OSMOSIS-RFID: Semantic File System to Incorporate Real Objects into the Virtual World

OSMOSIS-RFID: Sistema de Ficheiros Semântico para Incorporar Objectos Reais no Mundo Virtual

André Mendes
andre.mendes@ist.utl.pt

Universidade Técnica de Lisboa
Instituto Superior Técnico
Av. Rovisco Pais
Portugal

Abstract. With the increasing storage capacity of computers, it became evident that the use of a hierarchical file system no longer fits the interests of the user, i.e., with this kind of file system the user cannot find a file in an easy, quick and intuitive way. For this reason, the concept of Semantic File System became quite popular. This is a system for storing and organizing information, that allows associations between attributes and files and then searching those files using attributes (or keywords). All the limitations of hierarchical systems led us to the development of OSMOSIS. This is a semantic file system with the ability to integrate physical objects in the virtual world. OSMOSIS can be divided into two major parts. The first (OSMOSIS-SFS) is a semantic file system that can associate, with files, attributes extracted from its properties and contents, can explicitly relate two files, and allows users to manually apply a Tag to a file. The second part (OSMOSIS-RFID) is the integration of the semantic system into the real world. In other words, with this system it will be possible to attach RFID Tags to physical locations (e.g. living room) or physical objects, related to virtual files (e.g. photo album) and then use that information to find physical objects, or to anticipate what files the user may want to see, based on his/hers physical location.

1 Introduction

The increase in computers' storage capacity led to an enormous growth in the number of files stored in them. Due to this growth we can sometimes find files in our own computer that we ourselves don't remember posessing.

Nowadays, the majority of the Operating Systems uses a hierarchical directory system to organize files. Thus, for a user to access a file he/she must browse through the directories or specify the file's complete path. In any case, the user must remember where the file is stored. Despite the fact that this hierarchical file is useful to computers with a small number of files, when a large quantity of data is present it sometimes becomes difficult for a user to remember the exact

location of a certain file. This happens because a directory hierarchy is inflexible and does not allow to register associations between files from different directories, increasing the difficulty of finding a specific file, that can fit into more than one directory. Instead, it is much more intuituve and easy to remember parts of the file's name, words contained in it, or even the context in which it was created.

To help solve this problem, the notion of Semantic File System [6] was created. A Semantic File System is an information storage and organization system, that allows to automatically extract attributes from files and afterwards finding them through those attributes. The files' semantic is, in this sense, a set of features that are part of the files, but are not necessarily related with their location in the directory hierarchy.

Besides the attributes automatically extracted from files, it is also possible for the user to define other attributes through the placement of Tags. These Tags are keywords that can be assigned to a file, or even a set of files, and that allow to identify, in a more intuitive way, a given file. This is a concept used on a daily basis on the Web, but one that is still poorly explored in the context of local files, to which it can be applied. It is also necessary to highlight that the attributes that are automatically extracted by the system are essentially Tags, albeit automatically assigned.

Another type of Tags, apart from the ones referred previously, are the *Radio-Frequency IDentification* (RFID), or physical Tags. The RFID technology can be divided in two great components. The first are the RFID Tags, that are no more than integrated circuits where information about a certain entity is stored. Because RFID Tags transmit their information through a radio-frequency signal, the second part of this tecnology consists in a reader, with the purpose of receiving the radio signal transmitted by the Tag and converting it into useful information.

These physical Tags can be associated to or incorporated into physical objects in the real world, so a one-to-one relationship can be established between object and RFID Tag. As such, it is then possible to make an analogy between a file, or directory, and the RFID Tags, for instance, the `Photographs 2008` directory, containing the set of photographs from 2008 present in the PC, may correspond to the RFID Tag that is attached to the physical photography album from 2008.

1.1 Work Goals

The global objetive of the work is the creation of a system that integrates virtual objects, in file systems, and real objects, marked with RFID Tags. However, this global goal can be divided into two major objectives. The first is the creation of a Semantic File System (OSMOSIS-SFS), that will allow assigning Tags to files and afterwards use that semantic information to find more easily a given file. The second objective is the integration of the Semantic File System with the real world. In other words, we intend to allow physical objects to be marked with RFID Tags, which have a relationship with the virtual files existing in the file system native to the Operative System. This system is named

OSMOSIS-RFID, and will make possible, for instance, to enter a living room, reading its RFID Tag and learning which objects are inside.

