Session 9353 / 9394

VM File Systems Overview

Alan Altmark
IBM Corporation
Endicott, New York

Disclaimer

This presentation provides a comparison and contrast of the various file systems available for use in CMS. Characteristics, administration, programming interfaces, and relative performance are discussed.

References to IBM products, programs, or services do not imply that IBM intends to make these available in all countries in which IBM operates. Any reference to an IBM product, program, or service is not intended to state or imply that only IBM's product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe on any of the intellectual property rights of IBM may be used instead. The evaluation and verification of operation in conjunction with other products, except those expressly designed by IBM, are the responsibility of the user.

The following terms are trademarks of IBM Corporation in the United States or other countries or both:

S/390 VM/ESA IBM AIX OS/390

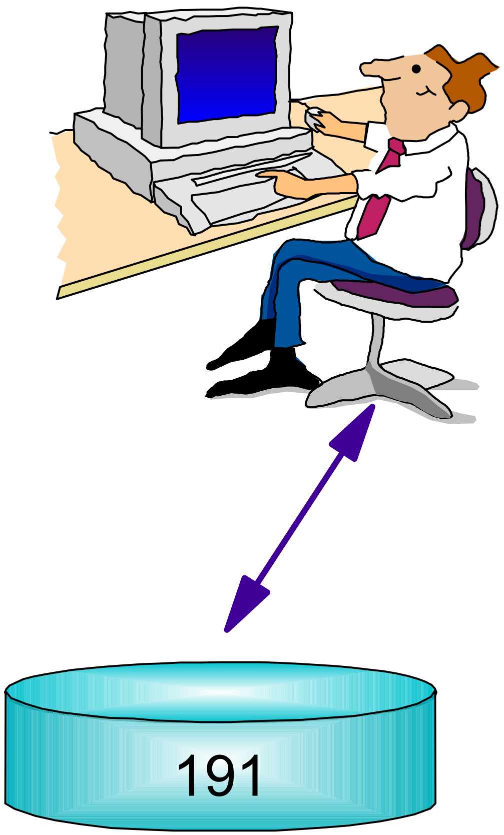
Other company, product, and service names, which may be denoted by double asterisks (**), may be trademarks or service marks of others.

History

- VM available as a product for 26 years!
- Minidisk support since The Beginning
- Shared File System (SFS) introduced in 1988
- Byte File System (BFS) introduced in 1995
- Enhanced Network File System (NFS) support in 1998 and 1999

