No detailed assumptions are made as to the technology or structure of the distributed environment. The communication subnet could be packet-switched, circuit-switched or a multiaccess channel. The terminology used in this chapter will have counterparts in all communication subnets, though our analysis of file migration is biased towards store-and-forward networks. A host is a computer system that is a potential user and/or supplier of resources in the distributed operating system. A user is a person or a program that interacts with the host. A user process is the process associated with a user. A switching node is a device, in many cases a small computer, that accepts data and control from the host and sends it over the communication links, with the possible cooperation of other switching nodes, to the destination. The collection of switching nodes and communication links is the communication network or communication subnet. We assume that all communication between hosts is viewed as interprocess communication, and that it can be performed reliably [Cerf74, Cerf74a, Sunshine75]. Figure 5.1 illustrates such a distributed network environment.

One of the primary goals of the DFS presented in this thesis is to perform the necessary functions using distributed algorithms, under the assumption that no centralized information source or point of control exists. Such an assumption is necessary in order to preserve the reliability of the DFS. There are usually many points of control in a distributed algorithm, and so if any should go down then it is usually possible to continue to function (possibly in a degraded fashion). We will investigate the extent to which conventional algorithms used in implementing file systems can be distributed.

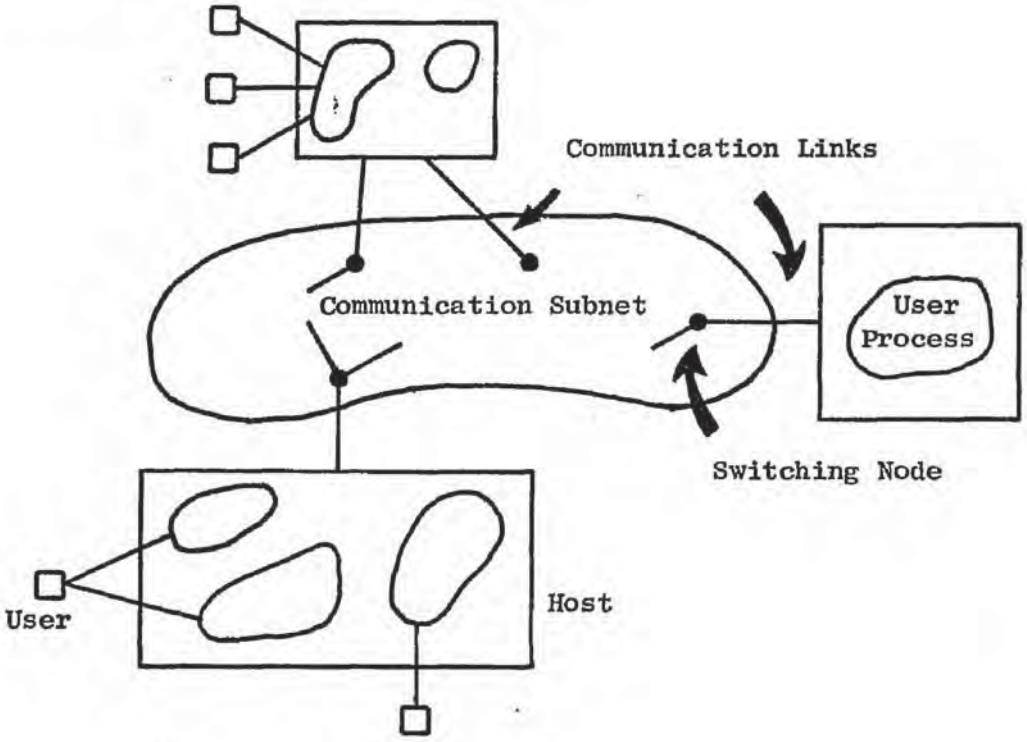
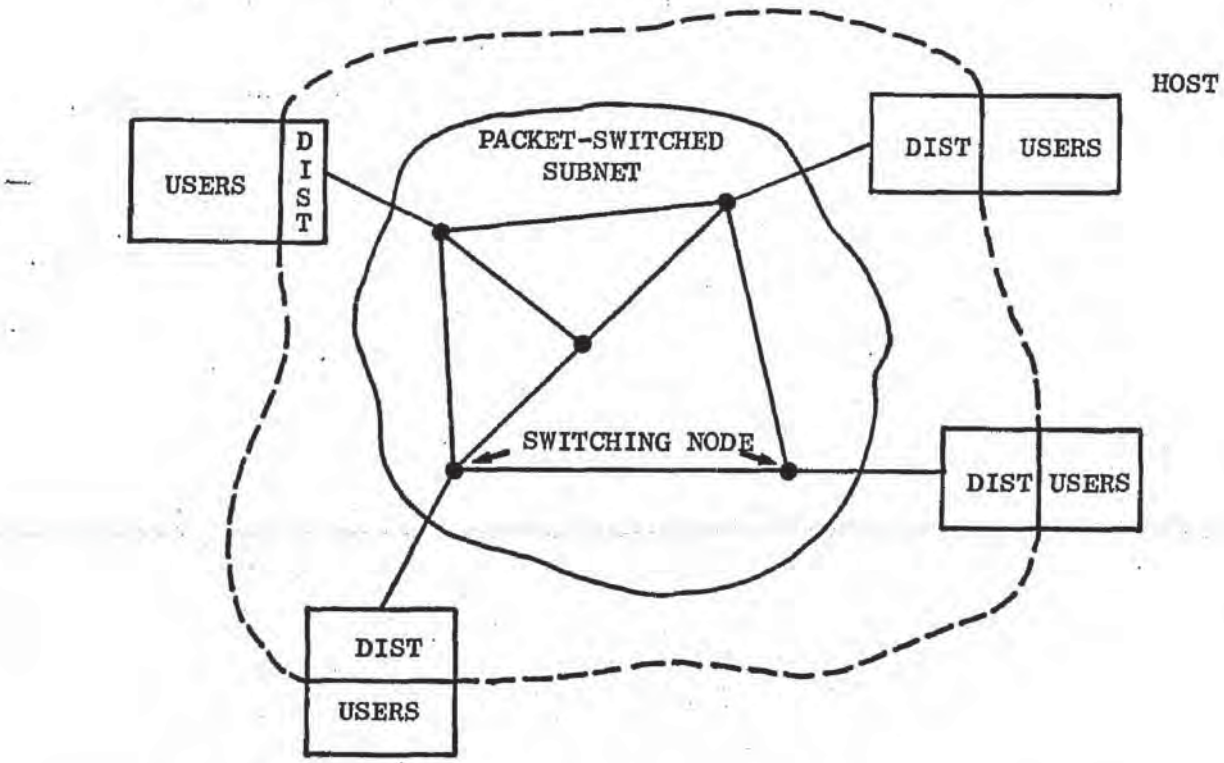


