

TOWARD ONTOLOGY-BASED FILE SYSTEMS

**Ba-Hung NGO⁺, Christian BAC*,
Quyêt-Thang LE⁺, Frédérique SILBER-CHAUSSUMIER ***

** Institut National d es Télécommunications
9 Charles Fourier, 91011 Evry Cedex - France*

*email {Hung.Ngo_Ba, Christian.Bac, Frederique.Silber-Chaussumier}@int-
evry.fr*

and

*+ College of Information Technology - Cantho University
01 Ly Tu Trong, Can Tho, Viet Nam*

email:lqthang@cit.ctu.edu.vn

Abstract:

Semantic file systems enhance standard file systems with ability of file searching based on file semantics. In this paper, we propose to integrate support for ontologies into a file system to build efficient semantic file systems whose file semantics can be shared by users, applications and semantic file systems themselves. We call it ontology-based file system. We identify three existent types of file semantics: property-based semantics, content-based semantics and context-based semantics and adopt multi-ontology layer approach to share them between agents. The challenges on ontology-based file system are also discussed.

Keywords: File System, Semantics, metadata, Ontology

1.INTRODUCTION

Current file systems use *files* as information storage units and let users organize them in a *directory* hierarchy. The information stored in a file system may be accessed by browsing directories or by giving an accurate *path name* to the file. In both cases, the user has to remember information about the location of his file. This organization doesn't scale well to satisfy user's current needs because of the explosion of the amount of information, the multiple sources and types of the information that one has to manage. Text-based search tools, such as *grep* in *UNIX* operating system, scan file content for a given term and return all the hit files. These tools take much time so others like *Glimpse* [1], *Google Desktop* [2] and *Beagle* [3] index for content to speed up searching process. However, user usually has to navigate in a numerous number of hit files to find out the interesting ones.

The abstract goal of a file system is to store, retrieve, locate, and manipulate information [4]. Recent research on *semantic file systems* (SFS) deals with integrating the *semantics* into file system to enhance it with the ability to locate information by *file semantics*. File searching based on file

Toward ontology-based file systems, Dalat 2006

semantics is called *semantic-based searching*. In existing SFS, the applications, users and SFS, called *agents*, don't share file semantics: each has its own. The same term will be understood by different agents with different meanings. On the contrary, two different terms will represent the same meanings. This prevents from sharing file semantics between different agents. To solve this, we propose to integrate support for ontologies into a file system. This support would help build efficient SFS sharing file semantics in all agents.

This paper discusses file semantics in section 2, gives a short review of the file systems that belong to the trend of SFS in section 3, argues the need of integrating ontology into a file system and proposes an ontology-based file system model in section 4. It then discusses the challenges on ontology-based file system design in section 5. Some related works are presented in section 6. The last section is our conclusion and future works.

2.FILE SEMANTICS

When talking about SFS, the frequently asked question is “what is file semantics?” Most SFS use metadata to represent file semantics. In general, metadata is defined as data about data. In information searching context, the definition of Dempsey and Heery [5] about metadata is more precise: “metadata is data associated with objects which relieves their potential users of having to have full advance knowledge of their existence or characteristics. A user might be a program or a person”. In SFS, metadata generally represent three levels of file semantics: property-based semantics, content-based semantics and context-based semantics.

Property-based semantics is general, content and context independent information related to file, such as owner, creation date, last modified date, file type, directory, name,... SFS create property-based semantics for all files and share it for all applications and users.

Content-based semantics is information getting by analysing the file content. Content-based semantics usually represents internal organization of file content: roles played by pieces of content data. For example, a piece of content data plays the role of *title*, others play the roles of *headings* and *paragraphs*. Metadata whose syntax and semantics are defined by an application is usually used to represent valid roles in a file structure.

Context-based semantics is the information about the interrelated conditions in which a file exists. Context-based semantics expresses relationships of a file with other subjects: files, concepts, persons... For instance, context-based semantics created by a mail user agent can express that file *rapport.pdf* is an *article* talking about *semantic file system*, attached in an *email* sent by *Mr. BAC*. In this example, *article*, *semantic file system* and *email* are concepts ; *Mr. BAC* is a person.