Semantic File System As stated previously, one of the work's objectives is the creation of a Semantic File System. In this sense, what we intend to create is a new file system, coexisting with the hierarchical system from the Operating System, but one that allows to find a certain file without being necessary for a user to remember its complete path. This system will allow to assign to files attributes that are extracted automatically from their properties and content, as well as explicit relationships between two files and even user-created attributes (Tags).

Integration with the Real World The second phase of the work has as its objective to integrate OSMOSIS-SFS, previously described, with objects and locations in the physical world. To make that possible we intend to develop a version of OSMOSIS-SFS with less functionality (remote client), so it can be installed in a PDA. This version will no longer make available the functionality to automatically extract Tags from files, and neither will it allow the user to insert *Tags* manually. However, it will be possible to associate RFID Tags to physical locations (i.e. living room) or physical objects, related to virtual files (i.e. photography album), and afterwards use that information to help finding physical objects, or to anticipate which files the user might want to see, based in his/hers physical location.

In section 2 an analysis is conducted of systems developed in the area of semantic file systems, in section 3 the OSMOSIS architecture, in 4 we describe the evaluation and, finally, we present this work's conclusions.

2 Related Work

Based on the kind of semantic information, the existing work in the semantic file system area can be divided into three main categories:

- Content-based semantics.
- Context-based semantics (also including content).
- Semantics based on the user-created Tags.

In the first category, the systems automatically obtain the semantical information from attributes, as for instance a file's author, creation date, modification date, name, size and type. Besides, these systems are also characterized by the fact that they obtain semantic information from the very content of the file as, for example, words found inside the file. In this case there is the disadvantage that this analysis can only be performed on text files. Systems that fall into this category are described in [2,4,5,6,8,11,12,13].

As for context, systems can, other than obtaining semantic information of the same kind of the previous systems, obtain this information from the file's use context (or, concisely, the file's context). The context of a file can be, for instance, other files who are accessed concurrently with it, or even tasks the user performs simultaneously with the file access. Besides, this context can also be related to the real world, as for example when a user classifies an MP3 song as good, normal or bad. This kind of semantic information allows, in this way, to establish relationships between a file and other files, people and concepts. In this category are systems as those described in [1,10,14,15,17,18,20].

In the third category, systems use as semantic information the Tags inserted by the users. These Tags are keywords associated to files. In these systems, as the same Tag can be assigned to one or more files, all those who are marked with it are related to each other. Systems in this category are described in [3,19].

Apart from the system mentioned above, there are others that do not fit in only one of these categories, but in several of them. There is the case of DBFS [7] and Insight [9], that posess characteristics from the categories "Content-based Semantics" and "Semantics Based on Assigned Tags". The SemDAV system [16], in turn, possesses characteristics from the "Context-based Semantics" and "Semantics Based on Assigned Tags". These systems, that we may call hybrid, will be described in the last section of this chapter.

2.1 Content-based Semantics

Based on the studied "Content-based Semantics" systems we may conclude that all of them support a large variety of file types. Furthermore, we may also conclude that a large quantity of these systems uses virtual directories, but there are also some that use traditional search and even greatly elaborated graphical user interfaces. Finally we can verify, based on the systems for which we possess suficient data, that the way semantic information is organized and represented varies greatly from system to system. It is in this category that the majority of the systems developed in the semantic file system area fall into, however there is the disadvantage of them not being able to capture the files' use context, or allow the assignment of Tags by the user.

2.2 Context-based Semantics

To fill a flaw of content-based semantic systems, work was developed falling into the "Context-based Semantics". These systems, besides obtaining attributes from files' content, also resort to those files' use context as a source of semantic information. It is, actually, a way to infer relationships between files, allowing to increase the general usability of the system.

