業務システム向け FAXサーバーソフト　　スターファクス サーバー エスディーケー

STARFAX® Server SDK

VC 開発向け
クライアント送信プログラミング

はじめに

本書は、STARFAX Server SDK本体をセットアップしていない別のパソコンからFAX送信を行う方法のご説明と、ユーザープログラムの作成方法をご説明しています。
なお、本書は、Visual C++ 6.0 を開発ツールとしてプログラムを作成する方を対象としていますが、STARFAX Server SDKとユーザープログラムとのインターフェイスは、ファイル入出力等の簡単な操作ですので、他の開発ツールへの応用も容易です。
本書をお読みになる前に、「STARFAX Server SDK セットアップマニュアル」をお読みいただき、STARFAX Server SDKの動作をご理解していただくようお願いいたします。

■　ご注意

本書に登場するシステム名・製品名は、一般に開発メーカーの登録商標です。

本書の構成について

本書は、次のような内容で構成されています。

- 第Ⅰ章　ユーザープログラムの開発について
- 第Ⅱ章　FAX送信命令フォルダを共有してFAX送信
- 第Ⅲ章　[メール de ＦＡＸ]でFAX送信

まずは第Ⅰ章に、開発の手順等をご説明していますので、第Ⅰ章をご覧になってから、第Ⅱ章、または、第Ⅲ章にお進み下さい。

目次

第Ⅰ章

ユーザープログラムの開発について

ユーザープログラムの開発の手順についてご説明しています。

1.1　開発の手順

1.1　開発の手順

STARFAX Server SDK をセットアップしていない別のパソコンから FAX 送信を行う為には、１台のパソコンに STARFAX Server SDK 本体のセットアップを行い、もう１台のパソコンにクライアント環境のセットアップを行う必要がります。そして、ユーザープログラム開発を行う前に、STARFAX Server SDK 本体の操作を簡単に理解しておく必要があります。
それらを考慮して、以下の手順でユーザープログラム開発を行うことをお奨めします。

① 「STARFAX Server SDK セットアップマニュアル」をお読みください。

- ［ STARFAX Server SDK のセットアップ(本体) ］を行って下ください。
- STARFAX Server SDK の基本的な操作を理解してください。
- もう１台のパソコンに 以下を行ってください。
  ［ クライアント送信のサンプルとＯＣＸ のセットアップ ］
  ［ プリンタドライバ のセットアップ ］(※)
  ［ ビューア のセットアップ ］(※)
  (※サンプルプログラムによって必要となります)

②［ STARFAX Server SDK 本体をセットアップしているパソコン ］と［ クライアント送信を行　うパソコン ］の連絡を、共有フォルダを利用して行う場合は、

第Ⅱ章　FAX 送信命令フォルダを共有して FAX 送信を、ご覧ください。

メール送信を利用して連絡を行う場合は、

第Ⅲ章　［ メール de ＦＡＸ ］で FAX 送信を、ご覧ください。
そして、クライアントパソコンから FAX 送信する仕組み、及び、各種サンプルプログラムを学習してください。

③ ユーザープログラム を作成してください。

- ①～② を踏まえて、ユーザープログラムの作成・テストを行って下さい。

第Ⅱ章

FAX送信命令フォルダを共有してFAX送信

FAX送信命令フォルダを共有して、STARFAX Server SDK本体をセットアップしていないパソコンからFAX送信する方法についてご説明します。

2.1　FAX 送信命令フォルダを共有して FAX 送信

FAX 送信命令フォルダを共有して、STARFAX Server SDK 本体をセットアップしていないパソコンから FAX 送信する方法は、以下の通りです。

以降、STARFAX Server SDK 本体をセットアップしているパソコンを [サーバー側]、STARFAX Server SDK 本体をセットアップしていないパソコンを [クライアント側] とします。

① [サーバー側]で STARFAX Server SDK を起動します。
(STARFAX Server SDK の起動は、「STARFAX Server SDK セットアップアニュアル」P21 参照)

② [サーバー側] の 送信命令フォルダ[制御関連インターフェイスフォルダ¥SENDMIS] を共有設定して、他のパソコンからアクセス(読み書き)できるようにします。
(※フォルダの共有設定方法は、ＯＳにより異なります)

