第 13 部

広域分散ファイルシステム

第 1 章

DFS ワーキンググループ

インタネットの情報基盤としての社会的な位置づけはますます重要なものとなりつつある．最近のブームである WWW は，その一部分であり，WWW を含めた情報共有サービスの役割りは大きい.

WIDE プロジェクトの分散ファイルシステム・ワーキンググループ (DFS-WG) では、インタネット上での分散ファイルシステムについて研究、試作および検討を行なってきた。DFS-WG は、この分散ファイルシステムを従来のオペレーティングシステムのファイル機能の拡張としてとらえるだけでなく、情報共有サービスに代表される分散情報環境のプラットホームとして位置付け、関連する技術の研究を行なってきた。実際に分散ファイルシステムは、情報共有環境の実現手段として有効であり、情報共有環境に必要とされる技術も、分散ファイルのモデルに基づいて検討することにより、問題が明確になってくる。

94 年度は，従来からの検討課題である，広域分散ファイルのモデル，サーバ間プロトコル，セキュリティ，各種方式・アルゴリズムの検討を行なってきた．また，テレターミナルやセルラ，PHS といった無線網の整備がおこなわれるに従って，現実性が高まってきた携帯情報端末におけるファイル共有機構の検討も重要な検討課題であった．そこで，本報告書では，移動型計算機に適した、分散ファイルシステム Personal File System (PFS) について重点的に報告する.

第 2 章

PFS

2.1 はじめに

技術の進歩により、小型軽量な計算機でも UNIX などの OS が動作する程の性能を持つものが開発され、利用されるようになった。これらの計算機は容易に持ち運ぶ事ができ、どこでも利用できる事から、「移動型計算機」と呼ばれている。しかし移動型計算機は、その大きさや重さの制限から、ディスクの容量などの計算機資源が従来の計算機より劣る事が多い。そのため、他の計算機の資源を利用する分散ファイルシステムは、移動型計算機にとっても重要な技術である。

現在、Sun NFS [91] (Network File System) や AFS [92] (Andrew File System) など、多くの優れた分散ファイルシステムが開発され、広く利用されている。これらのファイルシステムは、基本的に確認応答プロトコルに基づいて設計されている。すなわち、クライアントがファイル操作要求をサーバに送り、サーバがそれを処理して回答を行うという動作をする。この機構は通常の (固定型の) 計算機においては効率良く動作するが、移動型計算機では問題が生じる。この問題については、2.2 節で述べる。

カーネギメロン大学で開発された CODA[93] は移動型計算機での利用を想定し、通信路が利用不能な状態での動作が可能になっている。CODA では、クライアントにキャッシュを用意し、操作対象のファイルがキャッシュ内にある限りはファイルサーバとの通信路が不通でも操作が可能となっている。また、通信路が利用できる場合も、キャッシュの内容が他のクライアントから書き換えられた際にサーバから通知される機構により、キャッシュを長時間有効利用できる。

しかしながら、CODA には二つの大きな問題がある。一つは UFS や NFS などの他のファイルシステムとの共存が難しい事であり、もう一つはクライアント、サーバ共に OS のカーネルの変更が必要な事である。

そこで、これらの問題を解決する新たな分散ファイルシステムとして、PFS (Personal File System) を設計、実装した。このファイルシステムは、専ら一人のユーザによって利用される移動型計算機で利用するように設計されている。また、他のファイルシステムとの共存も容易で、かつ基本的に OS のカーネルの変更も必要無い。また、PFS では移動型計算機に特化したアプリケーションを、ファイルシステムの側から支援するための機構と

して動作する事も考慮している。

2.2　移動型計算機の問題点

移動型計算機は、ネットワークとの接続に関して次のような問題がある。

- バンド幅が狭い通信回線を利用する事がある。
- 応答時間が長い通信回線を利用する事がある。
- 通信に利用するデバイスによってネットワークインターフェースを切替えなければならない。
- ネットワークへの接続地点が変化する。
- ネットワークへの接続が途絶えた状態でも動作を続ける。

このような移動型計算機の環境下で NFS を利用すると、次のような問題が生じる。

- 通信回線の遅延が大きいと、応答速度が極端に悪化する。
  これは、単純なファイル操作でも幾つかの NFS 要求を発行する必要があるのと、クライアントがサーバへの要求毎に、それに対する応答を待ってから次の要求を出すためである。
- サーバの停止や、通信の途絶があると利用不能になる。
  クライアント側のキャッシュを保証する機構が無く、頻繁にファイルの変化を確認する必要があるためである。
- 移動の度に再設定が必要である。
  ホストの移動について考慮されておらず、クライアントの IP アドレスから導かれるホスト名でのみアクセス制御を行っているため、クライアントが移動して IP アドレスが変化すると、変化先のホスト名でのアクセスを許可しなければならない。

