Middle-East Journal of Scientific Research 13 (6): 785-792, 2013
ISSN 1990-9233

DOI: 10.5829/idosi.mejsr.2013.13.6.2409

Towards Effective Information Retrieval Using Hierarchical Information Visualization

[1]Ali Sajedi Badashian, [1]Mehdi Najafpour,
[2]Mehregan Mahdavi and [3]Zahra Zebardast

[1]Department of Computer Engineering, Islamic Azad University, Lahijan Branch, Iran
[2]Department of Computer Science and Engineering, University of Guilan, Iran
[3]Department of Management, Islamic Azad University, Rudsar-Amlash Branch, Iran

Abstract: Nowadays, the personal needed information by users are spread everywhere, from personal computers to cell phones, from PDAs to Web pages on the Internet and so on. Although taking advantage of retrieval techniques may facilitate fast access to the files and folders in all these platforms, searching the data collections is sometimes a time consuming process; in other words, the verification of search results by the user is a tedious task that should be done one by one. This process even takes more time for users with trivial background about the final results (especially in Web Information Retrieval) or users that cannot recall appropriate search keywords. This is due to lack of end-users' knowledge (i.e., Known keywords) about the target. This paper introduces File Tree Search (FTS), a multi-stage search tool to enhance Information Retrieval (IR) by hierarchical visualization of the results and user assistance in learning/remembering the topics. The proposed method works not only for file search, but also for any classification based IR. The experimental results indicate that this method is fast, dynamic and friendly in nature as an IR system for different taxonomies.

Key words: Information Retrieval · Information Visualization · Tree Structure · Search · File System

INTRODUCTION

Nowadays, users deal with huge amounts of data in different forms such as files and folders, web pages and software tools in embedded devices. Managing these data needs structures and tools specialized for different categories.

Lots of tools are devised for searching information. However, the user should have some background information beforehand (i.e., keywords) in order to find the target. It is obvious that the more knowledge the user has, the faster access to the results would be resulted. The search process would be a tedious task when a user lacks the knowledge about the target. In other words, the common search methods may fail in some cases where the user does not know much about the final objectives of his/her search. This paper proposes a method to help users for finding the required information more effectively.

This paper is organized as follows: In the next sections, the characteristics of the system are considered after a short review on related literature. Then, the proposed method is introduced. The simulation system is described next. The experimental results are also presented. Finally, some theoretical and statistical issues on using the FTS approach are concluded.

Related Work: Files and folders are handled by a piece of software normally called File Manager or File System. Visualization is an important issue in such systems [1-4]. Searching in files and folders for finding the required information is a time consuming process. Although new storage and retrieval systems such as indexing techniques [5-7] reduce the access time, the problem still persists on the user side. In other words, due to the large number of files and folders that are normally ordered conceptually (or even unordered as a mass of stored files), the user deals with numerous results for each search. The search results are sometimes too many. As a result, the user might prefer to ignore the search results and instead pursuit several possible hierarchies for the desired content. The problem is intensified when the user does not know the location and complete name of the file(s).

Corresponding Author: Ali Sajedi Badashian, Department of Computer Engineering, Islamic Azad University, Lahijan Branch, Iran. Tel: +981412237598, Fax: +981412237598.

Providing context-awareness for virtual file systems is discussed in [8]. It proposes a backward-compatible, context-aware virtual file system. The system makes it possible to find a file without knowing its full path, but only by remembering the context. It is also possible to browse hierarchically with legacy path name as it is fully backward compatible.

In [9], a multi-dimensional search method is offered for personal information management systems. Current tools only consider structure such as file and directory and metadata such as date and file type, as filtering conditions. A multi-dimensional approach to semi-structured data search was proposed in this research by the use of fuzzy structure and metadata conditions in addition to keywords. This technique provides a complex query interface and considers three query dimensions; content, structure and metadata. It offers more comprehensive search capabilities than the content search. They assign each dimension individually and the dimensions are integrated together to make a unified score.

A group of authors from Microsoft Research, proposed a system for personal information retrieval and re-use [10]. A system called Stuff I've Seen (SIS) is considered in this research. It facilitates information re-use. This is performed by providing a unified indexing of information a person has seen. By using SIS, information can be found more easily.