③ [クライアント側]から以下の内容で、共有している送信命令フォルダに、送信命令ファイルを作成します。

- 送信命令フォルダに作業フォルダを作成します。
- 送信原稿、送付状は、作業フォルダに設定して、相対的に扱うモードを指定します。
- 送信命令処理後、作業フォルダを削除するように指定します。

(送信命令ファイルの詳細は「STARFAX Server SDK ファイル de FAX」P6「送信命令ファイルを作成し、FAX 送信を指示する」をご参照ください)

2.2　サンプルプログラム

サンプルプログラムは、３種類のプログラムを用意しています。それぞれ、元となるサンプルプログラムがあり、FAX 送信に関する部分を共有フォルダに出力する仕組みに変更して、クライアント動作するようにしています。
共有フォルダ以外の処理については、元の各種マニュアルをご覧ください。

- FAX 送信する (SendFAX.exe)

  元のプログラムは、「STARFAX Server SDK プログラミングマニュアル」の
  2.1 FAX 送信 をご覧ください。

- 印刷結果の FAX 送信 (PrtCli.exe)

  元のプログラムは、「STARFAX Server SDK VC 開発向けプリンタドライバとビューア」の
  2.1 印刷結果の FAX 送信 をご覧ください。

- TIFF ファイルの作成と FAX 送信 (PrtCli2.exe)

  元のプログラムは、「STARFAX Server SDK VC 開発向けプリンタドライバとビューア」の
  2.2 TIFF ファイルの作成と FAX 送信 をご覧ください。

2.2.1　FAX 送信

FAX 送信プログラム【SendFax.exe】 は、STARFAX Server SDK 本体をセットアップしていないパソコンから FAX の送信を行うサンプルプログラムです。本 CD-ROM の以下の位置に入っています。

¥サンプル¥応用サンプル¥クライアント¥共有フォルダ¥VC6¥SendFax.exe ...　FAX 送信プログラム
¥サンプル¥応用サンプル¥クライアント¥共有フォルダ¥VC6¥SendFax¥
... FAX 送信プログラム 開発プロジェクト

なお、このサンプルプログラムには、元となるサンプルプログラムがあり、FAX 送信に関する部分を共有フォルダに出力する仕組みに変更して、クライアント動作するようにしています。共有フォルダ以外の処理については、「STARFAX Server SDK プログラミングマニュアル」の *2.1 FAX送信* をご覧ください。

主な仕様、および操作方法は以下の通りです。
（以降、STARFAX Server SDK 本体をセットアップしているパソコンを [サーバー側]、STARFAX Server SDK 本体をセットアップしていないパソコンを [クライアント側] とします）

① [サーバー側]で STARFAX Server SDK を起動します。
（STARFAX Server SDK の起動は、「STARFAX Server SDK セットアップアニュアル」P21 参照）

② [サーバー側] の 送信命令フォルダ[制御関連インターフェイスフォルダ¥SENDMIS] を共有設定して、他のパソコンからアクセス(読み書き)できるようにします。
(※フォルダの共有設定方法は、ＯＳにより異なります)

③ [クライアント側]で FAX 送信プログラム【SendFax.exe】 を起動します。
(起動時の作業(カレント)フォルダの指定は特にありません。)

④ [共有設定した FAX 送信フォルダ(X)]を指定します。

⑤ 各種送信内容を指定します。

- [相手先(S)]ボタンを押して、相手先を指定します。
- [原稿(D)]ボタンを押して、原稿を指定します。
- 必要であれば、[送付状(C)]ボタンを押して、送付状を指定します。
- 必要であれば、[発信元(U)]ボタンを押して、発信元情報を指定します。

⑥ [送信(G)]ボタンを押して、FAX 送信を行います。
この後、FAX 送信が正常に動作していない場合は、[サーバー側] の STARFAX ログ管理プログラム で [イベント] の内容を確認して下さい。