The above analysis shows that the existence of file semantics is determined by an agent. So we use the notion *File semantic conception* to

Toward ontology-based file systems, Dalat 2006

specify file semantics that will be created by an agent. To provide users with semantic-based searching, SFS use *semantics generator* module to harvest file semantics created by all agents and then indexed them by an *index engine*.

3.SURVEY OF SEMANTIC FILE SYSTEMS

This section introduces some existing file systems and focuses on the evolution of their file semantics.

MIT Semantic File System (MIT-SFS) [6] is the vanguard in the trend of SFS. MIT-SFS supports both property-based semantics and content-based semantics. File semantics is properties such as *author, owner, date, type, subject, title, directory, name, category...* Each property is represented by an *attribute* under the form of a pair *name-value* where the name exactly identifies a file property. MIT-SFS uses one specified-type file semantics generator for each file type. Content-based semantics for a new file type can be extracted from file content by adding a new specified-type file semantics generator into the table that maintains the relations between specified-type file semantics generator and file type. In MIT-SFS, using content-based semantics one can find out not only files but also entities in file content (i.e. a procedure of C program). Nebula [7], object-oriented SFS supports property-based semantics. Nebula *file object* consists of a set of attributes that represents properties of files. A file object has to belong to a *file object type* that specifies the domain of acceptable attributes for a specific file type. A file object type is an interesting example about file semantic conception.

Sedar [8] also supports content-based semantics. Sedar uses *semantic vector* extracted from file content to identify a file. Sedar deals with locating the appropriate version of files in a deep archival file system. Rather than storing the semantic vector for each file version, it uses a *representative semantic vector* for several files having close semantic vectors.

pStore [9] is a platform that uses context-based semantics. pStore proposes five types of relations: *file versioning, hierarchical name space, associative semantics and context information*. pStore uses RDF to store context-based semantics. Spotlight [10] supports a little context-based searching: a set of related files are marked by a particular keyword to easily find them. M. Fisher and A. Sheth [11] uses context-based semantics to make annotations on the entities (words) of document contents in their content management system. Chirita *et al.* [12] use context-based semantics to store files and their context in emails, file hierarchies and browser caches applications. Another example on context-based semantics is *Connections* [13]. This system combines text-based searching with context information gathered from user activity. For *Connections*, the file's context includes other concurrently accessed files, the user's current task, any action or data that the user associates with the file's use. *Connections* maintains a separate *relation-graph* for each user based on their file

Toward ontology-based file systems, Dalat 2006

accesses and uses information to rank the text-based searching result. Logic File System (LFS) [14] distinguishes two property types: *intrinsic property* and *extrinsic property*. While intrinsic property belongs to property-based semantics and content-based semantics, extrinsic property belongs to context-based semantics. Extrinsic properties may be the concepts such as *music, pop, excellent* that a user can assign to a file. It is possible to define axioms between extrinsic properties to form a taxonomy. LFS can create categories in different domains as provided by ontologies. The way that LFS uses axioms as inference rules is similar the way search engines infer based on ontology rules.

Information interesting for a person, called *personal information* is created and stored by different applications. But, they have relationship in certain context. So *semantic desktop* researches want to relate personal information by context to provide user with context-based information searching. Xiao and Cruz [15] proposes a multi-ontology approach for personal information management. In this approach, *application ontologies* are used to specify person information created by an application. Generally, an application can concern many knowledge domains and a knowledge domains may be shared by many applications. Thus, the platform uses *domain ontology* to specify terms and concepts for each knowledge domain. These domain ontologies are then used to defined application ontologies. Through shared domain ontologies, personal information from different applications relates each other so that context-based personal information searching is also supported.

4. INTEGRATING ONTOLOGY INTO FILE SYSTEMS

The section 2 has showed that each agent has it owns file semantic conception determining file semantics that it will create. File semantics created by an application can be harvested by a SFS to provide users and other applications with semantic-based searching if they understand the corresponding file semantic conception. It means that file semantic conceptions should be shared among agents in a SFS. We are interested in using ontology to specify file semantic conception.