Users usually access their files by using folder navigation. There is another research that deals with improving search and navigation preferences in Personal Information Management (PIM) systems [11]. It evaluated the recent improvements in desktop search and its effect on PIM Systems. It used two systems for its evaluation, i.e., Microsoft Windows and Mac, using questionnaires. In Microsoft Windows, Google Desktop and Windows XP Search Companion have been studied and in Mac the comparison was carried out between Mac Spotlight and Sherlock.

Nowadays, mobile phones and devices have become very common; as a result, searching and finding desirable data in such devices is important. An approach is offered in [12] for searching personal information stored in mobile devices.

The KDE environment in Linux provides a tree-based filtering mechanism [13]. When the user opens the menu, it can be filtered based on an expression typed in by the user. All menu items that do not contain that expression in their sub menus will be disabled so that the user will know where to look for he desired items.

The Functionality Tree structure [14] is another useful structure to show large amounts of data in categorized, hierarchical order. The structure was proposed to include all activities of the system. The structure is a comprehensive tree structure prepared for a given subject (e.g., the contents of a web site or an application's functionalities). The Functionality Tree structure can be configured not only for the included capabilities of the software or device, but also for folders' hierarchy and the files inside them. The second approach is used in the current research. In other words, a Tree Search approach is used for the hierarchy of files and folders as the Functionality Tree. A simulation program has been developed in order to investigate the characteristics of the approach.

Exploring the Problem: As mentioned in the previous section, an important issue in retrieving data by the user is related to the users' level of knowledge about the search items and fields. For example, when purchasing a new model of a cell phone, one may not find some of its capabilities. However, everyone wants to be able to use all its capabilities. Moreover, it is desirable to be able to find such capabilities and learn how to use them in a short period of time.

For a more detailed example, we can consider a device with a wide variety of (new) capabilities. Using the Functionality Tree structure [14] presented here, different users might become familiar with these capabilities, especially the previously unknown capabilities. All the nodes of such a tree represent the information to be offered to the users or their categorization as taxonomy [15, 16]. The large size of this tree indicates both the accuracy of information (as a positive aspect) and the difficulty in its traversal (as a negative aspect). Although, complete information is provided, but the end user may be confused when finding an item by traversing this tree, especially when there are several paths from root to the leaves. The same issue may exist in "Help" section of common software programs. In such cases, the search capability can substitute the tree structure for having better view about the desired subject.

Considering the file search example, in a common hierarchical tree view of a hard drive on a PC, there are many files and folders. To estimate the number of nodes in such a tree, 10 Hard Drives and 4 memory cards have been examined thoroughly as an initial process. These systems were owned by ten computer engineering students. Eight of them were using MS Windows XP and two of them MS Windows Vista. For each system, a drive

Table 1: Average statistics of examined media.

Media	Capacity (GB)	Occupied Size (GB)	Number of Files	Number of Folders
HDD (10 cases)	168	113	113846	11687
Memory Card (4 cases)	4	3.29	2483	361

that excluded system files and folders was chosen to be evaluated. The survey showed that on average, there are more than 10,000 folders and 100,000 files in a 160 GB Hard Disk. The more detailed statistics are presented in Table 1. As a result, the number of nodes is very high, so that it cannot be easily traversed manually.

In this huge amount of data, current search methods might fail to offer the correct path of the desired file/folder. There are several reasons, including large number of suitable results or lack of users' knowledge about the file name/location.

There are lots of node-link diagrams and browsers for searching files, such as Cone Tree [17], Hyperbolic Browser [18], Space Tree [19] and tree juxtaposer [16], in which the visualization is more important. A number of them integrate browsing and searching together such as Lifelines [20], Space Tree [19], TaxonTree [21]. However, they cannot provide a complete and capable search for users. For example, Space Tree [16] is a tree browser which lets users search in big and deep trees by dynamic zooming and laying out the best suitable branches for the node links. By clicking on a node, the tree focuses on the node. Also TaxonTree is a tree like Space Tree with a little difference in visualization of the tree.

