Usability aspects of the inside-in approach for ancillary search tasks on the web

Marco Winckler, Ricardo Cava, Eric Barboni, Philippe Palanque, Carla Freitas

HAL Id: hal-01343048
https://hal.science/hal-01343048

Usability aspects of the inside-in approach for ancillary search tasks on the web

Marco Winckler[2], Ricardo Cava[1], Eric Barboni[2], Philippe Palanque[2], Carla Freitas[1]

[1] Institute of Informatics - Federal University of Rio Grande do Sul
Av. Bento Gonçalves 9600, PB 15064
91501-970 Porto Alegre, RS, Brazil
{racava, carla}@inf.ufrgs.br

[2] ICS-IRIT, University of Toulouse 3, 118, route de Narbonne
31062 Toulouse Cedex 9, France
{palanque, winckler, barboni}@irit.fr

Abstract. Given the huge amount of data available over the Web nowadays, search engines become essential tools helping users to find the information they are looking for. Nonetheless, search engines often return large sets of results which must be filtered by the users to find the suitable information items. However, in many cases, filtering is not enough, as the results returned by the engine require users to perform a secondary search to complement the current information thus featuring ancillary search tasks. Such ancillary search tasks create a nested context for user tasks that increases the articulatory distance between the users and their ultimate goal. In this paper, we analyze the interplay between such ancillary searches and other primary search tasks on the Web. Moreover, we describe the inside-in approach, which aims at reducing the articulatory distance between interleaved tasks by allowing users to perform ancillary search tasks without losing the context. The inside-in approach is illustrated by means of a case study based on ancillary searches of coauthors in a digital library, using an information visualization technique.

Keywords. Interaction gulfs, Web search, ancillary queries, nested user tasks.

1   Introduction

According to Hilbert and López [8], in 2007 almost 94% of our memory was already in digital form and most of it can be found through the Web nowadays. Over the years, users have become used to retrieve information from the Web, and for that they developed several strategies, which can be summarized as information lookup and exploratory search [9]. Whilst exploratory search requires time for scanning and reading documents, information lookup can be solved by simple factual question-answer interactions. Moreover, in this context of huge amount of data, information retrieval systems and search engines have become an integral part of our daily lives [7].

The user interface provided by such information retrieval systems must be simple enough to allow users to formulate queries and understand the results provided by

search engines [10][20]. Nonetheless, many users are still struggling to use them to obtain the results they need [6]. Many of the problems users have to face are related to increasing availability of data in the Web. For that, users must be very precise in the way they formulate their queries, and they must know how to interact with the display to identify the sought results in the large set of data.

Quite often, queries start by filling a search box with keywords. Formulating queries in this way is a daunting task that requires users to open a new window, to fill in a form with appropriate keywords, and then scan the list of results until finding the one that corresponds to their goal. Moreover, the standard way to display search results (obtained by either filling in a form or browsing documents) often imply to display in a new window/tab and/or replace the current window/tab's contents, which might deviate the users' focus of attention and creates an interruption between nested tasks.

As we will see, while some search tasks can be directly associated to a user's primary goal, many other searches are nested in other tasks and simply require to complement information they are currently reading [7]. For example, users reading an article in a Web page might be curious to know with whom the author of that particular article has published in the past. In this scenario, looking up for co-authors constitutes an ancillary search, which is not meant to divert users' attention from reading the article. For such kind of ancillary-search tasks, displaying results in a new window/tab might be unsuitable since it creates an articulatory distance between the origin of the request and the information display, making difficult to users to assess if their goal has been fulfilled or not by the query.

In this paper we claim that whilst the design of existing tools might fit for the purposes of primary search tasks, users still need better support for performing ancillary search tasks. This claim is supported by a model-based task analysis presented in section 2. Based on the Don Norman's cognitive model [15], we discuss in which extension the design alternatives for performing ancillary search tasks might increase cognitive gulfs of execution and gulfs of evaluation. Based on the lessons learned from the tasks analysis we have devised an alternative approach, called *inside-in* search, which aims at reducing the articulatory distance between interleaved tasks by allowing users to perform ancillary search tasks without losing the context. The inside-in approach is presented in the section 3. In order to demonstrate the feasibility of the *inside-in* approach, we have developed a supporting tool that is described in section 4. The last sections of the paper present related work, the conclusions and future work.

2   Task analysis of Web search tasks

In order to support the analysis of users' tasks we employ the model-based notation HAMSTERS, which stands for Human-centered Assessment and Modeling to Support Task Engineering for Resilient Systems [11]. Similar to CTT [16], HAMSTERS provides constructs (i.e., typology of tasks and operators) that allow fine-grained description of tasks as well as the simulation of models. Moreover, thanks to appropriate tool support, HAMSTERS models can be used to predict users' performance with interactive systems, which can be understood as an indirect (or simulated) knowledge about human behavior. Whilst a detailed description of the notation HAMSTERS is out of the scope of this paper, we provided the necessary information for understanding our models. Further details about HAMSTERS can be found elsewhere [11][12].

2.1 Overview of search tasks over the web

As argued by Yates and Neto [23] users can search information over the Web either by *navigating documents* and/or using specialized *information retrieval tools* (the so-called search engines). As s................................are distinct, they are interconnected. On one hand.....................................s is to direct users to Web pages that contain the...................................nd, documents might contain links embedding qu...............................s, which are aimed at helping users to find comple...........................lily available through a simple navigation [5].

point to document
that contains the
sought
information

search
details/further
information about
what is present in
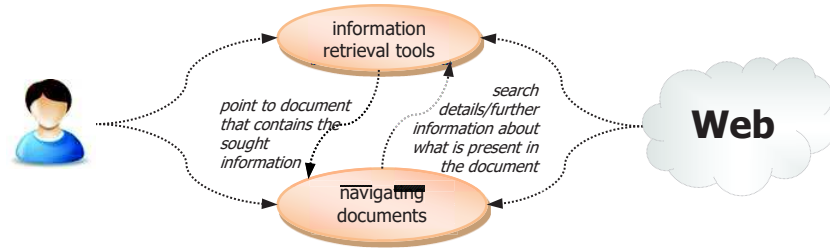the document

Web

navigating
documents

Fig. 1. Overview of alternative strategies for finding information; adapted from [23].