Ontology is a representation of a domain of knowledge. Ontologies are mainly used for knowledge sharing and reuse across different applications. Ontology is defined as specifications of representational vocabulary for a shared domain of discourse (definitions of classes, relations, functions and other objects) [16]. Ontologies permit to share meanings of terms to overcome barriers created by disparate vocabularies, approaches, representations, and tools in a given domain. Qin and paling [17] propose the conversion of controlled vocabularies into ontologies to get deeper semantics in describing digital objects, both conceptually and relationally. Ontology can be used in many application categories, such as common access information and ontology-based search [18]. Many ontology technologies have been developed, for example: ontology representation language (*RDF, OWL, XTM*), ontology search engine (*JENA, Ontopoly*),

Toward ontology-based file systems, Dalat 2006

query language (*RDQL*, *SPARQL*, *TMQL*), graphic ontology representation (*Vizigator*), ontology editor (*Protégé*).

A file semantic conception usually concerns many knowledge domains and one knowledge domain may be shared by many applications. We adopt multi-ontology layer approach proposed by Uschold and Jasper [18] and already used by Xiao and Cruz [15] to shared file semantic conception. Uschold and Jasper [18] proved that for sharing the ontologies at level L_i , it's required to refer to ontologies at level L_{i+1} . We define two types of ontologies: *domain ontologies* and *application ontologies* see figure 1.

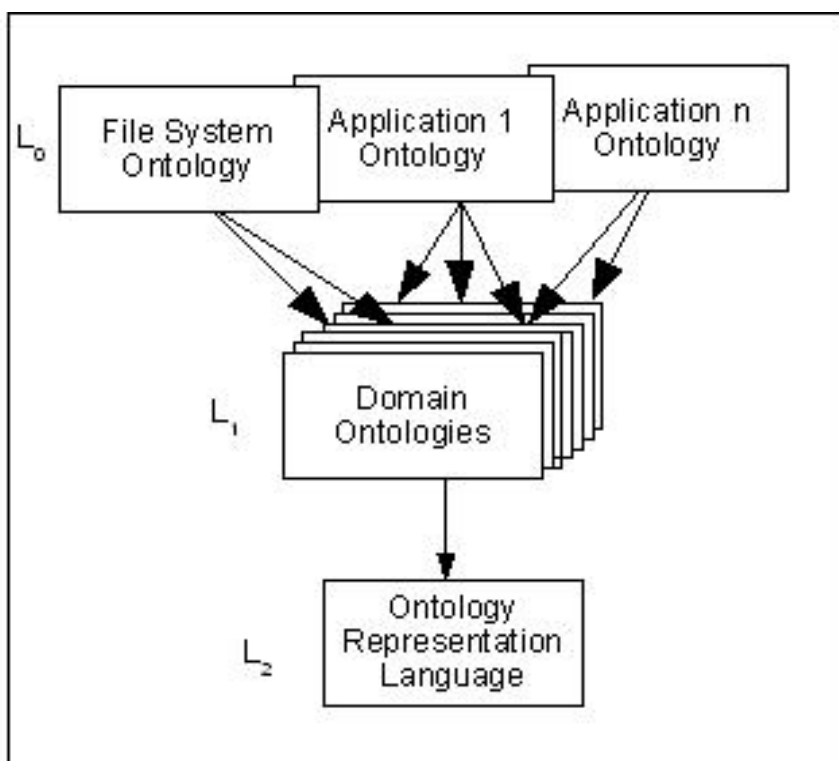


Figure1: Multi-ontology layer approach for SFS

Domain ontologies specifies terms and concepts for different knowledge domains. Application ontology specifies file semantic conception of an agent by using terms and concepts defined in shared domain ontologies. All domain ontologies are finally written in the same ontology representation language.

Toward ontology-based file systems, Dalat 2006

uses terms, conception and rules defined by domain ontologies to define its own.

- **Domain Ontologies:** As application ontology, domain ontologies also have names, specifications and sometime other domain ontologies that they needs. There are ontology libraries that can be reused, such as Ontolingua ontology library [19], DALM ontology library [20].

- **Ontology Registry:** Ontology registry maintains information about the application ontologies and domain ontologies that have already been registered. A file semantic concept is only available in a SFS when its application ontology is registered in ontology registry. The domain ontologies in scope of application ontology have to exist in ontology registry before the application ontology is registered. An ontology registry can support a local *ontology store* to store application and domain ontologies or uses other ontology search engines as ontology stores. This is interesting because it makes SFS less complicate, but a protocol has to be defined for communication between ontology registry and ontology stores.