In TS, there is a mechanism in which the user is able to eliminate the unnecessary nodes in each step of searching process and the results will be ordered. However, this is not the only available functionality. If required, the user can search in each level of the results over and over again until the final file is reached. Therefore, through this structure, the user can get more information in addition to finding the file (step-by-step or at once).

Manual browsing is also available during search, but it is usually difficult to find the file by only browsing and traversing. Therefore, the TS has the search capability integrated with browsing and traversal capabilities for helping users in terms of decreasing the difficulties. Sometimes, users need to search the information in the whole tree or only in some sub-trees. This is available in the TS in different ways as are described in the next section.

FTS Approach: As mentioned in the previous section, two problems might occur during a search session, finding either too many results or no results at all. In the first case, finding the desired item is tedious. However, the second problem is more important: this is because the user may remember only a part of the file name, its path, some clues with logical expressions (i.e. and/OR), or even some sentences from the contents of the file. This partial information usually fails in current file search applications. This is because current search tools focus on the final result, but these clues to the result could usually be applied during the search process that will eventually lead us to the final result.

The offered solution, FTS, uses this scattered information through the hierarchy of folders. It adds the search capability to the tree visualization of data. In fact, the tree view items (i.e., both files and folders) participate in the search. In other words, it integrates the search capability with the conceptual structure of the tree view visualization to overcome the lack of knowledge of the user about the respected items.

Of course, some systems try to make the searches easier and more conceptual such as Thematic Mapping [15] in which important concepts are collected in unstructured documents, then the concepts are arranged in a hierarchical tree from general to specific by meaningful labels. This could be taxonomy for the corpuses to help the users find the files in a conceptual tree. However, it is better for the users to see the results in original corpuses at the first step in which the results are related to the topic. In the FTS approach, the users would search just in these corpuses for each step. The users would continue these levels step by step toward getting the final file. During these steps they would get more information about the requested file because of such structure. The overall actions of TS are shown in Figure 1. The user may eliminate the tree nodes, search the tree using appropriate keywords, collapse some nodes in order to be more compact and friendlier and browse one or more paths. In this way, the user may be inspired towards more appropriate keywords that he/she did not know initially (or probably did not notice them). The process may be repeated several times. Each time the user may use the new inspired keywords to search in the eliminated tree view or the original one.

In a nutshell, the process is defined in two phases; first, making the information hierarchy. This is previously studied in [15, 21-23]. It can be formed by experts or

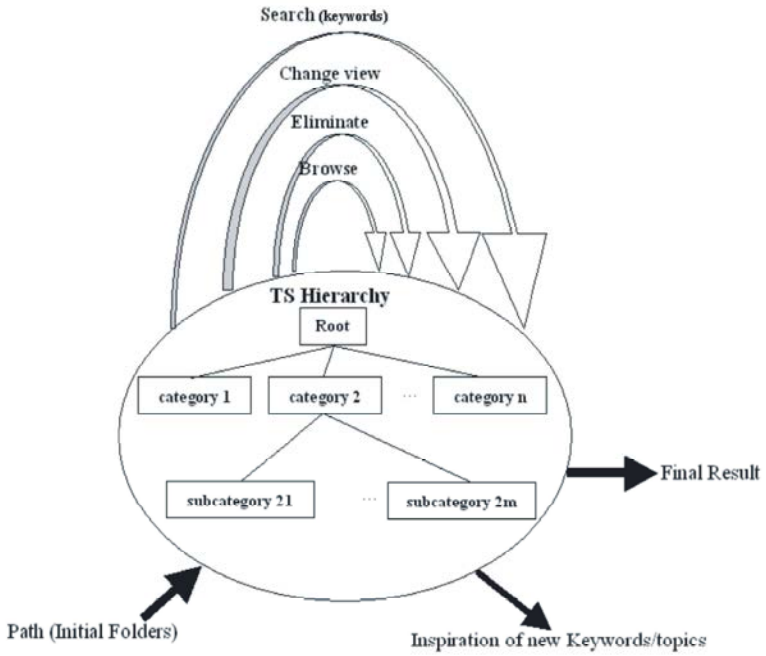

Fig. 1: An overview of TS and its different operations

automatically. Then the step-by-step search in both the hierarchies or final nodes and their elimination is applied to retrieve the desired documents.

