Can You Retrieve a File on the Computer in your First Attempt? Think to a New File Manager for Multiple Categorization of Your Personal Information

Ali Sajedi
Software Engineering Department, Azad University – Lahijan Branch, Iran
sajedi@Liau.ac.ir

Seyyed Hamidreza Afzali
School of ICT, Royal Institute of Technology (KTH), Sweden
hr.afzali@gmail.com

Zahra Zabardast
Department of Management, Islamic Azad university, roudsar and Amlash Branch, Roudsar, Iran
z.zebardast@yahoo.com

ABSTRACT

Recent advances in the two past decades in storage technology permitted the users gather more and more information. This caused an information explosion in different areas from personal computers and mobile cell phones to web sites and social networks. In the personal information management (PIM) studies, management of information was enhanced by utilizing hierarchical structures, search and tagging methods. In the case of files and folders, however, the file manager software is used the most. In almost all the file managers, the foundation of file storage/retrieval is the hierarchical file system. A file may fit to several paths but should be stored / accessed via only one path. Making the choice the user may face ambiguity in storage time. Hence redundancy may occur during the passage of time. In other words, different versions of a file may be stored undesirably. On the other hand, in retrieval of the desired file, again the ambiguity occurs in choosing a file (and path) between various candidates. The result is increasing access time and user frustration. This is because the infrastructure of file storage/retrieval is the same (as the previous decades), but the data and information are massively increased.

In this article, a new file manager, File & Concept Browser (FCB) is proposed that supports multiple categorizations. The general attitude in FCB is similar to the common hierarchical file managers, but it supports maintaining a file through different paths without multi-versioning, redundancy and ambiguity. The idea was implemented as a software prototype and experimental results show that in addition to avoiding redundancy, using FCB can reduce access time, retrieval failure and user frustration.

Categories and Subject Descriptors

D.3.3 [Operating Systems]: File Systems Management – directory structures, access methods.

D.3.3 [Information Storage and Retrieval]: Information Search and Retrieval – retrieval models, selection process, search process.

H.5.2 [Information Interfaces and Presentation]: User Interfaces – interaction styles.

General Terms

Design, Human factors.

Keywords

Directory Structure, File System, Information Retrieval, Classification, User Interface.

1. INTRODUCTION

Vast amounts of information with different usages are presented today in hierarchical structures such as taxonomies [4], site maps in the websites [6], help systems [7] file managers [5] and even email management systems [2]. Due to large amount of presented data in the mentioned structures, users are lost sometimes between the mass of data. As a result, they have to be categorized in a top-down manner. The proper the classification of objects is, the more the clarity and user satisfaction would be obtained.

Traditionally, views about the file management are based on folders' hierarchical structure which is known as folders tree view. The folders are categorized hierarchically so that each folder located somewhere in the tree view may contain one or more files.

The benefits of the hierarchical file managers are so much that they are used commonly in almost all operating systems [9], [10]. The problem is with the bulk of files and folders especially in the recent years. Users can be confused when using stored information on their own hard disks. Moreover, the mass of folders available on their local drives makes the decisions severe in some cases to place a file in which folder. The same story occurs when looking for a file in different categorized folders.

However, To overcome those issues along with problems such as redundancy, ambiguity and lack of clarity in file managers, some fundamental changes are needed in the structure of information storage (hence in information retrieval).

Let us describe the issue more clearly. A hierarchical structure is used for the file system in most operating systems for PCs and mobile devices. A hierarchical file structure is a tree structure in which files and documents are stored in nodes. These nodes are called folders. Looking back to the parents of file systems used in current operating systems shows that hierarchy was a fundamental consideration in the design of their structure. During the development of these file systems in past years the hierarchical structure is remained unchanged as the base of them. For example in Microsoft File Systems from FAT to NTFS wide improvements and changes are done in the system layer that were mostly on implementation and security issues [11]. Despite all these changes, the hierarchy is still the base of the file system as previous.

The wide use of hierarchy in file system structures is because of its benefits in classification of files and documents. As seen in everyday life, using a tree is an easy and simple way to classify objects. Everyone is familiar with tree structures and can easily understand them. Therefore, hierarchical structures are easy to use and this is an important issue in acceptance and usage of a system by users. The other important factor is the simplicity of hierarchical structures. It is important from both end-users and developers' point of view. As mentioned, users can easily interact with tree structures as they face many hierarchical structures in everyday life. From developers' point of view, a tree structure can be easily implemented with simple data structures. More importantly, the hierarchical structure gives the users a top-down sight on the information structure so that no effort is consumed in the top levels to the trivial articles. This way rapid access to the tiny materials is acquired by level-by-level consideration of top-down structured data in smaller groups. The information is also eliminated rapidly as a user moves forward minor items.

However, this hierarchical structure has a limitation. Each file has to be placed in a folder. In other words, the folder is considered as the container of the file and access to the file is permitted through the folder. This is originated from the primary view toward files where the users dealt with a limited number of files. Hence a simple categorization was utilized that was to relate a file to a folder (as a place like wrapper). The folders then became more important than the files! Especially after growth of the number of files and folders, the user had to remember a sequence of folders (as a path) to retrieve a file. Although search facilities simplify this problem, the user in many cases doubts and faces challenges in choosing the container of a file between thousands of folders. There is also hard to determine the container of a new file. Considering the best match between thousands of available folders and also new possible folders is not simple.

Hence, both retrieval and storage face challenges; On one hand, you should find the best location for storage (although it is boring and sometimes impossible) and this is only one location. Note that considering different aspects different locations may be selected. On the other hand, when retrieving a file, you have to remember that location again. Again, note that this is only one location and that quick retrieval needs the precise storage as mentioned, although difficult or sometimes impossible. Also note that the best location for a file may change during the time.

As a result, ambiguity appears emerging vast number of files located in complex hierarchical folder structures. To have a more flexible file storage and retrieval, this hierarchical view is to be refined. Something more than a simple tree is needed to support more flexible visualization. In this paper, a new approach -File & Concept Browser- toward storage and retrieval is introduced that provides easier storage and retrieval.