By analysing the systems that belong to this category, we can conclude that, essentially, there are two ways to relate files. Relationships can be inferred from the files' access patterns, or by explicitly defining relationships between them. We can also verify that, to extract semantic information from context, systems must monitor file accesses and infer relationships from that. Finally, we can also

see that in the Connections [18], Oxygen [15] and Copernicus [10] systems a graph is always used as a way to represent the context information. Despite the added advantages, this kind of systems possesses a greater complexity in discovering and managing Tags obtained from context.

2.3 Semantics Based on Assigned Tags

The "Semantics Based on Assigned Tags" category encompasses every system that use as semantic information a set of user-created Tags. Based on the studied systems, we may conclude that in this kind of applications the Tags can either be simple words assigned to files, with no relationship between them whatsoever, or words related to each other. These relationships can be of "inclusion" or "equivalence", also being assigned by the user.

However, and despite they allow to manually assign Tags to files, this kind of systems cannot automatically extract attriutes from files' properties, content or context.

2.4 Hybrid Systems

In order to fill the gaps of each category, hybrid systems exist, systems that possess characteristics from the three semantic file system categories. OSMOSIS can thus be considered a hybrid system, since it automatically extracts Tags from the files' attributes and content ("Content-based Semantics"), allows to (manually) define relationships between files ("Context-based Semantics") and allows to manually assign Tags to files or directories ("Semantics Based on Assigned Tags").

3 Architecture

The OSMOSIS system's architecture can be divided in two major components, corresponding to the two main work goals: OSMOSIS-SFS (Server) and OSMOSIS-RFID (PDA). It is important to refer that the OSMOSIS-SFS system can work as a semantic file system without a connection to OSMOSIS-RFID. The latter, however, needs OSMOSIS-SFS to function, since all the semantic information is on the server side. Thus, the whole functionality can only be used if communication to the server is available.

3.1 OSMOSIS-SFS Architecture

Figure 1 presents the module view of the developed Semantic File System architecture. We now explain each module's responsibilities.

As we can verify, OSMOSIS-SFS's architecture is composed by the following modules: `Applications`; `Tracer`; `File System`; `Content Analysis`; `User-created Tags`; `Database`; `Semantic Search Engine`; `User Interface`; `PDA Interface`;

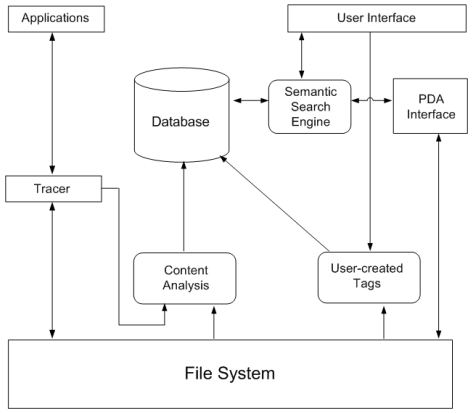

Fig. 1. Module view of the OSMOSIS-SFS architecture.

The `Applications` are all the programs existing in the PC that can access and modify files. These, while not being exactly a module of this system, are represented here so the execution flow, generated by file modification, can be understood.

The `Tracer` module monitors every file system access and, in this way, every time an application creates, modifies, renames or deletes files and directories, Tracer will register that action and alert the `Content Analysis` module of the fact.

The `File System` referenced in the figure is not a new file system, but the one existing natively in Microsoft Windows. Once again, this is not an OSMOSIS-SFS module, it is only represented in the figure so it can be understood that the file system can still be used normally, without interference from our application.

All functionality of the `Content Analysis` module is based on the information passed to it by the `Tracer`. It is responsible for extracting Tags from files' metadata and text files' content, and updating the database.

The `User-created Tags` is responsible for receiving and executing user requests to add Tags to a file or directory.

The `Database` works as a repository for all the semantic information needed for the correct operation of the OSMOSIS system.

For the collected semantic information to be useful, it is necessary to supply a search system to the user. This functionality is thus made available through the `Semantic Search Engine` module, where the whole search algorithm for the OSMOSIS system is implemented.

The `User Interface` module has as its main function to receive the user-submitted requests and present results (i.e. search results).

Lastly, the `PDA Interface` module will receive and answer the various requests from OSMOSIS-RFID. Essentialy, this module works as the server, for it is to it that the PDA will establish a connection and all requests are made.

