web browsing supported by visual means. The idea is to present additional visual cues within one and the same webpage to support Web browsing, and to reduce switching in between different webpages. To arrive at that goal, we need to cope with two main problems:

- Identify, analyze, and extract suitable data from the browsed webpage or other Web resources to further visually communicate on a current browsed webpage.
- Decide how the additional data should be visually communicated.

Within the scope of this paper, we focus on the second problem: the visualization of additional data onto a current browsed webpage. Therefore, we develop a 2-step-approach. With the first step, we introduce a general procedure about which visual supports can be provided: at the *micro* level, visual cues encoding additional information are directly bound to particular hypermedia items, and at the *macro* level, additional visual layers are flexibly attached onto the browsed webpage (Section III). With the second step, we adjust the settings with regard to a given browsing scenario (Section IV).

In Section V, the paper continues with the realization of ideas through the implementation and demonstration of VizMe – a Firefox addon, before the paper ends with conclusions and future work in Section VI. But now, we first consider the related work.

II. Related Work

The fundamentals of our work bear on augmented browsing – an emerging technology in Web development, and take advantages of advances in visualization and interaction research.

A. Augmented Browsing

Augmented browsing is an interesting technology for Web developments [3]. Because no webmaster can afford to support all the requirements of diverse users, it is expected that a webpage provided online should be designed with facilities to be able to self-modify its content-structure and page-layout on purpose. Popular sites such as Google, Yahoo!, or Facebook today allow the users to slightly customize their client-side presentation of the pages, typically concerning user interests and experiences. It is useful because a person can remove unnecessary parts of contents from her personalized pages or to re-allocate and re-order contents for better navigating and browsing. However, it is not enough as the user might want to flexibly add and manipulate further data or tasks which might be missed on the existing webpages. To deal with it, personalizing the webpages with further hypermedia items and providing more interactions from the client-side is the amendment solution. Webpages are not only to be rearranged or self-modified, they are to be added with additional items and manipulatable with embedded scripts. For instance, with GreaseMonkey on Firefox, programmers and users have been creating scripts to personalize webpages [4].

Anyway, Web content modifications and scripting manipulations are strict in terms that developers and users must know well the structure of the webpages before modifying. Additionally, it is also problematic that the added data would cause changes in original contents and purposes of the webmasters. An effort to reduce those matters is to provide data and interactions on additional browser extension's components, and use them on demand. To this end, because browser extensions are separately developed from the webpages, visual supports and interactions in between the added data and the existing Web contents are important, which is still the challenge in current user interfaces research.

B. *Visualization of Data on the Web*

Visualization on the Web recently gets much attention from visualization research community. Viegas et al. [5] develop IBM ManyEyes to provide a platform for the users in creating or uploading pre-formatted data sets to get visualizations in forms of scatterplots, bar charts, treemaps, tag clouds, and so on. Andrews and Lessacher [6] implement various gadgets integrated onto Google Doc Spreadsheets with the support of Google Visualization API. Tableau Public [7] is another useful public tool for data visualization on the Web. Ho et al. [8] develop a Web-based toolkit to represent data visualized in mutiple-views linked by various graphical components. And many more visualizations can be now deployed online thanks to the frameworks such as D3 [9]. However, visualizations of data on the numerous webpages are still not at such desire.

The above-mentioned visualizations get attractions from Web users. But the visualization of data is strict in terms that the visual components typically stand alone and deal with the already-specified data. The visualizations represent data which may be collected from webpages and can be embedded back into webpages, but the visualizations and the Web contents are not forced to strictly connect together. In other words, visualizations have not been created or provided on existing webpages with the motivation on augmented Web browsing.

To support augmentedly browsing existing webpages, more studies are needed. Fisher et al. [10] develop concepts and a framework where visualizations are embeddable into any compatible application, while Toomim et al. [11] present first ideas for the visualization of some selected Web contents on existing webpages with the supports of an additional panel of a browser add-on. Fisher at al.'s work is so general, while the work of Toomin et al. doesn't show enough guidelines for augmented Web browsing. And thus their ideas could be examined and extended here so that to adapt well into the context of Web-based developments.

III. Visual Supports for Augmented Browsing

Providing additional data onto webpages but to keep page-layouts and contents is challenging and thus need to be examined in the light of visualization and interaction design.

A. *A General Approach*

In this paper, we introduce a general approach to enrich webpages by visualizations and interactions in the sense of augmented Web browsing.

Generally speaking, when browsing Web contents, users access content elements in terms of selecting and performing different tasks. Selections can be done through mouse or other events (keyboard, touch, etc.) which typically mark and highlight contents of interest on the webpages. Then, other tasks can be accomplished through requests or other manipulations associated with selected Web contents – so that to receive responses with data and tasks updated from servers or within the client by using client-side scripts. In this regard, visual supports must be developed and provided for all of those activities.

