

(19) World Intellectual Property Organization
International Bureau



(43) International Publication Date
24 January 2008 (24.01.2008)

PCT

(10) International Publication Number
WO 2008/010847 A2

(51) International Patent Classification:
G06F 7/00 (2006.01)

(21) International Application Number:

PCT/US2007/000742

(22) International Filing Date: 11 January 2007 (11.01.2007)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:

60/807,438 14 July 2006 (14.07.2006) US

11/461,575 1 August 2006 (01.08.2006) US

11/461,588 1 August 2006 (01.08.2006) US

(71) Applicant (for all designated States except US): **BEA SYSTEMS, INC.** [US/US]; 2315 North First Street, San Jose, California 95131 (US).

(72) Inventors; and

(75) Inventors/Applicants (for US only): **FRIEDEN, Kurt** [US/US]; 1562 San Lorenzo Avenue, Berkeley, California 94707 (US). **FARHANG, Dax** [US/US]; 3558 Willowood Drive, San Jose, California 95118 (US). **MAHENDRA, Samir** [IN/US]; 1555 5th Avenue #301, San Francisco, California 94122 (US). **QUEZADA, Jose** [US/US]; 1095 Natoma Street, Unit 5, San Francisco, California 94103 (US).

(74) Agents: **MEYER, Sheldon, R.** et al.; FLIESLER MEYER LLP, 650 California Street, Fourteenth Floor, San Francisco, California 94108 (US).

(81) Designated States (unless otherwise indicated, for every kind of national protection available): AE, AG, AL, AM, AT, AU, AZ, BA, BB, BG, BR, BW, BY, BZ, CA, CH, CN, CO, CR, CU, CZ, DE, DK, DM, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, GT, HN, HR, HU, ID, IL, IN, IS, JP, KE, KG, KM, KN, KP, KR, KZ, LA, LC, LK, LR, LS, LT, LU, LV, LY, MA, MD, MG, MK, MN, MW, MX, MY, MZ, NA, NG, NI, NO, NZ, OM, PG, PH, PL, PT, RO, RS, RU, SC, SD, SE, SG, SK, SL, SM, SV, SY, TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VC, VN, ZA, ZM, ZW.

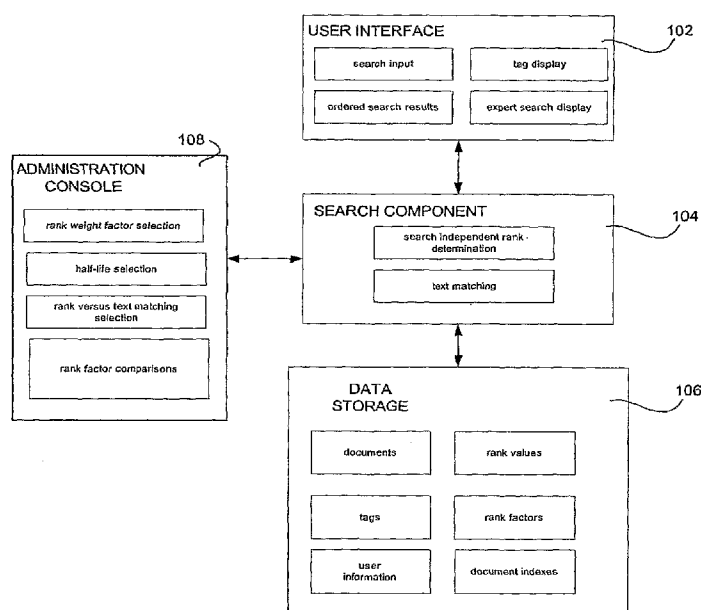
(84) Designated States (unless otherwise indicated, for every kind of regional protection available): ARIPO (BW, GH, GM, KE, LS, MW, MZ, NA, SD, SL, SZ, TZ, UG, ZM, ZW), Eurasian (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European (AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HU, IE, IS, IT, LT, LU, LV, MC, NL, PL, PT, RO, SE, SI, SK, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, ML, MR, NE, SN, TD, TG).

Published:

— without international search report and to be republished upon receipt of that report

[Continued on next page]

(54) Title: IMPROVED ENTERPRISE SEARCH SYSTEM



(57) Abstract: Ranks for documents can be made by calculating coefficients indicating connections between users and documents. The coefficients can be used to calculate search-independent rank values for the documents. The search-independent rank values can be combined with term matching indications to get a total relevance of the document.



For two-letter codes and other abbreviations, refer to the "Guidance Notes on Codes and Abbreviations" appearing at the beginning of each regular issue of the PCT Gazette.

IMPROVED ENTERPRISE SEARCH SYSTEM

CLAIM OF PRIORITY

U.S. Provisional Patent Application No. 60/807,438 entitled "Enterprise Search System", by Kurt Frieden *et al.*, filed July 14, 2006 [Atty. Docket No. BEAS-02040US0].

U.S. Patent Application No. 11/461,575 entitled "Enterprise-Based Tag System", by Kurt Frieden *et al.*, filed August 1, 2006 [Atty. Docket No. BEAS-02040US5].

U.S. Patent Application No. 11/461,588 entitled "Administration Console to Select Rank Factors", by Kurt Frieden *et al.*, filed August 1, 2006 [Atty. Docket No. BEAS-02040US6].

COPYRIGHT NOTICE

A portion of the disclosure of this patent document contains material that is subject to copyright protection. The copyright owner has no objection to the facsimile reproduction by anyone of the patent document or the patent disclosure, as it appears in the Patent and Trademark Office patent file or records, but otherwise reserves all copyright rights whatsoever.

BACKGROUND OF INVENTION

Search systems want to improve the quality and relevance of the top hits to improve the chances that the documents found by the searcher will be the documents that the searcher is looking for. Google™ uses the concept of links between documents in the Internet to determine page rank. Pages linked to by other highly ranked pages are ranked relatively high. The Google™ approach is ineffective for enterprise portal and other enterprise wide document systems since documents in such systems tend not to be highly interlinked.

BRIEF DESCRIPTION OF THE DRAWINGS

Figure 1A is a diagram of one embodiment of the present invention.

Figure 1B is a search display page showing tags associated with documents.

Figures 2A-2C illustrates an exemplary approach to creating a document rank of one embodiment.

Figure 3 shows an example of a matrix of one embodiment.

Figure 4 illustrates a flow chart of one embodiment.

5 Figure 5 illustrates an exemplary search page.

Figures 6A -6B illustrate administration console pages for selecting rank factors.

Figures 7A-7B illustrates tag administration pages.

DETAILED DESCRIPTION

10 Figure 1A shows an exemplary system of the present invention. User interface 102 can be a web page or other interface for getting user information and displaying results to a user. The user interface 102 can be used to input search terms to find objects. The objects can include documents, users, and tags. The documents can include word processing documents, images, web pages, discussion threads and any
15 other type of files. The user interface 102 can be used to display search results including ordered search results. Tags associated with the documents can also be displayed. Software component 104 can use information stored in memory 106 to provide functions of the present invention.

The search component 104 can produce search independent ranks for objects in
20 the system. The search component 104 can also provide for text matching of objects. The ordered results provided to the user can be a function of the search independent object rank and the text matching. This function and other rank factors can be selected by a system administrator from administrative console 108.

Each object (user, document and tag) can have search-independent rank of its
25 quality which does not depend on any search query. Each object's search-independent rank can be calculated before search time. This search-independent rank can be combined with a text matching score at search time to determine the order of results. For example, in one embodiment, where a is a value from 0 to 1:

$$\text{Relevance} = a (\text{search independent document rank}) + (1-a)(\text{text matching score})$$

30 The search-independent ranks can be determined in a variety of ways. For example, the search independent ranks of objects can be seen as contributions from

other objects based on a combination of actions with their associated weights and the contributor object's rank. In one embodiment, the search independent object rank can be implemented using matrix equations, such as using a damped, positive, column-stochastic matrix.

5 Figure 1B shows an exemplary display showing the use of tags to search for documents to the displayed associated with search for documents.

Object Rank Calculation

Embodiments of the present invention concern search independent object rank calculations. In one embodiment, coefficients indicating connections between objects
10 can be calculated. These coefficients can be determined based on user actions such as creating, viewing, and tagging documents. In one example, user actions are given a selectable action weight in calculating the coefficients. The coefficients can be used to calculate rank values for the objects.

In one embodiment, the rank of a user can depend on:

- 15 • The rank and number of pages and tags she creates
 • The rank and number of users who tag, view, and add her as a contact

In one embodiment, the rank of a page can depend on:

- The rank of its author
 • The rank and number of users who tag and view it

20 In one embodiment, the rank of a tag can depend on:

- The rank and number of people who apply and use it
 • The rank and number of page to which it is applied

 The ranking schema can be separate from the search schema and it can be supported on a different database server. This can isolate real-time production systems
25 from the impact of the ranking calculation.

A static copy of the ranking schema can be obtained for the rank calculation. This allows for data integrity and isolation.

 The coefficients can be part of a matrix indicating connections between objects, such as documents, tags and users. The matrix can be used to calculate a modified
30 matrix, such as a damped matrix, used to calculate an eigenvector solution containing the ranks.

