Towards Semantic File System Interfaces

Sebastian Faubel
Student of Computer Science
Georg-Simon-Ohm Hochschule
Nuremberg, Germany
sfaubel@users.sf.net

Christian Kuschel
Student of Computer Science
Georg-Simon-Ohm Hochschule
Nuremberg, Germany
ckuschel@users.sf.net

ABSTRACT

In this paper, we present our hypothesis that transition to semantic file system interfaces is possible by computing the organization of hierarchical file systems from semantic web data.

Categories and Subject Descriptors

D.4.3 [**Operating Systems**]: File systems management: file organization, directory structures, access methods

General Terms

Algorithms, Management, Experimentation, Human Factors

Keywords

Semantic Web, Semantic Classification, Path Projection

1. INTRODUCTION

Hierarchical file system user interfaces usually provide a static directory structure with limited features for information management. Conventional file system interfaces require users to interact with a hierarchical classification of their files (taxonomy). In this sense, the user of a hierarchical file system is confronted with most of the disciplines in knowledge management, which includes creating and maintaining a *consequent* vocabulary and naming conventions in order to be able to remember *where* to look for files in the hierarchy. This is opposed to the natural approach of memory, knowing *what* to look for. As of publication date, there have been little to no efforts made by vendors of operating systems to integrate knowledge management technologies into modern desktop environments which leaves the status quo of file management where it was over 25 years ago.

Desktop search engines create an index of the data stored in a hierarchical file system, optimized for keyword lookup. This provides fast, associative access to files and their information by allowing the user to formulate queries. However, searching cannot support the creation and management of file system taxonomies. Moreover, indexing can only take place after changes have been made to a file. Therefore, a user is still dependent on information management primitives [3] provided by the hierarchical file system and proprietary file management software.

Semantic web technologies [2] are designed to provide features for metadata storage, resource description and knowledge management on a global scale. They are suitable for modeling and querying the contents of one or many hierarchical file systems. It is therefore possible to use semantic web technology as an open, universal file system index. This offers a solution [1] to the mutually incompatible metadata formats produced by proprietary file management products.

Semantic web ontologies can be used to describe and infer on relations of files, ranging from specialized file management domains up to hierarchical classifications. Furthermore, these ontologies can be used to describe patterns on how to construct file system paths out of RDF data. This allows replacing the hierarchical file system as a *primary* user interface for file management. For example, semantic file selector dialogs could determine which classifications are suitable for a given file type and provide domain specific input controls.

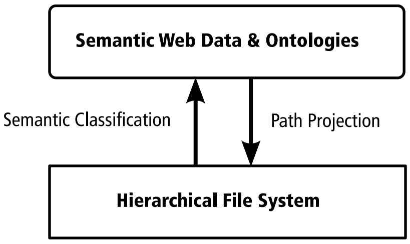

Figure 1: Abstraction layers and transitions

In the following sections, we introduce two key technologies of our ongoing research projects (see Fig. 1). The first section deals with efficient classification of existing file system contents in order to support the transition to semantic file system interfaces. The second section focuses on file system path construction using semantic web ontologies and RDF data.

2. SEMANTIC CLASSIFICATION

Firstly, the new term *file class* should be illustrated: A file class refers to what a user would call 'the content' of a file, e.g. a picture or a document. Moreover, agglomerations of files can also be considered one file class, e.g. a programming project, which consists of source code and documentation. So, file classes exist, which are composed of other file classes. Composed file classes often contain files which refer to other files via relative paths. It is essential to not change its path structure, because every change could affect referential integrity.

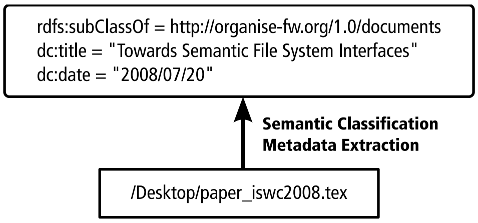
rdfs:subClassOf = http://organise-fw.org/1.0/documents
dc:title = "Towards Semantic File System Interfaces"
dc:date = "2008/07/20"
Semantic Classification
Metadata Extraction
/Desktop/paper_iswc2008.tex