We define two levels of visualizations and interactions provided onto existing webpages (see Fig. 1). On the one hand, at the so-called *micro* level, visual cues and interactions are developed for the particular hypermedia items of interest in terms of the two tasks: *selecting* and *annotating*. On the other hand, at the so-called *macro* level, visualizations and interactions are provided to support the *navigating* for data, *merging* data, *sharing* data on different webpages, and *brushing and linking* things on the webpages.

In the next two subsections, we describe all options in terms of designing visualizations and interactions for the both proposed levels.

B. *Visual Supports at Micro Level*

We firstly indicate visualizations and interactions at *micro* level. We call them as *micro* level visual supports because the visualizations are designed for and will be bound only to a particular hypermedia item which position is locally determined on a browsed webpage (and not relating to any

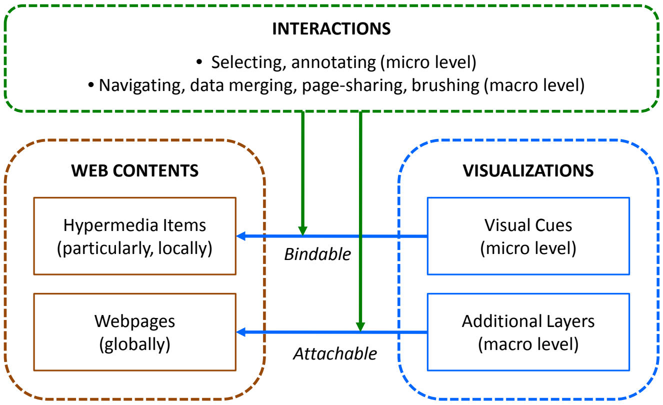


Fig. 1. Micro and macro level visualizations and interactions addable onto existing webpages for augmented Web browsing.

other contents, i.e. the global context, of that webpage). The additional data are visually encoded considering the basic HTML and CSS styles of the hypermedia item or through additional visual cues bound with the item. Interactions refer to that particular hypermedia item only.

1) Visualization Design: The visualization design to communicate information linked with existing hypermedia items is constrained with the characteristics of the hypermedia and the supports of Web browsers. In other words, the specifics of the selected hypermedia items (types, styles, available spaces, and their visual integrability) have to be considered:

- For hypermedia such as text, images, and other standard W3C HTML[1] in basic forms: Visualization can be taken place in terms of text font styles, color, background, border, and other visual styles of objects using CSS directly assigned to them.
- If basic CSS are not appropriate to communicate the additional data (such as dynamic information associating with a text, or suitable CSS are already used to encode other data), we need visual cues carrying the information and bind with the HTML items. For example, additional visual cues [12] can be mixed to text to show the time-references.
- For non-HTML-standard objects (e.g., complex graphical objects embedded on the webpages): Because the objects could be complexly created, mixing additional visual cues together with them is not straightforward. The additional visualization might change the original designs of the objects. Therefore, visual cues are suggested to be created and separately and closely bound with the objects.

2) Interaction: Although the additional visualizations provide further information, it would not be easy for all Web users to accept them. The visual cues may be sometimes and to somebody appropriate, and not for the others. Thus, visual cues are expected to be supported at run-time as needed. The items to be enriched with additional visual cues should be handled interactively:

- *Selecting*: Hypermedia content selection is the very common task on Web browsing, thus visualizations to support content selection are obviously needed at this *micro* level. When contents are selected, they should be emphasized with additional marking and highlighting.
- *Annotating*: After selecting specific items, additional data should be associated on demand. In this regard, showing them through CSS styles or additional visual cues bound with the hypermedia items can be taken place in terms of annotations. This can be accomplished through the requests sent by (e.g., Javascript) events. The visualizations are generated and shown at runtime, while the hypermedia items are changeless.

Visual supports at this *micro* level is beneficial for interacting and augmenting the browsing directly at particular hypermedia items. It provides run-time and instant information for hypermedia items under selection or examination. Visual hints with the support of CSS or simple additional visual cues are to be straightforwardly created, but creating complex visual cues for the non-HTML-standard objects can be challenging. The visual encodings are to be settled locally. However, they can extensively link with other data at the *macro* level as well, as discussed next.

C. Visual Supports at Macro Level

To aim the users in exploring contents arbitrarily and globally (i.e., somewhere) on a browsed webpage, or to reduce switching in between different webpages, visual supports are needed at *macro* level.

1) Visualization Design: Communicating additional data in connection to the whole webpage means that the developed visualizations should be flexibly created and provided on that webpage. How should we do that?

