

# Improving Browser History using Semantic Information

José Sousa, Marco Pereira and Joaquim Arnaldo Martins

*DETI - Department of Electronics, Telecommunications and Informatics, University of Aveiro,  
Campus Universitário de Santiago, 3810-193 Aveiro, Portugal*

*IEETA - Institute of Electronics and Telematics Engineering of Aveiro, University of Aveiro,  
Campus Universitário de Santiago, 3810-193 Aveiro, Portugal*

**Keywords:** Web Browser, Web History, Semantic Web.

**Abstract:** Revisiting web pages is part of the routine of web browser users, yet the use of tools designed specifically for this function is not widespread. In this paper we describe a browser extension called "Rdfa History" that attempts to improve the usefulness of the browser's history list by enhancing it with semantic information (in RDFa format) gathered from previously visited pages. The collected semantic information provides a richer pool of information than the traditional time/url/title fields provided by browsers, allowing users to perform queries not only by keywords but also by particular topics or entities that appear on previously visited pages. This extension brings to user's attention the usually invisible semantic web, and can be used both as tool that aids to revisit pages as well as a tool that helps to discover new pages related with the ones previously visited. It can also be used as a digital preservation tool that users can configure to collect and store history lists in a personal digital repository in order to access them from different browsers or devices, as well as use the history list as context provider for other documents that they might collect.

## 1 INTRODUCTION






For the majority of the population, using the Internet is a synonym of web browsing. It is not a stretch to claim that web browsing (for various purposes) is the most common activity performed when using a computer, both in number of browsing sessions as in time spent. Multiple studies (Tauscher and Greenberg, 1997), (Cockburn and Mckenzie, 2001), (Herder, 2005) have shown that an important part of the time spent browsing the web is not spent visiting newly discovered pages but instead revisiting those previously visited. There are several factors that influence why a given page is revisited (Obendorf et al., 2007), (Adar et al., 2009), (Kumar and Tomkins, 2010): from the type of page to the expectation to find new content on the page or to the particular navigation habits developed by each user.

With so many pages being revisited it would be reasonable to expect that the tools browsers provide to assist page revisits would be used often, yet as another study found out, the browser history list is hardly used, having originated only 0.2% of all page requests (Weinreich et al., 2008). Furthermore, when revisiting a web page users appear to either know the

URL, thus implying familiarity with the page itself and being able to use another of the modern browser page revisit helper, the URL auto-complete feature, or to simply attempt to retrace the steps that lead them to the intended page. This last scenario can result in frustration if the user is unable to recall another page to start the revisit process, or is unable to input (a reasonably close to) the original query in the search engine. This last scenario can result in frustration if the user is unable to recall another page to start the revisit process, or is unable to input (a reasonably close to) the original query in the search engine.

The low use of browser's history list can be attributed to a combination of factors: the relative low visibility of this feature contributes for some users not being aware of its existence at all, and those who actually use it think of it as a tool of last resort that they only use it if they are certain it will lead them to the web page they wish to revisit (Obendorf et al., 2007). Once the habit of not using in the history list settles in, it is maintained even if the users switch browsers and regardless of the history list features. If initially history lists in browsers provided little more than a linear list of visited pages (JasonSmith and Cockburn, 2003) sorted by access date (sometimes with precise

Table 1: Comparison of browser's history list features.

						
Search	Full-text	✓	✓	✗	✗	✓
	URL	✓	✓	✓	✓	✓
	Title	✓	✓	✓	✓	✓
	Other	✗	✗	✓ <sup>[1]</sup>	✓ <sup>[2]</sup>	✗
Sorting/Grouping	Date/Time	✗	✓	✗	✓	✓
	Date/URL	✓	✗	✗	✓	✗
	Date/Title	✗	✗	✗	✓	✗
	Only Date	✓	✗	✗	✗	✗
	Only URL	✓	✗	✗	✗	✗
	Only Title	✗	✗	✗	✗	✗
	Other	✗	✗	✗	✓ <sup>[2]</sup>	✗

<sup>[1]</sup>Frequently visited URLs

<sup>[2]</sup>User defined tags.

time discarded) and page title or URL, nowadays they are much more powerful, with default implementation on some browsers offering full text search of the visited page's contents, as summarised in 1. Being able to perform full text searches in the contents of a previously visited web page is an important step forward to increase the odds a user has to find a specific page, yet there are scenarios where it might fail to filter enough previously visited pages to be of any use. Full text search might not even be helpful in scenarios where the user remembers the topic of the page (or of something in the page) he wishes to revisit, but not any exact word contained in the page. To deal with these scenarios we propose to enhance the history list using semantic metadata in the RDF-in-attributes (RDFa) format (Adida and Birbeck, 2008) collected from the visited pages, a proposal implemented as an extension to the Chrome web browser. We also propose to use this extension as an entry point to long term browser history list preservation, using an hybrid local/cloud based personal digital repository that extends the amount of browsing information stored without affecting browser performance.

