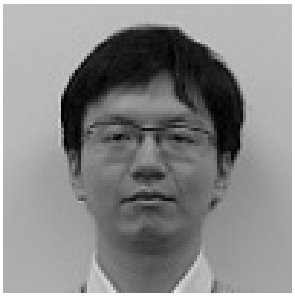
佐田 雄一

株式会社ミガロ.

システム事業部

[Delphi/400] 大量データ処理テクニック
ーFTPを利用したデータ転送



略歴
1985 年 12 月 6 日生
2009 年 甲南大学 経営学部卒
2009 年 4 月 株式会社ミガロ. 入社
2009 年 4 月 システム事業部配属

現在の仕事内容：
Delphi/400 を利用したシステム開発や保守作業を担当。Delphi および Delphi/400 のスペシャリストを目指して日々、精進している。

1.はじめに

クライアント PC から IBM i にデータ更新を行うプログラムにおいて、例えばバッチ系処理であれば何万件、もしくは何十万件という大量データを一括で更新する場合がある。こうした大量データ処理では、処理量に比例して時間を要するため、パフォーマンスが課題になることが多い。本稿では、Delphi/400 機能の 1 つである「Indy」を使用して、このような大量データ処理を行う際のレスポンス改善手法を紹介する。

2.Indyを利用したFTP転送

「Indy」とは、Delphi/400 で使用できるオープンソースのネットワーク関連コンポーネント群のことであり、Delphi/400 に標準で付属している。本稿ではこの Indy が持つ FTP 転送機能を用いて、クライアント PC から IBM i に、大量データの更新を行う手法を紹介する。

2-1. FTP転送の流れ

FTP（File Transfer Protocol）とは、サーバー間、またはサーバーとクライアント PC の間でファイルの送受信を行う際に利用される手段の 1 つである。IBM i とクライアント PC 間の FTP 転送は、主に SAVEFILE の送受信や、IBM i をファイルサーバーの代替として使用する場合に用いられる。

クライアント PC から IBM i へファイルを送信する場合、送信先には IBM i 内のライブラリを指定することが多いが、今回の処理では IFS（Integrated File System、統合ファイルシステム）領域を指定する。【図 1】

Delphi/400 と IFS の間でファイルの送受信を行う方法はほかにも存在するが、FTP 転送の技術は、一度習得すれば IBM i 以外のサーバー通信でも汎用的に活用することができる。

2-2. FTP転送を行うメリット

本稿のテクニックにおいて FTP 転送を利用する最大のメリットは、通信回数の削減によるレスポンスの向上である。

通常、アプリケーションのデータベース処理は 1 レコードごとに更新を行うため、処理レコード数が多い場合、同じ回数の通信も発生してしまう。しかし、クライアント PC から IFS 上に CSV 形式のファイルで更新内容を転送することができれば、IBM i 上では RPG で一括処理することができる。その際に便利なのが、IBM i の「CPYFRMIMPF」コマンドである。このコマンドを実行すると、その CSV ファイルの内容をデータベースファイルに直接転送することができる。

このコマンドと FTP でのファイル転送を組み合わせることで、従来であればレコード単位で発生していたクライアント PC と IBM i の間の通信が一度で済む。通信回数が削減された結果、処理時間を大幅に短縮することができるのであ

図1　IFSとクライアント間の通信イメージ

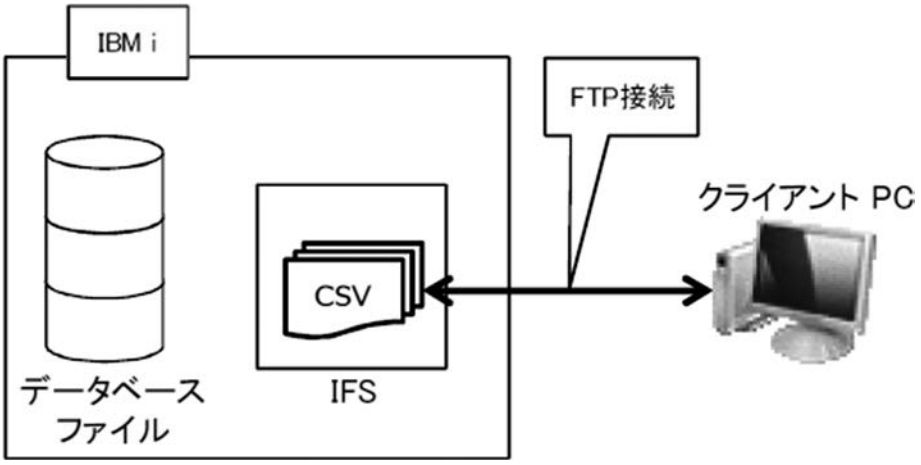


図2　10万件のレコードを同条件で更新した場合のレスポンス比較

更新方法	通信回数	処理時間
① 1レコード毎に都度RPGを呼び出して更新	10万回	約2～4分
② CSVをワークファイルに転送し、RPGで一括更新	2回(※)	約9～14秒

