# Collaborative Management of Global Directories in P2P Systems

Christopher Peery, Francisco Matias Cuenca-Acuna, Richard P. Martin, Thu D. Nguyen
{*peery, mcuenca, rmartin, tdnguyen*}*@cs.rutgers.edu*

Technical Report DCS-TR-510
Department of Computer Science, Rutgers University
110 Frelinghuysen Rd, Piscataway, NJ 08854

November 1, 2002

**Abstract.** *To date, much of the research in P2P information sharing has concentrated on methods for key-based or content-based location of shared information. In this paper, we argue that in addition to these techniques, the provision of a global browsable structure is also a critical component since users do not always know the appropriate key or set of keywords to locate needed information. To give substance to our argument, we propose a framework that will provide P2P communities with global browsable directories. Our proposal focuses specifically on how to support the communal organization and management of such a shared global directory in order to provide a coherent structure for users in a community to locate information via browsing.*

## 1 Introduction

A key challenge in building peer-to-peer (P2P) systems is the location of information distributed across a large number of users who can join and leave the online community at will. To address this challenge, much of the initial research in P2P has concentrated on techniques for massively scalable key-based object location (e.g., [25, 30, 22, 13]) and for content search with ranking (e.g. [3, 27]). Experience with the web, however, shows that people rely on a combination of both directories [7, 29] and content search and ranking [10, 1] for different types of information searches. Directories provide structure and organization for browsing while content search and ranking supports the structure-less location of information. Based on this observation, we argue that a critical next step in meeting the information location challenge in P2P systems is to provide a directory structure that allows users to *cooperatively* organize and browse the shared data collection.

One approach to providing a directory structure is to implement a file system [16, 15]. However, file system designers have historically been most concerned with storage issues such as allocation, management, and caching of file blocks rather than supporting the directory as a first class object. Furthermore, file systems embody established storage semantics (e.g., partial writes, appends, and locking) that may not be relevant for P2P systems that do not provide storage services.

Thus, in this paper, we focus instead on exploring the challenges of building a directory structure independent of any storage semantics. We assume that each user shares information by making a portion of his local file system available to the community via a software agent (as is the case for many current P2P systems). Then, we propose to build a unified framework that provides both browsing and search capabilities in a single global directory structure that spans all of the individual users' local directory structures. This effort faces several key challenges: (i) users must be allowed to collaboratively organize and maintain the global directory structure in a fully distributed manner, (ii) users must be able to leverage the flexibility of content search and ranking to assist them in organizing information, and (iii) the framework must assist users to maintain an easily browsable structure over time. To meet these challenges, we intend to imbue our framework with the following mechanisms: (i) a coherent mapping between each user's local directory structure and the global directory structure, (ii) embedded semantic directories embodying content queries [9, 11], and (iii) machine-assisted management of the global directory.

More specifically, in our framework, each user's local directory will be added to the global directory structure through a recursive merging procedure that combines directories with the same pathnames. This merging process permits a collection of loosely cooperating users to organize their local directories into a single shared global structure, allowing all members of the community to benefit from their organizational efforts. Also given the coherent mapping, users should be able to easily reason about the location of files in both the local and global directory structures, allowing for easy transition between the two.

The provision of semantic directories [9] will allow members to establish clusters of related information of interest to the community. Semantic directories are directories that are mapped to content queries and populated by files that are deemed as satisfying the respective query. In our framework, semantic directories may be continually updated over time to reflect the changing content of the shared information, providing an effective tool for tolerating the inherent dynamism in P2P communities. Similar to the HAC file system [11], semantic directories could be embedded anywhere in the global directory structure. This deliberate mixing of content and name addressing is intended to provide a powerful tool for organizing information: under this framework, a user could create a directory whose files will be continuously updated by results from content searches, and at the same time allow users to remove irrelevant documents, as well as add relevant ones not found by the search engine. Thus, users will be able to build shared repositories that are organized in both a hierarchical and content-addressable fashion.

Finally, we believe that automated assistance for management of the global directory structure will be critical for large communities. In particular, we will initially be interested in three problems: (i) when a user publishes one or more documents, can the system aid the user in finding the most useful places where these documents should be linked into the existing global directory structure, (ii) can the system monitor the global directory structure and suggest when related documents should be split into multiple branches, and (iii) can the system help to garbage collect the space. For example, as information is removed, portions of the name space may be come sparse, and require consolidation.

We are in the process of realizing our proposed framework in a system call WayFinder. In the remainder of the paper, we first outline the basic design of this prototype, addressing the technical challenges of building a coherent read/write directory structure on top of a dynamic P2P environment. Then, we speculate on how we might leverage techniques from knowledge representation and machine learning to attack the problems of automated assistance.

