2003－OS－94　(10)
2003／8／4

ファイルシステム情報を利用する分散ストレージシステム

宮澤 元

南山大学 数理情報学部 情報通信学科
〒 489-0863 愛知県 瀬戸市 せいれい町 27
miyazawa@it.nanzan-u.ac.jp

要旨

個人が複数の計算機を状況に応じて使い分けたり、ネットワーク接続が常時可能となる環境が一般化しつつある。我々は、このような環境を想定し、二階層構成法に基づく分散ファイルシステムを開発している。本稿では、この分散ファイルシステムのストレージ層となる分散ストレージシステムについて述べる。本システムはブロック単位の入出力インターフェースを持つ仮想ディスクを提供する。仮想ディスクの各ブロックが複製され、ネットワーク上の複数の物理ディスクに分散配置されるので、分散性を吸収するとともに冗長性を確保することができる。ファイルシステム層で管理されるさまざまな情報を本システムにヒントとして与えることにより、柔軟なシステム構築が可能になる。

An Adaptive Distributed Storage System Utilizing File-System Supplied Hints

MIYAZAWA Hajime

Dept. of Info. and Telecom. Eng., Nanzan University
27 Seirei-cho, Seto, Aichi, 489-0863
miyazawa@it.nanzan-u.ac.jp

Abstract

Recent technological trends make it possible for a user to use different computers, all of which is connected to network, according to his occasional purposes. Supposing such an environment, we are developing a distributed file system in the two-tiered approach. This paper describes a distributed storage system that works as the storage layer of the distributed file system. The storage system provides a virtual disk on the network with a block I/O interface. Since it distributes virtual-disk blocks and their replications onto physical disks on the network, it absorbs the network distribution and provides redundancy. It is possible to construct a flexible system by utilizing the information managed in file-system layer as hints.

1　はじめに

計算機ネットワークに関わる変化として、インターネットに常時接続できる環境が安価に提供されるようになったことが挙げられる。従来から企業や大学がインターネットに常時接続されていたのに加え、一般家庭においても広帯域のインターネット常時接続が ADSL (Asynchronous Digital Subscriber Line) や FTTH (Fiber To　The Home) などにより浸透した。また、無線 LAN (Local Area Network) を用いたホットスポットサービスも普及の兆しを見せており、時と場所を問わずインターネット接続が利用可能な環境が整いつつある。

一方で計算機自体の普及も進み、一人のユーザが状況によって複数の計算機を使い分けるようになっている。このため、これらの計算機で利用する情報の一貫性をユーザが適切に管理する必要性が生じている。これは、複数の計算機間で情報の一貫性を保つという意味で情報共有の一環として捉えることができるが、これまで計算機ネットワークを用いた情報共有において念頭に置かれていた複数ユーザ間の情報共有とは情報共有のパターンが異なり、全く同様に取り扱うことはできない。

我々はこのような状況を念頭に、新たな情報共有基盤としての分散ファイルシステムを設計・実装している。設計にあたって念頭に置くのは以下の 3 点である。

- さまざまなネットワークに対応できること
  比較的均質な通信環境を持つ LAN 上だけでなく、無線ネットワークを含む多様な特性を持つネットワーク媒体を利用するインターネットを介した広域分散環境でも効率的なファイル共有を行うことができるべきである。
- さまざまなクライアントに対応できること
  CPU パワーやメモリ容量、ディスク容量などのリソースの制約が比較的緩やかなワークステーションや PC だけではなく、リソースの限られた PDA (Personal Digital Assistant) や携帯電話などでも利用できることが望ましい。
- さまざまなファイル共有パターンに対応できること
  あるユーザが更新したファイルを他の複数のユーザが参照したり、一人のユーザが一つのファイルを複数の計算機で更新したりするなど、ファイル共有にもさまざまなパターンがあるが、どのようなパターンに対してもファイルの一貫性を保ちつつ、効率的なファイル共有を可能にすることが望ましい。

本稿では、この分散ファイルシステムの基盤となる分散ストレージシステムについて述べる。この分散ストレージシステムは、ネットワークに接続されたディスク装置群を単一の仮想ディスクとして見せるためのソフトウェア層であり、上位層であるファイルシステムに対してブロック単位の入出力インターフェースを提供する。本システムは、仮想ディスクの各ブロックの複製をネットワーク上の複数の物理ディスクに分散配置することにより、分散性を吸収するとともに冗長性を確保する。また、ファイルシステムで管理されるさまざまな情報を本ストレージシステムにヒントとして与えるインターフェースを設けることにより、ブロック配置を最適化するなど、より柔軟なシステム構築が可能になると期待できる。

