Metadata-Derived Keywords for Effective Navigation and Access in Personal File Hierarchies

Gontlafetse Mosweunyane[1], Leslie Carr[2], Nicholas Gibbins[2]
[1]Department of Computer Science, University of Botswana, P/Bag 00704, Gaborone, Botswana, mosweuny@mopipi.ub.bw
[2] School of Electronics and Computer Science, University of Southampton, Highfield, Southampton SO 17 1BJ, United Kingdom, {lac, nmg}@ecs.soton.ac.uk

ABSTRACT

Computer systems provide users with abilities to create, organize, store and access information. Most of this information is in the form of documents in files organized in the hierarchical folder structures provided by operating systems. Operating system-provided access is mainly through navigation of the file hierarchy, and through keyword search. An investigation with regard to access and utilization of these documents revealed a need to reconsider these navigation methods. An improved method of access to these documents is proposed based on previous effective metadata use in search system-retrieval and annotation systems. The underlying organization is based on a model for navigation whereby documents are represented using index terms and associations between them exposed to create a linked, similarity-based navigation structure. Evaluation of an interface instantiating this approach suggests it can reduce the user's cognitive load and enable efficient and effective access and navigation of the file hierarchy. This is enabled by provision of keywords as reminders for discovering and associating documents.

KEYWORDS

Personal, file hierarchy, Metadata, keywords, retrieval, Linked navigation.

1 BACKGROUND

1.1 Introduction

Information overload, especially with information in digital form, is a widely recognized and experienced phenomenon [1], as well as being a well-discussed issue in research. Users of computer systems have access to and create large amounts of information. Storage has become more affordable [2] resulting in an increase in storage capacity of devices, allowing individuals to store more data in what has been termed "personal archives" [3].

Most of this data is in the form of documents in files organized in hierarchies on the user's computer system. Gemmell *et al* [4] predicted that by now terabyte hard drives, with capacity to store 2900 1MB documents per day for a year, will be common and inexpensive. This is now the case with even capacity up to 2TB reported, although desktop hard drives are rarely above 500MB[1].

This paper proposes an automatic generation of tag-like keywords to aid linked navigation and discovery of documents stored in the hierarchical file/folder structure provided by operating systems.
This method was hypothesized to improve accessibility of documents

[1]http://en.wikipedia.org/wiki/Hard_disk_drive#cite_note-2TB-15

through exposure of the tag-like keywords from document properties to provide shorter and precise paths to, and a view of connections between documents.

1.2 Current Methods and Problems

Traditional operating systems employ the use of the desktop metaphor for organizing information, where the digital desktop is managed as the physical one, making desktops behave in a more physically realistic manner. The monitor of a computer represents the user's desktop upon which documents, and folders containing documents, can be placed. A document can be opened into a window, which represents a paper copy of the document. Files can also be spatially arranged on the desktop individually or in groups in different sections of the screen.

A large part of managing documents involves organizing them, and this is done mostly by using hierarchical structures provided by operating systems [16]. Systems based on the desktop metaphor allow for information to be stored in documents in files that can be named and placed in folders that can be nested to form hierarchical structures. This storage model is used on the most pervasive computer systems such as Microsoft Windows and the Apple Macintosh and is the only work environment known to many users and designers [17].

The hierarchical file system model matches the underlying data storage on devices and was initially used to provide efficient access to files on disk [7]. Operating system-provided access to information stored using this model is mainly through navigation guided by the structure which is based on the location of the files, and through keyword search.

The operating systems also help users to create personalized views of the search. Tags or keywords attached to files help the user and search systems locate more relevant items. For example, in Windows Vista search results can be organized based on file properties like filenames, file types, author, or descriptive keywords (“tags”) that the user added to the files. The files can also be arranged by type, for example, documents, spreadsheets, or presentations.

Newer operating systems such as Windows 7 (Microsoft Corporation, 2006) and Apple Macintosh OS X (Apple Inc) have incorporated advanced search technologies in their systems that allow users to sort or group results according to their needs. Both operating systems provide powerful search mechanisms based on indexed file metadata and contents, and methods to dynamically organize the results according to the file attributes. The Finder and Spotlight in Mac OS X and the Windows 7 Start Menu search box provide instant access to both documents and applications.

These system-provided methods have limited abilities to organize files spatially, temporally and logically [18]. The hierarchical file system method of organization provides simple and intuitive navigation of the whole file system [17], but it has also proved to be mainly static, and presents problems in categorizing, finding items later and reminding users of what items they have [19] among other problems.

2 OTHER FILE HIERARCHY ACCESS SOLUTIONS

2.1 File hierarchy Interfaces

Hierarchies are one of the most common and important information structures in Computer Science [31] and have been used in a wide range of applications, from file systems to ontology management and digital library categorizations [22]. Several methods have been applied by researchers to offer improved methods of displaying and visualizing the data stored in hierarchically structured information spaces.

Visualization tools increase the bandwidth of interfaces [32] by graphically encoding the information to help humans make sense of and analyze the information set. Hierarchy visualizations have been classified into broad categories such as the indented list (for example, Microsoft Windows Explorer), node-link trees (for example, two-dimensional node-link diagram), zoomable user interfaces, space-filling techniques (for example, TreeMap), and context + focus techniques (for example, hyperbolic tree) [22, 33]. Some of these hierarchy visualizations have been used to depict personal file hierarchies. Researchers have also found some to be more effective for specific browsing tasks.

2.2 Semantic Information Management Tools

2.2.1 Introduction

Tools using semantic information have been implemented for integrating information items on the desktop, including connections to reach documents, and the ability for the user to manage and explore the connections between these items. The main aim of these integration tools is to solve the problem caused by fragmentation of information across different applications and devices by bringing it together in one interface.