# 2 Preliminary Design

We are currently designing and prototyping WayFinder on top of PlanetP [4, 2], a P2P content addressing published/subscribed infrastructure. At heart, PlanetP supports the location, rank, and retrieval of published documents whose content matches a particular query. While WayFinder is not critically dependent on PlanetP, it does leverage PlanetP's particular strength: a globally ranked content search capability based on the TFxIDF vector-space ranking algorithm [23]. This capability not only allow WayFinder to implement semantic directories, it also allows WayFinder to rank content inside these directories, particular "incoming" information that has been recently published. In the future, we will likely make use of the more advanced features of PlanetP that includes an implementation of group multicast based on gossiping and automatic update notification.

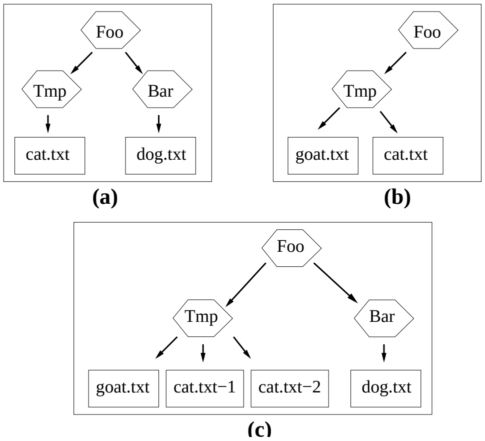

Figure 1: *(a, b) Two local directory structures, and (c) the merged view in the global directory structure.*

## 2.1 Constructing the Global Directory Structure

Currently, WayFinder supports a single hierarchical global directory structure per community. When a user first joins a WayFinder community, he specifies a directory in a local file system that will contain documents he wishes to share. Over time, this directory will be populated with documents published by the user as well as files that are downloaded for hoarding. This directory is the root of what we will refer to as the user's local directory structure.

Given a community of users, WayFinder constructs the global directory structure by overlaying the local directory structures of each user on top of one another, as shown in Figure 1. This overlaying process is performed recursively. A set of directories are merged together if they have the same path from the root[1]. To resolve name conflicts during the merging process, files names in the global directory structures can be extended using a (<local name>, <content hash>) pair to uniquely identify the files as shown in Figure 1(c).

Our method of constructing the global directory structure allows users with similar directory structures to consolidate related files in the same place. It also allows users to have a coherent view of the location of their files in both the global directory structure and their local directory structure. This allows users to browse in a similar fashion

[1] This overlaying approach is similar to the Federated File System's virtual directories [26].

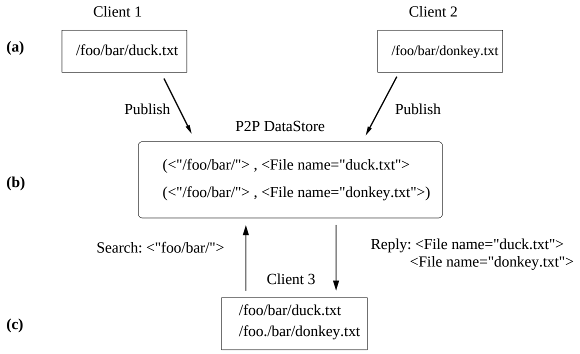

Figure 2: *(a) Two users publishing different files to the same directory; (b) The meta-inodes are published to PlanetP; (c) A merged view of the directory is achieved by querying PlanetP and then contacting the appropriate peers for the meta-inodes.*

while either online or offline. It also ensures that no single user is responsible for maintaining the structure of any given directory or ensuring its availability.

To share a new document, a user must publish the document to WayFinder, specifying the location in the global directory structure where the document should be linked to. Note that because of the coherence between the local and the global directory structure, this publication process can be as simple as moving the document to the appropriate place in the local directory structure. On the publication of a file, WayFinder creates a meta-inode for the file and publishes this meta-inode to PlanetP as shown in Figure 2. Each meta-inode is encoded in XML and includes the file's name, a reference to the file's physical location, a hash of the file's content, and a set of keywords representing the full path from the root to the parent directory of the file.

When a user attempts to browse a directory, a dynamically constructed snapshot, called a directory view, is created as demonstrated in Fig. 2(c). This view in essence represents the current online content of this directory; because PlanetP only maintains term-to-peer mappings in the global index, when composing a directory, we will know that some set of offline peers may have additional documents in the directory but we cannot determine which documents. Further, because of information propagation delays in PlanetP [4], such a view may be several minutes out-of-date with respect to newly published (or removed) documents. WayFinder periodically expires views that the user has browsed, refreshing them again when user returns to the corresponding directories.