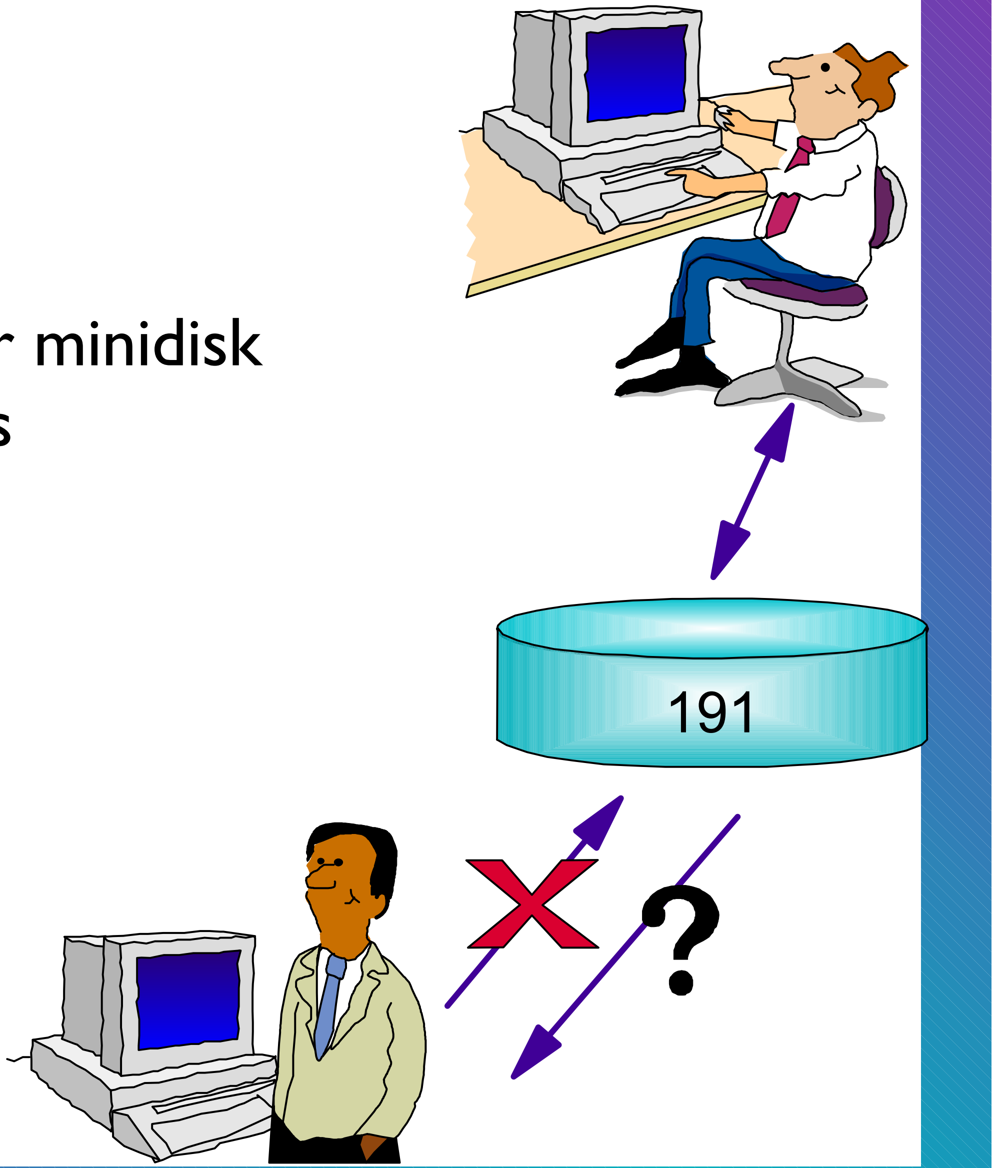
Minidisk

- Direct attachment of disk to user
  - excellent performance
  - writers can cause I/O errors for readers
  - security via minidisk password or ESM
  - mode 0 files are private, not secure!
- Fixed space allocation
  - manual or automated
  - files size limited to dasd size
  - changing size means copying disk



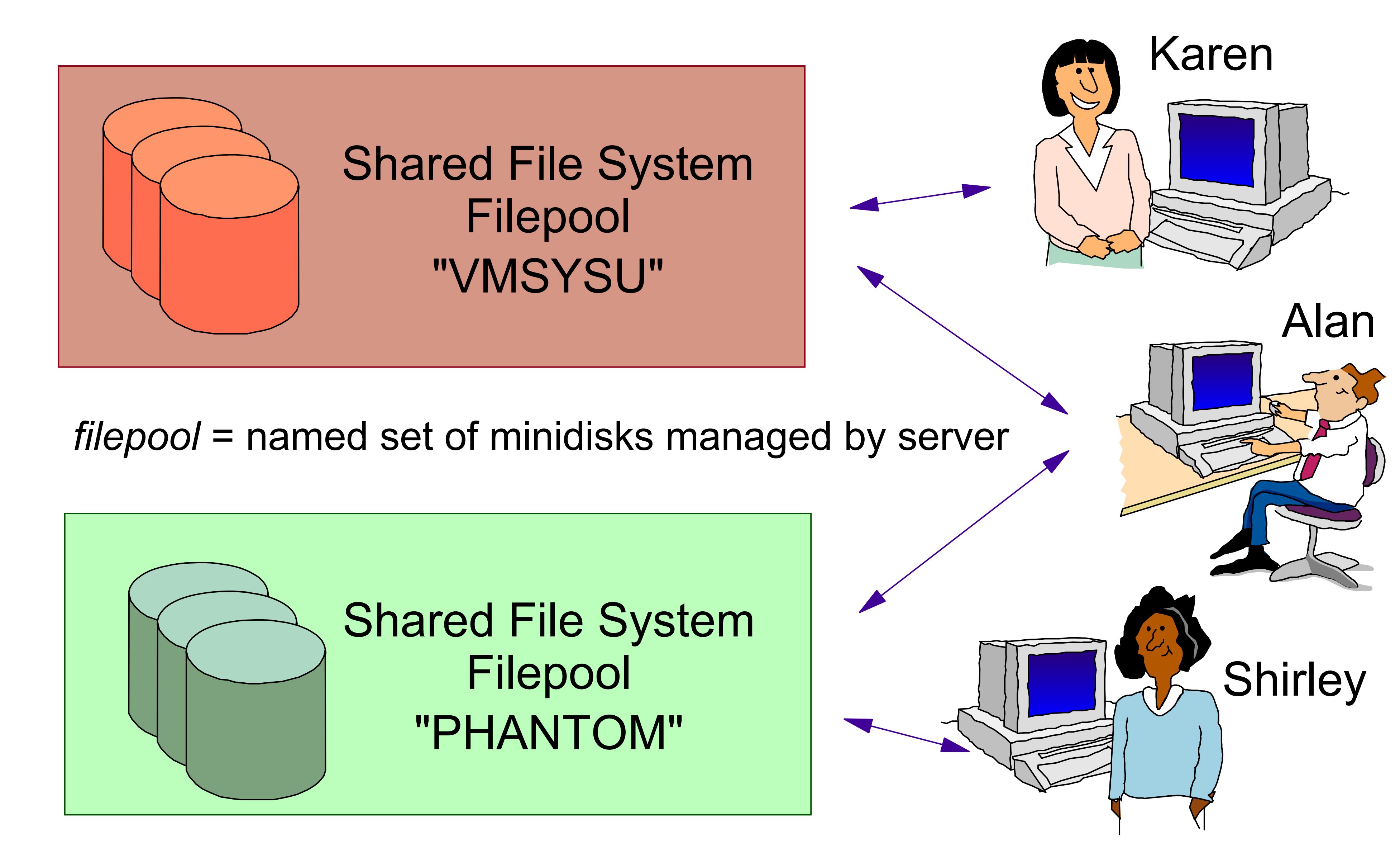
Minidisk

- No subdirectories
- Limited sharing capability
  - one writer, multiple readers per minidisk
  - no data integrity for the readers
  - esoteric programming practices
- Remote access
  - shared dasd
  - NFS



Shared File System

Shared File System (SFS)



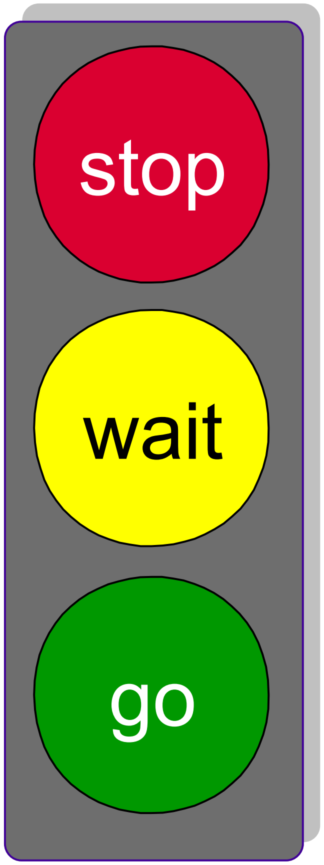
Shared File System Filepool "VMSYSU"

filepool = named set of minidisks managed by server

Shared File System Filepool "PHANTOM"