ApiSend.cpp ：

```
//*********************************************************************
// 関数名　　：MakeSendMissionToSharedFolder
// 機能　　：共有フォルダ形式の送信命令ファイル作成
// 呼び出し　：MakeSendMissionToSharedFolder(LPCTSTR pSharedFolder, SENDFAX_MISSION *pInfo, int *piError )
// 入力　　：pSharedFolder = 共有フォルダ
//　　：pInfo　　= 送信命令ファイル作成情報ポインタ
//　　：piError　　= エラー時、エラー取得用ポインタ
// 出力　　：返値　TRUE　：正常終了
//　　：　　FALSE：失敗
//　　：　　*piError = エラーコード
//　　：
//　　：　【エラーコード】
//　　：　　SENDFAX_ERR_GetTempFolder　　... 一時フォルダの取得に失敗しました
//　　：　　SENDFAX_ERR_GetTempFile　　... 一時ファイルの取得に失敗しました
//　　：　　SENDFAX_ERR_MakeMisName　　... 送信命令ファイル名の作成に失敗しました
//　　：　　SENDFAX_ERR_NoShareFolder　　... 共有フォルダが存在しません
//　　：　　SENDFAX_ERR_MakeShareDocFolder ... 共有フォルダに送信原稿フォルダを作成できませんでした
//　　：　　SENDFAX_ERR_MakeShareDocFile　... 共有フォルダに送信原稿をコピーできませんでした
//　　：　　SENDFAX_ERR_MakeShareCvrFile　... 共有フォルダに送付状をコピーできませんでした
//　　：　　SENDFAX_ERR_PARAM_INFO　　... 関数引数エラー: 送信命令ファイル作成情報が指定されていません。
//　　：　　SENDFAX_ERR_PARAM_SENDNUM　　... 関数引数エラー: 相手先数の指定が 0 です。
//　　：　　SENDFAX_ERR_PARAM_SENDINFO　　... 関数引数エラー: 相手先情報が指定されていません。
//　　：　　SENDFAX_ERR_PARAM_FAX　　... 関数引数エラー: FAX 番号が指定されていません。
//　　：　　SENDFAX_ERR_PARAM_DOCorCOVER　... 関数引数エラー: 送信原稿ファイル、送付状ファイルが、
//　　：　　　　ともに指定されていません。
//　　：　　SENDFAX_ERR_PARAM_DOCNAME　　... 関数引数エラー: 送信原稿ファイル名が指定されていません。
//　　：　　SENDFAX_ERR_PARAM_SHAREFOLDER ... 関数引数エラー: 共有フォルダが指定されていません。
//　　：
// 特記事項　　：　共有フォルダ形式の送信命令ファイルを作成します。
// 作成者　　：
// 作成日　　：2010.07.20
//*********************************************************************
BOOL APIENTRY MakeSendMissionToSharedFolder(LPCTSTR pSharedFolder, SENDFAX_MISSION *pInfo, int *piError )
{

　～
```

```
////////////////////////////////////////////////////////////////////////////
// (1)  簡単な引数の内容チェック
//

// 共有フォルダチェック
if(!pSharedFolder || !(*pSharedFolder) ) {
    *piError = SENDFAX_ERR_PARAM_SHAREFOLDER;
    Goto    EXIT;
}

～

////////////////////////////////////////////////////////////////////////////
// (2)  共有フォルダパスのチェック
//

if(GetFileAttributes((char *)pSharedFolder ) == -1 ) {
    *piError = SENDFAX_ERR_NoShareFolder;
    Goto    EXIT;
}

////////////////////////////////////////////////////////////////////////////
// (3)  共有フォルダに送信原稿フォルダを作成
//

if(!GetTempFileName(pSharedFolder, "SFCS", 0, szDocFolder ) ) {
    *piError = SENDFAX_ERR_MakeShareDocFolder;
    Goto    EXIT;
}

DeleteFile(szDocFolder );

if(!CreateDirectory(szDocFolder, NULL ) ) {
    *piError = SENDFAX_ERR_MakeShareDocFolder;
    Goto    EXIT;
}

SUB_GetFileName(szDocFolder, szDocFolderName );

bDocFolder = TRUE;
```

