Integrating Ontology into Semantic File Systems

Ba-Hung Ngo, Christian Bac, Frédérique Silber-Chaussumier

Integrating Ontology into Semantic File Systems

Ba-Hung Ngo, Christian Bac, Frédérique Silber-Chaussumier

Abstract— **Semantic file systems enhance standard file systems with the ability of file searching based on file semantics. In this paper, we propose to integrate the support for ontologies into a file system to build efficient semantic file systems whose file semantics can be shared between users, applications and semantic file systems themselves. We call it ontology-based file system. We identify three existing types of file semantics: property-based, content-based and context-based semantics and adopt multi-ontology layer approach to enhance file semantic systems with four file semantic characteristics: file semantic representability, file semantic extensibility, file semantic standardization, and interoperability.**

Index Terms—**File Systems, Semantics, Ontology, Metadata**

I. INTRODUCTION

The abstract goal of a file system is to store, retrieve, locate, and manipulate data [1]. 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. Recent researches on *semantic file systems* (SFS) deal with integrating semantics into file system to enhance it with the ability to locate information by means of *file semantics*. File searching based on file 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 meaning. To solve this, we propose to integrate the support for ontologies into a file system. This support will help build efficient SFS where file semantics is shared by all agents. We found that, the sharing of file semantics increases the efficiency of SFS in two ways. First, it supports the interoperability: file semantics created by an agent can be understood and used by other agents for file searching. Second, it standardizes file semantics created from different agents so that operations on collected file semantics, such as ranking of searching result to improve the relevance of searching result, become more precise.

This paper discusses file semantics in section 2, argues the need of integrating ontology into a file system and proposes an ontology-based file system model in section 3. Some related works are presented in section 4. The last section is our conclusion and future works.

Authors are with Institut National des Télécommunications, 09 Charles Fourier, 91011 Evry, Cedex, France. Email: {Hung.Ngo_Ba, Christian.Bac, Frederique.Silber-Chaussumier}@int-evry.fr.

II. FILE SEMANTICS

SFS are based on the principles of information retrieval systems (IR system) where syntactic and semantic information of index terms or keywords are extracted from document text and used to match user information need [2]. However, SFS collect not only index terms but also other relevant metadata. The information on which SFS base to provide users with their file searching service is called *file semantics*. We have adapted Dempsey and Heery definition of metadata [3] as follows: "File semantics is information associated with files 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". We can point out the existence of file semantics by classifying them into three levels: 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. The existing SFS [4]-[10] all support property-based semantics represented by *attributes* under the form of pairs *name-value* where the name exactly identifies a file property.

Content-based semantics is information obtained by analyzing file content. Content-based semantics represents internal organization of file content: roles played by pieces of content data. The general role, called *text*, is used to specify all text in a document. The conventional text-based search tools, such as Glimpse [11], Google Desktop [12] and Beagle [13] index terms or keywords in text to provide their searching services. Text-based searching is also supported in the existing SFS [4]-[10]. There are others interesting roles to be used for file searching. In MIT-SFS [4], the roles such as *imports*, *exports*, *function* can be used for searching objects in files. MP3 music files can be retrieved with roles as *artist*, *album*

title, and *year* in LFS [10]. HTML files can be searched for following characteristics: having *heading* containing words "Album" and "Paris", containing "JPEG" *images* taken by a specific *camera*. In the last example, the heading, image and camera are the roles played by content data. To represent valid roles in a file structure, metadata whose syntax and semantics are defined by an application is usually used.

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: other *files*, *concepts*, *persons*... It can be the context in the computer world, such as *concurrently accessed files*, the *user's current task*, any *action* or *data* that the user associates with the *file's use* in Connections [9]. Context can be connected to the real world: *user's opinion* on mp3 music files (*exciting*, *normal* or *boring*) [10], *user's classification* of JPEG files (*family*, *friend*, *summer*, *vacation*, *wedding*, *work* ...). Context-based semantics can be obtained from SFS and the application that manipulates files. For instance, when a file is attached to an email, its context consists of the *sender*, the *receiver* and the *subject* of the email. At the receiver side, if the file is extracted from the email and saved normally, it looses this context. However, if supported, the file will be saved in SFS together with its context-based semantics so that the file can be searched later by the sender, receiver or subject of the email that it was attached.