AFS は広域ネットワークを通して利用する事が配慮されており、サーバの複製を作る機構や、他のクライアントがキャッシュしているファイルを書き換えた際に、サーバがその情報をファイルをキャッシュしているクライアントに通知する機構などがある。このため、遅延の問題はある程度解消されるが、通信が途絶すると NFS 同様に利用不能になる。

2.3　PFS の設計

2.3.1　基本機能

PFS は移動型計算機で利用するために、次の機能を持つ。

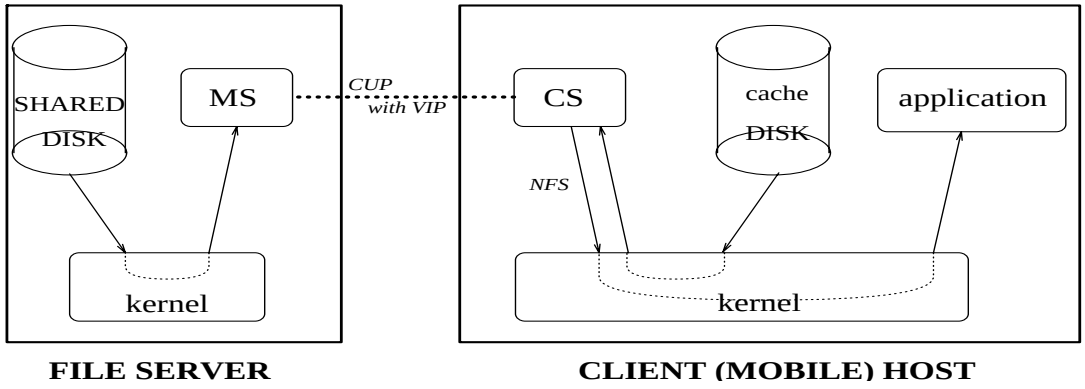

図 2.1: PFS の概要

- サーバとの通信が途絶えても利用可能
- 低速な通信路を利用していても高速な応答性
- 他のファイルシステムとのファイルの共有
- 計算機の移動を意識せずにファイルアクセスが可能
- 移動型計算機側の資源を大量に消費しない

これらの機能を実現する為に、いくつかの制限事項を認める。

- 通信断中などに、キャッシュのサーバとの一貫性を完全には保証しない
- 書き込みはユーザのホームディレクトリに限定する

クライアントマシンのユーザは、通常自分のホームディレクトリのみを書き換えるので、これらの制限は致命的なものではない。この問題については後述する。

2.3.2 基本構造

PFS はファイルサーバとクライアントから構成される。サーバホストでは、MS (Master Server) が通常のファイルシステム上のファイルを、クライアントに供給する。クライアントホストでは、CS (Cache Server) が MS と通信しつつ、クライアントホスト上のアプリケーションにファイルを供給する (図 2.1)。

CS はキャッシュディスクを管理し、それを利用してアプリケーションへのファイル供給を行う。CS はアプリケーションからの要求とは非同期に MS との通信を行い、キャッシュの更新を行う。この機構により、アプリケーションはファイルサーバとの通信が不通でもファイル操作を継続する事が可能になる。

さらに、PFS は次のような機構を持つ。

- 必要な時にファイルにアクセスするために、ユーザは作業に必要なファイルのファイル名を WST (Working Set Table) に登録する事ができる。WST に登録されたファイルはあらかじめキャッシュに読み込まれる。

- ユーザやアプリケーションが PFS の動作をコントロールするために、アプリケーションから CS を操作する機構を用意する。これにより、ファイルの更新が明らかに不必要な時には自動更新を停止したり、強制的にキャッシュの内容をサーバに反映させたりする事が可能になる。

- アクセス速度を上げるために、各 MS は他のファイルサーバ上のファイルをキャッシュし、CS に中継する事ができる。これにより、各 CS はもっとも近い MS に接続すれば良くなる。この機構をマルチレベルキャッシングと呼ぶ。

- 広域ネットワーク中の不必要なファイル転送を避けるために、複数の MS が、同じ内容のファイルを保持している時、これを等価ボリュームと呼び、各 CS は等価ボリュームを持つ MS 中でもっとも近い MS を選択して接続することができる。

- クライアントのディスク使用量を減らすため、CS はキャッシュ中で長時間アクセスされないファイルを圧縮して保持する。更に長時間アクセスされず、かつ WST にないファイルは消去される。