As discussed above, additional data must not change the existing page-layouts. Therefore, we suggest not to create visualizations as wrapped objects inline existing webpages, but as extra components on additional layers which are attachable and floatable anywhere on any browsed webpage. The hereafter-mentioned layer can be a typical HTML layer or other browser-integrable ones (such as an Adobe Flash canvas). The concrete designs (number of layers, which visualization components are developed on layers, which information do they communicate, and so on) depend on real scenarios (such as those developed in Section IV). To this end, two typical types of layers can be built:

- HTML-panel: A layer can be designed in the same manner as common HTML panels. But its visual components should be displayed and interacted with existing Web contents in specific ways. We suggest that HTML panels and components are adjustable with values indicated by the many CSS of the HTML.
- Graphical components: For more complex visualizations, we suggest to create layer components carrying visualizations implemented in HTML5 canvas or SVG that are well-supported by many Web browsers (other libraries such as Adobe Flash or Microsoft Silverlight are considerable as well).

2) Interaction: By providing additional visual components at *macro* level, the visualizations are expected to provide visual hints and insights flexibly so that end-users have further means to browse and explore the diverse Web contents. To achieve suitable visual supports, interactions are needed for specific tasks at hand. At this point, interaction design might follow Shneiderman's *visual seeking mantra* [13] (i.e., presenting overview of information, brushing and linking components, then showing details on demand), while to position layers, magic lenses [14] or folding visualizations [15] could be interesting choices.

- *Navigating*: By default, additional visual layers provide hints for the navigation to locate the Web contents of interest. Whenever the layers are created, positioned, or changed, the visualizations on components are interactively updated.

[1] http://www.w3.org/html

- *Data merging*: The additional data on the layer might refer to contents somewhere on a browsed webpage. Therefore, they are to be merged from diverse parts of the webpage. This is special because the merged data provide abstract and aggregated information for the data of interest.
- *Page sharing*: Because additional data can be from any webpage, sharing them in between webpages are necessary so that users do not need to switch in between the opened windows. The data are to be updated and shared amongst the browsed webpages so that to provide the cross-referencing of information on the Web.
- *Brushing*: Macro level visualizations support the navigating for Web contents. But to aim the navigation, in some cases, brushing and linking the data forth and back and explicitly in between the additional layers and the Web contents is a need.

Macro level visual supports on additional layer components provide flexible ways in displaying additional data and interacting with Web contents on existing webpages. In reality, they might be designed not separately, but in combination, or even blurred with visualizations at *micro* level when appropriate to completely support the different tasks of augmented Web browsing.

IV. Augmented Browsing Scenario and the Case of Geospace and Time

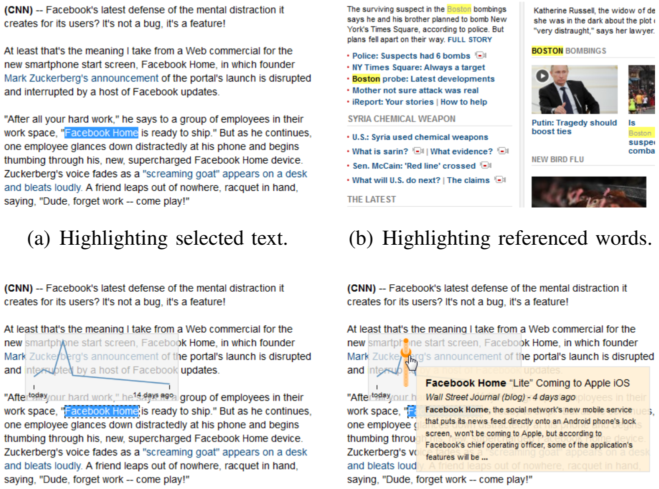
The above discussions refer to the first step, the generic design. Now, we want to consider the second step, the specific design of visualizations and interactions with regard to a concrete augmented browsing scenario – especially the case of Web snippets linking with geospace and time.

At *micro* level (Section IV-A), we design visual supports for highlighting and showing annotations close to selected Web snippets (i.e., small portions of Web contents) with time-referenced googled data. At *macro* level (Section IV-B), we support further tasks: navigating on additional layers, merging of data from webpages, sharing data amongst webpages, and brushing the additional layers with the existing Web contents (containing geo-referenced nameplaces).

A. The Micro Level Visual Supports

When *selecting* hypermedia contents on an existing webpage, we need to indicate the selections on screen, e.g. by highlighting them (such as the selected text "`Facebook Home`" in Fig. 2(a) or the highlighted words "`Boston`" in Fig. 2(b)). Based on the selection, further data can be combined such as the additional googled news with temporal information in terms of *annotations* so that to augment the browsing task.