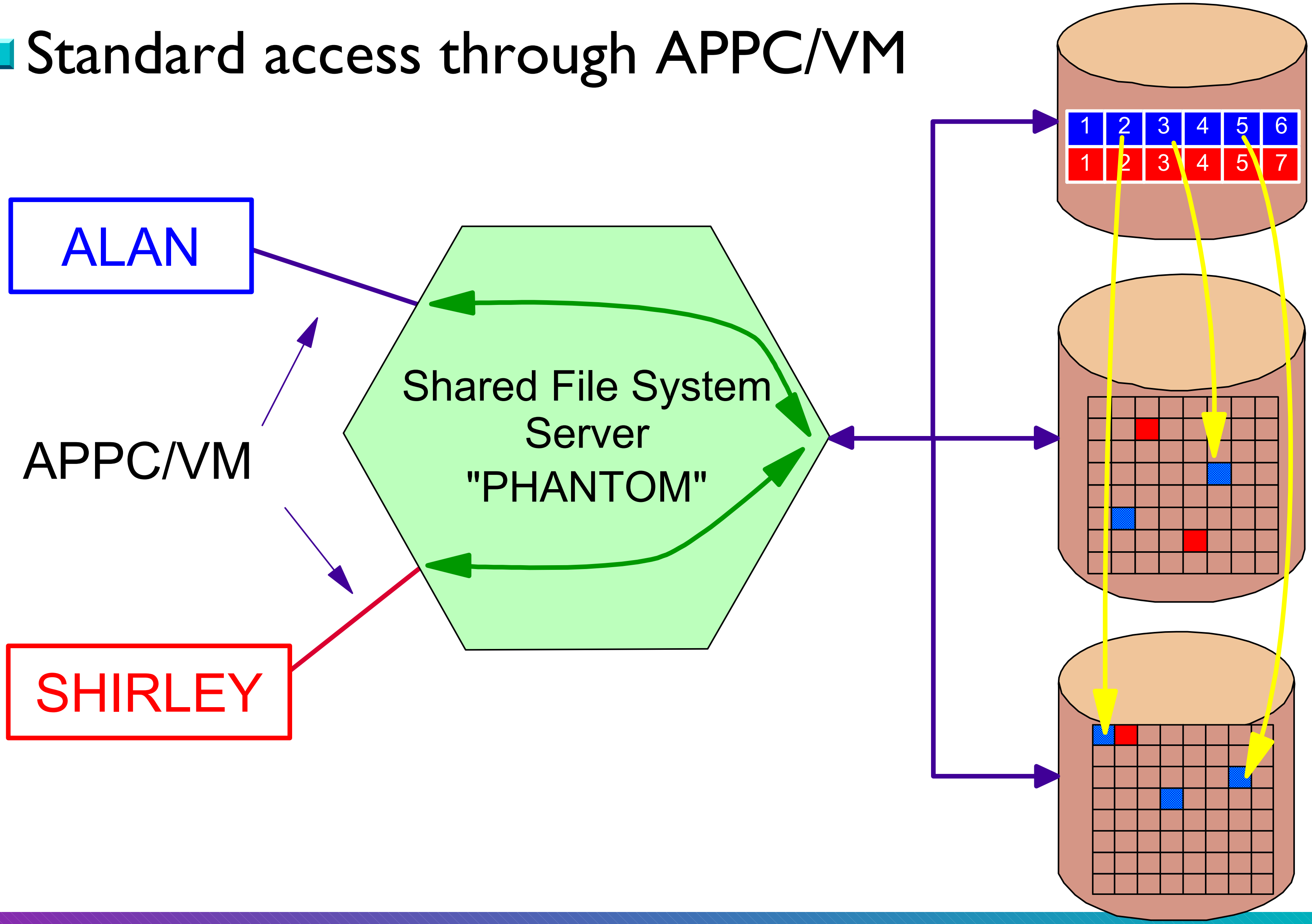
Shared File System

- Server-moderated access to files
  - access control
    - based on VM user ID
    - owner determines permissions
  - space management
  - filepool owned by server
  - file sharing controls
  - backup / restore
- Multiple servers for multiple policies