The structure of the paper is as follows; in section 2, the literature review is shortly studied. Section 3 is devoted to introducing FCB solution and the prototype software. In section 4, experimental results on the developed prototype and a survey are described. Finally, section 5 concludes the paper.

2. Literature Review

Hierarchical information visualization has been extensively studied in [1][2][3][4][5] in different fields. For example, a new hierarchical integrated web classification approach is proposed in [1]. Their approach uses image-based and text based methods mixed together. They perform classification on a hierarchy differently on different levels of tree. Instead of using a flat classifier for text and image classification, texts are used for branches and images used at leaves.

A comprehensive method is offered in [4] for organizing enterprise content efficiently. This process includes creating taxonomies and building classification models for them. It mainly discusses about thematic mapping and has been applied to a large number of corpora of different genres. Newspaper articles, Usenet news group documents, patent documents, web documents from various content providers can be mentioned as examples.

An Email Management System is proposed in [2] which does some tasks in email management. Sorting messages to virtual folders, prioritizing, reading and replying automatically or semi-automatically, archiving and deleting mail items can be mentioned as these tasks.

In [5] the visualization of file system is addressed. It presents a novel approach for file system's search section. There is the ability to browse files and documents from a remote file system. Notice that there was no need to have them copied to the file system. It is an important facility for collaborative systems. The main idea is to make access to several remote file systems by a limited system with limited display and human input.

Finally, in [8] a prototype file manager called VennFS is designed to overcome some limitations of current file managers which are caused by hierarchical structures. It gives the ability of categorizing documents so that a file may be in several categorizes at a moment. Venn diagram is used to visualize categories. It allows users to use proximity to show similarities and relations between categories and files. VennFS is a novel idea in visualizing file system. A great work is done to solve the problem of categorization limitations in hierarchical file systems in cases that documents belong to several aspects and have to be accessed through several folders simultaneously. They have also provided tools for presenting relationships between categories and files by the help of proximity. Users can set data relationship as spatial relations. The other interesting capability is the ability of filtering documents according to their date. As recently accessed files are to be accessed by a higher approximation, they have provided an indication on recent use of files. It is shown by "hot" and "cold" for recently accessed and old files. However, they abandoned hierarchical visualization of the file system. Also, while system works well in many cases, finding an item with growth of the bulk of data would be tedious. Moreover, they can address a file from within at most four folders.

Suppose a user wants to have the video file of his/her paper presentation accessible both in "My Papers" and "My Videos" folders. This is because of logical dependency between the file and the two folders. According to [12], it is obvious that a precise organization of the information can lead to better retrieval. However, the first possible solution is to copy the video file to both folders. This leads to data redundancy that is undesirable. The redundancy problem causes ambiguity in having multiple versions of a single file in different folders, especially when the versions are updated frequently. It is also possible to use shortcut that is a small file containing the address of a target file or folder [13]. The problem is that shortcuts do not make two-side relationships. They are actually static files and cannot be synchronized by most possible changes that happen to the target file. Therefore, if the target file is renamed, deleted or moved, the shortcut becomes useless. Using more advanced available facilities also cannot solve the problem. For example, although Symbolic Links (Soft Links) and Hard Links [2] enhanced the shortcuts in MS Windows and Linux systems, they do not solve this problem totally and many issues remain unsolved. For example, Hard Links of a file fail to work when changing the place of the file. They save the attributes of the target file separately, so all of them change due to redundant information in the Hard Links. Symbolic links do not support pursuit addressing. They remain orphan when deleting the target file. Besides, changing the place of the target file makes its symbolic links useless. Using Junction Points [2] in Microsoft Windows is also a means of mounting a file or folder to another partition to be appeared here, but actually being stored somewhere else. It is used due to space limitations in partitions of disk. On the other hand, all of them add an extra layer to users' understanding of the file system even though all the mentioned aspects are corrected. As users prefer simpler systems, they would like a single layer system which covers both files and shortcuts in a single context.

In all the mentioned applications, the simple but powerful hierarchy was utilized. However, as mentioned in the previous section, this hierarchy causes ambiguity and some other problems in file system. Hence some modification and improvement are suggested in this paper. The approach of this paper differs from other optional links such as shortcut, Symbolic Link, Hard Link and Junction Point. It solves all the mentioned problems along with a proper visualization of conceptual relations between a file and all applicable folders.

3. File & Concept Browser (FCB)

As mentioned in section 1, maintenance of files and folders sometimes becomes ambiguous with growth of folders' hierarchy. However, the user chooses an available folder or makes a new folder to save the new file in. Of course it is one of the appropriate locations for that file, but not necessarily the best location. So it affects the retrieval progress later. While obtaining best locations in storage, there is no guarantee about retrieval on the first attempt (or one of the first few attempts). Nevertheless, there isn't always applicable to maintain the best locations; hence the problem is intensified in both storage and retrieval sides. On one hand, in storage of the files, usually a good location is chosen (or a new folder is created), but not necessarily the best location. On the other hand, several good locations (i.e., related locations to the respective file) will be looked for in order, but not necessarily the best location. In other words, failing to retrieve a file in the first attempt (or first few attempts) is accompanied with absence of exact best location for a file and generally the ambiguity problem in file storage.

This is not all the story. The ambiguity problem in storage phase causes redundant documents (rather with the same name or different names) due to the fact that the user doesn't find the document in one or more possible paths and gives up. So another copy of document is placed somewhere else (logically near, but maybe far in the folder hierarchy). Our survey (in section 4) shows that this happens frequently resulting two other problems: redundancy and multi versioning. The former is due to multiple copies of a document created over time and the latter is consequence of the ambiguity in retrieving and editing the different documents over time. All in all, the current folders' containment rule isn't perfect enough to pursue the needs of file storage/retrieval. It causes ambiguity, redundancy and multiple versioning.