In that regard, we create visual cues to be shown at *micro* level binding with the selected hypermedia item as follows: To visually communicate overview of time-referenced googled data, a typical time plot is employed on the fly: the horizontal indicates the records' timestamps normalized as time points (e.g., dates), and the vertical relatively expresses the number of googled records at each time point (Fig. 2(c)). Then,


(CNN) -- Facebook's latest defense of the mental distraction it creates for its users? It's not a bug, it's a feature!

At least that's the meaning I take from a Web commercial for the new smartphone start screen, Facebook Home, in which founder Mark Zuckerberg's announcement of the portal's launch is disrupted and interrupted by a host of Facebook updates.

"After all your hard work," he says to a group of employees in their work space, "Facebook Home is ready to ship." But as he continues, one employee glances down distractedly at his phone and begins thumbing through his, new, supercharged Facebook Home device. Zuckerberg's voice fades as a "screaming goat" appears on a desk and bleats loudly. A friend leaps out of nowhere, racquet in hand, saying, "Dude, forget work -- come play!"

(a) Highlighting selected text.

The surviving suspect in the Boston bombings says he and his brother planned to bomb New York's Times Square, according to police. But plans fell apart on their way. FULL STORY

- Police: Suspects had 6 bombs
- NY Times Square: Always a target
- Boston probe: Latest developments
- Mother not sure attack was real
- iReport: Your stories | How to help

SYRIA CHEMICAL WEAPON

- U.S.: Syria used chemical weapons
- What is sarin? | What evidence?
- Sen. McCain: 'Red line' crossed
- What will U.S. do next? | The claims

THE LATEST

Katherine Russell, the widow of de she was in the dark about the plot "very distraught," says her lawyer.

BOSTON BOMBINGS

Putin: Tragedy should boost ties

Is Boston suspe comba

NEW BIRD FLU

(b) Highlighting referenced words.

(CNN) -- Facebook's latest defense of the mental distraction it creates for its users? It's not a bug, it's a feature!

At least that's the meaning I take from a Web commercial for the new smartphone start screen, Facebook Home, in which founder Mark Zuckerberg's announcement of the portal's launch is disrupted and interrupted by a host of Facebook updates.

"After all your hard work," he says to a group of employees in their work space, "Facebook Home is ready to ship." But as he continues, one employee glances down distractedly at his phone and begins thumbing through his, new, supercharged Facebook Home device. Zuckerberg's voice fades as a "screaming goat" appears on a desk and bleats loudly. A friend leaps out of nowhere, racquet in hand, saying, "Dude, forget work -- come play!"

(c) Annotating with a time plot.

(CNN) -- Facebook's latest defense of the mental distraction it creates for its users? It's not a bug, it's a feature!

At least that's the meaning I take from a Web commercial for the new smartphone start screen, Facebook Home, in which founder Mark Zuckerberg's announcement of the portal's launch is disrupted and interrupted by a host of Facebook updates.

"After all your h
work space, "Fa
one employee g
thumbing throu
Zuckerberg's vo
and bleats loud
saying, "Dude, forget work -- come play!"

Facebook Home "Lite" Coming to Apple iOS
Wall Street Journal (blog) - 4 days ago
Facebook Home, the social network's new mobile service that puts its news feed directly onto an Android phone's lock screen, won't be coming to Apple, but according to Facebook's chief operating officer, some of the application's features will be ...

(d) Annotating with a popup.

Fig. 2. Concrete visual supports at micro level.

through interactions, users can explore more detailed googled data on demand: When the user hovers on the timeplot, we calculate the mouse position for a relative time point, then show a vertical line with dots to indicate the relevant amount of googled records; the user can then continue hovering on each separated dot, which is magnified, to prompt a pop-up annotation containing the detailed data returned by Google (Fig. 2(d)).

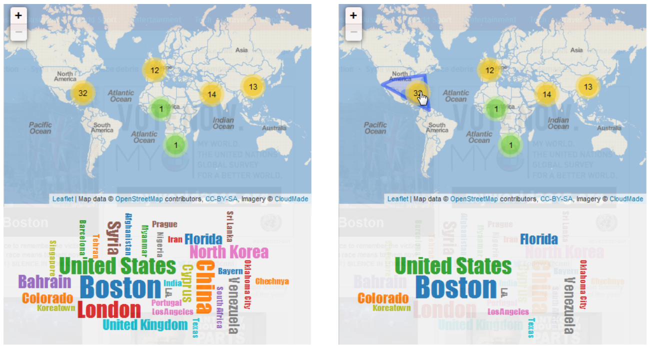
B. The Macro Level Visual Supports

At the *macro* level, we provide visual supports for the *navigating* with additional layers, *merging* of data from webpages, *sharing* data amongst webpages, and *brushing* the additional layers with the existing contents (containing geospatial information) on a current browsed webpage.