- 移動透過性を実現するために、CS と MS の通信には TCP/VIP[94] (Virtual Internet Protocol) を用いる。通信用の下位層として VIP を利用する事により、移動透過性の実現や、再接続時の TCP 再送タイマの管理などを扱う必要が無くなり、構造を簡略化し、機種依存性を低くすることが可能になる。

実際に広域ネットワークで運用された時の接続例を図 2.2 に示す。この図では fileserver A は “HOME” と “USR” の 2 つのボリュームを、fileserver B は透過ボリュームの “USR” ボリュームとキャッシュディスクをそれぞれ持っている。LAN a/b 及び network provider はそれぞれ広域ネットワークを介して接続している。network provider を通してネットワークに接続する position c との距離は、position a の方が position b よりも近い。

クライアントが position a の位置に接続している場合は、“HOME” と “USR” の両ボリュームは fileserver A から供給される。

その後、position b に移動すると、クライアントは最も近い server B からファイルの供給を受けるようになる。この場合、“HOME” は server A のみが保持しているので、server A から server B のキャッシュに取り込まれ、クライアントは server B を通じてファイルの供給を受ける。“USR” に関しては等価ボリュームなので server B から直接供給を受ける。

position a から position c へ移動した場合は、最も近い fileserver A からのファイルの供給を受け続ける。

ネットワークから切り離された position d の場合は、CS のキャッシュのみで動作を継続する。

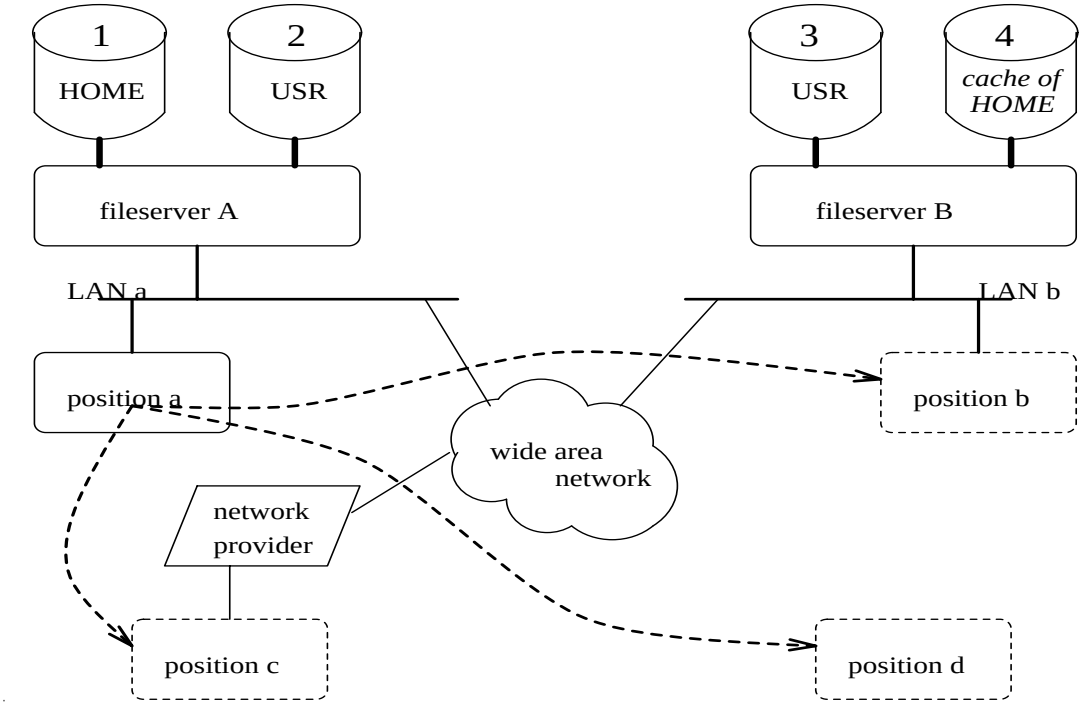
1
HOME
2
USR
3
USR
4
cache of
HOME
fileserver A
fileserver B
LAN a
LAN b
position a
position b
wide area
network
network
provider
position c
position d

図 2.2: PFS の接続例

このように、PFS ではできるだけ近く、できるだけ大容量のキャッシュを利用して動作できるように設計されている。

2.3.3　ファイルの一貫性

クライアント上でアプリケーションがファイルに書き込むと、PFS はそれをまずキャッシュに書き込み、その書き込み内容を MS に反映する前にアプリケーションに対して書き込みの終了を通知する。この機構により、PFS は高速な書き込みと通信断中の操作を実現しているが、ファイルの一貫性が損なわれる可能性がある。