The task model described herein aims at analyzing search tasks from a high level of abstraction that do not imply any particular implementation of tools. Indeed, we assume that it would be possible for users to perform searches either by using a dedicated information retrieval tool and/or by triggering a search directly from a Web document. Thus, regardless the information retrieval algorithms and users' needs for information, a search can be summarized as a set of the following subtasks: at first, users *formulate a query*, then the system will *process the query* and *show the results* to the users which can, at this point, to *refine the results* until selecting the appropriate entry that corresponds to the sought information. Moreover, users can decide for any reason, and at any time, to *stop the search*.

One of the main advantages of task model notations such as HAMSTERS is to support the iterative decomposition of tasks in a tree-like structure to reach the level of detail that is required for the analysis. The corresponding modeling of search task using the notation HAMSTERS is illustrated in **Fig. 2**. Notice that the top-level task *search* is decorated at its left side with the symbol ▦, to indicate that *search* is an iterative task that can be repeated indefinitely by the user. Subtasks are connected by the operator >> that determines the sequence of task execution. The operator [> associated to the task *stop query* indicates that, when this task is performed, it interrupts the sequence of other subtasks. Notice that specific icons are used to indicate the typology of tasks; for example, *process a query* is typically automated by the system, *stop search* is typically a user task, which might require a simple user decision, and tasks such as *formulate a query, show results* and *refine results* require user interaction with the system to be performed, so they are also called interactive. The symbol ▦, next to the task *refine results,* is used to indicate that this task is optional.
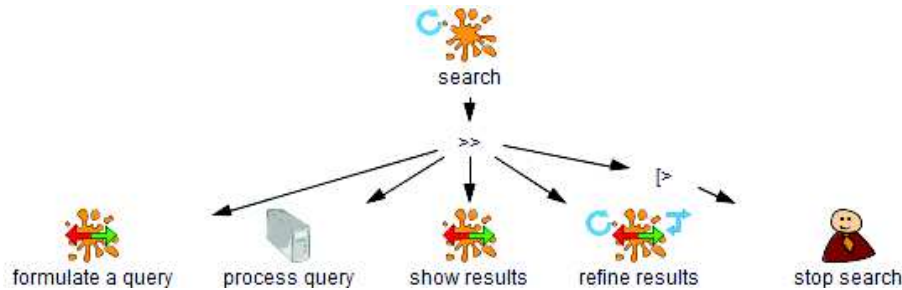
Fig. 2. Overview of a Web search task described using the notation HAMSTERS.

Fig. 3 shows a further level of task decomposition with alternative ways to perform some tasks. As for the task *formulate query,* we can identify additional subtasks including *provide keywords* and *select a database*. The operator |=| indicates that these tasks can be performed in any order. It is worthy to notice that the task *provide keywords* is decomposed in alternative subtasks: *type keyword* and *select word*. The choice between tasks is indicated by the operator []. Some of the alternative tasks consider the case of manual versus automated execution. For example, the tasks *select database* can be done by prompting the user to explicitly *identify a database*. The automated alternative task assumes that the system *uses a predefined database* and the user cannot change it.

The next task is to *create the display,* which is shown as a simple *output* task in the HAMSTERS' taxonomy. The creation of the display offers several alternatives to perform the task *interact with display*. Basically, from this point, user tasks depend on the location of the display and the number of entries in the set of results. For the location, users might be led to *use new window* or to *use the current window*. For the number or entries, a user can either *get all at once* (i.e. all results appear in the same display) or *browse subset* (i.e. results are divided in subsets that can be navigated by the user).

Users can adopt two main strategies to *refine results*: to apply *filtering* mechanisms or to perform a *(new) search*. A *(new) search* task follows the same pattern of a full *search task,* which means that search tasks can be recursive. To indicate that a subtask corresponds to a pattern of a known task, HAMSTERS provides a particular type of construct called *copy task,* which is clearly indicated in **Fig. 3** as decorations around the tasks *(new) search*.

It is noteworthy that only leaf tasks in a HAMSTERS model are ever executed. The level of decomposition of tasks in HAMSTERS is arbitrary, but we assume that the details provided in **Fig. 3** are sufficient to illustrate our analysis.
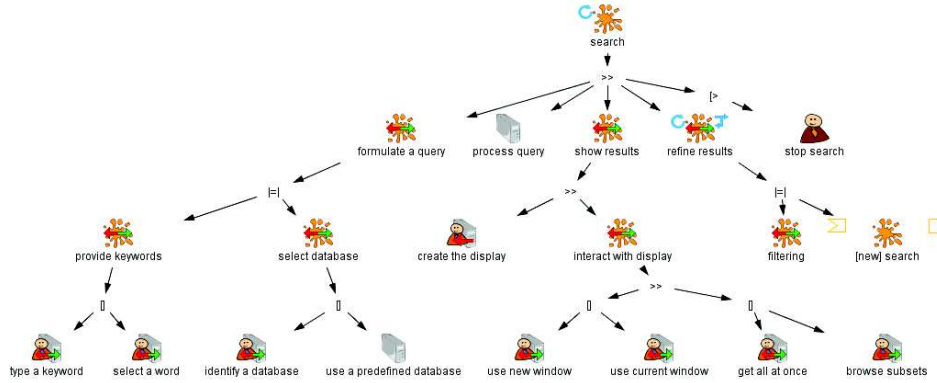
Fig. 3. Expanded view of user tasks including alternatives ways for performing a search task.

2.2 Overview of Web search as primary task or ancillary tasks

By observing users behavior we found that they often have many searches running in parallel (or executed in very short time intervals). Whilst some searches might refer to completely disjoint user's goals (for example, look for a restaurant in town for tonight and plan a trip for the weekend), other searches are indeed part of an overall primary user goal (for example, search for a hotel, then search for a flight whilst planning a trip). It is also interesting to notice that many parallel searches don't correspond to a primary user's goal but they are performed to get information for achieving a previous task (for example, looking for currency exchange rates to calculate prices given in a foreign currency whilst booking a hotel). **Fig. 4** illustrates the differences between search tasks according to users' goal, which can be formalized as follows:

- *Primary search tasks,* which correspond to a user's primary need for information. Ideally, primary searches encompass a single cycle of question-answer interaction with the search engine. Nonetheless, if the results are not satisfying, users ought to reformulating the terms used in the query and perform a new search. It is interesting to notice that users might perform many queries but every query is treated as unique by the system. As a consequence, the entries in the results provided by different searches might highly differ according to the keywords used.