- **Semantic Generator:** Semantic Generator maintains a list of AFSG which are responsible for creating file semantics committing application ontologies already registered in ontology Registry. AFSG is usually created by application developers or persons who understand file semantic conception of applications. AFSG should be design as plug-in code modules so that new application ontologies can be introduced into SFS. An AFSG has to specify the information about application ontology that it's implemented for and events that it's interested in. While adding a new AFSG, Semantic Generator has to verify the existence of corresponding application ontology and registry the interested events with Event Manager. So Ontology Registry has to provide services to explore its information.

- **Event Manager:** Event Manager of Semantic Generator maintains a list of events and corresponding AFSG. Event Manager has to wait for these events from operating system. When an interesting event occurs, Event Manager will start one ore more corresponding AFSG to harvest file semantics. Protocols for accessing the file store and semantic store should be defined so that AFSG can access to files and file semantics.

- **Semantic Store:** file semantics is usually represented under triples {fileRef, proRef,value} where *fileRef* is a reference to a file in file store, *proRef* is a reference to a property defined in a domain ontology and *value* may be a string or an instance of domain ontology. File semantics deals with organization and storage of file semantics and ontology instances. SFS can support a local semantic store or use external semantic search engine supporting ontology as semantic store.

- **Semantic Search Engine:** Data in ontology store and semantic store has to be indexed by semantic search engine that supports ontology. When ontology store and semantic store aren't the local ones

Toward ontology-based file systems, Dalat 2006

of a semantic search engine, protocols for interchange data between ontology and semantic store with semantics search engine need to be defined. Semantic search engine has to be designed to support new ontologies and frequent changes of file semantics. Query language should permit in its parameters the combination of application ontologies, domain ontologies, ontology instances and file semantics.

- **File store:** File store provides other components with its file accessing service. Depending on chosen approach to SFS architecture, SFS can implement a new file store, the integrated approach, or use current file systems as its file store, the augmented approach. The survey of the Vasudevan and Pazandak [21] shows that the integrated approach is the quicker way to get the realization of SFS, if end-users are willing to accept more drastic changes to their operating systems. While the augmented approach provides less perfect implementations of the functionality that a user can expect from an integrated SFS, but with almost no negative perturbation to the user's current file access capabilities. An augmented SFS can come closer in functionality and implementation to an integrated one if there is more supports from operating system in notification of data change and greater access to file content. At present, the augmented approach is more prevalent than integrated approach because required supports from operating system have been improved, for example new notification mechanism on Linux 2.6 platforms. The augmented approach SFS leaves file system calls unchanged, while providing more file semantic system calls for querying and manipulating semantically file information.

- Final discussion is how users can access to new ontology-based file system features. Usually, users access to file system via shell programs (command line or graphics). To offer users with new ontology-based file system features, these shell programs should use the new file semantic system calls and the new searching service provided by ontology-based file system. A shell program could also define its own application ontology to allow more convenient usages in file management.

6. RELATED WORKS

"The *Semantic Web* is an extension of the current Web in which information is given well-defined meaning, better enabling computers and people to work in cooperation" [22]. The World Wide Web Consortium [23] promotes the Semantic Web technologies to provide a common framework that allows data to be shared and reused across application, enterprise, and community boundaries. It is based on RDF, which integrates a variety of applications using XML for syntax and URIs for naming. While XML is a powerful, flexible syntax for structured documents, it imposes no semantic constraints on the meaning of these documents. RDF [24] is for semantics and provides a powerful framework for supporting the exchange of knowledge on the Web. OWL [25] provides a language for defining structured, web-based ontologies which deliver richer integration and

Toward ontology-based file systems, Dalat 2006

interoperability of data among descriptive communities. According to Eric Miller, W3C Semantic Web Activity Lead, "The Semantic Web is made through incremental changes, by bringing machine-readable descriptions to the data and documents already on the Web." [26]

