



US007272601B1

(12) **United States Patent**
Wang et al.

(10) **Patent No.:** **US 7,272,601 B1**
(45) **Date of Patent:** **Sep. 18, 2007**

(54) **SYSTEMS AND METHODS FOR ASSOCIATING A KEYWORD WITH A USER INTERFACE AREA**

(75) Inventors: **Niniane Wang**, Santa Clara, CA (US);
Stephen R. Lawrence, Mountain View, CA (US)

(73) Assignee: **Google Inc.**, Mountain View, CA (US)

(*) Notice: Subject to any disclaimer, the term of this patent is extended or adjusted under 35 U.S.C. 154(b) by 488 days.

(21) Appl. No.: **10/814,053**

(22) Filed: **Mar. 31, 2004**

(51) **Int. Cl.**
G06F 7/00 (2006.01)
G06F 17/00 (2006.01)

(52) **U.S. Cl.** **707/4; 707/1; 707/2; 707/3; 707/5**

(58) **Field of Classification Search** **707/4, 707/3, 1, 2, 5**

See application file for complete search history.

(56) **References Cited**

U.S. PATENT DOCUMENTS

6,006,222	A	12/1999	Culliss
6,014,665	A	1/2000	Culliss
6,078,916	A	6/2000	Culliss
6,182,068	B1	1/2001	Culliss
6,581,056	B1	6/2003	Rao
2003/0167266	A1	9/2003	Saldanha et al.
2004/0143569	A1	7/2004	Gross et al.

OTHER PUBLICATIONS

Czerwinski et al., "Visualizing Implicit Queries For Information Management and Retrieval", CHI '99, May 15-20, 1999, ACM, pp. 560-567.*

U.S. Appl. No. 10/749,440, filed Dec. 31, 2003, Badros et al. 80-20 Software—Products—80-20 One Search, <http://www.80-20.com/products/one-search/retriever.asp>, printed Mar. 16, 2004.

"askSam™ Making Information Useful," askSam.—Organize your information with askSam, <http://www.asksam.com/brochure.asp>, printed Mar. 15, 2004.

Alexa® Web Search—Toolbar Quick Tour, http://pages.alexa.com/prod_serv/quicktour.html, pp. 1-5, printed Mar. 16, 2004.

Barrett, R. et al., "How to Personalize the Web," IBM Research, <http://www.almaden.ibm.com/cs/wbi/papers/chi97/wblpaper.html>, pp. 1-13, printed Mar. 16, 2004.

Battelle, J., CNN.com "When geeks go camping, ideas hatch," <http://www.cnn.com/2004/TECH/ptech/01/09/bus2.feet.geek.camp/index.html>, pp. 1-3, printed Jan. 13, 2004.

Boyan, J., et al., "A Machine Learning Architecture for Optimizing Web Search Engines," School of Computer Science, Carnegie Mellon University, May 10, 1996, pp. 1-8.

Bradenbaugh, F., "Chapter 1 The Client-Side Search Engine," *JavaScript Cookbook*, 1st Ed., Oct. 1999, O'REILLY™ Online Catalog, <http://www.oreilly.com/catalog/iscook/chapter/ch01.html>, pp. 1-30, printed Dec. 29, 2003.

Brin, S., et al., "The Anatomy of a Large-Scale Hypertextual Web Search Engine," <http://www7.scu.edu.au/programme/fullpapers/1921/com1921.htm>, pp. 1-18, 1998.

Budzik, J., et al., User Interactions with Everyday Applications as Context for Just-in-time Information Access, Intelligent Information Laboratory, Northwestern University, pp. 1-8, no date.

DEVONthink, <http://www.devon-technologies.com/products/devonthink.php>, printed Mar. 16, 2004.

dtSearch® —<http://www.dtsearch.com/>, printed Mar. 15, 2004.

Dumais, S., et al., "Stuff I've Seen: A System for Personal Information Retrieval and Re-Use," Microsoft Research, *SIGIR '03*, Jul. 28-Aug. 1, 2003, pp. 1-8.

(Continued)

Primary Examiner—Jeffrey Gaffin

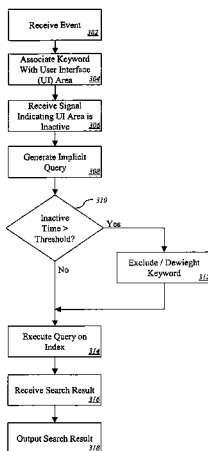
Assistant Examiner—Michael Hicks

(74) *Attorney, Agent, or Firm*—Fenwick & West LLP

(57) **ABSTRACT**

Systems and methods for associating a keyword with a window are described. In one described system, a computer program, such as an indexer, captures processor, or query system, associates a keyword with a first user interface area. The query system or other computer program receives a signal that the first user interface is inactive and that a second user interface area is active and, in response, generates an implicit search query that includes the keyword associated with the first user interface area.