## 2 RELATED WORK

Supporting web page re-visitation is a research topic that throughout the years has been approached from two complementary perspectives: one attempted to study browser user's habits and create new re-visitation tools to support their habits, while the other attempted to improve the visibility and usefulness of existing re-visitation tools. The work of (Kawase et al., 2011) follows the first approach and introduces PivotBar, a Firefox based toolbar that provides dynamic contextual recommendations. These recom-

mendations attempt to remind the user about previously visited pages (supplied by Firefox's history list) that might be related with the page currently being visited and are selected using an algorithm derived from the study of a set of browser users habits.

The work of (Shirai et al., 2006) introduces the concept of history-centric browsing, which is supported by a tool (demonstrated in Internet Explorer) that allows the user to obtain information about previously visited pages related with the current page. Initially it displays an overlay near the top of the page with three controls that can be used to obtain information about previous visited pages using a temporal criterion (like the one used in traditional browser history list), a url proximity criterion (that allows to track down navigation within a given site) or a content similarity criterion (based on similarity between the words in the page title). Once the user chooses one of these criteria a semi-transparent overlay is used to display further informations, including thumbnail versions of the previously visited pages. The use of page thumbnails is also proposed in the work of (Won et al., 2009) as a way to improve the usefulness of traditional history lists. The work focused on increasing the visibility of the browser's history list with the produced prototypes being effectively an alternative implementation of this feature in the Firefox browser. One of the prototypes addresses the scenario where users attempt to retrace their steps using a search engine by displaying links to previously visited pages (along with their thumbnails) near the top of the search engine results page.

## 3 RDFa HISTORY EXTENSION

Semantic markup in its various forms (be it RDFa, microformats (Khare and Çelik, 2006) or industry initiatives<sup>1</sup>) appears to have a slow yet steady increase adoption rate, driven primarily by the need to optimise web pages for search engines' ranking algorithms (Davis, 2006), (Yu, 2011). When used, semantic markup is an intrinsic part of a page's underlying code that is transferred to a user's browser. While some forms of semantic markup (such as the correct use of header tags) have a direct visual impact on the page's presentation, most forms of semantic markup are not directly used by web browsers. These forms are silently ignored, and users are not aware of the presence of extra information in the visited pages. We propose that instead of discarding this extra information, a browser can use it to help

<sup>1</sup><http://schema.org/>

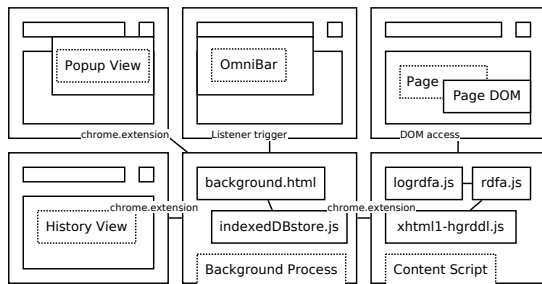


Figure 1: RDFa history extension architecture.

users revisit pages by making it accessible in the context of the browser's history list. This has the effect of giving access to users to one of the criterion search engines can use to discover web pages and to allow users to actually discover related information that they might have missed when a given page was visited for the first time. In order to gather and expose semantic information we have created an extension for the Chrome web browser (whose architecture can be seen in 1). Chrome extensions are bundles that contain web pages, images and JavaScript code that can be used to alter the behaviour of the browser and have access to both browser specific and standard web APIs. While Chrome extensions can use native code (using NPAPI), the created extension avoids its use in order to be a cross-platform extension. It should also be noted that the created extension respects the browser's private mode (*incognito mode*) and does not collect any information when the browser is used in this mode. The extension also supports disabling history collection from local files, secure http addresses or even individual pages.

