New Special-Purpose Features in the File Manager File manipulation enhancements

Version 7.0 contains a number of specialized functions that give you more control over various kinds of file manipulation. You can use these functions to examine the information in a volume's catalog, to track files on a volume, and to manipulate access privileges on non-Macintosh file systems.

A Quick, Thorough Catalog Search

Version 7.0 introduces the **PBCatSearch** function, a new function for examining a volume's **catalog,** which contains descriptions of all the files and directories on the volume.

A single call to the **PBCatSearch** function can replace a series of indexed calls to the **PBGetFInfo** , **PBHGetFInfo** , or **PBGetCatInfo** function, which all return a collection of catalog information about an individual file or directory. In MFS, you could examine all catalog entries on a volume by calling **PBGetFInfo** repeatedly, using an index to step through the catalog. On an HFS volume, indexed calls to the equivalent function, **PBHGetFInfo** examine the files and directories in only one directory. To examine the catalog information for all files on an HFS volume with **PBHGetFInfo** , or all the files and directories with **PBGetCatInfo** , you have to perform a recursive search through the hierarchy. Especially on a large hierarchical volume, searching the catalog with a series of individual calls can be time-consuming.

The **PBCatSearch** function lets you search the entire catalog with a single procedure call. It compares each catalog entry with a set of specifications you provide, and it gives you a list of all entries that meet your search criteria. For a detailed description of how to use the **PBCatSearch** function, see "Searching a Volume" under **Using the File Manager** .

File IDs

Version 7.0 introduces the **file ID,** a tool for identifying a file that your application may need to find again later. The file ID lets you reference a file through its file number in the volume catalog.

A file number is a unique number assigned to a file when it is created. The **File Manager** can set up an internal record in the volume's catalog that records the filename and parent directory ID of the file with a given file number, establishing the file number as the file's ID and enabling you to reference the file by that number. (For more information about the volume's catalog, see the **File Manager** .)

> **Note:** The file ID is a low-level tool and is unique only on one HFS volume. In most cases, your application should track files using the **Alias Manager**. The **Alias Manager** can track files across volumes. It creates a detailed record describing a file that you want to track, and, when you need to resolve the record later, it performs a sophisticated search. The **Alias Manager** uses file IDs internally.

A file ID is analogous to a directory ID. A file ID is unique only within a volume. A file ID remains constant even when the file is moved or renamed. When a file is copied or restored from backup, however, the file ID changes. Like file numbers, file IDs are unique over time-that is, once a number has been assigned to a file, that number is not reused even after the file has been

deleted.

The file ID represents a permanent reference for a file, a reference that a user cannot change. Your application can store a file ID so that it can locate a specific file quickly and automatically, even if the user has moved or renamed it on the same volume.

File IDs are intended only as a tool for tracking files, not as a new element in file specification conventions. Neither the high-level nor the low-level **File Manager** functions accept file IDs as parameters. If you want to use file IDs, you must use the new functions for manipulating them, described in **Tracking Files With File IDs** under **Using the File Manager** and the functions **PBCreateFileIDRef**, **PBDeleteFileIDRef** and **PBResolveFileIDRef**.

Shared Environments

AppleShare, Apple's file-server application, allows users to share data, applications, and disk storage over a network. System software version 7.0 introduces a local version of AppleShare that allows users to make some or all of the files on a volume available over the network.

Most applications do not have to accommodate shared environments explicitly. As long as you follow the programming guidelines recommended in *Inside Macintosh,* your application should work in a shared environment. If your application directly manipulates files across a network, however, it should use the **File Manager Extensions**.

This section introduces two new **File Manager** features for use in shared environments: volume mounting and manipulating privilege information in foreign file systems.

Remote Mounting

The user mounts remote shared volumes through the Chooser. The version 7.0 **File Manager** provides a set of calls that you can use to collect the mounting information from a mounted volume and then use that information to mount the volume again later, without going through the Chooser.

Privilege Information in Foreign File Systems

Virtually every file system has its own **privilege model,** that is, conventions for controlling access to stored files. A number of non-Macintosh file systems support access from a Macintosh computer by mapping their native privilege models onto the model defined by the AppleTalk Filing Protocol (AFP). Most applications that manipulate files in foreign file systems can rely on the intervening software to translate AFP privileges into whatever is required by the remote system.

The correlation is not always simple, however, and some applications require more control over the files stored on the foreign system. The A/UX privilege model, for example, recognizes four kinds of access: read, write, execute, and search. The AFP model recognizes only read and read-and-write access. If a shell program running on the Macintosh Operating System wants to allow the user to set native A/UX privileges on a remote file, it has to communicate with the A/UX file system using the A/UX privilege model.

System software version 7.0 provides two new functions, **PBGetForeignPrivs** and **PBSetForeignPrivs** , for manipulating privileges in a non-Macintosh file system. These functions do not relieve a foreign file system of the need to map its own privilege model onto the AFP calls.