LB-8　シングルイメージクラスタシステムにおける高信頼ストレージ機能の設計

Design of High Available Storage for Single Image Cluster System

矢野 浩邦† 佐藤 記代子† 前田 誠司†
Hirokuni Yano Kiyoko Sato Seiji Maeda

1　はじめに

近年の社会において，情報システムは必要不可欠な存在となりつつある．しかし，システムの故障やオペレータのミスによるトラブルは多い．このようなトラブルを防ぐために，我々は『簡単で柔軟』をキーワードに，一般的な計算機を用いた高信頼クラスタシステムを開発している．

本システムは,3 台以上の計算機ノードから構成されるソフトウェアによるシングルイメージクラスタシステムであり，システム全体を 1 台の計算機に見せている．システム全体をシングルイメージとして見せることによって，計算機の分散や障害を意識する必要がなくなり，システムの開発，運用，保守が容易になる．

本クラスタシステムのイメージを図 1 に示す．

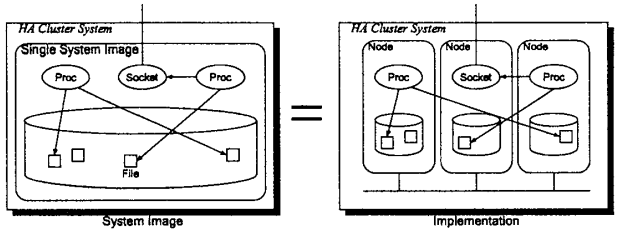

図 1: 本システムの概要

筆者らは，上記シングルイメージクラスタシステムの高信頼ストレージ機能の開発を行っている．

本システムでは，システム上のプロセスに対してシングルイメージ化されたストレージ機能を提供する．プロセスに対して見せるファイルの実体を複数の計算機ノードのハードディスクに多重化して格納することで，システム上のデータの保護を実現する．また，ストレージをシングルイメージとして見せる際に，ファイルに対するデータの入出力の機能だけを提供する『仮想ファイル』という概念でストレージ機能を抽象化することによって，アプリケーションに応じた最適化を実現する．

本稿では，我々が開発しているシングルイメージクラスタシステムにおける高信頼ストレージ機能と従来のストレージの構成との違いについて述べ，本システムで導入する仮想ファイルの概念と，その有効性と実現方法について説明し，今後の計画を述べる．

2　従来のクラスタのストレージ構成との違い

本章では，クラスタシステムのストレージ機能をシングルイメージとして提供する場合の，従来の構成における問題点と，我々のシステムのストレージ機能の構成による解決方法について述べる．

2.1　従来の高信頼ストレージの構成と問題点

従来のクラスタシステムでは，システム上のデータを保護するために，RAID(Redundant Arrays of Inexpensive Disks) に代表される高信頼ストレージ装置を用いて，データを格納することが多い.RAID を使用すると，ディスク故障からのデータの保護を行うことはできる．しかし，RAID が接続されている計算機に障害が発生すると，その計算機が単一障害点となり，RAID 内のデータにアクセスすることができなくなり，システム全体を保護できないという問題がある．

この問題は，SAN(Storage Area Network) 等に代表される共有ディスクを用いて，ストレージ装置へのパスを冗長にすることによって解決されている．しかし，SAN は，共有ディスク上のデータに対するアクセスの際に，データの一貫性を保つために，上位で排他制御を行う必要がある．また，共有ディスク上の区画管理も上位に委譲されているため，上位のソフトウェアが複雑になるという問題がある．