For navigating with additional visualizations, the additional layers and visual components are created and attached onto any browsed webpage with regard to the tasks at hand. In particular, we provide a HTML layer attached directly to the webpage that carries the googled data so that the user doesn't need to switch to the Google site for other data of interest. The googled data are also shared amongst all opened windows, where similar HTML layers are created and attached onto all the browsed webpages. Googled data are automatically synchronized if one of the open webpages updates its contents.

We similarly provide another HTML layer component containing a rich text editor on the browsed webpage, and allow the user to select, drag and drop, and edit contents gathering from the webpages. The gathered contents are shared amongst the open webpages as well.

Another important design is that we support merging and showing the summary of text on the VizMe panel. Through interactions, important words or placenames from text on a webpage are analyzed and extracted, and then visualized in terms of tag clouds (sizes of the words indicate their frequencies), where the beautiful Wordle [16] is employed.



(a) Nameplaces wordle in association with map markers.

(b) As a marker is hovered, related words are highlighted.

Fig. 3. Concrete visual supports at macro level.

The wordle of placenames gives hints for geographical places on the webpage. But for geographical data, it is more intuitive to be shown on geographical maps. Thus, for each placename indicating a location, it should be shown as a marker on the maps. Taking in mind that a placename might appear many times in the Web contents, we need to include that information on the maps as well. To this end, the visual solution proposed by [17] is employed. This solution also solves the matter that many markers for different placenames are too close together on the maps: within a square area (e.g., 60×60 pixels) the markers with numbers are aggregated. The bounding area of the markers is optionally shown (in blue) as the user hovers on a marker. Fig. 3 illustrates the placenames in terms of map markers and their wordle.

So far we have provided supports for visual navigating, data merging, and page-sharing. Now the task brushing is considered. With the just-mentioned design, visual components should communicate the relational information in between the maps, placenames wordle, and existing Web contents. Thus, they should be brushed-and-linked. We employ the mechanism affecting the associated data when the user moves and pans the maps. As an icon on the map is hovered, besides the blue bounding area, related words on the wordle are highlighted as well. When the user clicks on an icon marker or pan the maps, the maps views are refreshed, and the new marker icons and new wordle are updated. Besides, as the user hovers on a word of the wordle, associated words on Web contents are highlighted or visually linked through yellow straight lines. At that point, the visualizations provide hints about the places of the words in the Web contents so that the user is able to navigate to, e.g., by mouse scrolling the webpage.

V. The Firefox Add-on VizMe

We have presented the 2-step approach in designing visualizations and interactions to support augmented Web browsing. This section realizes the ideas with use cases demonstration.

A. Environment

VizMe is implemented as a Firefox extension on Firefox add-on builder[2]. It leverages standard Web technologies (including JavaScript, HTML, CSS, and SVG) and employs many open-source libraries and Web services to handle data at the client-side and visually communicates them on the screen.

1) Interface: The interface and functionalities on VizMe are designed with common tools to broadly support diverse end-users. Thus, at *macro* level we provide with VizMe a main HTML panel togglable and shown with transparencies. The panel is toggled through a button on Firefox's main toolbar or through hot-keys or context-menus. Depending on the tasks, we show on the panel different visual components, which are also selectable through the menus at the top of the panel. The menus include: "`Google`" with "`Web`", "`News`", "`Images`", and "`Videos`" for googled data, "`Edit`" with a rich text editor, and "`Visualize`" with "`Wordle`" for tag clouds and "`Places`" for geographical maps. Through the panel, users can navigate with visual components, merge and share data, and interact to brush them with existing Web contents. And at *micro* level, VizMe supports showing timeplot and pop-up annotations close to specific Web snippets when they are selected and examined.

Now, everything can be accomplished with the supports of events managed by Firefox chrome, as illustrated in the following use cases.