Figure 5.1. A DISTRIBUTED NETWORK ENVIRONMENT.



DIST ≡ LOCAL FILE SYSTEM RESOURCES THAT ARE PART OF THE DFS

5.2 User Interface and Catalog Structure

The DFS will consist of local file systems at each of the hosts. Their resources are available to all users. Figure 5.2 shows this model. In general, the host may divide its local file system into two parts; a private part and a sharable part. The host may use the private part to provide storage for permanently resident files for its local users. It may even copy files from the DFS into the private file system, in order to have a permanently resident copy outside the domain of the DFS. Such copies may become inconsistent with their counter-parts in the DFS. The file system provided by the NSW consists of a sharable global NSW file space, as well as a private, non-sharable local file space at every host [Schantz76]. We do not make use of any such techniques in this thesis, and therefore assume independence between these two parts, and so leave the private part out of all subsequent discussions.

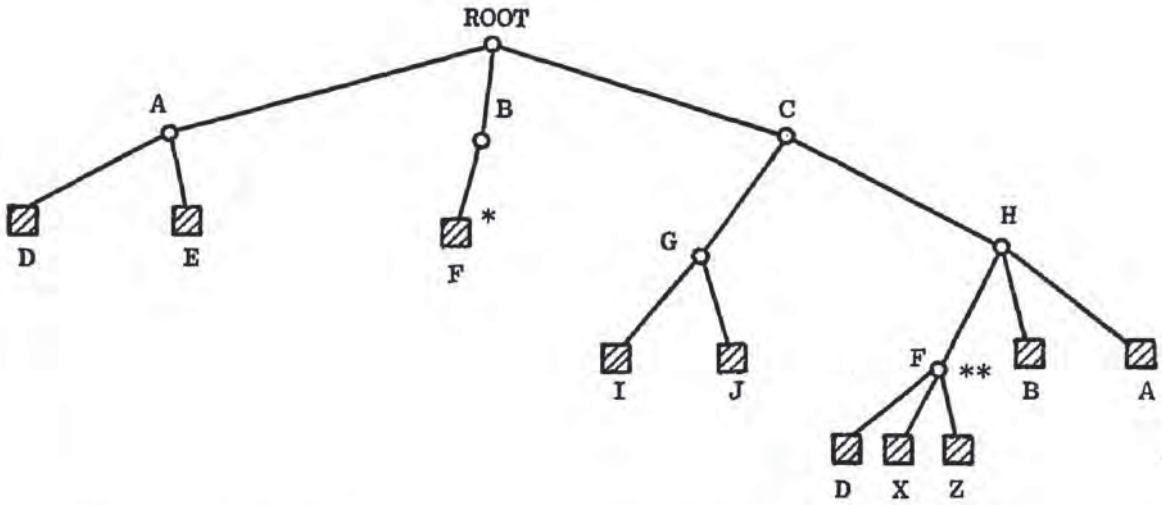
We will assume that the entire file system will have a hierarchical structure, since it seems most appropriate from the users' point of view [Daley65]. Files will be referenced symbolically and shared by a number of users. File names must be unique and must not change even when files migrate to other hosts, so that this movement is transparent to the users of the files. The file structure may be thought of as a tree of files, some of which are directories. Except for the root directory, each file finds itself pointed to by exactly one branch in exactly one directory. The tree name of a file will be its name relative to the root directory. Figure 5.3a illustrates such a structure. Links may

now be superimposed on this structure, such that files may be accessed from directories other than those present in their respective tree names. Note that tree names do not contain any links. The path name of a file is its name relative to the root directory and may contain links. Hence, in general, a file can have only one tree name but many path names. The tree name and path name of a file can be specified relative to a working directory other than the root. Since the tree name of the working directory is known the absolute tree name of the file can always be determined. Figure 5.3b illustrates a directory structure with links. The file systems of Multics [Organick72], Unix [Ritchie74] and Tenex [Bobrow72] have similar structures. In a conventional monolithic file system, the directory files in the catalog provide a reference point for naming files, access control to the files that are their offspring, and indicating where on secondary storage they reside.