2.2.2 SEMEX

SEMEX (SEMantic EXplorer) is a personal information system offering search-by-association [23, 24]. Information browsing is provided through an underlying ontology which can be personalized by users. Users can browse their personal information by semantically meaningful associations previously created to allow for easier later integration by the user.

SEMEX provides a generic domain model of classes and associations between them and uses this to organize data. This model can also be extended by users, for example, by their browsing pattern. A database of objects and associations between them is represented as RDF, and this is stored and retrieved using Jena. Lucene is used to index object instances by the text in their attribute values. The database supports “on-the-fly" integration of personal and public data by keeping associations and previous activities that the user performed. Users can then browse association links or do keyword search, selection-query search or association-query search.

When executing a query, SEMEX also tries to deduce other related objects that are related to the matches found, but not necessarily specified in the query. Heterogeneous data is managed and many different references to the same real-world object are reconciled.

2.2.3 Haystack

Haystack [25, 26] is a Java-based, open source system that aims to cater for different users' preferences and needs by giving them control and flexibility over storage and retrieval. It caters for storage and manipulation of task-oriented information sources like email, calendar and contacts. Users can define and view connections between their personal data.

A uniform resource identifier (URI) is used to name all individual information objects of interest to the user. These can then be annotated, linked to other objects, viewed and retrieved [26]. RDF is used to represent the data and to record the relationships between the objects. The data is extracted from applications and stored in an in-memory database.

Capabilities are provided for users to browse their personal information in one location such that information from different applications and applications such as email, address book, documents hierarchies and the web are brought together in a single view. The user can also add properties to capture any attributes or relationships between the information. These properties can be used as query arguments, for metadata-b-ased browsing, or as relational links to support the associative browsing as in the World Wide Web. A search is also offered as an alternative to the task-specific starting points provided. Multiple views of the same object are offered to allow the user to use an appropriate view based on their task. Views in the system can also be customised using view prescriptions, which are collections of RDF statements describing how a display region should be divided up and which constants and related objects should be shown in the subdivisions.

Items could be grouped into collections, and views like calendar view and menu view are provided especially for these. In addition the lens view is provided to allow customization of presentation of objects, for example, to show certain properties. The user can view email messages and select people to view data related to them.

2.2.4 Gnowsis

The Gnowsis system [5, 6] is a semantic desktop prototype which aims to integrate desktop applications and data managed on desktop computers using Semantic Web technology. Desktop resources are treated as Semantic Web resources. A data integration framework is employed to extract information on the fly from common applications. The data and relationships between resources are then represented as RDF. Semantic Web interfaces are added to common desktop applications, allowing the users to browse their desktop like a small personal Semantic Web.

To relate information to the user's personal view of the world the Personal Information Model (PIMO) approach is used. The PIMO framework is made up of six components. PIMO Basic defines the basic language constructs and the superclass “Thing" of other classes. A domain-independent ontology containing subclasses of Thing is defined in PIMO Upper, while PIMO Mid integrates various domain ontologies and provides classes for Person, Project, Company etc. The domain model component describes a concrete domain of interest of the user. The user can also extend the above-mentioned models for personal use in PIMO User. Gnowsis now tries to incorporate web 2.0 features to the desktop by having users import their tags from tagging websites such as del.icio.us and Flickr and integrate them into their PIMO ontology [5].

2.3 Differences with this Research

The suggestions and attempts for improvement of visualizing hierarchies emphasized the need for provision of overviews over a collection, together with striking a good balance between provision of context and detail. Hierarchy visualization methods seemed to do well at providing context and overview by utilizing the available screen space for displaying the whole structure. Expand and contract or zoom mechanisms are used in some cases to change focus according to user needs while still maintaining context.

Space-filling techniques and node-link trees in particular have proved more suitable for tasks involving grouping or comparisons between files based on attributes. These techniques have already been found to perform better when users are more concerned mostly about leaf nodes and their attributes and do not need to focus on the topology of the tree, or the topology is trivial [34]. Good provision of overview and context through these visualizations therefore aids comparisons more than locating and navigation or viewing connections between information items.

Semantic Information tools differ from our approach in that they focus on integrating documents as part of a bigger focus of integrating web resources such as email, documents, and desktop applications and as such only help when one is looking for associated information items. On the other hand our idea is to integrate documents and improve navigation of desktop document. They are however similar in that they utilize attribute values to build indexes and utilize them for browsing associations.

3 DOCUMENT PROPERTIES as the SOLUTION

Metadata has been recognized as an important part of document organization systems and search systems, providing further information on documents for organization purposes or being utilized to enhance search results. Metadata about documents, in particular, has been used to help users understand more aspects of the documents, by search systems to improve search results and by classification systems to categorize documents.

For desktop documents metadata comprises file attributes that are supported either as part of the filesystem, known as *built-in* file properties, or as an additional feature that allows users to define and associate files with metadata outside the filesystem (these are known as *extended file attributes*). In Windows systems the built-in file properties include *user-defined* values like filename, author, keywords and comments, and *system-controlled* values such as creation date, last save time and number of pages.

The built-in file properties in Windows Systems are incomplete, with some fields empty, but this metadata, however minimal, can provide useful index terms which are useful for retrieval and discovery of documents as will be presented later in this paper.

By saving documents in folder hierarchies users are explicitly categorizing and linking documents. This view of folders as conceptual categories and document attributes has been expressed in literature before. By creating the hierarchy users are already specifying tags or annotations for the documents which can be utilized as document index terms to provide a shorter and more precise path to the documents. These, together with other file attributes, have been seen to play an important role in helping users view documents in their hierarchies. File attributes which include the folder hierarchy (paths) already define their context and form a comprehensive framework to the hierarchy which can be exploited to provide exploration lines for reminding and helping users discover information in their personal document archives.