例えば、クライアントが通信断中に、異なるクライアントを利用している複数のユーザが同じファイルに変更を加えると、ファイルの一貫性が損なわれ、ユーザの知らない内に同じ名前の内容の異なるファイルが生成される事になる。このような場合、各ユーザは自分の行っている変更をそのまま続け、後でそれらのファイルを統合するのが望ましい事が多い。

このような場合に対応するために、PFS では二つの機構を用意する。

- 各ユーザは自分の変更中のファイルにアクセスしつづける事ができる。
- 同じファイル名で異なるバージョンのファイルが生成された場合は、PFS はユーザに通知し、後にファイルの統合を行えるようにする。

特に前者の機構は VCFS[95] (View Control File System) によって提供される。VCFS はアクセスするユーザやホストを鍵に、同じファイル名でアクセスされるファイルを切替える事のできるファイルシステムである。PFS では MS で供給元のファイルシステムに VCFS を通してアクセスする事により、異なる内容のファイルを同一のファイル名で管理する事を可能にしている。

2.4　実装

現在、PFS は BSD 系の UNIX に実装されている。現在までに動作を確認した OS は、SONY NEWS-OS 4.2, Sun SunOS 4.1.3, BSDI BSD/386 1.1 である。プログラムは殆ど C 言語で記述されている。Sun RPC[96] (Remote Procedure Call) インターフェース及び、XDR[97] (eXternal Data Representation) ネットワーク透過データ表現の符号化及び復号化プログラムには、それぞれ RPC 言語と XDR 言語を用いて記述し、OS 付属の `rpcgen` コマンドを用いて C 言語に変換した。

現在の実装は以下の制限がある。

- MS と CS がそれぞれ一つづつだけ動作する
- 単一のボリュームのみ

- キャッシュの自動消去無し
- キャッシュの自動圧縮無し
- VCFS を用いた一貫性制御無し
- CS コントロールの機構無し

2.5 評価

評価には、9600bps の SLIP (Serial Line IP) で接続した二台のワークステーションを用いて、代表的な分散ファイルシステムである NFS との比較を行った。

2.5.1 応答性

20 個のエントリのあるディレクトリを定期的に読み出すのにかかる時間を NFS と比較した。測定には UNIX の `/bin/time` コマンドを用いて、`/bin/ls -l` コマンドでディレクトリを読み出すのにかかる実時間を測定し、NFS と比較した。使用したワークステーションはサーバが SONY NEWS NWS-1750、クライアントが同 NWS-3250 である。

測定はキャッシュが空の状態で開始し、1 秒間隔と 60 秒間隔で読みだしを行った。1 秒間隔の結果を 2.3 に、60 秒間隔の結果を 2.4 に示す。

PFS、NFS 共に初回はファイルサーバからの読み込みに時間がかかる。読みだし間隔が 1 秒の時は、PFS、NFS ともにキャッシュが有効に働き 2 回目以後は比較的高速な読み出しになっている。この場合でもキャッシュの有効性を確認するためにサーバの応答を待つ NFS に対して、読みだし要求とは独立してキャッシュの有効性を確認している PFS では、より高速な読み出しになっている。

周期が 60 秒になると、NFS ではキャッシュの効果がほとんど無くなり、毎回低速な読み出しになっている。それに対し、PFS ではこの場合もキャッシュから読み出すことができるため、高速なままである。

このように、低速な回線でも良好な応答性が確認できた。

2.5.2 スループット

10Mbps の Ethernet による接続と、9600bps の SLIP 接続を用いて、ファイルの読み書きのスループットを測定するベンチマークプログラムの一つである `iozone` を用いて、PFS と NFS との間でスループットを測定した。使用したワークステーションはサーバ、クライアントともに Twinhead SUBNOTE 4SC/33 で、OS は BSD/386 1.1 である。測定に際し、9600bps での NFS については無用な再転送を避けるため、動作パラメタを rsize=1024,wsize=1024,timeo=10 に変更した。結果を表 2.1 に示す。

