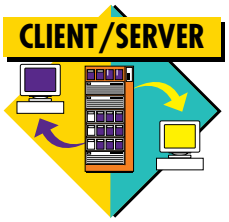
DCE DFS Interoperability in Data Sharing Environments



By Jean E. Pehkonen

As enterprises migrate to the Distributed Computing Environment (DCE) and Distributed File Services (DFS), existing environments may use legacy data or systems, which can make moving to a pure DFS environment impossible or impractical. In these cases, DFS may need to work in environments that contain Network File System (NFS®) or Andrew File System (AFS®). Fortunately, DFS can operate in most of these situations. This article discusses possibilities for coexistence and interoperability of DCE DFS in both NFS and AFS environments.

As DCE environments become the enterprise solution of choice, many DCE applications can provide distributed services such as data sharing, printing services, and database access. As part of the Open Software Foundation's (OSF) DCE, DFS provides filesystem services for these environments. DFS uses Remote Procedure Calls (RPCs) for data transfer, Cell Directory Services (CDS) for naming, and DCE Security for authentication services to provide data sharing services.

In addition to its use of DCE services, DFS itself is rich in features. It provides a uniform global filespace that allows all DFS client users to see the same view of the filespace. It caches filesystem data at the client for improved scalability and performance by reducing network traffic to file servers. DFS also supports advisory file locking.

One DFS feature is the ability to export the operating systems's native filesystem. In AIX, the native filesystem is the Journaled File System (JFS). In addition, DFS also provides its own physical filesystem—the DCE Local File System (LFS). The DCE LFS supports DCE Access Control Lists (ACLs) on files and directories for securing access to data and advanced data management capabilities such as replication and load balancing.

NFS and DFS Differences

Although DCE DFS and NFS are both distributed filesystem products, the two products differ in their data-sharing models and semantics. NFS relies on a peer-to-peer client/server model between machines, while DFS operates within an autonomous administrative unit called a *cell*. Semantically, NFS is a stateless system implying that the NFS server does not maintain information about the client requests. DFS maintains the state of file and directory information to provide UNIX single-site semantics by using an internal token manager.

Administration of a DFS environment is centralized—a system administrator can complete the majority of filesystem administration from a single system within the cell. Administration of an NFS environment, on the other hand, often involves updating information on each NFS client system in the environment.

Given these differences, the two products seem unlikely to coexist or interoperate in one environment. However, there are several scenarios where this may be desirable and indeed possible.

Coexistence with NFS

Both DFS and NFS maintain the concept of "exporting" data. In both contexts, a filesystem is

Jean E. Pehkonen

made available to which client systems can remotely gain access. Clients then access this data by mounting the filesystem remotely.

One difference between NFS and DFS becomes apparent in this area. Once the data is exported, each NFS client must remotely mount that filesystem in order to gain remote access to it. Because DFS uses a uniform global filespace, a DFS client only needs to mount the root of the filespace (/...) to get access to all data exported through DFS. The DFS system administrator is responsible for organizing the filespace and determining where the mount points for the data will be located. Once the administrator has set up the tree structure, all clients see the same view.

