FTP(TCP/IP)利用調査研究報告書

～ EDIの利用に向けて ～

（第1.0版）

平成８年２月

財団法人　日本情報処理開発協会
産業情報化推進センター

はじめに

近年、企業内外におけるネットワークのＬＡＮ化の進展により、ＬＡＮ上のネットワークプロトコルとしてＴＣＰ／ＩＰ（Transmissin Control Protocol / Internet Protocol）が多くのユーザ企業に利用されつつあります。また、ＴＣＰ／ＩＰはＬＡＮ環境以外にもルータ経由のＷＡＮにも広く利用されています。

ＴＣＰ／ＩＰ上で動作する代表的なアプリケーション・プロトコルには、ファイル転送を行うＦＴＰ（File Transfer Protocol）、電子メールを転送するＳＭＴＰ（Simple Mail Transfer Protocol）などがあります。

その中で、ＦＴＰは、コマンドの入力によりファイル転送サービスが提供されるプロトコルであり、市場に流通する一般的なＦＴＰ製品も、利用者がマン・マシン・インタフェースを介してコマンドをキー入力することにより、サーバシステムとの間でファイル転送などが行われます。しかし、ＥＤＩ等の業務への適用のため、マン・マシン・インタフェースを介さずに、システムに組み込み式のＦＴＰパッケージ製品が各ベンダより提供されつつあります。このような状況下で、現状のＦＴＰ製品の範囲内で、どの程度、ＥＤＩ業務への適用が図れるかを調査研究し、それをユーザ向け利用ガイドラインとして取りまとめることが望まれます。

当センターでは、上記の趣旨により、「新手順検討委員会」の下部組織として「ＦＴＰ検討ＷＧ」を設置し、我が国のＥＤＩではファイル転送の業務形態が多く利用されていることからＦＴＰにスポットを当て、ＥＤＩ業務への適用について検討を行い、本報告書を作成しました。

なお、必要に応じて、ＥＤＩ業務への適用に当たり現状のＦＴＰ製品が満足できないレベルであれば、今後、実装ガイドラインを取りまとめることも必要となるかもしれません。

最後に、本調査研究にご協力を頂きました「新手順検討委員会」及び「ＦＴＰ検討ＷＧ」の委員・メンバーなどの関係各位に感謝の意を表します。

財団法人　日本情報処理開発協会
産業情報化推進センター

ＦＴＰ検討WG委員名簿

（敬称略・五十音順）

主　査	木村　道弘	日本電気株式会社
委　員	岩上　公一	日本ユニシス株式会社
〃	小川　義和	株式会社日立情報ネットワーク
〃	金武　　貢	日本アイ・ビー・エム株式会社
〃	口田　順次	株式会社富士通神戸エンジニアリング
〃	二階堂　秀治	三菱電機株式会社
〃	美門　伸也	日本電気ソフトウェア株式会社
〃	森　　玲子	日本アイ・ビー・エム・システムズ・エンジニアリング株式会社
〃	山口　博章	株式会社東芝
〃	吉村　政幸	日本電気ソフトウェア株式会社
ｵﾌﾞｻﾞｰﾊﾞ	三木　良治	財団法人　日本情報処理開発協会　産業情報化推進センター
事務局	老川　健治	財団法人　日本情報処理開発協会　産業情報化推進センター
〃	向山　洋二	財団法人　日本情報処理開発協会　産業情報化推進センター

目　次

1．目的と適用範囲 …… 1
　1．1　目的 …… 1
　1．2　適用範囲 …… 1

2．ＦＴＰの概要 …… 2
　2．1　ＦＴＰの利用モデル …… 2
　2．2　ＦＴＰとＦ手順／全銀手順との違い …… 4
　2．3　主な外部仕様 …… 4

3．ＦＴＰに対するリクワイアメント …… 6

4．利用上のガイドライン …… 7
　4．1　データ圧縮 …… 8
　4．2　ゼロ件データ転送 …… 10
　4．3　データ転送の強制中断 …… 10
　4．4　二重交換防止 …… 11
　4．5　送受信時のジョブ起動 …… 12
　4．6　セキュリティ機能 …… 15
　4．7　ファイル成立管理 …… 17
　4．8　ファイル名と文字種別 …… 20
　4．9　転送データ・タイプ …… 20
　4．10　ＡＰＰＥＮＤモード …… 21

5．実装ガイド …… 22
　5．1　ＶＡＮセンターの対応 …… 22
　5．2　障害／ステータス通知 …… 23

6．ＦＴＰ利用におけるファイアウォール環境 …… 26

付録Ａ．ＥＤＩ利用時の標準機能（Ｆ手順対応：産業情報化推進センター編）
　　とＦＴＰのサポートレベル …… 30
付録Ｂ．ＲＦＣ９５９規約書（抜粋） …… 31

1．目的と適用範囲

1．1　目的

本書では、ＦＴＰをサポートした製品が、既に市場に流通していることを踏まえ、現状のＦＴＰ製品を改造することなくそのまま使ってＥＤＩを行うために、どのような考慮が必要か調査研究し、ユーザ向け利用ガイドラインとして取りまとめます。

なお、現状のＦＴＰ製品でサポートされていなくても、ＥＤＩとして必要な機能項目については、必要に応じてその実装例についても記述します。

1．2　適用範囲

本ガイドラインの適用範囲は、以下の通りです。

① 対象とするＥＤＩモデルは、取引相手との継続取引により、ＥＤＩに関する業務の取り決めを当事者間で事前に行い、かつ相手先が固定的に決まっている『従来型ＥＤＩ』を想定しています。更に、このモデルでは、ＥＤＩ用システムが、専用システムとして構築されていることを想定しています。

② ここで示すガイドラインは、性格上、必ずしも流通しているＦＴＰ製品（または、システム）の全てに適用できることを意図したものではありません。従って、必要に応じて導入予定製品の実装状況の事前確認が必要です。

2．ＦＴＰの概要

ＦＴＰ(File Transfer Protocol)は、ＴＣＰ／ＩＰ上で動作するファイル転送プロトコルであり、予め定められた権限を持つユーザに対して、相手側の計算機にログインし、ファイルのコピーや簡単なコマンドを実行するサービスを提供します。

ＦＴＰの仕様は、ＲＦＣ９５９（付録Ｂ参照）に基づいており、この中では、ＦＴＰの通信モデル、データのタイプ、転送データの構造、転送モードなどが決められています。

ＲＦＣ(Request For Comments)は、インターネットでのレポート、提案、プロトコルなどのドキュメントとして広く知られています。

2．１　ＦＴＰの利用モデル

図２－１に示すとおり、ＦＴＰには、サーバＦＴＰとクライアントＦＴＰの２種類のプログラムがあり、一方の計算機にはサーバＦＴＰを、もう一方の計算機にはクライアントＦＴＰを、対の形に組み合わせて実行することによりファイル転送を行うことができます。

ＦＴＰの起動はクライアントＦＴＰのみで可能で、これは起動側計算機に必要なプログラムです。そして、サーバＦＴＰが実装されている相手計算機（＝被起動側計算機）へ、ファイルのコピーや簡単なコマンドを発行し、ファイルの転送などを行います。一方、サーバＦＴＰは、被起動側計算機に常駐し、起動側計算機からの接続要求やコマンドを処理するプログラムです。

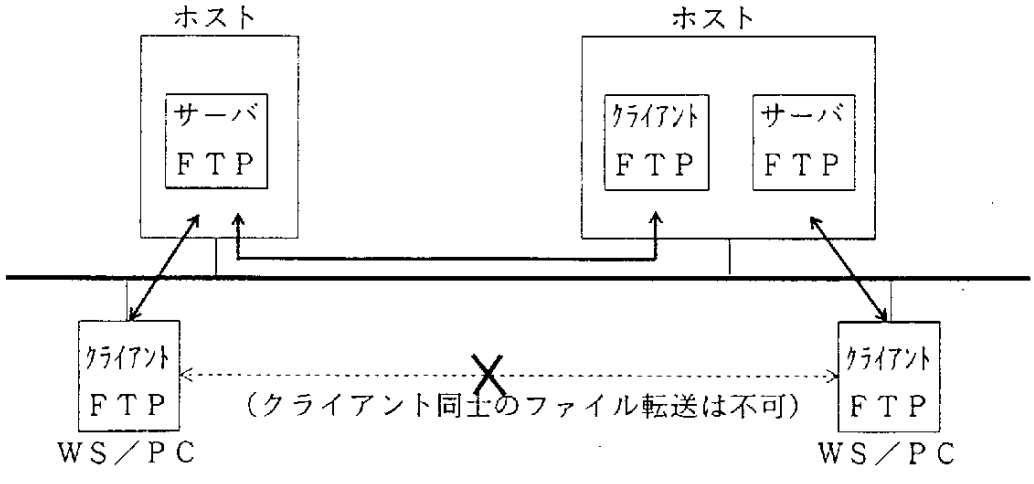


図２－１　利用モデル

また、多くの計算機には、サーバＦＴＰとクライアントＦＴＰの両方が実装されていますが、中には、サーバＦＴＰのみ（汎用機に多い）、クライアントＦＴＰのみ（パソコンに多い）しか実装していない計算機もあり、お互いの計算機の実装状況について十分に確認しておく必要があります。

次に、サーバＦＴＰとクライアントＦＴＰの通信モデルを図２－２に示します。サーバＦＴＰとクライアントＦＴＰは、２種類のＴＣＰコネクションを作成します。一つはサーバＰＩ（プロトコル・インタプリタ）とユーザＰＩ（クライアント側）間でファイル転送の制御を行うために使う制御用のＴＣＰコネクションです。このコネクション上の制御用のコマンドはTELNETプロトコルを使用して通信します。もう一つはデータ転送時にサーバＤＴＰ（データ転送プロセス）とユーザＤＴＰ間のデータ転送に使う、データ転送用のＴＣＰコネクションです。なお、サーバとクライアントのファイルシステムの違いは各ＤＴＰごとに吸収されています。

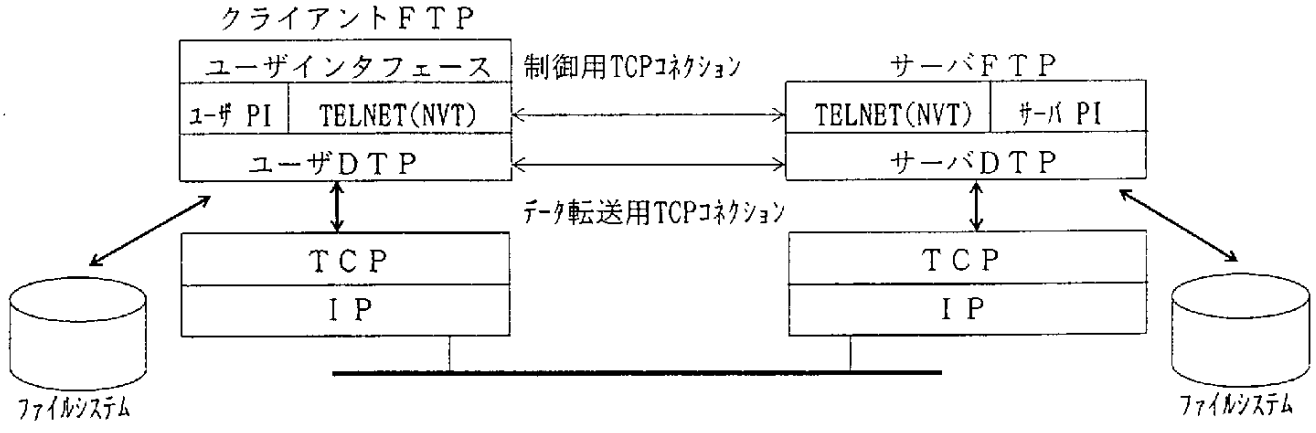


図２－２　通信モデル

2．2　ＦＴＰとＦ手順／全銀手順との違い

ＦＴＰは相手側の計算機にログインすることにより、そのログインユーザの権限を持つファイルに対してコピーしたり、内部コマンドを使用してファイルを削除したりするなど、ユーザ責任でファイルを自由に操作させることを想定したプログラムです。しかし、Ｆ手順／全銀手順ではデータ転送用コネクションを直接設定し、データの転送を行うだけです。

また、ＦＴＰでは、ファイルが存在しない場合はファイルを自動的に生成し、ファイルが存在する場合は上書きされます。一方、Ｆ手順／全銀手順は定型業務を想定しており、両計算機側にそれぞれ、ファイル転送に必要な情報を設定しておくことによる転送を行います。

2．3　主な外部仕様

（1）データ・タイプ

ＦＴＰで扱うデータ・タイプとして以下の４種類があります。データ・タイプによりＦＴＰで扱うファイルデータの表現形式（データの型、フォーマットなど）が指定され、通常はＡＳＣＩＩとＢＩＮＡＲＹがよく使われます。

① ＡＳＣＩＩ

テキスト形式のファイルで、送信側でローカルファイル形式からＴＥＬＮＥＴのＮＶＴ－ＡＳＣＩＩ形式に変換し、転送します。

② ＥＢＣＤＩＣ

テキスト形式のファイルで、送信側でローカルファイル形式からＥＢＣＤＩＣ形式に変換し、転送します。

③ ＢＩＮＡＲＹ

バイナリデータを変換しないで、そのまま転送します。

④ ＬＯＣＡＬ

バイトサイズの違うマシン間の転送に使用します。

（2）転送データの構造

ＦＴＰで扱う転送データの構造として以下の３種類があります。通常はＦｉｌｅ構造がよく使われます。

① Ｆｉｌｅ構造

単純なバイト列から構成される、ファイル（レコード構造なし）です。

② Ｒｅｃｏｒｄ構造

End-of-Line(デリミタ）で区切られたレコードの列から構成されるファイルです。

③ Ｐａｇｅ構造

ページヘッダとデータから構成されるファイルで、ランダムアクセス可能なファイルです。

（３）転送モード

ＦＴＰで扱う転送モードとしては以下の３種類があります。通常はストリームモードがよく使われます。

① ストリームモード

バイト列として転送する方法で、データにはヘッダなどは付加されません。

② ブロックモード

ブロック単位に区切り、ブロックごとにヘッダを付加して転送する方法です。

③ 圧縮モード

転送時にデータを圧縮して転送する方法です。

3．ＦＴＰに対するリクワイアメント

『2．ＦＴＰの概要』にて、ＦＴＰの主な仕様を示しましたが、ここでは、ＦＴＰに対するリクワイアメントとして、ＥＤＩに利用する際に最低限必要となるＦＴＰコマンドを表３－１に示します。ＦＴＰコマンドにはユーザが利用するためのコマンド（ＰＵＴ，ＧＥＴなど）とプロトコルを制御するコマンド（ユーザには見えません）があります。ここで示しているコマンドは後述のＦＴＰコマンドのことです。

なお、既に市場に流通しているＦＴＰ製品は、これらのコマンドを満足しているので、ユーザにとって特に問題はありません。

表３－１　最低限必要なＦＴＰコマンド一覧

項番	コマンド名	機　　能
1	ＵＳＥＲ	ftp 開始用のユーザ名の指示
2	ＰＡＳＳ	ftp 開始用のパスワードの指示
3	ＱＵＩＴ	ftp 終了指示
4	ＰＯＲＴ	データコネクション用のポート番号連絡
5	ＴＹＰＥ ・type-code:A,I ・form-code:N	データ・タイプの指示 A:ASCII , I:Image(BINARY) N:Non-print
6	ＳＴＲＵ ・sturucture-code:F	ファイル構造の指示 F:File
7	ＭＯＤＥ ・mode-code:S	転送モードの指示 S:Stream（ストリームモード）
8	ＲＥＴＲ	受信要求指示
9	ＳＴＯＲ	送信要求指示
１０	ＮＯＯＰ	Noop（サーバへの応答要求など）

（注）　上表のコマンド一覧中、ＰＡＳＳコマンド以外はＦＴＰにおいて最低限実装が必要な扱いであるため、通常のＦＴＰ製品では実装されていると思われます。また、ＰＡＳＳコマンドについても通常の運用上では必要と考えられるため、一般には、実装されているものと思われます。なお、必要に応じて導入する製品での実装状況を確認して下さい。

4．利用上のガイドライン

本章では、『従来型ＥＤＩ』の運用に通常必要な運用機能の内、表４－１に示す考慮すべき項目のガイドラインを概説します。なお、概説の対象とする運用機能は、Ｆ手順で実現しているサポートレベルを想定しています。また、参考として、『付録Ａ　ＥＤＩ利用の標準機能（Ｆ手順対応：産業情報化推進センター編）とＦＴＰのサポートレベル』を示します。

表４－１　運用機能項目とガイドライン

項番	運用機能項目	ガイドラインの概要
1	データ圧縮	特に必要性がある場合は、外付けの仕掛けで対応する事をガイドとして示します。
2	ゼロ件データ転送	ゼロ件データの転送上の扱いがＦ手順とは異なるため運用時の考慮事項をガイドとして示します。
3	データ転送の強制中断	仕掛かり中の伝送を強制的に中断したいときの注意点を示します。
4	二重交換防止	二重交換防止は、運用で対応する必要があります。その運用例を示します。
5	送受信時のジョブ起動	ファイル送受信後のジョブ起動をローカルで作成する場合の運用例を示します。
6	セキュリティ機能	セキュリティ機能の不足分をサーバ側での運用レベルで考慮する事をガイドとして示します。
7	ファイル成立管理	ファイル成立管理が既存のＦ手順／全銀手順の場合とどう異なるかを明示します。
8	ファイル名と文字種別	ファイル名として使用する文字種別や、ファイル名の長さについての扱いをガイドとして示します。
9	転送データ・タイプ	ＥＤＩ利用時の転送データ・タイプには「ＢＩＮＡＲＹ」を使用する事をガイドとして示します。
１０	ＡＰＰＥＮＤモード	受信ファイルの追加書きを使用した場合の問題点を挙げ、使用しない事をガイドとして示します。

4．1　データ圧縮

ＦＴＰの規約上、転送モードの一つとして圧縮モードが仕様化されていますが、実際に圧縮モードをサポートしているＦＴＰは、ほとんど存在しません。

このため、データ圧縮が必要な場合は、外付けの圧縮ツールを用いて、転送の前後に転送ファイルの圧縮・伸張を行うようにします。圧縮・伸張に関する手順の例を、図４－１に示します。