This structure works even when the user is not familiar with the file/folder's name and location. The items can be found by their logical relationship to the nodes of the tree from the top level to the more detailed nodes.

Search Process: The user is assumed to have some information about the file name, location, type or concept. First, some initial searchable items are determined and the first-step of search is applied on the large amount of data. The results are visualized in their original places in the tree view. Only the extra nodes (i.e., the nodes containing no desired items) are eliminated. The experimental results indicate that usually most of the nodes are removed in the first step. After that, the user can view the results (in the "Results" tree view) and eliminate some nodes based on their concept or probability of containing the desired items. This can be done in each level of the tree view. Note that eliminating a first level node in the hierarchical structure may eliminate numerous redundant irrelevant results. Then the search process can be continued on the remaining nodes or even the selected nodes within them. Finally, the results can be found after several searches in results and path selection. This way, the user can concentrate on the more related items based on both logical and textual relations.

FTS Software: In order to test and analyze the suggestions and evaluate the impact of the mentioned structure on the search process, we have implemented the FTS method. The implemented software provides a number of functionalities (as are mentioned later in this section) to facilitate the search process. After running the program, the original window of the program will appear ("Tree Search" window as in Figure 2).

In the "Path" section, the program receives the user's path for searching. The path can be typed manually or can be copied and pasted into this section. Furthermore, the program provides a wizard for the users to select any path by pressing the "Browse" button and entering the "Browse for folder" window. Notice that, if the users do not select the path and the path section becomes empty, the default path will be the whole "My Computer" including all hard drives, CD-DVD ROMs, Cool Disks, etc.

In the next step, after selecting the path, the user can press the "Refresh" button. After a short time, the whole path will be loaded in the tree view (The "Synchronize" button acts similarly by synchronizing the tree view with the hard disk or the other specified path). From this point on, the search process is performed in this tree view (instead of hard drives) due to speed considerations. However, the time of loading depends on the number of files and folders which are in the selected path.