以下、2 節では、ファイルシステム情報を利用する分散ストレージシステムについて述べる。現在実装中のシステムの設計について 3 節で述べる。4 節では、システム設計上の問題点について議論する。5 節で関連研究を示し、6 節を本稿のまとめとする。

2　ファイルシステム情報を利用する分散ストレージシステム

本節では、分散ファイルシステムをストレージ層とファイルシステム層から構成する方法について論じた上で、ファイルシステム情報を利用する分散ストレージシステムについて述べる。

2.1 分散ファイルシステムの二階層構成法

ネットワークで接続された計算機同士の情報共有には、分散ファイルシステムが従来から広く用いられてきた。さまざまな研究が行われてきた他、Sun NFS[21, 18] のような商用システムも開発され、計算機ネットワークを用いた情報共有の枠組みとして実用的にも広く利用されている。

伝統的な分散ファイルシステムはネットワークを介して別の計算機にあるファイルを参照するための技術という性格が強く、ネットワーク全体でファイルを一元的に管理するためには、単一のファイルサーバを用意して他の計算機はクライアントとしてファイルサーバにアクセスするようにするなどの運用上の工夫が必要である。しかし、ネットワークの大規模化に伴いシステムが扱う情報量が増大し、単一のファイルサーバで情報を集中管理するのは困難である。また複数のファイルサーバを用いる場合、管理・運用コストが増大するという問題がある。

このような問題を解決するため、ネットワークに接続されている全ての計算機にファイルを分散配置するサーバレスファイルシステムの研究が進んだ [5, 4, 11, 2, 3]。しかし、これらのシステムはファイル管理、ディスク管理、分散性の吸収などを全てファイルシステムの階層で解決するので非常に複雑なシステムとなってしまっている。

これに対し、分散性を吸収してネットワーク全体で単一の仮想ディスクを提供するストレージ層と、この上でさまざまなファイルサービスを提供するファイルシステム層とから分散ファイルシステムを構成するアプローチからの研究が行われている [7, 13, 22, 20]。本稿では、このようなアプローチを分散ファイルシステムの二階層構成法と呼ぶことにする。二階層構成法をとることによって、ファイルシステムの設計を単純化して自由度を上げることができるので、さまざまな利点が期待できる。

- ストレージ層で分散性を吸収し、ファイルシステム層に対しては物理ディスクと同様のブロック単位の入出力インターフェースを備える仮想ディスクを提供するので、ファイルシステム層は目的に応じてさまざまな設計が可能である。例えば、ネットワークを考慮しないファイルシステムを仮想ディスク上で運用することもできる。
- マルチメディア情報など、サイズが巨大なファイルもブロック単位で管理できるので、このようなファイルをファイル単位で管理する場合に比べ、管理コストを低減できる。
- 集中サーバを持たないので、スケーラビリティを確保できる。
- ストレージ層の設計によっては、ブロック単位で複製を行うことにより比較的容易に冗長性を確保し、耐故障性を向上できる。

2.2 ファイルシステム層とストレージ層の協調

二階層構成法には利点が多いが、効率の点で問題が生じる可能性がある。ストレージ層は個々のブロックがどのように利用されているかの情報を持たないので、適切なブロック管理ができない場合がある。

我々は、ファイルシステム層で管理される情報をブロック管理のヒントとしてストレージ層に与えることにより、ストレージ層でのブロック管理を最適化できるのではないかと考えている。以下に、このようなヒントとして利用できる可能性がある情報の例を示す。

- ファイルに対するアクセスモード
  ファイルの許可情報や、ユーザプログラムからファイルオープン時に与えられるアクセスモードを利用して、ファイルがどのようにアクセスされるかが分かる。例えば、あるブロックが頻繁に更新されることが分かれば、作成するブロックのキャッシュの個数を必要最小限に抑えることによって一貫性保持の効率を向上できる。

- ブロックに格納されている情報の種類
  例えば、inode のようにメタデータが格納されるブロックとファイルが格納されるブロックとでブロックの一貫性管理の厳密性を変えることにより、ブロック管理を効率化できる。