3.2 OSMOSIS-RFID Architecture

The OSMOSIS-RFID architecture is substantially simpler than that of OSMOSIS-SFS, due to not extracting files' attributes, not allowing the manual Tag insertion, not saving semantic information and not possessing its own search mechanism.

Figure 2 presents the OSMOSIS-RFID architecture, and afterwards a description of each module is given.

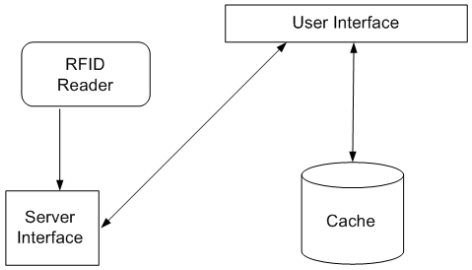

Fig. 2. Module view of the OSMOSIS-RFID architecture.

As we can see in figure 2, the OSMOSIS-RFID is composed by the following modules: `RFID Reader; Cache; User Interface; Server Interface;`

The `RFID Reader` is the module responsible for reading RFID Tags and, whenever that happens, must inform the Server (where OSMOSIS-SFS is installed).

In order to increase the OSMOSIS-RFID performance, a `Cache` was created in the PDA, that basically works as a repository where results from previous searches are kept, and may be used in future searches.

The `User Interface` existing in the PDA has as its main function to receive the user-submitted requests and present results (for instance, search results).

Finally, the `Server Interface` module is responsible for handling every detail of the communication with the server, sending the requests for OSMOSIS-SFS and processing the received response.

4 OSMOSIS System Evaluation

To measure OSMOSIS's performance and usability, a set of tests were performed on the developed application. These tests primarily intend to verify the system's usability (qualitative tests), and secondly the OSMOSIS system's performance (quantitative tests).

The qualitative tests intend to assess the system's general usability or, in other words, how useful are the functionalities made available by OSMOSIS. For these tests, we asked several users to perform a set of tasks in our system, and in the end we registered each one's comments. Results show that, generally, our

system fulfills the main objectives for which it was developed. However, some limitations were identified in the user interface, and some improvement is needed in the existing functionality.

Finally, we performed a set of quantitative tests in order to measure OSMOSIS's performance, namely the duration of critical tasks and the memory used by the application. Based on the obtained results for OSMOSIS-SFS, we may conclude that the indexing time of a file does not affect the global system performance, the memory consumption is not elevated, the space semantic information occupies in the hard drive does not affect computer performance and that our system's search mechanism is substantially faster than Microsoft Windows's "traditional" search. Analyzing the results for the OSMOSIS-RFID, we can state that 1 second is a very reasonable value for performing any of this system's functionalities, and that the use of a cache substantially increases its search performance.

5 Conclusions and Future Work

In this work we have approached the semantic file system theme, and in which way they can be useful for interaction with our own PC. Besides, we have discussed, implemented and evaluated a semantic file system (OSMOSIS-SFS), as well as a system (OSMOSIS-RFID) that allows to integrate real objects into the virtual world. These two "subsystems" are part of the global goal of this work, the OSMOSIS system.

At the end of this work we can state that we have developed a semantic file system, working on a layer above the native Windows file system. In other words, while the way users traditionally navigate between files remains the same, we supply a way to semantically explore the file system. Besides, we have managed to implement a system that allows to assign RFID Tags to physical object or locations (related to virtual files) and afterwards use that information to help finding the objects, or even to directly visualize the file related to the object.

The developed solution, while presenting very positive results (whether qualitative or quantitative), can always be object of improvement. These may range from optimizing the current system to the addition of new functionality, from which we can highlight:

- Currently, our system does not assign different levels of importance to each file's Tags, what would be useful for the search engine. Essentially, we would obtain a better search result ordering. It would be possible, for instance, to assign greater importance to Tags for which a greater number of searches are performed, and even implement a kind of garbage collector for Tags, extracted automatically, and for which no searches are performed for a long time.
- Advantage can be taken of the files' use context as a source of semantic information. Other than explicit relationships between files defined by the user, the system can, for instance, define automatically this kind of relationship when a file access pattern is detected.