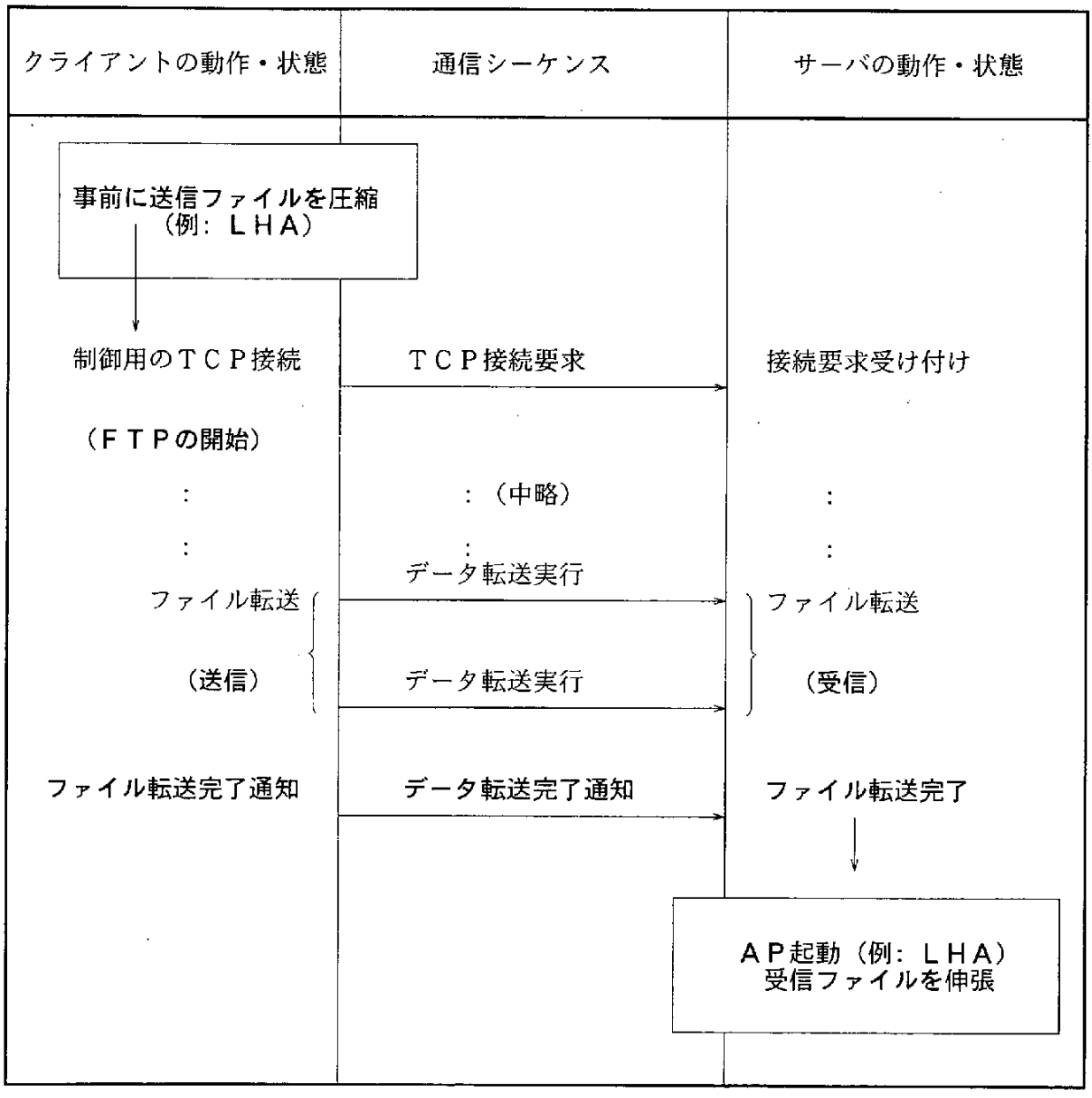


図４－１　外付け圧縮ツールによる圧縮・伸張の手順例

同一の圧縮形式を持つ圧縮ツールにおいて、伸張側ツールの圧縮形式のバージョンは、圧縮側ツールの圧縮形式のバージョンと同一か、上位でなければなりません。

以下に外付けの圧縮ツール例を示し、表４－２に使用機種との関係を示します。

① ＬＨＡ

圧縮機能と伸張機能の両方を持つフリーソフトウェアです。

② ＺＩＰ、ＧＺＩＰ

圧縮機能と伸張機能の両方を持つフリーソフトウェアまたはシェアウェアです。

ＺＩＰは圧縮ツール固有の名称ですが、各種存在するＺＩＰ形式の圧縮ツールの一般的な呼称でもあります。例示のＺＩＰおよびＧＺＩＰはこのうちの代表的なツール名です。

③ ＣＯＭＰＲＥＳＳ／ＵＮＣＯＭＰＲＥＳＳ

圧縮機能と伸張機能が独立したＵＮＩＸの標準ツールであり、一般的にコマンドとして提供されます。

表４－２　外付け圧縮ツール例

伸張側 ＼ 圧縮側		機種		
		パソコン	ＵＮＩＸ	メインフレーム
機種	パソコン	・ＬＨＡ ・ＺＩＰ	・ＬＨＡ	—
	ＵＮＩＸ	・ＬＨＡ	・ＣＯＭＰＲＥＳＳ ／ＵＮＣＯＭＰＲＥＳＳ ・ＬＨＡ ・ＧＺＩＰ	—
	メインフレーム	—	—	（注）

（注）機種を限定した圧縮ツールが若干あり。

4．2　ゼロ件データ転送

ＥＤＩ運用では、通常、指定された日時に定期的にデータ転送を行うことが一般的ですが、その際、転送するデータがない場合などもあります。このため、Ｆ手順では、ゼロ件データ転送を正常転送と扱うか、異常転送と扱うかをユーザ側で選択できる機能をサポートしています。

これは、転送するデータがない場合のゼロ件（正常転送の扱い）か、転送するデータの準備が遅れてゼロ件（異常転送の扱い）か、を切り分ける事ができるようにするものです。

一方、ＦＴＰでは、この機能はありません。即ち、ＦＴＰでは、ゼロ件転送は、常に正常転送と扱われますので、この事を前提とした運用を行う必要があります。

4．3　データ転送の強制中断

大きなファイルを間違って送信し始めた場合など、途中で中断したいことがあります。

ＦＴＰ専用ではありませんが、プログラムを中断するキー（例えば、 Ctrl+C や Attn キーなど。システムによって異なります。）で、多くの場合転送を中断できます。ただし、ＴＥＬＮＥＴ経由でＦＴＰを操作している時など、中断できない場合もありますので、大きなファイルを手作業でＷＡＮ経由で転送する時は、注意が必要です。

また、Ｆ手順では転送が中断された時には受信側にファイルが残ることはありませんが、ＦＴＰでは多くの場合中断するまでに転送されたデータをファイルに残しますので、この方法で中断した場合、ファイルを無効にして処理する必要があります。

いずれにしても、転送時間が短く中断が間に合わない場合もあり、運用には、間違ったファイルが転送された場合の訂正処置を織り込む必要があります。特にファイル転送後の処理が自動化されている場合は、転送前の状態に回復する手順など考慮が必要です。

4．4　二重交換防止

ＦＴＰの仕様上、二重交換となるファイル転送の結果は、常に正常扱い（上書き）になります。

このため、二重交換防止を図る必要がある場合には、接続相互間の取り決めにより、このことを前提とした運用を行う必要があります。

なお、外付けの二重交換防止機能を利用者が作成することで対処する方法もあります。

ただし、外付けの二重交換防止機能を作成する場合、ＦＴＰによる実際のファイル転送タイミングと、転送ファイルのデータ処理（格納・取り出し）のタイミングに不整合が発生しないように制御することは、以下の枠内に例示するような問題点があるため一般的に困難です。例示した問題点を解決しようとする場合、外付けの二重交換防止機能の機能性（完全性）は、サーバ側の外付け二重交換防止機能における管理精度に依存します。

以上のことから、外付けの二重交換防止機能を作成する場合でも、不完全な管理部分に関しては、運用による回避処置を考慮しておく必要があります。

＜外付け二重交換防止機能に関する問題点＞

クライアント側の外付け二重交換防止機能を作成する場合、クライアント側の送信ファイルに関する転送状況はすべて自己側で管理することが可能であるため、ＦＴＰによる実際の転送を行う前に転送状況を確認することにより、同一ファイルの二重交換を防止することが可能なように見えます。

しかし、クライアント側の二重交換防止機能が、自己側の管理だけから、自己側で新規に作成したデータの送信ファイルの転送要求を二重交換ではないと判断しても、サーバ側ＦＴＰにおいて前回に受信した当該転送ファイルのデータを、サーバ側のデータ処理（ＡＰ）が取り出し済みか否かは、クライアント側では判断することが不可能です。

4．5　送受信時のジョブ起動

本機能は、Ｆ手順において「受信時のジョブ起動」、「転送終了後のファイル処理連動（送信時のジョブ起動）」として規定されていますが、ＦＴＰでは、ファイル転送と連携したジョブ起動の機能をサポートしていません。

このため、ジョブ起動が必要な場合は、以下に示す対処で実現を図ることができます。

（対処１）既存機能の組み合わせによるジョブ起動機能の作成

使用するＯＳなどが持っている既存の機能を利用して、運用に応じたジョブ起動機能をユーザが作成します。

既存機能の組み合わせによるジョブ起動機能の運用例を表４－３に示します。また、転送ファイルに対応させて起動するＡＰの管理方法の運用例を表４－４に示します。

なお、同一のファイルに対する次回の転送は、当該ファイルに対応させて起動するＡＰの実行が終了するまで防止する処置（二重交換防止）などを考慮する必要があります。

（対処２）サポートパッケージの利用

ジョブ起動機能をサポートしているファイル転送パッケージを選定して利用します。

ただし、各パッケージがサポートするジョブ起動機能のサービス内容（ジョブの起動タイミング、起動ジョブの管理方法、実装方法など）は、全て各パッケージ固有のものです。

したがって、導入予定パッケージの事前確認が必要です。

表４－３　既存機能の組み合わせによるジョブ起動作成の運用例

例１．転送イベントの利用によるジョブ起動

ＦＴＰが出力する転送イベント（転送終了のメッセージなど）を動的に取得することにより、新たに転送されたファイルを検知して、当該ファイルに対応させたＡＰを起動します。

＜注意点＞

・転送イベントを動的に取得するために、ＦＴＰの運用中はジョブ起動機能を常駐させておく手段の考慮が必要です。

例２．タイムスタンプの利用によるジョブ起動

運用上の適当なインターバルで、転送ファイルの存在するディレクトリ内のタイムスタンプを取得して比較することにより、新たに転送されたファイルを検知して、当該ファイルに対応させたＡＰを起動します。

＜注意点＞

・通常のファイルシステムのタイムスタンプは、分単位です。

・設定したインターバルの間に、同一のファイルに対して複数回の転送が行われていても検知することができません。

・タイムスタンプだけでは、転送結果（正常終了／異常終了など）を判断することはできません。このため、転送結果により起動ＡＰを振り分けたりする場合は、運用での対処を考慮する必要があります。

例３．ログ情報の利用によるジョブ起動

ログ機能を持つＦＴＰの場合に実現可能な方法です。運用上の適当なインターバルで、ログ情報を参照することにより、新たに転送されたファイルを検知して、当該ファイルに対応させたＡＰを起動します。

＜注意点＞

・設定したインターバルの間に、同一のファイルに対して複数回の転送が行われていても参照したログ情報の転送ごとにＡＰを起動することはできません。

・ログ情報は、ＦＴＰの運用中でも動的に参照できることが必要です。

表４－４　転送ファイルに対する起動ＡＰの管理方法の運用例

例１．定義ファイルへ事前登録する方式

キーアクセス可能な定義ファイルに、転送ファイル名（パス名）をキーとした定義レコードを生成し、ＡＰの起動パス名と、起動条件を設定する起動パラメタを登録しておく方式です。

定義ファイル名をＦＴＰの起動パラメタで指定したり、定義ファイルを専用のディレクトリに置くようにします。この方式では、任意のディレクトリに存在するＡＰを、複数の転送ファイルのための起動対象として共有利用することができます。

＜注意点＞

・ＡＰの格納先ディレクトリを変更した場合、同時に定義ファイル情報の変更が必要となります。

例２．起動ＡＰ専用のディレクトリへ事前登録する方式

専用のディレクトリに、転送ファイルと同一ファイル名のＡＰを登録する方式です。

使用するファイルシステムに応じて、ＡＰのファイル名には起動可能なファイル種別のサフィックス（ＥＸＥ、ＣＯＭ、ＢＡＴなど）を付加する必要があります。

＜注意点＞

・起動条件（正常終了・異常終了時など）ごとの起動はできません。（困難です。）

・転送ファイルごとに起動ＡＰを登録するため、同一の起動ＡＰを共有利用できません。

・転送ファイル名はシステム内でユニークでなければなりません。

例３．共有ディレクトリへ事前登録する方式

転送ファイルと同一のディレクトリに、転送ファイル名から変換可能なファイル名を持つＡＰを登録する方式です。

使用するファイルシステムに応じて、ＡＰのファイル名には起動可能なファイル種別のサフィックス（ＥＸＥ、ＣＯＭ、ＢＡＴなど）を付加する必要があります。

＜注意点＞

・起動条件（正常終了・異常終了時など）ごとの起動ができません。（困難です。）

・転送ファイルごとに起動ＡＰを登録するため、同一の起動ＡＰを共有利用できません。

4．6　セキュリティ機能

（1）利用者の認証

ＥＤＩでは、ファイルを送信・受信する場合は、事前に許可を得なければなりません。ＦＴＰのサーバに接続するためには、事前にユーザＩＤとパスワードを登録し、ログイン時に入力しなければならないので、利用者を限定するという認証機能は満たされています。

（2）ファイルの保護

Ｆ手順は、例えばシステムＡがファイルのデータを送信しシステムＢがその受信したデータでファイルを作成することを想定しています。一方ＦＴＰは、クライアントがサーバにログインした後は、クライアント側からサーバ上のファイルシステムを（事前に許可された範囲内で）自由に操作して、ファイルをクライアントとサーバ間でコピーすることを想定しています。つまり、ユーザはサーバ側のディレクトリを操作（読み出し／作成／消去など）したり、いろいろなファイルへアクセス（書き込み／読み出し／作成／削除など）したり、システム管理者が実行するレベルの操作ができる状態です。しかし、ＥＤＩとして利用する際、自由な操作などができることは好ましくありません。ＦＴＰのサーバ側に、ディレクトリの操作を含めて、以下のとおり利用者単位で操作範囲を設定する必要があります。

①　ディレクトリ操作禁止

対象となる該当ディレクトリに対してのみ、読み出しを許可します。

②　ファイルアクセスの制限

操作が許可されているディレクトリ配下の、必要なファイルに対してのみ、読み出し、書き込み、またはその両方をを許可します。

ＦＴＰのプロトコル自体にはこの設定機能は定義されていませんが、複数の利用者を想定したメインフレーム、ＵＮＩＸ、ＰＣのファイル・サーバなどのシステムでは、通常ファイルシステムがこの機能を持っています。個人を対象としたＰＣをＦＴＰのサーバ側に使用する時は、そのシステムがこの機能を持っているか確認する必要があります。

以下に、アクセス制御があるＵＮＩＸ系のファイルシステムでの設定例を示します。なお、設定例では、図４－２に示すとおり、Ａ社用の File-Waと File-RaをDirectory-Ya下に、Ｂ社用の File-Rbを Directory-Yb 下に置き、表４－５に示すアクセス制御の設定を想定します。

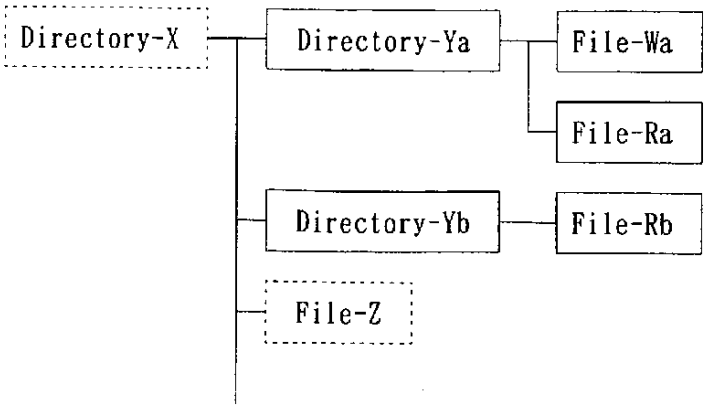


図４－２　ディレクトリ構造の設定例

表４－５　アクセス制御表の設定例

	A社ユーザID		B社ユーザID	
	Read	Write	Read	Write
Directory-X	-	-	-	-
Directory-Ya	Y	-	-	-
Directory-Yb	-	-	Y	-
File-Wa	Y 注	Y	-	-
File-Ra	Y	-	-	-
File-Rb	-	-	Y	-
File-Z	-	-	-	-

Y :許可
Read :ユーザ側受信
Write : ユーザ側送信

注：File-Waは送信用ですが、この例では確認のため読み返しができるようにReadも設定しています。

上記の設定例において、A社は、File-Waの送信(Write) と File-Raの受信(Read)はできますが、他のファイルにはアクセスできません。また、 Directory-Xにはアクセスが許可されていないので、Directory-Yaから外へ移動できず、他のファル情報を知ることもできません。さらに、Directory-YaにWrite が設定されていないので、この下に新しいファイルを登録することもできませんし、ディレクトリの作成・変更・消去もできません。

同様に、B社は File-Rbの受信ができるだけです。なお、送信・受信対象外のファイルは、File-Zの様に他のディレクトリ下に置きます。

4．7　ファイル成立管理

Ｆ手順ではファイルクローズ終了のＦ－ＣＬＯＳＥプリミティブが正常終了した時点でファイ転送の成立とみなされ（図４－３参照）、全銀手順では送受信された全ての通信制御電文、ファイル制御電文が正常に処理されて初めてファイル転送の成立とみなされます。（図４－４参照）。

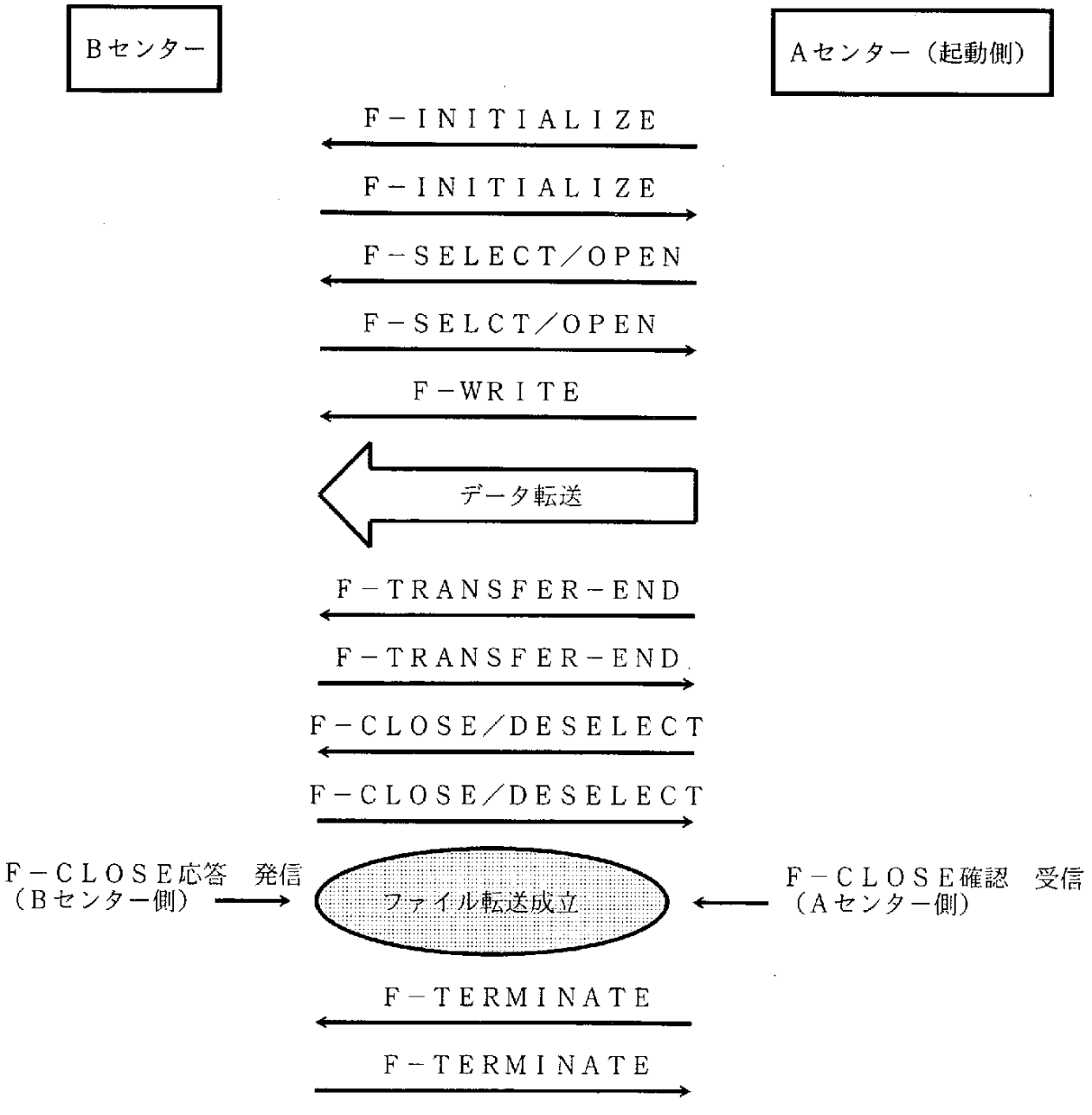


図４－３　Ｆ手順を利用する際のファイル成立管理

一方、ＦＴＰは制御用ＴＣＰコネクションとデータ転送用ＴＣＰコネクションの２本のコネクションを利用しながらファイル転送を行います。ファイル転送の成立はデータ転送用ＴＣＰコネクションが正常に切断された時点となります（図４－５参照）。