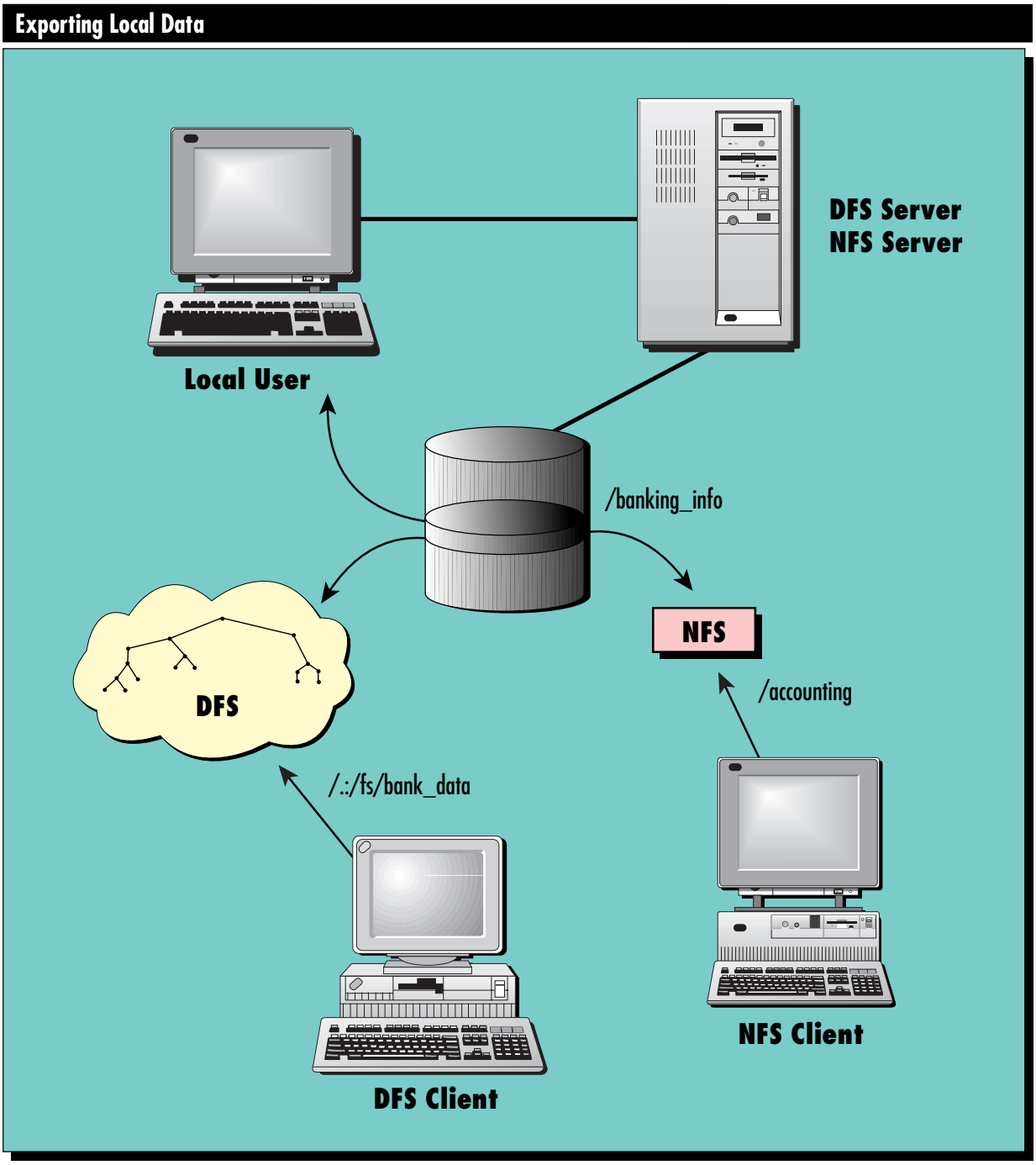


Figure 1. Exporting local data

On the server system, native JFS filesystems can be exported to NFS for use by NFS clients. At the same time, the filesystem can also be exported to DFS for use by DFS clients. Figure 1 shows this scenario. A RISC System/6000 (RS/6000) system is providing data through NFS and DFS simultaneously. The local filesystem, `/banking_info`, has been exported to both DFS and NFS so client systems in the environment can access the data. If the system is running the NFS client software, the user can use the data through the NFS mount point, `/accounting`. DFS clients running DCE and DFS software can use the data by accessing it through the path `/.:/fs/bank_data`.

In addition to being accessible to NFS and DFS client users, the filesystem still remains accessible to any local users who may have access to the RS/6000 through a local account login. These users can access the data using the local path `/banking_info`.

Because NFS is a stateless system, NFS clients may not necessarily see data changes from the server immediately. Data will be maintained consistently among DFS clients and local users due to UNIX single-site semantics.