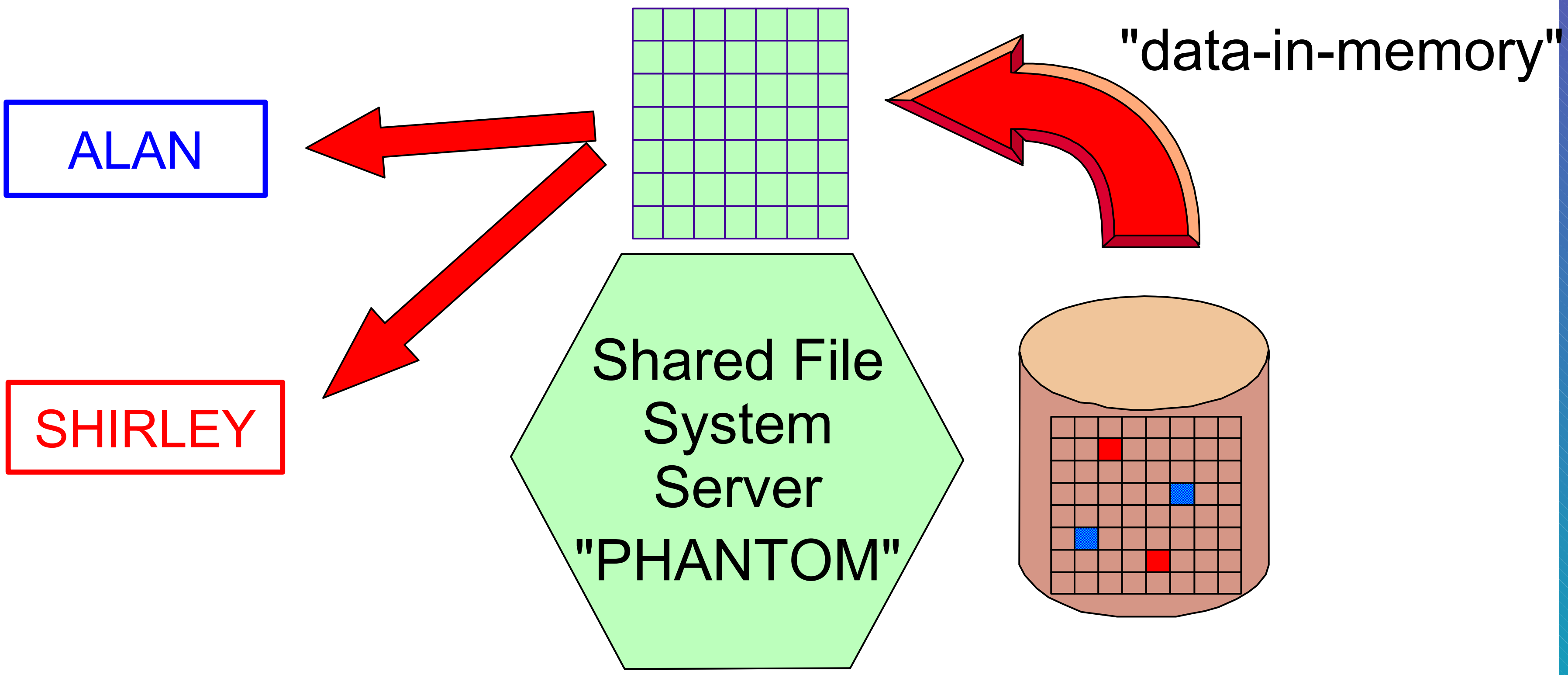
Shared File System

- Standard access through APPC/VM



Shared File System

- VM Data Spaces can be used to avoid trip to server for data
  - data read into memory as needed

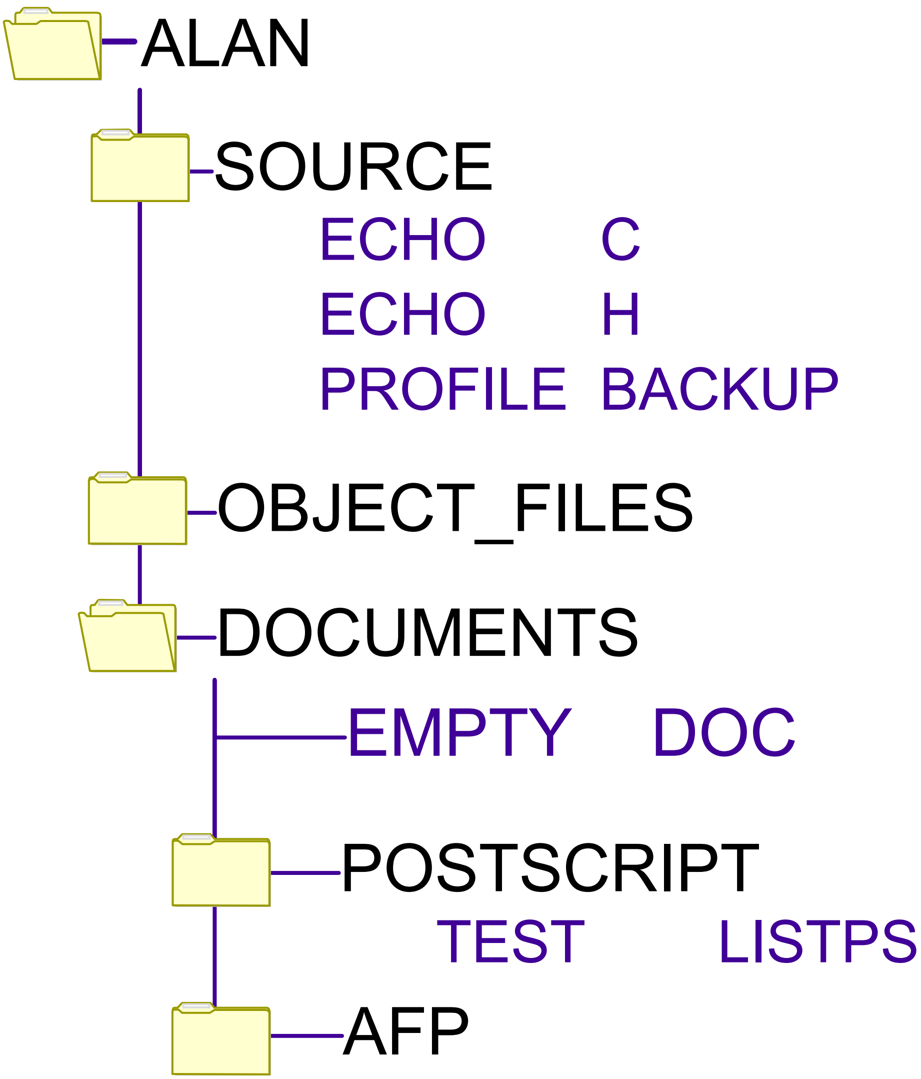


Shared File System

- Hierarchical directory structure
  - File Control
    - File updates shown whenever file is opened
  - Directory Control
    - Access-to-Release consistency for all files
    - Eligible for VM Data Spaces

- File sharing
  - one writer, many readers for a file
  - data integrity
  - no I/O errors

Shared File System File Space



```
ALAN
  SOURCE
        ECHO     C
        ECHO     H
        PROFILE  BACKUP
  OBJECT_FILES
  DOCUMENTS
        EMPTY    DOC
    POSTSCRIPT
          TEST      LISTPS
    AFP
```

```
enroll user alan phantom ( blocks 10000
```

```
> access phantom:alan.source b
> listfile * * b
```

Directories may be empty

Files may be empty, too!