4 METHODOLOGY

4.1 Metadata Harvesting and Specification

We extract all properties for commonly used documents types on the desktop using Windows Management Instrumentation commands. The commands require technologies such as the Component Object Model (COM) to enable a program to interact with the Windows Operating System through languages like Microsoft Visual Basic, C++ and scripting languages on Windows that can handle Microsoft ActiveX objects such as VBScript. These are used to crawl the hard drive within a given root, or a given top-level directory, to recursively explore the sub-folders and files contained therein in turn. For each file the built-in properties are extracted in the form of property-value pairs.

We extract all properties for commonly used documents types. The commonly used document types are Microsoft Office documents (Word, Excel and Powerpoint), html, htm and portable document format (pdf). Visual Basic Scripting Edition (VBScript) is used as a lightweight approach to implement the WMI classes on Windows systems, with the advantage of its ability to utilize the Windows Scripting Host to run directly on a user's computer. An example of some of the commands used for crawling a folder is given below.

```
StrFolderName=
CreateObject("Scripting.FileSys
temObject").GetAbsolutePathName
(".")

Set colSubfolders =
objWMIService.ExecQuery _

    ("Associators of
{Win32_Directory.Name='" &
strFolderName & "'} " _

        & "Where AssocClass =
Win32_Subdirectory " _

            & "ResultRole =
PartComponent")

arrFolderPath =
Split(strFolderName, "\")

strNewPath = ""

For i = 1 to
Ubound(arrFolderPath)

    strNewPath = strNewPath &
"\\" & arrFolderPath(i)

Next

strPath = strNewPath & "\\"
```

On the other hand similar Perl commands are used on Macintosh and Linux systems. Although the Perl script could be used on Windows systems this also requires a Perl interpreter to be installed, which is not ideal when dealing with test users. On the other hand Perl comes packaged with Linux and Mac systems while VBScript is included in Windows systems, hence the decision to use both. To make it easy to identify the files, properties and values extracted and relate them the metadata is structured into Semantic Web form (using the Resource Description Framework - RDF) and based on an ontology.

The ontology is designed to define the domain of desktop files in order to be able to represent metadata about files on the desktop in a concise and identifiable way. Some terms are reused from standardized ontologies like the Dublin Core and languages such as the RDF Schema (RDFS) for defining the metadata and the ontology specifies the additional ones that could not be found in the commonly existing schemas. The ontology was defined using SWOOP version 2.2.1 [20], a tool for creating and editing Web Ontology Language (OWL) ontologies.

Uniform Resource Identifiers (URIs) for files were defined following the specification RFC 1738 [21] which describes Uniform Resource Locators (URLs), strings that allow for location and access of resources via the Internet. A URL is in the form

file://<host> <path> >

where host is the fully qualified domain name of the system on which the path is accessible, and path is a hierarchical directory path. The URI provides a direct link from the metadata, and therefore from the application implemented, to the document. The overview of the ontology is given in Figure 1 and an example RDF description for a file is given in Figure 2.

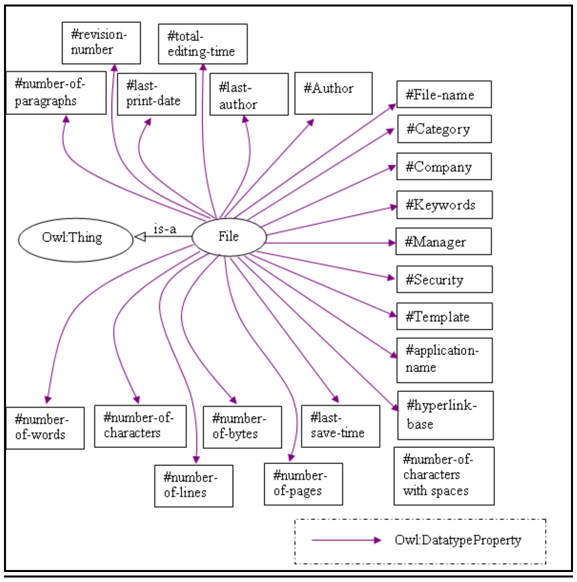


Fig. 1 Overview of the File Ontology

4.2 Index Derivation and Clustering

A lightweight model that utilizes the documents attributes is proposed to facilitate retrieval and discovery of documents from file system hierarchies. The model is based on the metadata derived from the file system as above, and determination of similarity between documents.

In hypertext and hyperindexing indexes serve as aids that facilitate the location of objects [8]. If ordered, index entries can facilitate the quick location of relevant entries. The semantic metadata is processed to build for each document a forward index by deriving terms from the metadata values which then serve as index terms or keywords in document recognition, retrieval and clustering. The terms are actually any text fragments which are recognized by the algorithm as single words but might only make sense to the creator as they may be “codes” used, for example, to describe files and folders. The algorithm for building the indexes for each document or file is as follows.

First the metadata is queried for all property values relevant to a given document. The extracted property values are all treated the same irrespective of property value. While being extracted some pre-processing is carried out to select useful terms in the multi-valued attribute values. This is done by removing stopwords (for example words like “and”, “the”, “is”), punctuation (replace with spaces to separate terms) and converting all the values to low case to facilitate easy comparison. Stemming, which is usually done in indexing to eliminate variation caused by presence of different grammatical forms of the same word (for example, “research” and “researched”), is not carried to avoid intervening with user's intended meaning of concepts. This is mainly for purposes of presentation of these for view by the user.