Figures 2A-2C show one example of a method to determine connections between objects, such as documents, users and tags. In this example, directed lines show authority given from one object to another. In figure 2A, Bill creates a page (producing a weight of "10" to the page and vice versa), clicks on a tag (giving a weight of "1" to the tag); and adds a user Jill to his contacts (giving a weight of "3" to Jill). Figures 2B and 2C show the result of Jill's and Jack's actions.

Figure 3 shows an example of a matrix for the example of Figures 2A-2C. A column of the matrix shows an object's contribution to other objects expressed as a ratio of the object's total contribution to all of the other objects. For example, column 302 has the coefficients of the contribution of Jill to other objects. The rows indicate the coefficients of the incoming contributions to an object. For example, row 304 indicates the coefficients of incoming contributions for page 1.

In figure 3, X is an eigenvector of the matrix equation. The coefficients of the eigenvector could indicate the search independent rank values of the objects. Because of the size of the matrix, it can be hard to find the eigenvector solution to such a matrix equation. As described below, one way to obtain rank values is to use a damped matrix that can be solvable using the Perron-Frobenius Theorem.

The objects in the system can be enumerated $O_1, \dots, O_n$. W_{ij} can denote the total weight of all the connections between O_j and O_i divided by the total weight of all of O_j 's connections. x_i can denote the coefficient for object O_i of eigenvector X of figure 3. This means:

$$x_i = W_{i1}x_1 + \dots + W_{in}x_n.$$

which is a series of n equations with n unknowns.

The formula can be slightly modified so that it can be solved using the Perron-Frobenius Theorem. g_i can denote the rank of O_i . The parameter d can be a damping factor that can be set between 0 and 1. W can be the nxn matrix whose entries are W_{ij} , g can be the 1xn column vector whose coefficients are g_i , and E can be the matrix whose entries are all $1/n$. The damped formula can be expressed as:

$$g = Gg$$

where

$$G = (1-d)W + (d)E$$

Because of the damping, $\mathbf{G}$ is positive. $\mathbf{W}$ by itself is usually not positive and typically has many zero coefficients. Because $\mathbf{E}$ and $\mathbf{W}$ are both column-stochastic with the values in each column adding up to 1, $\mathbf{G}$ is column-stochastic. $\mathbf{W}$ is column-stochastic because the values in each column represent the relative outgoing connection weights for each object.

The Perron-Frobenius Theorem tells us that $\lim_{k \rightarrow \infty} \mathbf{G}^k \mathbf{g}_0$ exists for any choice of an initial starting vector $\mathbf{g}_0$, as long as its coordinates add up to 1. The theorem also states that the limit is an eigenvector of $\mathbf{G}$ with eigenvalue 1, so the limit must be $\mathbf{g}$. This provides a way to calculate $\mathbf{g}$. The initial vector $\mathbf{g}_0$, can be repetitively multiplied with the matrix until the values settle down. The initial vector $\mathbf{g}_0$ can be $[1/n, \dots, 1/n]$.

Other Initial vectors can also be used. In one embodiment, the coefficients relating to different object categories, such as users, tags and documents, in $\mathbf{g}_0$ can use different constants. For example, if users as a category tend to be ranked higher than documents as a category, the initial vectors values can reflect this.

Alternately, $\mathbf{g}_0$ can be calculated by setting $\mathbf{g}_0$ equal to the sum of all of the coefficients of the row i of $\mathbf{G}$ scaled by a factor to make the sum of the coefficients of $\mathbf{g}_0$ equal to 1.

$\mathbf{g}_0$ can be determined from a previously calculated rank vector. For example, if objects have been added, the coefficients of the previous rank vector can be used to determine some of the initial rank vectors values. New objects can be assigned constants for the initial vector.

The $\mathbf{g}_0$ can also be the result of one or more multiplications of a precursor vector with the undamped matrix followed by a rescaling.

Matrix Calculation Method

One embodiment of the present invention comprises a computer-implemented method for operating on a large matrix that is too unwieldy to maintain in local memory. Such a method can be used for the matrix calculation of object ranks. The method can include using a core data structure. The core data structure can be stored in external memory and brought in to local memory row by row for the calculation.

In one embodiment, for each row of a core data structure, a row of the core data structure is brought into local memory. The row can be inflated by inserting missing

zeros in the row. This can be significant if the matrix is a sparse matrix. The inflated row can be converted into a row of a damped matrix. The damped matrix can be positive and column-stochastic. The row of the damped matrix can be multiplied by the current vector to get a value of the next vector. For example:

5
$$\text{row}_i \times \text{old vector} = \text{next vector}[i]$$

The next vector can be compared with the current vector to get a difference value. If the difference value is greater than a minimum error value, the next vector can be set as the current vector and the steps can repeat otherwise, a result is determined from the next vector.

10 In one example, the next vector is used to determine the ranks of objects.

The core data structure can include skip counts since the core data structure is likely to be sparse. Skip counts can indicate the number of zero coefficients between each non-zero coefficients of the sparse matrix and thus allow the core data structure to be inflated.

15 In one embodiment, the first byte of a skip count can encode a number of next zero values in a row if the number is less than a threshold or an indication of additional bytes that encode the number if the number is greater than a threshold. This can aid in the packing of the core data structures.

20 Figure 4 shows an example of an exemplary method. Step 402 includes initializing the initial vector g_0 . One example of a g_0 is the vector $[1/n, \dots, 1/n]$ whose coefficients add up to "1".

In one embodiment, for each iteration of the algorithm, for $i = 1$ to numRows:

- o Read in row of core(A) (step 406)
- o Inflate this into one row of A (step 408)
- 25 o Convert this into a row of G and multiply this row by g_k to produce i^{th} element of g_{k+1} (step 410)
 - In detail: for $j = 1$ to numColumn
 - o Stochasticise a_{ij} using the j^{th} column sum
 - o Use damping to produce g_{ij}
 - 30 o $g_{k+1}[i] += g_{ij} * g_k[j]$
- o Calculate e_k from g_k and g_{k+1}

As shown in step 412, the method can repeat until an error condition is met. Alternately the method can be repeated for a fixed number of times as shown in step 412.

Tag-Based Enterprise System

5 One embodiment of the present invention is a tag-based system for the enterprise. Users can apply tags to objects. The tags can be used to provide user access to enterprise objects, such as documents.

One embodiment of the present invention is a system that automatically creates initial tags for objects. The tags can automatically be created based on document
10 location information. For example, documents in a folder entitled "project X" can be given that name as an initial tag. Existing document metadata can also be used to create initial tags. For example, WordTM or other types of documents can have metadata that can be examined to determine tags.

Initial tags can automatically be created using translation rules. The translation
15 rules can be such that if a first term is associated with the document, a second term can be used as the initial tag. For example, all documents with the folder name "Jamesk" can be associated with a tag "James Kite" if a translation rule so indicates this relationship. The first term can be a folder name, metadata, a document name or other type of term.

20 Tagging can allow users to accurately define the knowledge encapsulated by the content in a distributed fashion. Tags can be terms associated with objects. However, unlike traditional document metadata or properties, tags can be primarily defined by the content users. Tag ownership and administration can be decentralized. While a document property can be defined by a single individual, the user base as a whole can
25 determine the knowledge embodied by a particular document.

The tags can form a folksonomy. Unlike taxonomies that are rigid, these folksonomies can be constantly evolving to reflect the aggregated wisdom of the user base.

System users can still be able to utilize document metadata as search criteria or
30 to further refine result sets. This can ensure that results are returned when no applicable tags exist. When exposed as a preference, it can allow individuals to choose whether

they trust the crowd or a single individual. For example, a user might select the tag named “operator” and sort or filter the result set to display document authored by Jane Smith.

5 The application can also be able to auto-tag documents with terms using document metadata or logical attributes of the document using a system rule.

The tags can be used in a search for users. One embodiment of the present invention can include associating users with tags and using connections between the tags and users to determine rank values for the users.

10 The connections between the user and objects can be used to classify the users. Users can be classified as experts. For example, an expert search can search for experts associated searches by examining the tags written about the expert, documents that the experts have written which are associated with tags, or tags that the expert creates. The expert search can automatically occur along with a document search.

15 In one embodiment of the present invention, searching for experts can be based on search terms. For example, experts can be returned based on their association with the objects found in a search. The objects can be, for example, documents associated with users, tags associated with users, or user profile pages.

20 The system can allow end-users to more easily locate experts. End-users can be able to directly identify another end-user as an expert by adding a tag with that user. For example, an end-user can be able to indicate that “Jane Smith” is an expert on “java” by associating the “java” tag to Jane. The application can also derive experts from usage statistics.

25 In some cases, users will not be able to find the information they are looking for. This might be because the user is looking in the wrong location, or the user is looking for a level of detail that is not covered in the available content. Some users just prefer to talk to people instead of reading a document. In each of these circumstances, users will want to locate other individuals who might be able help them fulfill their knowledge discovery needs. Expert identification can include returning a list of experts based on a search query for documents.

30 The system can derive the panel of experts using tracked user actions. For example, the author of the most relevant document in a result set can be identified as

one of the experts. Each user can be measured based on the same set of metrics to determine that user's expertise score.

The expertise score can be determined from metrics such as: links between users and documents (authorship, submitting, tagging, viewing); links between users (users tagging other users); and text in the user profile page (if the search matched any of the tags applied to the user).

The users with the top scores can be displayed by default. An administrator can be able to set the number of users that are displayed from the administrative interface.