- *Ancillary search task*, which are aimed at providing details on demand about the results that are current in display. *Ancillary searches* depend on the results obtained from previous search and/or available information over a Web page. Ideally, once users find the answers he should be prompt to return to the context of the task he was performing before launching the *ancillary search*.
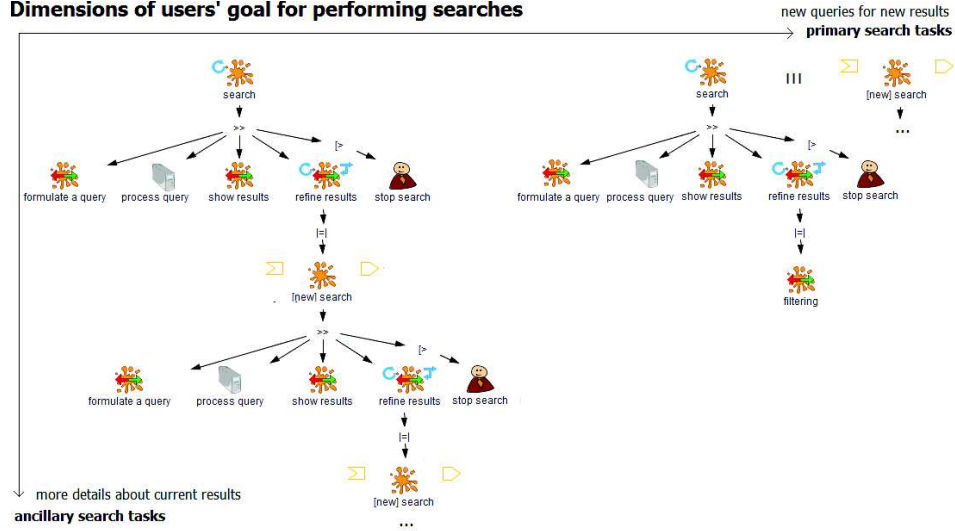
Fig. 4. Tasks models in HAMSTERS illustrating dimensions of users' goal for performing search tasks, thus featuring nested *ancillary search tasks* and disjoint *primary search tasks*.

Is worthy of notice that *primary search* tasks are typically treated as disjoint tasks by the system, so they can occur in parallel. However, *ancillary searches* are deeply dependent of existing contents for which it is aimed for providing further details. It is interesting to notice that *ancillary search* leads to nested queries that create a trail of searches performed by the users while *primary search tasks* are treated independently by the system. Both types of tasks can be combined according to the user's actual needs for information. Thus, a *primary search task* can run in parallel with other searches that require the decomposition into *ancillary searches*.

2.3   Execution and evaluation goals in Web search tasks

The tasks that users perform during a search establish a type of question/response communication between the user and the system. Based on Don Norman's cognitive model [15], it is possible to assess the communication mismatch between user's internal goals for performing a search task and the user's expectations with respect to the availability of information specifying the state of the world (the Web). Communication mismatches occur in terms of inputs (i.e., gulf *of execution*), output (i.e. gulf of *evaluation*), or both [14]. In order to illustrate the meaning of the *execution gulf* and the *evaluation gulf* in search tasks, we present in **Fig. 5** a revised version of the Norman's cognitive model explicitly showing the *articulatory distance* and the *semantic distance,* both in terms of user *input* (i.e. when users formulate the query) and in terms of system *output* (i.e. when the system shows the results to be assessed by the user).
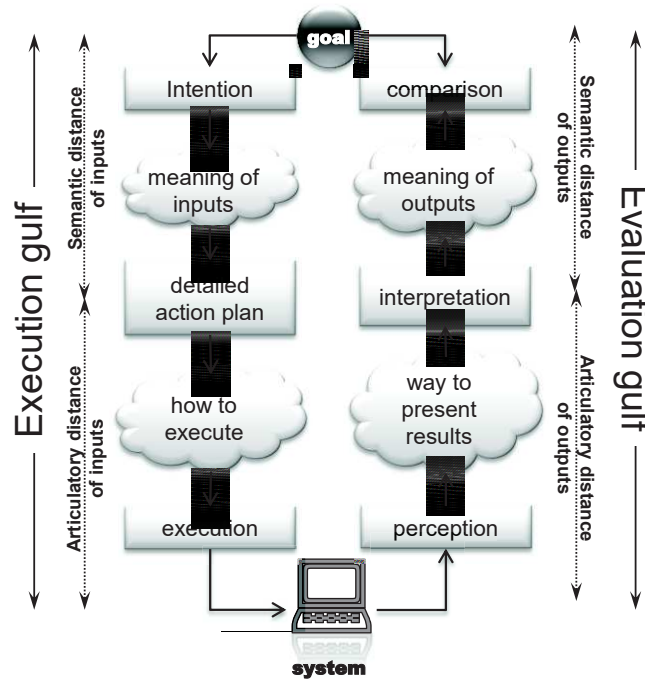
Fig. 5. Execution and evaluation gulfs in search tasks, adapted from [15].

The length of the *gulf of execution* is described by Norman as the difference between the intentions of the users and what the system allows them to do or how well the system supports those actions. In the case of a Web search, if the users find an unknown word while navigating the web, they might expect that clicking on a link (on that word) would provide them with the complementary information required to understand the meaning of the word. In the user's language, "click the link" defines the goal for obtaining the word's meaning. However, if the link does not provide the expected results, users have to execute additional actions, such as opening a new window, visiting a search web site, typing the adequate keywords to specify the search, and, finally, browsing the list of results until getting the desired definition.

The *gulf of evaluation* refers to the way the results provided by the system are meaningful or understandable by the users and in accordance with their goals. In other words, the *gulf of evaluation* is the degree to which the system or artifact provides representations that can be directly perceived and interpreted in terms of the user's expectations and intentions. Thus, if the system does not "present itself" in a way that lets the users derive which sequence of actions will lead to the intended goal, or infer whether previous actions have moved them closer to their goal, there is a large *gulf of evaluation*. In our case, users must spend a considerable amount of effort and significant attentional resources to perform a query in a new window, to identify the answers that correspond to their expectations and, then, to put the word's meaning back in the appropriate context.

Table 1 shows a comparison between the user interface alternatives for tasks (actually, we only take into account the leaf subtasks). A large *execution gulf* is expected when users have to *formulate a query*. Nonetheless, the semantic distance for the task

select word is smaller than *type keyword* because recognition of words is less cognitive demanding than choosing the words. The *articulatory distance of input* is also shorter for the task *select a word* as it just requires a click, which is also faster than *typing keywords*. Moreover, users can make mistakes (e.g., use the wrong word) and slips (e.g., introduce typos) while typing keywords.

Table 1. Tradeoffs of design alternatives for performing tasks in a Web search.