- ファイルの所有者情報
  ファイルを構成するブロックの複製を、ファイルの所有者が頻繁に利用する計算機に配置することができる。

- マウント情報
  仮想ディスクがパーティション[1]に区切られている場合、あるパーティションに含まれるブロックはそのパーティションをマウントしている計算機からしかアクセスされないことを前提にしてブロック管理を行うことができる。

- アクセス統計
  ファイル、あるいはマウントされたパーティションに対するアクセス統計を利用して、特定のブロックの複製を頻繁にアクセスされる計算機に配置するように最適化できる。

3 分散ストレージシステムの設計

現在我々は、二階層構成法に基づく分散ファイルシステムを実装中である。本システムのストレージ層では、ネットワーク全体で単一の仮想ディスクを提供する。仮想ディスクのブロック番号をネットワーク上の各計算機が持つ物理ディスクのブロック番号と対応させて入出力を行うことにより、ストレージ層でネットワークの分散性を吸収する。また、各ブロックの複製を複数の計算機に置くことにより、冗長性を確保するとともに、ブロックアクセス性能の向上をはかる。

例えば図 1 では、ホスト 1 でブロック A が書き込まれている。この書き込みはストレージ層で複製され、ホスト 1、ホスト 2 の物理ディスクにそれぞれ書き込まれる。これはホスト 2 でもブロック A にアクセスするからである。実際、ホスト 2 がブロック A の読み出しをする場合には、ネットワーク転送は発生せず、ホスト 2 の物理ディスクからブロックが読み出される。

本節では、特にこのシステムのストレージ層の設計について述べる。設計にあたっては、以下の 3 点を重視する。

- さまざまなネットワーク媒体を含む広域分散環境への対応
- 計算機間で異なるリソース量への対応
- さまざまなファイル共有パターンへの対応

3.1 広域分散環境への対応

比較的均質な通信環境を備えた LAN と異なり、広域分散環境で分散ストレージシステムを利用するためには、さまざまなネットワーク特性に対応する必要がある。例えば無線ネットワークを想定すると、通信状態によっては通信が非常に遅かったり、接続が切れたりすることがありうる。Coda 分散ファイルシステムでは、このような環境に hoarding と呼ばれる技術を用いて対応している [12, 14]。

本ストレージシステムでも、hoarding に似た技術を応用し、広域分散環境への対応を行う。具体的には、ファイルシステム層から与えられるヒントを用いることにより、必要なブロックの複製を各計算機のローカルディスクに保持する他、ユーザがファイルシステムに対して明示的に必要なファイルを指定するようなインターフェースを用意する。このように各計算機のディスクにブロックの複製を作成し、ネットワークからの切断時には複製のみを用いてファイル操作を行う。ネットワーク接続が回復した時点で、切断中に行われたファイル操作に対応したブロックの一貫性保持動作を行う。

ネットワーク接続が切断されるのではなく、計算機が明示的に停止させられることによって、他の計算機との接続が切断されることがありうる。こ

[1] パーティションという用語は物理ディスクに対して意味を持つもので、ここで想定する仮想ディスクに対して用いるのは厳密には正しくない。

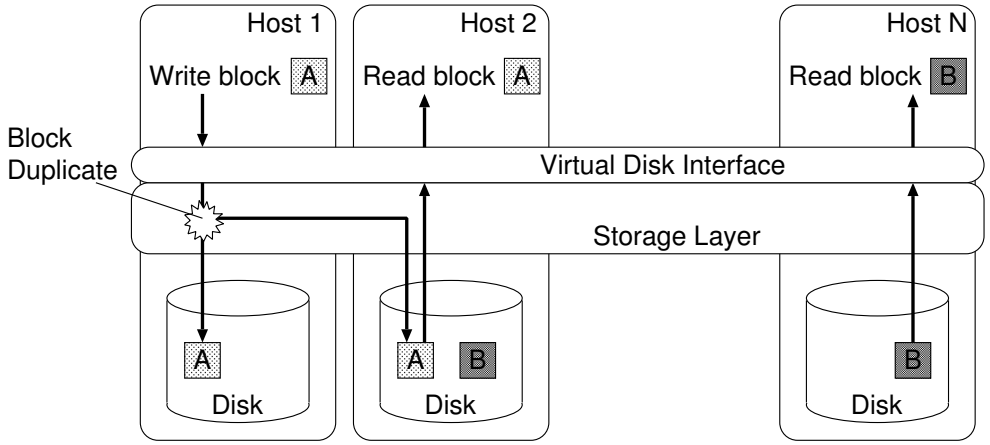

図 1: 本ストレージシステムの概念的な動作

の場合も、再起動時に上記のように一貫性保持動作を行う。

3.2 計算機間で異なるリソース量への対応

本システムでは、各計算機のローカルディスクにブロックの複製を設けることを前提としているが、計算機の種類によってはディスク容量が不十分だったり、全くディスクを持たないということもありうる。このような場合、ブロックアクセスを常にリモートディスクに対して行う動作モードを用意して対応する。また、ユーザがファイルシステムに対して必要とするファイルのみをメモリキャッシュに持つように指示するためのインターフェースを設けることも検討している。