19 Claims, 4 Drawing Sheets



OTHER PUBLICATIONS

- Enfish, <http://www.enfish.com>, printed Mar. 16, 2004.
- Fast Search & Transfer—Home—Enterprise Search, http://solutions.altavista.com/en/news/pr_020402_desktop.shtm, printed Mar. 16, 2004.
- Fertig, S., et al., “Lifestreams: An Alternative to the Desktop Metaphor,” <http://www.acm.org/sigchi/chi96/proceedings/videos/Fertig/etf.htm>, pp. 1-3, printed Mar. 16, 2004.
- Geisler, G., “Enriched Links: A Framework for Improving Web Navigation Using Pop-Up Views,” pp. 1-14, 2000.
- ISYS Search Software—ISYS: desktop, <http://www.isysusa.com/products/desktop/index.html>, printed Mar. 16, 2004.
- Joachims, T., et al., “WebWatcher: A Tour Guide for the World Wide Web,” 1996.
- Markoff, J., “Google Moves toward Clash with Microsoft,” *The New York Times*, May 19, 2004, <http://www.nytimes.com/2004/5/19/technology/19google.html?ex=1085964389&ei=1&e...>, pp. 1-4, printed May 19, 2004.
- Naraine, R., “Future of Search Will Make You Dizzy,” *Enterprise*, May 20, 2004, <http://www.internetnews.com/ent-news/article.php/3356831>, pp. 1-4, printed May 21, 2004.
- “Overview,” Stuff I’ve Seen—Home Page, <http://research.microsoft.com/adapt/sis/index.htm>, pp. 1-2, printed May 26, 2004.
- Rhodes, B., “Margin Notes Building a Contextually Aware Associative Memory,” *The Proceedings of the International Conference on Intelligent User Interfaces (IUI’00)*, Jan. 9-12, 2000.
- Rhodes, B., et al., “Just-in-time information retrieval agents,” *Systems Journal*, vol. 39, Nos. 3&4, 2000, pp. 685-704.
- Rhodes, B., et al., “Remembrance Agent—A continuously running automated information retrieval system,” *The Proceedings of the First International Conference on the Practical Application of Intelligent Agents and Multi Agent Technology (PAAM ’96)*, pp. 487-495.
- Rizzo, T., “WinFS 101: Introducing the New Windows File System,” Longhorn Developer Center Home: Headline Archive: WinFS 101: Introducing the New . . . , <http://msdn.microsoft.com/Longhorn/archive/default.aspx?pull+/library/en-us/dnwinfs/htm...>, pp. 1-5, printed Apr. 21, 2004.
- “Searching for the next Google—New trends are helping nimble startups elbow in to the plundered market,” *Red Herring—The Business of Technology*, Mar. 9, 2004, <http://redherring.com/PrintArticle.aspx?a=4782§or=Capital>, p. 1-5, printed Mar. 30, 2004.
- “Selecting Task-Relevant Sources for Just-In-Time Retrieval,” pp. 1-3, no date.
- Sherman, C., “HotBot’s New Desktop Search Toolbar,” www.searchenginewatch.com, http://searchenginewatch.com/searchday/print.php/34711_339921, pp. 1-3, printed Apr. 14, 2004.
- “Standardization Priorities for the Directory—Directory Interoperability Forum White Paper,” The Open Group, Dec. 2001, pp. 1-21.
- Sullivan, D., “Alta Vista Releases Search Software,” *The Search Engine Report*, Aug. 4, 1998, pp. 1-2.
- WebWatcher Home Page, “Welcome to the WebWatcher Project,” <http://www-2.cs.cmu.edu/~webwatcher/>, printed Oct. 15, 2003.
- “WhenU Just-In-Time Marketing,” <http://www.whenu.com>, printed Mar. 19, 2004.
- X1 instantly searches files & email. For outlook, Outlook, <http://www.x1.com/>, printed Mar. 15, 2004.
- Zellweger, P., et al., “Fluid Links for Informed and Incremental Link Transitions,” *Proceedings of Hypertext’98*, Pittsburgh, PA, Jun. 20-24, 1998, pp. 50-57.