4.2 Index Derivation and Clustering

A lightweight model that utilizes the documents attributes is proposed to facilitate retrieval and discovery of documents from file system hierarchies. The model is based on the metadata derived from the file system as above, and determination of similarity between documents.

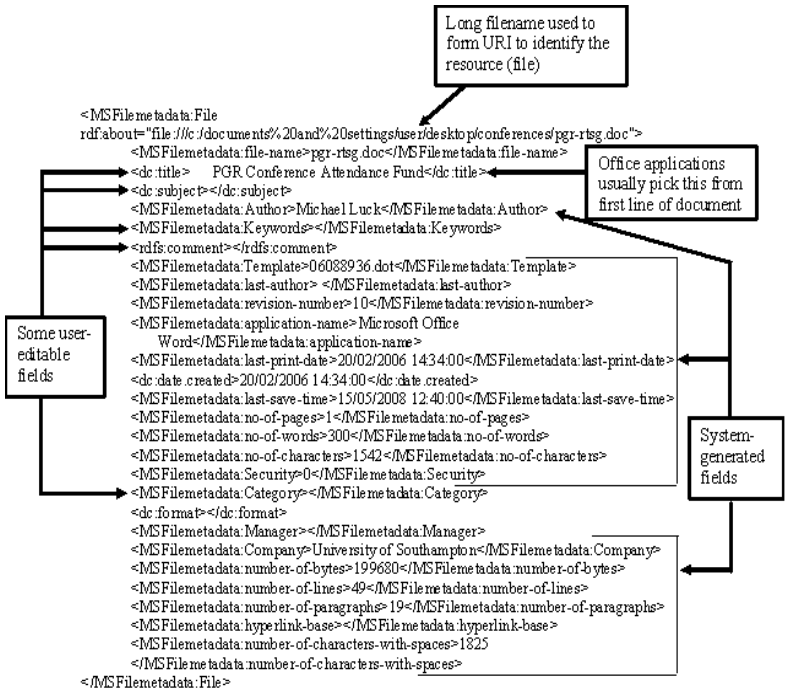
In hypertext and hyperindexing indexes serve as aids that facilitate the location of objects [8]. If ordered, index entries can facilitate the quick location of relevant entries. The semantic metadata is processed to build for each document a forward index by deriving terms from the metadata values which then serve as index terms or keywords in document recognition, retrieval and clustering. The terms are actually any text fragments which are recognized by the algorithm as single words but might only make sense to the creator as they

may be “codes” used, for example, to describe files and folders. The algorithm for building the indexes for each document or file is as follows.

First the metadata is queried for all property values relevant to a given document. The extracted property values are all treated the same irrespective of property value. While being extracted some pre-processing is carried out to select useful terms in the multi-valued attribute values. This is done by removing stopwords (for example words like “and”, “the”, “is”), punctuation (replace with spaces to separate terms) and converting all the values to low case to facilitate easy comparison. Stemming, which is usually done in indexing to eliminate variation caused by presence of different grammatical forms of the same word (for example, “research” and “researched”), is not carried to avoid intervening with user's intended meaning of concepts. This is mainly for purposes of presentation of these for view by the user.

The string value is then divided into terms according to spaces. The derived terms are stored as a document index set: Doc Uri → {t1, t2, ...tn}.

All the documents' forward indexes are then used to compile a unified metadata index for the whole document set after parsing and term recognition. The indexes are further enhanced by lookup, selection then addition of synonyms from WordNet, large-scale English lexical database developed at Princeton University [9]. This is done by matching then grouping semantically related words that are


Long filename used to form URI to identify the resource (file)
Office applications usually pick this from first line of document
Some user-editable fields
System-generated fields

```
<MSFilemetadata:File
rdf:about="file:///c:/documents%20and%20settings/user/desktop/conferences/pgr-rtsg.doc">
        <MSFilemetadata:file-name>pgr-rtsg.doc</MSFilemetadata:file-name>
        <dc:title>    PGR Conference Attendance Fund</dc:title>
        <dc:subject></dc:subject>
        <MSFilemetadata:Author>Michael Luck</MSFilemetadata:Author>
        <MSFilemetadata:Keywords></MSFilemetadata:Keywords>
        <rdfs:comment></rdfs:comment>
        <MSFilemetadata:Template>06088936.dot</MSFilemetadata:Template>
        <MSFilemetadata:last-author> </MSFilemetadata:last-author>
        <MSFilemetadata:revision-number>10</MSFilemetadata:revision-number>
        <MSFilemetadata:application-name>Microsoft Office
            Word</MSFilemetadata:application-name>
        <MSFilemetadata:last-print-date>20/02/2006 14:34:00</MSFilemetadata:last-print-date>
        <dc:date.created>20/02/2006 14:34:00</dc:date.created>
        <MSFilemetadata:last-save-time>15/05/2008 12:40:00</MSFilemetadata:last-save-time>
        <MSFilemetadata:no-of-pages>1</MSFilemetadata:no-of-pages>
        <MSFilemetadata:no-of-words>300</MSFilemetadata:no-of-words>
        <MSFilemetadata:no-of-characters>1542</MSFilemetadata:no-of-characters>
        <MSFilemetadata:Security>0</MSFilemetadata:Security>
        <MSFilemetadata:Category></MSFilemetadata:Category>
        <dc:format></dc:format>
        <MSFilemetadata:Manager></MSFilemetadata:Manager>
        <MSFilemetadata:Company>University of Southampton</MSFilemetadata:Company>
        <MSFilemetadata:number-of-bytes>199680</MSFilemetadata:number-of-bytes>
        <MSFilemetadata:number-of-lines>49</MSFilemetadata:number-of-lines>
        <MSFilemetadata:number-of-paragraphs>19</MSFilemetadata:number-of-paragraphs>
        <MSFilemetadata:hyperlink-base></MSFilemetadata:hyperlink-base>
        <MSFilemetadata:number-of-characters-with-spaces>1825
        </MSFilemetadata:number-of-characters-with-spaces>
</MSFilemetadata:File>
```

Fig. 2 RDF Description for a Word document.

already in the metadata using WordNet synsets, thereby implicitly clustering documents with similar keywords. The process is shown in Figure 3.