To gather semantic information from visited pages the extension injects in each page a *content script* to access the page's DOM. The *content script* uses a modified version of W3C's RDFa in Javascript library (Adida et al., 2007) to extract semantic information in the RDFa format. The extracted information is sent to a background process (a background HTML page that is never displayed to the user) that acts as a mediator and serialises requests from the *content script* and user interface to an underlying indexed database (Mehta et al., 2011). Using an indexed database to store semantic information instead of a dedicated triple store helps the extension to be self contained by avoiding the need to use native code to communicate with an external database, yet it also prevents the extension from perform any reasoning over the collected semantic data or the use of a semantic query language such as SPARQL. The user interface has four components: a button placed on Chrome's main user interface (2a), an *omnibar* (Chrome's address and search bar) integration module (2b), an options

page for extension configuration and the semantic history list page itself (2c). The button placed on the user interface acts both as a non-intrusive indicator that reports if the current page contains valid RDFa (the number of statements found is shown in the button's tooltip) and the status of the parsing/storage of those statements in the indexed database; when pressed it allows the user to perform quick searches. The *omnibar* module is an alternative search method that is activated (or suggested to the user) by typing the string "rdfa" in the address bar. This module is able to provide suggestions (from the semantic history list) to users as they type, thus providing a more "guided" approach to history search. The semantic history list page is designed to mimic Chrome's native history list while offering more focused query options. While Chrome's native history list is able to perform a full text search and displays the results sorted by date/time, forcing users that know the approximated time the page they want to revisit was accessed to manually search within the results, the extension's history list allows users to specify the date range where the search should be performed, as well as fragments of the triples that might have been collected. This means that a user can search for pages that mention the title of a song without knowing the exact song title or the artist that performs it, and obtain as results pages containing song titles instead of pages that have the word title in it.

When displaying results the extension attempts to represent the vocabularies used in a human friendly form. The vocabulary to which a term belongs is identified by an icon, the term description is highlighted while the actual URI that identifies the term is downplayed (as seen in 2c). This representation is applied to recognised vocabularies (currently only FOAF<sup>2</sup>, Dublin Core<sup>3</sup>, Open Graph Protocol<sup>4</sup> and Creative Commons<sup>5</sup>). Each recognised vocabulary is implemented as a module, and might have vocabulary-specific actions (accessed by clicking on the vocabulary icon), such as displaying the licences under which a resource falls in the Creative Commons vocabulary, showing other related information like use guidelines and other associated licensing details.

## 4 IMPACT ON THE BROWSER

As was described in the previous section, each time a page is loaded by the browser, the extension injects in