– The OSMOSIS architecture can be adapted in order to be able to be used in a corporate environment as, for instance, an insurance company. Due to the large quantity of information existing in that kind of environments, our system would allow a greater ease in document searching, whether at a virtual level (through OSMOSIS-SFS) or at the physical level (through OSMOSIS-RFID).

References

1. A. Ames, C. Maltzahn, N. Bobb, E. L. Miller, S. A. Brandt, A. Neeman, A. Hiatt, and D. Tuteja. Richer file system metadata using links and attributes. In *Proceedings of the 22nd IEEE / 13th NASA Goddard Conference on Mass Storage Systems and Technologies*, pages 49–60, Apr. 2005.
2. Beagle. Website. http://beagle-project.org/Architecture_Overview.
3. S. Bloehdorn, O. Gorlitz, S. Schenk, and M. Volkel. TagFS-Tag semantics for hierarchical file systems. In *Proceedings of the 6th International Conference on Knowledge Management (I-KNOW 06)*, pages 6–8, 2006.
4. C. M. Bowman, C. Dharap, M. Baruah, B. Camargo, and S. Potti. *A File System for Information Management*. Pennsylvania State University, Dept. of Computer Science and Engineering, College of Engineering, 1994.
5. S. Fertig, E. Freeman, and D. Gelernter. Lifestreams: an alternative to the desktop metaphor. In *CHI '96: Conference companion on Human factors in computing systems*, pages 410–411, New York, NY, USA, 1996. ACM.
6. D. K. Gifford, P. Jouvelot, M. A. Sheldon, and J. W. O. Jr. Semantic file systems. *ACM SIGOPS Operating Systems Review*, 25(5):16–25, 1991.
7. O. Gorter. Database file system - alternative to hierarchy based file systems. *University of Twente, the Netherlands*, 2004.
8. D. R. Hardy and M. F. Schwartz. Essence: A resource discovery system based on semantic file indexing. In *Winter USENIX Technical Conference*, pages 361–374, 1993.
9. D. Ingram. Insight: A semantic file system.
10. A. W. Leung, A. Parker-Wood, and E. L. Miller. Copernicus: A scalable, High-Performance semantic file system. Technical Report UCSC-SSRC-09-06, University of California, Santa Cruz, Oct. 2009.
11. U. Manber and S. Wu. GLIMPSE: a tool to search through entire file systems. In *Proceedings of the USENIX Winter 1994 Technical Conference on USENIX Winter 1994 Technical Conference*, pages 4–4, Berkeley, CA, USA, 1994. USENIX Association.
12. C. Marshall. Birch: A metadata search file system.
13. P. Mohan, V. S, and D. A. Siromoney. Semantic file retrieval in file systems using virtual directories. In *13th Annual IEEE International Conference on High Performance Computing*, 2006.
14. H. B. Ngo, C. Bac, F. Silber-Chaussumier, and T. Q. Le. Towards ontology-based semantic file systems. In *Research, Innovation and Vision for the Future, 2007 IEEE International Conference on*, pages 8–13, 2007.
15. Oxygen. Website. http://oxygen.lcs.mit.edu/KnowledgeAccess.html♯web.
16. B. Schandl. SemDAV: a file exchange protocol for the semantic desktop. In *2nd Semantic Desktop and Social Semantic Collaboration Workshop at the ISWC 2006*, Athens, GA, USA, Nov. 2006.

17. C. A. N. Soules and G. R. Ganger. Toward automatic context-based attribute assignment for semantic file systems. *Parallel data laboratory, Carnegie Mellon University. June*, 2004.
18. C. A. N. Soules and G. R. Ganger. Connections: using context to enhance file search. *ACM SIGOPS Operating Systems Review*, 39(5):119–132, 2005.
19. Tagsistant. Website. http://www.tagsistant.net/.
20. Z. Xu, M. Karlsson, C. Tang, and C. Karamanolis. Towards a semantic-aware file store. In *Proceedings of the 9th conference on Hot Topics in Operating Systems-Volume 9*, pages 31–31, Berkeley, CA, USA, 2003. USENIX Association.