B. Use Cases

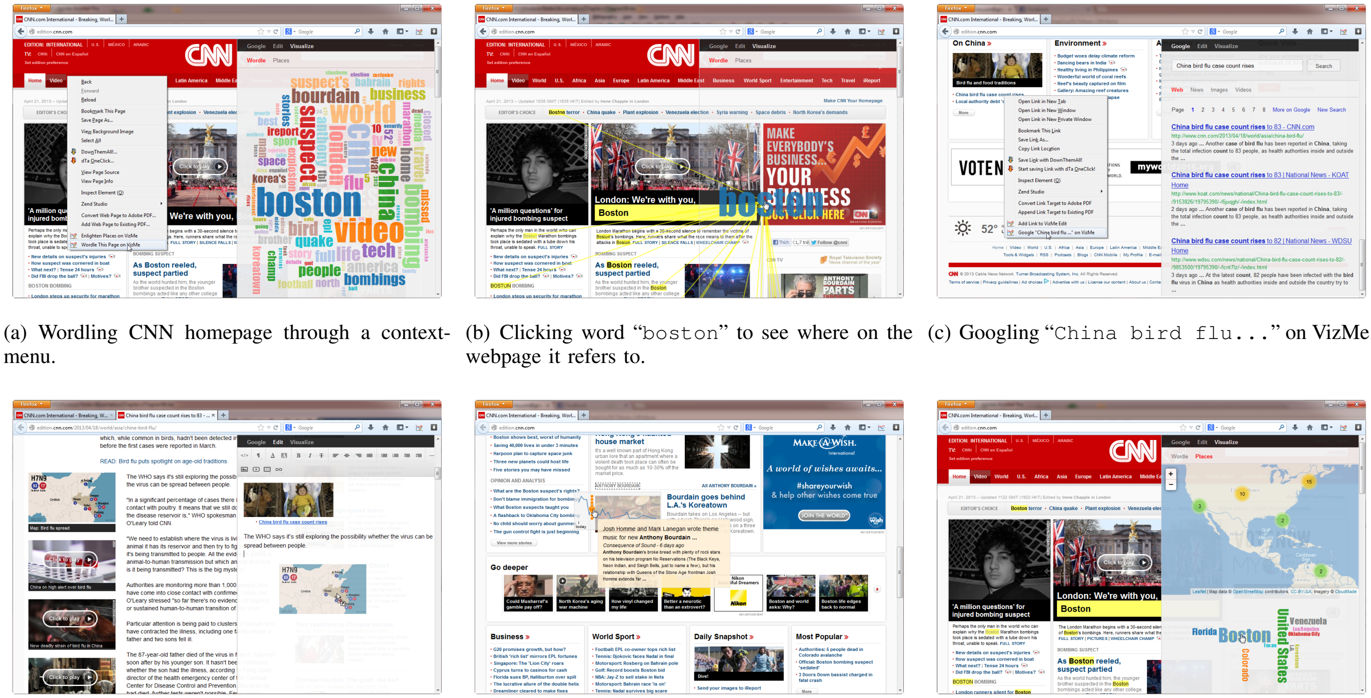
Supposing that Jane is surfing CNN online. We support her to browse, google, and gather Web contents, and to interact with additional visualization components to explore data of interest – using the above-developed *micro* and *macro* level visual supports.

1) Mimicking Google Integrated as a Browser Extension: Searching the WWW and the many other tasks supported by Google, in our opinion, if provided as browser's extensions, facilitate Web users in better *augmentedly* browsing the webpages as follows.

a) Navigating with Summary: Firstly, when Jane is surfing CNN homepage, it is expected that Jane has to skim all tips and headlines on the page to know what CNN presents at that time. It would be interesting if there is a summary of all news in terms of a wordle as shown in Fig. 4(a) to help her to have an overview about the contents of that page. In other words, as browsing any webpage, instead of reading the many words for necessary data, one might skim just prominent words in a wordle representation for useful information. At this point, we employ D3-Cloud library[3] for the implementation of the visualization component (*macro* level). Now, with the wordle created on the webpage, Jane can hover on each word to brush the associated highlight words on the webpage (*micro* level), and explicitly link to them on the webpage or navigationally scroll to relevant places on the webpage to access the detailed contents (*macro* level) (Fig. 4(b)). The wordle is shown only on the associated webpage.

[2] https://builder.addons.mozilla.org/

[3] http://www.jasondavies.com/wordcloud/



(a) Wordling CNN homepage through a context-menu.

(b) Clicking word "boston" to see where on the webpage it refers to.

(c) Googling "China bird flu..." on VizMe.

(d) Dropping image "Map: Bird flu spread" to the editor.

(e) Showing time-referenced news annotation for "ANTHONY BOURDAIN".

(f) Updating markers and nameplaces as the maps in Fig. 3 change view.

Fig. 4. Screenshots of tasks performing on CNN webpages (April 21 2013) with VizMe.

b) Browsing Googled Data: One of the word on the wordle that Jane notices is "flu". By brushing and scrolling to the bottom of CNN homepage, Jane finds out an article entitled "China bird flu case count rises". Now, instead of reading the article, Jane wants to know more about "China bird flu". Thus, she googles ‘‘China bird flu ...’’ on VizMe through a context-menu. Fig. 4(c) shows on the add-on panel (*macro* level) recent results responded by Google Search API. To this end, Jane can browse those related websites, google for more results, or skim the googled results for the information that she actually needs.

