B_037

ユーザによるファイルシステム拡張機能の管理操作を可能にするコマンドファイル方式の提案

石井　陽介†　薗田　浩二†

（株）日立製作所 システム開発研究所†

1　はじめに

近年、広く利用されているファイルサーバは、電子ファイルの集約管理ならびに複数ユーザによるデータ共有を実現する。また、ユーザの利便性向上のために、ファイルサーバが提供するファイルシステムのさらなる高機能化もなされている。新しいファイルシステムの拡張機能を利用するために、ユーザからの操作が必要な場合は、クライアントマシンにおける従来プロトコルの拡張あるいは新規プロトコルの追加が必要となる。従来、プロトコルの拡張や新規追加のためには、OS の改造を必要としていた。しかし、様々なベンダや団体が多様な OS を提供している現況において、クライアントマシン側の OS を全て改造することは難しいという問題があった。そこで本稿では、既存インタフェースを利用することで、拡張機能を利用するためのユーザ操作を可能にするコマンドファイル方式を提案する。本方式により、ユーザは、OS 改造を要するプロトコルの拡張や新規追加なしに、ファイルシステム拡張機能の利用が可能になる。

2　課題

従来、サーバマシン上ファイルシステムの拡張機能利用のために、クライアントマシンから操作が必要な場合は、図1に示すように、クライアントマシン側の OS に拡張インタフェースを提供する改造を施す必要があった。しかし、クライアントマシンで利用可能な OS の種類は多岐に渡るため、これら全ての OS の改造は難しい。また、一般に、ネットワーク環境では、セキュリティ対策としてフィルタリング機能により未使用ポートを塞ぐ設定をしている。ここで、拡張インタフェースや新規プロトコルの追加を行う場合、新しいポートを利用するためには、設定変更が必要となる。しかし、安易な変更は、セキュリティホールを新たに生む可能性がある。したがって、機能拡張のために既存ネットワーク環境のセキュリティ設定変更は難しい。

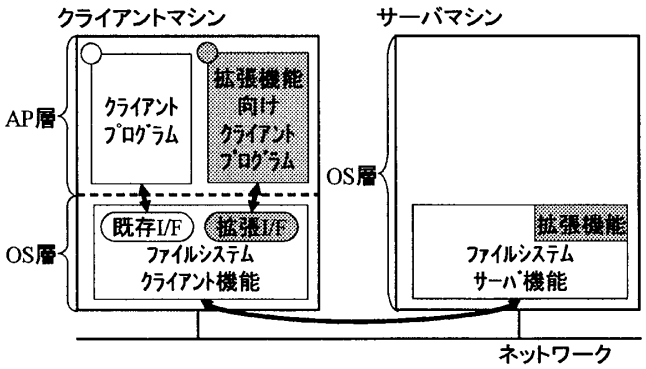

図1　従来の拡張方式

拡張機能を利用するために、ユーザからの操作を可能にするためには、以下の課題を解決する必要がある。

（課題1）OS非改造による拡張I/F提供

クライアント側に提供する拡張インタフェースの実現方式には、OS 改造あるいは OS 非改造による方式がある。OS 改造による方式には、前述のように、様々な OS へ個別に対応するアプローチがあるが、実施困難である。また、拡張インタフェースを標準化して、各 OS にサポートしてもらうアプローチもある。しかし、多くの OS がサポートするまでに長い期間を要してしまう。このため、OS 非改造で拡張インタフェースを提供する必要がある。

（課題2）既存ネットワーク環境への影響局所化

拡張機能の操作要求をマシン間で通信可能にするためには、当該通信用に専用の通信ポート番号を新たに割当てる方式あるいは既存のポート番号を利用する方式がある。ポート追加は、前述のように、既存環境におけるセキュリティ設定の変更が必要であり難しい。このため、既存環境への影響を与えないよう、新たなポート追加が不要な後者の方式にする必要がある。

3　コマンドファイル方式

3.1 課題解決の着眼点

前章の課題解決にあたり、我々は、ファイルシステムが持つファイルの入出力機能に着目した。既存のファイル I/O インタフェースを利用することで、ユーザは、サーバマシンにバイトス

Proposal of the Command File Method for Management Operation of Extended Filesystem Function by Users

Yousuke ISHII† and Koji SONODA†

† Systems Development Laboratory, Hitachi, Ltd.