～

```
////////////////////////////////////////////////////////////////////////////
// (5)  一時ファイルへ書き込み
//

～

// セクション名: [Doc]        ... 送信原稿
// ・ Num          ... 原稿数
if(pInfo->iDocNum ) {

   ～

   // ・ RelPath    ... 原稿ファイルパスを送信命令フォルダの相対パスとする
   //                   ("0":絶対パス, "1":相対パス)
   //                   (指定がない場合は "0":絶対パス です)
   WritePrivateProfileString("Doc", "RelPath", "1", szTempFile );
}

// セクション名: [Cover]     ... 送付状
// ・ Name         ... 送付状ファイルパス(.txt)
if(pInfo->pCoverName ) {

   ～

   // ・ RelPath    ... 送付状ファイルパスを送信命令フォルダの相対パスとする
   //                   ("0":絶対パス, "1":相対パス)
   //                   (指定がない場合は "0":絶対パス です)
   WritePrivateProfileString("Cover", "RelPath", "1", szTempFile );
}

～
```

```
    // セクション名: [Delete]    ... 削除要求
    //    ・ Folder      ... 命令ファイル処理後、削除フォルダ

    // 共有フォルダの相対パス
    WritePrivateProfileString("Delete", "Folder", szDocFolderName, szTempFile );

    // ・ RelPath    ... 削除フォルダを送信命令フォルダの相対パスとする
    //                   ("0":絶対パス, "1":相対パス)
    //                   (指定がない場合は "0":絶対パス です)
    WritePrivateProfileString("Delete", "RelPath", "1", szTempFile );


    ～


    return bRet;
}
```

2.2.2　印刷結果のＦＡＸ送信

印刷結果のＦＡＸ送信プログラム【PrtCli.exe】 は、STARFAX Server SDK 本体をセットアップしていないパソコンで 印刷結果の表示と FAX 送信を行うサンプルプログラムです。本 CD-ROM の以下の位置に入っています。ワード・エクセル等のアプリケーションから手動で印刷後、プリンタドライバからユーザープログラムが起動されます。

¥サンプル¥応用サンプル¥クライアント¥共有フォルダ¥VC6¥PrtCli.exe
... 印刷結果のＦＡＸ送信プログラム
¥サンプル¥応用サンプル¥クライアント¥共有フォルダ¥VC6¥PrtCli¥
... 印刷結果のＦＡＸプログラム 開発プロジェクト

なお、このサンプルプログラムには、元となるサンプルプログラムがあり、FAX 送信に関する部分を共有フォルダに出力する仕組みに変更して、クライアント動作するようにしています。共有フォルダ以外の処理については、「STARFAX Server SDK VC 開発向けプリンタドライバとビューア」の *2.1 印刷結果の FAX 送信* をご覧ください。

主な仕様、および操作方法は以下の通りです。
(以降、STARFAX Server SDK 本体をセットアップしているパソコンを [サーバー側]、STARFAX Server SDK 本体をセットアップしていないパソコンを [クライアント側] とします)

① [サーバー側]で STARFAX Server SDK を起動します。
(STARFAX Server SDK の起動は、「STARFAX Server SDK セットアップアニュアル」P21 参照)

② [サーバー側] の 送信命令フォルダ[制御関連インターフェイスフォルダ¥SENDMIS] を共有設定して、他のパソコンからアクセス(読み書き)できるようにします。
(※フォルダの共有設定方法は、ＯＳにより異なります)

③ [クライアント側]で、以下のプリンタドライバの動作に関するレジストリを指定します。

- HKEY_LOCAL_MACHINE¥Software¥MEGASOFT¥SfCs¥OutFolder … ファイル出力フォルダ
  文字列項目で、任意の作業フォルダを作成して指定します。
  (例: "C:¥Program Files¥SfCs¥Temp")
- HKEY_LOCAL_MACHINE¥Software¥MEGASOFT¥SfCs¥DocName …ドキュメント名
  文字列項目で、 このサンプルプログラムの場合、任意の文字列を指定します。
  (例: "SFCSPRN")
- HKEY_LOCAL_MACHINE¥Software¥MEGASOFT¥SfCs¥ExecFlag …プログラム実行フラグ
  DWORD 項目で、 1 を指定します。
- HKEY_LOCAL_MACHINE¥Software¥MEGASOFT¥SfCs¥ExecPath …プログラムパス
  文字列項目で、 PrtCli.exe をフルパスで指定します。
  (例: "C:¥Program Files¥SfCs¥PrtCli.exe")