3.3 さまざまなファイル共有パターンへの対応

さまざまなファイル共有パターンに対応するために、ファイルシステム層から与えられるヒントを用いてブロック管理の最適化を行う。

- ファイルの所有者情報を利用して、そのファイルを構成するブロックの複製を所有者が利用する計算機に優先的に配置する。
- ファイルの許可情報を利用して、ファイルを構成するブロックの複製間の一貫性制御プロトコルを変更する。例えば、所有者のみが書き込み可能なファイルに対しては、migratory プロトコルを用いて実際に書き込みが行われる計算機に書き込み権を移動する。
- inode などのメタデータブロックは一般のファイルに比べて多数のユーザからアクセスされる可能性が高いと予想されるので、厳密な一貫性管理を行い、各計算機ができるだけ早く最新の更新を利用できるようにする。メタデータに対する書き込みのサイズは一般のファイルと比べて小さいので、一貫性保持をこのように行うことによる負荷は比較的小さいと考えられる。

4 議論

本節では、今後考えなければならないシステム設計上のいくつかの問題点について議論する。

4.1 ファイルシステムのバックアップ

大規模なファイルシステムを運用する上で、システム障害や誤操作によるデータの消失を防ぐため、バックアップは不可欠である。本ストレージ

システムでは、ブロックの複製をネットワーク上の複数の計算機に配置することにより冗長性を確保するので、ある程度の耐障害性を備えると言える。一方、誤操作によるデータの消失の危険性は依然として残るので、何らかのバックアップ手段を備えることは必須である。

現状では、本システムでは各ファイルがブロックに分割され、ネットワーク上の各計算機に配置されるので、単純にバックアップを行うことはできない。従って、バックアップのために何らかの機能を追加する必要がある。

4.2 障害からの復旧

ネットワークに接続された計算機の一部が障害により異常停止した場合、復旧する方法について考慮する必要がある。本システムでは複数の計算機がブロックの複製を保持するので、一部の計算機が停止した場合でもシステム全体での情報の喪失はある程度抑えられると考えられる。しかし、停止した計算機が再起動した場合、停止中に他の計算機が行ったファイル操作を反映する必要がある。これに関しては、3.1 節で述べたネットワーク接続が切断された計算機が再接続する際の手続きと同様のものを利用し、再起動時に一貫性保持動作を行うことができる。

4.3 セキュリティ

本システムは、現状ではセキュリティについてほとんど考慮していない。しかし、インターネットのような広域分散環境を想定すれば、何らかの形でセキュリティを考慮した設計変更をする必要がある。具体的には、以下の 2 点が必要となる。

- ネットワークを介して転送される情報の暗号化
- 計算機・ユーザの認証機構

また、システムを構成する各計算機が信頼できないという前提に立てば、更に考慮しなければならないことがあると考えられる。

4.4 高性能入出力

並列入出力を行って入出力性能を向上させるためにディスクストライピングを利用する研究は数多く行われているが、本システムが想定している作業負荷ではそれほど高性能の入出力を必要としないので、並列入出力については考慮しない。ただし、一定の入出力性能は必要なので、cooperative caching[6] のような技術を用いて、メモリ上のブロックキャッシュ管理を物理ディスクブロック管理とネットワーク全体で統合して入出力性能を向上させることは必要かもしれない。

5 関連研究

ネットワークに接続された複数の計算機に接続されたディスク装置を用いて、全体として単一のファイルシステムとして動作させるようなシステムは、これまで数多く提案されている。

Swift[5, 4] は、ネットワークで接続された複数の計算機の持つディスクを用いてストライピングを行った最初のシステムである。ストライピングによる I/O の高速化を目的としていた他、パリティディスクを用いて冗長化を行うことによる信頼性向上にも言及している。