3.1 FCB Solution

The goal is to access to a file as rapid as possible (i.e., in the first attempt). The user doesn't want to inspect several paths looking for a document. As mentioned before, the difficulties originate from storage limitation since a file should be located somewhere in the folder hierarchy. The determined folder in the chosen path will be considered the owner of the file, hence the file will belong to the owner and it needs only one owner.

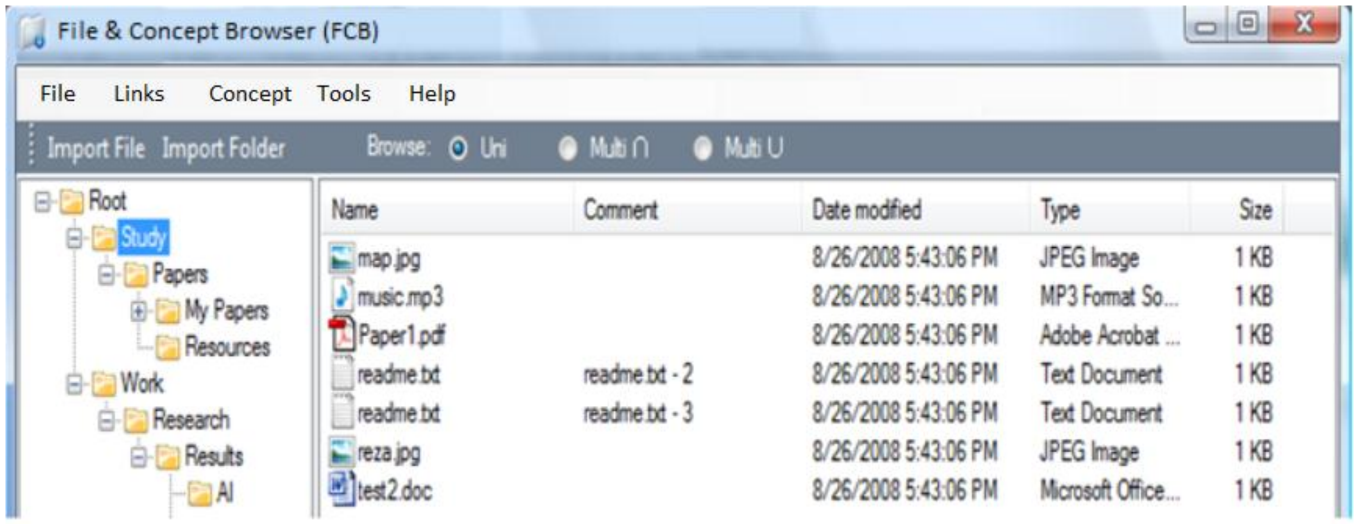
The solution changes this view so that a folder no longer is considered as the owner or container of files; nevertheless, it is only a concept to facilitate access to the files. So there will be no restriction in the owner ship or containment of the files. Files should be independent. The abstract view over the files necessitates their independence. However, the paths can remain unchanged to a large extent. As mentioned, a file can be addressed through the new notion toward folders, i.e., "Concept". A concept is a folder without ownership (or containment) toward files. A concept can point to several files as its members. At the same time, the files are free to be member of other concepts. Hence the abstract view over the files is preserved along with minor changes in file access from the point of view of the user.

In order to implement this structure, all the files have to be placed in a repository regardless of their membership in different concepts. In fact, a file's storage doesn't determine the place of that file. Instead, it needs to at least one concept to provide access to that file. While the files are stored in this flat repository, they are member of one or more concepts. So each file's icon is shown in the related concepts.

The user copies or moves the files in different concepts as in current file managers. No change in his/her transactions. But this is only the users' point of view. The system keeps no file ownerships. It preserves only the memberships; one or more for each file. Each membership connects a file to a concept. And the file's icon is shown in the all related concepts to preserve the current hierarchical view over the file system. As a result, a file is accessible through as many concepts as the user wants. In other words, multiple categorizations will be available.

By using "concept" instead of "folder", the user benefits freedom of copying a file in all the related concepts without actually duplicating it. There is no need to choose the best location. On the other hand, the retrieval phase doesn't face problems because the file is accessible through multiple paths (all the related paths based on the user's point of view can lead to the file). So no ambiguity in storage and retrieval takes place. Besides, the files are not duplicated anywhere by keeping a file only once in the repository and storing all the file-concept relationships to make several access points via different concepts.

The FCB system keeps all the information about the followings:



Name	Comment	Date modified	Type	Size
map.jpg		8/26/2008 5:43:06 PM	JPEG Image	1 KB
music.mp3		8/26/2008 5:43:06 PM	MP3 Format So...	1 KB
Paper1.pdf		8/26/2008 5:43:06 PM	Adobe Acrobat ...	1 KB
readme.txt	readme.txt - 2	8/26/2008 5:43:06 PM	Text Document	1 KB
readme.txt	readme.txt - 3	8/26/2008 5:43:06 PM	Text Document	1 KB
reza.jpg		8/26/2008 5:43:06 PM	JPEG Image	1 KB
test2.doc		8/26/2008 5:43:06 PM	Microsoft Office...	1 KB

Figure 1. FCB Environment.

-Files (in a flat repository, hidden from the user)

-Concepts (managed by the user as folders)

-Files' membership in concepts (managed by the system, also indirectly by the user with file operations, i.e. copy-paste, cut-paste, delete, etc.)

Note that the user performs file operations just as he/she does now. He/she finds concept's hierarchy (as folders') and when choosing a concept (and making it highlighted), is noticed by icons of the files that are member of the respective concept (or have relationship with the concept).

FCB not only has the abilities of hierarchical file systems with resolved ambiguity and redundancy problems, but also benefits new capabilities that are not available in hierarchical structures. Concepts, which are actually mathematical sets, are defined to be used instead of folders supporting multiple categorizations.