Useful to preserve authorizations

Shared File System Benefits

- Flexible space allocation
  - uses multiple minidisks
  - files may span minidisks
  - blocks allocated as needed
  - can dynamically add minidisks
  - sum of user limits may exceed physical dasd
- May be part of System Managed Storage
  - a better "temp disk" - survives logoff
  - data can expire or be migrated

Shared File System

SHARE
Boston
July 2000

- Remote access
  - VM users via collection or VTAM (AVS, LU 6.2)
  - NFS
- Work units allow explicit commit control
  - programs can explicitly rollback or commit work
  - rollback automatic in case of user abend
  - commit automatic when you return to Ready;
  - Coordinated Resource Recovery (CRR) ensures updates across multiple SFS file pools are all committed or all rolled back

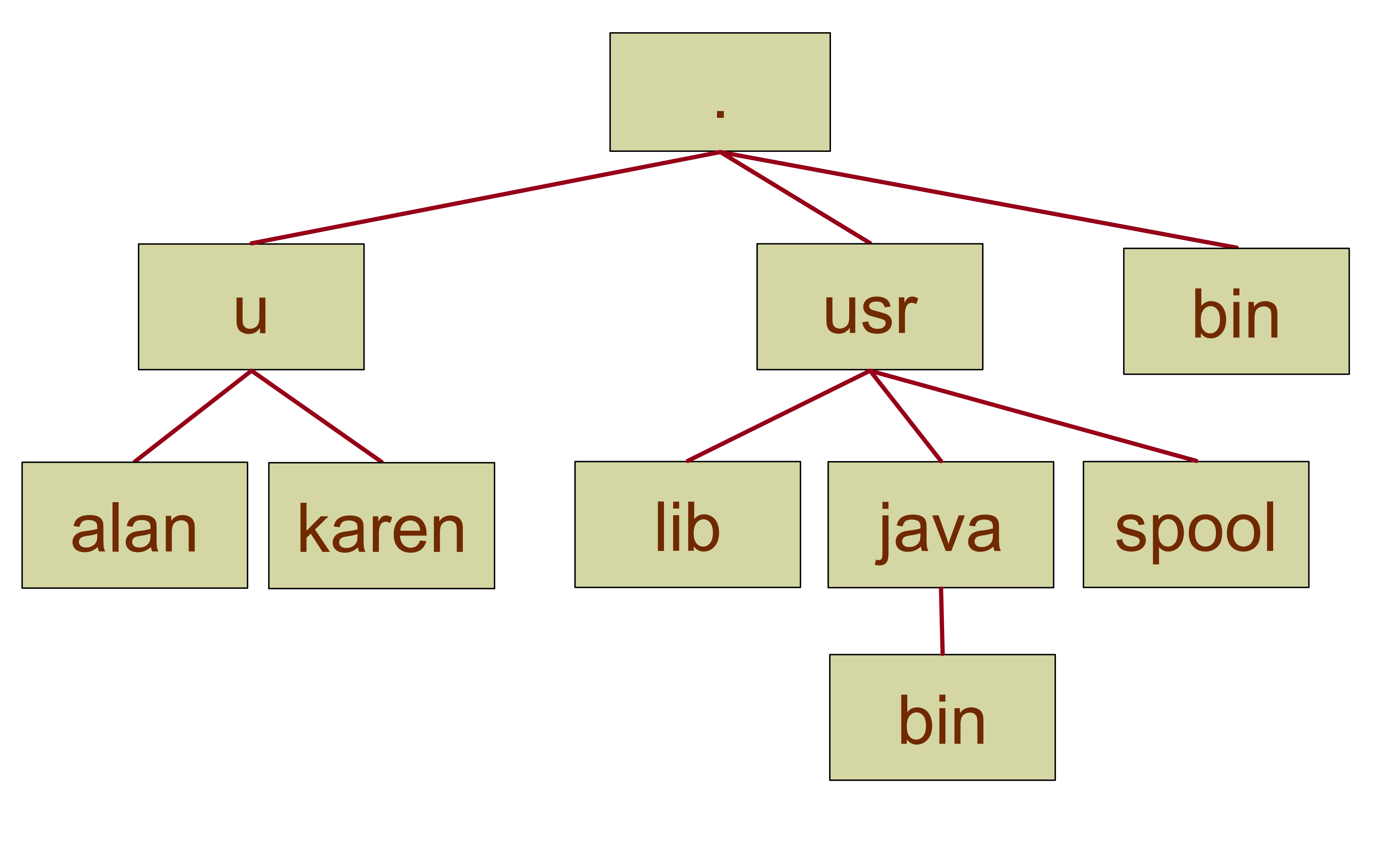
Byte File System

Byte File System (BFS)