Topic Maps published in the standard ISO/IEC1325 define a model for the semantic structuring of knowledge networks. Topic map is designed to manage the information glut, build valuable information network over any kind of information resources [27]. Topic maps can be regarded as the International standard for codification that is the necessary prerequisite for the development of tools that assist in the generation and transfer of knowledge [28]. Subjects are the starting point for Topic Maps. A subject may be a conception, idea, notion or "anything whatsoever" that is worth of becoming a *topic* in a topic map. A topic describes a subject by its three characteristics: topic names - human readable name for the subject, occurrences - link to information resources relevant for topic and associations with other topics. Topic maps can be used as a mechanism for controlled vocabularies and to represent taxonomies, thesauri, faceted classification, synonym rings and authority files [29]. The standard XTM [30] interchange format for topic maps has been standardized. A standard schema language for topic maps, called TMCL (Topic Map Constraint Language) [31] is under development. Many free topic map engines and tools have been developed [32]. OKS Samplers is an example of using topic maps for creating ontologies, instances of ontologies and ontology-based searching [33].

Semantic Desktop deals to transfer the semantic web consisting of technology, philosophy and people involved to desktop computers. Sauermann and al.[34] define that "A Semantic Desktop is a device in which an individual stores all her digital information like documents, multimedia and messages. These are interpreted as Semantic Web resources, each is identified by a Uniform Resource Identifier (URI) and all data is accessible and queryable as RDF graph. Resources from the web can be stored and authored content can be shared with others. Ontologies allow the user to express personal mental models and form the semantic glue interconnecting information and systems. Applications respect this and store, read and communicate via ontologies and Semantic Web protocols. The Semantic Desktop is an enlarged supplement to the user's memory". Sauermann and al.[34] also show that ontology-based file system is one of the building blocks supporting data and information for Semantic Desktop.

7.CONCLUSION AND FUTURE WORKS

The paper has motivated the need for integrating support for ontologies into file system to build efficient SFS whose file semantics can be shared by all agents. We have answered the question about file semantic by identifying three existent types of file semantics. We adopt multi ontology layer approach into our proposed ontology-based file system to

Toward ontology-based file systems, Dalat 2006

satisfy our goal. The challenges on ontology-based file system design are also analysed to expose the feasibility of our proposal. However, there are still many research problems that need to be addressed to realize an ontology-based file system. We propose some of them below:

- Evaluate the ontology-based file system model proposed in section 4;
- Investigate the design of ontology-based file system. In particular, how to reuse and integrate existent ontology technologies such as ontology representation languages, domain ontology libraries, semantic search engines and query languages;
- Test how semantic file services cooperate with standard file services.
- Evaluate performances and accuracy of applications that use a prototype ontology file system.

REFERENCES

- [1] Udi Manber and Sun Wu., 'Glimpse: a tool to search through entire file systems', *Proceedings of the USENIX Winter Conference*, page 23-32, January 1994
- [2] Google, 'Google Desktop', <http://desktop.google.com/>
- [3] 'Beagle', http://beaglewiki.org/Main_Page
- [4] Dominic Giampaolo, 'Practical File System with the Be File System', *Morgan Kaufmann Publishers*, San Francisco, California, 1999
- [5] Lorcan Dempsey and Rachel Heery, 'Metadata: a current view of practice and issues', *Journal of Documentation*, 54 (2), page 145-172, Mar-1998
- [6] Mark A. Sheldon, David K. Gifford, Pierre Jouvelot and James W. O'Toole Jr., 'Semantic File Systems', *Proceedings of the 13th ACM Symposium on Operating Systems Principles*, page 16-25, 1991
- [7] C. Mic Bowman, Chanda Dharap, Mrinal Baruah, Bill Camargo, and Sunil Potti, 'A File System for Information Management', *Proceedings of the International Conference on Intelligent Information Management Systems*, 1994
- [8] Mallik Mahalingam, Chunqiang Tang, and Zhichen Xu, 'Towards a Semantic, Deep Archival File System', *The 9th International Workshop on Future Trends of Distributed Computing Systems*, 2003
- [9] Z. Xu, M. Karlsson, C. Tang, and C. Karamanolis, 'Towards a Semantic-Aware File Store', *Workshop on Hot Topics in Operating Systems*, page 145-150, 2003
- [10] Apple Computer, Inc., 'Tiger Developer Overview Series - Working with Spotlight', 2005, <http://developer.apple.com/macosx/spotlight.html>
- [11] M. Fisher and A. Sheth, 'Semantic Enterprise Content Management', *Practical Handbook of Internet Computing*, M. Singh (ed), CRC Press, page 145-150, 2003