3.2 FCB Software and its Operations

The FCB prototype was designed and developed with Microsoft Visual Studio 2008 based on .NET 2 Framework. It contains six Windows Forms and three Modules. Because of the ease of use for end-users, Microsoft Access is chosen as its database to save the file-concept relationships. In this part, first, the UI is introduced, and then the fundamental file operations are discussed by describing different parts of the software.

3.2.1 Software Environment

FCB looks like usual file browsers at first look (see Figure 1), but it benefits many fundamental differences with them. First of all, the structure of file storage systems they are developed for are different. There are also some differences in the environment that are described here.

Just like many other file browser software, there is a tree structure in the left side of FCB environment and a file panel in the right. The tree view shows the classification of concepts. The proposed file manager uses two selection methods and three browse modes: Selection methods are used to choose concepts while browse modes are used to refresh file list:

- Selection methods (for concepts):

1- Select a single concept by clicking on its name
2- Select one or more concepts by placing a check mark on the provided check box near each concept

These can be selected simultaneously. However, only one of them (i.e., the highlighted concept or all the ticked concepts) are inspected based on the selected browse mode.

- Brows modes (to show files):
  1- Uni
  2- Multi
    - ∩
    - ∪

In Uni mode, the highlighted concept is observed and all the files that are in relationship with that concept are shown in the file panel. But in Multi modes (∩, ∪), the ticked concepts are observed and the following operations are performed:

-∩: All the files that are in relationship with all the ticked concepts are shown in the file panel.

-∪: All the files that are in relationship with each of the ticked concepts are shown in the file panel.

These two modes are useful when dealing with overcrowding concepts or looking for a file in sparse concepts. They let the user limit the browsed files.

For example, when facing so many files dealing with concepts, using "Multi ∩" mode and placing checkmarks beside two or more concepts will be so powerful in retrieving the desired documents. Or in a less common situation when the user doesn't see the desired file via neither of the browsed concepts (usually in sparse concepts when most of the concepts are linked to no files or only a few files), using "Multi ∪" mode and placing checkmarks beside two or more concepts again will be so powerful.

In addition to the browse as benefits of FCB, the difference here comparing with traditional file managers is that hence the file panel can be result of set operations (i.e., ∩, ∪), it doesn't contain concepts.

Consider Figures 2 and 3. If the files F1, F2, F3 and F4, are members of study concept and files F3, F4, F5, F6, and F7 are

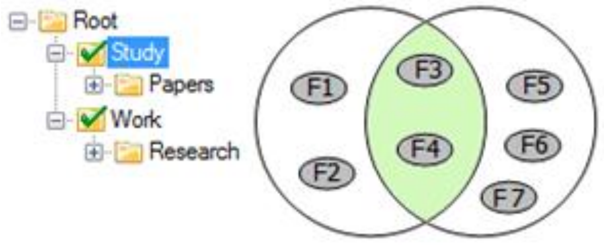


Figure 2. "Multi ∩" browses mode and the results shown graphically

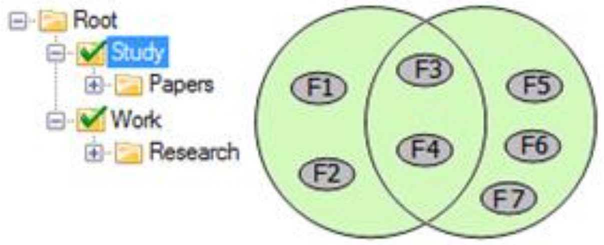


Figure 3. "Multi ∪" browse mode and the results shown graphically

members of work concept, then the result of browsing in "Multi ∩" mode will be F3 and F4 (see figure 2), but the result of browsing in "Multi ∪" mode will be all the files from F1 to F7 (see Figure 3). However, the result of Uni browse mode in both figures will be F1 to F4 because the concept "Study" is highlighted.

As mentioned before, Multi modes help the user avoid over populating the file panel. For example in Figure 4, seven files are in relationship with concept "Study" (i.e., "Study" has 7 members), but only two of them are also in relationship with "Work" (i.e., are members of "Work").

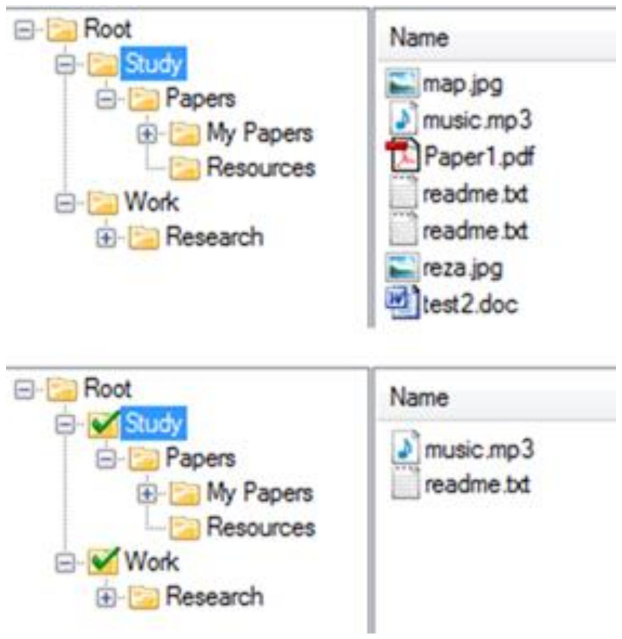


Figure 4. Comparing the results in Uni and "Multi ∩" browse mode.

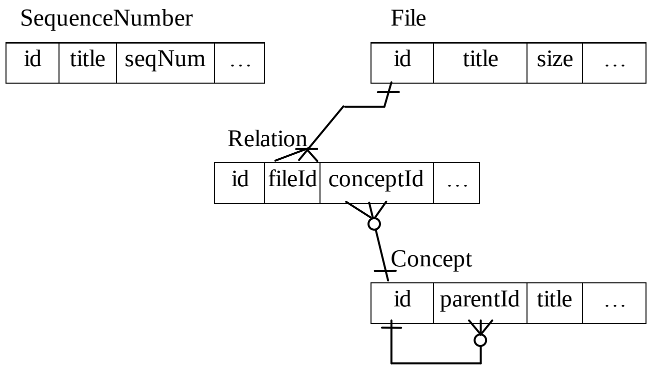


Figure 5. The basic tables needed for FCB in database manner.