* cited by examiner

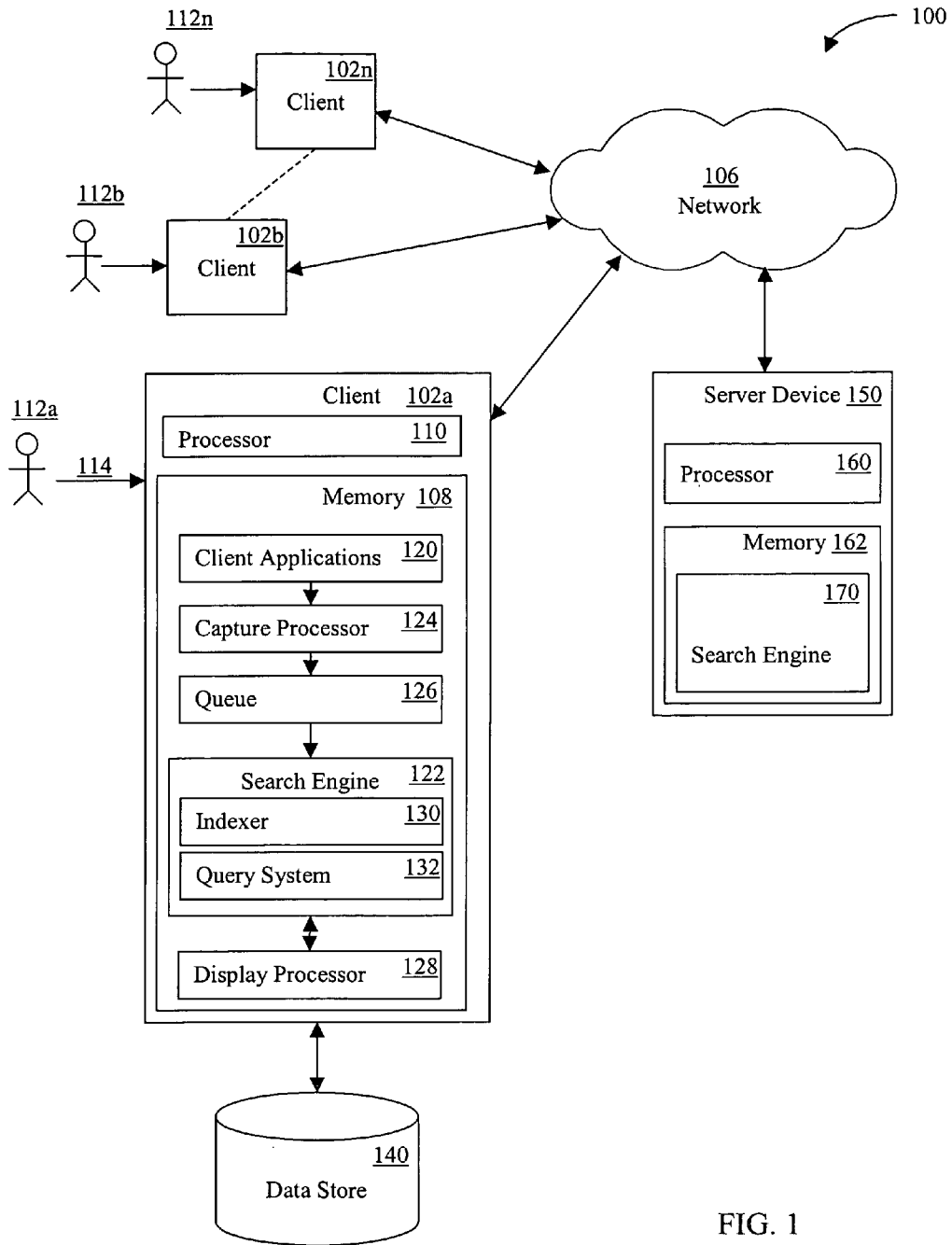


FIG. 1

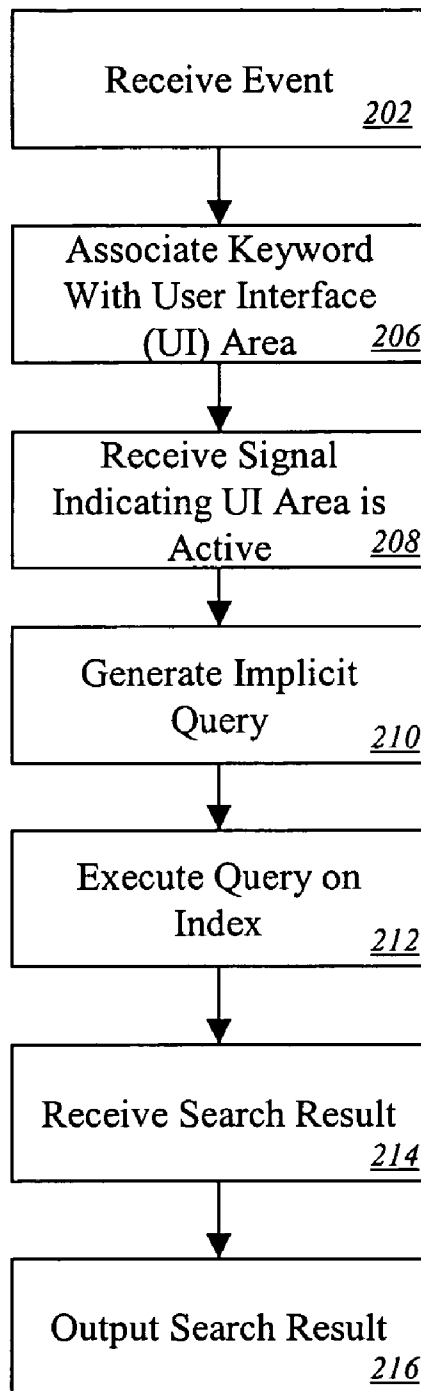


FIG. 2

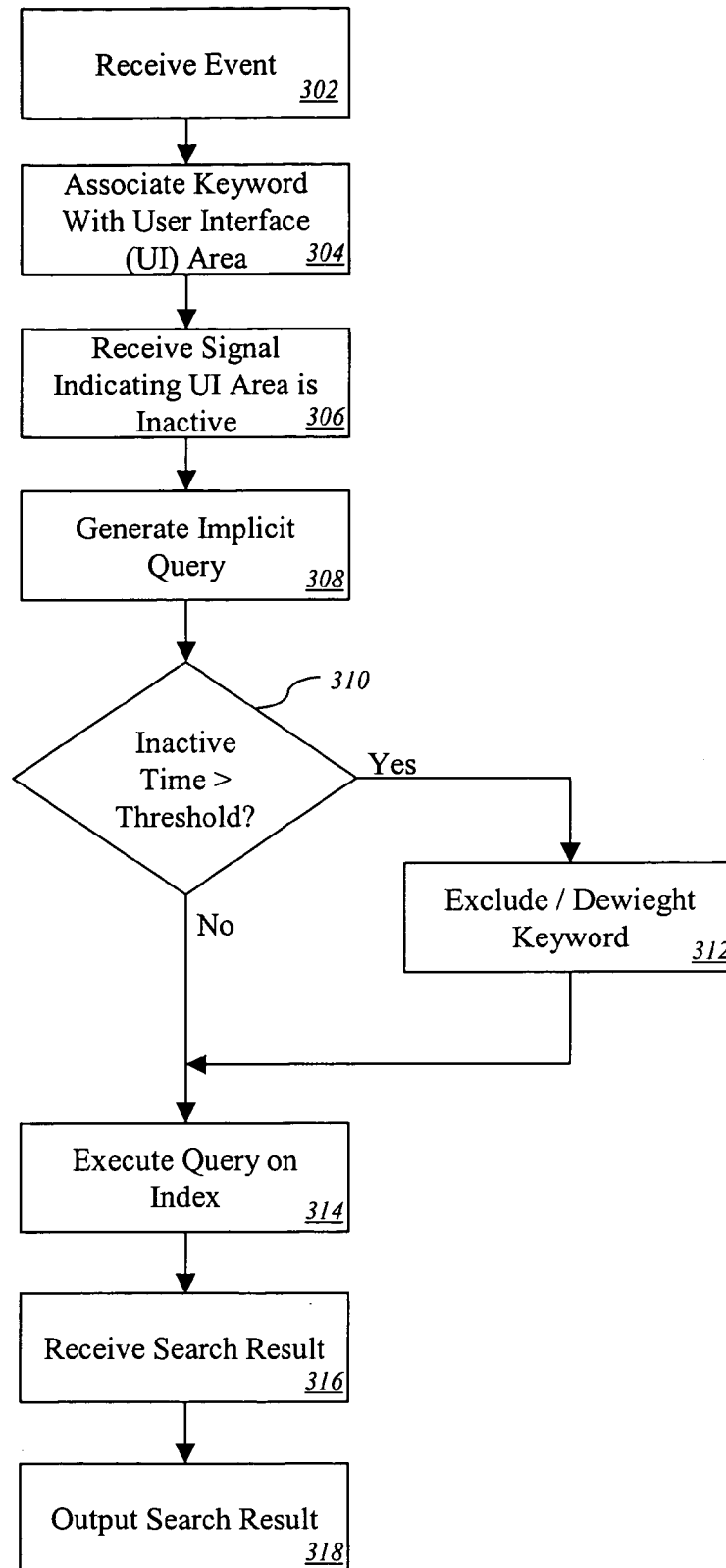


FIG. 3

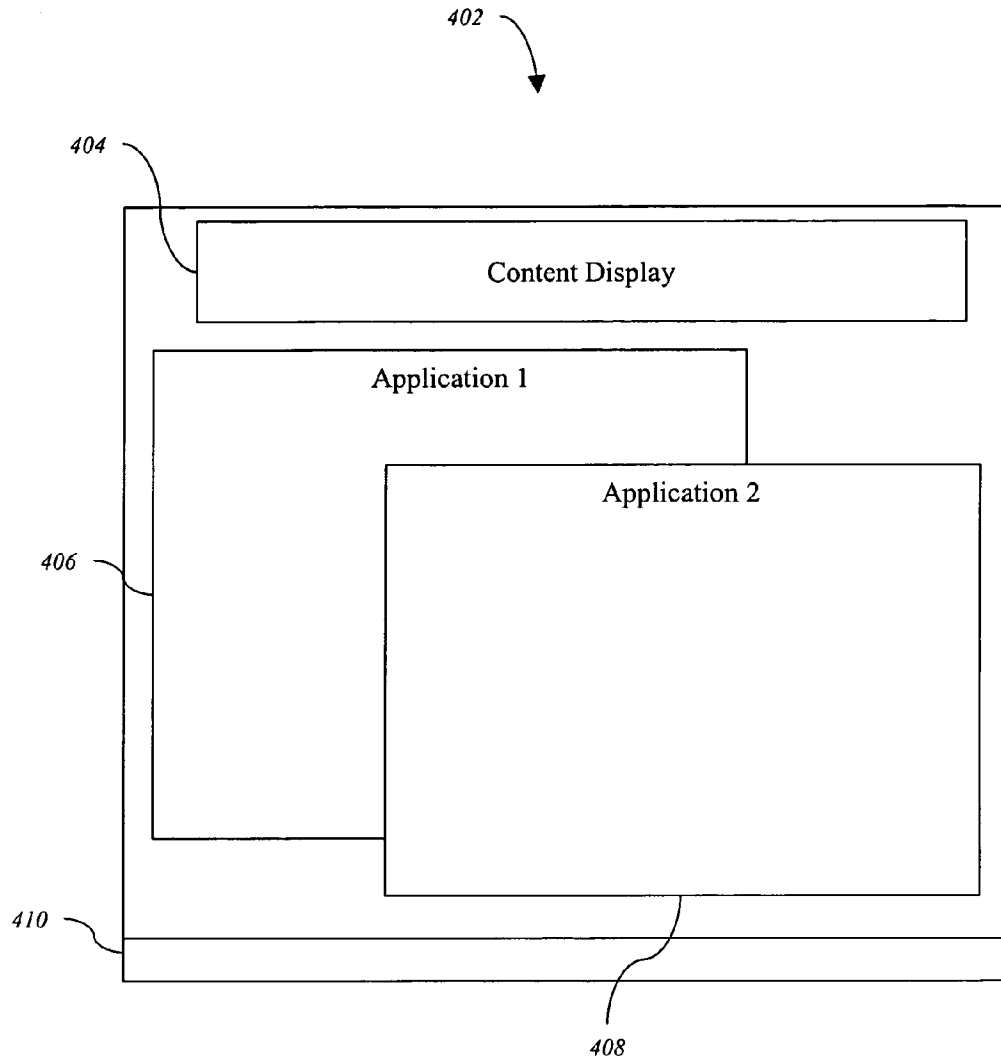


FIG. 4

SYSTEMS AND METHODS FOR ASSOCIATING A KEYWORD WITH A USER INTERFACE AREA

RELATED APPLICATIONS

This application relates to:

application Ser. No. 10/814,908, filed herewith, titled “Systems and Methods for Generating Multiple Implicit Search Queries”;

application Ser. No. 10/814,871, filed herewith, titled “Systems and Methods for Extracting a Keyword from an Event”;

application Ser. No. 10/814,074, filed herewith, titled “Systems and Methods for Weighting a Search Query Result”;

application Ser. No. 10/814,056, titled “Systems and Methods for Refreshing a Content Display”;

application Ser. No. 10/814,368, filed herewith, titled “Systems and Methods for Constructing and Using a User Profile”;

application Ser. No. 10/814,365, filed herewith, titled “Systems and Methods for Identifying a Named Entity”;

application Ser. No. 10/814,486, filed herewith, titled “Systems and Methods for Analyzing Boilerplate”;

application Ser. No. 10/814,875, filed herewith, titled “Systems and Methods for Ranking Implicit Search Results”;

application Ser. No. 10/814,908, filed herewith, titled “Systems and Methods for Generating a User Interface”; and

application Ser. No. 10/814,872, filed herewith, titled “Systems and Methods for Providing Search Results,”

the entirety of all of which are incorporated herein by reference.

FIELD OF THE INVENTION

The present invention relates generally to methods and systems for information retrieval. The present invention relates particularly to systems and methods for associating a keyword with a user interface area.

BACKGROUND

Conventional search engines receive a search query from a user and execute a search against a global index. Such conventional search engines typically use one or more conventional methods for performing a search. For example, one known method, described in an article entitled “The Anatomy of a Large-Scale Hypertextual Search Engine,” by Sergey Brin and Lawrence Page, assigns a degree of importance to a document, such as a web page, based on the link structure of the web. The search results are often presented in a list format, including article identifiers and brief snippets about the documents in a web page that can be resized.