※　CSVファイルのFTP送信、更新RPGのCALLの計2回。

図3　FTP接続許可の確認

る。

【図2】は、IBM i上のあるデータベースファイル内のレコード10万件に対して、それぞれの手法で同じ更新を実行した場合の、通信回数と処理時間の例である。

この結果を比較すると、通信回数、処理時間ともにその差は大きく、②の手法が効果的であることがわかる。次章では、ここまでに紹介したテクニックをプログラムに実装する方法をサンプルとして紹介する。

3.クライアントPCとIBM i間のファイル、レコード転送

本章では、サンプルプログラムの作成を通じて、クライアントPCからIBM iへのFTP転送、およびCPYFRMIMPFコマンドを使用する方法を紹介する。なお、CPYFRMIMPFコマンドは、V4R3以降のIBM iで使用可能となっている。

3-1. FTP接続許可の確認

IBM iに対してファイルのFTP転送を行うための前提条件として、まずIBM iでFTP転送が許可されている必要がある。

確認方法を説明する。5250画面で「NETSTAT *CNN」コマンドを実行すると、IPV4接続状況の一覧が表示される。【図3】

この一覧の中で、「ftp-con」または21番のローカルポートが「接続待機」になっていれば、FTP接続が許可されている。存在しない場合は、5250画面側で「STRTCPSVR SERVER（*FTP)」コマンドを実行すると、開始することができる。

また、IBM i側で接続待機になっていてもFTP接続ができない場合は、ファイアウォールの設定でIBM iとのFTP接続が許可されているか確認していただきたい。

3-2. CSVファイルの書式

次に、IBM iに転送するCSVファイルの書式について説明する。【図4】はCSVファイルに格納したデータの例である。

CSVファイルの項目の並び順は、レコード転送先となるデータベースファイルのフィールド順と揃える必要がある。また、文字フィールドの値がブランクにならないように注意していただきたい。ブランクが指定されているとCPYFRMIMPFコマンド実行時にNULLとして認識されるため、意図的にNULLを渡す場合を除き、ブランクを転送するフィールドには1文字以上の半角スペースを指定する。

それ以外の制約についてはCPYFRMIMPFコマンドのパラメータに依存するため、あとはフィールドの過不足と桁あふれに気を付ければよい。CPYFRMIMPFコマンドの詳細については後述する。

3-3. サンプルプログラムの作成

上記の準備事項が確認できたら、CSVからワークファイルへの基本的なレコード転送と、更新RPGの呼び出しを行うプログラムを実際に作成していく。なお、本稿で作成しているサンプルプログラムは、Delphi/400 Version XE3を使用している。

本項でのサンプルプログラムは、次の手順で作成する。

(1)コンポーネントの配置
(2)ファイル指定［…］ボタンクリック時処理の作成
(3)送信ボタンクリック時処理の作成
(4)コマンド実行処理の作成

なお、事前準備として、IFS領域の「/QIBM/UserData」フォルダ内に「TR07/CSV」フォルダを作成しておく。本稿のサンプルプログラムでは、このフォルダに対してFTP転送を行う。また、CSVファイル名、更新先のライブラリ名およびワークファイル名は、環境に合わせて適宜読み替えていただきたい。本稿のサンプルプログラムにおけるデータの流れを【図5】に示す。

(1)コンポーネントの配置

このサンプルプログラムでは、Indyの機能の1つである「TIdFTP」コンポーネントを用いて、IBM iとのFTP転送を行う。

最初に、ネイティブ接続を行うためのTAS400、接続先情報と送信ファイルパスを入力または指定するためのTEdit、FTP転送を行うためのTIdFTP、送信ファイル指定のためのTOpenDialog、ならびにTLabelやTButtonをそれぞれ画面に配置する。【図6】

(2)ファイル指定［…］ボタンクリック時処理の作成

TOpenDialogコンポーネントを使用し、アップロードするファイルを選択するダイアログを表示させる。【ソース1】

(3)送信ボタンクリック時処理の作成

配置した送信ボタンのOnClick処理を作成したら、TIdFTPコンポーネントを使った接続・転送のソース記述を行う。【ソース2】

以下に、TIdFTPコンポーネントが持つプロパティや、今回の処理で行っているメソッドについて解説する。

(3)-①接続設定と接続処理

Host、Username、Passwordの各プロパティに、FTP転送を行うための値を設定する。Hostには接続先IBM iのサーバー名（IPアドレス)、UsernameとPasswordには接続に使用するユーザー名とパスワードを指定する。接続設定が完了したら、Connectメソッドで接続を行う。接続後は、try～finallyで処理を囲み、処理終了後はDisConnectメソッドで接続を終了する。