Subtasks		User interface alternatives for tasks	Execution gulf		Evaluation gulf	
			Semantic distance of input	*Articulatory distance of input*	*Articulatory distance of output*	*Semantic distance of output*
formulate a query	provide keywords	*type a keyword*	know how and where to type keywords, know the spelling	require a fill-in -form, typing is time consuming and error prone	the size of the fill-in-form should accom-modate the key-word	require recognize what was typed
		select a word	recognize a word is easier than recall it	might require a single click	perceive a word as link	recognize the selected word
	select database	*identify a database*	user must know the database	require user interaction, error prone	see the source of information (database) of the items in the results	recognize the source of the results
		use prede-fined database	no user input required			
show results	interact with display	*use new window*	predict where results will be shown	move to the new window to see results	require to locate new window in the display	manage multiple windows is cognitive demanding
		use current window		keep focus on the display	require to locate results in the current display	replace/superimposing existing content might be confusing
		get all at once	users might be prompted to select where they prefer to see results	might require a simple selec-tion	require to locate scroll in the display and eventually use it	require the identification of the sought item, the best option depend on the posi-tion of the items in the set of results
		browse subset		might require (or not) users to set the number of entries in the subset	require to locate interactors, navigate between subsets and eventually use it	
refine results		*filtering*	know how to apply filters	select filters	perceive filters and results	recognize results that corre-spond to filters
		[new] search	require a new instance of search task…			

Legend: short distance,   *large distance,*   *depends on the context.*

For the *evaluation gulf,* users have to pay attention to be sure that the keywords they provide appear in the display properly with no error. With respect to the alterna-tives for *selecting a database,* to *identify a database* (either by typing the name of the information source or selecting it from a list) has larger *semantic* and *articulatory* dis-

tances of input when compared to the *use of a predefined database* that is known by the system and do not require any user interaction. For the *evaluation gulf*, we can assume that semantic and articulatory distance are similar in both cases since users must be able to see and recognize the sources of information whatever it is.

The tasks *interact with display* implies alternatives for the location of the display (*new window* or *current display*) and for the number of entries in the set of results (*get all [results] at once* or *browse subsets*). As far as location is a concern, we might say that all options have similar semantic distance of input, since users must be able to predict where results will be shown. However, *use current window* for showing results is less demanding in terms of articulatory distance of input as users can keep the focus on the display and do not need to move to another window. Using a new window for the display would also require users to locate where results are located in the display, which increases articulatory distance of output. For the semantic distance of output, *use new window* requires to manage multiple windows while *use current window* implies that previous content of the window is lost or new content is superposed to the existing one, which might be confusing. The relative advantage of these options can only be decided once the context for the user search is known.

For the task *refine results,* there are two options: *filtering* mechanisms and perform a *(new) search*. Many *filtering* mechanisms exist and a deep analysis of them is out of the scope of this paper, but we can assume that filtering might help to locate an item of information in the set of results. Nonetheless, when the sought information cannot be found in the set of results provided by the search engine, the only alternative users have is to make a *(new) search*. As we will see, the execution of a *(new) search* is recursive, and requires users to go through all the subtasks starting by formulating a new query. For all these reasons, we consider that making a *(new) search* always requires extra effort from users when compared to exploring results that users might expected to find in the display.

Table 2 provides an analysis of the tradeoffs between the two alternative strategies for performing a search task. As we can see, users might perform a *(new) search* as a *primary search task* or as an *ancillary task.* If the *search task* is performed as a *primary task*, users have to manage the articulation of possibly disjoint searches performed in parallel and formulate new queries from scratch. Assuming that users expect to keep the results of parallel searches separated from each other, the steps required for performing a *search task* do not have an extra impact in the execution and evaluation gulfs.

Table 2. Tradeoffs between Web searches as primary and ancillary tasks.

Web search	Execution gulf		Evaluation gulf	
	Semantic distance of input	*Articulatory distance of input*	*Articulatory distance of output*	*Semantic distance of output*
primary task	decide on disjoint searches	formulate new query from scratch	perceive the contexts of new searches	identify results in the display
ancillary task	keep in mind the nested searches	formulate new queries to refine previous results as input	follow the nested results as part of the context of a single task	recognize results in the display as part of a nested search

However, if the search users want to perform is an ancillary task, some design options that would be acceptable for a *primary* search can dramatically increase the execution and the evaluation gulfs. As for the input, semantic distance is larger because

users have to keep in mind that the *(new) search* is nested in the context of another task. Moreover, users should be able to indicate which part of the results should be refined in a new query, which might increase *articulatory distance of input* if users cannot directly select keywords from the existing results in the display. As for the evaluation gulf, users need to perceive the new results as nested in a previous task which, without proper design, can be challenging and increase *semantic distance of the output*. Finally, *semantic distance of output* can be increased if users are not able to easily recognize the results in the display as being part of a previous task.

2.4   Coordination of multiple Web searches

Every search task creates a context of use that requires users' attention for formulating a query and interacting with the display. Moreover, according to the users' goals and needs, *Web searches* might become iterative and/or nested tasks. The occurrence of multiple searches running in parallel or at least in short time intervals is an additional source of dispersion of users' attention. However, for determining whether (or not) coordination between multiple search tasks is needed one should consider the users' goals.

Indeed, there are many situations where creating new contexts for a search task is not perceived as a major issue for the users: for example, when users want to test keywords in specific queries, or when users want to compare the results of two searches preserving the original context, or when parallel searches have no dependency with other user tasks. These are typical examples of *primary search* tasks. In these cases, coordination between tasks occurs in the user's mind since keeping the search disjoint corresponds to a user's goal. However, when performing *ancillary search tasks,* what users want is to obtain further details about information already in the display, so there is a clear dependency between what users see on the screen and the results they expect from the new search. It is interesting to notice that every time the user has to perform an *ancillary search task,* this is actually an interruption of a task that cannot be accomplished due to lack of the necessary information. In such cases, creating new contexts might be perceived as misleading as it increases the semantic and articulatory distances of the input (i.e. execution gulf) by dispersing users' attention from the primary task towards an *ancillary search task*. The dispersion of user attention will still increase in terms of output, and requesting users to interact within multiple contexts will break the inner dependency between the users' tasks adding an unexpected interruption. As discussed before [22], resuming a task after an interruption is difficult and may take a long time; interrupted tasks are perceived as harder than uninterrupted ones; interruptions cause more cognitive workload, and they are quite often annoying and frustrating because they distract people from completing their work.