以上のことから、ＦＴＰにおいてもＦ手順や全銀手順と同レベルの信頼性は確保されています。

なお、障害時のファイルの破棄規定に関しては、Ｆ手順／全銀手順に相当する規定が定義されておらず、障害発生時までに転送されたデータはファイルとして残存するため、双方であらかじめ、障害時の運用手順について決めておく必要が有ります。

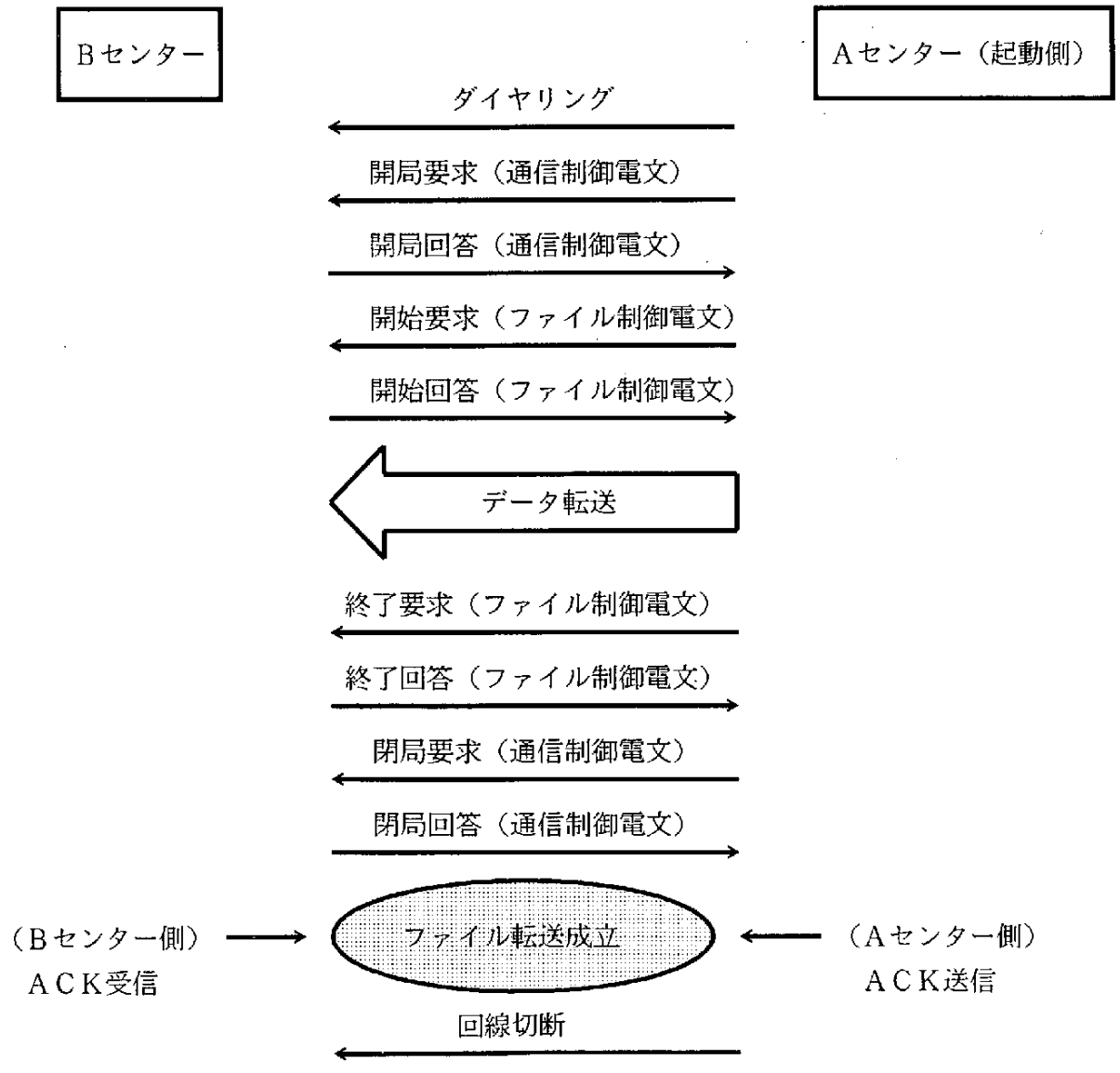


図４－４　全銀手順でのファイル成立管理

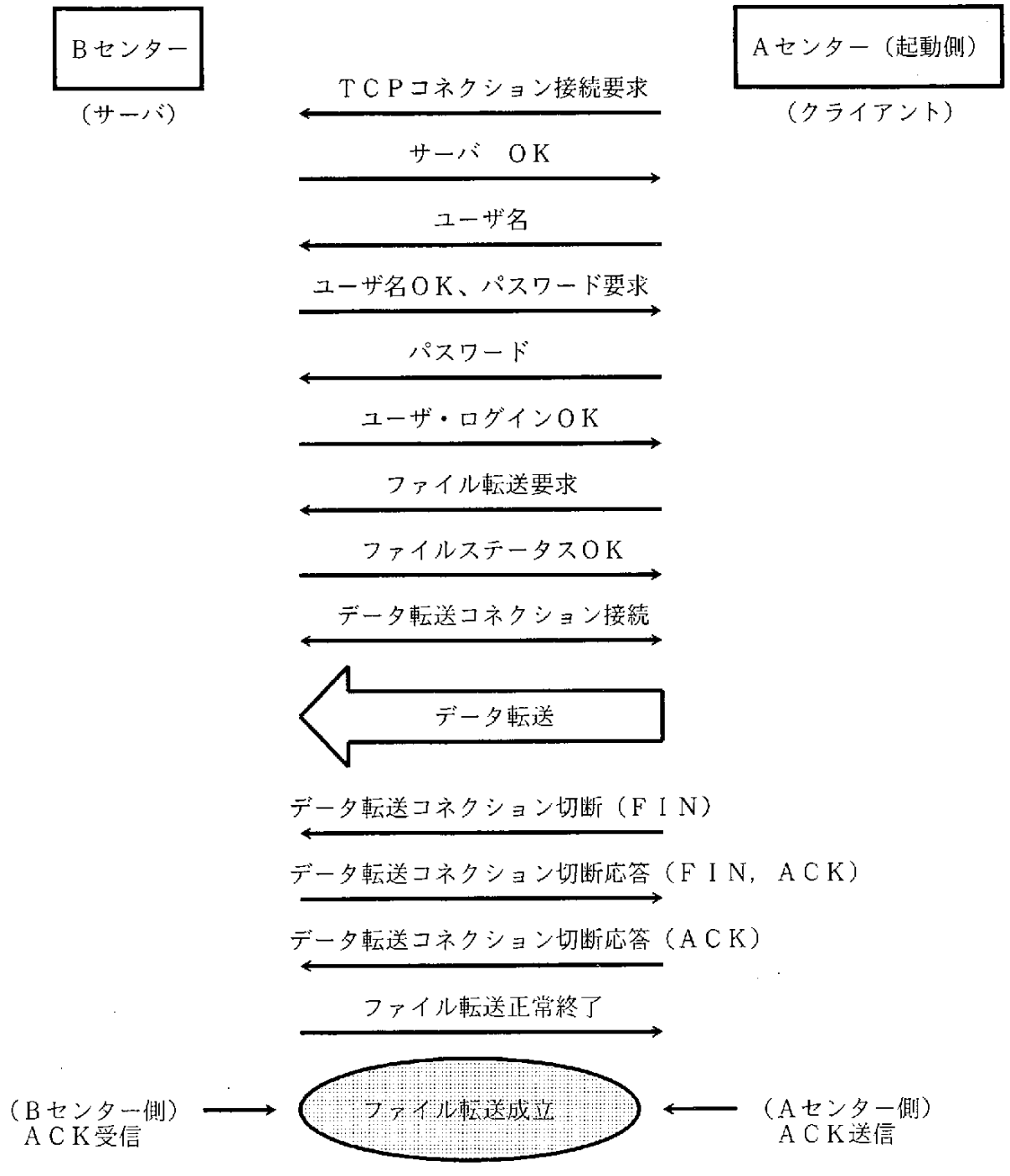


図４－５　ＦＴＰを利用する際のファイル成立管理

4．8　ファイル名と文字種別

ＦＴＰにおけるファイル名は、Ｆ手順の仮想ファイル方式と異なり、実際のファイル名を指定します。

ＦＴＰでは、当事者間でデータ交換時のファイル名を取り決める必要があります。以下に一般的な留意事項について説明します。

（1）ファイル名の文字種別

一般的なＥＤＩにおいて、英大文字（Ａ～Ｚ；ただしスペースは除く）、数字（0～9）は利用できます。その他の日本語、特殊文字（¥，＃，@等）などは個々に確認して下さい。

（2）ファイル名の長さ

互いの計算機システムにおけるファイル名の長さにも制約が生じる場合があるので注意が必要です。一般的には、各々のシステムで許可しているファイル名の長さまで利用可能です。

4．9　転送データ・タイプ

ＥＤＩデータを転送する際の転送データ・タイプ（ファイル転送時のデータ・タイプ）は、『3．ＦＴＰに対するリクワイアメント』でも記述しているとおり、「ＡＳＣＩＩ」か、「ＢＩＮＡＲＹ」のいづれかのデータ・タイプを指定する必要があります。

ただし、ＥＤＩ利用時の転送データには以下の特色があります。

① ＥＤＩ標準により使用できる文字コードが決められているため、お互いのシステム間でコード変換の必要がないこと。

② 転送データはテキストデータとバイナリデータで構成されているケースが多く、どの箇所にバイナリデータが含まれているかＦＴＰ側では判断できないこと。

これらから判断し、どのような文字コードも透過的に転送できる「ＢＩＮＡＲＹ」を指定することを推奨します。「ＢＩＮＡＲＹ」を指定した場合、ビット・パターンがそのまま相手システムに転送されるため、文字コードと通信システムの問題を切り離して考えることができます。

よって、ＥＤＩ利用時は、そのＥＤＩ標準で取り決められている文字コードを使用し、転送データ・タイプに「ＢＩＮＡＲＹ」を指定してデータ交換を行って下さい。

4．10　APPENDモード

ファイル転送方式のＥＤＩでは、前回交換したファイルが相手方で処理されないうちに次のファイルを交換すると、前回送ったデータの上に新しいデータが上書きされてしまい、結果として前回交換したファイルが失われてしまいます（もし注文情報であれば、前回の注文はキャンセルした結果になります。）。そこで、二重交換防止機能を用いて前回送信ファイルが未処理のうちは、次のファイルを交換できないようにしたり、サイクル管理を用いて上書きを防止しつつ頻繁な交換を可能にします。

しかし、転送間隔が短いか、あるいは転送時刻が不定期な場合、頻繁に二重交換防止機能が働き交換エラーが頻発したり、それを防止するためにサイクル管理を導入すると運用が複雑になる等の問題があります。このような時、ＡＰＰＥＮＤモード（追加モード）を活用すると、以下の利点があり、特にＶＡＮにおける交換サービス（例：ＣＩＩ－ＥＤＩサービス）で、ユーザからＶＡＮセンタへの送信時によく利用されています。

- 追加モードでは、本質的にファイル内に既に存在するデータの上に上書きすることがなく、したがって、短い転送間隔でも不定期な交換に対応可能。
- 二重交換防止機能もサイクル管理も使う必要がないこと。

ＲＦＣ９５９では追加モードのプロトコルが定義されており、多くのＦＴＰ製品でサポートされており、便利に使えそうです。しかし、このモードを使うと以下の問題点があります。

- 障害により途中で転送が中断すると、その時点まで転送されたデータが追加されてしまい、その後のリカバリが非常に困難。

そこで、追加モードは使用しないことを推奨します。しかし、ＥＤＩサービスにおけるＶＡＮセンターとＥＤＩユーザーとのデータ交換には、前述したように、ファイル追加書込機能が必要なので、特別なＦＴＰをＶＡＮ側で作成し実装することを推奨します。この実装のポイントについては、『５．１　ＶＡＮセンターの対応』を参照して下さい。

5．　実装ガイド

5．1　ＶＡＮセンターの対応

（1）ＶＡＮセンター

ここでいうＶＡＮセンターとは、ＥＤＩサービスをサポートしているＶＡＮ事業者のセンターとします。ＥＤＩサービスでは、多数の企業（ＥＤＩユーザ）とＶＡＮセンターとをスター状に結び、ＥＤＩユーザ間のデータ交換（Interchange)を、ＶＡＮセンター経由でサポートしており、ＥＤＩユーザに対して、交換回線と同じような機能が提供されています。

多くのＥＤＩサービスでは、ファイル転送方式を活用しており、バッチ型交換になっていますが、そうではないＥＤＩサービスもあります。以下では、ファイル転送をベースとしたＥＤＩサービスに関する事項について述べます。

（2）ＥＤＩユーザとＶＡＮセンターとの接続

ＥＤＩユーザとＶＡＮセンターとは、ファイル転送手順を用いて接続します。ただし、より厳密には、ＥＤＩユーザとＶＡＮセンターのアクセスポイント間の接続というべきかもしれません。

この接続では、第４．１０項で述べたように、ＡＰＰＥＮＤモード（追加モード）がよく使われますが、ＦＴＰの追加モードをそのまま使うことができません。そこで、ＶＡＮ側では特殊なＦＴＰを開発し使う必要があります。

（3）特殊ＦＴＰのポイント（ＲＦＣ９５９の解釈）

このＦＴＰは、サーバ側専用であり、一部のプロトコル仕様の解釈は、ＲＦＣ９５９に準拠しません。準拠しない部分を以下に示します。

① このＦＴＰは、サーバ専用となります。

② ファイルは読出専用と書き込み専用に分離されます。すなわち、読出も書き込みも可能なファイルは取扱いません。

読出専用ファイルには、転送完了を示すフラグを設けます。

③ クライアント側（ＥＤＩユーザ側に相当）が読出モードで起動し、書き込み専用ファイルをアクセスした時は、無条件にエラーにします。

④ クライアント側が読出モードで起動し、サーバ内に存在する読出専用ファイルにアクセスした時、パスワード等のセキュリティチェックの結果有効なアクセスであ

れば、通常の転送を行い、転送が成功すれば、転送完了フラグを転送済にします。

⑤　クライアント側が書き込みモードで起動し、読出専用ファイルをアクセスした時は、無条件にエラーにします。

⑥　クライアント側が書き込みモードで起動し、サーバ内に存在しないファイルにアクセスしようとした時は、無条件にエラーにします。

⑦　クライアント側が書き込みモードで起動し、サーバ内に存在する書き込み専用ファイルにアクセスした時は、クライアント側での指定を無視して、常に追加書き込みとします。

（4）実装上のポイント

この特殊ＦＴＰは、クライアント側の通常仕様のＦＴＰと完全につながらなければなりません。クライアント側が通常仕様のＦＴＰを使えることで、安価にＥＤＩを構築できることになるからです。

すなわち、ＲＦＣ９５９のプロトコル仕様のフォーマットには完全に準拠しなければなりませんが、解釈が一部異なります。例えば、クライアント側では、通常書き込みをしているつもりでも、サーバ側では追加書き込みしているようにです。さらに、転送が正常終了しない場合は、転送そのものがなかった状態となるよう注意する必要もあります。

読出専用ファイルに設けた転送完了フラグは、ＥＤＩサービス仕様の作り方によって使用方法が異なります。

例えば、このフラグを二重交換防止用に使うことができます。ただし、ＥＤＩユーザが読出データを失った場合に備えて、強制二重交換可能なように、フラグリセット機能を設ける必要があります。

なお、ＶＡＮセンターでは、読出専用の転送完了フラグの転送済状態を確認してから、次にＥＤＩユーザに送るべきデータをファイルにセットし、フラグを未転送状態にします。

書き込み専用ファイルには、特にフラグを設ける必要はありません。

5．2　障害／ステータス通知

ＦＴＰを利用したデータ交換における障害通知は、一般的に「ＦＴＰ規約（ＲＦＣ９５９）」で十分に行うことができます。

ここでは、Ｆ手順／全銀手順相当の障害通知及びステータス通知を行いたい場合について説

明します。

ＦＴＰは、クライアント／サーバ型の通信システムのため、Ｆ手順／全銀手順相当の通信システムに比べて、以下に示す処理を行う場合、ユーザ側での考慮が必要となります。

・障害通知情報として、業務に依存した情報を通知する場合

・障害通知情報として、起動側異常を応答側に通知する場合

・ステータス情報の通知を行いたい場合

ここで示す方法は、「起動側」、「応答側」共にサーバ／クライアン機能を有しているシステムに限ります。以下に、その実現方法などについて記します。

障害／ステータス通知を実現するために、「応答側でのファイル転送結果」と「起動側でのファイル転送結果」を「送達確認電文（障害通知電文）」として相互に交換します。図５－１の実装例では、「送達確認電文（障害通知電文）」をファイルデータとして転送しています。

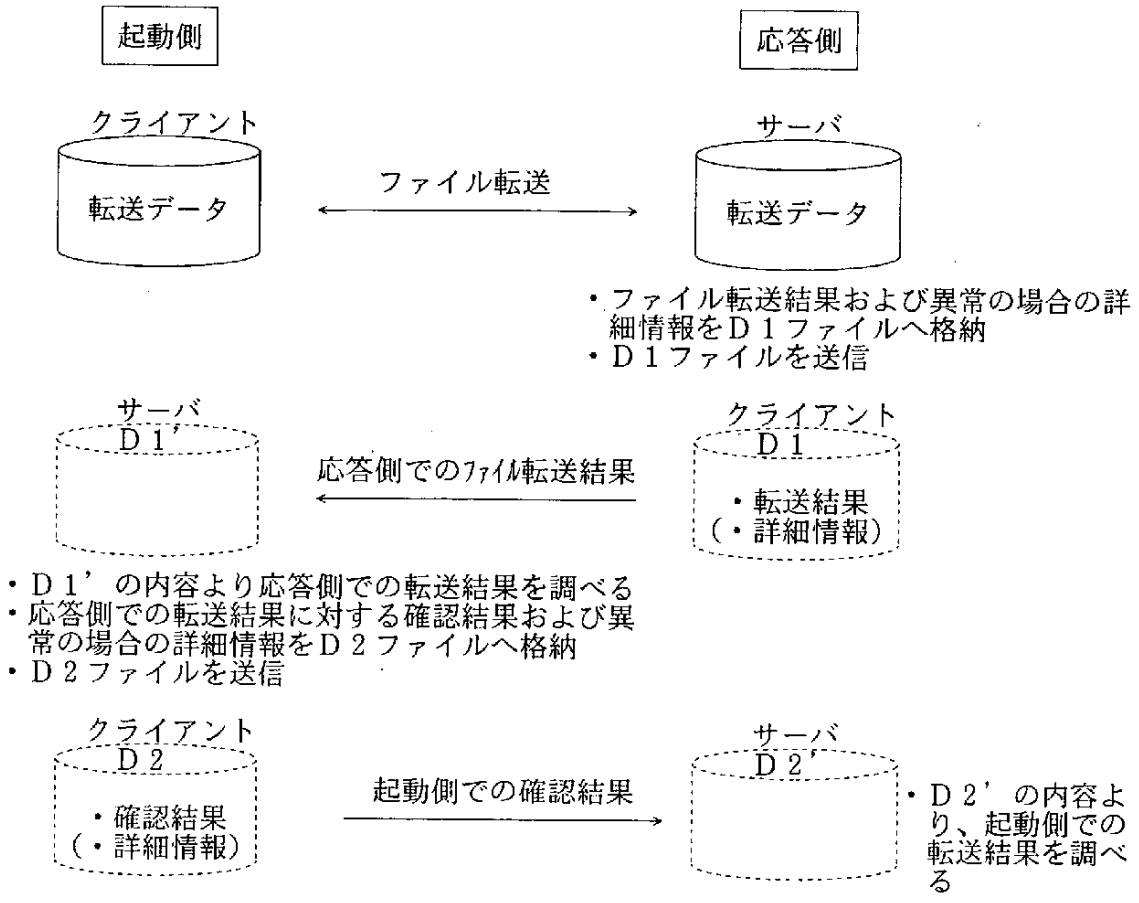


図５－１　障害／ステータス通知の実装例

また、図５－１の実装例を使用しステータス通知を行う一例を図５－２に示します。この図は、ファイル転送結果を格納し、それをファイルデータとして起動側に転送することにより、ステータス通知を行っている例です。