Users can also be able to tag other users. As noted above, these tags can also be used when deriving the panel of experts. In one embodiment, of the various metrics, the text in the user profile page will be weighted the highest.

For example, if Jane has been tagged with the term "java guru", then Jane can be returned at or near the top of the list of experts when a user searches for *java guru* or clicks the *java guru* tag.

Experts can be displayed in a separate pane in the search page. Clicking on a user's name in the list can open up the user's profile page.

In some cases, it can be advantageous for end-users if they can create a private library of information. The system can allow users to create both personal and custom libraries of tags. Personal tags can be explicitly associated with a single user. In one embodiment, no other end-user will be able to edit the personal tags. Custom views can be controlled using a common security service as an underlying foundation. Through this mechanism, end-users can be able to combine the information contributed by any combination of users and groups to create a custom library. Security on the documents within each view can still be respected across the application. If a user creates a new tag and associates it with a particular document, a different user will only be able to see that tag if they have access to the document itself. Through this methodology, the system can leverage the common security service to create virtual libraries of knowledge without being forced actually segment the information.

The system can allow users with the appropriate capability to create multiple views of the information. A view can be a filter on the information in the system.

These filters can be applied to tags and usage statistics. In one embodiment, document display will be determined by security.

Everyone: This view can be the default view in the system. It can display all tags and all usage history can be used to rank result sets. This view may also be referred to as the global view.

Personal: Unlike the global view, the personal view can display only those tags which have been applied by a single-user. Each user will be able to toggle to their personal view.

Custom: End-users can be able to define custom views as well. In custom views an end-user can select the user(s) and group(s) that will be considered part of the view. Custom views can filter the tags only to those tags which have been associated with content by members of the specified view. The users and groups are the same entities that exist in the deployment. Usage history can also be filtered by group view. Content can have a different ranking from one group to the next. This will allow groups to define content as it is relevant to them without vying for relevance with another definition. For example, two users may be looking for entirely different sets of information when they each submit the term *operator*. Group delineation can satisfy this need by allowing the information that is relevant to each group to bubble up to the top of the result set through usage history. The number of views that each user can define can be determined by an administrator.

An end-user can select experts and elect to preview the view using those experts as criteria. From the preview view UI, an end-user can elect to create a new view or add the users (experts) to an existing custom view. An end-user can also elect to select, create, edit, or delete a custom view using a custom view menu.

End-users can be able to execute both full-text and parameterized queries. Full-text queries can search within all of the content that is indexed for each object. Parameterized queries can allow end-users to query specific properties or metadata.

Figure 5 shows a representative search page. Each search can return a content result set, a set of associated tags, as well as a list of experts on the result set. The display of experts can be something that an administrator can disable. The content and expert results can be returned based on the rank associated with each object in the

system. The set of associated tags that are displayed can be determined by the end user's preference and the tags that are associated with the content in the result set.

The system can provide user preferences and advanced search options. The advanced options can include sorting, filtering, metadata display, the content query language, and right-click options.

Users can sort result sets based on any column heading the in the results pane. This can include the ability to sort by relevance, name, object type, last modified date, and author. Results can be sorted by query relevance by default for each end-user session. Any changes to the sorting preference can be enabled for the remainder of the end-user's session. When a result set is sorted by a property that has multiple equal values, query relevance will be used as the secondary result ordering.

An advanced query build can allow an end-user to build a complex query without understanding the content query language. They can select words to include (or exclude) from the search result. End-users can search for explicit tags using the advanced search UI. Users can also filter their result set based on the value of a particular property on the content.

Users can also be able to determine which properties are displayed in the details section of each document result. Similarly, to property filtering, the list of available properties can be determined by the properties that are defined as searchable.

Users can also be able to explicitly execute a parameterized search either through search query language or an advanced search UI. For example the query, *author:Jane*, can query the objects to return results which contain "Jane" as part of the value for the "author" property.

The system can use a query independent way of assigning a rank to users, tags and pages. This can be computed ahead of time in order to improve performance, and it can be combined with the term frequency search algorithm to achieve good ranking in search results.

The search independent rank calculation can be done periodically. There can be a threshold number of searchable objects and user activity which can force the customer to install the search independent Rank Engine on a separate machine from the web server.

Application administrators can use an administrative interface to modify or delete tags. In this interface, administrators can be able to perform these operations against a single tag or all instances of a tag. Figure 7A shows an exemplary tag administration interface. From this UI, administrators can search for any tag that is in the system. Administrators can also restrict their search to manual tags, auto tags, or all tags. The interface can display the information about each tag such as, name, Rank score, total number of people who have applied the tag, total number of documents the tag has been associated with, total number of users the tag has been associated with, if the tag is restricted, date the tag was created and date the tags was last applied.

The administrator can delete or rename a tag by selecting the checkbox next to the tag and selecting the delete or rename buttons respectively. The administrator can also restrict a tag (mark it as inappropriate) by selecting the checkbox and selecting the restrict button. If an administrator restricts a tag, which is already in use, then the application can warn the administrator that the tag already exists.

Administrators can have the ability to add and delete terms from a list of restricted tags, as shown in figure 7B. Restricted tags are terms that cannot be used as tags on documents or users. Administrators can also have the ability to bulk upload a list of inappropriate words. Inappropriate tags can also be stemmed and they will apply to multi-word tags. For example, if an administrator adds "idiot" to the list. Then both "idiots" and "idiot proof" can be automatically disallowed.

Administrators can also be able to administrate auto-tags. Auto-tags are tags that are programmatically applied to content. This feature can be commonly used when content is imported. Auto-tagging can also be used during the initial product installation to seed an existing index with tags. Auto-tag values can be reconciled after they have been created. For example if the value in an auto-tagging rule changes, then the values that were previously applied via that rule can be modified. If a rule is deleted then all values that were applied via that rule can be deleted.

Administrators can define auto-tagging rules through a simple rules administrator. Rules can be associated with specific folders within the system hierarchy. Each rule can also be associated with a particular object type and content type if the target object(s) are documents. Each folder, object type, and document type

can have multiple rules associated with it. Auto-tagging values can be either an explicit string or the value of a property. The list of applicable properties can be determined by the document properties that are associated with the specific object type. An administrator will have the ability to control tags on end-users. A role-based security model can be used based on an Access Control List (ACL) management.

A role can be a collection of capabilities, or rights. Every object type in the system can have associated with it a set of capabilities, such as create, read, update, manage and delete. For a given role, users can define a set of capabilities for each object type; for example, the 'Librarian' role might have the ability to create and prescribe Views, where the 'Tagging User' role may instead have the ability to create Views, but not prescribe them. Once a role is defined, users/groups can then be mapped to those roles.

The system can have a set of out-of-the-box roles to which users can be mapped. These roles are intended to help customers get a head start in securing their system.

Custom roles can also be defined. Users and groups can be mapped to roles. When a user or group is mapped to a role, they can inherit the capabilities afforded by that role.

Correct resolution of content authors to users can be important for the expert system. In order to achieve this there can be an administrative UI where an administrator can select an end-user and apply all of the aliases that this user might be identified as. This list can be prioritized from top to bottom. So when a document is imported into the system, the author can be resolved to the first user in the list with a matching alias. Customers can also use an asterisk to indicate a wildcard match. This can be used to make sure that a specific user is applied as the author in the event that no explicit match is found. If the wildcard is not used and no match is found, then the value in the author property will be displayed as the "author" of the page. This can also be denoted as "unqualified" (i.e. not confirmed) in the UI.

The browser toolbar can provide the system a full-time browser presence. It can also provide users an easy mechanism to search, submit, and tag content. Rather than navigating to the application and submitting via the system UI, the end-user can be able to interact directly with system from any location on the web.

An office toolbar can allow end-users to easily submit an office document to the system without leaving the native office application. Similar to the browser toolbar, when a user elects to submit a document via the office toolbar, they can have the ability to define the title and tags associated with the document in the system.

5 In one embodiment, the font size of the tags is determined by the search-independent ranks of the tags. Tags with a greater rank can have a greater tag font size. This can aid users by indicting the more valuable tags.

End-users can be able to browse tags. A variety of UI implementations can be used for tag navigation. The system may incorporate all, some or one of these
10 implementations based on ongoing UI discussions.

Tag Cloud: This is the most common tag navigation mechanism used today. In the tag cloud each tag's font weight can determined by the number of documents associated with it. So tags with a large number of documents will display as larger tags, and can be thought of as "broader" categories. The search-independent ranks of the tags
15 can also be used.

Tag List: The tag list is a simple method for tag display. In the tag list, each tag can be displayed using the same font weight. The number of documents associated with each tag should be displayed as well. Users can be able to sort the tag list alphabetically or by the number of associated documents.

20 Tag Tree: The tag hierarchy could also be displayed in a windows-like tree structure. In this navigation paradigm, each tag can be displayed as a folder. In this UI a tag could be the child of multiple folders.

Administration Console to Select Rank Factors

One embodiment of the present invention is an administration console that
25 allows a user to input rank factors. The rank factors can be used to adjust the operation of the system. The administration console can use a graphical element, such as a slider, to allow users to select the relative weights.

An exemplary rank factor is an indication of the relative weight of search-independent ranks and text matching and a search component to use the relative weight
30 indication to order the results of searches.

A linear combination of the search independent ranks and the text matching can be used to order the search results. A relative weight indication can be used to determine the linear combination.