We are currently exploring the caching of old views so that once we gain information about a document in a directory, we do not loose sight of that link when the peer hoarding that file goes offline. This is of course difficult in the presence of deletes.

**Future Work.** While we argue that the coherence between the local directory structures and the global directory structure provided by our current composition approach is a powerful communal organizational tool, it has a number of limitations. For example, if users have slightly different names for directories, e.g., movies vs. videos, they will show up as distinct branches in the global directory structure. Thus, we plan to explore virtual mappings between the local directory structures and the global directory structure, perhaps even providing multiple global directory structures that are composed by using different virtual mappings. Maintaining the current coherence will be a particular important issue that must be solved. Further, such virtual mappings would benefit greatly from automatic assistance, which we will discuss in Section 3.

## 2.2 Semantic Directories

As already mentioned, WayFinder supports the creation of semantic directories in the global directory structure. A semantic directory is a directory whose name maps to a content-query [9]. At creation time, WayFinder populates a semantic directory with documents *within the directory's scope* that are relevant to the query. Subsequently, WayFinder periodically re-executes the query to find new relevant documents. Documents inside a semantic directory can be ordered according to PlanetP's relevance ranking.

Scoping for semantic directories is defined by the following rules: (i) A semantic directory that has a normal directory as a parent may have its scope define as either the content of the parent directory or the content of the entire community. (ii) A semantic directory that has a semantic directory as a parent has its scope defined as the content of its parent directory.

**Future Work.** We intend to make semantic directories first class objects, allowing users to manipulate them in the same way that they do normal directories and allow these changes to persist across re-evaluations of the directory's query. While Gopal and Manber have explored many of the issues arising from such a design decision in the HAC file system [11], their work may not extend in a straightforward manner to a P2P environment, especially one where the normal directories themselves are dynamic, perhaps incomplete views of the actual directory structure. Thus, we are currently in the process of working out the details of this design decision.

### 2.3 Propagating Updates

In many P2P file-sharing communities, users do not access the shared data while online. Instead, they typically download (hoard) documents that they are interested in for offline access [19]. Unfortunately, in an environment where the shared documents can change, there is currently no way for the user to know when a hoarded copy has become stale except by manually checking.

To ease hoarding in this case, WayFinder maintains an association between files and their hoarded copies, automatically flowing changes from an altered copy to all other copies. In particular, WayFinder maintains a version vector in the meta-inode of each file. When notified that a file has been changed, WayFinder computes a diff and publishes it to the underlying indexing layer with a new unique version. Other nodes may then learn of this new version by querying a predefined keyword that will return information indicating the set of available versions and their corresponding diffs for a given file.

Periodically, at each peer, WayFinder checks the underlying indexing layer for changes to files that are being hoarded locally. When a diff is located for a file, WayFinder will attempt to update that file. Concurrent updates are automatically merged using a process similar to CVS [5]. If a conflict is identified, the user is notified as in Coda [14].

### 2.4 Consistency

As already discussed, directories are browsable snapshots of the global directory structure, reflecting a portion of the shared information that is currently accessible through the set of online peers. We intend to explore the caching of views to maintain visibility of the entire directory structure, even as peers dynamic (re)join and leave the community. However, a number of difficult issues will have to be resolved, such as when the cached views become stale because of deletion.

The existence of a file in a directory view does not guarantee accessibility—all peers holding copies of that file may be currently offline—nor does it guarantee current, or future, availability unless the user has downloaded a hoarded copy. Users can autonomously add files to any directory at any time.

For files, WayFinder supports a weak, write-any-copy consistency model [12]. The peer performing an update is initially responsible for providing the actual update information to the community as a diff. As the update is applied to other replicas, those respective nodes will themselves become sources for the retrieval of this update.

### 2.5 Security

While WayFinder will never support a security infrastructure sufficient for operation in a hostile environment, clearly, some level of access control is needed, if just to prevent users in a large community from making mistakes with large repercussions. We are currently exploring the design of a security model based on the SDSI framework [21] but do not discuss it here because of space constraints.

## 3 Directory Structure Management

A WayFinder global directory structure represents a joint effort by users in a community to organize the available information. In large communities, this directory structure will be constantly changing as users enter and leave the community and as users publish and remove information. The management of such a dynamic entity will likely be quite onerous if the users must do it manually. Thus, we intend to investigate whether it is possible to leverage techniques from knowledge representation and machine learning to provide automatic assistance for this management task[2].