Zebra[9, 10, 11] は、ネットワークに接続された計算機のディスク装置を RAID (Redundant Arrays of Inexpensive Disks)[16] と同様に用いてストライピングを行うことにより、ファイルアクセス性能を向上する分散ファイルシステムである。Log-structured ファイルシステム [17, 19] の手法を用いることによって、サイズの小さなファイルの書き込みにおいても性能低下を押さえるような工夫がなされている。xFS[2, 3] は、Zebra の手法を改良し、分散処理をさらに進めたシステムである。

FARSITE[1] は、安全ではなく互いに信頼できない計算機群を用いて安全で信頼できる仮想ファイルサーバを構成するシステムである。集中サーバを廃したスケーラブルなシステムであるが、ファイルシステム層とストレージ層を分離する二階層

構成法は取っていない。

Logical Disk[7] は、ネットワーク上の計算機が持つディスクを用いて仮想ディスクを提供するシステムである。我々の知る限り、ファイルシステム層とストレージ層を分離する二階層構成法に基づく初めてのシステムである。ブロック管理に Log-structured ファイルシステムの手法を取り入れている。

Petal[13] と Frangipani[22] はファイルシステム層とストレージ層の二層構造をとる分散ファイルシステムである。前者が分散ストレージシステムであり、後者が前者の上で動作するファイルシステムにあたる。Petal では、分散性の吸収やブロックの複製による冗長性の確保が行われているほか、copy-on-write を用いて仮想ディスクのスナップショットを取ってバックアップを行う機能も提供されている。しかし、ファイルシステムである Frangipani の管理情報を Petal で利用するようなことは行われていない。

Swarm[8, 15] は、ブロック単位のインターフェースではなく、log-structured ファイルシステムの手法に基づくインターフェースを備えた分散ストレージシステムである。Striped log と呼ばれる抽象化を提供することにより、Swarm の上で必要なファイルサービス・ブロックサービスを効率的に構築することができる。

PersonalRAID[20] は、分散ストレージシステムである。単に複数の計算機にブロックを複製するだけではなく、持ち運び可能なリムーバブルディスクにも書き込みを行ったブロックをログとして記録する。このディスクを持ち運ぶことにより、互いにネットワークで接続されていない計算機同士の間でもファイル共有を透明に行うことができる。しかし、ネットワーク接続が利用できる場合にこれを積極的に利用してファイル共有を行うようなことは考慮されていない。

6 まとめ

我々は、個人が複数の計算機を状況に応じて使い分け、ネットワーク接続が常時可能となる環境を想定し、ファイルシステム層とストレージ層の二階層構成法に基づく分散ファイルシステムを開発している。本稿では、この分散ファイルシステムのストレージ層として動作する分散ストレージシステムについて述べた。この分散ストレージシステムは、上位層であるファイルシステムに対してブロック単位の入出力インターフェースを持つ仮想ディスクを提供する。仮想ディスクのブロックとその複製ブロックを物理的なディスクに分散配置することによって、分散性を吸収するとともに冗長性を確保する。ファイルシステムで管理されるさまざまな情報をストレージシステムにヒントとして与えるインターフェースを設けることにより、ブロック管理を最適化し、より柔軟なシステム構築が可能になると期待できる。

謝辞

この研究は、財団法人 堀情報科学振興財団 および 南山大学パッヘ研究奨励金 (Pache Research Subsidy)I-A の助成を受けています。

参考文献

[1] Atul Adya, William J. Bolosky, Miguel Castro, Gerald Cermak, Ronnie Chaiken, John R. Douceur, Jon Howell, Jacob R. Lorch, Marvin Theimer, and Roger P. Wattenhofer. FARSITE: Federated, Available, and Reliable Storage for an Incompletely Trusted Environment. In *Proceedings of the 5th Symposium on Operating Systems Design and Implementation (OSDI '02)*, December 2002.

[2] Thomas E. Anderson, Michael D. Dahlin, Jeanna M. Neefe, David A. Patterson, Drew S. Roselli, and Randolph Y. Wang. Serverless Network File Systems. In *Proceedings of the 15th ACM Symposium on Operating Systems Principles*, pages 109–126, December 1995.

[3] Thomas E. Anderson, Michael D. Dahlin, Jeanna M. Neefe, David A. Patterson, Drew S.

Roselli, and Randolph Y. Wang. Serverless Network File Systems. *ACM Transactions on Computer Systems*, 14(1):41–79, February 1996.

