Changing Times: OpenEdition HFS Files

BY STEVE PRYOR

MVS is changing! One of the highest-profile trends in data processing today is the growth of client/server or "open systems" applications that run on UNIX-based platforms. As these applications become more widespread, powerful, and critical to business operations,their requirements for storage and processor capacity often exceed the capability of smaller hardware systems to support them. In what has become a classic problem, systems that support dozens or hundreds of users relatively well bog down when scaled up to enterprise-wide proportions. Performance suffers, storage capacity is exhausted, and backup techniques are inadequate to manage the volume of data.

High transaction rates and large amounts of data are nothing new to MVS systems programmers and storage administrators, however. The extremely high reliability and high capacity of MVS (now OS/390) systems is the result of many years of experience during which IBM, vendors, and customers painstakingly developed and refined the systems and methodologies needed for successful large-scale operations.

A NEW CHALLENGE FOR THE STORAGE ADMINISTRATOR

What is new is the use of OS/390 to run the applications that previously required UNIX platforms. With the introduction of OS/390 Version 1 Release 2, the OpenEdition component of the operating system provides a full UNIX environment within MVS/ESA (XPG4 UNIX Profile branding). For the user, this means that applications can take advantage of OS/390's strengths in reliability, capacity, and security, while preserving the UNIX environment (the command shell and file system) that they are accustomed to. For the storage administrator, this means a new challenge, that of managing OpenEdition/MVS files in addition to traditional MVS data.

**Omnia mutantur,
nos et mutamurin illis
(All things change,
and we change with them).
— Matthias Borbonius:
Deliciæ Poetarum**

The OpenEdition file system is a hierarchical file system. In a manner familiar to anyone who has ever used Windows or MS-DOS, files are contained within directories, and directories may be contained within other, higher-level directories. The highest level directory is called the root directory. The full set of names required to specify a file within the hierarchical file system is called a path. Thus, the path "/payroll/test/data/june" specifies a file called "june" which can be found in the "data" directory by following the directories from the root ("/") to "payroll" to "test" to "data". (Note that the slashes are in a forward direction, unlike Windows or MS-DOS). The path names for hierarchical files may be, and typically are, specified in lowercase. This is obviously quite a different scheme from "normal" MVS data. Although hierarchical files are usually accessed from UNIX programs or the OpenEdition shell, they can be accessed from MVS, and this has resulted in the introduction of a number of JCL parameters that may be unfamiliar to mainframe programmers. Figure 1 shows how a hierarchical file might be created in a batch job.

The storage administrator has more to be concerned with than a new file naming system, however. He/she may be responsible for installing OpenEdition, setting up the hierarchical file system, establishing allocation, backup, and migration policies for OpenEdition data, and ensuring that disaster recovery procedures are in place for open systems data.

The first task for a storage administrator who must install OpenEdition is to ensure that the SMS subsystem is available and that the ACS routines allow data to be SMS-managed. At least one SMS-managed volume is required to run OpenEdition because all OE "file systems" (directories and the subdirectories and files under them) are stored within SMS-managed datasets created with the DSNTYPE=HFS parameter. An HFS dataset contains an entire "mountable file system." Figure 2 shows an example of allocating an HFS dataset. The dataset name of the HFS dataset is unrelated to the OpenEdition directories and files that may be contained within it. In order for an OE file system to be known to OpenEdition, it must first be logically "mounted" at a "mount point" (an empty directory). The TSO MOUNT command accomplishes this by associating the HFS dataset name with a mount point. See Figure 3.

HFS datasets are an extended type of PDSE, consisting of 4KB blocks that are controlled by the FAMS (File Attribute Management System) component of DFSMS in the same way as PDSEs; however, the internal format of an HFS dataset is completely different from that of a PDSE. Like PDSEs, HFS datasets must be SMS-managed and are limited to a single disk volume, but may be expanded to include up

Figure 1: Allocating an OpenEdition File

```
//ALOC EXEC PGM=IEFBR14
//DD1 DD PATH='/payroll/test/data/june',PATHDISP=(KEEP,DELETE),
// PATHOPTS=(OWRONLY,OCREAT,OEXCL),PATHMODE=(SIRWXU,SIRGRP)
```

Figure 2: Allocating an HFS Dataset

```
//BR14 EXEC PGM=IEFBR14
//DD1 DD DSN=HFS.TEST.DATA,SPACE=(CYL,(40,1,1)),DCB=(DSORG=PO),
// DSNTYPE=HFS,DISP=(,KEEP,CATLG),STORCLAS=STANDARD
```

Figure 3: Logically Mounting a File System (Associating an HFS Dataset With an OE Directory)

Note: This command can be issued only by an OE user with superuser authority.

```
MOUNT FILESYSTEM(HFS.TEST.DATA) MOUNTPOINT('/payroll/test') TYPE(HFS)
```

to 123 extents on that volume. Storage management reporting tools recognize HFS datasets by the presence of the DS1PDSE and DS1PDSEX bits (X'80' and X'20') in the SMS indicators of a dataset's format-1 DSCB. These bits were reserved prior to the introduction of SMS; however, some DASD management programs used these fields for their own purposes in the past. If the SMS address space is being activated for the first time in order to take advantage of open systems data, it may be important to scan the VTOCs to ensure that no residual data resides in the SMS indicator fields.

FACTORS AFFECTING PERFORMANCE

When setting up the OE hierarchical file system, the storage administrator should take into account factors that will affect the performance of the open systems applications. Too many HFS datasets on a single volume can cause I/O bottlenecking, for example, so it may be important to spread the HFS files across devices. Each OE user should have his/her own mountable file system (HFS dataset) to avoid contention between locks required on the file systems by OE processes. The root file system, which is a separate HFS file, should be defined as read-only in the BPXPRMxx member of SYS1.PARMLIB. This will not only improve the I/O performance of the OE system, but will also prevent the root file system from accidentally being corrupted. Finally, it may be useful to put the /tmp directory in a separate HFS dataset on a high-performance volume, since this directory may be subject to heavy use if there are many OE users.

In addition to placement and performance strategies, the storage administrator is likely to be responsible for disaster recovery and compatibility issues when implementing OE. As with any system, it is important to ensure that the correct level of hardware and software will be available at a disaster recovery site. The disaster recovery site should be able to run DFSMS level 1.2 or higher, and any system that shares the HFS files should either be at this level or have toleration PTFs applied. Large installations also need to consider that HFS files can be shared by OE systems across a sysplex only if they are read-only.

Once OpenEdition applications are in place, the storage administrator's task shifts from planning and setup to providing data management services. Space availability, backup and recovery strategies, and cross-system considerations must be addressed. Next month's column will examine some of these issues and the techniques the storage administrator can use to help provide the benefits of the "big iron" to open systems applications. ts

NaSPA member Steve Pryor has more than 15 years of experience in storage management, disaster recovery, software development, and technical support. Steve can be contacted via the Internet at pryor@atlanta.com.