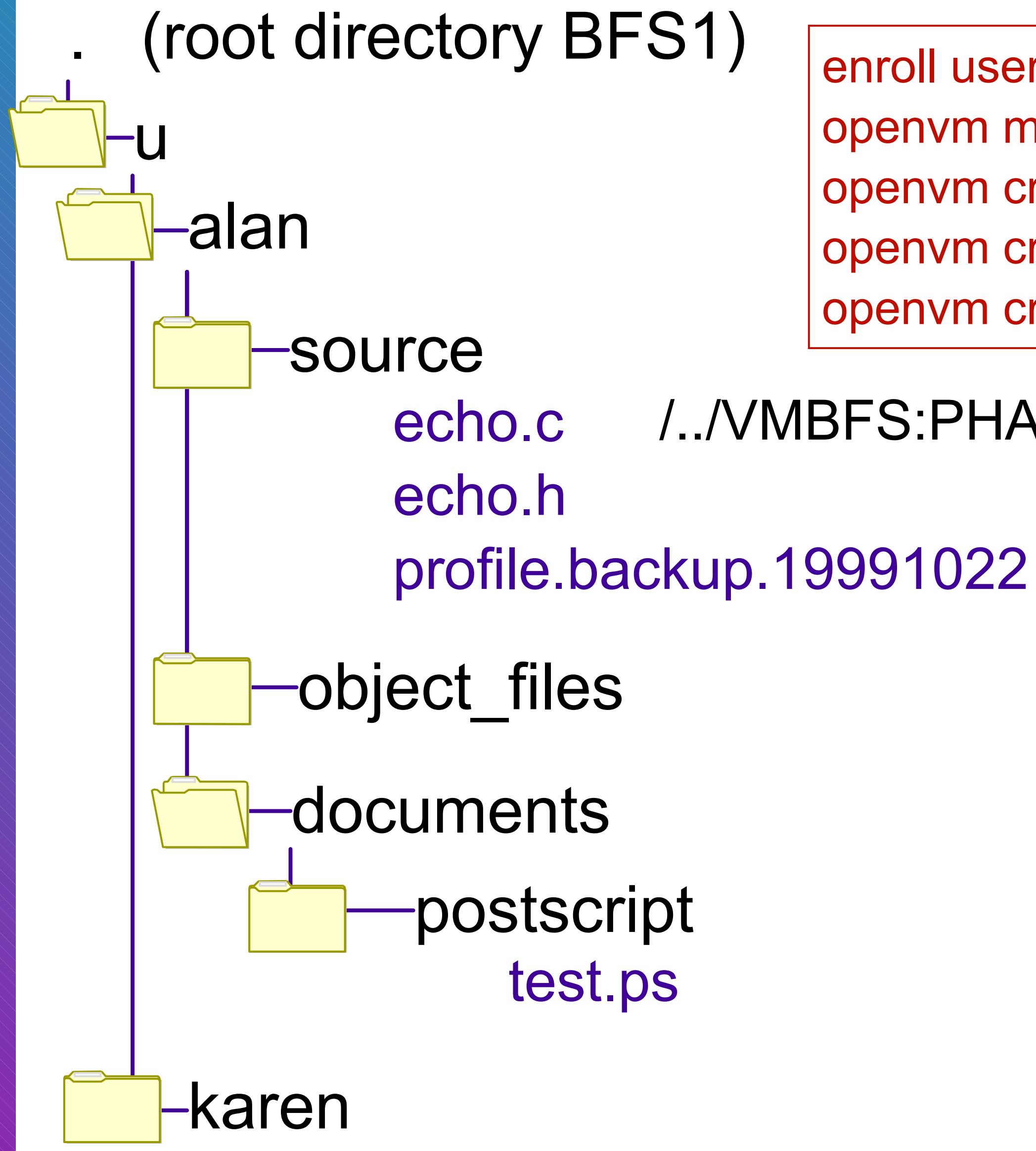
- Unix** file system
  - streams instead of records
  - multiple users can write to the same file
  - hierarchical directories
  - all users in a single file space
- Access based on POSIX UID and GID
  - defined in CP directory or ESM for each user
  - permissions: owner, group, other
  - authorizations: read, write, execute
  - VM userid translated to POSIX UID/GID

Byte File System File Space

SHARE
Boston
July 2000



Hierarchical BFS Directory



. (root directory BFS1)

u

alan

source

echo.c

echo.h

profile.backup.19991022

object_files

documents

postscript

test.ps

karen

```
enroll user bfs1 phantom (blocks 200000 bfs
openvm mount /../VMBFS:PHANTOM:BFS1/  /
openvm create directory /u
openvm create directory /u/alan
openvm create directory /u/karen
```

/../VMBFS:PHANTOM:BFS1/u/alan/source/echo.c

Byte File System (BFS)