Interoperability with NFS

The DFS filespace (`/...`) can also be exported through NFS, which allows NFS clients to mount and access DFS even though they are not running a DFS client. This may be advantageous in environments where DFS client software is not available for all hardware platforms

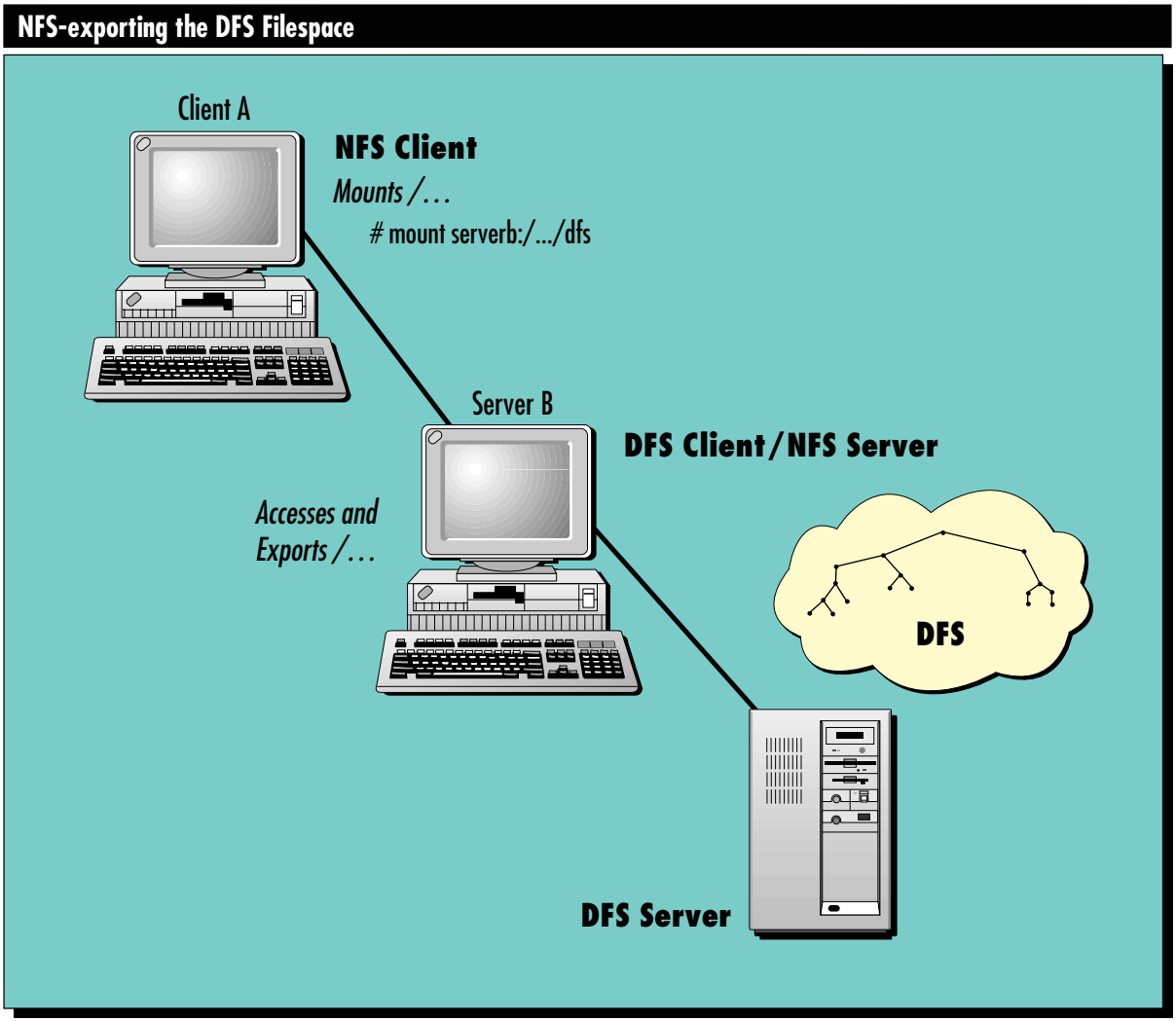


Figure 2. NFS-exporting the DFS filespace

in the enterprise, such as DOS-based PCs. Of course, the platform does need an NFS client software package available.