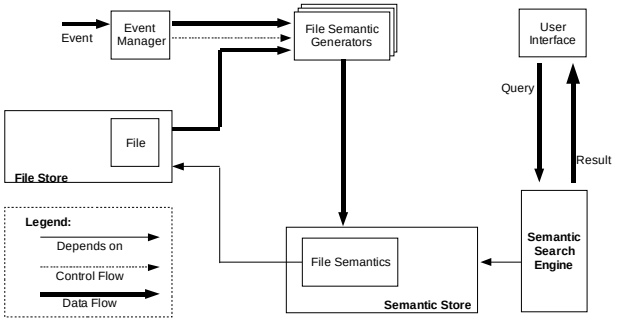

Fig. 1. General semantic file system model.

The existence of file semantics is determined by agents. For instance, standard file systems usually create file metadata that can be considered as property-based semantics for all files; applications can define their own file structures that determine content-based semantics; application execution context contains contexts for files handled by the application; users via applications can assign their opinions to files. We use the notion *concept of file semantics* to specify all types of semantic data that can be created and associated to files handled by an agent. Each agent has its own concept of file semantics. To be able to collect file semantics for SFS, one has to understand the corresponding concept of file semantics.

III. INTEGRATING ONTOLOGY INTO SEMANTIC FILE SYSTEM

The use of file semantics in SFS to provide semantic-based searching is described in the *figure 1*. File semantics is collected by many *File Semantic Generators*. Depending on the occurred event and the handled file type, a file semantic generator is called to collect file semantics. An agent can define its file semantic generators to collect interesting file semantics. Usually, SFS define file semantic generators to collect property-based semantics and content-based semantics for popular file types. For specific or new file types, only the application designers know what the content-based semantics is and how to collect them from the file content. So the application designers are responsible for defining new file semantic generators. Similarly, there will be many file semantic generators supported by applications to collect context-based semantics in different context. Although collected from different sources, all file semantics is finally stored together in a *Semantic Store* in a structure that is convenient for *Semantic Search Engine* to explore and provide users with semantic-based searching via a User interface. Effective semantic-based searching should provide general information system characteristics as file semantic representability, file semantic extensibility, interoperability and file semantic standardization.

The file semantic representability implies that SFS have to use methods for file semantic representation. In IR systems, the meaning of index terms or keywords is interpreted by users. IR systems are based on pattern-matching principle. They do not assure that same terms in user query and in the searching result have the same meaning. SFS should overcome this with a mechanism that controls the meanings of the terms in SFS. File semantic representability is essential in SFS.

The file semantic extensibility requires that new concept of file semantics can be added to SFS without modifying or reconfiguring the structure of SFS. In the first step, SFS integrate some built-in concepts of file semantics such as property-based semantics, content-based semantics of some popular file types or context-based semantics for popular applications. Next, to provide file semantics of new applications, SFS should extend predefined semantics. So SFS have to be extensibility.

The interoperability requires that file semantics created by an application, once collected by SFS, can be understood and used by other agents without prior communication. This can be done if the concept of file semantics is understood by the agents using it. So SFS should provide mechanisms to publish and share file semantics between all agents in the system.

The file semantic standardization requires that the appearance of the same file semantics in different concepts of file semantics has to be recognized. It means that the same concept that appears in different agents must have the same meaning. File semantic standardization makes the searching result become more relevant.

In order to provide an effective SFS, we propose to integrate ontology into SFS. 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) [14]. 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 [15] 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 [16]. Many ontology technologies have been developed, for example: ontology representation language (RDF [17], OWL [18], XTM [19]), ontology search engine (JENA [20], Ontopoly [21]), query language (RDQL [22], SPARQL [23], TMQL [24]), graphic ontology representation (Vizigator [25]), ontology editor (Protégé [26]).

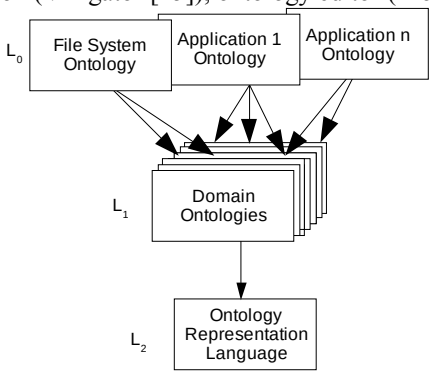

Fig. 2. Multi-ontology layer approach for semantic file systems.

In our SFS model, ontology is used to specify concept of file semantics. For instance, a SFS can use ontology to specify property-based semantics that he can created for files. A mail user agent can specify context-based semantics that he can saved together with attached files by means of ontology. Usually a concept of file semantics 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 [16] and already used by Xiao and Cruz [27] to share concept of file semantics. Uschold and Jasper [16] proved that for sharing 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 the *figure 2*.