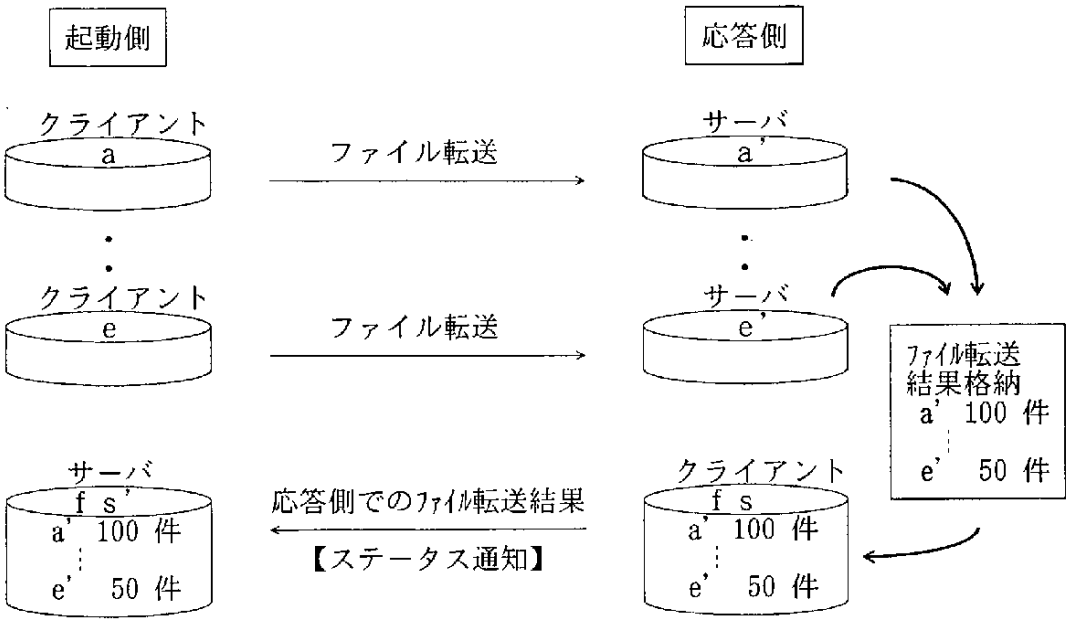


図５－２　ステータス通知の例

6．ＦＴＰの利用におけるファイアウォール環境

企業間をＴＣＰ／ＩＰで接続することで、より柔軟なシステムが実現できる一方、それは逆に、外部ネットワーク上の数多くのユーザから、内部ネットワークへアクセスされる可能性があります。

このような場合、危険にさらされる可能性のある計算機やサービスを制限するなど、ある程度の利便性を犠牲にして、内部ネットワーク全体のセキュリティを確保させるための手段としてファイアウォールが利用されます。

ファイアウォールとして通信に対し制限を加える方法には、フィルタリング、ＩＰパケット転送機能の停止、経路制御があります。その中でもフィルタリングはルータにも組み込まれていることが多く、安価にファイアウォールを構築することができるため、もっともポピュラーな方法として利用されています。フィルタリングは、ＴＣＰのサービスポートやＩＰアドレスなどをパラメータにしてＩＰパケットの通過を許可したり、禁止したりする方法です。

ファイアウォール構築にあたっては、内部ネットワーク全体のセキュリティとして、どこまで制限を加えればよいか、あらかじめ見極めておく必要があります。例えば、電子メール（ＳＭＴＰ）については許可し、ファイル転送（ＦＴＰ）については禁止するのが一般的です。

しかしながら、ＦＴＰを単に禁止してしまうと、この上に実現しているＥＤＩは利用できませんので、特定の相手計算機に限りＦＴＰを許可し、閉域接続グループを実現するようなファイアウォールの構築を考える必要があります。

オープン（ＴＣＰ／ＩＰ）環境で従来型ＥＤＩを運用していくには、セキュリティの面からも、ファイアウォールを利用した閉域接続環境を構築することが望ましいと言えます。

以下に、ＥＤＩを利用する場合のファイアウォール構築のための留意点について説明します。

（１）専用線経由ファイアウォールの場合

ＦＴＰを使用してＥＤＩを構築するためには、データ交換を行う企業のネットワーク（または、特定の計算機）との間に限り、セキュリティ確保のため特定のＦＴＰパケットの通過を許可する設定を事前に行う必要があります。

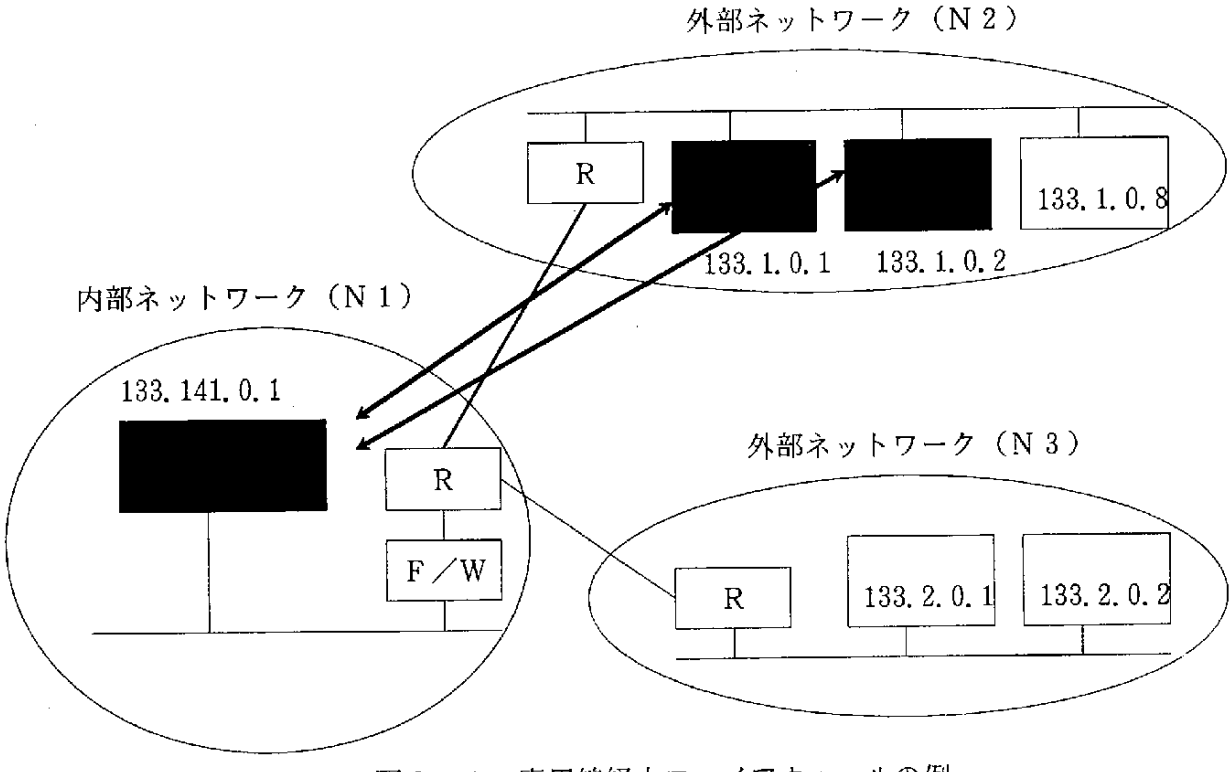


図６．１　専用線経由ファイアウォールの例

例えば、図６．１に示すとおり、内部ネットワークＮ１にファイアウォールを設置し、Ｎ１内の計算機133.141.0.1 と外部ネットワークＮ２内の計算機133.1.0.1 または133.1.0.2 との間のみでＦＴＰ（ＥＤＩ）の利用を許可する場合は、フィルタ情報としてＦＴＰのポート番号と通信を行う計算機のアドレスの設定を行います。

内部ﾎｽﾄ(IPｱﾄﾞﾚｽ)	ポート番号	外部ﾎｽﾄ(IPｱﾄﾞﾚｽ)	動作モード
133.141.0.1	20,21(FTP)	133.1.0.1	permit
133.141.0.1	20,21(FTP)	133.1.0.2	permit

この方法では、ＥＤＩを利用する計算機が増えるたびに、追加設定が必要となり、管理が複雑などのデメリットもありますが、事前に接続相手を特定しておき、登録、管理するということでは、従来型ＥＤＩの考え方に沿っています。

（２）ＷＡＮ経由ファイアウォールの場合

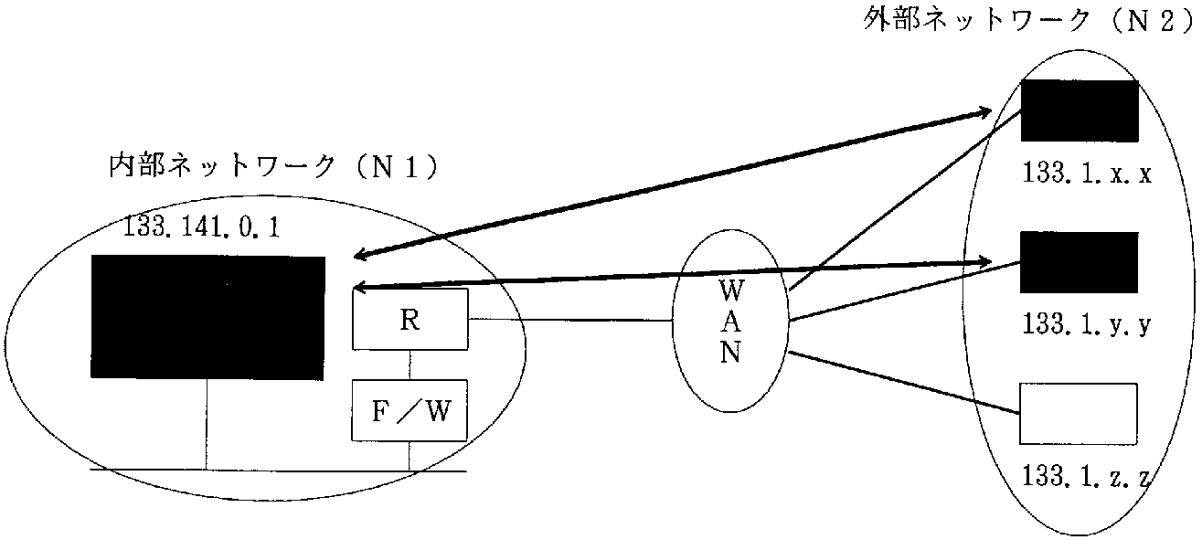


図６．２　ＷＡＮ経由ファイアウォールの例

図６．２のように、ＦＴＰの相手計算機がＷＡＮ接続されている場合は、通常、接続時にルータが動的に端末のアドレスを付与するため、前述のように、特定の計算機に対して、事前にフィルタ情報を設定することはできません。例えば、以下のように設定することで、133.1 の外部ネットワークに存在する全ての端末とのＦＴＰ利用に限り許可するという設定は可能ですが、 133.1.z.zの端末を遮断することはできません。

内部ﾎｽﾄ(IPｱﾄﾞﾚｽ)	ポート番号	外部ﾎｽﾄ(IPｱﾄﾞﾚｽ)	動作モード
133.141.0.1	20,21(FTP)	133.1	permit

このような場合は、ファイアウォールでのフィルタ設定と端末、ルータ間の接続時のセキュリティ機能（例えばＰＰＰのＣＨＡＰ機能）を併用することで代替が可能です。

（３）ＥＤＩ用に独立した計算機がある場合

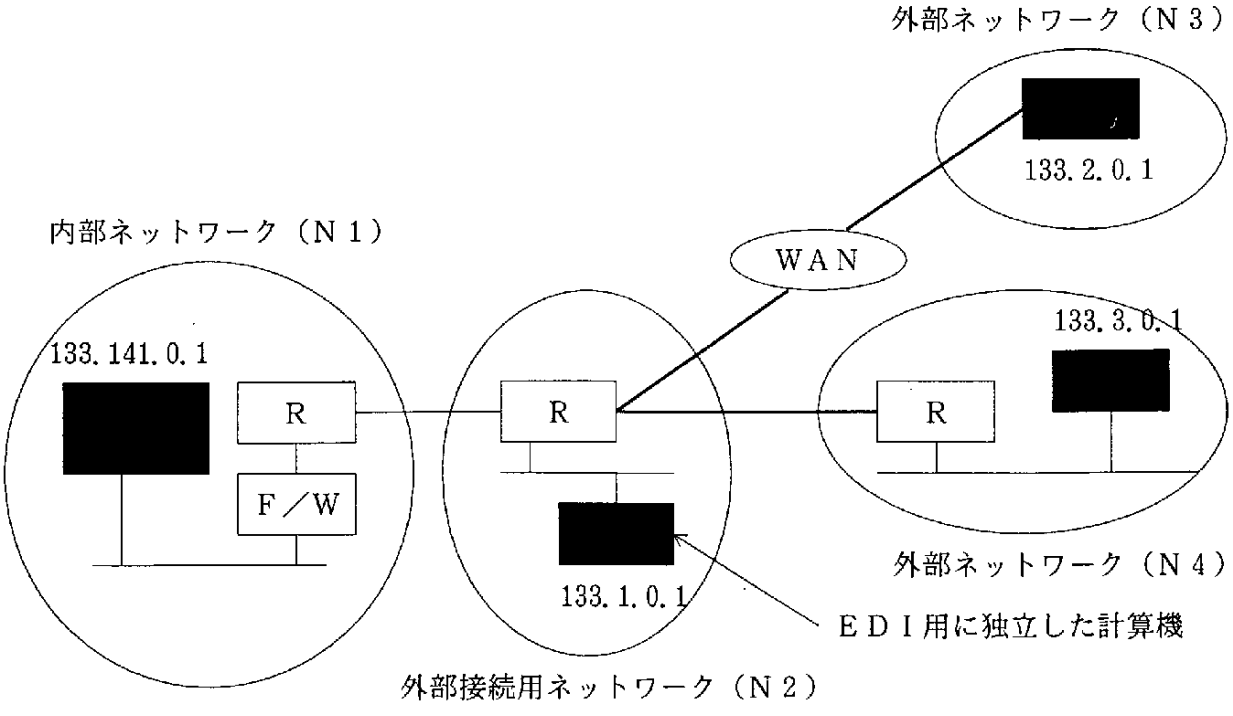


図６．３外部接続用ネットワークを利用した例

図６．３のように内部ネットワーク（Ｎ１）からＥＤＩ接続するための計算機を外出しにすることにより、外部ネットワーク（Ｎ３，Ｎ４）から内部ネットワーク（Ｎ１）に対するＦＴＰの利用を禁止させることができます。この場合、Ｎ１内の計算機133.141.0.1 とＮ２内の計算機133.1.0.1 とのアクセスのみを許可するようファイアウォールを設定することにより、ＥＤＩ用に独立した計算機を経由して内部ネットワーク（Ｎ１）と外部ネットワーク（Ｎ３，Ｎ４）の間でデータ交換が行われることとなります。

付録Ａ　ＥＤＩ利用時の標準機能（Ｆ手順対応：産業情報化推進センター編）とＦＴＰのサポートレベル

＜凡例＞

サポートレベル；　○：標準で可　△：外付けで可　×：不可

Ｆ手順機能名	ＥＤＩでの必要性			ＦＴＰでのサポートレベル	全銀手順でのサポート状況
	大	中	小		
データ圧縮		○		△	○
マルチファイル転送		○		○	○
代表名によるファイル読み出し			○	×	－
ゼロ件データ転送	○			× 注２	－
データ転送の強制中断	○			○ 注１	－
サイクル管理			○	×	○
二重交換防止		○		×	○
転送許可時間			○	×	○
送信・受信ファイル状況確認			○	△	－
転送状態問い合わせ			○	△	－
処理履歴管理		○		○	－
送信時の自動取上げ		○		△	－
受信時のジョブ起動	○			△	－
転送終了後のファイル処理連動　注３	○			△	－
端末からの制御			○	○	－
プライオリティ制御			○	×	－
セキュリティ機能	○			○ 注１	○
ファイル成立管理	○			○ 注１	○
障害検知／再送機能		○		○	○
障害状態問い合わせ			○	△	－

注１：Ｆ手順／全銀手順と差異あり。

注２：運用で対応可能。詳細は『４．２　ゼロ件データ転送』を参照。

注３：「送信時のジョブ起動」と同等な機能。

付録B　ＲＦＣ９５９規約書（抜粋）

ＦＴＰの仕様はＲＦＣ９５９規約書（注１）として発行されています。本章では、本編と関連深い部分をＲＦＣ９５９規約書から抜粋し、当センターにて独自に和訳したものを以下に示します。

（注１）正式なドキュメント名称は以下のとおり。

Postel, Jon, and Joyce Reynolds, "FILE TRANSFER PROTOCOL(FTP)", RFC959, ISI, October 1985.