<sup>2</sup><http://xmlns.com/foaf/spec/>

<sup>3</sup><http://dublincore.org/documents/dces/>

<sup>4</sup><http://ogp.me>

<sup>5</sup><http://wiki.creativecommons.org/RDFa>

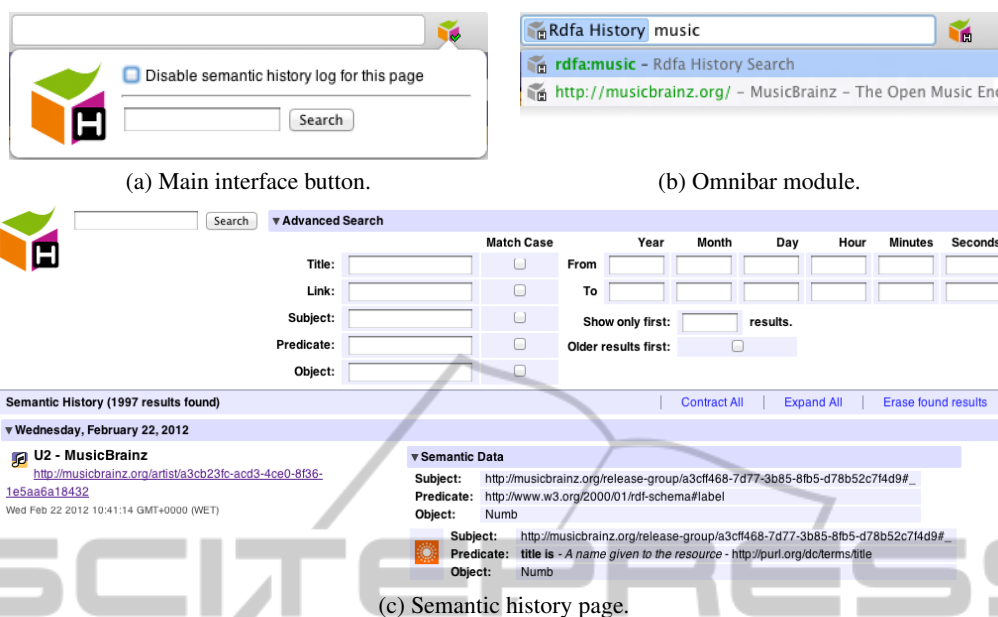


Figure 2: Interface elements.

it a *content script*, that is responsible for parsing and extracting semantic information from the page. The extracted semantic information is then relayed to the background page that manages the actual creation and storage of an history list entry in the indexed database. *Content scripts* are executed in the visited page's render process, and therefore can affect user's perception of how responsive the page (and the browser) is when using an extension. This extension's *content scripts* are injected (but not executed) by the browser immediately after the page's CSS file but before the DOM is constructed or any of the page's own scripts are executed. As such the extension introduces an increase in page loading time, yet since this increase is related to script initialisation and happens while other page scripts and elements are still being transferred to the browser its impact can be said to be negligible. The content scripts are executed in response to two DOM events, *DOMContentLoaded* and *DOMSubtreeModified*, that correspond respectively to when the visited page DOM has finished loading (but other elements such as images might still be loading) and when part of the DOM is modified. Since content scripts are executed in the same thread responsible for performing rendering actions needed by the page and to execute the page's own scripts, they might have a visible impact on user perception of browse responsiveness. The history list entry creation, although performed by the background page in another process, can also have an indirect impact in browser responsiveness perception, since each time an history list entry needs to be saved there is a temporary increase in CPU and mem-

ory usage.

To assess the impact of the of the content script and of the background page entry creation, we have measured the time taken by each of these operations when visiting a set of pages. The test was repeated 10 times for each page, with the average results displayed in 2. Looking at the time taken to parse a page we can see that the number of statements present in the page, while important, is not the decisive factor that increases parse time. Instead parse time increases with the complexity of the page's (X)HTML code, which means that complex pages such as Amazon or DeviantArt will require more time to parse even though they yield relatively few statements. Since (depending on the actual page) the content script will run while any images present are still being loaded the impact of the extension is masked by image loading time, with pages with multiple images or a single large image increasing this effect. As such although some of the parsing times are high we consider them to be acceptable and not overly intrusive. Regarding the entry creation time we can see that the time taken to create an entry scales near linearly with the number of statements extracted from the visited page, with entries for MusicBrainz taking nearly 1 second to be created. This operation is performed in its own background process and as such the user will not be directly aware of these processing times, yet if for some reason the browser is closed, any pending entry creation operations will be aborted and there will be a corresponding loss of information. Since history list entry creation operations are serialised by the back-

Table 2: Time taken to parse a web page and to store semantic information in the indexed database.

	Number of statements	Parse (ms)	Storage (ms)
Shopforia	3	51,3	67,2
Amazon	10	667,5	156,7
DeviantArt	10	439,9	123,2
Popular Mechanics	15	272,3	141,1
BBC	32	186,8	266,8
MusicBrainz	217	130,3	1024,8

ground page this issue can affect multiple pages at the same time if the user opens in quick succession multiple tabs with distinct pages that possess a large number of statements and then close the browser. A possible solution would be to have the extension listen to a browser close event and at least write a minimum history entry on that case, yet Chrome does not provide such an event in spite of being a common request<sup>6</sup>.

## 5 LONG TERM HISTORY

In order to prevent performance issues browsers limit the number of entries held by the history list. Opera keeps the last 1000 visited pages, Firefox determines the number of entries to keep based on computer specifications<sup>7</sup> and Chrome keeps an unlimited number of entries, yet it removes them automatically 10 weeks after the visit date. While this makes sense from a performance perspective it discards part of the user's on-line interactions, and can prevent the history list from helping users revisit a page if they are trying to look for a page visited either a long time ago or if the users happen to visit a large number of pages each day. Additionally since each page a user visits has the potential to provide a clue about that particular user's interests at a given point in time, letting this information disappear can contribute to user's "*personal digital dark age*".