Toward ontology-based file systems, Dalat 2006

- [12] Paul Alexandru Chirita, Rita Gavriloaie, Stefania Ghita, Wolfgang Nejdl, and Raluca Paiu, 'Activity Based Metadata for Semantic Desktop Search', *LS3 Research Center, University of Hanover, Deutscher Pavillon, Expo Plaza 1, 30539 Hanover Germany, 2004*
- [13] Craig A. N. Soules and Gregory R. Ganger, 'Connections: Using Context to Enhance File Search', *Proceedings of the 20th ACM Symposium on Operating Systems Principles 2005*, Brighton, UK, page 119-132, 2005
- [14] .Yoann Padioleau, 'Logic File System, un système de fichier basé sur la logique', *Université de Rennes 1*, février 2005
- [15] Huiyong Xiao and Isabel F. Cruz, 'A Multi-Ontology Approach for Personal Information Management', *Proceeding of 1st Workshop on The Semantic Desktop*, International Semantic Web Conference 6 November 2005, Galway, Ireland, 2005
- [16] Thomas R. Gruber, 'Toward principles for the design of ontologies used for knowledge sharing', Guarino & Poli (Eds), 1993
- [17] Jian Qin and Stephen Paling, 'Converting a controlled vocabulary into an ontology: the case of GEM', *Information Research*, Vol. 6 No. 2, 2001
- [18] M. Uschold and R. Jasper, 'A Framework for Understanding and Classifying Ontology Applications', *Proceedings of the IJCAI99 Workshop on Ontologies and Problem-Solving Methods(KRR5)*, 1999
- [19] 'DAML Ontology Library', 2004, <http://www.daml.org/ontologies/>
- [20] Ontolingua, 'Ontolingua', <http://www.ksl.stanford.edu/software/ontolingua/>
- [21] Venu Vasudevan and Paul Pazandak., 'Semantic File Systems', *Object Services and Consulting, Inc*, 1996, <http://www.objs.com/survey/OFSExt.htm>
- [22] Tim Berners-Lee, James Hendler and Ora Lassila, 'The Semantic Web', *Scientific American*, May 2001
- [23] The World Wide Web Consortium, <http://www.w3.org/>
- [24] W3C Recommendation, 'RDF Primer', 10 February 2004, <http://www.w3.org/TR/2004/REC-rdf-primer-20040210/>
- [25] W3C Recommendation, 'OWL Web Ontology Language Overview', 10 February 2004
- [26] 'World Wide Web Consortium Issues RDF and OWL Recommendations', <http://www.w3.org/2004/01/sws-pressrelease>
- [27] H. Holger Rath, 'White Paper - The Topic Maps HandBook', Empolis GmbH, 2003
- [28] ..Steve Pepper, 'The TAO of Topic Maps - Finding the Way in the Age of Infoglut', *Ontopia*, <http://www.ontopia.net/topicmaps/materials/tao.html>
- [29] Lars Marius Garshol, 'Metadata? Thesauri? Taxonomies? Topic Maps! Making sense of it all', *Journal of Information Science*, volume 30, number 4, Chartered Institute of Library and Information Professionals, pages 378-391, 2004
- [30] TopicMaps.Org, 'XML Topic Maps (XTM) 1.0', <http://www.topicmaps.org/xtm/1.0/>, 2001

Toward ontology-based file systems, Dalat 2006

- [31] ISO/IEC JTC1/SC34, 'Topic Maps Constraint Language',
<http://www.isotopicmaps.org/tmcl/tmcl-2005-02-12.html>
- [32] <http://www.topicmap.com/topicmap/tools.html>
- [33] Ontopia, 'OKS Samplers',
<http://www.ontopia.net/download/freedownload.html>
- [34] Leo Sauermann, Ansgar Bernardi and Andreas Dengel, 'Overview and Outlook on the Semantic Desktop', *Proceeding of Semantic Desktop Workshop 2005*