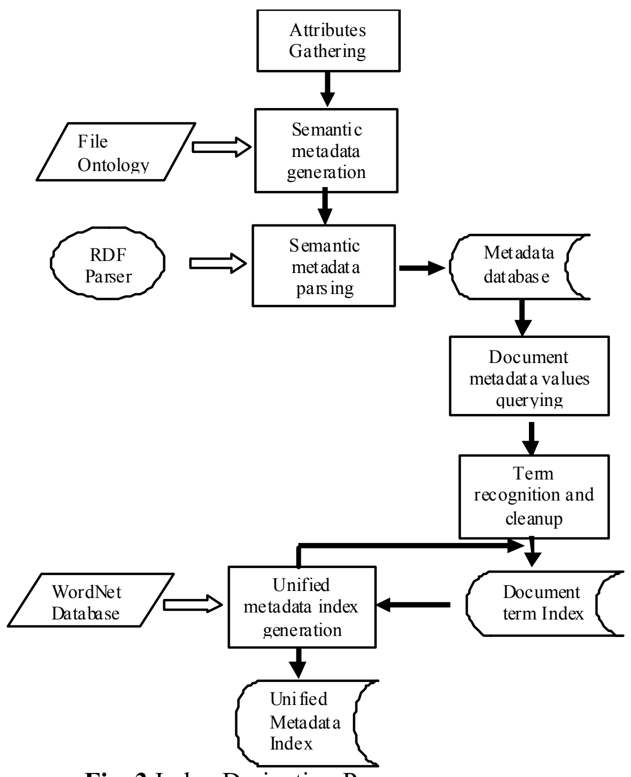

Attributes Gathering
File Ontology
Semantic metadata generation
RDF Parser
Semantic metadata parsing
Metadata database
Document metadata values querying
Term recognition and cleanup
WordNet Database
Unified metadata index generation
Document term Index
Unified Metadata Index

Fig. 3 Index Derivation Process

4.3 Term-based Documents' Similarity

Retrieval and hypermedia systems employ some kind of similarity measurement to determine and present or integrate related items together. Document similarity has been measured especially using the vector space model, a method makes use of weighted sets of terms for documents to compute their similarity. Because the terms we extracted include those from the hierarchy structure where the folder names form a structure showing relations by location between the documents, distance-based semantic relatedness measures [30] may be applicable, even if only as part of the whole solution. These are based on counting the number of edges (which can also be assigned a weight based on depth or density of a node or the type or strength of a link) between the concepts, with a shorter distance signifying more relatedness.

The methods were not used for in research since they are already covered by considering the commonality of terms in the above approach (documents in the same directory are more likely to share more terms), and the fact that the mental model behind assignment of documents to folders has not been well studied and is not clearly defined enough to warrant assignment of weights to the terms themselves.

A document-document similarity matrix is then built after the index is created based on pair-wise comparisons of terms in documents' forward indexes. Each row (document) is then extracted and ordered by cell value on a similarity request (selecting a document thereby requesting more details) on the interface during browsing. Given D_x as the set of terms for document x, and D_y as those for document y, $S(D_x, D_y)$ represents similarity between document *x* and document *y*. The whole matrix is initialized to zero before the comparison starts.

$$D_x = \{T_{x1}, T_{x2}, \ldots T_{xn}\}$$
$$D_y = \{T_{y1}, T_{y2}, \ldots T_{ym}\}$$
$$\forall i, \forall j, \text{ if } T_{xi} \text{ like } T_{yj} \text{ then } S(D_x, D_y) = S(D_x, D_y) + 1$$

That is, $S(Dx, Dy) = (Dx \cap Dy) = \sum_{i=1}^{n} \sum_{j=1}^{m} (Txi \approx Tyj)$ (1)

The comparison is described as likeness or approximation of similarity (rather than equality) between terms since partial matching based on stemming and synonymy is taken into account. The partial matching is also a result of the data cleanup that was done during term extraction (removal of punctuation, stopwords and conversion to lowercase). With this approach document D_i is more similar to document D_j than to document D_k if $S(D_i,D_j) > (D_i,D_k)$, that is, more common terms between D_i and D_j than D_i and Dk.

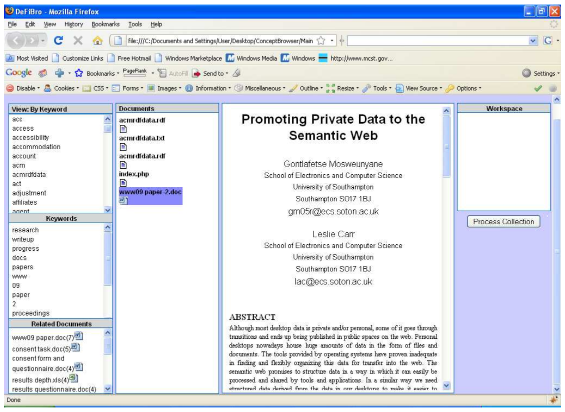
5 THE DESKTOP FILE BROWSER

The model is then used as an underlying organization to develop an interface for supporting retrieval and discovery of documents. The interface implemented, the Desktop File Browser (DeFiBro) offers a browsing interface for retrieval of desktop documents. The following are a description of important aspects of the interface shown in Figure 4.