Domain ontologies specify terms and concepts for different knowledge domains. Application ontology specifies agent's concept of file semantics by using terms and concepts defined in shared domain ontologies. All domain ontologies are finally written in the same ontology representation language. While ontology representation language helps SFS having the file semantic representability, the sharing domain ontologies between application ontologies helps SFS having the file semantic standardization.

We propose a SFS model that supports multi-layer ontology approach in the *figure 3*. This model inserts the new *Ontology Registry* component into the general SFS model presented at the beginning of this section. Ontology Registry maintains information about the application ontologies and domain ontologies that have already been registered. A concept of file semantic is only available in SFS if its application ontology is registered in Ontology Registry. The domain ontologies must be inserted in Ontology Registry before being referenced by application ontology. In the same way, a File Semantics Generator has to register its application ontology before collecting semantic data. Conversely, it is necessary to register the corresponding ontologies before searching for files. One could add new application ontologies into Ontology Registry for extensibility. By browsing Ontology Registry, one agent can discover the concept of file semantics of other agents. As a result, Ontology Registry also provides interoperability.

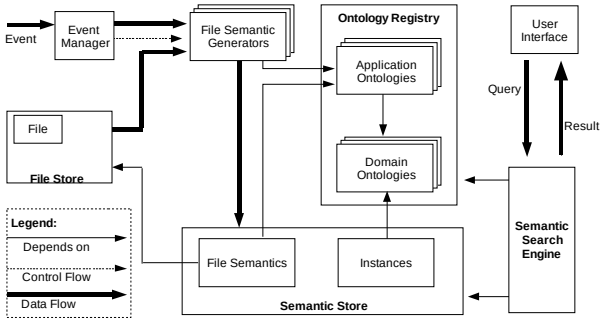

Fig. 3. Ontology-based semantic file system model.

IV. 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" [28]. The World Wide Web Consortium [29] 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 [17] is for semantics and provides a powerful framework for supporting the exchange of knowledge on the Web. OWL [18] provides a language for defining structured, web-based ontologies which deliver richer integration and interoperability of data among descriptive communities. According to Eric Miller, W3C Semantic Web Activity Leader, "The Semantic Web is made through incremental changes, by bringing machine-readable descriptions to the data and documents already on the Web." [30]. The semantic technologies developed for Semantic Web such as Semantic Representation Language, Domain Ontologies for different knowledge domains, Semantic search engines, ... are good references for designing SFS.

Topic Maps published in the standard ISO/IEC1325 defines a model for the semantic structuring of knowledge networks. Topic Maps is designed to manage the information glut, build valuable information network over any kind of information resources [31]. 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 [32]. Subjects are the starting point for Topic Maps. A subject may be a concept, idea, notion or "anything whatsoever" that is worth of becoming a *topic* in a topic maps. 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 to control vocabulary and to represent taxonomies, thesauri, faceted classification, synonym rings and authority files [33]. The standard XTM [19] interchange format for topic maps has been standardized. A standard schema language for topic maps, called TMCL (Topic Map Constraint Language) [34] is under development. Many free topic map engines and tools have been developed [35]. OKS Samplers is an example of using topic maps for creating ontologies, instances of ontologies and ontology-based searching [36]. The ontological engineering, open source topic map software, topic map visualization can provide a foundation for SFS.

Semantic Desktop deals with transferring the semantic web consisting of technology, philosophy and people involved to desktop computers. Sauermann *et al*. [37] 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 *et al.* [37] also show that ontology-based file system is one of the building blocks supporting data and information for Semantic Desktop. SFS can share with Semantic Desktop the way in which ontological technologies are used to give meaning to information.

V. Conclusion and Future Works

Based on existing semantic file systems and related works, we identified three types of file semantics: property-based, content-based and context-based file semantics. Controlling vocabulary is a major issue of file semantics: it would provide interoperability between file systems, applications and final users. In this article, we propose to address this issue in integrating ontologies into a file system. We explore semantic file system design and propose a multi-layer ontology approach. In the next months, we will further investigate the design of ontology-based file system. In particular we want to investigate how to reuse and integrate ontology technologies such as representation languages, domain ontology libraries, semantic search engines and query languages and provide a first prototype.