The structure of any directory structure reflects an ontology, if only implicitly. By ontology, we mean a schema for organizing concepts based on a formal representation of a shared conceptualization in a given domain [18]. Ontologies can be used to reason about the semantic relationships between objects it pertains too. For this reason we are exploring ways of constructing ontologies for both the global directory structure and a user's local directory structure to reason about how the two structures relate to each other.

Although a directory structure can be directly thought of as an ontology, it is one that is extremely poor in semantics. To effectively apply known techniques in knowledge representation, we must enrich the ontologies of the local directory structures. Some approaches for this enrichment include: (i) relying on the users to explicitly provide descriptive text about their organizational structure (as part of each directory, say), (ii) apply information retrieval and/or machine learning techniques to infer information about relationships between files in the same directory of local directory structures, and (iii) use the World Wide Web as a source of information about these relationships. With respect to the last method, for example, the web currently provides access to collections of various large information databases (e.g., [28, 7]). These databases can be used to

[2] We are exploring this as joint work with our colleagues Alex Borgida and Haym Hirsh. We thank them for helping us shape the initial ideas discussed in this section.

expand existing keyword by finding related terms and by learning from the object's location in existing ontologies.

To make this discussion more concrete, one of the automatic method we are exploring to implement a more intelligent merging of local directory structures involve using the TFxIDF vector space measure of similarity. In this approach, when merging a local directory structure into the global directory structure, we use TFxIDF to compute the similarity between a directory of the local directory structure and directories in the global directory structure. In essence, we treat the set of terms derived from the documents in a directory of the local directory structure as a query. Then, we match this query to the content of each directory in the global directory structure, mapping the local directory to some global directory if the similarity is sufficiently high. If there is no significant match, then the local directory is merged into the global directory structure as discussed in Section 2.

We are also exploring techniques for automatic assistance in placing a newly published document in the global directory structure. A project that has successfully addressed somewhat similar goals is the SwiftFile Project, where an intelligent assistant helps users organize their e-mail into folders [24]. This success provides some confidence that our approach holds promise.

Finally, as already noted, the shared content of the community will be constantly changing. As this content changes, portions of the directory structure may become too dense, or, inversely, too sparse to be useful. We plan to explore ways of automatically expanding or collapsing of such portions of the directory structure.

## 4 Related Work

Our work here proposes a framework that supports the construction of a global directory structure from the local directory structures of individual users in a P2P community. A number of works in file systems have also explored the construction of global directory structures over divergent underlying file systems, e.g., [12, 17, 20]. Our work differ from these efforts in two important manners: (1) WayFinder must address the new constraints that arise from its P2P context, and (2) WayFinder is more concerned with effective communal organization of the information shared in the community than coherent accessing files from different underlying file systems.

Several on-going projects are exploring the implementation of file systems on top of P2P indexing networks. Both the Secure Read-Only File System [8] and the Cooperative File Systems [6] are block-level, read-only read file systems. Both allow a form of update for the publisher. The Pasta [15] and Ivy [16] file systems provide read/write file systems where writes are not restricted to only the publisher. Pasta also supports the altering of directory structure structures that belong to other users by creating locally owned copies of necessary index blocks. These copies can then be overlaid to add, or delete files by importing both views of the index blocks. This feature of Pasta is particularly related to WayFinder. However, this approach creates different directory structures for individual users, similar to Prospero [17]. Our approach is different in that we want to present a unified global view to all users of the same community.

## 5 Conclusions

In this paper, we advocate that a browsable global directory structure is a critical component of P2P information sharing infrastructures. To lend substance to this argument, we have discussed our preliminary work on the WayFinder framework, where we intend to integrate a combination of browsing, searching, and automatic assistance to help users of large P2P communities organize and manage their global directory structures. WayFinder's embedding of search capabilities in the global directory structure provides an initial powerful tool for automatic assistance. We believe that there is potentially substantial additional payoff from adapting other techniques from areas such as information retrieval, knowledge representation, and machine learning and are actively pursuing this research direction.

## References

[1] Askjeeves: http://ask.com/, October 2002.

[2] F. M. Cuenca-Acuna and T. D. Nguyen. Text-Based Content Search and Retrieval in ad hoc P2P Communities. In *International Workshop on Peer-to-Peer Computing*, 2002.

[3] F. M. Cuenca-Acuna, C. Peery, R. P. Martin, and T. D. Nguyen. PlanetP: Infrastructure support for P2P information sharing. Technical Report DCS-TR-465, Department of Computer Science, Rutgers University, 2001.

[4] F. M. Cuenca-Acuna, C. Peery, R. P. Martin, and T. D. Nguyen. PlanetP: Using Gossiping to Build Content Addressable Peer-to-Peer Information Sharing Communities. Technical Report DCS-TR-487, Department of Computer Science, Rutgers University, 2002.