Figure 2: Semantic classification

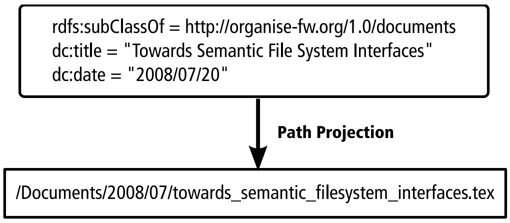
rdfs:subClassOf = http://organise-fw.org/1.0/documents
dc:title = "Towards Semantic File System Interfaces"
dc:date = "2008/07/20"
Path Projection
/Documents/2008/07/towards_semantic_filesystem_interfaces.tex

Figure 3: Path projection

Secondly, the description of file classes are considered. When trying to describe file classes, the natural approach is to use logical if-then clauses. Since these clauses have to be created by the user, it is appropriate to specify them in a legible format. Furthermore, this specification should also allow computational deduction. Thus ontologies can be considered very similar to the aforementioned requirements. Consequently, only ontologies are required to perform semantic classification.

The task of a classification, in this context, is to assign each file to a class of a given set of classes. Suppose an index of a hierarchical file system, e.g. the file system translated into an RDF store and metadata is added to each file (see Fig. 2). To assign a file to a class, only the vocabulary used to describe this file needs to be considered. Since every distinct file type uses its own vocabulary, it is possible to infer to an appropriate file class from the vocabulary used.

Applying the path projection algorithm (see following section) would result in rearranging the entire file system according to the given classes. This is not always desired as there might be relations between files in the hierarchy which must not be broken. Therefore, we propose the semantic classification which also takes the file system structure into account, in addition to the classification mentioned above.

The semantic classification assigns each folder and file to at least one file class. Since some file classes consist of other file classes, it is mandatory that some folders / files are assigned to multiple file classes. In addition, multiple file classes enable a more flexible view on the data stored. The obtained information is saved in the RDF store and vital file system structures are preserved.

3. PATH PROJECTION

Algorithmic computation of file system paths is a key technology of semantic file system managers. It enables the complete abstraction from the hierarchical file system while maintaining backward compatibility. Projection of a user's semantic organization allows one to recognize the accustomed taxonomy when directly interacting with hierarchical file systems.

The primary goal of our project is to develop a projection algorithm which works with standard semantic web ontologies as input source. The intention is to allow the formulation of projection patterns by ontology creators, as well as end-users. Since ontology authors usually are experts within their domain, they can formulate different projection patterns which are optimized for different applications.

When using a specific projection pattern, it needs to be ensured that all required information is available for path construction. Semantic web ontologies feature mechanisms to test the availability of certain predicates for a given knowledge domain. Furthermore, they allow for the determination of patterns that suffice for a given set of attributes. Hence, providing or computing a required set of attributes enables file system organization to be done *consequently* by an algorithm (see Fig. 3).

The proposed approach allows generic file system interfaces to be built that primarily confront a user with the *semantic aspects* of their files opposed to *coding locations* using directories and file names. Moreover, a user can actively be supported in the specification of a file's attributes by using metadata extraction techniques. With path projection, the compatibility to hierarchical file system interfaces is not only preserved, but supported.

4. CONCLUSION

We have conducted experiments to prove the feasibility of the presented architecture. The semantic classification as well as the path projection have been efficiently implemented as proof of concept [4].

Using semantic web technologies for hierarchical file system abstraction could have considerable advantages to the usability of file system interfaces, interoperability of applications and portability of files. Since most of the applications in use today produce metadata, further research in this area is strongly recommended.

5. ACKNOWLEDGMENTS

Our thanks to Prof. Dr. Thomas Fuhr and Prof. Dr. Thomas Huke for supporting our student projects and the writing of this paper. Thanks to Elizabeth Ströhlein for proofreading.

6. REFERENCES

[1] NEPOMUK – The Social Semantic Desktop, 2008. http://nepomuk.semanticdesktop.org.

[2] Grigoris Antoniou and Frank van Harmelen. *A Semantic Web Primer*. The MIT Press, 2008.

[3] C. Mic Bowman and Chanda Dharap et al. A File System for Information Management. In *Proceedings of the ISMM International Conference on Intelligent Information Management Systems*, March 1994.

[4] Sebastian Faubel and Christian Kuschel et al. Organise Framework. http://www.organise-fw.org.