3   Inside-In approach for ancillary search tasks

The ultimate goal of the inside-in approach is to mitigate the effect of some subtasks that have been identified as difficult to accomplish when performing ancillary searches, such as: to formulate the query, use a new window to interact with the results, and keep a straightforward context for nested search tasks. We start the description of a working scenario that characterizes an ancillary search task.

3.1 Working scenario

In order to ground the scenarios around the same application domain, we have chosen to illustrate them with data about co-authorships, as follows:

> *"John has been recently appointed as expert member of the jury that will evaluate the research of a Graduate Program in Computer Science. John received a Web form which contains the list of ~400 researchers for which he has to provide an assessment based on the number of co-authors and relevant publications. The number of publications and co-authors is required to calculate two important metrics: the researcher's productivity (accordingly to a formula that takes into account the number of co-authors to estimate the individual effort for the publication) and the size of collaboration network (considered that successful scientific collaborations ultimately lead to joint publications). So, John starts by making a Google search on the Web using the name of the first researcher in the list. Finding the right researcher's Web page was not easy as the Google search engine returns many entry points including homonymous and some trash pages. After fixing typos and refining the terms of the query, John finds the researcher's Web page where he can count the number of his publications; the mental calculation to accomplish this task is easy. Now, for assessing the size of the research network, things are more complicate. John considers two options: i) to create manually a side-list with the names of co-authors; or ii) to look for them in the DBLP web site. John chooses the second option, so he types the name of the researcher on the search box of the browser, goes to DBLP web site, scrolls down to reach the zone where co-authors are displayed, and open up the list of co-authors. Now, John is ready to fill in the form, but then he realizes that the DBLP content now occupies the window that previously contained the Web form… For the next 399 researchers John decided to create new tabs for keeping the DBLP search apart from the Web form. Then, he finds out himself being performing repetitive copy-and-paste between tabs, which definitely does not improve his overall performance…"*

3.2 Rationale for the inside-in approach based on the scenario

In our working scenario, searching co-authors is an ancillary search that complements the user's main task, which is filling in the Web form. From this scenario, we find some issues that make the following users' tasks difficult:

- Formulating queries is error-prone (might contain typos) and also time consuming (typing takes time).
- Keywords might be ambiguous, and generic search engines will return broad results. Users may have to scan the list of results until finding the ones that correspond to their goals.
- There are many alternative locations for showing results (including new windows/tabs); choosing the best location for displaying results depends on where the results are meant to be used.
- Some queries might be repetitive; so, saving a few seconds in the time required to complete a single task might represent a huge improvement in the overall task performance at the end of the day.

We claim that the issues raised above can be solved (or at least minimized) with our inside-in approach including the following mechanisms aimed at supporting ancillary search tasks:

- Launching queries from words available in the current Web page can reduce typos. Keywords can be selected with mouse clicks, which is sensibly faster than typing in using a keyboard.
- Ambiguous results are often the result of a broad search. This problem can be reduced by providing specialized queries that operate on specific application domains using user-selected keywords.
- Query results can be shown inside the current page, inline to the selected keywords. This is one of the keystones of the inside-in approach, but queries should be launched on user's demand. If the system systematically launches queries without user's request, the Web page will become polluted by ancillary results, and the benefits of the inside-in approach will be lost.
- Results obtained from ancillary searches should support some kind of interaction to allow users to perform nested queries. This element is important for repetitive tasks, which are often associated to contexts where ancillary searches are required.

The selection of keywords in the text and the use of predefined queries aim at reducing the gulf of execution. This reduction is achieved by minimizing the users' effort in informing keywords to the system (articulatory distance of inputs) and by favoring recognition of keywords and queries rather than imposing the formulation of complete queries (semantic distance of inputs). Predefined queries also help to reduce the evaluation gulf as the results are focused on a particular application domain (semantic distance of outputs). By showing results in the same page and allowing the user to perform nested queries, the inside-in approach helps to reduce the articulatory distance of outputs.

For the sake of simplicity, the scenario used in this section is minimalist and only covers a single level of nested search. However, based on the same principle it would be possible to extend the number of nested search tasks.

4 Tool support for the inside-in approach

In this section we present the set of the tools, featuring a framework, which we have developed to demonstrate the feasibility of the inside-in approach. Later on, in this section, we provide some preliminary results we have obtained in an empirical study using our tools.

4.1 Architecture

The overall architecture of the framework is briefly illustrated in **Fig. 6**. It was built upon the concept of Web augmentation [4], which defines strategies for implementing tools that can extend the set of elementary tasks users can do while navigating on the Web. Whilst the full details about the implementation of that framework are out of the

scope of this paper, it is interesting to notice that it includes a client-side module and a broker at the server-side.
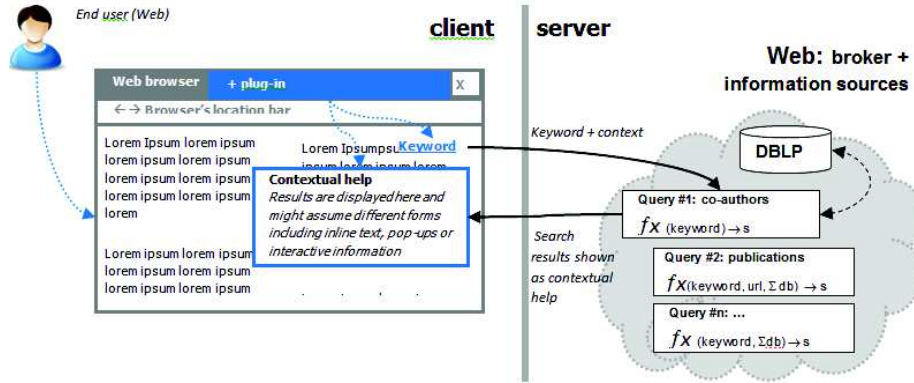

Fig. 6. Overview of the framework architecture for supporting ancillary searches over the Web.

At the server side we have developed a broker, which can be connected to many information sources. This broker was only implemented to illustrate the inside-in approach, and it could eventually be replaced (or connected) with any search engine. The broker contains a set of preprogramed query functions that are made available to the end users [3]. The set of predefined queries is large, and it aimed at to fit different users' goals for ancillary tasks, for example: finding co-authors, finding publications of a particular author, etc. The number of queries accessible from the client-side can be configured dynamically, but for the purposes of this paper we are just using a specific one, which returns the co-authors of a given researcher. These queries as well as the choice of DBLP database as information source of results are totally arbitrary but justified by the fact that they are related to our case study. Indeed, many different queries are possible to match an ancillary search with diverse users' goals. We suggest that the broker would embed some kind of intelligent behavior for suggesting ancillary searches according to the users' previous search. Nonetheless, in the current implementation of the Web broker we have not integrated any recommender system yet. So, the selection of the predefined search tasks is currently done on the client-side tools.