- Overview - provision of some sort of overview over a document collection is essential. Facets have already been seen to provide good overviews [10, 15]. The interface implemented provides two kinds of overviews: an alphabetic and numeric index of document names, and overlapping facets in the form of authors, organisations and keywords (terms) derived from all the metadata values.
- Context - Initially the context of documents in the filesystem was the hierarchy structure itself. In our solution the folder structure have been “flattened”, the layout and positioning of documents in the structure has been relinquished in favor of a flatter and more visible, linked keyword structure. Within the facets provided the user can select items of interest to view clusters of documents based on the facet values. Context for a selected document is provided by related details consisting of linked metadata terms (to get the documents to select one another based on similarity of terms) and similar (associated) documents to the selection. In addition to providing context, linked metadata and similar documents add additional dimensions for navigation within the facet-based overview.
- Details - File thumbnail representations have been found to be effective for locating and organizing documents in user interface studies dealing with documents [11, 12, 13 and 14). At a glimpse the user can see whether a document is a picture, table or plain text and can quickly decide whether there is need to investigate further by opening the document or whether to check other possible links. They can be especially helpful if the contents are clearly visible and the text is readable as is provided in the implemented interface. A preview of the first page of the document in the form of an enlarged thumbnail is

given on hovering the mouse pointer over a listed document thumbnail.

- Extract and Transfer - A `workspace' panel is provided to enable selection and extraction of desired documents with their metadata. The feature is implemented as a solution to user needs involving difficulty of working with documents across several folders and to address the need for creating desired groupings for immediate work purposes that might not be satisfied by the similarity clustering implemented.


DeFiBro - Mozilla Firefox
File Edit View History Bookmarks Tools Help
file:///C:/Documents and Settings/User/Desktop/ConceptBrowser/Main
Most Visited Customize Links Free Hotmail Windows Marketplace Windows Media Windows http://www.mcst.gov...
Google Bookmarks PageRank AutoFill Send to Settings
Disable Cookies CSS Forms Images Information Miscellaneous Outline Resize Tools View Source Options
View: By Keyword
acc
access
accessibility
accommodation
account
acm
acmrdfdata
act
adjustment
affiliates
agent
Keywords
research
writeup
progress
docs
papers
www
09
paper
2
proceedings
Related Documents
www09 paper.doc(7)
consent task.doc(5)
consent form and questionnaire.doc(4)
results depth.xls(4)
results questionnaire.doc(4)
Documents
acmrdfdata.rdf
acmrdfdata.txt
acmrdfdata.rdf
index.php
www09 paper-2.doc
Workspace
Process Collection
Done

Promoting Private Data to the Semantic Web

Gontlafetse Mosweunyane
School of Electronics and Computer Science
University of Southampton
Southampton SO17 1BJ
gm05r@ecs.soton.ac.uk

Leslie Carr
School of Electronics and Computer Science
University of Southampton
Southampton SO17 1BJ
lac@ecs.soton.ac.uk

ABSTRACT
Although most desktop data is private and/or personal, some of it goes through transitions and ends up being published in public spaces on the web. Personal desktops nowadays house huge amounts of data in the form of files and documents. The tools provided by operating systems have proven inadequate in finding and flexibly organizing this data for transfer into the web. The semantic web promises to structure data in a way in which it can easily be processed and shared by tools and applications. In a similar way we need structured data derived from the data in our desktops to make it easier to

Fig. 4 Selecting a document shows linked keywords, a preview and related documents

The user selects a facet to browse (Author, Keyword, Organisation) and the values of the selected facet are shown in top-most left-hand side panel, in figure 4.When an item in this panel is selected a list of documents matching the criteria satisfied by the selected value is provided in the *Documents* panel (second column from left). Browsing by filename presents a submenu where the first character of the file name can be selected and the corresponding filenames shown in the *Documents* panel. Pointing to a document name or associated icon reveals its preview in the *preview* panel and selecting it reveals the keywords extracted from its metadata in the keywords panel and *related* documents are shown in the panel below that. These keywords are linked, that is, one can select the keyword to display documents whose criteria it satisfies. While browsing the user can collect documents of interest into a collection "basket", the *workspace* panel. Documents in this panel can be previewed and selected just as documents in the Documents and *Related documents* panels. Documents can also be opened in their respective applications by simply double-clicking on the document as in the desktop.

6 EVALUATION – HIERARCHY vs LINK-BASED NAVIGATION

The implemented system aimed to demonstrate the importance of metadata and flattening of hierarchies in assisting users to locate, associate and discover documents while browsing their personal file hierarchies. To verify that users can indeed benefit from these, we perform an experiment that involves hands-on session tasks with users involving locating documents. To demonstrate the need to apply these concepts to improve operating system-assisted browsing of document hierarchies created by users on the desktop, we base these tasks on two environments. These are the Windows visualization method and our implementation, DeFiBro. With the Windows visualization method there are two ways: Windows Explorer or the simple zoomable visualization interface that involves clicking on folders to expand and view files and folders contained therein. Both have been found to have similar performance in locating tasks involving either familiar or unfamiliar hierarchies [22]. This is done in order to compare the two and to clarify the benefits of our approach through our implementation, while at the same time advocating for improvement of browsing documents on the desktop. At the end of the task users were asked to answer a few questions based on a five-point Likert scale and qualitative questions about how the two systems perform in comparison to each other.