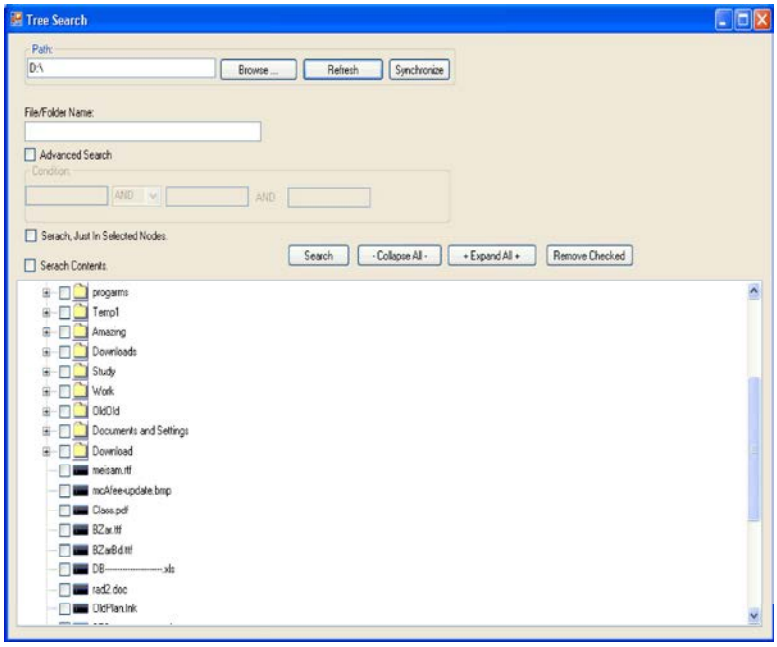

Fig. 2: A simple view of the program

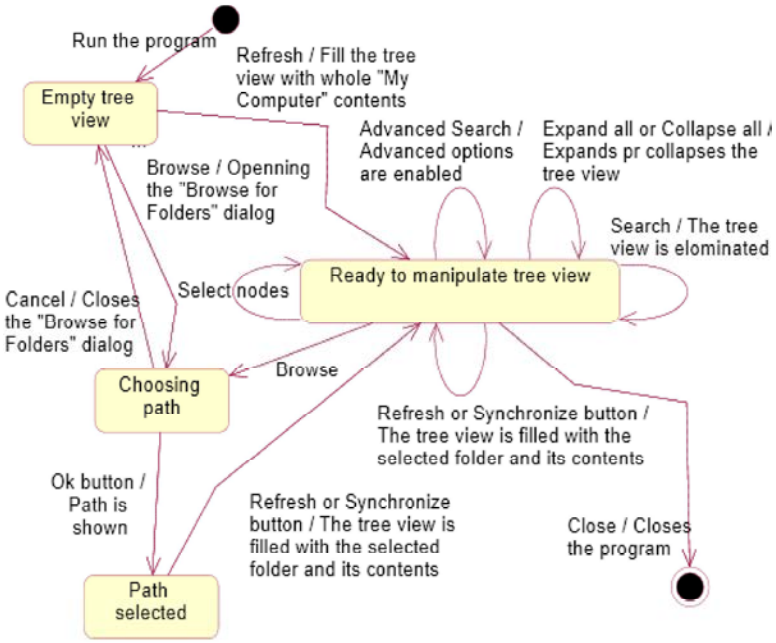

Fig. 3: FTS State Chart diagram

As mentioned before, the nodes of this tree are files and folders of the selected path. Also, besides each file/folder, a check box is provided to determine the folders in which the search will be performed. After making the tree, two options are also available for the users in the form of check box. The first check box will help the users to specify whether the search should be done in all the existing nodes of the tree or just in the selected nodes. The second check box is to specify where to search; just in the name of the files and folders or in their contents as well. These options are unchecked by default.

The keywords have to be typed before clicking on "Search" button or just pressing Enter. Advanced search is also available to provide advanced conditional statements (e.g., "AND" or "OR" operators).

There are some other buttons in this software that are described as follows: Because, after doing the search, the results are collapsed as nodes of the tree, there is a button named "Expand All" which expands the whole tree. The "Collapse All" button acts the opposite. It means that all the expanded nodes will be collapsed. If the user wants to eliminate some of the specific nodes manually, it is enough to tick the nodes in the tree manually and then press the "Remove Checked" button by which all the specific nodes will be eliminated from the tree.

Further search is available within the results. They can be selected for searching or being eliminated from the results by the dedicated check box for each item and the specified options (i.e., "Search just in selected nodes" check box and "Remove checked" button).

The state diagram of FTS and its main operations are shown in Figure 3. The starting state is when the user runs the program. This is the status shown in Figure 2. By choosing a path and passing from two temporary steps, the user enters the main state in which the most TS operations are available. All the operations mentioned in Figure 1 are useful in this state. In this step, using simple or advanced search, the tree is eliminated. Moreover, the user can eliminate the tree manually. In other words, as he/she proceeds in the hierarchy and the search steps, he/she may select some nodes that are considered useless and drop them from the FTS. The operations of this step are usually repeated several times. Finally, there are two possible situations for the last state; when the results are found or when the user is sure that there is no appropriate answer in the available nodes.

The program was developed in C#.Net. It is available online from www.sajedi.ir/projects/FTSSimulation. The program can be linked to Windows Explorer if required (e.g., "open containing folder" -in Windows Explorer-, "Copy", "Cut" and "Paste" commands. These are available through right clicking on a file/folder).

Experimental Results: In order to test and evaluate the program, six computer engineering students were selected. After they spent a few days learning and practicing the software, they were asked for their feelings and thoughts towards TS. Different aspects of the software were questioned and they gave a score between 0 and 100 to each question. These aspects were as follows:

Being Easy to Learn and Use: They answered questions regarding transparency of the system, UI, search process and clarity of the system. TS was scored 75 out of 100 in this part, meaning that it is friendly and easy to use.