Figure 2 shows this scenario. A DFS client system within the cell can access the DFS filespace. In addition, the DFS client can run NFS server software and make the DFS filespace available for mounting by adding it to the `/etc/exports` file. An entry such as `/...` will allow this. NFS clients may, in turn, mount the filesystem the same way they mount other remote filesystems. The NFS client will then have access through the path, `/dfs`. True DFS clients will have access to the same information through the path `/.:/fs`.

DFS provides a higher level of data protection by using ACLs on both files and directories for data stored within DCE LFS filesets. (DCE ACL support is not available for JFS filesystems exported to DFS.) This level of protection usually means that a user must be authenticated as a DCE principal to gain access to the DFS filespace. NFS clients cannot be DCE-authenticated, so any access they gain will be as an unauthenticated user. Figure 3 shows an example of an ACL set to protect the DFS filespace to allow access only by authenticated users.

```
# SEC_ACL for /.:/fs:
# Default cell = /.../dfsfvt.cell.austin.ibm.com
user_obj:rwxcid
group_obj:r-x--
other_obj:r-x--
```

Figure 3. ACLs on the DFS filespace

```
# SEC_ACL for /.:/fs:
# Default cell = /.../dfsfvt.cell.austin.ibm.com
mask_obj:rwx-id
user_obj:rwxcid
group_obj:r-x--
other_obj:r-x--
any_other:rwx-id
```

Figure 4. ACLs allowing write permission for NFS client

```
$ dfsiauth -add -r blizzard.austin.ibm.com -i 1022 -u jean
Enter Password:
dfsiauth: <blizzard.austin.ibm.com, 1022> mapping added
DCE principal: jean
```

Figure 5. Establishing an authentication mapping

In this situation, any NFS clients wishing to gain access to the DFS filespace will be denied access since they do not have any DCE authentication associated with them. For NFS clients to gain access, the `any_other` ACL can be added to the DFS objects. Figure 4 shows an `any_other` ACL on a DFS directory, which allows an NFS client to have write permission to the data in that directory. The `any_other` ACL allows unauthenticated users—those who do not match any of the other ACLs—access to the file or directory. Adding the `any_other` ACL opens the filespace for NFS clients, but it also allows any unauthenticated DCE user to access the data. This may not be desirable, depending on the confidentiality of the data being stored.

NFS/DFS Gateway

An additional product introduced in AIX DCE Version 1.3 allows NFS users authenticated access to DFS. The NFS/DFS Authenticating Gateway allows NFS users to maintain an authentication mapping to DFS so they can access data without using the `any_other` ACL. Data in DFS can be maintained under ACL control, but NFS clients can still access the data.

The NFS protocol maintains security by using a hostname, userid pair for each client. For example, user jean with userid 1022 on the system, blizzard.austin.ibm.com will be sent to an NFS server. However, this pair means nothing to DFS, which maintains accessibility based on DCE principals and DCE ACLs. The NFS/DFS Gateway enables a mapping to be established between the NFS information and the DCE information. Figure 5 shows a mapping being established using the `dfsiauth` command.

Once this mapping has been established, requests from the NFS client, blizzard.austin.ibm.com, and the user 1022 on that system will be authenticated as the DCE principal, jean. Any files or directories for which the DCE principal, jean, has accessibility can be accessed by that particular NFS client.

AFS/DFS Differences

Conceptually, DFS is similar to AFS. Like DFS, AFS maintains a uniform global filespace, centralized administration, and information caching at the client systems. However, AFS

does not offer all the same features as DFS. DFS features such as ACLs on files and directories (AFS maintains ACLs on directories only), UNIX single-site semantics, and file locking are among some features that are not available for AFS. AFS cannot export the operating system's native filesystem. This limitation may inhibit the interoperability allowed for NFS and DFS clients; however, some accessibility is still possible in these types of data sharing environments.