Figure 6A shows an exemplary page for setting rank factors and the half-life of some transactions.

Administrators can have the ability to modify the values in the rank-scoring algorithm. In addition, they can take snapshots of the values so that they can be used later. This can ease administration since the administrator will not be forced to document the various values before changing them.

Figures 6A and 6B show exemplary ranking factors that can be modified for objects, such as documents, users, and tags. In this example, each factor can be modified using the slider or by modifying the value in the text box to values between 0 and 1.

The administration console can allow a user to select an indication of how the importance of certain actions to search-independent ranks decreases over time and a search component to update the search independent ranks using the indication. The indication can be a half life indication that reflects the decrease of the importance of a user viewing or tagging an object over time.

Over time the documents that are tagged and viewed the most can continue to rise in the result set. This can create a positive feedback loop since many users often open one or more results at the top of the result set, regardless of relevance. In order to mitigate this cycle, administrator can define the half-life for these values. The half-life can allow an administrator to make the tags applied and number of views less valuable over time. The shorter the half life, the quicker the application will “forget” about the previous tags applied or views of the content.

Figure 6B outlines miscellaneous settings that an administrator can be able to set. Manual submissions to the system can upload the document to a directory. The administrator can have the ability to define the target folder via these settings. The administrator can also define the analysis sample size. This is the number of search results that the application will consider when displaying both the associated tags and experts. From this UI, the customer can also modify the scheduling of the operation

that calculates the rank on each object. Administrators can also determine the balance between search-independent ranking and the term frequency ranking built into the Search.

A statistics collection component can be used to collect statistics concerning user interaction with search result pages. The administration console can allow the display of comparisons of statistics collected on searches with different selected indications. This can allow the user to tweak the values to improve the search function.

The administration console can display a comparison of the order of selected objects on searches with the different indication values. Statistics can include an indication of the average order of a selected object in response to a search.

An admin page can let administrators analyze how the rank was determined for a particular object and general data on how successful end user searches are. In one embodiment, the following metrics can be available for the administrator: total number of documents, total number of users/experts and total number of tags. In addition to the totals listed above, administrators can have the ability view the metrics below. Exemplary metrics can include: total documents accessed and % of total available, total tags accessed and % of total available, total users active and % of total available, total experts accessed and % of total available, average rank of document access (normalized against the size of all result sets), average rank of expert access (normalized against the size of all result sets) and total number of orphaned searches.

An administrator can also be able to select any object in the system and view the values from the ranking algorithm that determine that objects overall rank in the system. This can help administrators to understand why some objects are ranked very high and why others are not.

Usage tracking can help the system improve the quality of results for the end-user. First, through the analysis of tracked events the system can improve the ranking of result sets that are returned against a particular search. For example, the application can track the fact that most users after searching for "operator" or clicking on the "operator" tag all opened the same document. With this quantitative calculation, the application can increase the relevancy ranking of the document for future searches on

“operator”. Conversely, the relevance ranking of documents associated with “operator” that are rarely accessed can decrease at the same rate.

Usage tracking can also help the application suggest terms or documents that might be related or worth review. In one example, if many users who searched “operator” also searched for “conductor”, the system could suggest the additional term “conductor” to users who search for “operator”.

This level of usage tracking can remain anonymous to the user base. While a user can see that another user executed a series of subsequent actions when searching on the same term, users will not be able to see exactly who searched on a particular term or selected a specific document. This can help ensure user privacy.

One embodiment may be implemented using a conventional general purpose or specialized digital computer or microprocessor(s) programmed according to the teachings of the present disclosure, as will be apparent to those skilled in the computer art. Appropriate software coding can readily be prepared by skilled programmers based on the teachings of the present discloser, as will be apparent to those skilled in the software art. The invention may also be implemented by the preparation of integrated circuits or by interconnecting an appropriate network of conventional component circuits, as will be readily apparent to those skilled in the art.

One embodiment includes a computer program product which is a storage medium (media) having instructions stored thereon/in which can be used to program a computer to perform any of the features present herein. The storage medium can include, but is not limited to, any type of disk including floppy disks, optical discs, DVD, CD-ROMs, micro drive, and magneto-optical disks, ROMs, RAMs, EPROMs, EEPROMs, DRAMs, flash memory or media or device suitable for storing instructions and/or data stored on any one of the computer readable medium (media), the present invention includes software for controlling both the hardware of the general purpose/specialized computer or microprocessor, and for enabling the computer or microprocessor to interact with a human user or other mechanism utilizing the results of the present invention. Such software may include, but is not limited to, device drivers, operating systems, execution environments/containers, and user applications.

The forgoing description of preferred embodiments of the present invention has been provided for the purposes of illustration and description. It is not intended to be exhaustive or to limit the invention to the precise forms disclosed. Many modifications and variations will be apparent to one of ordinary skill in the relevant arts. For example, steps preformed in the embodiments of the invention disclosed can be performed in alternate orders, certain steps can be omitted, and additional steps can be added. The embodiments where chosen and described in order to best explain the principles of the invention and its practical application, thereby enabling others skilled in the art to understand the invention for various embodiments and with various modifications that are suited to the particular used contemplated. It is intended that the scope of the invention be defined by the claims and their equivalents.

CLAIMS

What is claimed is:

- 5 1. A computer-implemented method comprising:
associating users with tags; and
using connections between the tags and users to determine rank values for the
users.
- 10 2. The computer-implemented method of claim 1, wherein a user is associated with
tags because the tags are written by the user.
3. The computer-implemented method of claim 1, wherein searches use the rank
values of the users in the calculations.
- 15 4. A computer-implemented method comprising:
associating users with tags; and
using connections between the tags and users in a search for users.
- 20 5. The computer-implemented method of claim 4, wherein a user is associated with
tags because the tags are written by the user.
6. A computer-implemented method comprising:
associating objects with users; and
25 using connections between the user and objects to classify the users.
7. The method of claim 6, wherein objects include documents.
8. The method of claim 6, wherein objects include tags.
- 30 9. The method of claim 6, wherein objects include other users.

10. The method of claim 6, wherein users are classified as experts.

11. A computer-implemented method comprising:

5 associating tags with objects;

providing multiple filtered views of the objects.

12. The computer-readable method of claim 11, wherein one filtered view is a group
10 view.

13. The computer-readable method of claim 11, wherein one filtered view is a personal view.

14. The computer-readable method of claim 11, wherein one filtered view is a full
15 view.

15. A computer-implemented method comprising:

associating tags with objects; and

20 aggregating other users tags into custom views.

16. A computer-implemented method comprising:

searching objects with search terms; and

25 returning experts based on association with the objects.

17. The computer-implemented method of claim 16, wherein the objects include tags associated with the user.

18. A computer-implemented method comprising:

30 creating a tag rank based on the rank of other objects in a system;

using the tag rank to determine a font size of a tag display.

19. The computer-implemented method of claim 18, wherein the display of tags is a tag cloud.

20. The computer-implemented method of claim 18, wherein the display is provided
5 in response to a search.

21. The computer-implemented method of claim 18, wherein the rank is determined using a matrix equation.

22. The computer-implemented method of claim 18, wherein the tags are shown
10 associated with documents.

23. A computer-based search system comprising:

An administration console that allows a user to input an indication of the relative
15 weight of search-independent ranks and text matching; and
a search component to use the relative weight indication to order the results of searches.

24. The computer-based search system of claim 23, wherein the administration
console uses a graphical element to allow users to select the relative weight.

25. The computer-based search system of claim 24, wherein the administration
console uses a slider to allow users to select the relative weight.

26. The computer-based search system of claim 23, wherein the search logic uses
25 connections between users and documents to create the search independent ranks.

27. The computer-based search system of claim 23, wherein the search logic uses
connections between tags and documents to create the search independent ranks.

28. The computer-based search system of claim 23, wherein the search system uses
30 matrix operations to create the search independent ranks.

29. The computer-based search system of claim 23, wherein the search system uses a linear combination of the search independent ranks and the text matching to order the search results, wherein a relative weight indication is used to determine the linear combination.

30. A computer-based search system comprising:
an administration console to input indications of the importance object ranking factors; and
search logic that uses the indications to determine objects search-independent ranks, the search independent ranks being used to order the results of searches.

31. The computer-based search system of claim 30, wherein the administrative console uses graphical elements to allow users to select the indications.

32. The computer-based search system of claim 31, wherein sliders are used to allow users to select the indications.

33. The computer-based search system of claim 30, wherein the administration console displays default values of the indications.

34. The computer-based system of claim 30, wherein the indications include indications that concern how rank is applied to documents.

35. The computer-based system of claim 30, wherein the indications include indications that concern how rank is applied to tags.

36. The computer-based system of claim 30, wherein the indications include indications that concern how rank is applied to users.

37. The computer-based system of claim 30, wherein the administration console allows the user to select an indication of how the importance of certain actions to the ranks decreases over time.

5 38. A computer-based search system comprising:
an administration console that allows a user to select an indication of how the importance of certain actions to search-independent ranks decreases over time; and search component to update the search independent ranks using the indication.

10 39. The computer-based search system of claim 38, wherein the indication is a half life indication.

40. The computer-based search system of claim 38, wherein the indication determines the decrease of the importance of a user viewing an object over time.

15

41. The computer-based search system of claim 38, wherein the indication determines the decrease of the importance of a user tagging an object over time.