2.2　本システムにおける構成

従来の構成と我々の提案する構成の違いを図 2 に示す．左図が従来の構成であり，右図が我々が提案する構成である．

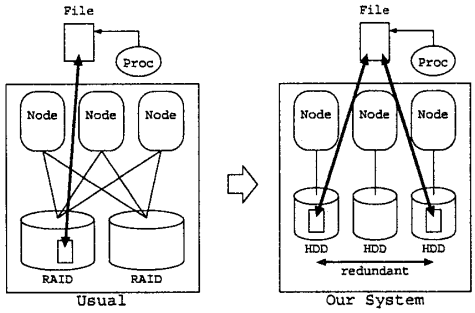

図 2: 高信頼シングルイメージクラスタのストレージ構成

我々のシステムでは，システム上のプロセスに見せるファイルの実体 (実ファイル) を複数の計算機ノードのハードディスク上に格納し，プロセスに見せるファイルと実ファイルの対応付けを多重化するという手法を用いて，従来のストレージ構成にあった問題を解決する．実ファイルを複数用意し，計算機ノード間で多重化を行うので，実ファイルを格納してある計算機ノードに障害が発生した場合においても，他の計算機ノードに多重化されて格納されている実ファイルを利用することが可能であり，システム全体の保護を実現できる．また，ストレージ上のデータの排他制御や区画管理の機能もシステムで提供することによって，上位のソフトウェアの作成を容易にする．

3　ストレージ機能の抽象化手法

本章では，ストレージをシステム上のプロセスに対して，シングルイメージとして見せる際の，我々がとった抽象化手法と，その利点について述べる．

3.1　従来の抽象化手法と問題点

クラスタシステム上のプロセスに対してストレージを単一のリソースとして見せる方法として，

- 連続したひとつの大きな記憶領域
- ファイルシステム

という抽象化が一般的に多く用いられている．

ストレージ機能を，1 台のハードディスクのような連続したひとつの大きな記憶領域として提供する場合は，上位でその領域上にファイルシステムを構築して，領域内の区画を管理し，データを格納することになる．しかし，この方法を用いると，ア

†(株) 東芝 研究開発センター コンピュータ・ネットワークラボラトリー

プリケーションがファイルという単位でデータにアクセスしているにも関わらず，クラスタシステム側ではその単位を知ることができず，アクセス単位の最適化を行うことが難しい．

一方，分散ファイルシステムとして提供する場合は，アプリケーションに対して，その分散ファイルシステムがファイルを扱うセマンティクスを強制してしまう．そのためアプリケーション側ではデータを扱う方法をファイルシステムに合わせる必要があり，ファイルシステムがアプリケーションの性能を低下させる原因となる場合もある．たとえば，データベースマネージメントシステムのような，独自にデータ構造やファイルへの入出力を管理するアプリケーションの場合，ファイルシステムのオーバーヘッドが大きいと，アプリケーションの本来の性能を引き出すことができなくなるという問題がある．

3.2 仮想ファイルの導入

一般的に，ファイルシステムには，大きく，

- ファイルに対するデータ入出力
- ファイル間の関係の管理

の 2 つの役割があると考えられる．

前者のデータの入出力に関しては，計算機アプリケーションの多くがデータをファイルという単位でストレージに保存しており，また，どのアプリケーションでも同様に用いていることから，必要な役割といえる．

我々はこの点に着目し，ファイル間の関係を管理することはシステム側では行わず，アプリケーションがデータを格納するための最低限の機能である『ファイル』という単位でのみストレージ機能を提供する．このアプリケーションが扱うデータの単位であるファイルを，仮想ファイルと呼んでいる．

ファイルという単位でストレージ機能を提供することにより，システム側でアプリケーションがデータを扱う単位を知ることが可能となり，システム側で，データ単位の負荷分散や高信頼化を行うことができる．

そして，図 3 のように，本システムの上で動作するミドルウェアが，対象のアプリケーションに適した方法で仮想ファイル間の関係を定義し，専用のファイルシステムを作ることが可能となる．

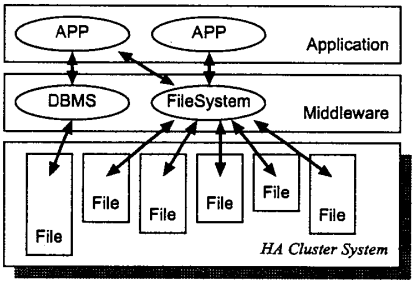

図 3: 仮想ファイルとミドルウェア

このように，プリミティブかつ扱いやすい単位でストレージ機能を抽象化して提供することにより，データ本体の入出力の機能をシステムで提供しつつ，アプリケーションに応じた最適化が可能となる．