Coexistence with AFS

On client systems, it is possible to run the software for the DFS client and the AFS client simultaneously. Because each product uses a local disk cache (or memory cache) for storing information brought across the network, two separate caches must be available on the system. Each product maintains information in its cache differently, so it is not possible for both products to share one filesystem for the local cache. This coexistence may be advantageous in situations where data needs to be migrated from an existing AFS cell to a new DCE DFS cell. Since the data from both cells is accessible at one system, data can be moved from one cell to another by using AIX `cp` and `mv` commands. Once all data from AFS has moved to DFS, the AFS cell may be unconfigured.

mounted	mounted over	vfs
AFS	/afs	afs
DFS	/...	dfs

Figure 6. DFS and AFS clients

Figure 6 shows the partial results of an AIX `mount` command where both DFS and AFS clients are running.

It should also be noted that both AFS and DCE maintain system time by keeping the system clock in sync against a server system in the cell. However, if the servers are running on different machines, they may not be maintaining the same clock time. Therefore, AFS on the client system will attempt to synchronize its time while DCE using DTS time services is synchronizing its time to another server. This scenario may lead to confusion at the client system. So it is recommended that for systems running DFS and AFS clients, the `afsd` process should be invoked with the `-nosettime` option. This option will stop AFS from maintaining time and allow the DCE time services to provide one-time service for the client system.

DFS and AFS fileserver system software may be run on one system. However, because AFS does not allow exportation of native filesystems, it is not possible for AFS and DFS to serve the same data in a single filesystem. Data can exist in the JFS or DCE LFS format for DFS, and may also be duplicated in AFS partitions to be served to AFS clients on one server system.

Due to their heredity from Transarc® several administrative commands in AFS and DFS have maintained the same names. This may cause confusion for system administrators if they do not know which command they are using. This problem can be avoided by specifying the command using its full pathname. The DFS commands reside in the `/usr/bin` directory on AIX systems. The AFS commands reside in the `/usr/afs/bin` directory. Those commands with overlapping names include the `bos` command, the `bosserver`, `upclient`, and `upserver` daemons, the `udebug` command, and the `scout` monitoring utility.

The NFS/DFS Authenticating Gateway allows NFS users to maintain an authentication mapping to DFS.

Interoperability with AFS

With their AFS 3.3 client software, Transarc added support to allow AFS clients access to DFS cells as well as to AFS cells. Accessing a DFS cell is nearly transparent, from the user's point of view. The DFS cell represented as `/.../<cellname>/fs`, for example, may be accessed as `/afs/<cellname>`. Figure 7 shows an AFS 3.3 client running on a system. Users on this system may access the AFS cell, `/afs/xyz.com`. They may also access the DFS cell using the path `/afs/dcexyz.com`. DFS users will access the DCE DFS cell using the path, `/.../dcexyz.com/fs`. AFS client users must use a special login command called `dlog`, which allows them to authenticate to DCE and access the DFS filespace. The AFS `fs` command has also been extended to allow AFS clients to list and modify DCE ACLs on files and directories in the DFS filespace.

In order to provide interoperability for the AFS 3.3 clients on the server side, the DFS servers must run a daemon named `adapt`, a protocol translator. This daemon intercepts requests from AFS clients and redirects them to DFS. Without this daemon, it is not possible for AFS