Since the research is mainly concerned with the browsing or navigation of personal information structures, in particular desktop document hierarchies, during knowledge tasks for retrieval of information sources (documents). Kelly [27] recommends using naturalistic approaches that allow people to perform Personal Information Management (PIM) behaviors in familiar environments with familiar tools and collections. The document browser implemented would have to be therefore evaluated within the context of personal document hierarchies in settings that match as much as possible real user settings and tasks. Owing to the difficulty of using the user's hierarchy (confidentiality issues and difficulty of coming up with a task from personal information) a test file hierarchy (one of the authors' own) was used. Users were allowed to familiarize themselves with the hierarchy before the commencement of the tasks. The tasks involved locating documents in the given test hierarchy using DeFiBro and Windows Visualization alternatively, given either the file name or a description of the document(s). The documents were located at the most popular levels where documents are stored, as established by a previous user study, of 2, 3 and 4.

The results of the times the test users took to complete each task in the two environments were recorded in seconds and the averages computed. The summary of the results is shown in figure 5.

Users reported the positive aspects of DeFiBro as being user friendly, offering of browsing by association and "searching" without providing search terms as well as helping out in a "badly organized" situation and providing features that enable more effective browsing like previews.

Suggestions for improvement by test users were dominated by the need to quickly move within the results (documents found by browsing, keywords, organizations, author) by automatic selection/shortcuts based on keyboard characters. Other suggestions were for the cosmetic appearance of interface to be improved through conversion of the interface to a more graphic one, showing attribute-value pairs instead of only attributes and providing other attributes to locate files such as date.

About half of the users indicated that they would have like the system to somehow be connected to the file hierarchy to provide for a way to get to the original location of the file. One user commented "...because in the first place I was trying to locate the file" to emphasize the need for such a system to be intertwined with the original way the files were organized and possibly the sense of attachment the users have with the way their "organized" file structure.

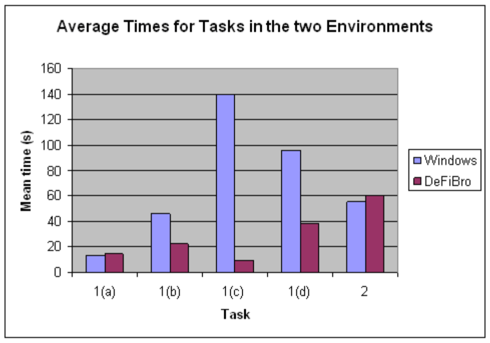


Fig. 5 Average Times for Windows Visualization and DeFiBro for all tasks

7 CONCLUSIONS and FUTURE WORK

The implementation utilizes terms that represent document identity and context and might be useful for recognizing and retrieving documents. The interlinked nature of this information is presented to end users to utilize in browsing documents. The browsing structure provided is expected to integrate data (documents) in a different manner to what the users are used to, and provide an interface which will expose data which will have otherwise been "hidden" in a person's data collection.

The navigation structure provided through the interface is evaluated against system-provided navigation of the file system on the Microsoft Windows system using locating and associating tasks. The efficiency of the two environments is measured by its performance in relation to the users' speed and effort expended in locating the documents in the tasks given. Time taken to locate documents in the given tasks is used as a measure of these attributes. The results show that DeFiBro performs reasonably better than Windows file browsing for tasks involving locating files at deeper levels (levels 3 and 4 in the hierarchy) than at shallow levels (levels 1 and 2). The approach adopted for the evaluation has mainly been concerned with whether and how the system meets the user's needs, which is the approach usually adopted in information seeking evaluations [28]. Although time was used as the basis for evaluation, it may not accurately reflect the actual benefit of the exploratory interface, as it has been discovered before that a longer time might indicate more beneficial browsing in terms of discovering relevant material [29]. An extended user evaluation with the user's hierarchy and follow-up over a longer time might then be more appropriate. In addition other evaluation factors such as simplicity and efficiency of the Windows system and the implemented system have to be considered to make a fully informed decision on which system is better.

Other further work include use of indexes already created by operating systems and their applications, use of extended attributes and use of the indexes to refine Google results.

6 REFERENCES

1. Edmunds, A., Morris, A.: The Problem of Information Overload in Business Organizations: A review of the Literature. Intl. J. of Info Mangt, 20 (1), 17--28 (2000)
2. Dong, X., Halevy, A.: A platform for Personal Information Management and Integration. Conference on Innovative Data Systems, January 4-7, pp. 119-130 (2005)
3. Kelly, L.: Context and Linking in Retrieval from Personal Digital Archives. In: ACM SIGIR Conference on Research and Development in IR, pp. 899--899 (2008)
4. Gemmell, J., Bell, G., Lueder, R., Drucker S., Wong, C.: MyLifeBits: Fulfilling the Memex Vision. ACM Multimedia 02, Juan Lens Pins, France, December 1-6 (2002)