Efficiency in Surfing Information: They believed that TS can enable surfing the information efficiently and gave a score of 80 to this category of questions.

Providing Comprehensive, Top-Down View over Files and Folders: This can be considered as the main strength point of TS and was scored 84. They believed that they can focus on their keywords step-by-step using TS.

Search Speed: The search speed was scored 76. Although it does not seem very good, but the search process can be further enhanced.

Preferring Ts Instead of Other Search Tools: This was scored 84 out of 100 indicating their preference towards TS against other search tools.

In the last section they were asked to express pros and cons of TS. A number of suggestions were made about providing "Tooltips" in the next versions of the software. The speed of the program was suggested to be enhanced in refresh section. However, the "Synchronize" section compensates this latency to some extent. Some facilities were also suggested for the novice users. Some of the candidates also had suggestions about adding new capabilities to the "Advanced Search" section of the software.

Another suggestion was to add an extra capability to save/load paths and hierarchies. For the paths it can be easily adapted, but the hierarchies need more comprehensive data structures and efforts. Adding some navigation buttons such as “back” and “forward” is also another useful suggestion that can make the browsing more convenient.

Conclusion and Future Works: Considering the problems of information retrieval and information management, a visualization approach has been offered for combining search and navigation. In this approach, the concentration is both on the overall position in the hierarchy and the known keywords. The overall position in the hierarchy helps to eliminate additional branches by their concept, but the known keywords help to reach the destination by file/folder characteristics (i.e., name, suffix, contents and so on).

The approach is useful when the user does not remember the file name and path (or even parts of them). It allows the user to overcome the lack of knowledge about the destination file/folder using a step-by-step search and revise it inside the hierarchy. Nodes of the tree can be eliminated manually along with filtering by file type, name or contents.

Because of the hierarchical manner of the program associated with the search capability, different users with different levels of knowledge are supported. This was verified during the experiments of this research.

The FTS software was developed to confirm the claims and identify the drawbacks. Users' considerations after using FTS software confirms that using this approach makes finding files and folders easier in almost all cases, even in complicated cases (e.g., when the file name, location and contents are all undetermined). The step-by-step search (i.e., search in results) capability and navigation, branching and bounding the tree between the steps improves both reaching the target file and remembering the destination path.

This approach works not only for file search, but also for other types of information visualization and retrieval. It helps to manage information retrieval by a natural visualization of the information in their original hierarchy. As a result, the approach is capable of including information of Web portals (e.g., Yahoo). More generally, the search methods can also be customized to be used in web-based applications. Fortunately, the dynamic and object oriented design of the program expedites its usage in web-based programs.

Currently, the results of the search engines are shown in order (by the recognized relevancy to the given keywords). Showing these results in the categorized forms helps the users easily eliminate unrelated items and concentrate on the most important items.

The FTS approach can be represented and implemented for a variety of systems such as help systems, site map of web sites or generally taxonomies. Also enhancements in the FTS may be considered for every use. For example, "Favorites" list helps gaining rapid access to folders. Finally, some additional functionality such as zooming capabilities (as offered in [10]) and saving/loading the tree structure can also be useful as were suggested by some of the participants in our experiments.

