NAME

file_system – Shore registered objects

DESCRIPTION

A Shore file system looks much like a Unix file system. A file system corresponds to a *volume,* which is a *device,* such as a raw disk or a Unix file. On each volume is a hierarchically named set of objects. At the top of the hierarchy is an unnamed object, called the *root* of the volume. The root is given a name when the volume is *mounted,* the means by which the data on the volume are made available for use.

A Unix file system contains both named and unnamed or **anonymous** objects. Named objects are called *registered* object. All Shore objects have *type,* and file system objects are no exception. Registered objects can have these types:

Directory
: A directory in Shore is the same as a Unix directory. It contains hard links to the objects it "contains."

Symbolic link
: A symbolic link in Shore is like a Unix symbolic link. It is a path name. It may dangle, and it may cross file system.

Cross reference
: Cross references have no Unix analog. A cross reference is an object that, like a symbolic link, points to another object, but the value stored in a cross reference is not a path, but an object identifier. The object referenced may be anonymous or registered. A cross reference can dangle if the target is deleted after the cross reference is made. A cross reference may address an object on another file system.

Pool
: A pool has no analog in Unix. A pool is an object that identifies a place in which anonymous objects "reside". A pool can be scanned to identify or read all the objects in it.

User-defined
: Objects whose type is defined by an application or "user", can be registered or anonymous.

Unix file
: Registered objects whose types have an attribute of type TEXT are accessible through the NFS server portion of the SVASVAS, regardless of what other attributes their types have. Objects that are created by the NFS server are given the type "Unix file" which has only a single attribute, of type TEXT. All "Unix file" are registered.

Shore also has hard links, as in Unix. There is no type for a hard link, because a hard link is not an object's type. It is a reference to an object. There can be many links to a Shore (registered) object. Each registered object contains a reference count for the number of links to the object. An object is deleted only when its reference count reaches zero. A hard link always points to an object; it never dangles. Hard links cannot cross file system boundaries.

All registered objects reside in the same Storage Manager **file.** Anonymous objects reside a Storage Manager files associated with their pools.

Path names, symbolic links, and cross references are followed and expanded much as in Unix, and there are limits to the length of a path, both before and after expansion of symbolic links. The SVAS takes its limits from the operating system on which it is running. The following limits are used, and their values are taken from sysconf(), pathconf(), or compile-time constants as described below:

OpenMax
: The maximum number of open files is taken from *sysconf(_SC_OPEN_MAX),* if it is available, otherwise from *getdtablesize(),* if it is available, otherwise from constants in the operating system include files. OpenMax is used internally for several purposes: it limits the number of shared-memory pages that a client can share with a SVAS (to facilitate communication), and it limits the number of clients a server can serve at any one time.

LinksMax

The maximum number of links to an object is taken from pathconf, if it is available, otherwise from constants in the operating system include files.

SymlinksMax

This is the maximum number of symbolic links that will be expanded in a single attempt to follow a path. It is taken from constants in the operating system include files, if available, otherwise 20 is used. The purpose of this is to avoid looping forever if a symbolic link references itself, directly or indirectly.

PathMax

This is the maximum length of a path, in characters, after expanding symbolic links. Its value is taken from pathconf, if it is available, otherwise from constants in the operating system include files, if available, otherwise its value is 1024.

NameMax

The maximum length of a register object's file name is taken from pathconf, if available, otherwise its value is 255.

All values taken from pathconf() are relative to the Unix root directory ("/").

ENVIRONMENT

Directories, cross references, symbolic links, and pools are never copied into the application's address space. All operations on these objects are performed on the server. Objects of user-defined types and Unix file objects can be copied into the application's address space by a *readObj* operation.

VERSION

This manual page applies to Version 1.1 of the Shore software.

SPONSORSHIP

The Shore project is sponsored by the Advanced Research Project Agency, ARPA order number 018 (formerly 8230), monitored by the U.S. Army Research Laboratory under contract DAAB07-91-C-Q518.

COPYRIGHT

SEE ALSO

mount(svas), pool(svas), readObj(svas), and **text(svas).**