Figure 4 (First part) also indicates that a concept that a concept can address more than one file with the same name but different contents (e.g., "readme.txt" in this figure). This was one of the problems in early discussions in FCB analysis and design. The conclusion was that hence the files are to be seen abstract –and separate from folders (or concepts)- and they will be saved in a flat repository, it is both reasonable and practical to allow multiple files with the same name via a concept. Abstract view over files says that the files are not identified by only their names. As a result, a unique comment is created for each file as its identifier. It is constructed from the name of the file following a "-"and a sequence number issued by the FCB system considering the flat repository for the file name in all the files. For example, if there are three files named "readme.txt" in the system, the fourth "readme.txt" file's comment will be "readme.txt - 4". Note that there is no need to look for a name in the entire repository to count the number of that file to issue the comment. Nevertheless, there is an independent table (named SequenceNumber) containing different names of files and the last sequence number issued. Figure 5 shows this table along with two tables for general information about files and concepts and another for their relationships.

Note that the comments are hidden from the users as much as possible (although there is an option available from "Tools → Concept Options" to show the comments everywhere; see Figure 6). So the comments aren't shown unless the results of browse are two files with the same name in the file panel. In this case, the comments isolate them from each other.

3.2.2 Importing Files and Folders

"Import File" and "Import Folder" can be used to import new files and folders to FCB. Considering that there are two methods of choosing where to add the selected files, if the browse type is set to "Uni" mode, files will be related to the selected (highlighted) concept. But if it is set to any of the "Multi" modes, files will be related to all the marked concepts by checkmarks. A whole folder can also be imported using "Import Folder" option. By importing a folder, a concept is created in the concept tree for each of its sub-folders. All the files are created physically in the system and the appropriate relationships will be added with links to the respective concepts.

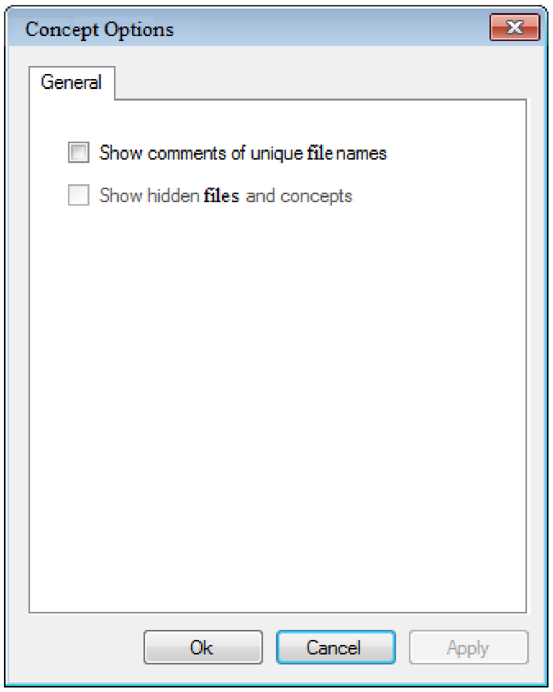

Concept Options
General
Show comments of unique file names
Show hidden files and concepts
Ok
Cancel
Apply

Figure 6. Concept Options

3.2.3 *Concept Pop-up Menu*

Concepts drop-down menu (Figure 7) consists of seven items: New, Cut, Copy, Paste, Rename, Delete and Uncheck All. It appears when right clicking on a concept in the concept tree view. Definitions of them are explained bellow:

New: add a new concept inside the selected concept.

Cut: stores selected concept (and all is sub-concepts hierarchically) in clipboard with cut flag.

Copy: stores selected concept (and all is sub-concepts hierarchically) in clipboard with copy flag.

Paste: copies the concept in the clipboard (and all its sub-concepts plus all the links with the files) to the selected concept or moves the concept in the clipboard (and all its sub-concepts plus all the links with the files) to the target, based on the last copy/cut triggered operation. Note that the browse mode doesn't affect the

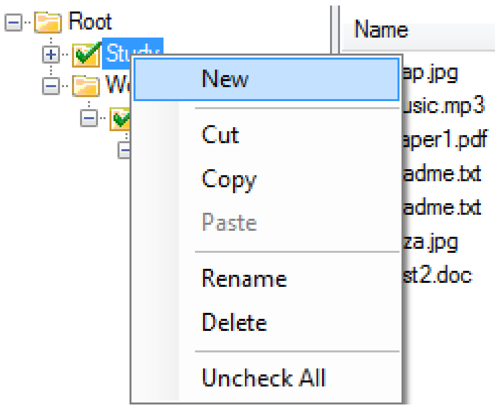


Figure 7. Concept Pop-up Menu

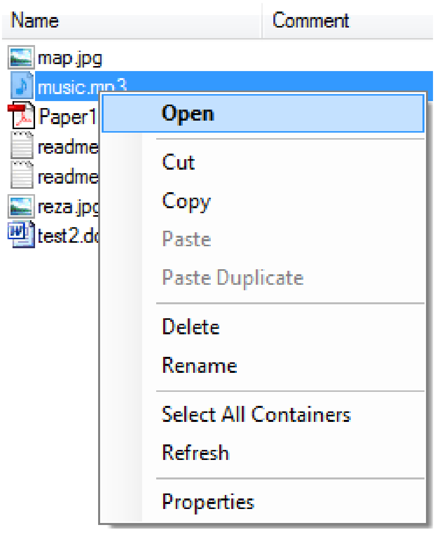


Figure 8. File Pop-up Menu

concept operation considered in this section. They only affect the manner of visualizing the files and working with files panel. Also note that both copy-paste and cut-paste operations don't do anything with the original file stored in the flat repository. They only manage relationships between the file(s) and the highlighted concept. For example, when copying file "readme.txt" into "Research", it only links it to the concept "Research".