Instead of letting entries in the history list fade away with time (or sheer number of visited pages) we propose an hybrid solution: store older entries in online personal digital repositories and when an explicit query is issued by the user include in the results relevant entries from both the local history and those stored in the personal digital repository. This approach has the advantage of preserving the entire history list for future use while still being compatible with browser's performance oriented policies. If this

<sup>6</sup><http://code.google.com/p/chromium/issues/detail?id=30885>

<sup>7</sup>[https://bugzilla.mozilla.org/show\\_bug.cgi?id=674210](https://bugzilla.mozilla.org/show_bug.cgi?id=674210)

extension is implemented in other browsers, users will gain access to their history list in a different browser or device. Furthermore in a personal digital repository the history entries can be associated with other types of information, for example documents, photos, music or emails, allowing users to construct a more precise image of why they visited a given page at that particular time. The downside of this approach is that the extension is no longer completely self-contained (since it requires access to an external server). To counter this downside the extension makes this functionality completely optional and requires users to activate and configure it (opt-in).

After configuring the extension to act as a long term preservation tool, the extension's configuration page can be used to manually export the entire (or part of the) local history to the personal digital repository. The extension can also be configured to automatically export the history list entries that are older than a given temporal criterion (by default one month). Conversely users can manually import part (or the entire) remote history list, or configure the extension to automatically import to local history either a set number of entries or any number of entries based on a temporal criterion (by default pages visited in the last month). When exporting the local history list it is the personal digital repository's responsibility to ensure that no duplicate entries are added to the cloud based history list, while when importing it is the extension's responsibility to deal with any duplicates it might receive. When the personal digital repository receives an export message it parses it and store both the semantic information as well as actual history entries in a triple store, enabling the use of reasoners to derive additional information and of SPARQL to perform complex queries. Queries to the personal digital repository can take the form of import requests or search requests: in an import query the extension merges entries received from the repository with the local history while on a search request the extension merges entries received from the repository with local search results to create a unified history list page.

## 6 CONCLUSIONS

In this paper we described the *Rdfa History* extension to Chrome web browser, an extension that aims to increase the visibility and usefulness of the browser's history list by enhancing it with the semantic information collected from visited web pages and by providing a tool that when coupled with a personal digital repository can serve as a gateway to long term history preservation. The extension is self-contained

and created using only standard (or working drafts) of interfaces available in HTML and Javascript. It alters Chrome's default interface by placing a button on the browser's main interface that serves the dual purpose of increasing the visibility of the history list and notifies users that the page they are currently visiting has semantic information available. It also integrates with the browser's address bar in order to provide users the opportunity to search in the history list as if they were navigating to a web page. Both of these interface modifications allow the user to perform quick searches that attempt to match the expression in any available fields, and display the results in a new browser tab. Results are presented to users following a layout as close as possible to the default layout of Chrome's native history list. Accessing the semantic history page allows users to perform advanced searches where they are able to construct queries that can limit the results by a specific triple pattern in addition to traditional date/time and title fields. The extension can be configured to act as a digital preservation tool by storing entries of the history list that would be marked by the browser as expired and removed from local history list in a personal digital repository. These entries can then be taken into account in any future queries performed by users, even though they are not part of the local browser history list. It should be noted that this feature is experimental and not enabled by default. Users must choose to use it, and by choosing to use it they have the option to send the entire (or part of the) local history list to the personal digital repository, or to import to the local history list part (or the entire) list from the personal digital repository using the extension's configuration page.

The *Rdfa History* extension is still an early prototype and as such there are improvements that can be made. One possible improvement is the inclusion of a thumbnail of each visited page in the history list page. Previous works (Won et al., 2009), (Aula et al., 2010) have shown that combining visual with textual clues provides better results than using any of those by themselves and can offer the visual context (at the epoch it was captured) when dealing with entries in the long term history list. Yet adding a thumbnail will result in a more pronounced visual distinction between the extension's history list implementation and the browser's default implementation, and will increase storage requirements. Early tests show that the impact of having to re-parse the entire visited page to extract semantic information appears to be limited by performing this task while loading other pages components, and since storage of history entries is done in a background process the impact of having to store a variable amount of information per visited

page does not have a direct impact on the perception of browser's responsiveness. While being able to enhance history list entries with semantic information without having a major impact on the browsing experience is important, usability tests are required in order to determine if the functionalities this extension introduces provide additional support for users trying to re-visit pages or even if it encourages users to increase their usage of the browser history feature.