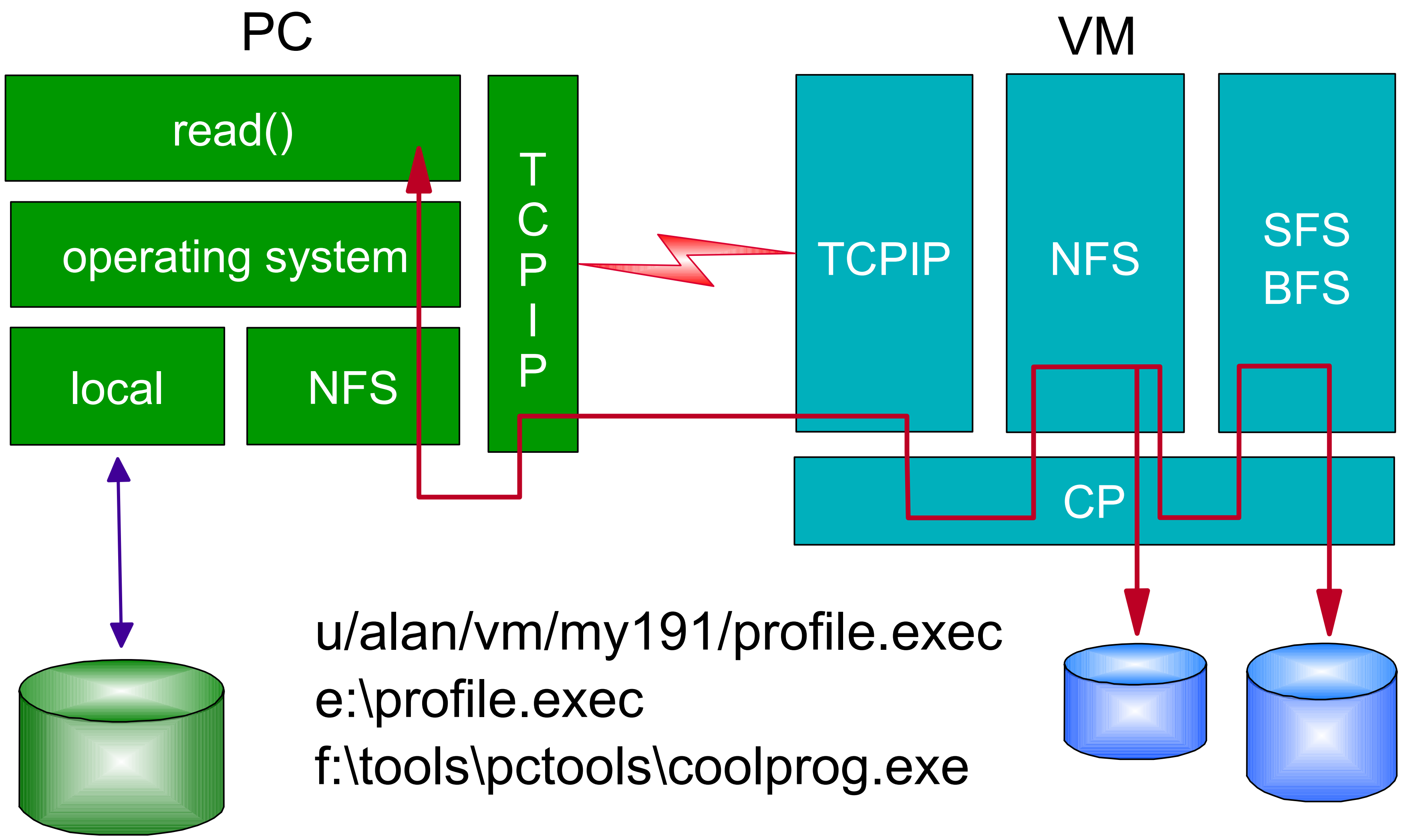
- As for SFS, the server moderates access
  - BFS is just another kind of file space in an SFS filepool
- OPENVM MOUNT
  - Cannot use ACCESS command
  - easiest to use with Shell & Utilities Feature
- Remote access
  - VM users via collection or VTAM (AVS, LU 6.2)
  - NFS

Network File System

Network File System** (NFS)

- Enables VM system to act as open systems file server
- Insert CMS directory or minidisk into the client's directory structure or assign it a new drive letter
- Client applications use local file I/O interfaces
  - device drivers use NFS remote procedure calls
  - transparent to applications

NFS - A Thousand Words



u/alan/vm/my191/profile.exec
e:\profile.exec
f:\tools\pctools\coolprog.exe

Network File System (NFS)

- BFS is an exact match for NFS capabilities
- Ordering note:
  - Separate feature
  - Additional charge

Why NFS?

- Data is available to any system which has an NFS client
- Data is shared, not duplicated
  - everyone sees the same version
  - updates available immediately

Why NFS?

- Enjoy all of the advantages of keeping your data on VM
  - established facilities to back up and recover data
  - System Managed Storage
  - host access control
  - centralized administration of space allocation

How Do I Use NFS?

- NFS clients available for common platforms
  - commonly known as "mount"
  - available for OS/390
  - standard for Unix and AIX
  - add-on for others
    - Hummingbird - NFS Maestro**
    - NetManage - InterDrive**, ViewNow** NFS
- Currently no NFS client for VM

How Do I Use NFS?

- Mount command defines
  - read-only or read-write
  - authentication - VM user ID and password
  - how to perform ASCII-EBCDIC translation

- For SFS and minidisk the Mount command defines how to
  - convert CMS file names into PC file names
  - convert CMS records into Unix streams
  - recognize end of line

Read More About NFS

- Mount command examples available from the VM TCP/IP NFS home page
  - http://www.ibm.com/s390/vm/nfs
- See "Inside NFS" article in Sterling Software's Fall 1998 *VM:News*

Programming Interfaces

Minidisk Programming Interfaces

- Assembler
  - FSSTATE, FSOPEN, FSPOINT, FSREAD, FSWRITE, FSCLOSE
- OS and DOS simulation
  - OPEN, CLOSE, GET, PUT, READ, WRITE
- Callable Services Library (CSL) routines
  - DMSOPEN, DMSREAD, DMSWRITE, DMSCLOSE
- Standard HLL file system APIs

SFS Programming Interfaces

- When directory is accessed, same as for minidisk
- CSL routines work for unaccessed directories
- Work units allow commit control

BFS Programming Interfaces

- Since you cannot ACCESS a BFS directory, traditional file system APIs won't work
- CSL Routines
  - BFS-specific BPX1....
  - SFS using directory name
    - `OPENVM LISTFILE (NAMES`
- C for VM/ESA standard file I/O
  - fopen("./conference.data", "r")
  - read(), fwrite(), scanf(), etc.