１．序文

ＦＴＰの目的は、１）ファイル（コンピュータ・プログラム及びまたはデータ）の共用を推進すること、２）遠隔コンピュータの（プログラムによる）間接的または暗黙的の使用を推進すること、３）ホスト間のファイルシステムの違いからユーザを保護すること、４）信頼できる方法で効率的にデータを転送すること、である。ＦＴＰは、ユーザが端末から直接使用することができるが、主としてプログラムによる使用を目的として設計されている。

本仕様における試みは、大規模ホスト（maxi-hosts）、ミニホスト（mini-hosts）、パーソナルワークステーション、ＴＡＣ(Terminal Access Controller)の多様なニーズを、単純で、かつ容易に実現できるプロトコル設計とすることである。

本書は、ＴＣＰ（Transmission Control Protocol）［２］、Ｔｅｌｎｅｔプロトコル［３］に関する知識をベースに記述している。なお、これらの文書は、ＡＲＰＡ－Internetプロトコル・ハンドブック［１］に記載されている。

２．概観

本セクションでは、歴史、用語、ＦＴＰモデルについて論じる。ここで定義される用語は、ＦＴＰにおいて特別な意味を持つ用語のみである。用語の中には、ＦＴＰモデルに特有のものもあり、用語を確認しながら、ＦＴＰモデルに関するセクションも同時に見たいと思われる読者もいるだろう。

2．1　歴史

（省略）

2．2　用語

・ＡＳＣＩＩ

ＡＳＣＩＩ文字セットは、ＡＲＰＡ－Internetプロトコル・ハンドブックにおいて定義されるとおりである。ＦＴＰでは、ＡＳＣＩＩ文字は８ビットのコードセットの下位半分に範囲が限定される（すなわち最上位ビットはゼロである）。

・アクセス制御

アクセス制御は、システムの使用、及び同システム内のファイルに対するユーザのアクセス特権を定義する。アクセス制御は、許可されていない、または偶発的なファイルの使用を回避するために必要である。アクセス制御を行使することは、サーバＦＴＰプロセスの特権である。

・バイトサイズ

ＦＴＰに関係するバイトサイズには、ファイルの論理バイトサイズと、データの転送に使用される転送バイトサイズの２つがある。転送バイトサイズは常に８ビットである。転送バイトサイズは、必ずしもシステムにデータが記憶されるバイトサイズ、あるいはデータの構造を解釈するための論理バイトサイズとは限らない。

・制御コネクション

コマンド及び応答の交信を目的とする、ユーザＰＩ（プロトコル・インタプリタ）とサーバＰＩ間の通信経路。本コネクションはＴｅｌｎｅｔプロトコルに従う。

・データコネクション

特定のモード及びタイプで、データが転送される全二重コネクション。転送されるデータは、ファイルの一部、ファイル全体、あるいは複数のファイルの場合がある。同経路は、サーバＤＴＰ（データ転送プロセス）とユーザＤＴＰの間、あるいは２つのサーバＤＴＰの間になると考えられる。

・データポート

データコネクションをオープンするために、受動データ転送プロセスがデータポート上で、能動転送プロセスからの接続を確認するために「リスン（listen）」する。

・ＤＴＰ（データ転送プロセス）

ＤＴＰは、データコネクションを確立、管理する。ＤＴＰは受動にも、能動にもなり得る。

・End-of-Line

End-of-Line シーケンスは、印刷行の分離を定義する。同シーケンスは、キャリッジ・リターン（復帰）、次にライン・フィード（改行）である。

・ＥＯＦ（End-of-File）

転送されるファイルの終わりを定義する、End-of-File条件。

・ＥＯＲ（End-of-Record）

転送されるレコードの終わりを定義する、End-of-Record条件。

・エラー回復

ユーザが、ホストシステムまたは転送プロセスのいずれかの障害といった、一定のエラーから回復できるようにする手順。ＦＴＰでは、エラー回復に、所定のチェックポイントからファイル転送のリスタートを行う場合がある。

・ＦＴＰコマンド

ユーザＦＴＰからサーバＦＴＰプロセスに流れる制御情報を含む一連のコマンド。

・ファイル

パス名で一意に識別される、任意の長さの、順序付けられたコンピュータデータ（プログラムを含む）の集まり。

・モード

データコネクションを経由して転送されるデータの形式。モードはＥＯＲ及びＥＯＦをはじめとする、転送中のデータの形式と定義される。ＦＴＰにおいて定義される転送モードは、『３．４　転送モード』の章に記述されている。

・ＮＶＴ（Network Virtual Terminal）

Ｔｅｌｎｅｔプロトコルで定義されるとおりのネットワーク仮想端末（Network Virtual Terminal）。

・ＮＶＦＳ（Network Virtual File System）

ネットワーク仮想ファイルシステム（Network Virtual File System）。標準コマンドとパス名規定により、標準ネットワーク・ファイルシステムを定義した概念。

・ページ

１つのファイルは、ページと呼ばれる一連の独立部分により形成される場合もある。ＦＴＰは、独立した索引付ページとして連続していないファイルの転送をサポートする。

・パス名

パス名は、ファイルを識別するために、ユーザがファイルシステムに入力しなければならない文字列であると定義される。パス名は通常、装置名及び／またはディレクトリ名、並びにファイル名の指定を含む。ＦＴＰではまだ標準パス名規定が決まっていない。各ユーザは、転送に関わるファイルシステムのファイル命名規定に従わなければならない。

・ＰＩ

プロトコル・インタプリタ（Protocol Interpreter）。プロトコルのユーザ及びサーバ側が、ユーザＰＩ及びサーバＰＩで実現される明確な役割を有する。

・レコード

レコードと呼ばれる多数の連続部分として、順編成ファイルを構築することができる。レコード構造はＦＴＰによりサポートされるが、ファイルがレコード構造を持つ必要はない。

・応答（Reply）

応答は、ＦＴＰコマンドへの応答として、制御コネクションを経由してサーバからユーザに送られる（肯定または否定の）応答である。応答の一般的形態は、完了コード（エラーコードを含む）とテキスト・ストリングである。コードはプログラムによる使用を目的としており、またテキストは通常、人間のユーザを対象としている。

・サーバＤＴＰ

データ転送プロセス（ＤＴＰ）は、通常の「能動」状態で、「リスン(listen)中の」データポートとのデータコネクションを確立する。ＤＴＰは転送及び格納用のパラメタをセットアップし、ＰＩからのコマンドに応じてデータを転送する。ＤＴＰは、データポート上のコネクションを起動するのではなく、リスンするために、「受動」状態に置かれる場合もある。

・サーバＦＴＰプロセス

ユーザＦＴＰプロセス、または別のサーバと協力して、ファイル転送機能を実現する一つまたは一連のプロセス。同機能は、ＰＩ（プロトコル・インタプリタ）とＤＴＰ（データ転送プロセス）で構成される。

・サーバＰＩ

サーバＰＩは、ユーザＰＩからのコネクションを受けるために、ポートＬ上で「リスン(listen)」し、制御通信コネクションを確立する。同ＰＩはユーザＰＩから標準ＦＴＰコマンドを受け取り、応答を送り、サーバＤＴＰを管理する。

・タイプ

データの転送及び格納用に使用されるデータ表現形式。タイプは、データ格納とデータ転送の時間の間の一定の変換を意味する。ＦＴＰで定義される表現形式は、『３．２　データコネクションの確立』の章に記述されている。

・ユーザ

ファイル転送サービスを利用する個人、または個人に代わるプロセス。人間のユーザが、サーバＦＴＰプロセスと直接対話することも可能だが、プロトコル設計は自動化実現の方向で、ユーザＦＴＰプロセスを使用することがが望ましい。

・ユーザＤＴＰ

ＤＴＰ（データ転送プロセス）は、サーバＦＴＰプロセスからのコネクションを受けるために、データポート上で「リスン(listen)」する。２つのサーバが両者の間でデータを転送している場合、ユーザＤＴＰは活動停止状態になる。

・ユーザＦＴＰプロセス

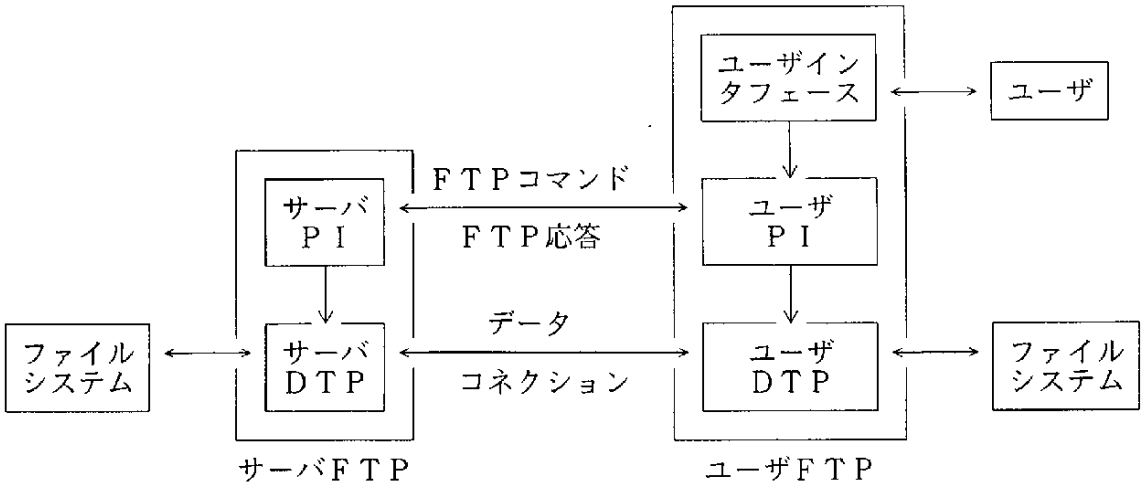
ＰＩ（プロトコル・インタプリタ）、ＤＴＰ（データ転送プロセス）、ユーザインタフェースから成る一連の機能で、１つまたはそれ以上のサーバＦＴＰプロセスと共同し、ファイル転送機能を実現する。ユーザインタフェースは、ユーザとのコマンド／応答の交信において、ローカル言語の使用を認める。

・ユーザＰＩ

ユーザＰＩは、ポートＵからサーバＦＴＰプロセスへの制御コネクションの起動、ＦＴＰコマンドの起動を行う。また、当該プロセスがファイル転送の一部である場合には、ユーザＤＴＰを管理する。

2．3　ＦＴＰモデル

上記の定義を念頭に置いて、ＦＴＰサービスに関して、以下のモデル（図１に示される）を図示することができる。



注：１．データコネクションはいずれの方向でも、利用可能である。
　　２．データコネクションは、常に存在する必要はない。

図１　ＦＴＰ使用のモデル

図１に記述されるモデルでは、ユーザＰＩが制御コネクションを起動する。制御コネクションはＴｅｌｎｅｔプロトコルに従う。ユーザの起動時に、標準ＦＴＰコマンドがユーザＰＩにより生成され、制御コネクションを経由して、サーバプロセスに転送される。（ユーザは、例えばＴＡＣ端末から、サーバＦＴＰに対して直接制御コネクションを確立し、ユーザＦＴＰプロセスをバイパスして、独立して標準ＦＴＰコマンドを生成することができる。）標準応答が、コマンドに対応して、制御コネクションを通して、サーバＰＩからユーザＰＩに送られる。

ＦＴＰコマンドは、データコネクションに関するパラメタ（データポート、転送モード、表現形式、構造）、及びファイルシステム操作の性質に関するパラメタ（格納、検索、追加、削除等）を指定する。ユーザＤＴＰまたはその代替は、特定のデータポート上で「リスン(listen)」し、サーバは指定されたパラメタに従って、データコネクション及びデータ転送を起動する。データポートは、制御コネクションを経由してＦＴＰコマンドを起動する同じホスト内にある必要はないが、ユーザまたはユーザＦＴＰプロセスは特定のデータポート上での「リスン(listen)」を保証しなければならないことに注意すべきである。さらに、データコネクションは同時送信及び受信のために利用できることにも注意すべきである。

また別の状況で、ユーザは、いずれもローカルホストではない２つのホスト間でファイルを転送することを希望するかもしれない。ユーザは、２つのサーバへの制御コネクションをセッ

トアップし、さらに両者間でのデータコネクションをアレンジする。この方法で、制御情報はユーザＰＩに渡されるが、データはサーバＤＴＰ間で転送される。以下に示すのが、このようなサーバとサーバの対話モデルである。

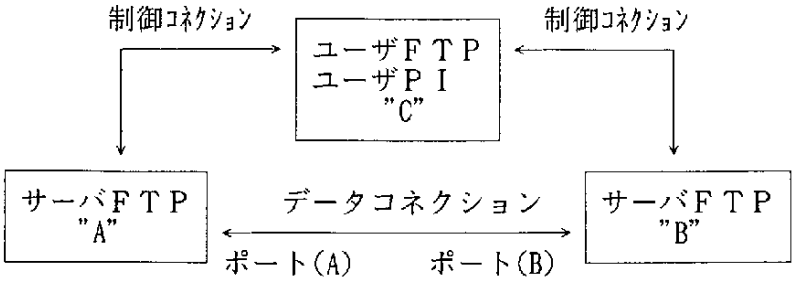


図２

プロトコルは、データ転送が行われている間、制御コネクションがオープンであることを要求する。ＦＴＰサービスの利用を終了した際に、制御コネクションをクローズすることを要求するのはユーザの責任であり、一方、クローズ動作を起こすのはサーバである。サーバは、制御コネクションがコマンドなしにクローズされた場合、データ転送を打ち切ることができる。

ＦＴＰとＴｅｌｎｅｔの関係：

ＦＴＰは、制御コネクション上でＴｅｌｎｅｔプロトコルを利用する。これは次の２つの方法で達成することができる。１つは、ユーザＰＩまたはサーバＰＩは、直接各々の手順の中で、Ｔｅｌｎｅｔプロトコルのルールを実現することができる。もう１つは、ユーザＰＩまたはサーバＰＩは、システム内の既存のＴｅｌｎｅｔモジュールを利用することができる。

インプリメンテーションの容易さ、コードの共有、モジュラ・プログラミングの点で、２番目のアプローチに軍配が上がる。効率及び独立性という面では、一番目の方法を支持できる。実際には、ＦＴＰがＴｅｌｎｅｔプロトコルに依存している程度はきわめて低いので、１番目の方法が必ずしも多くのコードを含むわけではない。

３．データ転送機能

ファイルは、データコネクションを経由してのみ転送される。制御コネクションは、実行されるべき機能を記述するコマンドの転送、及び当該コマンドへの応答（『４．２　ＦＴＰ応答』の章を参照のこと）のために利用される。コマンドの中には、ホスト間のデータ転送に関する

ものがある。これらのデータ転送コマンドには、データのビットをどのように転送するかを指定するＭＯＤＥコマンド、並びにデータが表現される方法を定義するために利用されるＳＴＲＵｃｔｕｒｅコマンド及びＴＹＰＥコマンドが含まれる。転送と表現は基本的に独立しているが、「ストリーム」転送モードはファイル構造属性に依存しており、「圧縮」転送モードが利用される場合、充填文字バイト(filler byte)の性質は表現形式によって決まる。

３．１　データ表現及び格納

データは、送信側ホスト内の記憶装置から、受信側ホスト内の記憶装置に転送される。２つのシステムのデータ記憶表現は異なるため、データ上で一定の変換を行うことが必要な場合がある。例えば、ＮＶＴ－ＡＳＣＩＩは異なるシステム内で、異なるデータ記憶表現を持っている。DEC TOPS-20sは通常、36ビット・ワードで左寄せされた５個の７ビットＡＳＣＩＩ文字として、ＮＶＴ－ＡＳＣＩＩを格納する。ＩＢＭメインフレームでは、８ビットのＥＢＣＤＩＣコードとしてＮＶＴ－ＡＳＣＩＩを格納する。Multics は、36ビット・ワードで４個の９ビット文字として、ＮＶＴ－ＡＳＣＩＩを格納する。異なるシステム間でテキストを転送する際に、文字を標準ＮＶＴ－ＡＳＣＩＩ表現に変換するのは望ましいことである。送信側及び受信側は、標準表現と各々の内部表現の間の必要な変換を実施する必要があるだろう。

ワードの異なるホストシステム間でバイナリデータ（文字コードでない）を伝送する場合、表現の相違の問題が発生する。送信側がデータを送り、また受信側がそれを格納すべき方法は、必ずしも明確ではない。例えば、ワードが32ビットのシステムから36ビットのシステムに、32ビット・バイトを伝送する場合、（効率と有効性の理由から）32ビット・バイトを後者のシステムの36ビット・ワードで、右寄せにして格納するのが望ましいと考えられる。いかなる場合にも、ユーザはデータ表現と変換機能を指定するオプションを持つべきである。ＦＴＰはきわめて限られたデータタイプ表現しか備えていない点に注意すべきである。この限られた能力を超えて望まれる変換は、直接ユーザによって実行されるべきである。

３．１．１　データ・タイプ

ＦＴＰでは、データ表現は表現形式を指定するユーザにより処理される。表現形式は、暗に（ＡＳＣＩＩまたはＥＢＣＤＩＣにおける場合のように）または明示的に（ローカルバイトにおける場合のように）、「論理バイトサイズ」と呼ばれる解釈用バイトサイズを定義すること

ができる。これは「転送バイトサイズ」と呼ばれる、データコネクションを経由した伝送のために使用されるバイトサイズとは無関係である点に注意し、この2つを混同しないようにしなければならない。例えば、ＮＶＴ－ＡＳＣＩＩは8ビットの論理バイトサイズを持つ。同タイプがローカルバイトであれば、ＴＹＰＥコマンドは、論理バイトサイズを指定する必須の第2パラメタを持つ。転送バイトサイズは常に8ビットである。

3．1．1．1　ＡＳＣＩＩタイプ

ＡＳＣＩＩタイプは省略時の値であり、ＦＴＰを実装する際にサポート必須である。双方のホストがＥＢＣＤＩＣタイプの方が都合がよいとする場合を除いて、ＡＳＣＩＩタイプはテキストファイルの転送が第一の方法である。

送信側は、データを内部文字表現から、標準の8ビットＮＶＴ－ＡＳＣＩＩ表現に変換する（Ｔｅｌｎｅｔ仕様を参照のこと）。受信側は、送られてきたデータを標準モードから自己の内部モードに変換することになる。

ＮＶＴ標準に従って、テキストの行の終わりを示す必要がある場合には、〈CRLF〉シーケンスを使用しなければならない。（『3．1　データ表現及び格納』の最後の、ファイル構造に関する議論を参照のこと。）

標準ＮＶＴ－ＡＳＣＩＩ表現を使用するということは、データは8ビット・バイトで解釈されなければならないことを意味する。

ＡＳＣＩＩ及びＥＢＣＤＩＣタイプ向けのＦｏｒｍａｔパラメタについては、以下で論じる。

3．1．1．2　ＥＢＣＤＩＣタイプ

ＥＢＣＤＩＣタイプは、内部文字表現としてＥＢＣＤＩＣを使用しているホスト間での効率的な転送を目的としている。

転送時には、データは8ビットのＥＢＣＤＩＣ文字として表現される。ＥＢＣＤＩＣタイプとＡＳＣＩＩタイプの機能仕様の唯一の相違点は文字コードにある。

End-of-Line（End-of-Recordに対抗するものとしての。構造に関する議論を参照のこと）は、構造を示す目的でＥＢＣＤＩＣタイプと共に使われることは稀だと思われるが、必要な場合には〈NL〉文字を使用すべきである。

3．1．1．3　Ｉｍａｇｅタイプ

データは、転送に関しては８ビットの転送バイトにパックされる連続ビットとして送られる。受信側は、同データを連続ビットとして格納しなければならない。記憶システムの構造は、なんらかの都合の良い境界（バイト、ワード、またはブロック）に合わせたファイル（またはレコード構造のファイルに関しては、各レコード）のパディングを必要とする可能性がある。同パディングは、すべてゼロでなければならないが、ファイルの終わり（または各レコードの終わり）でのみ発生すると考えられ、またファイルが検索される場合にはパディング・ビットが除去されるように、同ビットを識別する方法が存在しなければならない。パディング変換は、ユーザが格納側でファイルを処理できるように、十分に公表されるべきである。

Ｉｍａｇｅタイプは、ファイルの効率的格納と検索、及びバイナリデータの転送を目的としている。このタイプはＦＴＰを実装する際にサポートすることが奨励されている。

3．1．1．4　Ｌｏｃａｌタイプ

データは、必須の第２パラメタで指定されるサイズ（Byteサイズ）の論理バイトで転送される。Byteサイズの値は10進整数でなければならず、省略値はない。論理バイトサイズは、必ずしも転送バイトサイズと同一であるとは限らない。Byteサイズに相違がある場合は、論理バイトは転送バイト境界を無視し、連続してパックされなければならず、最後でなんらかの必要なパディングを行う。