[4] Luis-Felipe Cabrera and Darell D. E. Long. Exploiting Multiple I/O Streams to Provide High Data-Rates. In *Proceedings of the 1991 Summer Usenix Conference*, pages 31–48, 1991.

[5] Luis-Felipe Cabrera and Darrell D. E. Long. Swift: Using Distributed Disk Striping to Provide High I/O Data Rates. *USENIX Computing Systems*, 4:405–436, 1991. Swift: Using Distributed Disk Striping to Provide High I/O Data Rates.

[6] Michael D. Dahlin, Thomas E. Anderson, David A. Patterson, and Randolph Y. Wang. Cooperative Caching: Using Remote Client Memory to Improve File System Performance. In *Proceedings of the First Symposium on Operating Systems Design and Implementation*, pages 267–280, 1994.

[7] Wiebren de Jonge, M. Frans Kaashoek, and Wilson C. Hsieh. The Logical Disk: A New Approach to Improving File Systems. In *Proceedings of the 14th ACM Symposium on Operating Systems Principles (SOSP)*, pages 15–28, December 1993. The Logical Disk: A New Approach to Improving File Systems.

[8] John H. Hartman, Ian Murdock, and Tammo Spalink. The Swarm Scalable Storage System. In *Proceedings of the 19th IEEE International Conference on Distributed Computing Systems (ICDCS '99)*, pages 74–81, June 1999. Also available as Technical Report TR99-06, Department of Computer Science, University of Arizona, March 1999.

[9] John H. Hartman and John K. Ousterhout. Zebra: A Striped Network File System. In *Proceedings of the Usenix File Systems Workshop*, pages 71–78, May 1992.

[10] John H. Hartman and John K. Ousterhout. The Zebra Striped Network File System. In *Proceedings of the Fourteenth ACM Symposium on Operating Systems Principles*, pages 29–43, 1993.

[11] John H. Hartman and John K. Ousterhout. The Zebra Striped Network File System. *ACM Transactions on Computer Systems*, 13(3):274–310, August 1995.

[12] James J. Kistler and M. Satyanarayanan. Disconnected Operation in the Coda File System. In *Proceedings of 13th ACM Symposium on Operating Systems Principles*, pages 213–225, October 1991.

[13] Edward K. Lee and Chandramohan A. Thekkath. Petal: Distributed Virtual Disks. In *Proceedings of the 7th International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS VII)*, pages 84–92, October 1996.

[14] Lily B. Mummert, Maria R. Ebling, and M. Satyanarayanan. Exploiting Weak Connectivity for Mobile File Access. In *Proceedings of the 15th ACM Symposium on Operating Systems Principles*, pages 143–155, December 1995.

[15] Ian Murdock and John H. Hartman. Swarm: A Log-Structured Storage System for Linux. In *Proceedings of FREENIX Track: 2000 USENIX Annual Technical Conference*, June 2000.

[16] David A. Patterson, Garth Gibson, and Randy H. Katz. A Case for Redundant Arrays of Inexpensive Disks(RAID). In *ACM SIGMOD*, pages 109–116, June 1988.

[17] Mendel Rosenblum and John K. Ousterhout. The Design and Implementation of a Log-Structured File System. In *Proceedings of 13th ACM Symposium on Operating Systems Principles*, pages 1–15, October 1991.

[18] Russel Sandberg, David Goldberg, Steve Kleiman, Dan Walsh, and Bob Lyon. Design and Implementation of the Sun Network Filesystem. In *Proceeding of the Summer 1985 USENIX Conference*, pages 119–130, Portland OR (USA), June 1985. USENIX.

[19] M. Seltzer, K. Bostic, M. K. McKusick, and C. Staelin. An Implementation of a Log-Structured File System for UNIX. In *Proceedings of the Winter 1993 USENIX Technical Conference*, pages 307–326. USENIX Association, 1993.

[20] Sumeet Sobti, Nitin Garg, Chi Zhang, and Xiang Yu. PersonalRAID: Mobile Storage for Distributed and Disconnected Computers. In *Proceedings of the FAST 2002 Conference on File and Storage Technologies*, January 2002.

[21] Sun Microsystems, Inc. NFS: Network File System Protocol Specification. RFC 1094, Network Information Center, SRI International, March 1989.

[22] Chandoramohan A. Thekkath, Timothy Mann, and Edward K. Lee. Frangipani: A Scalable Distributed File System. In *Proceedings of the Sixteenth ACM Symposium on Operating Systems Principles*, pages 224–237. ACM, October 1997.