(3)-② TransferTypeプロパティ

ftASCII、ftBinaryの2種類が存在し、ファイルの送受信をテキスト形式、バイナリ形式のどちらで行うかを事前に指定することができる。しかし、ftASCIIを指定するとテキスト形式であっても送受信の結果に文字化けが発生する場合があるため、常にftBinaryに設定しておくことを推奨する。

なお、ftBinaryはIdFTPCommon.pasで定義されているので、uses節に「IdFTPCommon」を追加する。

(3)-③ ChangeDirメソッド

引数で指定した名前のディレクトリを、カレントディレクトリ（ユーザーが現時点で作業を行っているディレクトリ）に設定する。

《第1引数》カレントディレクトリに

図4　転送するCSVの項目設定例

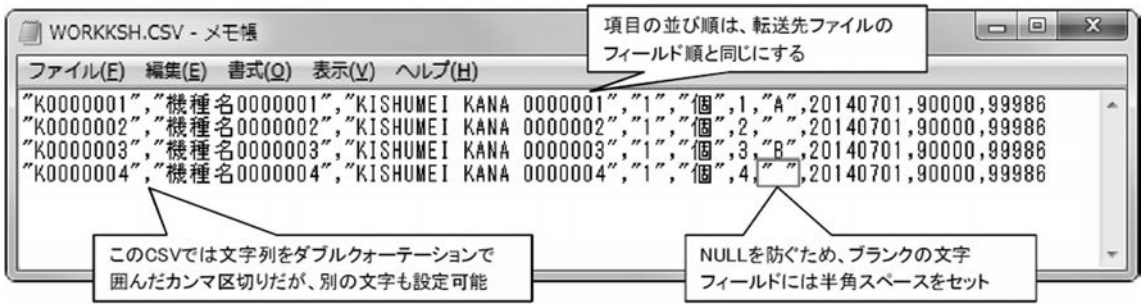


図5　サンプルプログラムのデータの流れ

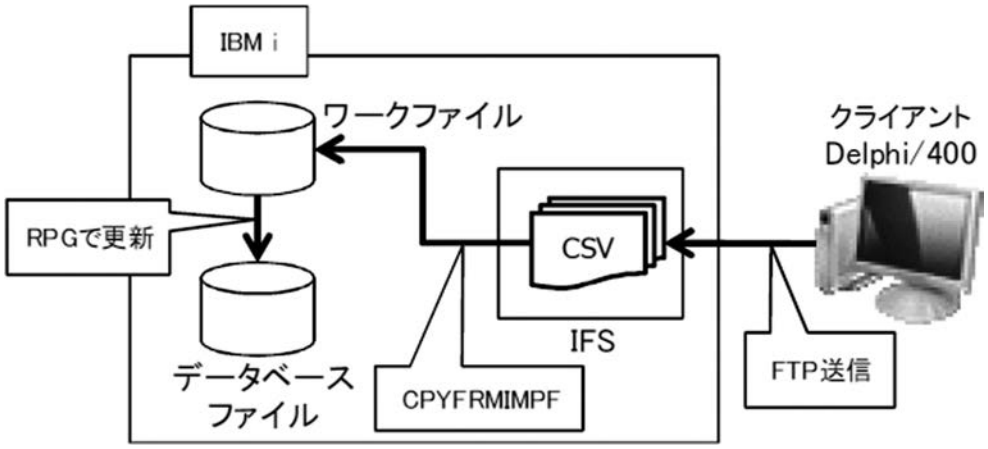


図6　サンプルプログラム画面とコンポーネント配置

設定するディレクトリパスを指定する。
※ ChangeDirUp メソッドを使うと、名前を指定せずに1つ上のディレクトリをカレントディレクトリに設定する。

(3)-④ Put メソッド

引数で指定した条件で、ファイルをFTP送信先にアップロードする。第2以降の引数は省略可能である。
《第1引数》アップロード元ファイルのフルパスを指定する。
《第2引数》アップロード先のファイル名を指定する。省略時は、第1引数と同じファイル名になる。
《第3引数》True を指定すると、上書きする。省略時は False。

(4) コマンド実行処理の作成

上記（3）の処理の最後に、IBM i 側で取り込んだ CSV のレコード転送コマンド、および更新 RPG を実行するコマンドを記述する。【ソース 3】ではコマンドとして発行しているが、TCall400 コンポーネントを利用して CL として呼び出して処理を行うこともできる。