For the interaction at the client site we have developed a client side module that can be installed as a plugin of the Web browser. This module allows users to select keywords in the current web page and to trigger the queries for ancillary search available at the server side. Once the broker replies, this module modifies the current Web page to display the search results as a kind of contextual help. For that, the DOM structure of the current Web page is adapted using a set of JavaScript functions, called augmenters [4]. As demonstrated in a previous work [25], adaptations created by augmenters are volatile and do not affect the application running in the Web server. The client-side module can display ancillary data using different interaction techniques, including simple interaction tables displaying a simple list of results or more complex information visualization techniques such as IRIS, which is presented below.

4.2   User interaction with client-side tools

Once the client-side module is running, it grants access to predefined queries available in the broker. Ancillary search queries can be executed by simple selection of words available in any Web pages as shown at **Fig. 7**.a. With a right-click, the user can select a list of predefined queries and trigger the ancillary search at server side. Query results are returned to the client and displayed in the form of contextual help that superimposes existing contents on the web page, as shown by **Fig. 7**.b. If no co-author is found for the selected word, the JavaScript function popups an error message informing the user.
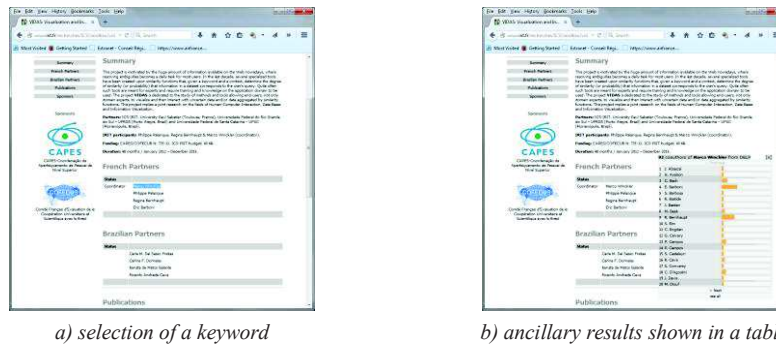

a) selection of a keyword   *b) ancillary results shown in a table*

Fig. 7. Overview of user interaction with the client-side tool: a) the selection of a keyword over the Web page; b) the ancillary query results shown in a tabular form next to the keyword.

As shown in **Fig. 8**, the name of the author used as keyword is placed at the topmost row of the table, which also includes the total number of results found. The first column represents co-authors ordered by name, and the second one contains a bar graph depicting the ranking of co-authored papers between the author and each co-author. The details for a ranking can be obtained via tooltips, as shown by **Fig. 8**.b.

We can also see in **Fig. 9**.a that the table features 20 items. For more results users can navigate the subsets using the links "∧ *previous*" and "∨ *next*", or get all results via the option "*see all*". Notice that the option "20 results" requires browsing subsets, while the option "*see all*" requires users to scroll down, as shown in figure **Fig. 9**.b.
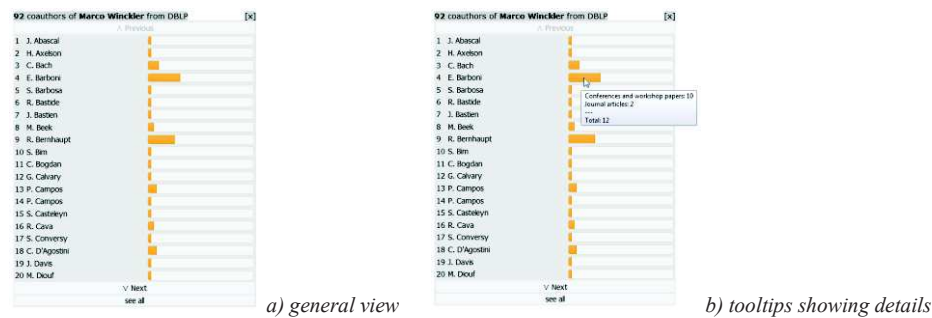

a) general view   *b) tooltips showing details*

Fig. 8. Ancillary results in a tabular form: a) general view; b) tooltip with details of ranking.

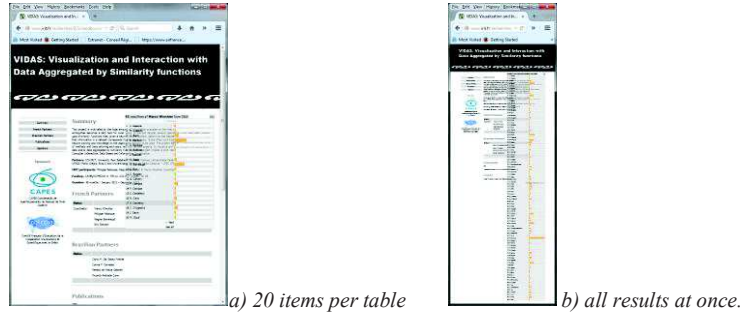*a) 20 items per table*   *b) all results at once.*

Fig. 9. Presentation of tabular view showing: a) 20 items per table; b) all results.

The display of the tabular view for showing ancillary results can be easily customized, for example, by modifying the number of columns to show more results. But more important than the customization of the table view is the possibility of replacing it in the client-side module by another type of visualization that would be a better fit to display ancillary results. Indeed, the inside-in approach we propose does not impose any kind of particular visualization. To illustrate this point, we show in **Fig. 10** the same results using a visualization technique called IRIS (which stands for Investigating Relationships between Indexes of Similarity), which features a radial layout and provides different interactors for exploring the set of ancillary query results, including animated transitions and additional tooltips.
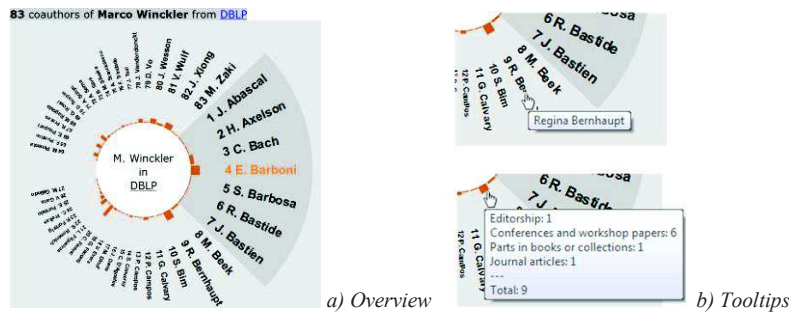

a) Overview   *b) Tooltips*

Fig. 10. IRIS visualization of co-authorship of information from DBLP.