In the current implementation, we show on VizMe's main layer the Web, news, images, and videos Google search results. However, according to the available Google services, various facilities provided on official Google website can be missed in the tool.

The contents to be googled can be text selections, hyperlink texts, or image filenames selected from any webpage. For hyperlinks, the text string inside the <a href> is used, while for images, their file names are temporarily used. Different with the wordle that is strictly linked only to the associated webpage, the googled results are shared (*macro* level) in between the many browsed tabs. This will help users to keep the googled results for more explorations when accessing many webpages. In addition to that, users can also amend more text (e.g. from other selections or other webpages) to an existing search text.

c) Merging Contents to Editor: As reading a webpage, Jane might want to collect contents of interest for further usages. Of course, she can copy and paste to Google Docs[4] or other document-editors. But, it would be more convenient if she collects them directly onto a VizMe interface layer, where she can later access them, e.g., visualizing with Wordle, or share on other opened webpages (*macro* level). Fig. 4(d) demonstrates the contents from two different webpages that were selected and drag-dropped onto the rich text editor Redactor[5] on VizMe main panel.

2) Exploring Spatial and Temporal Web Snippets with Visual Supports: So far we have presented the demonstration examples where the mimic Google facilities are developed as Firefox extensions. Now we will go on with the examples concerning the visual exploration of Web snippets with geospatial and temporal references on webpages.

a) Annotating with Time-referenced News: Following the *micro* level visual supports designed in Section IV-A, we implement to show time-referenced googled news relating to a selected Web snippet under examiniation. Activating through hot key (default is the Shift key) and when Jane is selecting a Web snippet or hovering on a hyperlink, we retrieve Google search results and visually communicate them as annotations close to the Web snippet or hyperlink.

Currently Google API v1.0 returns at most 64 records for each search. Most of them are recent Web resources (commonly within around one month) posted on popular news websites. Each of the record contains a title of the related

[4] https://docs.google.com/

[5] http://imperavi.com/redactor/

article, its URL, the publisher, a short descriptive text, the timestamp (in UTC time format), and other data. In that way, we can easily compute the distribution of news records over time. We accumulate the records in terms of days, and show them as an overview time plot annotation.

Through interaction, Jane can explore for more details on demand, such as hovering over the time plot to see the number of records for each time point (day) in forms of dots on orange (if having) or gray (if not having) vertical line. She can also able to read the detailed googled data as hovering on each dot in terms of a pop-up annotation (see Fig. 4(e)), and click on the dot to open the full article on another Web browser tab (or window) following the URL.

b) Navigating with Geographical Maps: Another interesting task in surfing webpages is exploring name entities from the Web contents. With current advancements on semantic Web, it is expected that important contents could be analyzed, extracted, and somehow displayed to the users. In this implementation, we employ Thomson Reuters Calais[6] to extract the name entities from Web contents, then get the placenames and visualize them on VizMe at *macro* level. Following the design provided in Section IV-B, we visualize the placenames as marker icons on Leaflet maps[7] together with a wordle visualization for the names of the places.

When surfing CNN homepage, Jane is interested in knowing which locations CNN is talking about. She uses VizMe to show the placenames on geographical maps and wordle. Recalling Fig. 3, she easily know that most of the news on 04/21/2013 relate to America. By zooming and panning, Jane can explore details on demand the markers and placenames. For instance, she hovers and clicks on the marker numbered 32 to update the wordle as well as the markers as map changes its view. Also, similarly to the functionality presented in the previous examples, any placename on the wordle can be hovered or clicked to link to the contents on the webpage (Fig. 4(f)).

VI. Conclusion

As Web browsing is an everyday online activity, providing visual means to support Web browsing while not necessarily leaving the browsed webpages is a need. This paper provided a 2-step-approach (generic solutions and a real scenario) for the manipulations of Web contents on any webpage through the Firefox add-on VizMe. At the *micro* level, visual supports are provided at particular hypermedia elements, and at the *macro* level, visualizations and interactions are flexibly attached onto the browsed webpages.

While to many people googling can be a familiar task, interacting with other visualizations on VizMe can be strange. However, because visualizations support users with chances to comprehend Web data in meaningful ways, providing visual supports on webpages is useful in terms of casual Infovis [18].

The add-on VizMe was developed for Firefox with some first facilities. In the future, we will combine further Web services and visualizations into it, especially the visualizations for numerical data and statistical tables on webpages, benefiting by various examples provided e.g. on D3[8]. In addition, the implementation of VizMe will also be carried out for Google Chrome and other Web browsers. Finally, as current add-on is developed for Web browsing on PCs, future work aims at the development of tools for multi-touch screens and small mobile-device screens as well.

