A New Approach to File Directory Systems

5U－2

Jing-Hua Min, Hidehiko Tanaka*

1 Introduction

Most traditional file directory systems use tree structure. Tree is a simple but weak structure. (1) It fits hierarchical classifications of files well, but precludes non-hierarchical (i.e. orthogonal) classifications. (2) It does not support automatic sharing of files which is often needed by users. (3) It is a single general operating space which becomes more and more complex as a system domain goes larger and larger. So, a new approach is desired. Here we proposed a new directory structure called MSD to solve these problems.

Multi-Space Directory (MSD) is proposed to keep the strength and overcome the weakness of tree directory. MSD is different from tree directory in the following aspects: (1) MSD consists of multiple **directory spaces (DSs)**, each DS is an operating domain; (2) MSD is a **set hierarchy** which fits both hierarchical and orthogonal classifications of files, and supports both vertical and horizontal automatic sharings of files.

We describe the structure of MSD in the next section and summarize its characteristics in the conclusions.

2 Multi-Space Directory (MSD)

2.1 General description

MSD consists of multiple **directory spaces (DSs)** in a **set hierarchy**. Each DS is composed of a **node space (NS)** and a **leaf space (LS)**. A NS is a set of low-level DSs in a **set model**. A LS is a set of files in a **tree model**. **Inheritance links (ILs)** can be set up from a working DS to tool DSs so that the tools in tool DSs can be used in the working DS. **Naming** DSs, NSs, LSs and files follows the structure of MSD. Figure 1 illustrates a typical MSD. The following subsections explains the notions referred above in bold characters.

2.2 DSs and set hierarchy

A DS are a capsule. Multiple DSs are nested capsules. For example, root DS that is the top DS contains DSs

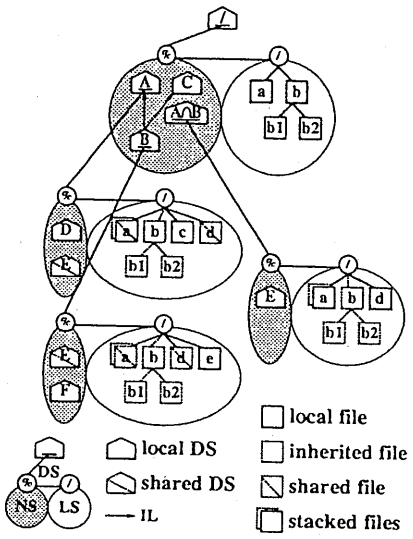

Figure 1: A typical MSD

A, B, C and $A \cap B$; DS A contains DSs D and E; and so on. The DSs in the same NS are called peer-DSs. For example, A, B and C are peer-DSs.

Set hierarchy means that a collection of sets are constructed in such a hierarchy that a low-level set include all elements in its high-level sets and the elements with the same name at different levels are stacked up when appearing in the same place. For example, a and b in the root LS are inherited by low LSs. The as at different levels are stacked. The sets at the same level and with the same parent are called peer-sets. The operation of intersection can be applied to peer-sets. The elements in an intersection are shared by the peer-sets that form it. For example, a and d in the LS of $A \cap B$ are shared by A and B.

2.3 NS and set model

A NS is a set of peer-DSs in a set model. By set model, we means that peer-DSs can been seen as a pair of sets <NS, LS> and two set operations, called intersection and purification, are defined on them.

The intersection of peer-DSs is defined as the intersection of their NSs and LSs respectively. For example,

*Dept. of Electrical Engineering, Univ. of Tokyo

$A \cap B =< \{D, E\} \cap \{E, F\}, \{a, b, c, d\} \cap \{a, b, d, e\} >$
$=< \{E\}, \{a, b, d\} >$
The DSs in an intersection are shared by the peer-DSs that form it. For example, the local E in $A \cap B$ has its mirrors, i.e. the shared Es, in A and B.