データが受信側ホストに到達した時、同データは論理バイトサイズ及び特定のホストに従属した方法で変換される。この変換は、反転可能（すなわち、同じパラメタが使用された場合、同一のファイルが検索可能）でなければならず、ＦＴＰの実装者により十分に公表されるべきである。

例えば、36ビットの浮動小数点の数値を32ビット・ワードを持つホストに送るユーザは、当該データを36の論理バイトサイズを持つローカルバイトとして送ることが可能であろう。受信側ホストは、同論理バイトを容易に操作できるように格納することが期待されるだろう。この例では、36ビットの論理バイトを64ビット倍ワードにおさめれば十分なはずである。

別の例では、36ビットの語表を持つ１組のホストが、「ＴＹＰＥ　Ｌ　３６」を使用することにより、互いにワードでデータを送ることができる。同データは、９つの転送バイトが２つのホストワードを搬送するようにパックされた８ビットの転送バイトで送られることになるだろう。

3．1．1．5　書式制御

ＡＳＣＩＩタイプ及びＥＢＣＤＩＣタイプはまた、第２パラメタ（オプション）を指定できる。これは、ファイルに関連した垂直紙送り制御がある場合、その種類を示すためのものである。ＦＴＰでは、以下のデータ表現形式が定義されている。

文字ファイルは、印刷、格納及び後からの検索、処理という３つの目的のうちの１つを目的として、ホストに転送されると考えられる。印刷を目的としてファイルが転送される場合、受信側ホストは垂直紙送り制御がどのように表現されるか知らなければならない。２番目の目的の場合は、ホストでファイルを格納し、全く同じ形式で後からそのファイルを検索することが可能でなければならない。最後の場合、ファイルをひとつのホストから別のホストに移動させ、当該ファイルを不当なトラブルなしに、第２のホストで処理する必要がある。ＡＳＣＩＩまたはＥＢＣＤＩＣの単一フォーマットでは、これらの条件を全て満たすことはできない。したがって、これらのタイプは以下の３つのフォーマットの１つを指定する第２のパラメタを持つ。

3．1．1．5．1　NON　PRINT

これは、２番目の（フォーマット）パラメタが省略された場合に使用される省略時のフォーマットである。ＮＯＮ　ＰＲＩＮＴフォーマットは、ＦＴＰを実装する際にサポート必須である。

ファイルは、垂直紙送り情報を含む必要はない。ファイルが印刷処理に渡されたら、同プロセスは行間及びマージンに関して、標準値を想定することが考えられる。

通常、同フォーマットは処理または単なる格納を目的としているファイルとともに使用される。

3．1．1．5．2　ＴＥＬＮＥＴフォーマット制御

ファイルには、印刷処理が適切に実行されるために、ＡＳＣＩＩ／ＥＢＣＤＩＣの垂直紙送り制御文字（すなわち<CR>、<LF>、<NL>、<VT>、<FF>）が含まれる。<CRLF>はこのままのシーケンスで行の終わりを示す。

3．1．1．5．3　復帰制御（ＡＳＡ）

ファイルは、ＡＳＡ（FORTRAN）の垂直紙送り制御文字を含む。（RFC 740の付録Ｃ、及び1964年10月発行の「Communications of the ACM」 Vol.7、 No.10、P.606を参照のこと。）ＡＳＡ標準に従ってフォーマットされた行またはレコードでは、最初の文字は印刷されないことになっている。その代わり、ＡＳＡはレコードの残りの部分が印刷される前に起こる、紙の垂直移動を決定するために使用されるべきである。

ＡＳＡ標準は以下の制御文字を指定する。

文字	行送り
ブランク	紙を１行分上に移動させる
0	紙を２行分上に移動させる
1	紙を次ページの先頭に移動させる
＋	移動なし、すなわち重ね印刷

明らかに、印刷処理が構造主体の終わりを識別するなんらかの方法があるはずである。ファイルがレコード構造を持っている場合（以下を参照）、これは問題ではない。レコードは転送及び格納時、明示的にマークされる。ファイルがレコード構造を持たない場合は、<CRLF> End-of-Lineシーケンスが印刷行を区分するために使用されるが、これらの書式制御は、ＡＳＡ制御により無効となる。

3．1．2　データ構造

種々の表現形式に加えて、ＦＴＰはファイル構造を指定することもできる。ＦＴＰでは、以下の３つのファイル構造が定義されている。

Ｆｉｌｅ構造　　：　内部構造は存在せず、ファイルはデータバイトの連続的シーケンスとみなされる。

Ｒｅｃｏｒｄ構造：　ファイルは順次レコードで構成される。

Ｐａｇｅ構造　　：　ファイルは独立した索引付ページで構成される。

Ｆｉｌｅ構造は、ＳＴＲＵｃｔｕｒｅコマンドが使用されなかった場合に想定される省略値であるが、「テキスト」ファイル（すなわちＡＳＣＩＩまたはＥＢＣＤＩＣタイプのファイル）

使用時には、Ｆｉｌｅ構造とＲｅｃｏｒｄ構造はともに、ＦＴＰの実装においてサポート必須である。ファイルの構造は、ファイルの転送モード（『３．４　転送モード』の章を参照のこと）、並びにファイルの解釈及び格納の両方に影響を及ぼす。

ファイルの「自然」構造は、どのホストがファイルを格納するかによって決まる。ソース・コードファイルは普通、ＩＢＭメインフレーム上では固定長レコードで記憶されるが、DEC TOPS-20 では、例えば<CRLF>により、行に区分された文字の流れとして記憶される。このような異種のサイト間でのファイルの転送が有効となるには、ひとつのサイトが、別のサイトのファイルに関する想定を認識するなんらかの方法がなければならない。

あるサイトはファイル指向で、別のサイトがレコード指向であれば、あるひとつの構造をもつファイルが、別の構造指向のホストに送られる場合に問題が生じると考えられる。テキストファイルがＲｅｃｏｒｄ構造により、ファイル指向のホストに送られる場合、当該ホストはレコード構造に基づく当該ファイルに内部変換を適用しなければならない。このような変換が有益であることは明らかであるが、同時に同一ファイルがレコード構造を使用して検索できるように、可逆的に利用可能でなければならない。

ファイルがＦｉｌｅ構造によりレコード指向のホストに送られる場合、当該ファイルをローカルに処理され得るレコードに分割するため、ホストはどのような基準を使用すべきかという問題が生じる。このような分割が必要な場合、ＦＴＰの実装では、区切記号として、End-of-Line シーケンス、ＡＳＣＩＩに関しては<CRLF>、あるいはＥＢＣＤＩＣテキストファイルに関しては<NL>を使用しなければならない。ＦＴＰの実装でこの技法を採用する場合には、Ｆｉｌｅ構造によってファイルが検索される場合に、逆変換する準備がなされていなければならない。

3．1．2．1　Ｆｉｌｅ構造

Ｆｉｌｅ構造は、ＳＴＲＵｃｔｕｒｅコマンドが使用されなかった場合に想定される省略値である。

Ｆｉｌｅ構造においては、内部構造は存在せず、ファイルはデータバイトの連続的シーケンスとみなされる。

3．1．2．2　Ｒｅｃｏｒｄ構造

Ｒｅｃｏｒｄ構造は、「テキスト」ファイル（すなわちＡＳＣＩＩまたはＥＢＣＤＩＣタイプのファイル）使用時、ＦＴＰの実装においてサポート必須である。

Ｒｅｃｏｒｄ構造においては、ファイルは順次レコードにより構成される。

3．1．2．3　Ｐａｇｅ構造

非連続のファイルを伝送するために、ＦＴＰはページ構造を定義する。この種のファイルは、「ランダムアクセス・ファイル」、あるいは「多孔（holey ）ファイル」として呼ばれる場合もある。これらのファイルでは、ファイル全体（ファイル記述子等）、ファイルの一部分（ページアクセス制御等）、あるいはその両方に関連した別の情報が存在する場合がある。ＦＴＰでは、ファイルのセクションはページと呼ばれる。

様々なページサイズ及び関連情報を提供するために、各ページはページヘッダを付けて送られる。ページヘッダは以下の様に定義されるフィールドを有する。

ヘッダ長

当該バイトを含む、ページヘッダ内の論理バイト数。最小のヘッダ長は４である。

ページ・インデックス

ファイルの当該セクションの論理ページ番号。これは当該ページの伝送上のシーケンス番号ではなく、ファイルの当該ページを識別するために使用されるインデックスである。

データ長

ページデータ内の論理バイト数。最小のデータ長は０である。

ページ・タイプ

ページのタイプ。以下のページタイプが定義されている。

0＝ラストページ：

ページ構造の転送の終わりを示すために使用される。ヘッダ長は4、データ長は0でなければならない。

1＝単純ページ：

ページレベルに関連した制御情報を伴わない、単純ページ構造を持ったファイルにおいて、通常時に利用されるタイプである。ヘッダ長は4でなければならない。

2＝記述子ページ：

このタイプはファイル全体に関して、記述情報を伝送するために使用される。

3＝アクセス制御ページ：

このタイプは追加ヘッダフィールドが定義され、これは、ページレベルのアクセス制御情報を伴う、ページ構造を持ったファイルのために利用される。ヘッダ長は5でなければならない。

任意フィールド

ページ毎のアクセス制御のように、ページ毎の制御情報を提供するために、さらなるヘッダフィールドを使用することができる。

フィールドは全て、長さは1論理バイトである。論理バイトサイズは、ＴＹＰＥコマンドにより指定される。

パラメタに関する注意事項：検索されたバージョンが初めに転送されてきたバージョンと同一である場合は、ファイルの格納及び検索は同じパラメタで行わなければならない。逆に言えば、ファイルの格納及び検索に使用されるパラメタが同じ場合は、ＦＴＰインプリメンテーションはオリジナルと同一のファイルを戻さなければならない。

3．2　データコネクションの確立

データ転送の方法は、適切なポートへのデータコネクションのセットアップと、転送用パラメタの選択から成る。ユーザ及びサーバＤＴＰはともに、省略時のデータポートを持つ。ユーザプロセスのデフォルト・データポートは、制御コネクションポート（すなわちＵ）と同じである。サーバプロセスのデフォルト・データポートは、制御コネクションポートに隣接するポート（すなわちＬ－１）である。

転送バイトサイズは８ビット・バイトである。同バイトサイズは、データの実際の転送に関してのみ問題となり、ホストのファイルシステム内のデータ表現とは無関係である。

受動ＤＴＰ（ユーザＤＴＰまたは第２のサーバＤＴＰということになるだろう）は、転送要求コマンドを送る前に、データポート上で「リスン(listen)」するものとする。ＦＴＰ要求コマンドがデータ転送の方向を決定する。サーバは、転送要求を受け取ると同時に、当該ポートへのデータコネクションを起動する。コネクションが確立されると、ＤＴＰ間でデータ転送が開始され、サーバＰＩはユーザＰＩに確認応答を送る。

どのＦＴＰインプリメンテーションも、デフォルトのデータポートの使用をサポートしなければならず、ユーザＰＩのみが非省略時ポートへの変更を起動することができる。

ユーザは、ＰＯＲＴコマンドを使って、代替データポートを指定することが可能である。ユーザは、ＴＡＣラインプリンタ上にファイルをダンプしたり、第三者のホストからファイル検索するのを希望することがある。後者の場合、ユーザＰＩは双方のサーバＰＩとの制御コネクションをセットアップする。次に一方のサーバは、（ＦＴＰコマンドにより）もう一方が起動するコネクションを「リスン(listen)」するように命令される。ユーザＰＩは、一方のサーバＰＩに、もう一方のデータポートを指示するＰＯＲＴコマンドを送る。最終的に、両方に適切な転送コマンドが送られる。ユーザ制御装置とサーバの間で送られるコマンドと応答の正確な順序は、『４．２　ＦＴＰ応答』の章に定義されている。

一般に、データコネクションを維持する－－同コネクションを起動し、閉じる－－のはサーバの責任である。例外とされるのは、ユーザＤＴＰが転送モードでデータ送信している場合、ＥＯＦを示すためにコネクションをクローズする必要がある場合である。サーバは、以下の条件下で、データコネクションをクローズしなければならない。

１．サーバが、転送モードでデータの送信を完了しＥＯＦを示すためにクローズを要求する必要がある。
２．サーバが、ユーザからＡＢＯＲＴコマンドを受け取る。

3．ユーザからのコマンドで、ポート指定が変更されている。

4．制御コネクションが、合法的にまたはその他の方法でクローズされている。

5．回復不能のエラー条件が発生する。

その他の場合は、クローズはサーバの任意選択であり、サーバはクローズの実施を、２５０または２２６のいずれかの応答のみによって、ユーザプロセスに示さなければならない。

３．３　データコネクション管理

・省略時データコネクションポート：

ＦＴＰの全てのインプリメンテーションは、省略時のデータコネクションポートの使用をサポートしなければならず、ユーザＰＩのみが非省略時ポートの使用を起動させることができる。

・非省略時データポートの折衝：

ユーザＰＩは、ＰＯＲＴコマンドを使って、ユーザ側の非省略時データポートを指定することができる。ユーザＰＩはサーバ側に、ＰＡＳＶコマンドを使って、サーバ側の非省略時データポートを識別することを要求することができる。コネクションは２つのアドレスにより定義されるので、異なるデータコネクションを確立するにはこれらの動作のいずれかで十分であるが、それでもなお当該データコネクションの両端で新しいポートを使用するために、両コマンドとも実行することが認められている。

・データコネクションの再使用：

データ転送のストリームモードを使用中、ファイルの終わりはコネクションをクローズすることにより示されなければならない。この方法は、当該セッションで複数のファイルを転送する場合に、問題を生じる。信頼性の高い接続を保証するために、ＴＣＰは当該コネクションを一定時間保持している必要があるからである。したがって、当該コネクションはすぐに再びオープンすることはできない。

この問題に関しては、２つの解決方法がある。ひとつは非省略時ポートの使用を折衝することがある。もうひとつは別の転送モードを使用することである。

転送モードに関するコメント：

ストリーム転送モードでは、コネクションが途中でクローズしたかどうか、あるいはそうではなくクローズしたのかどうか判別することができないので、本質的に信頼性が低い。他の転送モード（ブロック、圧縮）では、ファイルの終わりを示すためにコネクションをクローズすることはない。それらのモードは、ファイルの終わりかどうかを判別するためにデータコネクションを分析するのに十分なＦＴＰエンコーディングを持っている。したがって、これらのモードを使用すれば、複数のファイル転送の際に、データコネクションをオープンの状態にしておくことができる。

3．4　転送モード

データを転送する際に次に考慮することは、適切な転送モードの選択である。同モードには３つの種類がある。データをフォーマット化し、リスタート手順を可能にするモード、効率的な転送のためにデータを圧縮するモード、ほとんどまたは全く処理を施さずに、データを受け渡すモードの３つである。最後の場合、同モードは処理の種類を決定するために、構造属性が関係する。圧縮モードでは、表現形式が充填バイトを決定する。

全てのデータ転送は、明示的に示されるか、またはデータコネクションをクローズすることにより暗黙的に示唆されるEnd-of-File（ＥＯＦ）で、完了されなければならない。レコード構造を持ったファイルについては、全てのEnd-of-Record（ＥＯＲ）マーカーが、最後のひとつを含めて明示的である。ページ構造で伝送されるファイルについては、「ラストページ」のページタイプが使用される。

注意：本セクションの残りの部分では、別途明示される場合を除いて、バイトという用語は「転送バイト」の意味で使用されている。

標準化の目的として、送信側ホストは、内部的な行の終わり（End-of-Line）またはレコードの終わり（End of Record ）を、転送モード及びファイル構造で記述される表現に変換し、また受信側ホストは内部的な表記法への逆変換を実行する場合もある。ＩＢＭメインフレームのレコード・カウントフィールドは相手側ホストでは認識できないので、End-of-Record 情報は、ストリームモードでは２バイトの制御コードとして、ブロックまは圧縮モードの記述子では、フラグビットとして転送される。レコード構造を持たないＡＳＣＩＩまたはＥＢＣＤＩＣファイルでのEnd-of-Lineは、それぞれ<CRLF>または<NL>で示されるものとする。これらの変

換は一部のシステムにとっては余分の作業を意味するので、非レコード構造のテキストファイルを転送する同一のシステムは、転送に関して、バイナリタイプ及びストリームモードを使用することを希望することもある。

ＦＴＰでは、以下の転送モードが定義されている。

３．４．１　ストリームモード

データは、バイトの連続として転送される。使用される表現形式に制約はなく、レコード構造も認められる。

レコード構造のファイルでは、ＥＯＲ及びＥＯＦはそれぞれ、２バイトの制御コードで指示される。制御コードの最初のバイトは全て１で、エスケープ文字である。２番目のバイトは、ＥＯＲに関しては最下位のビットはオンでその他のビットはゼロ、ＥＯＦに関しては２番目の下位のビットがオンとなる。つまり、同バイトはＥＯＲに関しては１の値、ＥＯＦに関しては２の値を持つ。ＥＯＲ及びＥＯＦは、両方の下位ビットをオンすることにより（すなわち値３になる）、転送される最後のバイト上でいっしょに示されることが可能である。全て１であるバイトがデータとして送られようとすれば、同バイトは制御コードの第２のバイトで繰り返されなければならない　。

構造がファイル構造である場合、ＥＯＦはデータコネクションをクローズする送信側ホストにより示され、全てのバイトはデータバイトである。

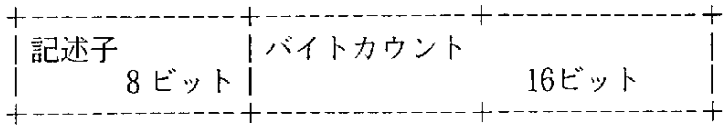
３．４．２　ブロックモード

ファイルは、ひとつまたはそれ以上のヘッダ・バイトを先頭に持つ一連のデータブロックとして伝送される。ヘッダ・バイトには、カウントフィールドと記述コードが含まれる。カウントフィールドは、データブロックの合計の長さをバイトで示し、したがって次のデータブロックの始まりをマーキングする（充填ビットはない）。記述コードは、ファイルの最後のブロック（ＥＯＦ）、レコードの最後のブロック（ＥＯＲ）、リスタート・マーカー（『３．５　エラー回復及びリスタート』の章を参照のこと）、あるいは疑わしいデータ（すなわち、転送されるデータにエラーの疑いがあり、信頼できない）を定義する。この最後のコードは、ＦＴＰ内でのエラー制御を意図するものではない。ある種のデータ（地震または天気に関するデータ）を交換するサイトの願望により、ローカルエラー（「磁気テープの読み込みエラー」等）

にもかかわらず、全てのデータを送受信する場合、当該伝送の中で、データの一定部分の真偽は疑わしいことを示す。この方式では、レコード構造は認められ、いかなる表現形式も使用可能である。

ヘッダは3バイトで構成される。ヘッダ情報の24ビットのうち、下位16ビットはバイトカウントを示し、上位8ビットは以下に示されるような記述コードを示すものとする。

ブロックのヘッダ



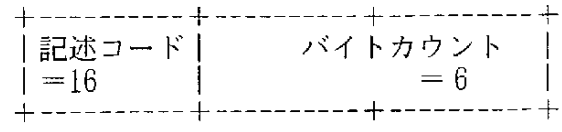
記述コードは、記述バイトの中でビットフラッグで示される。4つのコードが割り当てられており、各コード番号は、同バイト内の対応ビットの10進値である。