A user may also have access to other information stored on the user’s local machine or on other storage media accessible via a network that is relevant to a user. Typically, a user enters an explicit search that includes keywords and that is executed against a global or local index (As used herein, a “keyword” or “keywords” is defined broadly to mean words, sequences of words, acronyms or other characters, including spaces, based upon which a search may be performed).

SUMMARY

Embodiments of the present invention provide systems and methods for associating a keyword with a user interface area. In one embodiment of the present invention, a search system, which may be implemented in hardware, software or a combination thereof, associates a keyword with a first user interface area. The system receives a signal that the first user interface area is inactive and that a second user interface area is active. In response, the system generates an implicit search query that includes the keyword. In one embodiment, a computer-readable medium (such as, for example random access memory or a computer disk) comprises code from carrying out such a method.

These exemplary embodiments are mentioned not to limit or define the invention, but to provide examples of embodiments of the invention to aid understanding thereof. Exemplary embodiments are discussed in the Detailed Description, and further description of the invention is provided there. Advantages offered by the various embodiments of the present invention may be further understood by examining this specification.

BRIEF DESCRIPTION OF THE FIGURES

These and other features, aspects, and advantages of the present invention are better understood when the following Detailed Description is read with reference to the accompanying drawings, wherein:

FIG. 1 is a block diagram illustrating an exemplary environment in which one embodiment of the present invention may operate;

FIG. 2 is a flowchart illustrating a method for associating a keyword with a user interface area in one embodiment of the present invention;

FIG. 3 is a flowchart illustrating a method of reweighting a keyword based on an inactivity period in one embodiment of the present invention; and

FIG. 4 is a block diagram of a user display in one embodiment of the present invention.

DETAILED DESCRIPTION

Embodiments of the present invention provide systems and methods for associating a keyword with a user interface area. Exemplary embodiments are described below.

System Architecture

Referring now to the drawings in which like numerals indicate like elements throughout the several figures, FIG. 1 is a block diagram illustrating an exemplary environment for implementation of an embodiment of the present invention. While the environment shown reflects a client-side search engine architecture embodiment, other embodiments are possible.

The system **100** shown in FIG. 1 includes multiple client devices **102a-n** in communication with a server device **150** over a wired or wireless network **106**. The network **106** shown comprises the Internet. In other embodiments, other networks, such as an intranet, may be used instead. Moreover, methods according to the present invention may operate within a single client device.

The client devices **102a-n** shown each includes a computer-readable medium **108**. The embodiment shown includes a random access memory (RAM) **108** coupled to a processor **110**. The processor **110** executes computer-ex-

ecutable program instructions stored in memory **108**. Such processors may include a microprocessor, an ASIC, a state machine, or other processor, and can be any of a number of computer processors, such as processors from Intel Corporation of Santa Clara, Calif. and Motorola Corporation of Schaumburg, Ill. Such processors include, or may be in communication with, media, for example computer-readable media, which stores instructions that, when executed by the processor, cause the processor to perform the steps described herein.

Embodiments of computer-readable media include, but are not limited to, an electronic, optical, magnetic, or other storage or transmission device capable of providing a processor, such as the processor **110** of client **102a**, with computer-readable instructions. Other examples of suitable media include, but are not limited to, a floppy disk, CD-ROM, DVD, magnetic disk, memory chip, ROM, RAM, an ASIC, a configured processor, all optical media, all magnetic tape or other magnetic media, or any other medium from which a computer processor can read instructions. Also, various other forms of computer-readable media may transmit or carry instructions to a computer, including a router, private or public network, or other transmission device or channel, both wired and wireless. The instructions may comprise code from any suitable computer-programming language, including, for example, C, C++, C#, Visual Basic, Java, Python, Perl, and JavaScript.

Client devices **102a-n** can be connected to a network **106** as shown, or can be stand-alone machines. Client devices **102a-n** may also include a number of external or internal devices such as a mouse, a CD-ROM, DVD, a keyboard, a display, or other input or output devices. Examples of client devices **102a-n** are personal computers, digital assistants, personal digital assistants, cellular phones, mobile phones, smart phones, pagers, digital tablets, laptop computers, Internet appliances, and other processor-based devices. In general, the client devices **102a-n** may be any type of processor-based platform that operates on any operating system, such as Microsoft® Windows® or Linux, capable of supporting one or more client application programs. For example, the client device **102a** shown comprises a personal computer executing client application programs, also known as client applications **120**. The client applications **120** can be contained in memory **108** and can include, for example, a word processing application, a spreadsheet application, an email application, an instant messenger application, a presentation application, an Internet browser application, a calendar/organizer application, and any other application or computer program capable of being executed by a client device.

The user **112a** can interact with the various client applications **120** and articles associated with the client applications **120** via various input and output devices of the client device **102a**. Articles include, for example, word processor, spreadsheet, presentation, email, instant messenger, database, and other client application program content files or groups of files, web pages of various formats, such as HTML, XML, XHTML, Portable Document Format (PDF) files, and audio files, video files, or any other documents or groups of documents or information of any type whatsoever.

The memory **108** of the client device **102a** shown also contains a capture processor **124**, a queue **126**, and a search engine **122**. The client device **102a** shown also contains or is in communication with a data store **140**. The search engine **122** can receive an explicit query from the user **112a** or generate an implicit query and retrieve information from the data store **140** in response to the query.

The search engine **122** shown contains an indexer **130**, a query system **132**, and a formatter **134**. Events, real-time and historical, contextual and indexable, and performance data can be sent by the queue **126** to the query system **132** to provide the query system **132** with information concerning current user context. The query system **132** can use this information to generate an implicit query. The query system **132** can also receive and process explicit queries from the user **112a**.

The data store **140** can be any type of computer-readable media and can be integrated with the client device **102a**, such as a hard drive, or external to the client device **102a**, such as an external hard drive or on another data storage device accessed through the network **106**. The data store **140** may include any one or combination of methods for storing data, including without limitation, arrays, hash tables, lists, and pairs.