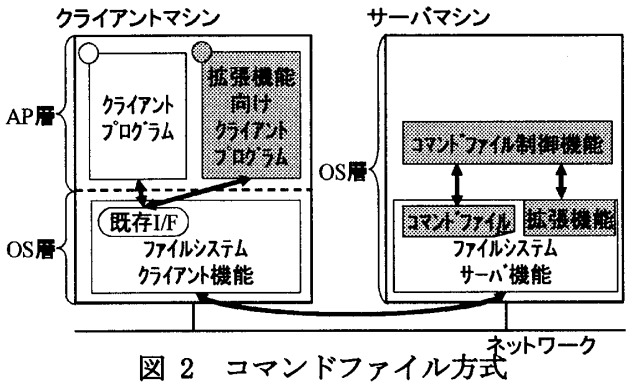

図 2　コマンドファイル方式

トリームを送ることができる。そこで、サーバマシンが、当該バイトストリームを制御コマンドとして解析し、必要な処理を実行するというアプローチを考えた。

3.2 基本方式

本稿では、前節のアプローチに基づいたコマンドファイル方式を提案する。図 2 に本方式のシステム概観を示す。本方式は、拡張機能向けの制御コマンドをラッピングしたデータを、既存インタフェースの read/write 操作を利用してファイルシステムとやり取りする方式である。

本方式は、既存ファイル I/O を利用して制御コマンドをサーバマシンに送信可能なため、クライアントマシンの OS 改造は不要である。また、既存ファイル I/O で操作要求を格納した制御コマンドを転送するため、既存ファイル I/O 用のポートを利用できる。新たなポート追加は不要であり、既存ネットワーク環境への影響もない。

3.3 実現方式

本方式を実現するためには、以下の解決すべき点がある。本節では、各々の解決法を述べる。

（1）I/Oデータと制御コマンドデータの分離

本方式では、制御コマンドのやり取りに既存ファイル I/O を利用するため、ファイル I/O 処理において、従来の I/O データと制御コマンドデータとを識別可能にする必要がある。ここでは、サーバマシンが特別な予約名を規定し、当該予約名を持つファイル（以降、コマンドファイルと呼ぶ）に書き込まれたデータを制御コマンドとして識別する。サーバマシンは、コマンドファイルへの書き込み処理毎に制御コマンドを解析し、対象拡張機能の処理を呼び出し、実行結果を当該制御コマンドデータ内でユーザから指定された名前を持つファイル（以降、結果ファイルと呼ぶ）に出力する。ユーザは、既存ファイル操作により、コマンドファイルならびに結果ファイルへの操作が可能である。

（2）拡張可能な制御コマンド書式

本方式では、制御コマンド書式に XML を利用する。また、制御コマンド用 XML の書式を拡張

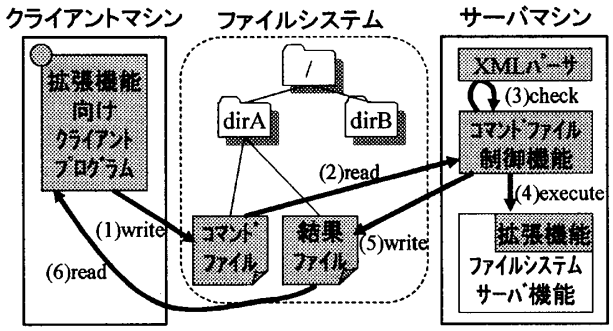

図 3　コマンドファイル方式の処理機構

可能にする。これにより、当該拡張機能をさらに拡張する場合にも、当該 XML 書式の拡張およびクライアント AP の改造で対応できる。さらに、ユーザからの要求受付ならびに処理実行のための統一制御フレームワークを提供できる。以上より、将来のさらなる機能拡張に備えた、拡張性のある統一制御コマンド書式を提供できる。

3.4 処理機構

図 3 に処理機構を示し、以下に説明する。

(1)クライアントプログラムは、予め規定された名前でコマンドファイルを作成し、XML 形式の制御コマンドデータを既存インタフェースの write 操作で書き込む。

(2)サーバマシンは、コマンドファイルへの write 操作を契機に、当該ファイルから制御コマンドデータを読み出す。

(3)サーバマシンは、XML パーサと連携し、読み出した制御コマンドデータを解析する。

(4)サーバマシンは、要求された処理を実行するよう対象拡張機能に依頼する。

(5)サーバマシンは、実行結果をファイルシステム上の指定された結果ファイルに書く。

(6)クライアントプログラムは、結果ファイルを既存インタフェースの read 操作で読み、実行結果を取得する。

4　おわりに

本稿では、拡張機能向け制御コマンドを、既存ファイル I/O を利用してファイルシステムとやり取りするコマンドファイル方式を提案した。ユーザは、OS 改造を要するプロトコルの拡張や新規追加なしに、ファイルシステム拡張機能の利用が可能になる。本方式により、OS 非改造による拡張インタフェース提供ならびに既存ネットワーク環境への影響局所化が可能となる見通しを得た。今後は、本方式を実装し、有効性を検証していく予定である。