References

[1] Pew Internet Research, "What Internet users do online," Pew Internet & American Life Project, Dec 2012, http://www.pewinternet.org/Static-Pages/Trend-Data-(Adults)/Online-Activites-Total.aspx.

[2] Y.-T. Zheng, Z.-J. Zha, and T.-S. Chua, "Research and applications on georeferenced multimedia: a survey," *Multimedia Tools and Applications*, vol. 51, pp. 77–98, 2011.

[3] O. Díaz, "Understanding Web augmentation," in *Current Trends in Web Engineering*, ser. LNCS, M. Grossniklaus and M. Wimmer, Eds. Springer Berlin Heidelberg, 2012, vol. 7703, pp. 79–80.

[4] J. P. Bigham and R. E. Ladner, "Accessmonkey: a collaborative scripting framework for Web users and developers," in *Proc. of 2007 W4A*. New York, NY, USA: ACM, 2007, pp. 25–34.

[5] F. B. Viegas, M. Wattenberg, F. van Ham, J. Kriss, and M. McKeon, "ManyEyes: a site for visualization at Internet scale," *IEEE Trans. on Visualization and Comp. Graphics*, vol. 13, pp. 1121–1128, Nov 2007.

[6] K. Andrews and M. Lessacher, "Liquid diagrams: Information visualisation gadgets," in *Proc. of 14th Int'l Conf. on Information Visualisation*. Washington, DC, USA: IEEE Computer Society, 2010, pp. 104–109.

[7] Tableau Software, "Free data visualization software," http://www.tableausoftware.com/public/, accessed April 2013.

[8] V. Q. Ho, P. Lundblad, T. Åström, and M. Jern, "A web-enabled visualization toolkit for geovisual analytics," *Information Visualization*, vol. 11, no. 1, pp. 22–42, 2012.

[9] M. Bostock, V. Ogievetsky, and J. Heer, "D^3 data-driven documents," *IEEE Trans. on Visualization and Comp. Graphics*, vol. 17, no. 12, pp. 2301–2309, 2011.

[10] D. Fisher, S. M. Drucker, R. Fernandez, and S. Ruble, "Visualizations everywhere: A multiplatform infrastructure for linked visualizations," *IEEE Trans. on Visualization and Comp. Graphics*, vol. 16, no. 6, pp. 1157–1163, 2010.

[11] M. Toomim, S. M. Drucker, M. Dontcheva, A. Rahimi, B. Thomson, and J. A. Landay, "Attaching UI enhancements to websites with end users," in *Proc. of the SIGCHI Conf. on Human Factors in Computing Systems*. New York, NY, USA: ACM, 2009, pp. 1859–1868.

[12] D. Q. Nguyen, C. Tominski, H. Schumann, and T. A. Ta, "Visualizing tags with spatiotemporal references," in *Proc. of 15th Int'l Conf. on Information Visualisation*. Washington, DC, USA: IEEE Computer Society, 2011, pp. 32–39.

[13] B. Shneiderman, "The eyes have it: A task by data type taxonomy for information visualizations," in *Proceedings of the 1996 IEEE Symposium on Visual Languages*. Washington, DC, USA: IEEE Computer Society, 1996, pp. 336–343.

[14] E. A. Bier, M. C. Stone, K. Pier, W. Buxton, and T. D. DeRose, "Toolglass and magic lenses: the see-through interface," in *Proc. of 20th Annual Conf. on Comp. graphics and Interactive Techniques*, ser. SIGGRAPH '93. New York, NY, USA: ACM, 1993, pp. 73–80.

[15] C. Tominski, C. Forsell, and J. Johansson, "Interaction Support for Visual Comparison Inspired by Natural Behavior," *IEEE Trans. on Visualization and Comp. Graphics*, vol. 18, no. 12, pp. 2719–2728, 2012.

[16] F. Viegas, M. Wattenberg, and J. Feinberg, "Participatory visualization with Wordle," *IEEE Trans. on Visualization and Comp. Graphics*, vol. 15, no. 6, pp. 1137–1144, 2009.

[17] L. Mahe and C. Broadfoot, "Too many markers!" Google Developers, Dec 2010. [Online]. Available: https://developers.google.com/maps/articles/toomanymarkers

[18] Z. Pousman, J. Stasko, and M. Mateas, "Casual information visualization: Depictions of data in everyday life," *IEEE Trans. on Visualization and Comp. Graphics*, vol. 13, pp. 1145–1152, November 2007.

[6] http://www.opencalais.com/APICalls

[7] http://leafletjs.com/

[8] https://github.com/mbostock/d3/wiki/Gallery