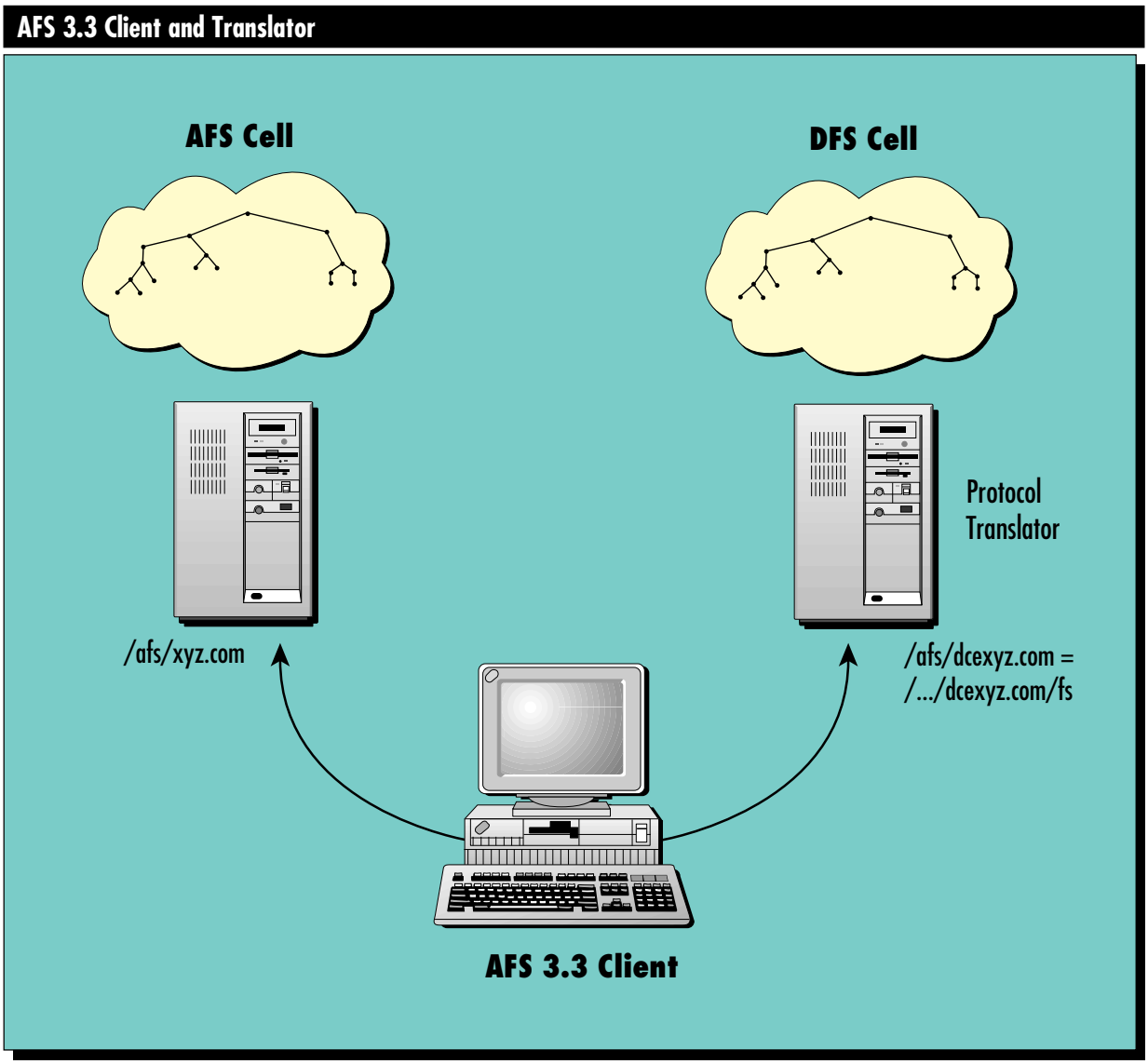


Figure 7. AFS client and translator

clients to access the DFS filespace. It is not possible to run this daemon on a system that is running an AFS server because it will attempt to send AFS requests to DFS, not allowing the AFS server to service the request. This translator is part of Transarc's AFS/DFS Migration Toolkit. In addition to running the translator, the AFS cell administrator must make the DCE DFS cell available by mounting it in the AFS filespace. This is done by executing an `fs mkmount` command for the DFS filespace root.

Summary

The interoperability options described make it possible for AFS and NFS users to use DFS. Legacy data in JFS filesystems can be easily shared between NFS and DFS. With the addition of gateways and translators, it is possible to further smooth the transition to DFS and allow a variety of clients to access DFS data.

Jean Pehkonen, IBM Corporation, 11400 Burnet Road, Austin, TX 78758. Internet: jean@austin.ibm.com. Ms. Pehkonen is a programmer in the DCE DFS Development department of IBM's LAN Systems Division. She has worked on IBM's AIX DFS products for the past five years. She has a BS in Computer Science from the University of Minnesota.