The data store **140** comprises a local index. The local index in the embodiment shown in FIG. **1** may comprise information, such as articles, which are associated with the client device **102a**, a user **112a** of the client device **102a**, or a group of users of the client device **102a**. For example, the local index in the data store **140** shown in FIG. **1** may comprise an index of articles created, edited, received, or stored by the client user **112a** using the client machine **102a**, or articles otherwise associated with the client user **102a** or the client machine **112a**. The local index may be stored in a client machine, such as in data store **140**, in a data store on a local network in a manner accessible by the client machine, on a server accessible to the client machine through the Internet, or in another accessible location.

In contrast, a global index may comprise information relevant to many users or many servers, such as, for example, an index of web pages located on multiple servers in communication with the World Wide Web. One example of a global index is an index used by the Google™ search engine to provide search results in response to a search query.

A single index may comprise both a local and a global index. For example, in one embodiment, an index may comprise both local and global information, and include a user or client identifier with the local information so that it may be identified with the user(s) or client(s) to which it pertains. Moreover, an index, local or global, may be present in one or multiple logical or physical locations.

In the embodiment shown in FIG. **1**, a user **112a** can input an explicit query into a search engine interface displayed on the client device **102a**, which is received by the search engine **122**. The search engine **122** can also generate an implicit query based on a current user context or state, which can be determined by the query system **132** from contextual real time events or other means. Based on the query, the query system **132** can locate relevant information in the data store **140** or other index and provide a result set. In one embodiment, the result set comprises article identifiers identifying articles associated with the client applications **120** or client articles. Client articles stored in the data store **140** include articles associated with the user **112a** or client device **102a**, such as the word processing documents, previously viewed web pages and any other article associated with the client device **102a** or user **112a**. In another embodiment, the result set also comprises identifiers identifying articles located on the network **106** or network articles located by a search engine on a server device. Network articles include articles located on the network **106** not previously viewed or otherwise referenced by the user **112a**, such as web pages not previously viewed by the user **112a**.

5

The result sets comprise one or more article identifiers. An article identifier may be, for example, a Uniform Resource Locator (URL), a file name, a link, an icon, a path for a local file, or anything else that identifies an article. In the embodiment shown, an article identifier comprises a URL associated with an article.

Messaging articles stored in the data store **140** include user's emails, chat messages, and instant messaging messages. Each time a message is received, sent, modified, printed, or otherwise accessed, a record is stored in the data store **140**. This information can later be searched to identify messages that should be displayed in the user interface.

An embodiment of the present invention may also store message threads in the data store **140**. In such an embodiment, messages are related together by various attributes, including, for example, the sender, recipient, date/time sent and received, the subject, the content, or any other attribute of the message. The related messages can then be retrieved as a thread, which may be treated as a document by the display processor **128**.

The formatter **134** can receive the search result set from the query system **132** of the search engine **122** and can format the results for output to a display processor **128**. In one embodiment, the formatter **134** formats the results in XML or HTML. The display processor **128** can be contained in memory **108** and can control the display of the result set on a display device associated with the client device **102a**. The display processor **128** may comprise various components. For example, in one embodiment, the display processor **128** comprises a Hypertext Transfer Protocol (HTTP) server that receives requests for information and responds by constructing and transmitting Hypertext Markup Language (HTML) pages. In one such embodiment, the HTTP server comprises a scaled-down version of the Apache Web server. In various embodiments, the functions described herein may be performed by various other components and devices.

Through the client devices **102a-n**, users **112a-n** can communicate over the network **106**, with each other and with other systems and devices coupled to the network **106**. As shown in FIG. 1, a server device **150** is also coupled to the network **106**. In the embodiment shown, the search engine **122** can transmit a search query comprised of an explicit or implicit query or both to the server device **150**. The user **112a** can also enter a search query in a search engine interface, which can be transmitted to the server device **150**. In another embodiment, the query signal may instead be sent to a proxy server (not shown), which then transmits the query signal to server device **150**. Other configurations are also possible.

The server device **150** shown includes a server executing a search engine application program, such as the Google™ search engine. Similar to the client devices **102a-n**, the server device **150** shown includes a processor **160** coupled to a computer-readable memory **162**. Server device **150**, depicted as a single computer system, may be implemented as a network of computer processors. Examples of a server device **150** are servers, mainframe computers, networked computers, a processor-based device, and similar types of systems and devices. The server processor **160** can be any of a number of or combination of computer processors, such as processors from Intel Corporation of Santa Clara, Calif. and Motorola Corporation of Schaumburg, Ill.

Memory **162** contains the search engine application program, also known as a search engine **170**. The search engine **170** locates relevant information in response to a search query from a client device **102a**. The search engine **122** then provides the result set to the client device **102a** via the

6

network **106**. The result set **134** comprises one or more article identifiers. An article identifier may be, for example, a URL, a file name, a link, an icon, a path for a local file, or anything else that identifies an article. In the embodiment shown, an article identifier comprises a URL associated with an article. The result set may include text, audio, video or any other type of content.

In the embodiment shown, the server device **150**, or related device, has previously performed a crawl of the network **106** to locate articles, such as web pages, stored at other devices or systems connected to the network **106**, and indexed the articles in memory **162** or on another data storage device. In other embodiments, a crawl is not performed. For example, in one embodiment, an index of articles is created manually.

It should be noted that embodiments of the present invention may comprise systems having different architecture than that which is shown in FIG. 1. For example, in some systems according to the present invention, server device **104** may comprise a single physical or logical server. The system **100** shown in FIG. 1 is merely exemplary, and is used to explain the exemplary methods shown in FIGS. 2 and 3.

Process

Various methods may be implemented in the environment shown in FIG. 1 and other environments, according to the present invention. Methods according to the present invention may be implemented by, for example, a processor-executable program code stored on a computer-readable medium.