[5] Cvs: http://www.cvshome.org/, October 2002.

[6] F. Dabek, M. F. Kaashoek, D. Karger, R. Morris, and I. Stoica. Wide-area cooperative storage with CFS. In *Proceedings of the 18th ACM Symposium on Operating Systems Principles*, 2001.

[7] Dmoz: http://dmoz.org/, October 2002.

[8] K. Fu, M. F. Kaashoek, , and D. Mazires. Fast and secure distributed read-only file system. In *Proceedings of the 4th USENIX Symposium on Operating Systems Design and Implementation*, 2000.

[9] D. K. Gifford, P. Jouvelot, M. A. Sheldon, and J. W. O. Jr. Semantic File Systems. In *Proceedings of the 13th ACM Symposium on Operating Systems Principles*, 1991.

[10] Google: http://www.google.com, October 2002.

[11] B. Gopal and U. Manber. Integrating Content-Based Access Mechanisms with Hierarchical File System. In *Proceedings of the 3th USENIX Symposium on Operating Systems Design and Implementation*. USENIX, 1999.

[12] R. G. Guy, J. S. Heidemann, W. Mak, T. W. Page, Jr., G. J. Popek, and D. Rothmeir. Implementation of the Ficus Replicated File System. 1990.

[13] J. Kubiatowicz, D. Bindel, Y. Chen, P. Eaton, D. Geels, R. Gummadi, S. Rhea, H. Weatherspoon, W. Weimer, C. Wells, and B. Zhao. Oceanstore: An architecture for global-scale persistent storage. In *Proceedings of ACM Architectural Support for Programming Languages and Operating Systems*, 2000.

[14] P. Kumar and M. Satyanarayanan. Flexible and safe resolution of file conflicts. In *Proceedings of the USENIX Winter 1995 Technical Conference*, 1995.

[15] T. D. Moreton, I. A. Pratt, and T. L. Harris. Storage, mutability and naming in pasta. In *International Workshop on Peer-to-Peer Computing*, 2002.

[16] A. Muthitacharoen, R. Morris, T. Gil, and I. B. Chen. Ivy: A read/write peer-to-peer file system. In *Proceedings of the 5th USENIX Symposium on Operating Systems Design and Implementation*, 2002.

[17] B. C. Neuman. The prospero file system: A global file system based on the virtual system model. *Computing Systems*, 5(4), 1992.

[18] B. Omelayenko. Learning of ontologies for the web: the analysis of existent approaches. In *the International Workshop on Web Dynamics*, 2001.

[19] A. Oram, editor. *Peer-to-Peer - Harnessing the Power of Disruptive Technologies*. O'Reilly&Associates, 1st edition, March 2001.

[20] H. C. Rao and L. L. Peterson. Accessing files in an internet: The jade file system. *Software Engineering*, 19(6), 1993.

[21] R. L. Rivest and B. Lampson. SDSI – A simple distributed security infrastructure. Presented at CRYPTO'96 Rumpsession, 1996.

[22] A. Rowstron and P. Druschel. Pastry: Scalable, distributed object location and routing for large-scale peer-to-peer systems. In *Middleware*, 2001.

[23] G. Salton, A. Wang, and C. Yang. A vector space model for information retrieval. In *Journal of the American Society for Information Science*, 1975.

[24] R. B. Segal and J. O. Kephart. Incremental learning in swiftFile. In *Proc. 17th International Conf. on Machine Learning*. Morgan Kaufmann, San Francisco, CA, 2000.

[25] I. Stoica, R. Morris, D. Karger, M. F. Kaashoek, and H. Balakrishnan. Chord: A scalable peer-to-peer lookup service for internet applications. In *Proceedings of the ACM SIGCOMM '01 Conference*, 2001.

[26] L. I. Suresh Gopalakrishnan, Ashok Arumugam. Federated file systems for clusters with remote memory communication. Technical Report DCS-TR-472, Department of Computer Science, Rutgers University, 2001.

[27] C. Tang, Z. Xu, and M. Mahalingam. psearch: Information retrieval in structured overlays. In *HotNets-I*, 2002.

[28] Wordnet: http://www.cogsci.princeton.edu/ wn/, October 2002.

[29] Yahoo: http://www.yahoo.com, October 2002.

[30] Y. Zhao, J. Kubiatowicz, and A. Joseph. Tapestry: An infrastructure for fault-tolerant wide-area location and routing. Technical Report UCB//CSD-01-1141, U. C. Berkeley, 2000.