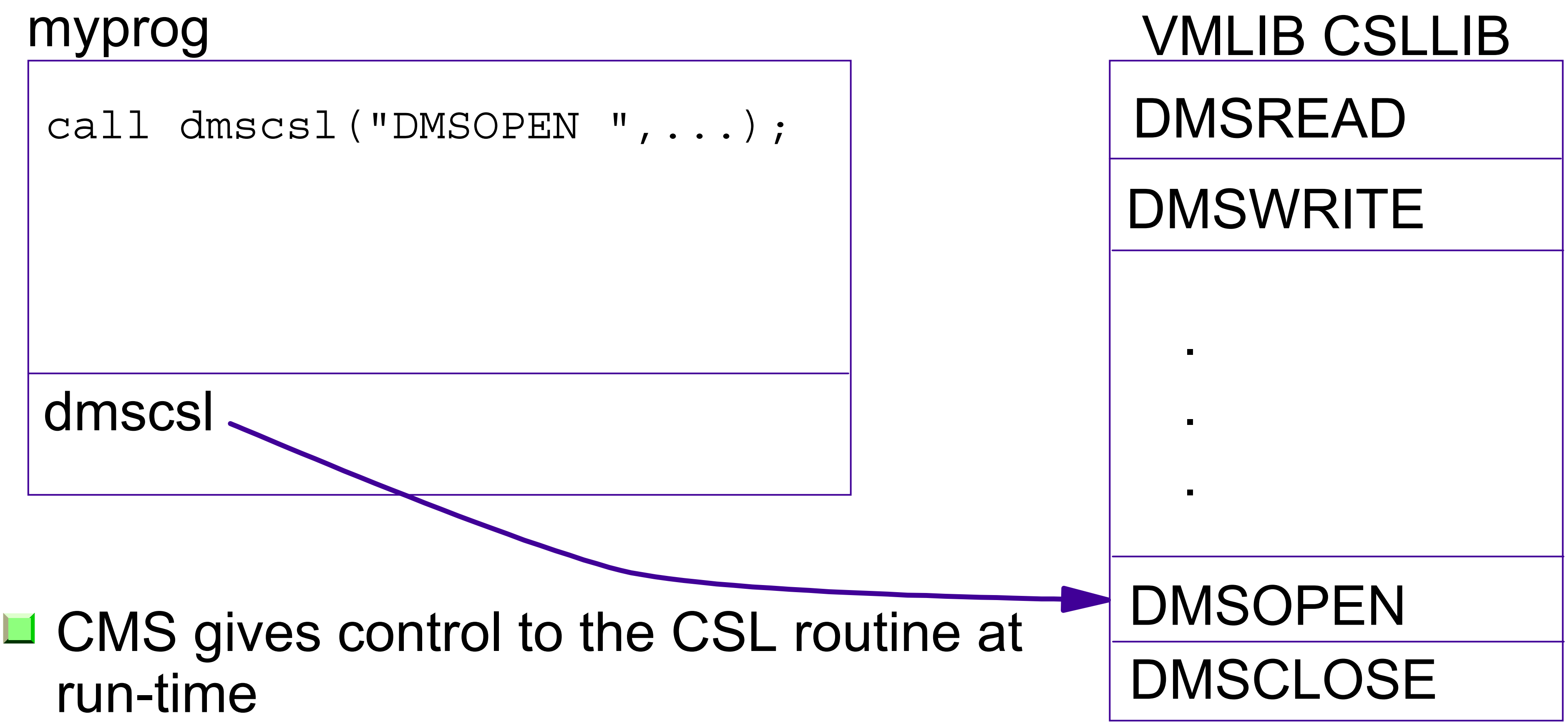
NFS Programming Interfaces

- NFS Version 2 - RFC 1094
- NFS Version 3 - RFC 1813
- Boring unless you are writing an NFS client

Callable Services Library (CSL)

- Introduced in VM/SP Release 6
- Languages supported
  - COBOL
  - FORTRAN
  - Pascal
  - PL/I
  - Rexx
  - C
  - Assembler
  - anything that can use OS CALL interface

Callable Services Library (CSL)



myprog

```
call dmscsl("DMSOPEN ",...);
```

dmscsl

VMLIB CSLLIB

DMSREAD

DMSWRITE

.
.
.

DMSOPEN

DMSCLOSE

- CMS gives control to the CSL routine at run-time
- Different versions of CSL routine can be used without changing your program
  - Similar to a DLL

Callable Services Library (CSL)

- files and directories
- virtual machine settings
- call a Rexx exec
- VM data spaces
- CMS program stack
- workunits
- data compression
- TCP/IP
- date and time
- event management
- timer services
- Monitor data
- more...

Callable Services Library (CSL)

- Create your own CSL libraries
- You can substitute your version for IBM version
- Excellent for creating well-defined interfaces
  - Between different products
  - Between components of same product
  - Exits
    - "If not loaded, don't call it"

File System References

- Usage
  - CMS User's Guide
  - OpenEdition User's Guide
  - OpenEdition Command Reference
- Programming
  - CMS Application Development References
  - CMS Application Multitasking
  - CMS Application Development Guide
  - OpenEdition Callable Services Reference
  - IBM C for VM/ESA Library Reference

SHARE
Boston
July 2000

Contact Information

By e-mail:	Alan_Altmark@us.ibm.com
In person:	USA 607.752.6027
The Web:	http://www.ibm.com/vm/devpages/altmarka
Mailing lists:	VMESA-L@listserv.uark.edu
On TalkLink:	VMESA CFORUM