4.3   Preliminary results

In order to test our tools we have run a remote empirical study with end-users. The evaluation was designed around a scenario where, given a list of names of researchers in a web page, users should perform ancillary tasks for checking co-authorship at the DBLP. Users were allowed to perform the same tasks by visiting the DBLP web site (http://www.informatik.uni-trier.de/~ley/db/) and by using our tools. The list of co-authors was obtained by parsing data from DBLP and displaying it using IRIS as shown in **Fig. 10**, so that users will get exactly the same data.

Users should start at the Web page members of VIDAS' project shown in **Fig. 9**. From there, users could visit the DBLP web site or use IRIS. Users were offered with

three different locations for displaying ancillary search results with IRIS: in a new page/tab, embedded into the Web page layout, shown as a floating window.

The study included an online survey that covers five main chapters as follows: i) presentation of the study, ii) occurrence of ancillary search in their daily life; iii) preference for formulating queries and interacting with results, iv) preference for the location of the search results in the display, and v) demographic data.

We have recruited 61 participants via mailing lists and social networks. Most of participants were male (77,1%) and, in average 26,1 years old (SD=5,7). Among the participants 44.3% were students, 39,3% researchers/professors and 16,4% work in the industry. We have got responses from Argentina, Austria, Brazil, France and Spain. They estimated to spend ~5,3 hours (SD=4,1) per week using search engines over the Web, most of which (~4,3 hours per week, SD=3,9) is spent looking for authors and publications. The amount of time participants perform searches related to authors and publications qualify them as typical users of tools with predefined queries for looking for information, such as searching for co-authors.

The results confirm that typing text to formulate a query is less appreciated than selecting keywords. Most participants found that selecting a term in a web page for launching a query is useful (85,3%), and that it improves performance (80,3%). Only 18,1% of participants prefers typing a text to formulate queries. Considering the alternatives for the location of display, 59% of participants said to prefer the option embedding results into the current page, while 36,1% liked more the design option showing the results in a floating window. Most participants prefer to see ancillary results in the same page (95,1%) rather than in a new tab/window (4,9%).

Most of the participants clearly pointed out that the options for showing results embedded into the Web document (either the floating window or the option changing the layout) presented the advantage of reducing the interruption created by search engines when showing the results in a new window/tab. The frequency on which such disruption was reported in the comments lets us think that participants really notice the articulatory distance created when new windows are open. Overall, users did not like the option that shows a new page because of the change of context. This result is compatible with our claims for the inside-in approach. The majority of positive comments were centered on the availability of the additional information right next to the search keyword.

5   Related Work and Discussion

Wilson [24] claims that the design of the user interface has an important cognitive impact on tasks performance; thus, search engines should evolve to take into account users' needs. Although these claims are valid, most of the research efforts in the area have been focused on two main subjects: (i) algorithms for improving the accuracy of search engines with respect to many users' concerns and (ii) approaches for improving the visualization of Web pages [21]. For example, Schwarz and Morris [18] describe an information visualization approach for augmenting Web pages with data about the credibility of the ranking proposed by the search engine. While such approach does not help users to formulate better queries, it might help users to better select the pages to

visit according to the rank of pages proposed by the search engines. Capra et al. [1] also proposed to augment the user interface of search results by adding images next to the ranking provided by search engines aiming at helping users to make better decisions. These few examples are illustrative of strategies for improving the design and display of the ranking of results from search engines.

In the last decades, several information visualization approaches have been developed for presenting search results coming either from search engines or widely used databases, such as DBLP, ACM DL, IEEE Xplore, etc. Some search engines with built-in visualization tools have also been developed. The first reports presenting and/or discussing visualization of search results date from the late 90's and early 2000's. However, although along the years, many different techniques have been evaluated [19] with results favoring visualizations, the majority of web search engines still provide textual lists ordered by some user or tool specified criteria.

As far as we know, the *inside-in* approach proposed in this paper is an original contribution that can improve users' performance while performing ancillary searches. In some extension, the principles of the inside-in approach can be related to the *focus+context* approach proposed by Card, Mackinlay & Shneiderman [2]. Moreover, it is fully compatible with the Shneiderman's visual information-seeking mantra "*Overview first, zoom and filter, then details-on-demand*" [16].

Although our tools are fully operational, they should be considered a simple proof of concept. Other implementations are possible, indeed, and we suggest that it might require a few extensions of Web browsers to support a native implementation of the inside-in approach. The preliminary results obtained with the survey confirmed our first hypothesis: most users prefer to launch queries directly from the web page by selecting a keyword. This is not a new finding [10] but indicates that we are in the right path. As for the other three hypotheses, they were confirmed: users also prefer search results being displayed through an interactive visualization technique, located near the search keyword. Regarding location, users expressed to prefer the display of results in a way that does not change their context, this being achieved by two alternatives – displaying the results embedded in the web page, by augmenting it, or displaying them in a floating layer over the same web page.

One of the interesting contributions of the inside-in approach is to allow users to explore the results and perform nested queries that are meant as ancillary-search tasks. The preliminary results of our tools confirm that the semantic and articulatory distances of inputs (*execution gulf*) in the search task are reduced because searching is launched by clicking on a keyword displayed in the Web page. The semantic and articulatory distances of output (*evaluation gulf*) are also reduced when ancillary search results are placed in the same page.

6   Conclusions and Future Work

This paper proposed a new perspective for looking at the way search user interfaces can be conceived for helping users to perform ancillary-search tasks on the Web. For that we have proposed the inside-in approach which aims at reducing both execution and evaluation gulfs in the user interaction with search engines. Indeed, one of the key aspects of this approach is to provide a better integration of search results into existing Web pages, where users require complementary information to make their decisions.

Overall the inside-in approach is generic and can be implemented using current search engines such as Google or Yahoo! Nonetheless, it can also be implemented using search engines that are suitable to provide more focused and accurate results about data in a specific application domain. Our framework follows this latter approach as illustrated with the implementation of queries for searching co-authors in the DBLP. While looking up for co-authors might be perceived as a very narrow and specific search, it is noteworthy that it is relevant and frequent in the domains of scientific research, and also is a concern to a large population of researchers, students, teachers, and experts from research funding agencies. Moreover, such specialized characteristic can be tuned and adapted according to specific users' needs. Indeed, the main challenge here remains the identification of relevant queries that are suitable to help users to accomplish their tasks.