5. Sauermann L., Grimmes G.A, Kiesel M., Fluit C., Maus H., Heim D., Nadeem D., Horak B., Dengel A.: Semantic desktop 2.0: The Gnowsis Experience. In: 5th International Semantic Web Conference, Athens, GA, USA (2006).
6. Sauermann L., Schwarz S.: Introducing the Gnowsis Semantic Desktop. In: International Semantic Web Conference (2004).
7. Henderson, S.: How do People Organize their Desktops? CHI '04 Extended Abstracts on Human Factors in Computing Systems, pp. 1047 -- 1048 (2004)
8. Bruza, P. van der Weide, T.: Two Level Hypermedia: An Improved Architecture for Hypertext. In: Tjoa, A., Wagner, R. (eds.) Proc. Database and Expert System Applications Conference., pp. 76--83. Springer-Verlag (1990)
9. Miller, G. A., Beckwith, R., Fellbaum, C., Gross, D., Miller, K. J.: Intro to WordNet: An On-line Lexical Database. Intl. J. of Lexicography, 3(4), 235--244 (1990)
10. Kobilarov, G., Dickinson, I.: Humboldt: Exploring Linked Data. In: Workshop (Linked Data on the Web 2008) at WWW2008, April (2008)
11. Faichney, J., Gonzalez, R.: Goldleaf Hierarchical Document Browser. In: User Interface Conference Proceedings. Second Australasian, pp. 13--20 (2001)
12. Lucas, P., Schneider, L.: Workscape: A Scriptable Document Management Environment. In: ACM CH1'94 Conference Companion (1994).
13. Mander R., Salomon G., Wong, Y. Y.: A 'Pile' Metaphor for Supporting Casual Organization of Information. In: ACM CHI '92, pp. 627--634 (1992)
14. Robertson, G., Czerwinski, M., Larson, K., Robbins, D.C., Thiel, D., van Dantzich, M.: Data Mountain: Using Spatial Memory for Document Management. In: ACM UIST'98, pp. 153--162 (1998)
15. Henderson, S.: Genre, Task, Topic and Time: Facets of Personal Digital Document Management. In: Proceedings of the 6th ACM SIGCHI New Zealand Chapter's International Conference on Computer-Human Interaction: Making CHI Natural CHINZ '05, Auckland, New Zealand, pp. 75--82 (2005)
16. Ravasio P., Schar G., Krueger H.: In Pursuit of Desktop Evolution: User Problems and Practices with Modern Desktop Systems. In: ACM Transactions on Computer-Human Interaction, 11(2): pp.156—180 (2004).
17. Kaptelinin V., Czerwinski M. In: Czerwinski M., and Kaptelinin V. (eds), Beyond the Desktop Metaphor: Designing Integrated Digital Work Environments, chapter - Introduction: The Desktop Metaphor and New Uses of Technology, pp. 19-48. MIT Press, (2007).
18. Henderson S.: How Do People Organize their Desktops? In: CHI '04 Extended Abstracts on Human Factors in Computing Systems, pp 1047--1048, (2004).
19. Freeman E., Gelernter D.: Czerwinski M., and Kaptelinin V. (eds), Beyond the Desktop Metaphor: Designing Integrated Digital Work Environments, chapter Beyond Lifestreams: The Inevitable Demise of the Desktop Metaphor, pp 19--48. MIT Press (2007).
20. MINDSWAP Research Group, SWOOP - A Hypermedia-Based Featherweight OWL Ontology Editor. http://www.mindswap.org/2004/SWOOP/.
21. Berners-Lee T., Masinter L., McCahill M., (eds).: Uniform Resource Locators. http://www.ietf.org/rfc/rfc1738.txt.
22. Golemati M., Katifori A., Giannopoulou E. G., Daradimos I., Vassilakis C.: Evaluating the Significance of the Windows Explorer Visualization in Personal Information Management Browsing Tasks. In: 11th International Conference Information Visualization, pp. 93--100 (2007).
23. Dong X., Halevy A. Y., Nemes E., Sigundsson S.B., Domingos P. SEMEX: Toward On-the-Fly Personal Information Integration. In: Workshop on Information Integration on the Web (2004).
24. Cai Y.,Dong X. L., Halevy A, Liu J.M. , Madhavan J.: Personal Information Management with SEMEX. In: SIGMOD 2005, pp.921--923 (2005).
25. Karger D. R. Jones W.: Data Unification in Personal Information Management. Communications of the ACM, 49(1): 77--82, (2006).
26. Karger D. R., Bakshi K., Huynh D., Quan D., Sinha V.: Haystack: A Customisable General-Purpose Information Management Tool for End Users of Semistructured Data. In: Proceedings of the 2003 CIDR Conference (2003).
27. Kelly, D.: Evaluating Personal Information Management Behaviors and Tools. Communications of the ACM, 49(1):pp. 84--86 (2006).
28. Kules W., Wilson M. L., schraefel m.c., Shneiderman B.. From Keyword Search to Exploration: How Result Visualization Aids Discovery on the Web. Technical Report 1516920080208, School of Electronics and Computer Science, University of Southampton (2008).

29. Capra, R., Marchionini Oh, G. J., Stutzman F., Zhang Y.: Effects of Structure and Interaction Style on Distinct Search Tasks. In Proceedings of the 7th ACM/IEEE-CS Joint Conference on Digital Libraries, pp. 442-- 451 (2007).
30. Cross, V. Tversky's Parameterized Similarity Ratio Model: A Basis for Semantic Relatedness. In Fuzzy Information Processing Society, NAFIPS 2006. Annual Meeting of the North American, pp. 541--546 (2006).
31. Stasko, J., Catrambone, R., Guzdial M., and Mcdonald K.: An Evaluation of Space-Filling Information Visualizations for Depicting Hierarchical Structures. International Journal of Human-Computer Studies, 5, 663--694, (2000).
32. Johnson, B. TreeViz: Treemap Visualization of Hierarchically Structured Information. In: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, pp. 369 -- 370, (1992).
33. Wilson, R.M., Bergeron, R.D.. Dynamic Hierarchy Specification and Visualization. In: Information Visualization, 1999. (Info Vis '99) Proceedings. IEEE Symposium on (1999).
34. Plaisant, C., Grosjean, J., Bederson, B.B.:. SpaceTree: Supporting Exploration in Large Node Link Tree, Design Evolution and Empirical Evaluation. In: Proceedings of the IEEE Symposium on Information Visualization 2002 (InfoVis02), pp. 57—64 (2002).