ここで、使用する CPYFRMIMPF コマンドで使用する主なパラメータについて解説する。

(4)-① FROMSTMF（FROM ストリームファイル）

先述の TIdFTP コンポーネントで IFS 内に送信したファイル（今回は「WORKKSH.CSV」）をパス付きで指定する。

(4)-② TOFILE（TO データベースファイル）

CSV の内容を更新するデータベースファイルを指定する（今回は「TR07LIB/WORKKSH」）。

(4)-③ MBROPT（追記 / 上書きの指定）

「*REPLACE」指定時は、更新するデータベースファイルの全レコードを①のファイルの値に置き換える。「*ADD」指定時は、レコードを追記する。レコード追記の際はエラーを防ぐため、ユニークキーの重複がないか転送前に確認しておく。

(4)-④ RCDDLM（レコード区切り文字の指定）

「*CRLF」指定時は、改行とそれに続く行送りのコードを識別して、レコードの分割を行う。改行コード以外の文字も指定可能である。

(4)-⑤ STRDLM（ストリング区切り文字）※任意

文字列を囲む際の囲み文字の指定を行う。デフォルト値は「*DBLQUOTE」で、ダブルクォーテーションで囲まれた文字列を認識する。半角文字列指定時（例：「'#'」）はその文字で認識する。「*NONE」指定時は認識を行わない。

(4)-⑥ FLDDLM（フィールド区切り文字）※任意

一般的な CSV と同様に、デフォルト値はカンマ（','）となっている。タブ文字区切りにする場合は「*TAB」を、その他の文字で区切る場合はその文字を指定する。

これらのパラメータのうち①～④が必須指定、⑤と⑥が任意指定となっている。ここに記載した以外のパラメータについては、5250 上での実行において、ヘルプなどを確認していただきたい。以上で、サンプルプログラムは完成である。
実行すると、この FTP を用いた仕組みでの大量レコード処理のパフォーマンスを確認できる。

4.まとめ

大量データ処理のプログラムにおいて、どれだけパフォーマンスよく処理できるかは、1つの重要なポイントであり、課題になることも多い。もしこうした大量データ処理でパフォーマンスが課題となった場合には、本稿のレスポンス向上テクニックを一度お試しいただきたい。1レコードごとの処理で何時間もかかっているようであれば、この仕組みによって大幅に処理時間を短縮できる可能性がある。
本稿で紹介したテクニックが、業務システムをさらによくするための一助となれば幸いである。

M

ソース1　送信元ファイル指定処理

```
{*************************************************************************
 目的：送信元ファイル指定
 引数：
 戻値：
*************************************************************************}
procedure TForm1.Button2Click(Sender: TObject);
begin
  if (OpenDialog.Execute) then
  begin
    edtFromFile.Text := OpenDialog.FileName;
  end;
end;
```

ファイル選択ダイアログが開き、
OKが押されるとFileNameがセットされる。
キャンセルされた場合はif～の中を通らない。

ソース2　クライアントのファイルをIBM iに送信

```
// FTP接続
if IdFTP.Connected then
begin
  IdFTP.DisConnect;
end;
IdFTP.Host     := edtHOST.Text;
IdFTP.Username := edtUSER.Text;
IdFTP.Password := edtPASS.Text;
IdFTP.Connect;

if IdFTP.Connected then
begin
  try
    IdFTP.TransferType := ftBinary; // バイナリ形式

    IdFTP.ChangeDir('/QIBM/UserData/TR07/CSV');

    IdFTP.Put(edtFromFile.Text, '', False);
  finally
    IdFTP.DisConnect; // 最後に接続終了
  end;
end;
```

①接続設定と接続処理

②TransferTypeプロパティ
テキストでもバイナリ形式を指定

③ChangeDirメソッド
転送先ディレクトリを設定

④Putメソッド
ファイルをアップロード

ソース3　転送および更新のコマンド実行

```
// IFS→データベースファイル 転送コマンドの実行
AS400.RemoteCmd(' CPYFRMIMPF ' +
                ' FROMSTMF(''/QIBM/UserData/TR07/CSV/WORKKSH.CSV'') ' +
                ' TOFILE(TR07LIB/WORKKSH) ' +
                ' MBROPT(*REPLACE) ' +
                ' RCDDLM(*CRLF) ');

// 更新RPGの実行
AS400.RemoteCmd(' CALL PGM(TR07LIB/TR07RPG) ');
```

更新RPGの実行

IFSからデータベースファイルへのレコード転送処理
（引数について、下記1.～4.の4パラメータが必須）

1. FROMSTMF	IFS内の転送元CSVファイル
2. TOFILE	転送先データベースファイル
3. MBROPT	追記/上書きの指定（上書き）
4. RCDDLM	レコード区切り文字の指定（改行キー）