We now consider different ways of physically structuring the catalog, and the assumptions and constraints required, in order to create a distributed system with the same logical structure as described above. Once files have been created and entered into the DFS catalog they may migrate and physically reside at any host. When a file moves, neither its tree name or path name changes, only its location.

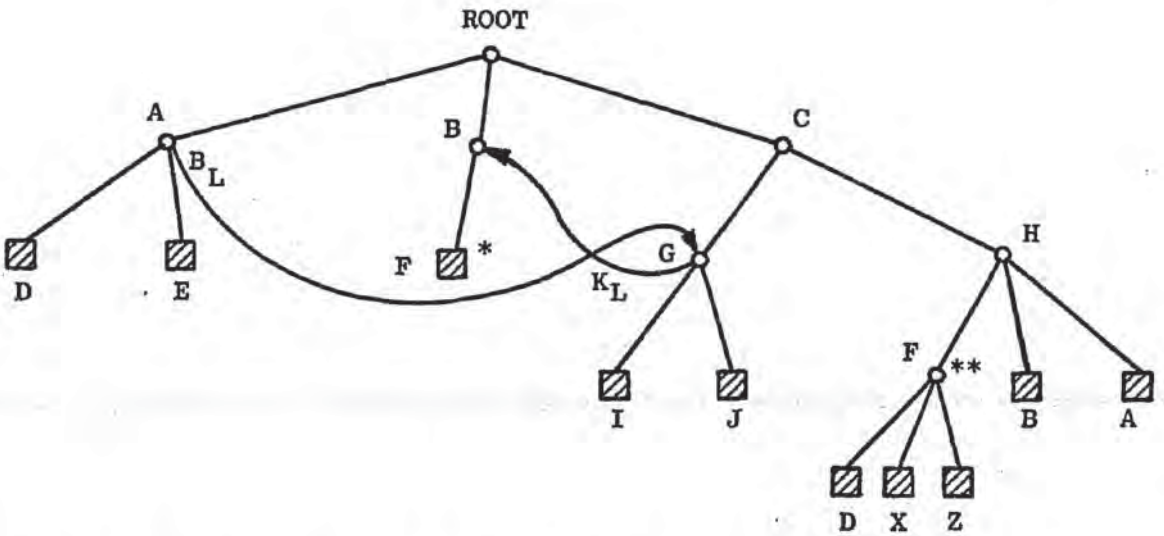
5.2.1 Centralized or Duplicated Catalogs

The logical catalog structure could be centralized at one host, with the non-directory files scattered among various hosts. Without getting into the implementation details, note that each host (except the



Tree name of file marked * relative to root is B.F and of file marked ** is C.H.F.

Figure 5.3a. A HIERARCHICAL CATALOG STRUCTURE WITHOUT LINKS.



Tree name of file marked * relative to root is B.F; path name of * relative to root C is G.K.F and, relative to the root, could be A.B.K.F.

○ Directory file; ▨ Nondirectory file; Links are subscripted with L

DOCKET ALARM

Explore Litigation Insights

Docket Alarm provides insights to develop a more informed litigation strategy and the peace of mind of knowing you're on top of things.

Real-Time Litigation Alerts

Keep your litigation team up-to-date with **real-time alerts** and advanced team management tools built for the enterprise, all while greatly reducing PACER spend.

Our comprehensive service means we can handle Federal, State, and Administrative courts across the country.

Advanced Docket Research

With over 230 million records, Docket Alarm's cloud-native docket research platform finds what other services can't. Coverage includes Federal, State, plus PTAB, TTAB, ITC and NLRB decisions, all in one place.

Identify arguments that have been successful in the past with full text, pinpoint searching. Link to case law cited within any court document via Fastcase.

Analytics At Your Fingertips

Learn what happened the last time a particular judge, opposing counsel or company faced cases similar to yours.

Advanced out-of-the-box PTAB and TTAB analytics are always at your fingertips.

API

Docket Alarm offers a powerful API (application programming interface) to developers that want to integrate case filings into their apps.

LAW FIRMS
Build custom dashboards for your attorneys and clients with live data direct from the court.

Automate many repetitive legal tasks like conflict checks, document management, and marketing.

FINANCIAL INSTITUTIONS
Litigation and bankruptcy checks for companies and debtors.

E-DISCOVERY AND LEGAL VENDORS
Sync your system to PACER to automate legal marketing.