42. A computer-based search system comprising:

20 an administration console that allows users to select at least one indication pertaining to search independent rank;

a search component to use the indication to order the results of searches; and

a statistics collection component to collect statistics concerning user interaction with search result pages; wherein the administration console allows the display of
25 comparisons of statistics collected on searches different indications.

43. The computer-based search system of claim 42, wherein the indication is a weight of search indication ranks and text matching.

30 44. The computer-based search system of claim 42, where the indication is an object factor ranking indication that is used to create search independent ranks for objects.

45. The computer-based search system of claim 42, wherein the indication indicates how the importance of certain actions on search-independent ranks declines over time.

5 46. The computer-based search system of claim 42, wherein the statistics include information concerning the order of user selected objects from a search.

47. The computer-based search system of claim 46, wherein the administration console display a comparison of the order of selected objects on searches with the
10 different indication values.

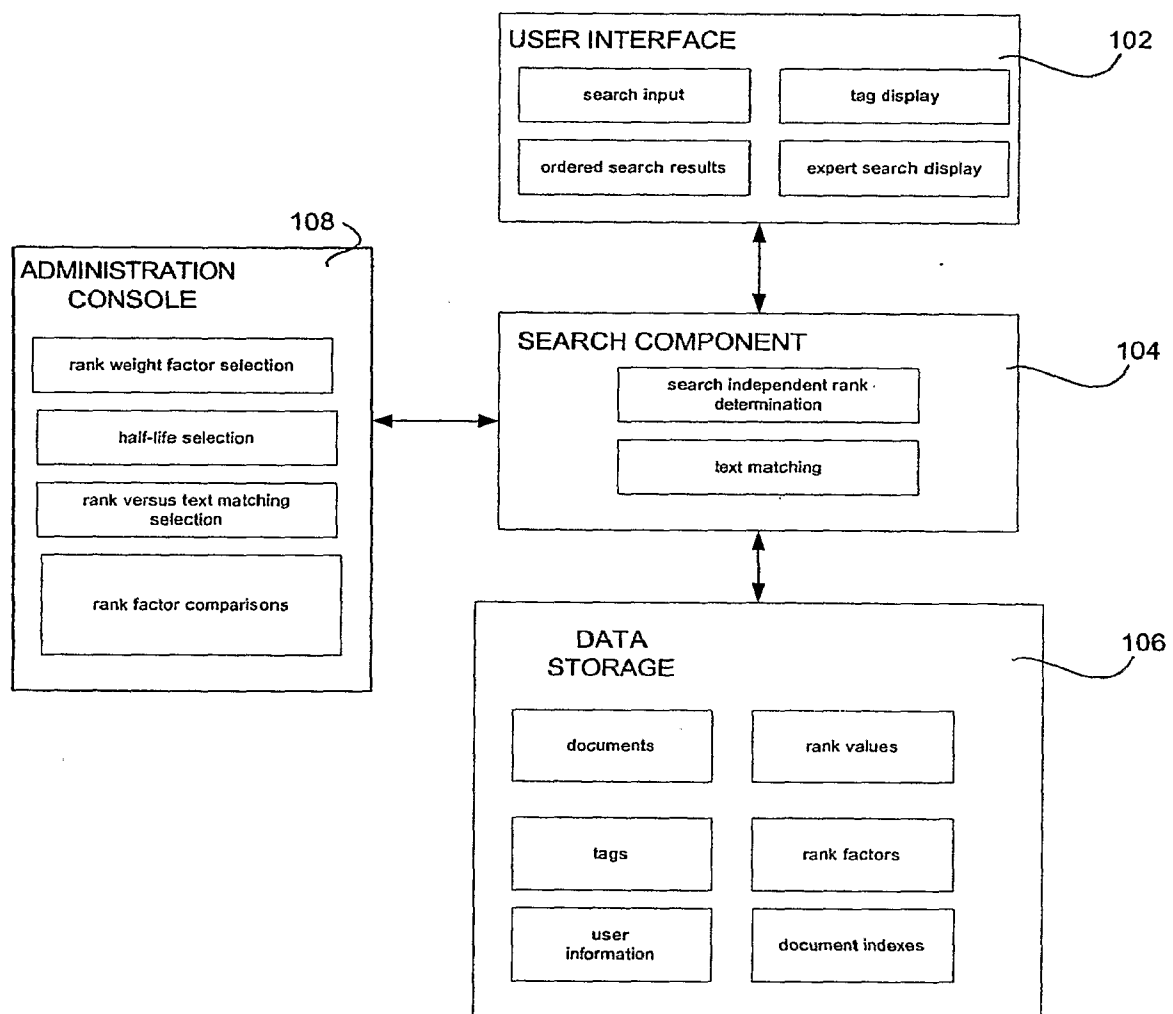


FIG. 1A

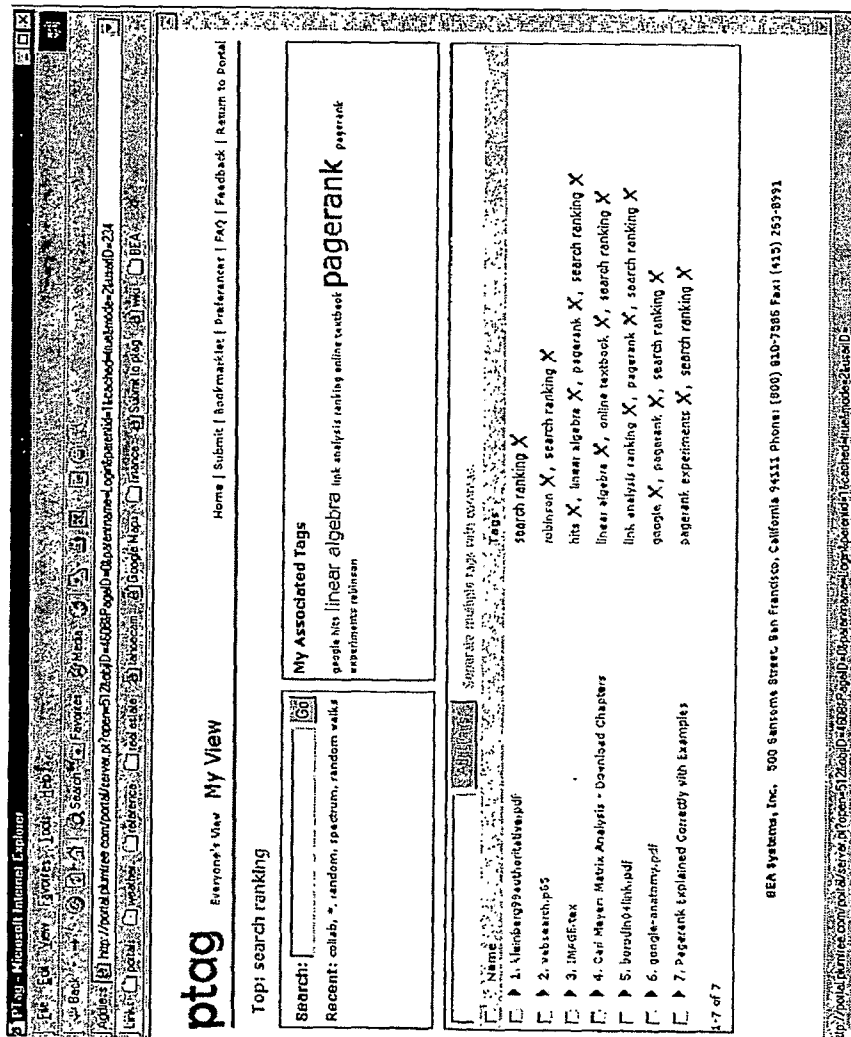


FIG. 1B

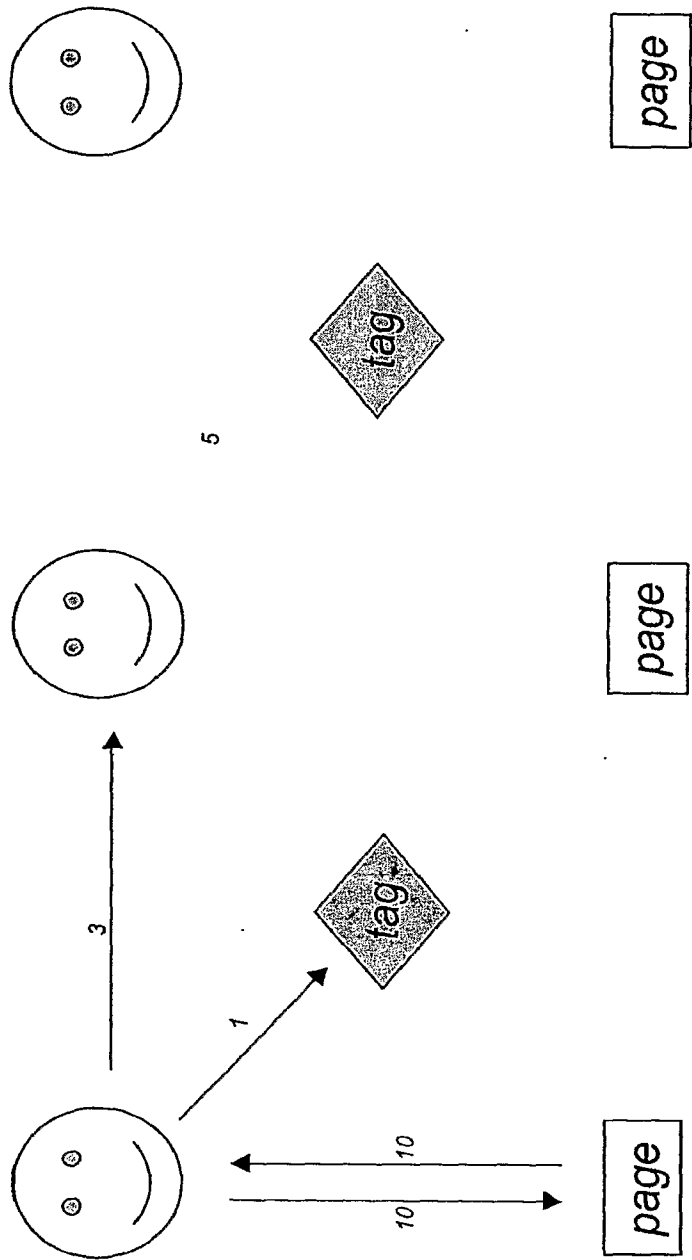


FIG. 2A

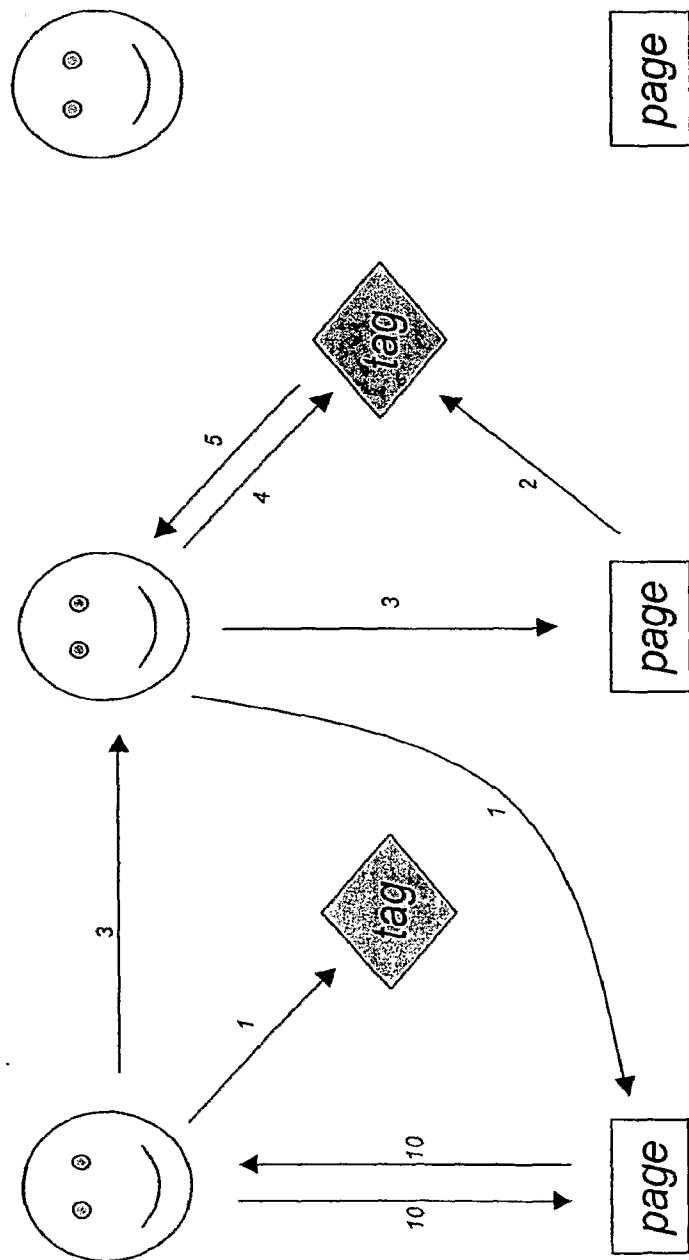


FIG. 2B

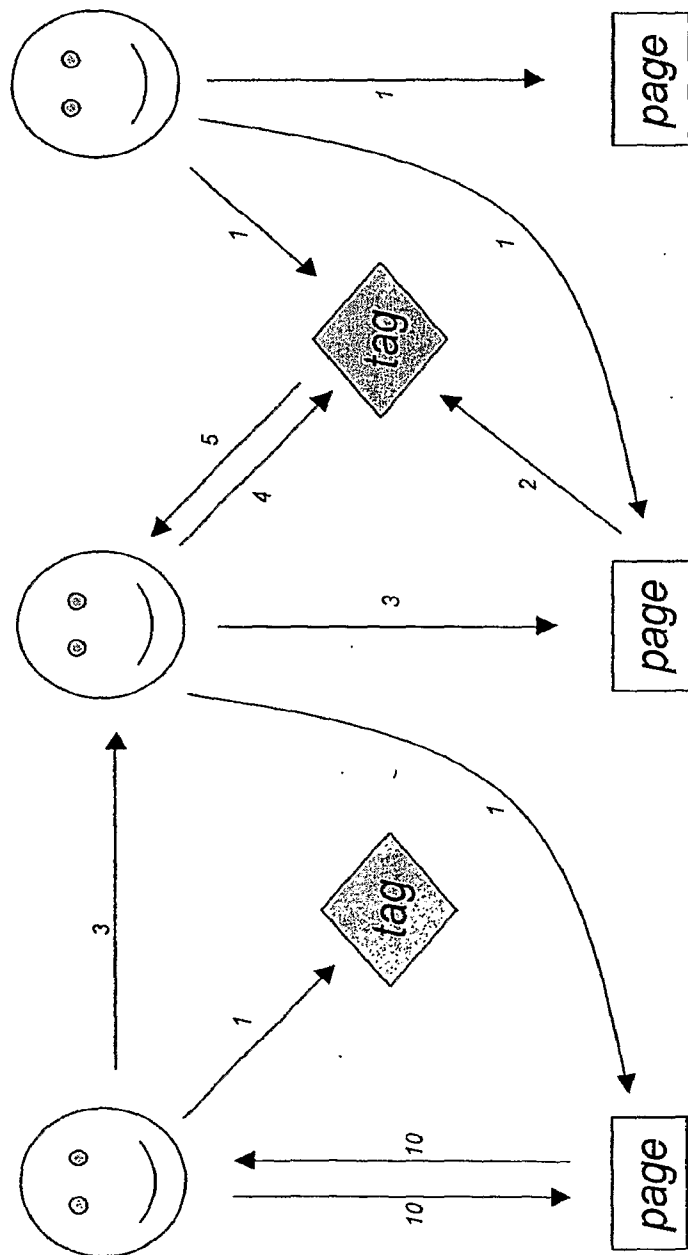


FIG. 2C

$$\begin{array}{c}
 \begin{array}{c} 302 \\ \downarrow \end{array} \\
 \left(\begin{array}{cccccccc}
 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\
 3/4 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\
 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
 1/4 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
 0 & 1/2 & 1/3 & 0 & 0 & 0 & 1 & 0 \\
 \begin{array}{c} 304 \\ \rightarrow \end{array} & 5/7 & 1/8 & 0 & 0 & 0 & 0 & 0 \\
 0 & 3/8 & 1/3 & 0 & 0 & 0 & 0 & 0 \\
 0 & 0 & 1/3 & 0 & 0 & 0 & 0 & 0
 \end{array} \right) \vec{X} = \vec{X}
 \end{array}$$