In one embodiment of the present invention, a system, such as indexer **130**, captures processor **124**, or query system **132**, associates a keyword with a first user interface area. The query system **132** receives a signal that the first user interface area is inactive and that a second user interface area is active. For example, in a Microsoft® Windows operating environment, the query system **132** may intercept an application programming interface (API) call directed to the operating system that instructs the operating system to maximize a window or other user interface area in which an application is executing. The query system **132** interprets this call as activating the window that is the subject of the call and as inactivating all of the other windows in the user interface. The query system **132** may respond to receiving the call by executing a separate API call to determine all of the windows that are currently executing in the operating system and are inactive (i.e., all of the windows other than the active window).

In response to receiving the signal that the first user interface area is inactive and the second is active, the query system **132** generates an implicit search query that includes the keyword. The keyword may be a single keyword or a plurality of keywords.

In one embodiment, a computer program also identifies the keyword to be associated with the user interface area. For example, the user interface area may include a document, such as a word-processing document. In one embodiment, a computer program is able to retrieve or receive a keyword associated with the document.

The query system **132** receives a second signal indicating that a second user interface area is active and generates an implicit query that includes the keyword from the first user interface area. In such an embodiment, use of the keyword in the implicit query may be discontinued after a period of time has elapsed, e.g., ten seconds in an exemplary embodi-

ment of the present invention. In another such embodiment, the keyword or results associated with the keyword are downweighted after a period of time has elapsed. For example, after ten seconds the results associated with the keyword are downweighted by fifty percent; after 20 seconds, the results are downweighted by seventy-five percent; and after thirty seconds, use of the keyword is discontinued. In another embodiment, the amount of downweighting is a function of the time since the keyword was extracted and/or the corresponding user interface area was active.

The association between the keyword and the user interface area may persist. For example, the keyword and a user interface area identifier may be stored in memory. In one embodiment, the keyword is an attribute of an event as described in relation to FIG. 1. The event may include other attributes, such as an identifier of the user interface area.

In one embodiment of the present invention, the query system 132 submits the search query to a local or global index. In response, the query system 132 receives a result set and causes the result set to be output.

In one embodiment, the query system 132 weights the keywords for the active user interface area (e.g., window) more heavily than keywords for inactive user interface areas. The weight may decrease proportionally to the time since the particular user interface area with which a keyword is associated was last active. If the user switches back to an inactive user interface area, then the keywords for that user interface area become weighted more heavily again. This is facilitated by keeping track of keywords for each active and inactive user interface area. In such an embodiment, sets of one or more keywords (e.g., from events), each of which has a corresponding user interface area identifier and a corresponding time. The weight at which the keywords are used varies depending on the active user interface area, the time since the keywords were created, and the time since the user interface area was last active if it is not currently active. Older keywords have lower weight, and keywords associated with an inactive application have lower weight. If the user interface area becomes active again the weight increases. In another embodiment, the time since an application was last inactive is used in the weighting scheme, with lower weight going to keywords associated with user interface areas that have been inactive for a longer period.

In another embodiment, the system tracks the frequency and total amount of time that a user interface area is active. For example, if a user continually switches back and forth between an IM user interface area and several other applications, the system may recognize the repeated accesses to the IM user interface area and adjust the weight for keywords associated with that user interface area accordingly.

In one embodiment, adjustment of the weighting is performed using a step function that changes at certain predefined thresholds. In another embodiment, the adjustment is performed using a smooth function based on the age of the keyword.

FIG. 2 is a flowchart illustrating a method for associating a keyword with a user interface area in one embodiment of the present invention. In the embodiment shown, a computer program, such as the query system 132, receives an event 202. The event signifies that some activity has occurred. For example, an event may signify that the user has received or sent an email, accessed a document, submitted an explicit query to a search engine, or performed some other activity. The event shown in FIG. 2 signifies that some activity has occurred that is associated with a user interface area. For example, the user may have maximized, minimized, or restored a window, entered text into a document, printed a

document, selected text from a web page, or performed some other activity associated with the user interface area.

The event includes attributes. For example, in the embodiment shown, the event includes an identifier of the user interface area containing the application that caused the event to be generated. For example, the user interface area may include a word-processing application. When the user completes a section of text, which is signaled when the user enters a punctuation mark, for example, an event is generated. The event includes an identifier of the word-processing application. The event may also include one or more keywords. The query system 132 associates the keyword(s) with the user interface area 204. The association may be stored in memory or otherwise persist. For instance, in one embodiment, a word processing application executes within a user interface area. When the user completes the typing of a sentence by, for example, entering a period, an event is generated. The event may comprise an identifier of the user interface area as well as a keyword or keywords from the sentence that the user just completed. The query system 132 saves the association between the keyword and the user interface area.

Subsequently, the query system 132 receives a signal indicating that the user interface area with which the keyword or keywords is associated is active 206. For instance, the user may receive an email message and click on the email program to access the message. The query system 132 receives a signal indicating the user interface area associated with the email program is now active. Subsequently, the user clicks on the user interface area in which the word processing application is executing. The query system 132 receives a signal indicating that the user interface area associated with the email program is now inactive, and that the user interface area associated with the word processing program is now active.

After receiving the signal or in response to receiving the signal, the query system 132 generates an implicit query 210. Since the user interface area is active, the embodiment shown in FIG. 2 includes the keywords associated with the user interface area in the implicit query. If the user interface area is not currently active, the keywords may or may not be included within the implicit query. In one embodiment, when keywords associated with non-active user interface areas are used, they are associated with lower weight than keywords corresponding to the active user interface area.

The query system 132 causes the search result to be executed against a global or local index 210. In the embodiment shown, the query system receives the search result or results 212 and causes them to be output to a user 214. For example, the query system 132 may receive or generate an HTML page including the search results and transmit the HTML page to the client 102a.

FIG. 3 is a flowchart illustrating a method of reweighting a keyword based on an inactivity period in one embodiment of the present invention. In the embodiment shown, the query system 132 receives an event 302. The query system 132 associates a keyword of the event with an application user interface area 304.

Subsequently, the query system 132 receives a signal indicating that the application user interface area is inactive 306. For instance, if the user clicks on another application in a different user interface area, the user interface area that is the source of the signal becomes inactive. In the embodiment shown in FIG. 3, once a user interface area is inactive, implicit queries may rely relatively less on keywords associated with the inactive user interface area than they would when the user interface area is active.

