Automatic Sharing of Multi-Space Directory (MSD)

4H－6

Jing-Hua Min, Hidehiko Tanaka*

1 Introduction

To share objects in multiple users is an absolute requirement. Sharing objects among multiple users can be extremely difficult in a largely scaled directory based on tree structure. There are some other approaches [1, 2] which alleviate the sharing difficulty by providing attribute-based access. Attribute-based access is efficient to discover a set of related objects in large-scaled distributed systems, but difficult to locate one object. There are at least two reasons resulting in the sharing difficulty.

- For the users who want to share some objects elsewhere, they have to know where the objects are placed exactly and go there to access them;
- For the users who want some objects to be shared by their fellows, they have to make efforts to notice or distribute the objects to everyone of their fellows.

How to provide convenient sharing mechanisms is still a challenging issue.

Multi-Space Directory (MSD) is a new directory structure we proposed. It presents several outstanding characteristics over traditional tree-structured directory. **Automatic sharing** is one of them. By automatic sharing, we mean that when an object should be shared by some users, the object automatically appears in the places where those users are. For example, in the case of Figure 1, the objects in */bulletin* should be broadcasted to all users Min, Joe and Jim under */home*, and the objects in */home/joe~jim* are shared by the users Joe and Jim. By automatic sharing, the objects in */bulletin* automatically appear in all user's home */home/min*, */home/joe* and */home/jim*, and the objects in */home/joe~jim* automatically appear in the homes of Joe and Jim. Automatic sharing of MSD is realized by embedding sharing strategies in directory structure.

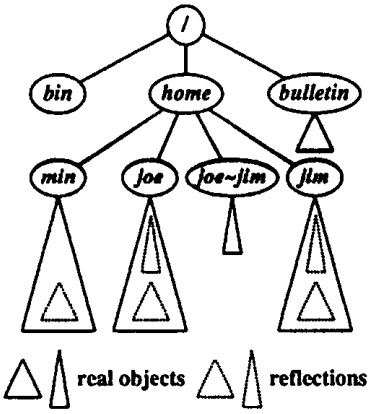

Figure 1: The idea of automatic sharing

2 Automatic sharing of MSD

MSD consists of multiple nested **Directory Spaces (DSs)** and each DS has **Directory Trees (DTs)** of its own (Figure 2). A DS can contain other DSs. The outest DS is called **root-DS**. DTs take tree structure and contain objects other than DSs. The DT which is directly contained in a DS is called **entrance-DT**.

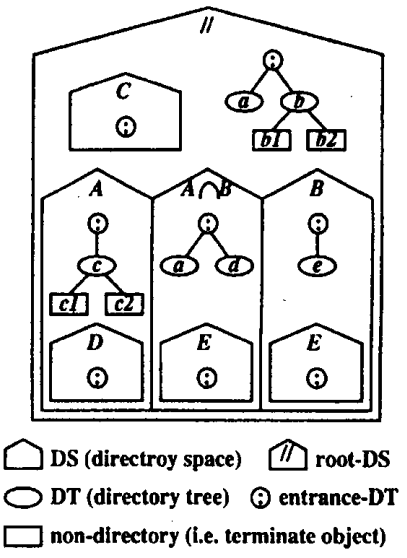

Figure 2: The structure of MSD

*Dept. of Electrical Engineering, Univ. of Tokyo

MSD allows a sub-DS to inherit the objects (except DSs) in its super-DSs, which is called **vertical sharing**. For example, sub-DSs A, B and $A \cap B$ inherit the objects (except DSs) in their super-DS //, such as a and b. It also allows DSs to intersect each other by making intersection-DSs. The objects (including DSs) in intersection-DSs have their reflections appear in the corresponding DSs, which is called **horizontal sharing**. For example, the objects (including DSs) in the intersection-DS $A \cap B$, such as E, a and d, are reflected in DSs A and B. These two kinds of sharings occur automatically under the structure of MSD, which is called **automatic sharing** (Figure 3). So, to share objects in MSD, what it is needed for users to do is just to set up a proper structure of MSD and put the original objects to be shared in the right places. For example, by placing objects a and b into DT // :, E into intersection-DS $A \cap B$, and a (another one) and d into DT $(A \cap B)$:, they can automatically appear in DSs A, B and their DTs. Objects with the same name may be stacked up. For example, objects a's in DTs // : and $(A \cap B)$: are stacked up in DTs A :, B : and $(A \cap B)$:, and objects E's in DSs $A \cap B$ and B are stacked up in DS B.