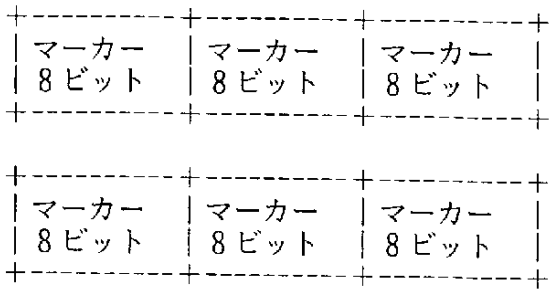
コード	意味
128	データブロックの終わりはＥＯＲ
64	データブロックの終わりはＥＯＦ
32	データブロック内で疑われるエラーがある
16	データブロックはリスタート・マーカーである

このような符号化により、特定のブロックに関して、ひとつ以上の記述コード化条件が存在する場合がある。必要な数だけのビットにフラグを立てることができる。

リスタート・マーカーは、制御コネクションで使用されている言語で（例：省略値－－ＮＶＴ－ＡＳＣＩＩ）、印刷可能な文字を表現する8ビット・バイトの整数として、データに組み込まれている。<SP>（スペース、適切な言語で）は、リスタート・マーカー内で使用されなければならない。

例えば、6文字のマーカーを伝送するためには、以下のように送られることになる。

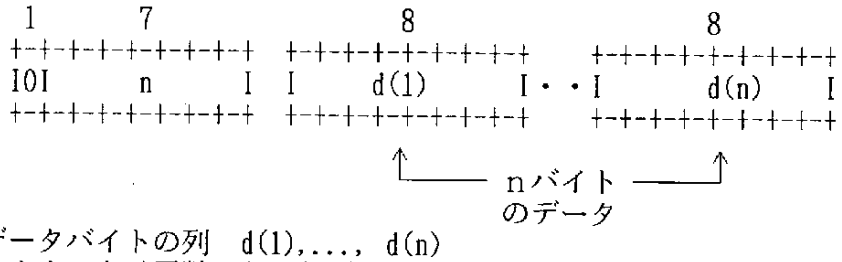




3．4．3　圧縮モード

送られる情報には次の3種類がある。バイトストリングで送られる通常のデータ、反復または充填文字で構成される圧縮データ、2バイトのエスケープ・シーケンスで送られる制御情報の3つである。n＞0バイト（最高127）の通常のデータが送られる場合、これらnバイトの前に、左端に0ビットと、右端にnという数字を表す7ビットが置かれる。

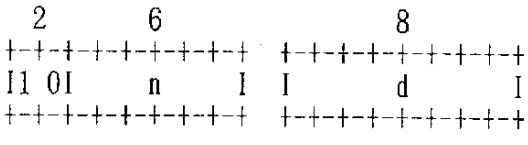
バイトストリングは以下の通りである。



nデータバイトの列　d(1),..., d(n)
nのカウントは正数でなければならない。

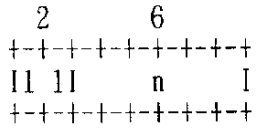
データバイトdのn反復の列を圧縮するためには、以下の2バイトが送られる。

反復バイト：



n充填バイトの列は、単一バイトに圧縮可能であり、充填バイトは表現形式により変化する。当該タイプがＡＳＣＩＩまたはＥＢＣＤＩＣである場合、充填バイトは<SP>（スペース、ＡＳＣＩＩコード 32、ＥＢＣＤＩＣコード 64）。当該タイプがイメージまたはローカルバイトの場合は、充填バイトは0バイトになる。

充填文字列：



エスケープシーケンスは2バイトで、最初の方はエスケープバイト（全て0）、後の方はブロックモードで定義される記述コードから成る。記述コードは、ブロックモードにおける場合と同じ意味を持ち、後に続くバイト列に適用される。

圧縮モードは、ちょっとした余剰ＣＰＵコストで、大量のネットワーク転送のため、帯域幅を拡大するのに有益である。同モードは、ＲＪＥ(Remote Job Entry)ホストにより生成されるようなプリンタファイルの大きさを縮小するために、最も効率的に使用することができる。

3．5　エラー回復及びリスタート

データ伝送において失われたり、スクランブルしたビットを発見する規定はない。このレベルのエラー制御はＴＣＰにより処理される。しかし、著しいシステム障害（ホスト、ＦＴＰプロセス、下位のネットワークの故障等）からユーザを保護するために、リスタート手順が準備されている。

リスタート手順は、ブロック及び圧縮モードのデータ転送に関してのみ定義されている。同手順は、データの送信側が、データストリーム中にあるマーカー情報とともに、特別なマーカーコードを挿入することを要求する。同マーカー情報は、送信側に対してのみ意味を持つが、制御コネクション上で、省略時または折衝された言語（ＡＳＣＩＩまたはＥＢＣＤＩＣ）から成る印刷可能文字で構成されていなければならない。マーカーは、システムがデータ・チェックポイントを識別できるような、ビットカウント、レコードカウント、またはその他の情報を表せる。データの受信側は、リスタート手順を実現する場合、受信システム内に同マーカーの対応位置をマークし、同情報をユーザに戻すことになる。

システムが故障した場合、ユーザはＦＴＰリスタート手順で、マーカーポイントを識別することにより、データの転送をリスタートさせることができる。以下の例は、リスタート手順の使用を示している。

データの送信側は、都合のよいポイントで、データストリーム中に、適切なマーカーブロックを挿入する。受信側ホストは、自己のファイルシステムの中に対応するデータポイントをマークし、直接または110応答の制御コネクションを介して（送り手が誰かにより決まる）、最

新の送信側及び受信側のマーカー情報をユーザに伝える。システムの故障の場合は、ユーザまたは制御装置プロセスが、サーバのマーカーコードをアーギュメントとして設定したリスタートコマンドを送ることにより、最後のサーバマーカーで、サーバをリスタートさせる。リスタートコマンドは、制御コネクションを介して転送され、すぐその後に、システムの故障が発生した時に実行されたコマンド（ＲＥＴＲ、ＳＴＯＲ、ＬＩＳＴ等）が続く。

4．ファイル転送機能

ユーザＰＩからサーバＰＩへの通信チャネルは、ユーザから標準サーバポートへのＴＣＰコネクションとして確立されている。ユーザＰＩは、ＦＴＰコマンドを送信し、受信した応答を解釈する責任を負う。サーバＰＩはコマンドを解釈し、応答を送信し、同ＰＩのＤＴＰ（データ転送プロセス）にデータコネクションをセットアップし、データを転送するように指示する。当該データ転送のセカンドパーティ（受動転送プロセス）がユーザＤＴＰであれば、ユーザＦＴＰホストの内部プロトコルを通じて管理される。同セカンドパーティが第二のサーバＤＴＰであれば、ユーザＰＩからのコマンドで、第二のサーバのＰＩにより管理される。ＦＴＰの応答については、次のセクションで論じられる。このセクションのコマンドの説明の中で可能な対応についても一部説明している。

4．1　ＦＴＰコマンド

本節では、本編『表３－１　最低限必要なＦＴＰコマンド一覧』で扱っているコマンドに関しての記述部分のみを対象に、ＲＦＣ９５９原文「５．３　コマンド」を基にして記述しています。（訳者注記）

ＦＴＰコマンドは、制御コネクションを介して伝送されるＴｅｌｎｅｔ文字列である。ＦＴＰコマンド及び構文を以下に示す。

ＦＴＰコマンドは、コマンドコードで始まり、その後にアーギュメントフィールドが続く。コマンドコードは４文字以下の英字である。大文字と小文字の英字は、全く同じに取り扱われる。したがって、以下に示されるいずれもが検索コマンドを示している。

RETR　　Retr　　retr　　ReTr　　rETr

このことはまた、ＡＳＣＩＩタイプに関するＡまたはａのように、パラメタ値を示すいかなる記号にもあてはまる。コマンドコードとアーギュメントフィールドは、１個以上のスペースで分離される。

アーギュメントフィールドは、ＮＶＴ－ＡＳＣＩＩに関しては<CRLF>（復帰改行）という文字シーケンスで終了する、可変長の文字列で構成される。その他の取り決められた言語に関しては、行の終わりを示す異なる文字が使用される場合もある。サーバは、行の終わりを示すコ

ードを受け取るまで、いかなる動作も起こさないことに注意しなければならない。

構文は、ＮＶＴ－ＡＳＣＩＩで以下に指定される。アーギュメントフィールドの全ての文字が、ＡＳＣＩＩ表現の10進整数をはじめとするＡＳＣＩＩ文字である。角括弧は、任意アーギュメントフィールドを示す。同オプションが選択されない場合、適切な省略値が暗に示される。

USER <SP> <username> <CRLF>

PASS <SP> <password> <CRLF>

ACCT <SP> <account-information> <CRLF>

CWD <SP> <pathname> <CRLF>

CDUP <CRLF>

SMNT <SP> <pathname> <CRLF>

QUIT <CRLF>

REIN <CRLF>

PORT <SP> <host-port> <CRLF>

PASV <CRLF>

TYPE <SP> <type-code> <CRLF>

STRU <SP> <structure-code> <CRLF>

MODE <SP> <mode-code> <CRLF>

RETR <SP> <pathname> <CRLF>

STOR <SP> <pathname> <CRLF>

STOU <CRLF>

APPE <SP> <pathname> <CRLF>

ALLO <SP> <decimal-integer>
[<SP> R <SP> <decimal-integer>] <CRLF>

REST <SP> <marker> <CRLF>

RNFR <SP> <pathname> <CRLF>

RNTO <SP> <pathname> <CRLF>

ABOR <CRLF>

DELE <SP> <pathname> <CRLF>

RMD <SP> <pathname> <CRLF>

MKD <SP> <pathname> <CRLF>

PWD <CRLF>

LIST [<SP> <pathname>] <CRLF>

NLST [<SP> <pathname>] <CRLF>

SITE <SP> <string> <CRLF>

SYST <CRLF>

STAT [<SP> <pathname>] <CRLF>

HELP [<SP> <string>] <CRLF>

NOOP <CRLF>

下記のＦＴＰコマンドのアーギュメントフィールドの構文は以下のとおりである。

<username> ::= <string>

<password> ::= <string>

<account-information> ::= <string>

<string> ::= <char> | <char><string>

<char> ::=１２８のＡＳＣＩＩコードのいづれか（<CR>, <LF>を除く）

<marker> ::= <pr-string>

<pr-string> ::= <pr-char> | <pr-char><pr-string>

<pr-char> ::= 印刷可能なＡＳＣＩＩコード（33　から 126）

<byte-size> ::= <number>

<host-port> ::= <host-number>,<port-number>

<host-number> ::= <number>,<number>,<number>,<number>

<port-number> ::= <number>,<number>

<number> ::=　１から２５５までの十進整数

<form-code> ::= N | T | C

<type-code> ::=　A [<sp> <form-code>]

　　| E [<sp> <form-code>]

　　| I

　　| L <sp> <byte-size>

<structure-code> ::= F | R | P

<mode-code> ::= S | B | C

<pathname> ::= <string>

<decimal-integer> ::= 十進整数

４．１．１　ＦＴＰコマンド概説

本節では、本編の『表３－１　最低限必要なＦＴＰコマンド一覧』として触れているコマンドのみ、ＲＦＣ９５９原文の「４．１．１　アクセス・コントロール・コマンド」～「４．１．３　ＦＴＰサービス・コマンド」から抜粋しています。なお、概説時に、各ＦＴＰコマンドのコマンドコードを括弧内に示します。（訳者注記）

・ＵＳＥＲ　ＮＡＭＥ（ＵＳＥＲ）

アーギュメントフィールドは、ユーザを識別するＴｅｌｎｅｔストリングである。ユーザ識別は、ファイルシステムにアクセスするためにサーバーが要求するものである。このコマンドは通常、制御コネクションが確立された後にユーザが転送する最初のコマンドとなる（これを要求するサーバもある）。サーバーによっては、パスワード及び／または課金コマンドの形で追加の識別情報も要求する場合がある。サーバは、アクセス制御及びまたは課金情報を変更するために、どこかの時点で新しいＵＳＥＲコマンドを入力することを認めるこ

とができる。これにより、既に提供されているユーザ、パスワード、及び課金に関する情報がリセットされ、新たにログイン・シーケンスが再び開始されるようになる。転送パラメタはすべて元の状態のまま変更されず、進行中のいかなるファイル転送も古いアクセス制御パラメタに基づいて完了される。

・ＰＡＳＳＷＯＲＤ（ＰＡＳＳ）

アーギュメントフィールドは、ユーザのパスワードを指定するＴｅｌｎｅｔストリングである。このコマンドの直前には、ＵＳＥＲコマンドが来なければならず、一部サイトに関してはアクセス制御のためのユーザの識別を完了する。パスワード情報は非常にデリケートなものなので、一般には同情報を「マスク」するか、タイプアウトを控えることが望ましい。サーバ側には、同行為を実施する絶対確実な方法はないようである。したがって、非常にデリケートなパスワード情報を隠すのは、ユーザＦＴＰプロセスの責任である。

・ＬＯＧＯＵＴ（ＱＵＩＴ）

このコマンドはＵＳＥＲコマンドを終了し、ファイル転送が進行中でなければ、サーバは制御コネクションをクローズする。ファイル転送が進行中であれば、同コネクションは結果応答のためにオープンの状態に維持され、サーバは後で同コネクションをクローズする。

制御コネクションでの不意のクローズは、サーバに異常終了（ＡＢＯＲ）及びログアウト（ＱＵＩＴ）を実行させる原因となる。

・ＤＡＴＡ　ＰＯＲＴ（ＰＯＲＴ）

アーギュメントは、データコネクションで使用されるデータポートに関するＨＯＳＴ－ＰＯＲＴ指定である。ユーザ及びサーバの両方のデータポートに関して省略値が存在し、通常の状況下では、このコマンド及び応答は必要とされない。同コマンドが使用される場合、アーギュメントは32ビットのインターネットホストアドレスと16ビットのＴＣＰポートアドレスを連結したものになる。同アドレス情報は８ビットのフィールドに細分され、各フィールドの値は10進数（文字列表現で）として伝送される。フィールドはコンマで区分される。

ポートコマンドは以下のようになる：

PORT h1, h2, h3, h4, p1, p2

ここでは、h1がインターネット・ホストアドレスの上位８ビットである。

・REPRESENTATION　TYPE（TYPE）

アーギュメントは、『3．1　データの表現及び格納』の章で記述されている表現形式を指定する。表現形式の中には、第二のパラメタをとるものもある。最初のパラメタはＴｅｌｎｅｔの単一文字により示される。ＡＳＣＩＩ及びＥＢＣＤＩＣ用の第二のＦｏｒｍａｔパラメタも同様である。ローカルバイト用の第二パラメタは、バイトサイズを示す10進整数である。同パラメタは、<SP>（スペース、ＡＳＣＩＩコード　32）で区分される。

以下のコードが、タイプに関して割り当てられている。

（第一パラメタ）		（第二パラメタ）
A－ＡＳＣＩＩ	⟶	N - Non-print（非印刷）
		T -Ｔｅｌｎｅｔフォーマットエフェクタ
E－ＥＢＣＤＩＣ		C -復帰制御（ASA）

Ｉ－Ｉｍａｇｅ

L<byte size>　⟶　ローカルバイトのバイトサイズ

省略時の表現形式は、ＡＳＣＩＩのNon-printである。フォーマットパラメタが変更され、後から最初のアーギュメントのみが変更された場合、フォーマットはNon-printの省略値に戻る。

・ＦＩＬＥ　ＳＴＲＵＣＴＵＲＥ（ＳＴＲＵ）

アーギュメントは、『3．1　データ表現及び格納』の章で記述されている、ファイル構造を指定する単一のＴｅｌｎｅｔ文字コードである。

以下のコードが、構造に関して割り当てられている。

Ｆ　－　Ｆｉｌｅ（非レコード構造）

Ｒ　－　Ｒｅｃｏｒｄ構造

Ｐ　－　Ｐａｇｅ構造

省略時の構造はファイルである。

・TRANSFER MODE（MODE）

アーギュメントは、『3．4　転送モード』の章で記述されている、データ転送モードを指定する　単一のＴｅｌｎｅｔ文字コードである。

以下のコードが転送方式に関して、割り当てられている。

S　－　ストリーム

B　－　ブロック

C　－　圧縮

省略時の転送方式はストリームである。

・RETRIEVE（RETR）

このコマンドは、サーバＤＴＰにパス名で指定されたファイルのコピーを、データコネクションのもう一方の終端で、サーバまたはユーザＤＴＰに転送するよう指示する。サーバ側での当該ファイルの状態及び内容は、影響を受けないものとする。

・STORE（STOR）

このコマンドは、サーバＤＴＰにデータコネクションを経由して転送されてきたデータを受け取り、同データをサーバ側でファイルとして格納するように指示する。パス名で指定されたファイルがサーバ側に存在している場合、その中身は転送されたデータと入れ換えられるものとする。パス名で指定されたファイルが既存していない場合は、サーバ側に新しいファイルが作成される。

・APPEND（APPE）

このコマンドは、サーバＤＴＰにデータコネクションを経由して転送されてきたデータを受け取り、同データをサーバ側のファイルの中に格納するように指示する。パス名で指定されたファイルがサーバ側に存在している場合、データは同ファイルに追加されるものとする。そうでない場合は、パス名で指定されたファイルがサーバ側に作成されるものとする。

・NOOP（NOOP）

このコマンドは、いかなるパラメタまたは以前に入力されたコマンドにも影響を及ぼさない。同コマンドはサーバが「ＯＫ」応答を送る動作以外は、いかなる動作も指定しない。

4．2　ＦＴＰ応答

ＦＴＰコマンドへの応答は、ファイル転送プロセスにおける要求と動作に対する同期を確実にし、ユーザプロセスが常にサーバの状態を知っていることを保証するように考案されている。全てのコマンドが、少なくともひとつの応答を生成しなければならない。ただし、ひとつ以上の場合もある。後者の場合、複数の応答は容易に識別される必要がある。また、コマンドの中には、ＵＳＥＲ、ＰＡＳＳなどのように順に出現するものもある。応答は、先行する全てのコマンドが成功している場合、中間状態の存在を示す。シーケンスのいずれかのポイントで障害が発生すると、シーケンス全体を最初から繰り返さなければならない。

ＦＴＰ応答は、３桁の数字（英字３文字として転送される）で構成され、その後に若干のテキストが続く。同数字は、次にどのような状態に入るべきかを決定するために、自動処理が行われるプロセスなどにより使用されることが意図されている。後に続くテキストは、人間のユーザを対象としている。この３桁の数字には、ユーザプロセス（ユーザＰＩ）が同テキストを調べる必要はなく、同テキストを捨てるか、ユーザにパスするかのいずれか適切な選択を行えるような、符号化された十分な情報を含んでいることが意図されている。特に、同テキストはサーバに依存しているかもしれないので、各応答コードに対して異なるテキストが存在する可能性がある。

応答は、３桁のコードを含むと定義されている。同コードの後には、スペース<SP>が続き、さらに１行のテキスト（行の最大長は指定されいている場合）が続き、ＴｅｌｎｅｔのEnd-of-Line（行の終わり）コードで終了される。しかし、テキストの長さが１行を超える場合もあると考えられる。そのような場合は、ユーザプロセスが、応答の読取りを停止（すなわち、制御コネクションでの入力を処理することを停止）し、別の作業に移ってよい時期を知ることができるように、テキストを大括弧で囲まなければならない。これには、１行目に、１行を超える行が入って来ることを示す特別なフォーマット、さらに最後の行に、それが最終行であることを指定する別の特別なフォーマットが必要である。少なくともこれらのうちの一つには、トランザクションの状態を示す適切な応答コードが含まれていなければならない。全ての機能を満たすために、第一行と最終行のコードは両方とも同一でなければならないと決められている。

したがって、複数行の応答に関するフォーマットは、第一行は要求された正確な応答コードで始まり、直後にハイフン「－」（マイナスとも呼べる）、さらにテキストが続くことになる。最終行は、同一のコードで始まり、直後にスペース<SP>、任意によりなんらかのテキスト、さらにＴｅｌｎｅｔのEnd-of-Line（行の終わり）コードが続く。