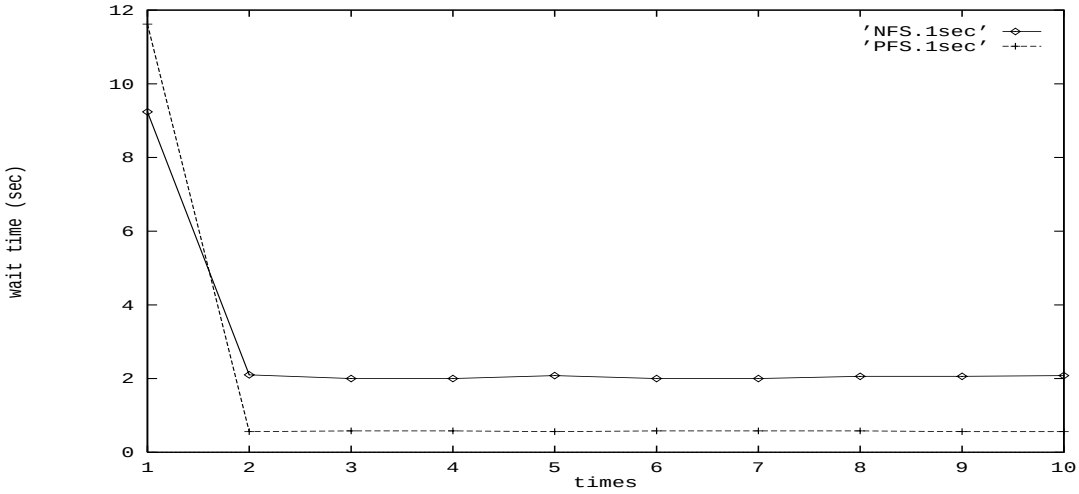

図 2.3: NFS と PFS の応答速度差 (1 秒間隔)

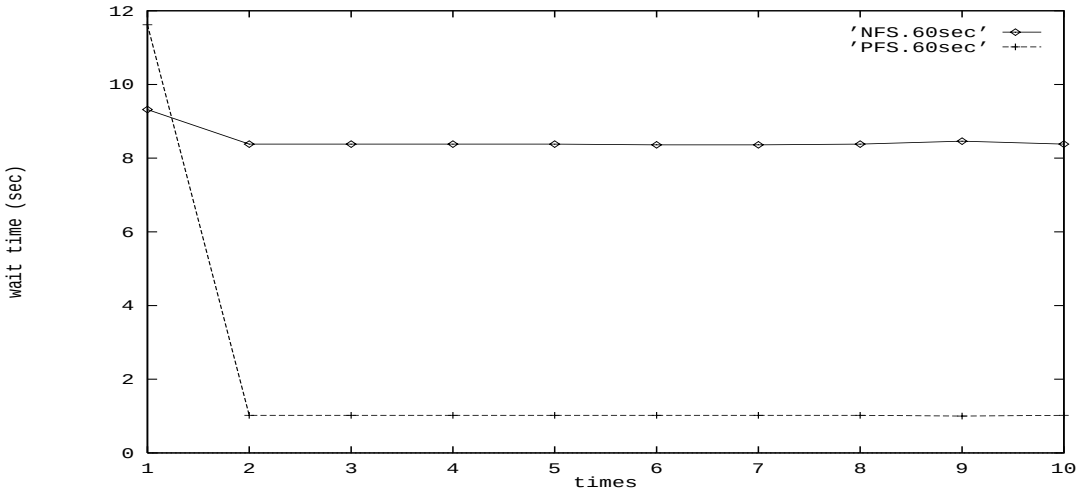

図 2.4: NFS と PFS の応答速度差 (60 秒間隔)

ファイルシステム	書き込み速度	読み込み速度
NFS (10Mbps)	75812	270032
NFS (9600bps)	599	617
PFS (10Mbps)	194994	182904
PFS (9600bps)	176301	182884

表 2.1: NFS と PFS のスループット (単位 bytes/sec)

この結果から、回線速度に殆んど影響されずに高いスループットを実現できた事が読みとれる。10Mbps など高速な回線を用いた場合は、ユーザプロセスとして実装している事もあり、カーネル内部に実装されている NFS よりも読み込み速度は 30% 程度低下している。書き込みに関しては PFS の方が高速になっているが、これは現在の実装ではキャッシュディスクへの書き込み時に、OS によるライトバックキャッシュを有効にしているためで、NFS プロトコルに厳密に従い、ライトバックキャッシュを無効にすると性能が低下すると推測される。

2.5.3 回線断中の操作

通信回線が切断されていても、キャッシュの中に存在するファイルに対しては、読み書きともに可能である事を確認した。また、通信回線が利用できるようになると、自動的にキャッシュ内の変更されたファイルをファイルサーバに反映する事を確認した。

2.6 まとめ

アプリケーションからのアクセスとは非同期にキャッシュの整合性を保持する機構により、低速な回線下でも高応答性/高スループットが実現できた。

しかしながら、今後 PFS を実用にするためには、今回の実装で省いた機能についても実装する必要がある。

特に、アプリケーションから CS をコントロールするプロトコルと、さらにそれを用いたアプリケーションを設計/実装し、実際に利用した際にどのような問題が生じるかを調査する必要がある。