Rename: used to rename a concept.

Delete: deletes a concept with all its sub-concepts and their relationship with files. Note that if a physical file has only relationships with some of these concepts (and has relationship with none of the other concepts), it will also be deleted physically.

Uncheck All: removes checkmarks of all checked concepts.

3.2.4 *Files Pop-up Menu*

By right-clicking on one or more files, a drop-down menu will be shown as seen in Figure 8. Ten items of this menu are Open, Cut, Copy, Paste, Paste Duplicate, Delete, Rename, Select All Containers, Refresh and Properties. Here is definition of each menu item:

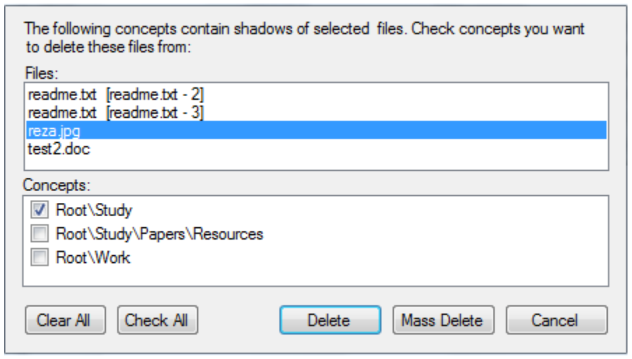
Open: opens the selected files with its appropriate application.

Cut: stores selected files in clipboard with cut flag.

Copy: stores selected files in clipboard with copy flag.

Past e: Again, ba sed on the last issued copy or cut command, it may only create new relationships between the copied file(s) and the target concept ("copy" case) or also remove the relationship(s) between the cut file(s) and source concept ("cut" case). As a result, copying or moving acts on file relationships (or file memberships) with the concepts. However, due to simplicity it may be referred to coping or moving files.

Based on the selection mode, the target may be the highlighted concept if the "Uni" browse mode is chosen or all the checked concepts if any of the "Multi" browse modes is selected. The physical file is not moved or copied and all operations are performed on relationships (i.e., memberships or links).


The following concepts contain shadows of selected files. Check concepts you want to delete these files from:

Files:

readme.txt [readme.txt - 2]
readme.txt [readme.txt - 3]
reza.jpg
test2.doc

Concepts:

[✓] Root\Study
[ ] Root\Study\Papers\Resources
[ ] Root\Work

Figure 9. Delete Dialog

Paste Duplicate: this item will be enabled when a file is copied into clipboard (it is disabled when a file is cut). It will make new duplications of copied files in selected (highlighted) concept or all the ticked concepts based on browse mode, ("Uni", "Multi ∩" and "Multi ∪").

Delete: used to delete the membership of the selected file(s). The physical file(s) may have many other links. All these links will be found and the user is then asked which instances (memberships or links) of each file is to be deleted, then a confirmation containing two list boxes which one of them lists selected files is appeared. By selecting a file, all its related concepts (that the file is a member of) are shown in the other list box with their paths (Figure 9). Users can select which one to be deleted and which one to be kept. There is an option to check all the concepts pointing to that file and also a button to delete all the relationships of all the files in the first box plus the physical files ("Mass Delete").

Rename: renames selected file. Consider that when renaming a file, it will be appeared with the new name while accessing from each of the linked concepts.

Select All Containers: all concepts which selected file is member of them will be checked after clearing the checkmarks previously ticked beside concepts (Figure 10).

Refresh: refreshes file panel.

Properties: It is similar to properties of files in Microsoft Windows, but there is also some extra information here. The name and comment of the file are shown at the top of the form (Figure 11). Following that, type, size and dates related to the physical file are seen. There is also a useful list box containing all linked concepts. These are in fact the equivalent access paths. At the end of the form, user can find information about attributes of the physical file such as being read-only or hidden.

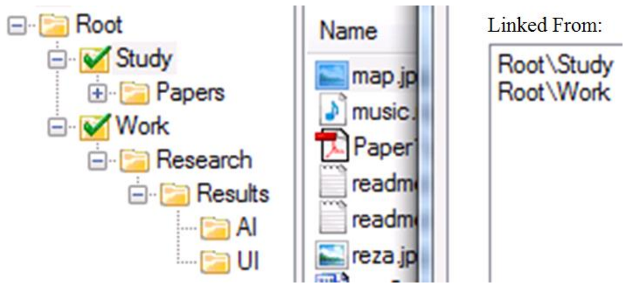
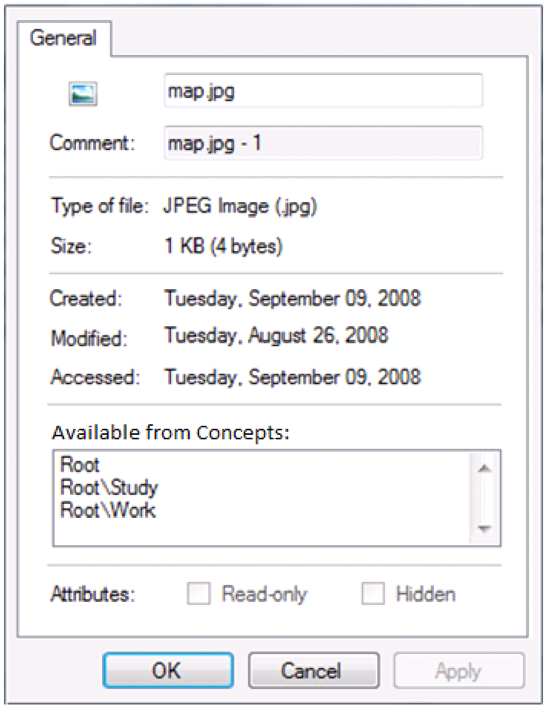


Figure 10. Select All Containers of a File



	map.jpg
Comment:	map.jpg - 1
Type of file:	JPEG Image (.jpg)
Size:	1 KB (4 bytes)
Created:	Tuesday, September 09, 2008
Modified:	Tuesday, August 26, 2008
Accessed:	Tuesday, September 09, 2008

Available from Concepts:

Root
Root\Study
Root\Work

Attributes: [ ] Read-only [ ] Hidden

Figure 11. File Properties Dialog

4. Experimental Analysis

The software was developed to evaluate FCB system's challenges and also the reflection of users against this approach. New abilities of the FCB were discussed through the analysis, design and implementation processes to achieve a consistent understanding.