References

[1] D. Giampaolo, Practical File System with the Be File System, Morgan Kaufmann Publishers, San Francisco, California, 1999

[2] R. Baeza-Yates and B. Ribeiro-Neto, Modern Information Retrieval, ACM Press, Addison-Wesley, 1999

[3] L. Dempsey and R. Heery, "Metadata: a current view of practice and issues," Journal of Documentation, 54 (2), 1998, pp 145-172

[4] D. K. Gifford, P. Jouvelot, J. J. W. O'Toole, and M. A. Sheldon, "Semantic File Systems," Proceedings of the 13th ACM Symposium on Operating Systems Principles,1991, pp 16-25

[5] M. Baruah, C. M. Bowman, B. Camargo, C. Dharap, and S. A Potti, "File System for Information Management," Proceedings of the International Conference on Intelligent Information Management Systems, 1994

[6] M. Mahalingam and C. Tang, Z. Xu, "Towards a Semantic, Deep Archival File System," The 9th International Workshop on Future Trends of Distributed Computing Systems, 2003

[7] C. Karamanolis, M. Karlsson, C. Tang, , Z. Xu, Towards a Semantic-Aware File Store. Workshop on Hot Topics in Operating Systems (2003) 145-150

[8] Apple Computer, Inc. Tiger Developer Overview Series - Working with Spotlight. http://developer.apple.com/macosx/spotlight.html (2005)

[9] G. R. Ganger and C. A. N. Soules, "Connections: Using Context to Enhance File Search," Proceedings of the 20th ACM Symposium on Operating Systems Principles, Brighton, UK, 2005, pp 119-132

[10] Y. Padioleau, "Logic File System, un système de fichier basé sur la logique," Université de Rennes 1, 2005

[11] U. Manber and S. Wu, "Glimpse: a tool to search through entire file systems," Proceedings of the USENIX Winter Conference, 1994, pp 23-32

[12] Google. Google Desktop. http://desktop.google.com/

[13] Beagle. http://beaglewiki.org/Main_Page

[14] T. R. Gruber, "Toward principles for the design of ontologies used for knowledge sharing," Guarino & Poli (Eds), 1993

[15] S. Paling and J. Qin, "Converting a controlled vocabulary into an ontology: the case of GEM," Information Research, Vol. 6 No. 2, 2001

[16] R. Jasper and M. Uschold, "A Framework for Understanding and Classifying Ontology Applications," Proceedings of the IJCAI99 Workshop on Ontologies and Problem-Solving Methods (KRR5), 1999

[17] W3C Recommendation. RDF Primer. http://www.w3.org/TR/2004/REC-rdf-primer-20040210/ (2004)

[18] W3C Recommendation. OWL Web Ontology LanguageOverview (2004)

[19] TopicMaps.Org. XML Topic Maps (XTM) 1.0. http://www.topicmaps.org/xtm/1.0/ (2001)

[20] http://jena.sourceforge.net/

[21] http://www.ontopia.net/

[22] http://www.w3.org/Submission/RDQL/

[23] http://www.w3.org/TR/rdf-sparql-query/

[24] http://www.isotopicmaps.org/tmql/

[25] http://www.ontopia.net/solutions/vizigator.html

[26] http://protege.stanford.edu/

[27] I.F. Cruz and H. Xiao, "A Multi-Ontology Approach for Personal Information Management," Proceeding of 1st Workshop on The Semantic Desktop, International Semantic Web Conference, Galway, Ireland, 2005

[28] T. Berners-Lee, J. Hendler, and O. Lassila, The Semantic Web, Scientific American, 2001

[29] The World Wide Web Consortium. http://www.w3.org/

[30] World Wide Web Consortium. Issues RDF and OWL Recommendations. http://www.w3.org/2004/01/sws-pressrelease.

[31] H. H. Rath, White Paper - The Topic Maps HandBook, Empolis GmbH, 2003

[32] S. Pepper, "The TAO of Topic Maps - Finding the Way in the Age of Infoglut", Ontopia, http://www.ontopia.net/topicmaps/materials/tao.html

[33] L.M. 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, 2004, pp 378-391

[34] ISO/IECJTC1/SC34. Topic Maps Constraint Language. http://www.isotopicmaps.org/tmcl/tmcl-2005-02-12.html.

[35] http://www.topicmap.com/topicmap/tools.html.

[36] Ontopia. OKS Samplers. http://www.ontopia.net/download/freedownload.html.

[37] A. Bernardi, A. Dengel, and L. Sauermann, "Overview and Outlook on the Semantic Desktop," Proceeding of Semantic Desktop Workshop, 2005