例：　123-First line
Second line
234 A line biginning with numbers
123 The last line

その後、ユーザプロセスは単純に、行の最初に<SP>（スペース）が後続する同じ応答コードが２度目に出現する行をサーチし、全ての中間行を無視する必要がある。中間行が３桁の数字で始まる場合、サーバは混乱を避けるために、先頭をパディングしなければならない。

この方式は、「人工の」第１行と最終行を付加することにより、応答情報に関して、標準システムルーチンを（例えば、ＳＴＡＴ応答に関して）使用することを可能とする。。これらのルーチンがいずれかの行の最初に、３桁の数字とスペース１個を生成することができる希な場合には、各テキスト行の始まりは、スペースのような中立的なテキストでオフセットされなければならない。

同方式は、複数行の応答はネストされないと仮定している。

応答の３桁の数字はそれぞれ、特別な意味を持つ。これは、ユーザプロセスによる非常に単純な応答から非常に複雑な応答まで、幅広く認めることを意図している。１桁目は、応答が正常か、不良か、あるいは不完全であるかどうかを示す。精巧でないユーザプロセスは、単純にこの１桁目を調べることにより、次の動作（予定通りに進行する、やり直す、削除する等）を決定することが出来る。どのようなエラー（例：ファイルシステムのエラー、コマンドの構文エラー）が発生したのかおおよそのことを知りたいと思うユーザプロセスは、２桁目を調べ、３桁目は最も精密な情報を知るために予約される。

応答コードの第１桁に関しては、５つの値がある。

1yz　肯定初期応答（Positive Preliminary reply）

要求される動作は開始されており、新しいコマンドを進行する前に、別の応答を期待している。（完了応答の前に別のコマンドを送信するユーザプロセスは、プロトコルに違反することになるが、サーバＦＴＰプロセスは先行するコマンドが進行している間、到着したコマンドを待ち行列に入れなければならない。）この種の応答は、同時に監視することが困難なインプリメンテーションに関して当該コマンドは受け入れられ、ユーザプロセスが現在、データコネクションに注目できる状態にあることを示すために使用することができる。サーバＦＴＰプロセスは、多くて１コマンドにつき一つのlyz応答を送ることができる。

2yz　肯定完了応答（Positive Completion reply）

要求される動作は成功裡に完了している。新たな要求を開始することができる。

3yz　肯定中間応答（Positive Intermediate reply）

当該コマンドは受け入れられているが、要求されている動作は一時停止状態に置かれていて、さらなる情報の受取りのため保留している。ユーザは、この情報を指定する別のコマンドを送らなければならない。同応答は、コマンドシーケンスグループに使用される。

4yz　一時的否定完了応答（Transient Negative Completion reply）

当該コマンドは受け入れられておらず、要求された動作も実施されなかったが、エラー状態は一時的で、当該動作を再び要求することが可能である。ユーザはコマンドシーケンスがある場合には、その始まりに戻らなければならない。２つの異なるサイト（サーバ及びユーザ・プロセス）が当該解釈に関して合意しなければならない場合は特に、「一時的（transient）」に意味を与えることは難しい。4yz項類における各応答は、わずかに異なる時刻の値を持つかもしれないが、その意図することは、ユーザプロセスが再試行するように促されているということである。応答が4yzあるいは5yz（永久否定）の項類にあてはまるかどうかを決定する経験則は、コマンドの形式あるいはユーザまたはサーバの特性で、いかなる変更もなしにコマンドを繰り返すことができれば、応答は4yzである（例：コマンドが同一のアーギュメントを繰り返す、ユーザが自己のファイルアクセスまたはユーザ名を変更しない、あるいはサーバが新しいインプリメンテーションをサポートしない等）。

5yz　永久的否定完了応答（Permanent Negative Completion reply）

当該コマンドは受け入れられず、要求された動作も実施されなかった。ユーザプロセスは、同一の要求（同一のシーケンスでの）の繰り返しを控えるように促される。一部の「永久的」エラー状態は修正も可能であるから、人間のユーザは自己のユーザプロセスに将来のある時点において、直接的動作により当該コマンドシーケンスを再起動させることを希望するかもしれない（例：入力文字が変更された後、あるいはユーザが自己のディレクトリ状態を変更した場合）。

以下のグループ化された機能が、第２桁で符号化されている。

x0z 構　文　　：これらの応答は、構文エラー、いかなる機能項類にも当てはまらない構文的に正しいコマンド、実現されていないまたは不必要なコマンドを示す。

x1z 情　報　　：これらは、状態またはヘルプなどの情報を求める要求に対する応答である。

x2z コネクション：制御及びデータコネクションに関する応答である。

x3z 認証及び課金：ログインプロセス及び課金手順に関する応答である。

x4z　　：まだ指定されていない。

x5z ﾌｧｲﾙｼｽﾃﾑ　：これらの応答は、要求された転送に対するサーバファイルシスムの状態、またはその他のファイルシステムの動作を示す。

第３桁は、第２桁により指定される各機能項類における、より詳細なレベルの意味を提供する。以下の応答リストはこれを示している。各応答に対応するテキストは、強制的というより、推奨されているのであり、各応答に関連しているコマンドに応じて、変更も可能であることに注意するように。一方、応答コードは最後のセクション（第４．２．１～２項）においてなされる指定に厳密に従わなければならない。つまり、サーバ・インプリメンテーションは、ここに記述されている状況とほんのわずかに異なる状況に関しても新しいコードを創り出すことはできず、むしろ既に定義されているコードを適応させなければならない。

実行が成功してもユーザプロセスにいかなる情報も提供しない、ＴＹＰＥ、ＡＬＬＯ等のコマンドには、２００応答が返される。コンピュータシステムと関連がないために、当該コマンドが特定のサーバＦＴＰプロセスにより実現されない場合（例えば、ＴＯＰＳ20サイトにおけるＡＬＬＯのように）も、単純なユーザプロセスが動作の過程を進行できるように、肯定完了応答は依然として望まれている。このような場合、２０２応答は、例えば「記憶域割当は不要」といったテキストとともに使用される。また一方で、同コマンドがサイト特有ではない動作を

要求して、それが実装されていない場合、応答は５０２である。同応答の内容をさらに詳細にしたものが、実装されているが未実装のパラメタを指定したコマンドに対する、５０４応答である。

４．２．１　機能グループごとの応答コード

200 コマンドOK。

500 構文エラー、コマンド認識されず。

これにはコマンド行が長すぎるといったエラーが含まれる場合もある。

501 パラメタまたはアーギュメントにおける構文エラー。

202 コマンドは実装されていない。このサイトでは不必要。

502 コマンドは実装されていない。

503 コマンドのシーケンス不良。

504 当該パラメタに関して、コマンドは実装されていない。

110 リスタートマーカー応答

この場合、テキストは、特定のインプリメンテーションに委ねられないで、正確に以下のようでなければならない：

MARK yyyy = mmmm

この場合、yyyyがユーザプロセスのデータストリーム・マーカーで、mmmmはサーバに対応するマーカーである（マーカーと「=」の間のスペースに注意すること）。

211 システムの状態、またはシステムに対するヘルプの応答。

212 ディレクトリの状態。

213 ファイルの状態。

214 ヘルプメッセージ。

サーバの使い方、または特定の非標準コマンドの意味に関するもの。同応答は人間のユーザにとってのみ有益である。

215 ＮＡＭＥシステムタイプ。

この場合、ＮＡＭＥはドキュメント〔Assingned Numbers(RFC1700)〕のリストの中の正式なシステム名である。

120 nnn分後にサービスの準備完了。

220 新ユーザに対して、サービスの準備完了。

221 サービスが制御コネクションをクローズ。

適切であれば、ログアウト。

421 サービスの利用不可、制御コネクションをクローズする。

当該サービスを終了しなければならないことを知っていれば、これはいかなるコマンドに対する応答にもなり得る。

125 データコネクションは既にオープンしている。転送は開始されている。

225 データコネクションはオープンしている。進行中の転送はなし。

425 データコネクションのオープン不可。

226 データコネクションのクローズ。

要求されたファイル動作は成功（ファイル転送、ファイル異常終了等）

426 コネクションはクローズされている。転送は異常終了。

227 受動モードに入る（h1, h2, h3, h4, p1, p2）。

230 ユーザがログインしている、プロセス進行中。

530 ログインされていない。

331 ユーザ名OK、パスワードが必要。

332 ログイン用の課金情報が必要。

532 ファイル記憶用の課金情報が必要。

150 ファイル状態OK、データコネクションがすぐにオープンされる。

250 要求されたファイル動作OK、完了済み。

257 「パス名」が生成される。

350 要求されたファイル動作はさらなる情報を待っている。

450 要求されたファイル動作が実施されない。

ファイルの使用不可（例：ファイルの使用中）。

550 要求された動作が実施されない。

ファイルの使用不可（例：ファイルが見つからない、アクセスできない）。

451 要求された動作の異常終了。ローカルエラーを処理中。

551 要求された動作の異常終了。ページタイプが不明。

452 要求された動作が実施されない。

システム内の記憶空間が不十分。

552 要求されたファイル動作の異常終了。

（カレントディレクトリまたはデータセット）に関する記憶域割当の超過。

553 要求された動作が実施されない。

ファイル名が認められない。

4．2．2　応答コードの番号順リスト

110 リスタートマーカー応答

この場合、テキストは、特定のインプリメンテーションに委ねられないで、正確に以下のようでなければならない：

MARK yyyy = mmmm

この場合、yyyyがユーザプロセスのデータストリーム・マーカーで、mmmmはサーバに対応するマーカーである（マーカーと「=」の間のスペースに注意すること）。

120 nnn分後にサービスの準備完了。

125 データコネクションは既にオープンしている。転送は開始されている。

150 ファイル状態OK、データコネクションがすぐにオープンされる。

200 コマンドOK。

202 コマンド実装されていない。このサイトでは不必要。

211 システムの状態、またはシステムに対するヘルプの応答。

212 ディレクトリの状態。

213 ファイルの状態。

214 ヘルプメッセージ。

サーバの使い方、または特定の非標準コマンドの意味に関するもの。同応答は人間のユーザにとってのみ有益である。

215 ＮＡＭＥシステムタイプ。

この場合、ＮＡＭＥはドキュメント〔Assingend Numbers(RFC1700)〕のリストの中の正式なシステム名である。

220 新ユーザに対して、サービスの準備完了。

221 サービスが制御コネクションをクローズ。

適切であれば、ログアウト。

225 データコネクションはオープンしている。進行中の転送はなし。

226 データコネクションのクローズ。

要求されたファイル動作は成功（ファイル転送、ファイル異常終了等）

227 受動モードに入る（h1, h2, h3, h4, p1, p2）。

230 ユーザがログインしている、プロセス進行中。

250 要求されたファイル動作OK、完了済み。

257 「パス名」が生成される。

331 ユーザ名OK、パスワードが必要。

332 ログイン用の課金情報が必要。

350 要求されたファイル動作はさらなる情報を待っている。

421 サービスの利用不可、制御コネクションをクローズする。

当該サービスを終了しなければならないことを知っていれば、これはいかなるコマンドに対する応答にもなり得る。

425 データコネクションのオープン不可。

426 コネクションはクローズされている。転送は異常終了。

450 要求されたファイル動作が実施されない。

ファイルの使用不可（例：ファイルの使用中）。

451 要求された動作の異常終了。ローカルエラーを処理中。

452 要求された動作が実施されない。

システム内の記憶空間が不十分。

500 構文エラー、コマンド認識されず。

これにはコマンド行が長すぎるといったエラーが含まれる場合もある。

501 パラメタまたはアーギュメントにおける構文エラー。

502 コマンド実装されていない。

503 コマンドのシーケンス不良。

504 当該パラメタに関して、コマンドは実装されていない。

530 ログインされていない。

532 ファイル記憶用の課金情報が必要。

550 要求された動作が実施されない。

ファイルの使用不可（例：ファイルが見つからない、アクセスできない）。

551 要求された動作の異常終了。ページタイプが不明。

552 要求されたファイル動作の異常終了。

（カレントディレクトリまたはデータセット）に関する記憶域割当の超過。

553 要求された動作が実施されない。

ファイル名が認められない。

5．宣言的仕様

5．1　最小限のインプリメンテーション

不必要なエラーメッセージなしに、ＦＴＰを動作可能とするために、全てのサーバに以下の最小限のインプリメンテーションが要求される。

データ・タイプ：ＡＳＣＩＩのNon-print

転送モード　：ストリーム

データ構造　：Ｆｉｌｅ、Ｒｅｃｏｒｄ

コマンド　　：USER、QUIT、PORT、
TYPE、MODE、STRU（省略値に関して）、
RETR、STOR、
NOOP

転送パラメタの省略値は以下の通りである。

ＴＹＰＥ　：ＡＳＣＩＩのNon-print

ＭＯＤＥ　：ストリーム

ＳＴＲＵ　：Ｆｉｌｅ

全てのホストが、上記内容を標準省略値として受け入れなければならない。

5．2　コネクション

サーバＰＩは、ポートＬ上で「リスン(listen)」するものとする。ユーザまたはユーザＰＩは、全二重制御コネクションを起動するものとする。サーバ及びユーザプロセスは、APRA-INTERNET プロトコルハンドブック［1］に指定されるとおり、Ｔｅｌｎｅｔプロトコルの規定に従わなければならない。サーバは、コマンド行の編集を準備するいかなる義務も負わされておらず、ユーザホストで行われるように要求することができる。制御コネクションは、ユーザの要求により、全ての転送及び応答が完了した後で、サーバによりクローズされるものとする。

ユーザＤＴＰは、指定されたデータポートで「リスン(listen)」しなければならない。同データポートは、省略時ユーザポート（Ｕ）か、ＰＯＲＴコマンドで指定されるポートが可能である。サーバは、指定されたユーザデータポートを使って、自己の省略時データポート（Ｌ－１）からデータコネクションを起動するものとする。転送の方向及び使用されるポートは、ＦＴＰサービスコマンドにより決定される。

ＦＴＰの全てのインプリメンテーションは、省略時ポートを使用してのデータ転送をサポートしなければならず、またユーザＰＩのみが、非省略時ポートの使用を指示することができることに注意するように。

２つのサーバＡとＢの間でデータが転送されることになる場合（図２を参照のこと）、ユーザＰＩであるＣは、両方のサーバＰＩとともに制御コネクションをセットアップする。その後でサーバの一方、例えばＡは、転送サービスコマンド受け取った時に、コネクションを起動するのではなく、自己のデータポートで「リスン(listen)」するように命じるＰＡＳＶコマンドが送られる。ユーザＰＩが、リスンが行われているポートとホストの身元を含むＰＡＳＶコマンドに対する肯定応答を受け取った時、ユーザＰＩはＡのポートであるａをＰＯＲＴコマンドでＢに送り、応答が戻って来る。ユーザＰＩはその後ＡとＢに対し、対応するサービスコマンドを送る。サーバＢはコネクションを起動し、転送が進行する。コマンド－応答シーケンスを以下に列挙する。ここではメッセージは垂直方向に同期しているが、水平方向には同期していない。

ユーザＰＩ － サーバＡ	ユーザＰＩ － サーバＢ
C->A : Connect	C->B : Connect
C->A : PASV	
A->C : 227 受動モードへ入る。	A1、A2、A3、A4、a1、a2
	C->B : PORT A1、A2、A3、A4、a1、a2
	B->C : 200 コマンドOK
C->A : STOR	C->B : RETR
B->A : Connect to Host-A, PORT-a	

図３

データコネクションは、『３．２　データコネクションの確立』の章に記述される条件の下で、サーバによりクローズされるものとする。ファイルの終わりを示すため、当該コネクションをクローズすることが要求されていないデータ転送では、転送に引き続きデータコネクションがクローズしたい場合には、サーバはただちにそうする必要がある。新しい転送コマンドを待つことは認められない。なぜならユーザプロセスは、「リスン(listen)」する必要があるかどうか知るために、既に当該データコネクションをテストしてしまった可能性があるからであ

る（ユーザは、転送要求を送る前に、クローズされたデータポートで「リスン （listen)」しなければならないことを忘れないように）。ここで競合状態を避けるために、サーバはデータコネクションをクローズした後で、応答（２２６）を送る（あるいは、当該コネクションがオープンのままであれば、「ファイル転送完了」応答（２５０）。この場合ユーザＰＩは、新しい転送コマンドを発行する前に、これら応答のうちのひとつを待つべきである）。

ユーザまたはサーバのいずれかが、もう一方の側でコネクションがクローズされていることを知ったときはいつでも、ユーザまたはサーバは、コネクションに関する待機中の残りのデータを直ちに読み取り、自己の側でクローズを行うべきである。

5．3　コマンド

（『4．1　ＦＴＰコマンド』を参照）

5．4　コマンド／応答シーケンス

（省略）

6．状態図

（省略）

7．典型的なＦＴＰシナリオ

ホストUにおけるユーザは、ホストＳへの（からの）ファイルの転送を希望している。

一般に、ユーザは介在するユーザＦＴＰプロセスを経由して、サーバと通信する。以下は典型的なシナリオのひとつと考えられるだろう。ユーザＦＴＰプロンプトは括弧で示され、「---->」はホストUからホストＳへのコマンドを示し、「<----」はホストＳからホストUへの応答を示す。

ユーザによるローカルコマンド	動作
ftp (host) multics<CR>	ホストS、ポートLへ接続 制御コネクション確立
	<---- 220 Service ready <CRLF>.
username Doe <CR>	USER Doe<CRLF>---->
	<---- 331 User name ok, need password<CRLF>.
password mumble <CR>	PASS mumble<CRLF>---->
	<---- 230 User logged in<CRLF>.
retrieve (local type) ASCII<CR>	
(local pathname) test 1 <CR>	ユーザFTPはASCIIタイプでﾛｰｶﾙﾌｧｲﾙをオープン
(for. pathname) test.pl1<CR>	RETR test.pl1<CRLF> ---->
	<---- 150 File status okay; about to open data connection<CRLF>.
	サーバはポートUへデータコネクションを起動
	<---- 226 Closing data connection, file transfer successful<CRLF>.
type Image<CR>	TYPE I<CRLF> ---->
	<---- 200 Command OK<CRLF>
store (local type) image<CR>	
(local pathname) file dump<CR>	ユーザFTPはｲﾒｰｼﾞ･ﾀｲﾌﾟでﾛｰｶﾙﾌｧｲﾙをオープン
(for.pathname) >udd>cn>fd<CR>	STOR >udd>cn>fd<CRLF> ---->
	<---- 550 Access denied<CRLF>
terminate	QUIT <CRLF> ---->
	サーバは全てのコネクションをクローズ

8．コネクションの確立

FTP制御コネクションは、ユーザプロセスのポートUとサーバプロセスのポートLの間で、TCPを介して確立される。同プロトコルはサービスポート21（8進数の25）を割り当てられているので、すなわちL＝21である。

【参考文献】

[1] Feinler, Elizabeth, "Internet Protocol Transition Workbook", Network Information Center, SRI International, March 1982.

[2] ostel, Jon, "Transmission Control Protocol - DARPA Internet Program Protocol Specification", RFC 793, DARPA, September 1981.

[3] Postel, Jon, and Joyce Reynolds, "Telnet Protocol Specification", RFC 854, ISI May 1983.

KEIRIN この資料は、競輪の補助金を受けて作成したものです。

平成8年　2月発行

発行所　財団法人　日本情報処理開発協会
産業情報化推進センター
東京都港区芝公園3丁目5番8号
機械振興会館内
TEL：（3432）9386

印刷所　有限会社　アルス
東京都港区西新橋2丁目6番3号
TEL：（3501）3030