さらに，我々の提案する仮想ファイルでは，2 章で述べたように，ひとつの仮想ファイルに対して，実ファイルを複数の計算機ノードに多重化して格納し，仮想ファイルと実ファイルの関係も多重化することで高信頼化を行い，シングルイメージ化することによって，ディスク障害とファイル配置を意識しないデータアクセスを実現する．

4 高信頼ストレージ機能の実現

本システムのストレージ機能である仮想ファイルの動作概要を図 4 に示す．ここでは，ある仮想ファイル A に注目して，代表的な動作を説明する．

仮想ファイルには，システム内で一意の ID(A) がつけられ，プロセスはその ID を元にファイルアクセスを行う．仮想ファイル A に対応する実ファイル A1, A2 は複数のノードに多重化されて格納されており，システムは，仮想ファイルと実ファイルの対応付けを管理している．

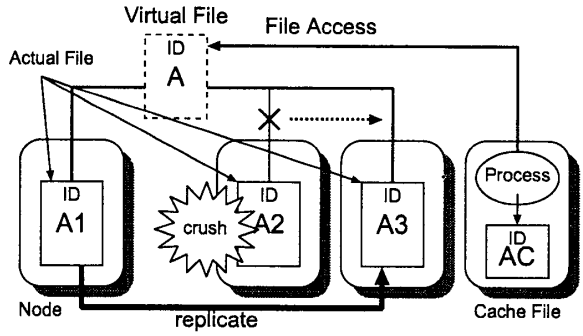

図 4: 仮想ファイルの実現

4.1 通常のファイルアクセス時の動作

プロセスから仮想ファイル A に対してアクセスが行われると，システムは実ファイル A1, A2 が格納されている計算機ノードを調べ，実ファイルに対してアクセスする．参照は，A1, A2 のどちらかの実ファイルに対して行えばよいが，更新は，実ファイルに障害が発生してもデータを失わないように，A1, A2 の両方に対して行う．更新時に，A1, A2 の内容が同一になるように，下位に追記型のファイルシステムを導入して，一括更新を行ってファイルの内容の一貫性を保つ機構も組み込んでいる．

4.2 障害時の動作

実ファイル A2 が障害によって使用できなくなった場合，正常な内容である実ファイル A1 を他の計算機ノードに複製し，新たな実ファイル A3 とする．この実ファイル A3 を新たに仮想ファイル A と結び付けることで，多重度を保つ．この動作は，システムで行うため，プロセスはファイルの障害を意識する必要なしにファイルアクセスを行うことができる．

4.3 キャッシュによる高速化

実ファイルを持たない計算機ノード上のプロセスから実ファイルへアクセスを行う場合，計算機ノード間の通信のオーバーヘッドによりファイルアクセスが遅くなる．計算機ノード間通信を減らすため，プロセスが動作しているノードにキャッシュファイル AC を作成し，キャッシュファイルにアクセスすることによって，ファイルアクセスを高速化する．

5 まとめ

我々は『簡単で柔軟』をキーワードに，一般的な計算機を用いた，ソフトウエアによる高信頼シングルイメージクラスタシステムを開発している．

本稿では，まず，我々の提案するシングルイメージクラスタシステムの高信頼ストレージ機能の設計において，その高信頼化の手法と構成上の特長を述べた．次に，ファイルに対する入出力の機能のみをシステムで提供する，仮想ファイルという抽象化の概念を導入することによって，アプリケーションに応じた最適化を実現していることを述べた．最後に，その基本的な動作を説明した．

現在，システムの実装を進めており，その実装を使っての評価を，高信頼性と性能面から行っていく予定である．

参考文献

[1] Sun Microsystems, Inc, Sun™ Cluster 3.0, `http://www.sun.com/software/cluster/`

[2] Eliezer Levy and Abraham Silberschatz, Distributed file systems: concepts and examples, ACM Computing Surveys Volume 22 Issue 4 pp.321-374, 1990