Despite the promising results, we know that these are preliminary and there is much work to be done. We would like to measure the distances in the gulfs by performing experiments with direct observation methods. We also intend to proceed with the development of different input and output techniques for performing search tasks since our framework was developed aiming at such studies. Future work also include empirical testing with users in a usability laboratory. This step would allow us to assess user performance when performing the tasks and collect more qualitative data via thinking aloud, which would better explain the user experience factors that influence the use of information visualization techniques for displaying search results. We also plan to develop new techniques for embedding ancillary results into Web pages and then investigate their effect in terms of usability and UX.

7 Acknowledgments

8 References

1. Capra, R., Arguello, J., Scholer, F. 2013. Augmenting web search surrogates with images. In Proceedings of the 22nd ACM international conference on Conference on information & knowledge management (CIKM '13). ACM, New York, NY, USA, 399-408.
2. Card, S. K., Mackinlay, J. D., Shneiderman, B.. Focus + Context. In: Card, Stuart K.; Mackinlay, J. D.; Shneiderman, Ben. Readings in Information Visualization: Using Vision to Think. San Francisco, California, USA. Morgan Kaufmann Publishers, 1999. p.307-309.
3. Dorneles, C. F., Gonçalves, R., dos Santos Mello, R. Approximate data instance matching: a survey. Knowledge Information Systems, 27(1): 1-21 (2011).
4. Firmenich, S., Winckler, M., Rossi, G. A Framework for Concern-Sensitive, Client-Side Adaptation. In Proc. of International Conference on Web Engineering (ICWE 2011), Paphos, Cyprus, June 20-24, 2011. Springer, LNCS 6757, pages 198-213.
5. Fleming, J. Web Navigation: Designing the User Experience. O'Reilly, 1998, 264 p.
6. Hassan, A., White, R. W., Dumais, S.T., Wang, Y-M. Struggling or exploring?: disambiguating long search sessions. WSDM 2014: 53-62.

7. Hearst, M. User Interfaces for Search, Chapter 2 of Modern Information Retrieval: The Concepts and Technology behind Search (2nd Edition), Addison Wesley, 2011.

8. Hilbert, M., López, P. The World's Technological Capacity to Store, Communicate, and Compute Information. Science, Vol. 332, No. 6025. (1 April 2011), pp. 60-65.

9. Marchionini, G. Interfaces for end-user information seeking. Journal of the American Society for Information Science, 43(2):156-163, January 1999.

10. Marchionini, G. Exploratory Search: From Finding To Understanding. Comm. of the ACM, 49(4):41–49, 2006.

11. Martinie, C., Palanque, P., Winckler, M. Structuring and Composition Mechanism to Address Scalability Issues in Task Models. IFIP TC13 Conference on Human-Computer Interaction (INTERACT 2011), September 05-09 2011, Lisbon, Portugal, Springer LNCS 6949, pages 593-611.

12. Martinie, C. (2011) A Synergistic Models-Based Approach to Develop Usable, Reliable and Operable Interactive Critical Systems. PhD thesis presented on December 25th 2011. Université Paul Sabatier. Available at: http://www.irit.fr/~Marco.Winckler/martinie-these2011.pdf

13. McCrickard, S. and Kehoe, C. Visualizing search results using SQUID. Poster at 6th WWWW Conference, April 1997. At: http://people.cs.vt.edu/~mccricks/papers/mk97.pdf

14. Norman, D. The Psychology Of Everyday Things. (1988). Basic Books; 1 edition (June 13, 1988). Basic Books. ISBN 978-0-465-06710-7

15. Norman, D. A., Draper, S. W. (eds.) (1986): User Centered System Design: New Perspectives on Human-Computer Interaction. Hillsdale, NJ, Lawrence Erlbaum Associates.

16. Paternò, F., Mancini, C. and Meniconi, S. ConcurTaskTrees: A Diagrammatic Notation for Specifying Task Models. In: Proc. of Interact'97. Chapman & Hall (1997), 362-369.

17. Shneiderman, B. 1996. The Eyes Have It: A Task by Data Type Taxonomy for Information Visualizations. In *Proceedings of the 1996 IEEE Symposium on Visual Languages* (VL '96). IEEE Computer Society, Washington, DC, USA, p. 336-343.

18. Schwarz, J., Morris, M. R. Augmenting web pages and search results to support credibility assessment. CHI 2011: 1245-1254.

19. Sebrechts, M., Vasilakis, J., Miller, M., Cugini, J., Laskowski, S. (1999). Visualization of Search Results: A Comparative Evaluation of Text, 2D and 3D Interfaces. Proc. of the 22nd ACM SIGIR Conference on Research and Development in Information Retrieval. pp 3-10.

20. Sutcliffe, A.G., Ennis, M. Towards a cognitive theory of information retrieval. Interacting with Computers, 10:321–351, 1998.

21. Suzuki, E., Ando, S., Hirose, M., Jumi, M. Intuitive Display for Search Engines Toward Fast Detection of Peculiar WWW Pages. In. WImBI 2006. N. Zhong et al. (Eds.): WImBI 2006, LNAI 4845, pp. 341–352.

22. ter Beek, M. H., Faconti, G. P., Massink, M., Palanque, P., Winckler, M. Resilience of Interaction Techniques to Interrupts: A Formal Model-based Approach In Proceedings of the 12th IFIP TC 13 Conference on Human-Computer Interaction (INTERACT 2009). LNCS 5726, pp. 494–509, Uppsala, Sweden, 24-28 August 2009.

23. Yates, R.B., Neto, B. R., Modern Information Retrieval: : The Concepts and Technology behind Search (2nd Edition). New York: ACM Press Books / Addison-Wesley Professional, 2 edition (February 10, 2011). 944 pages.

24. Wilson, Max L. Evaluating the Cognitive Impact of Search User Interface Design Decisions. EuroHCIR 2011: 27-30.

25. Winckler, M., Gaits, V., Vo, D-B., Firmenich, S., Rossi, G. An Approach and Tool Support for Assisting Users to Fill-in Web Forms with Personal Information. In Proc. of the ACM SIGDOC 2011, Pisa, Italy, 2011, ACM, New York, NY, USA, pp. 195-202.