REFERENCES

1. Joes, W., H. Bruce and A. Foxley, 2006. Project contexts to situate personal information. In: 29th annual international ACM SIGIR conference on Research and development in information retrieval, pp: 729-729. Seattle, Washington.
2. De Chiara, R., U. Erra and V. Scarano, 2003. VENNFS: A Venn-Diagram File Manager. In: Seventh International Conference on Information Visualization, pp: 120. Washington DC.
3. File Managers and Explorers: http:// en.softonic.com/ palm/ file-managers-explorers
4. Comparison of file managers: http:// en.wikipedia.org/ wiki/ Comparison_of_file_managers
5. Fagan, J., 1987. Automatic phrase indexing for document retrieval. In: the 10th annual international ACM SIGIR conference on Research and development in information retrieval, pp: 91-101. New Orleans, Louisiana.
6. He, X., D. Cai, H. Liu and W.Y. Ma, 2004. Locality preserving indexing for document representation. In: the 27th annual international ACM SIGIR conference on Research and development in information retrieval, pp: 96-103. Sheffield.
7. Poshyvanyl, D., M. Petrenko, A. Marcus, X. Xie and D. Liu, 2006. Source Code Exploration with Google. In: 22nd IEEE International Conference on Software Maintenance, pp: 334-338., Washington DC.
8. Song, Y. and D. Park, 2007. Providing context-awareness to virtual file system. In: Symposium on Applied Computing, pp: 1199-1200. Seoul, Korea.
9. Peery, C., W. Wang, A. Marian and T.D. Ngugen, 2008. Multi-dimensional search for personal information management systems. In: 11th international conference on Extending database technology: Advances in Database Technology, pp: 464-475. Nantes, France.
10. Dumais, S., E. Cutrell, J.J. Cadiz, G. Jancke, R. Sarin and D.C. Robins, 2003. Stuff I've seen: a system for personal information retrieval and re-use. In: 26th annual international ACM SIGIR conference on Research and development in information retrieval, pp: 72-79. Toronto, Canada.
11. Bergman, O., R. Beyth-Marom, R. Nachmias, Gradovitch, N., Whittacker, S., 2008. Improved search engines and navigation preference in personal information management, ACM Transactions on Information Systems (TOIS), 26(4): 1-24.
12. Akbas, M. and G. Singh, 2007. Personal information search on mobile devices. In: 4th international conference on mobile technology, applications and systems and the 1st international symposium on Computer human interaction in Mobile Technology, pp: 724-728. Singapore.
13. Desktop, K., XXXX. Environment, http:// www.kde.org
14. Sajedi, A., H. Afzali and M. Mahdavi, 2008. Improving learnability and usability of software applications. In: Proceedings of IADIS IHCI 2008, International Conference on Interfaces and Human Computer Interaction, Amsterdam, Netherlands.

15. Chang, C.Y., R. Lieu, J. Liu, A. Luk, J. Mao and P. Raghavan, 2002. Thematic mapping-from unstructured documents to taxonomies. In: eleventh international conference on Information and knowledge management, pp: 608-610. Virginia, USA.
16. Lee, B., Bederson, B.B., Campbell, D., Sims Par, C., 2004. How users interact with Biodiversity information Using TaxonTree. In: the working conference on Advanced visual interfaces (AVI), pp: 320-327. Gallipoli, Italy.
17. Robertson, G.G., J.D. Mackinlay and S.K. Card, 1991. Cone Trees: Animated 3D Visualizations of Hierarchical Information. In: Proceedings of Human Factors in Computing Systems (CHI 91), pp: 189-194. New Orleans, Louisiana.
18. Pirolli, P., S.T. Card and M.M. Van Der Wege, 2003. The effects of information scent on visual search in the hyperbolic tree browser. In: ACM Transactions on Computer-Human Interaction (TOCHI), 10(1): 20-53. ACM New York, NY.
19. Plaisant, C., J. Grosjean and B. Bederson, 2002. SpaceTree: Supporting Exploration in Large Node Link Tree, Design Evolution and Empirical Evaluation. In: IEEE Symposium on Information Visualization (InfoVis'02), pp: 57. Washington DC, USA.
20. Milash, B., C. Plaisant and A. Rose, 1996. LifeLines: visualizing personal histories. In: Conference companion on Human factors in computing systems: common ground, pp: 392-393. Vancouver, British Columbia.
21. Li, T. and S.S. Anand, 2008. Labeling Nodes of Automatically Generated Taxonomy for Multi-type Relational Datasets. In: the 10th international conference on Data Warehousing and Knowledge Discovery, pp: 317-326. Turin, Italy.
22. Gates, S.C., W. Teiken and K.F. Cheng, 2005. Taxonomies by the numbers: building high-performance taxonomies. In: 14th ACM international conference on Information and knowledge management, pp: 568-577. Bremen, Germany.
23. Song, R., Z. Luo, J.R. Wen, Y. Yu and H.W. Hon, 2007. Identifying ambiguous queries in web search. In: the 16th international conference on World Wide Web, pp: 1169-1170. Alberta, Canada.