FIG. 3

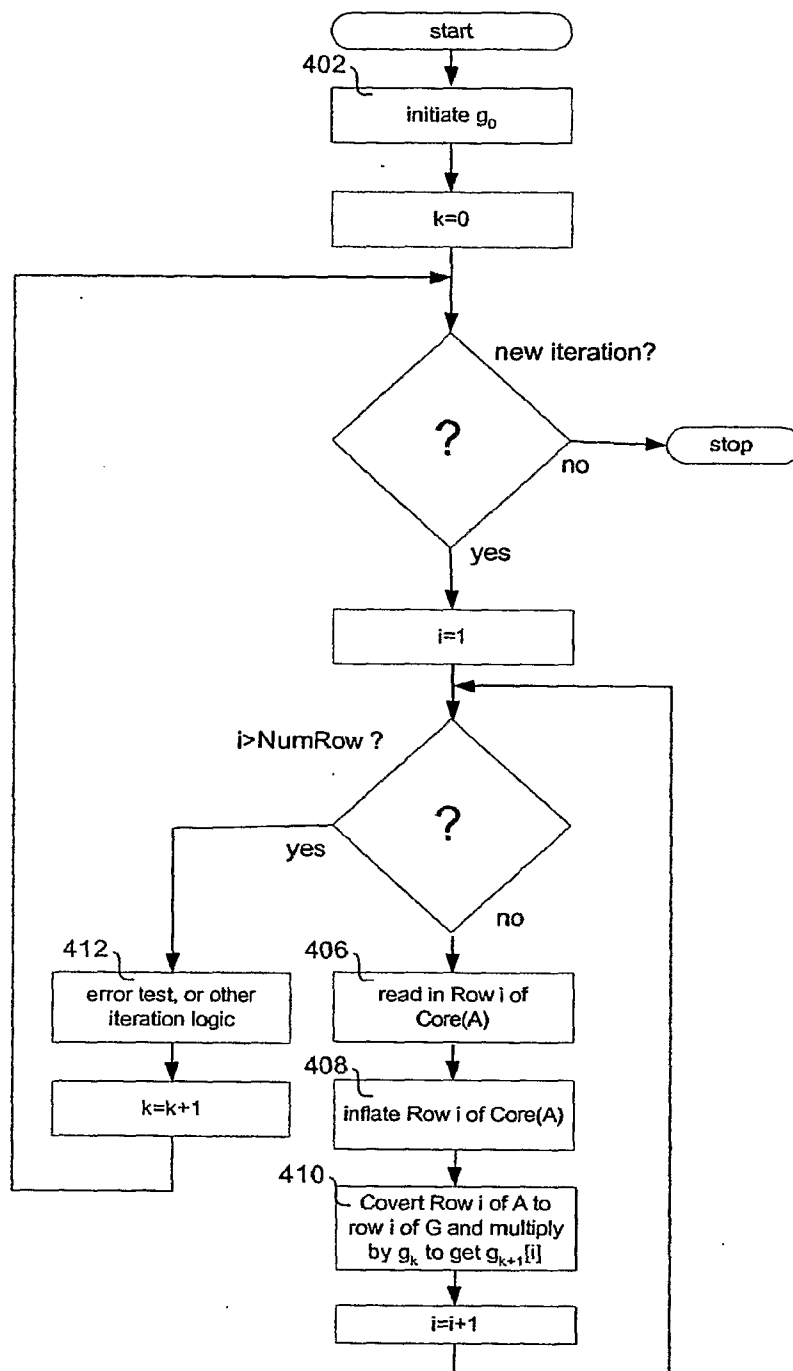


FIG. 4

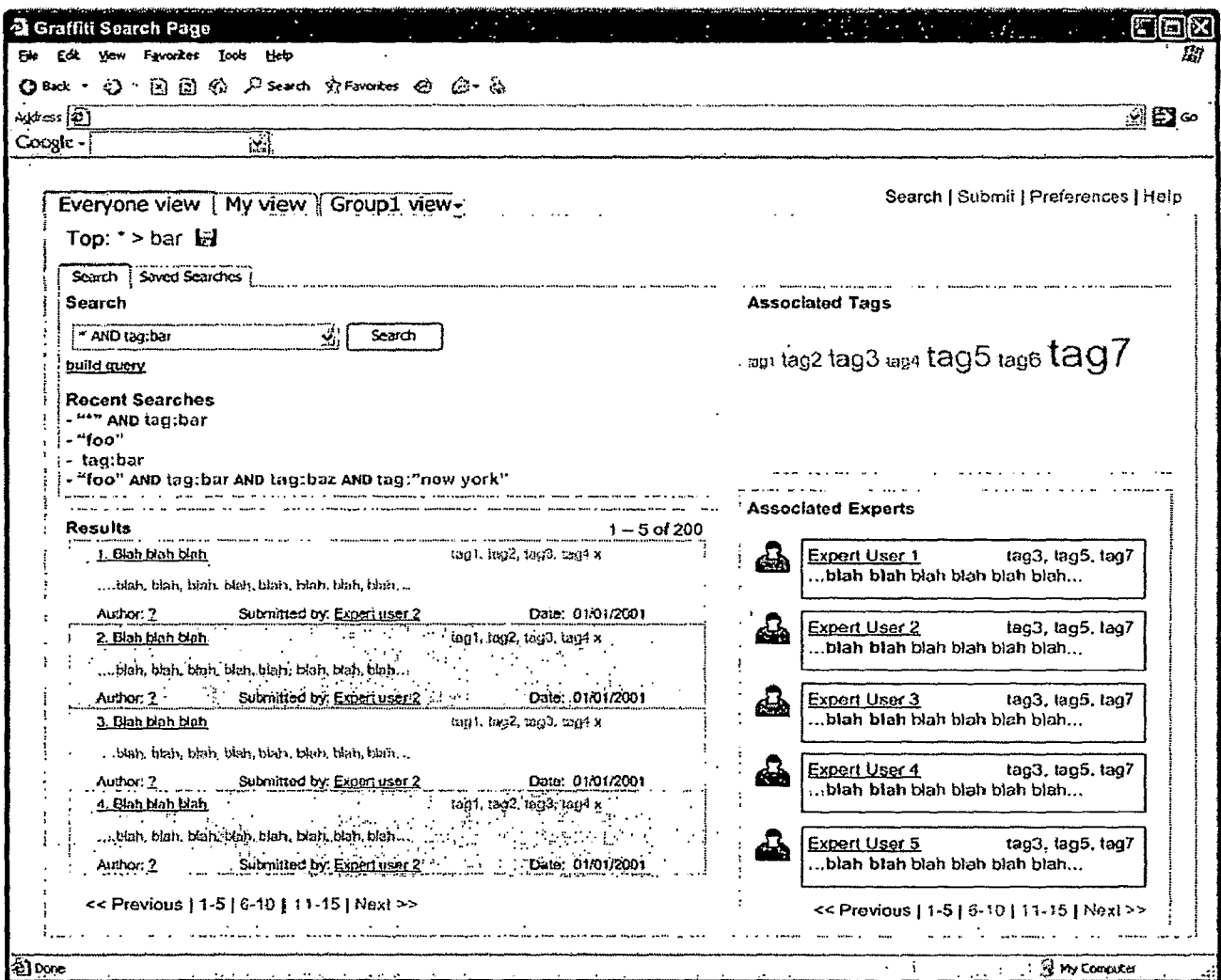


FIG. 5

Rank Administration

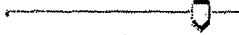
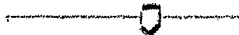
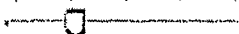
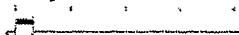
Graffiti Rank

Each object in the system is assigned a Graffiti Rank that does not depend on any search query. This page lets you control the job that calculates the rank, how the rank is incorporated into search results and the factors that make up the rank.

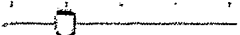
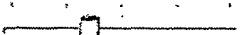
Rank Factors

The following factors determine how rank is calculated for each object. Adjust the sliders to change the importance of each factor.

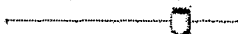
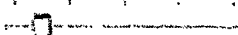
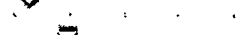
Pages

The rank of the author		0.87
The rank of the submitter (if author is unknown)		0.60
The rank and number of people who tag		0.22
The rank and number of people who view		0.09
The rank of other pages in the same container		0.03

Tags

The rank and number of people who apply		0.40
The rank and number of people who use		0.25
The rank and number of objects to which the tag applies		0.32

Users

The rank and number of pages authored		0.95
The rank and number of pages submitted		0.65
The rank and number of people who have this user as a contact		0.70
The rank and number of people who tag		0.11
The rank and number of people who view		0.03
The rank and number of tags created		0.25

Half Life

An object that receives many tags and views will rise in the search results and will therefore be more likely to receive tags and views. One way to dampen this positive feedback loop is to make tags and views less important over time. Enter the number of days after which the importance of each tag and view event is reduced by half. The shorter the half life, the faster each object will forget about being tagged or viewed. A value of 0 indicates that the half life is infinite.