Several experiments have been done in order to consider the effects of FCB. First, a questionnaire was designed and the user's opinions were collected to determine several directions. Then, several computers have been examined to determine the redundancy rate. Finally, the FCB prototype was used in order to see the overall users' feedback after using it.

4.1 Questionnaire Analysis

A questionnaire containing 20 closed questions categorized in 6 groups about current file systems was designed. It was offered to 120 students in different majors (but mostly engineering students) with different levels of computer expertise (expert, intermediate and novice; almost with equal distribution). 110 students filled the questionnaires. The summary of the result is shown in Table 1.

The results show that almost 69.7% of the students experienced multiple copies of a file in the folders occasionally. 44.1% of them have tried ambiguity when saving a file between several folders most of the times. 50.6% of them indicated that finding the files is ambiguous in their hard drive and 65% faced low disk space. Also 70.5 percent faced redundant files so often and failed to manage them.

The two last question groups are related directly to the benefits of the FCB system. As considering all the linked paths to a file in FCB is a useful capability, 70.5 percent of the users mentioned this as a desired capability. Also after linking a file to all related concepts in their storage time, FCB helps the users find the file in their first attempt. This is the mostly popular item between the

Table 1. Answered questions by the users (%)

Question	Result
Copy a file in multiple folders (for several different files)	69.7
Ambiguity in saving a file (between several folders)	44.1
Ambiguity in finding a file (between several folders)	50.6
Faced low disk space	65
Fail to manage redundant files.	70.5
Is desired to find all copies of a single file in the system at a glance.	77.6
Is highly desired to find a file in the first attempt	94.5

users (94.5% of the users believed that this would be a highly desired capability).

4.2 A brief PC Analysis

In order to obtain a general sight on the redundancy problems in current file managers, a limited number of computers of 10 software engineering students were examined.

Directory Opus 9.0 [14] was used in our study to evaluate several separate Windows hard drives on the PCs. The evaluation shows that in average 10,000 files out of 70,000 containing 1GB to 5GB out of 40GB to 80GB files available on the Hard Disk were redundant items. This result shows that a great amount of the space of a hard disk is occupied by 1 to 10 extra copies of existing files. Further studies on discovered redundant files showed that some of them were created automatically by the system and some were copied by users. However, a large number of redundant files were created and copied by users. As discussed before redundancy may be a result of ambiguity in file storage and wastes storage space.

4.3 Prototype Analysis

Nine computer students participated in a two-day experiment to store and retrieve documents using both paradigms (i.e., Folder paradigm with "MS Windows Explorer" or "My Computer" and FCB paradigm with our implemented prototype system). All of them were moderate or expert users.

The users were not familiar with FCB software before, or the FCB approach; hence the chance was given to them to familiarize themselves with the FCB prototype for at most one hour after demonstrating the idea and the program. Although the time is not sufficient to be proficient FCB users, they all finished the practice after 15 to 45 minutes. According to their opinion, the UI was easy-to-learn and it resembled the UI and the operations of Windows Explorer. Therefore, they were able to make hierarchies of concepts and categorize the documents easily.

A competition was set up for making best categorizations in either paradigm. The users were divided into two groups; the first four users examined the folder paradigm first and the other five examined the FCB paradigm first. Interestingly, there was no significant difference between the average results of two groups. Hence the final results are studied together.

First, a folder containing 88 documents was given to the users. They had to open each document, create folders or concepts regarding the examination (i.e. folder or FCB) and categorize them in the folders/concepts by cutting/copying and pasting operations provided in FCB and MS Windows. In MS Windows experiment, they could use drag and drop operations too.

They created an average of 15 folders and 25 concepts in either experiment. For FCB approach, a bit more time was needed to categorize the items due to the fact that each file had to be related to several concepts. In other words, a file should be connected to more than one concept; as a result, the time slightly increases (not so much) because the user does not have to place a file in its best location. He/she can paste a file in several concepts to increase retrieval efficiency without considering size or redundancy matters. On the other hand, they were unfamiliar with the FCB previously, but with lots of folder experiences in several years.

Then, in order to perform the test, 19 randomly selected files were demanded one by one and the time to find the files were recorded. The selected files were equal for the two experiments (i.e. folder and FCB paradigms) to achieve comparable results, but the users did not know this before finishing their categorization. Each file search was started by giving the name of the document or some clues from its contents.

In order to control the experiment synchronously and preventing exhaustion and give up, a maximum threshold of two minutes was applied for each test case. The test case result was recorded as "failure" if the user was not able to retrieve the file in this time limit. Trial experiments showed that in such cases the user usually gives up or spends a long time (longer than three minutes in average) for retrieving the file. As a result, the time 160 seconds was chosen for these cases.

The average access time in FCB was 57.0 seconds (σ=23.6) against 62 seconds (σ=18.6) in folder paradigm (7.9 percent faster in FCB) as shown in Table 2.

However, comparing the "failure rate" of the users are far more interesting;

On average, 2.22 cases out of 19 test cases were failed to be found in FCB (σ=1.2) against 4.33 cases in folder paradigm (σ=2.3). 47.8% decrease in failed attempts as shown in Table 2.