The query system **132** next determines whether the amount of time that the user interface area has been inactive exceeds a threshold **310**. The threshold may be predetermined or may be based on user or client-specific attributes. For example, the user may explicitly set a limit for the amount of time keywords associated with an inactive user interface area are utilized to perform implicit queries. In an embodiment of the present invention, one or more thresholds may be used. In the embodiment shown, if the inactive time period exceeds the threshold, the keyword is either excluded from implicit queries or the results based on the keywords associated with the inactive user interface area are downweighted in comparison to a keyword associated with an active user interface area **312**. In another embodiment, the amount of downweighting is a function of the time since the keyword was extracted and/or the corresponding interface area was active.

The query system **132** executes the implicit query or causes the implicit query to be executed on a global or local index **314**. In response, the query system **132** receives search results **316**. The query system **132** then causes the query results to be output **318**.

EXAMPLE

FIG. 4 is a block diagram of a user display in one embodiment of the present invention. The embodiment shown includes a user interface screen **402**. The user interface screen **402** includes a content display **404** for displaying the results of search queries. The user interface screen also includes two user interface areas, application windows **406** and **408**. The application windows **406**, **408** contain two applications, Application **1** and Application **2**, respectively, i.e., the two applications execute within the respective windows. The user interface screen **402** also includes a toolbar **410**.

In a method according to one embodiment of the present invention, the user enters data into the application executing within the first window **406**. As the user enters data, one or more keywords in the data are associated with that window and implicit queries are generated based on the keywords. The results of the implicit queries are displayed in the content display **404**.

The user then clicks on the window in which the second application is executing **408**. In the embodiment shown, implicit queries continue to be generated periodically (e.g., once per second). Initially, after the first window becomes inactive, the implicit queries include keywords from both the first window **406** and the second window **408**. After a specified period of time, the keywords associated with the inactive window **406** are either downweighted in the query or result set, or the keywords are excluded from implicit queries. The time period may be equal to zero, i.e., those keywords are downweighted or excluded immediately when the window becomes inactive. The association of the keyword and the window may persist even though the keyword is not currently used in implicit queries.

In one embodiment, when the user clicks on the window in which Application **1** is executing **406**, the keywords that were previously associated with the window are used to form implicit queries. In this way, the content display provides search results to the user that are relevant to the application on which the user is focused or at least are relevant to the currently active application.

The foregoing description of embodiments of the invention has been presented only for the purpose of illustration and description and is not intended to be exhaustive or to limit the invention to the precise forms disclosed. Numerous modifications and adaptations thereof will be apparent to those skilled in the art without departing from the spirit and scope of the present invention.

That which is claimed:

1. A method comprising:

associating a keyword with a first user interface area; receiving a signal indicating that the first user interface area is inactive and a second user interface area is active; generating an implicit search query comprising the keyword; and

downweighting a measure of a result in a result set associated with the implicit query based at least in part on a period of time elapsed.

2. The method of claim **1**, wherein the first user interface area comprises a user interface window.

3. The method of claim **1**, wherein the first user interface area is associated with a first application.

4. The method of claim **1**, wherein the keyword comprises a plurality of keywords.

5. The method of claim **1**, wherein the first user interface area comprises a document.

6. The method of claim **1**, wherein the keyword comprises an attribute of an event.

7. The method of claim **1**, further comprising identifying the keyword in the first user interface area.

8. The method of claim **1**, further comprising discontinuing use of the keyword after a second period of time has elapsed.

9. The method of claim **8**, wherein the second period of time comprises at least 10 seconds.

10. The method of claim **1**, wherein associating the keyword with the first user interface area comprises storing the keyword and a user interface area identifier in a memory.

11. The method of claim **1**, further comprising:

receiving the result set associated with the implicit search query; and

causing the result set to be output.

12. A method comprising:

associating a keyword with a first user interface area; receiving a signal indicating that the first user interface area is inactive and a second user interface area is active;

determining an elapsed time since the first user interface area became inactive;

generating an implicit search query comprising the keyword;

receiving a result set associated with the search query; and downweighting a result in the result set based on the elapsed time.

13. A computer-readable storage medium on which is encoded program code, the program code comprising:

program code for associating a keyword with a first user interface area;

program code for receiving a signal indicating that the first user interface area is inactive and a second user interface area is active;

program code for generating an implicit search query comprising the keyword; and

11

program code for downweighting a measure of a result in a result set associated with the implicit query based at least in part on a period of time elapsed.

14. The computer-readable medium of claim 13, further comprising program code for identifying the keyword in the first user interface area. 5

15. The computer-readable medium of claim 13, further comprising program code for discontinuing use of the keyword after a second period of time has elapsed.

16. The computer-readable medium of claim 13, wherein program code for associating the keyword with the first user interface area comprises program code for storing the keyword and a user interface area identifier in a memory. 10

17. The computer-readable medium of claim 13, further comprising:
program code for receiving a result set associated with the implicit search query; and
program code for causing the result set to be output. 15

18. A computer-readable storage medium on which is encoded program code, the program code comprising:
program code for associating a keyword with a first user interface area; 20

program code for receiving a signal indicating that the first user interface area is inactive and a second user interface area is active; 25
program code for determining an elapsed time since the first user interface area became inactive;

12

program code for generating an implicit search query comprising the keyword;

program code for receiving a result set associated with the search query; and

program code for downweighting a result in the result set based on the elapsed time.

19. A method comprising:
associating a plurality of keywords with a plurality of user interface areas;

generating an implicit search query comprising at least one of the plurality of keywords;

receiving at least one query result based on the implicit search query; and

weighting at least one of the keywords included in the query and the at least one query result based at least in part on one or more of whether the user interface area associated with the keyword is active, the time since the keyword was determined, the time since the user interface area associated with the keyword was last active, and the frequency and time periods over which the user interface area associated with the keyword has been active.

* * * * *