Tagging days 

Viewing days 

FIG. 6A

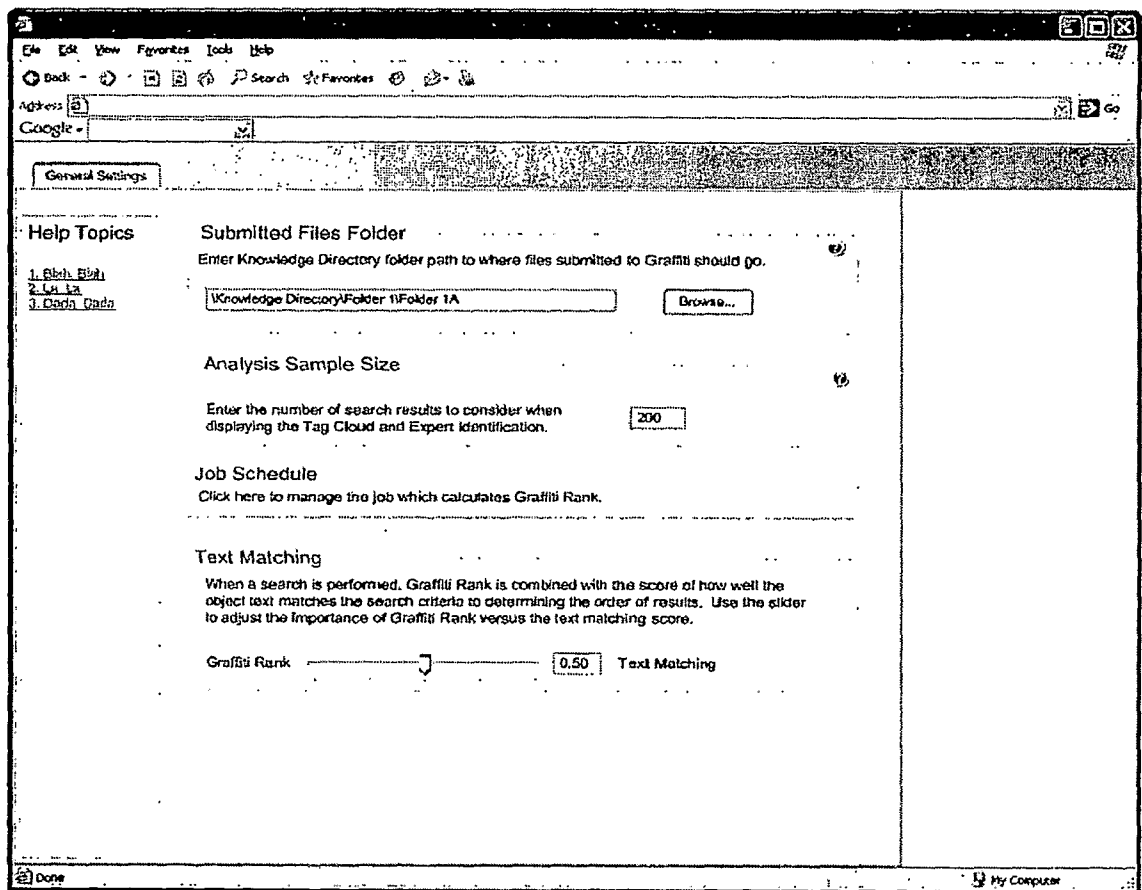


FIG. 6B

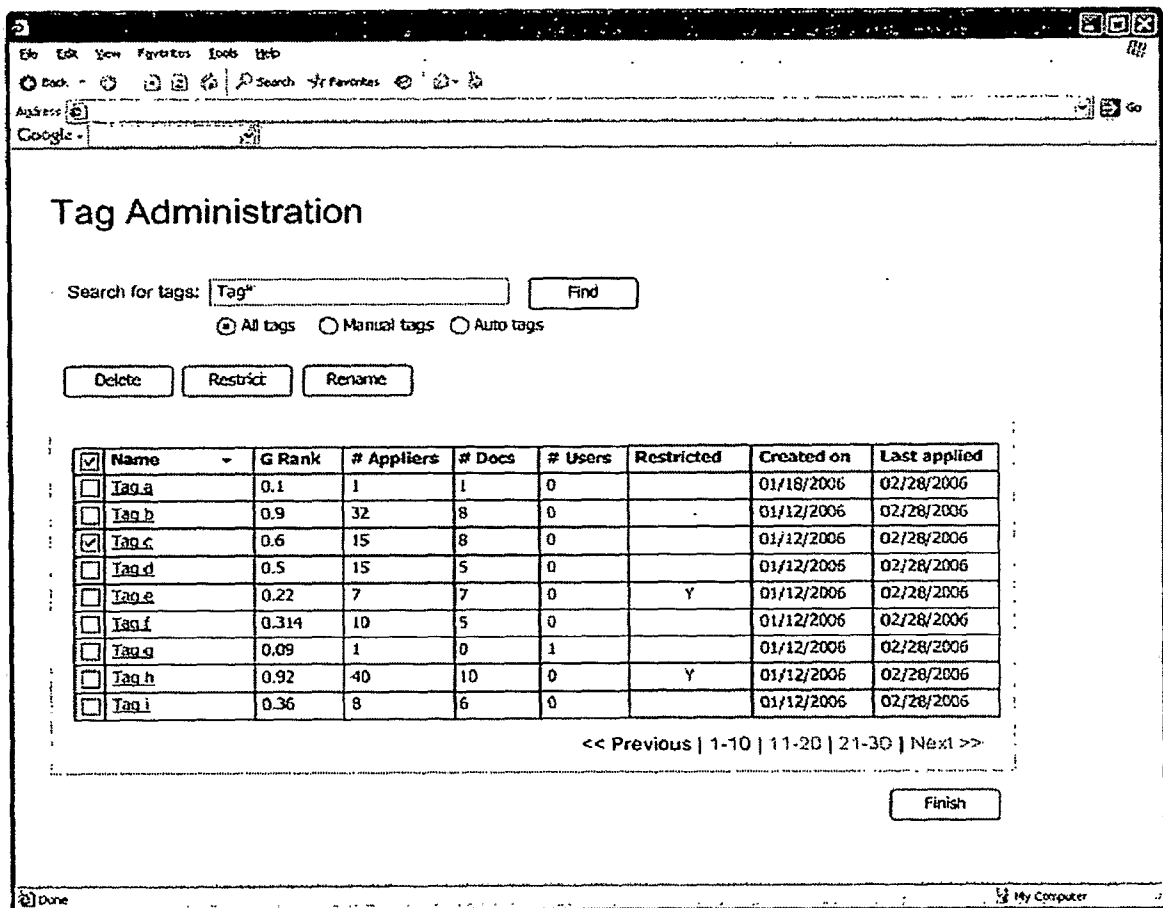
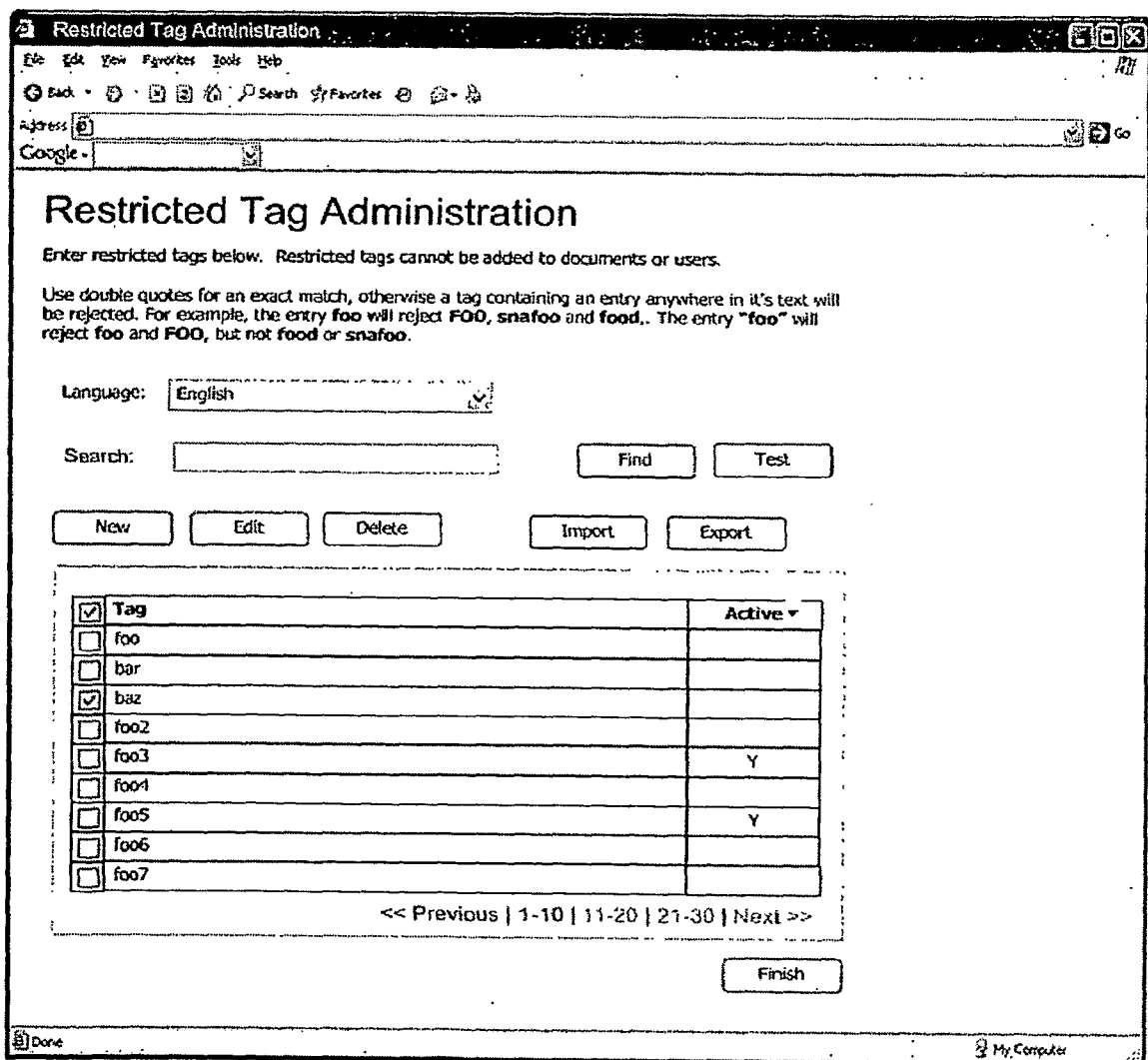


FIG. 7A



Restricted Tag Administration

File Edit View Favorites Tools Web

Back Forward Stop Search Favorites Go

Address Google

Restricted Tag Administration

Enter restricted tags below. Restricted tags cannot be added to documents or users.

Use double quotes for an exact match, otherwise a tag containing an entry anywhere in it's text will be rejected. For example, the entry `foo` will reject `FOO`, `snafoo` and `food`.. The entry `"foo"` will reject `foo` and `FOO`, but not `food` or `snafoo`.

Language: English

Search: Find Test

New Edit Delete Import Export

☒ Tag	Active
☐ foo	
☐ bar	
☒ baz	
☐ foo2	
☐ foo3	Y
☐ foo4	
☐ foo5	Y
☐ foo6	
☐ foo7	

<< Previous | 1-10 | 11-20 | 21-30 | Next >>

Finish

Done My Computer

FIG. 7B