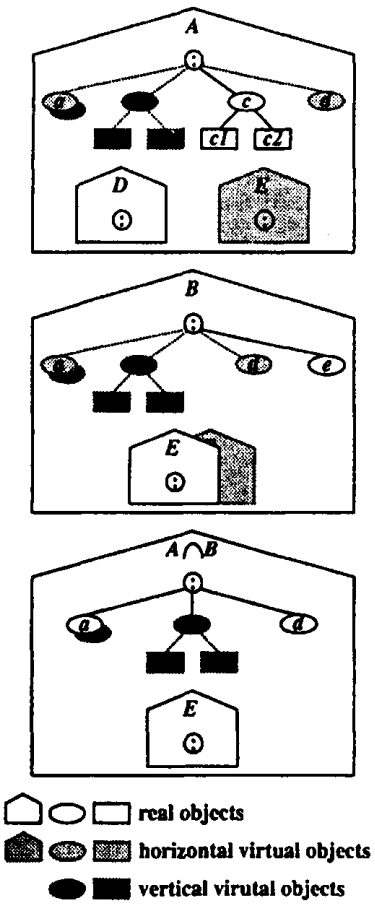

Figure 3: The two kinds of automatic sharing in MSD

Automatic sharing leads to two kinds of objects in DSs and their DTs. The first kind is **real objects** which are real or original. For example, objects A, B, C and $A \cap B$ in DS //, a and b in DT // :, D in DS A, and c in DT A : are real objects. The second kind is **virtual objects** that are the reflections of real objects through automatic sharing. For the reason of two kinds of automatic sharing, virtual objects also have two kinds: vertical and horizontal. **Vertical virtual objects** are the reflections of real objects through vertical sharing. For example, objects a (the bottom one) and b in DT A : are vertical virtual objects. Pay attention to that DSs can not be vertically shared. For example, DS A can not introduce itself and the other DSs B, C and $A \cap B$ within super-DS //. **Horizontal virtual objects** are the reflections of real objects through horizontal sharing. For example, objects E in DS A, and a (the top one) and d in DT A : are horizontal virtual objects.

3 Consistency

Present directory systems can not support automatic sharing. Although you can copy an object to another place, it becomes two same objects not one object, and you must make efforts to keep the two objects same all the time. A symbolic link (or alia in the case of System 7 OS of Macintosh) can be set up to exempt the copy, but it has the following inconsistent problem. The symbolic link becomes invalid when the linked object is renamed or moved, or still remains when the linked object is deleted. So, it is inapplicable to the objects that are unstable in term of their names and locations. Both copies and symbolic links are not created automatically.

In contrast, automatic sharing in MSD guarantees the consistency between real objects and their virtual objects. Any change of the contents and attributes (including name) of a real object samely appears in its virtual objects in the other places.

4 Conclusions

MSD supports automatic sharing by allowing users to embed certain sharing strategies in its directory structure. Automatic sharing mechanism makes it easier for multiple users to share objects in a largely scaled distributed system, and guarantees the consistency between real objects and their virtual objects. We have built a prototype of MSD system. It has proved the feasibility and effectiveness of automatic sharing idea.

References

[1] E. Levy and A. Silberschatz. Distributed file systems: Concepts and examples. *ACM Computing Surveys*, 22(4):321–374, Dec. 1990.

[2] M. Satyanarayanan. A survey of distributed file systems. Technical Report CMU-CS-89-116, University of Carnegie Mello, 1989.