## ACKNOWLEDGEMENTS

This work was funded in part by the Portuguese Foundation for Science and Technology grant SFRH/BD/62554/2009

## REFERENCES

- Adar, E., Teevan, J., and Dumais, S. T. (2009). Resonance on the web: web dynamics and revisitation patterns. In *Proceedings of the 27th international conference on Human factors in computing systems*, CHI '09, pages 1381–1390, New York, NY, USA. ACM.
- Adida, B. and Birbeck, M. (2008). *RDFa Primer: Bridging the Human and Data Webs*. W3C. <http://www.w3.org/TR/xhtml-rdfa-primer/>.
- Adida, B., Yergler, N., and Tension, J. (2007). *RDFa in Javascript*. W3C. <http://www.w3.org/2006/07/SWD/RDFa/impl/js/>.
- Aula, A., Khan, R. M., Guan, Z., Fontes, P., and Hong, P. (2010). A comparison of visual and textual page previews in judging the helpfulness of web pages. In *Proceedings of the 19th international conference on World wide web*, WWW '10, pages 51–60, New York, NY, USA. ACM.
- Cockburn, A. and Mckenzie, B. (2001). What do web users do? an empirical analysis of web use. *Int. J. Hum.-Comput. Stud.*, 54:903–922.
- Davis, H. (2006). *Search engine optimization: building traffic and making money with seo*. O'Reilly, first edition.
- Herder, E. (2005). Characterizations of user web revisit behavior. In *LWA 2005, Lernen Wissensentdeckung Adaptivität*, pages 32–37, Saarland University, Saarbrücken, Germany. German Research Center for Artificial Intelligence (DFKI).
- JasonSmith, M. and Cockburn, A. (2003). Get a way back: evaluating retrieval from history lists. In *Proceedings of the Fourth Australasian user interface conference on User interfaces 2003 - Volume 18*, AUIC '03, pages 33–38, Darlinghurst, Australia, Australia. Australian Computer Society, Inc.
- Kawase, R., Papadakis, G., Herder, E., and Nejdil, W. (2011). Beyond the usual suspects: context-aware revisitation support. In *Proceedings of the 22nd ACM*

- conference on Hypertext and hypermedia*, HT '11, pages 27–36, New York, NY, USA. ACM.
- Khare, R. and Çelik, T. (2006). Microformats: a pragmatic path to the semantic web. In *Proceedings of the 15th international conference on World Wide Web*, WWW '06, pages 865–866, New York, NY, USA. ACM.
- Kumar, R. and Tomkins, A. (2010). A characterization of online browsing behavior. In *Proceedings of the 19th international conference on World wide web*, WWW '10, pages 561–570, New York, NY, USA. ACM.
- Mehta, N., Sicking, J., Graff, E., Popescu, A., and Orlow, J. (2011). *Indexed Database API*. W3C. <http://www.w3.org/TR/IndexedDB/>.
- Obendorf, H., Weinreich, H., Herder, E., and Mayer, M. (2007). Web page revisitation revisited: implications of a long-term click-stream study of browser usage. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, CHI '07, pages 597–606, New York, NY, USA. ACM.
- Shirai, Y., Yamamoto, Y., and Nakakoji, K. (2006). A history-centric approach for enhancing web browsing experiences. In *CHI '06 extended abstracts on Human factors in computing systems*, CHI EA '06, pages 1319–1324, New York, NY, USA. ACM.
- Tauscher, L. and Greenberg, S. (1997). How people revisit web pages: empirical findings and implications for the design of history systems. *Int. J. Hum.-Comput. Stud.*, 47:97–137.
- Weinreich, H., Obendorf, H., Herder, E., and Mayer, M. (2008). Not quite the average: An empirical study of web use. *ACM Trans. Web*, 2:5:1–5:31.
- Won, S. S., Jin, J., and Hong, J. I. (2009). Contextual web history: using visual and contextual cues to improve web browser history. In *Proceedings of the 27th international conference on Human factors in computing systems*, CHI '09, pages 1457–1466, New York, NY, USA. ACM.
- Yu, L. (2011). *A Developers Guide to the Semantic Web*. Springer Publishing Company, Incorporated, 1st edition.