Before we define the purification of peer-DSs, we first define the purification of sets. Given sets $S_1, ..., S_i, ..., S_n$, the purification of $S_1, ..., S_i$ is defined as
$S_1 \cap ... \cap S_i \cap = (S_1 \cap ... \cap S_i) - (S_{i+1} \cup ... \cup S_n)$

The purification of peer-DSs is defined as the purification of their NSs and LSs respectively, where the given NSs are all the peer-NSs and the given LSs are all the peer-LSs and the super-LS. For example,
$A \cap =< \{D, E\} \cap, \{a, b, c, d\} \cap >$
$=< \{D\}, \{c\} >$
$A \cap B \cap =< \{D, E\} \cap \{E, F\} \cap,$
$\{a, b, c, d\} \cap \{a, b, d, e\} \cap >$
$=< \{E\}, \{a, d\} >$

2.4 LS and tree model

A LS is a set of files in a tree model. Here the tree model is the same as the one in tree directory. However, LSs are in the background of the set hierarchy of MSD. There are three kinds of files in LSs: local files, inherited files and shared files, as shown in Figure 1. When the files with the same name go to the same place, they are stacked up. The sequence from top to bottom is local file, shared file and inherited files.

2.5 Inheritance link

Tools are the files that are shared by many users. We have the purpose to introduce multiple individual spaces by using MSD. In multiple individual spaces, tools are classified and combined into various tool spaces to support various working spaces, and user's working spaces are connected to corresponding tool spaces. This kind of connections is inheritance link (IL). That is, when ILs are set up from a working DS to tool DSs, the tools in the tool spaces can be used in the working space. A tool space can also be a working space where the tools are developed.

2.6 Naming

We have DSs, NSs, LSs, files and two operations in MSD. Naming them follows the structure of MSD. Table 1 illustrates their names in case of Figure 1.

3 Conclusions

This research focuses on solving the problems of tree directory, such as severe constraints on file naming, weak support to file sharing, and single operating space, by introducing a new structure called MSD. MSD exhibits several outstanding characteristics over traditional tree directory (we have no room to do further explanation here):
(1) MSD supports both hierarchical and orthogonal classifications of files.
(2) MSD supports both vertical and horizontal automatic sharings of files.
(3) MSD supports to build suitable-to-use individual working spaces in a large system domain.
(4) MSD supports to provide better mechanisms for protection and consistency.
MSD can be seen as an extension of tree directory, which conceptually enhances the expression power of directory.

name	*DS*	*NS*	*LS*
	object		
/	/	/%	/%/
	/%/a /%/b	/%/b/b1	/%/b/b2
A	/%A	/%A%	/%A%/
	/%A%/a	/%A%/b	/%A%/c /%A%/d
A∩	/%A/	/%A/%	/%A/%/
	/%A/%/c		
A∩B	/%A/B	/%A/B%	/%A/B%/
	/%A/B%/a	/%A/B%/b	/%A/B%/d
A∩B∩	/%A/B/	/%A/B/%	/%A/B/%/
	/%A/B/%/a	/%A/B/%/d	
D	/%A%D /%A/%D	/%A%D% /%A/%D%	/%A%D%/ /%A/%D%/
E	/%A%E /%B%E /%A/B%E /%A/B/%E	/%A%E% /%B%E% /%A/B%E% /%A/B/%E%	/%A%E%/ /%B%E%/ /%A/B%E%/ /%A/B/%E%/

Table 1: Naming in MSD

References

[1] Peterson, L. L. The profile naming service. *ACM Trans. Comput. Syst.*, 6(4):341–364, Nov. 1988.

[2] Bowman, M., Peterson, L., and Yeatts, A. Univers: An attribute-based name server. *Software-Practice and Experience*, 20(4):403–424, April 1990.

[3] Sechrest, S. and McClennen, M. Blending hierarchical and attribute-based file naming. In *Proceedings of the 12th International Conference on distributed Computing System*, pages 572–580, Yokohama, Japan, June 9-12 1992. IEEE Computer Society and Information Processing Society of Japan.