In the real situations the user gives up when he/she fails finding the file by examining several possible locations and creates an extra copy of the file often with a different name. Our "PC Analysis" supports this claim.

However, we can expect better results for FCB when the experience time is increased. In other words, more practice with the software will generate better results.

Table 2. FCB test results compared with folder paradigm

	Average access time	Average failures (out of 19 cases)
Folder paradigm	62 (σ=18.6)	4.33(σ=2.3)
FCB paradigm	57(σ=23.6)	2.22(σ=1.2)

5. Conclusion and Future works

A method was presented to gain a flexible, flat visualization to the file system. In this method, the “folders” are changed to “concepts” and there is no longer the containment rule for them. So the paths become only alternative ways to access a file instead of keeping them.

The proposed method reduces ambiguity in both storage and retrieval periods. On the storage time, instead of looking for best container or owner, the user just can copy a file in several related concepts. On the other hand, it will be accessed often in first attempt although this needs an accurate storage relating the file to all (or almost all) related concepts.

It is also mentioned that it decreases redundant documents while preserving ease of access. On the users’ point of view, they can copy a file everywhere, but in fact only the handle to the file is distributed nor the file itself.

Since FCB benefits addressing to a file via different concepts, in general, the files addressed from a concept may be overcrowded in some cases. In this case, using two additional browse modes (“Multi ∩” and “Multi ∪”) help the user gain quick access to the file. Hence it can be used as a search tool as well as a file browser. It can also be used as a PIM tool allowing the users benefit multiple categorization of their personal content (from files and folders to e-mail messages).

The FCB prototype was developed to both investigate the practical aspects and evaluate its application. Comparison between the multiple categorizations offered by FCB approach and the traditional hierarchical categorizations shows that FCB not only reduces access time to files, but also prevents shortcoming, failure and frustration in retrieving documents. In addition, the first two experimental results show that massive amounts of redundant items are scattered throughout the systems. They also show that this is an undesirable situation emerged gradually by massive growth of files and folders.

The information breakthrough in these decades took place in both sides; Information volume and information storage devices (software and hardware). On one hand, information volumes in different forms from files and folders to web information, from mobile contacts to e-mail messages, etc., became thousands and even millions of times increased. On the other hand, software and hardware devices renovated to new comprehensive friendly devices. In case of files and folders, although there are valuable improvements in supporting large volumes of data, the classification styles follow the primary hierarchical method. Almost all the file managers have the same hierarchical foundation. The information society needs something more than simple hierarchies to be utilized in storage and retrieval. FCB was a solution to this demand.

However, during the design, implementation, examination and user studies some new ideas appeared;

First, utilizing much more visualizations can improve the program. For example, utilizing more than one hierarchy and using “Back” and “Forward” keys to traverse the concepts more efficiently are useful.

Then, Embedding the FCB system in routine and usual programs that deal with files and folders would be valuable. For example, asking to save a file in several concepts in a “Save as” dialog.

Finally, exploiting the proposed idea in other different platforms such as e-mail management systems would be worthwhile.

6. REFERENCES

[1] Lu, C., Drew, M.s.2001, "Construction of a hierarchical classifier schema using a combination of text-based and image-based approaches," Proceedings of the 24th annual international ACM SIGIR conference on Research and development in information retrieval. ACM New York, NY, USA. 438 – 439. DOI=http://doi.acm.org/10.1145/383952.384075

[2] Ho, V., Wobcket, W., Compton, P. 2003, "EMMA: An Email Management Assistant," Proceedings of the IEEE/WIC International Conference on Intelligent Agent Technology. IEEE Computer Society Washington, DC, USA. 67. DOI= http://doi.ieeecomputersociety.org/10.1109/IAT.2003.1241050

[3] Kandogan, E., Shneiderman, B. 1996, "Elastic Windows: improved Spatial Layout and Rapid MuRipme Window Operations," ACM Special Interest Group on Multimedia. 29– 38. DOI=http://doi.acm.org/10.1145/948449.948454

[4] Chung, C.Y., Lieu, R., Jinhui, L.,Alpha, L., Jianchang, Mao., Prabhakar, R. 2002, "Thematic mapping - from unstructured documents to taxonomies," Proceedings of the eleventh international conference on Information and knowledge management," McLean, Virginia, USA. 608-610. ACM,1-58113-492-4. DOI= http://doi.acm.org/10.1145/584792.584892

[5] Collins, A. 2007, "Exploring Tabletop File System interfaces," CHI 2007, April 28—May 3, 2007, San Jose, California,USA. ACM 978-1-59593-642 4/07/0004.

[6] N. Bansal, S. Guha, N. Kudas, "Ad-hoc aggregations of ranked lists in the presence of hierarchies," Proceedings of the 2008 ACM SIGMOD international conference on Management of data, PP. 67-78, ACM New York, NY, USA, 2008.

[7] Matthew Willis, " Building effective help systems: modelling human help seeking behavior," Proceedings of the 18th Australia conference on Computer-Human Interaction: Design: Activities, Artefacts and Environments, pp. 433 - 436, ACM New York, NY, USA, 2006.

[8] D. chiara, R.,Erra ,U.,Scarano ,V, "VENNFS: A Venn-Diagram File Manager," Seventh International Conference on Information Visualization, pp. 120. IEEE Computer Society Washington, DC, 2003.

[9] File managers, http://en.wikipedia.org/wiki/Comparison_of_file_managers

[10] File managers, http://www.linux.com/articles/59043. 2009

[11] Microsoft File Systems from FAT to NTFS, http://support.microsoft.com/kb/100108. 2009

[12] Information Organization meets Information Retrieval: Rethinking the iSchool Core – Panel; Borgman, C.,

Mayernik, M., Larsen, R., Glushko, R., Hemerly, J., , February 8-11, Seattle, US, iConference 2011

[13] Shortcut, http://kb.iu.edu/data/abhm.html. 2009

[14] GP Software, http://www.gpsoft.com.au/ 2009