● HKEY_LOCAL_MACHINE¥Software¥MEGASOFT¥SfCs¥ExecParam …追加パラメータ
文字列項目で、 何も指定していない状態(″″) を設定します。

④ 印刷可能な適当なアプリケーション(ワード等)から プリンタ名 ″MEGASOFT STARFAX Engine″ に対して印刷を行うと 印刷結果のＦＡＸ送信プログラム【PrtCli.exe】 が起動して、印刷結果リスト に 印刷結果が 登録された状態になります。
(または、印刷結果のＦＡＸ送信プログラム【PrtCli.exe】 を起動して、[テスト印刷(T)]ボタンでテスト印刷 を行っても 同じ状態になります)

⑤ [共有設定したFAX送信フォルダ(X)]を指定します。

⑥ [表示(V)]ボタンを押して、印刷結果の内容を確認します。

⑦ [ＦＡＸ送信(S)]ボタンを押して、FAX送信を行います。
この後、FAX 送信が正常に動作していない場合は、[サーバー側] の STARFAX ログ管理プログラム で [イベント] の内容を確認して下さい。

ApiSend.cpp ：

```
//*********************************************************************
// 関数名　　：MakeSendMissionToSharedFolder
// 機能　　：共有フォルダ形式の送信命令ファイル作成
// 呼び出し　：MakeSendMissionToSharedFolder(LPCTSTR pSharedFolder, SENDFAX_MISSION *pInfo, int *piError )
// 入力　　：pSharedFolder = 共有フォルダ
//　　：pInfo　　= 送信命令ファイル作成情報ポインタ
//　　：piError　　= エラー時、エラー取得用ポインタ
// 出力　　：返値　TRUE　：正常終了
//　　：　　FALSE：失敗
//　　：　　*piError = エラーコード
//　　：
//　　：　【エラーコード】
//　　：　　SENDFAX_ERR_GetTempFolder　　... 一時フォルダの取得に失敗しました
//　　：　　SENDFAX_ERR_GetTempFile　　... 一時ファイルの取得に失敗しました
//　　：　　SENDFAX_ERR_MakeMisName　　... 送信命令ファイル名の作成に失敗しました
//　　：　　SENDFAX_ERR_NoShareFolder　　... 共有フォルダが存在しません
//　　：　　SENDFAX_ERR_MakeShareDocFolder ... 共有フォルダに送信原稿フォルダを作成できませんでした
//　　：　　SENDFAX_ERR_MakeShareDocFile　... 共有フォルダに送信原稿をコピーできませんでした
//　　：　　SENDFAX_ERR_MakeShareCvrFile　... 共有フォルダに送付状をコピーできませんでした
//　　：　　SENDFAX_ERR_PARAM_INFO　　... 関数引数エラー: 送信命令ファイル作成情報が指定されていません。
//　　：　　SENDFAX_ERR_PARAM_SENDNUM　　... 関数引数エラー: 相手先数の指定が 0 です。
//　　：　　SENDFAX_ERR_PARAM_SENDINFO　　... 関数引数エラー: 相手先情報が指定されていません。
//　　：　　SENDFAX_ERR_PARAM_FAX　　... 関数引数エラー: FAX 番号が指定されていません。
//　　：　　SENDFAX_ERR_PARAM_DOCorCOVER　... 関数引数エラー: 送信原稿ファイル、送付状ファイルが、
//　　：　　　　ともに指定されていません。
//　　：　　SENDFAX_ERR_PARAM_DOCNAME　　... 関数引数エラー: 送信原稿ファイル名が指定されていません。
//　　：　　SENDFAX_ERR_PARAM_SHAREFOLDER　... 関数引数エラー: 共有フォルダが指定されていません。
//　　：
// 特記事項　　：　共有フォルダ形式の送信命令ファイルを作成します。
// 作成者　　：
// 作成日　　：2010.07.20
//*********************************************************************
BOOL APIENTRY MakeSendMissionToSharedFolder(LPCTSTR pSharedFolder, SENDFAX_MISSION *pInfo, int *piError )
{

　～
```

```
////////////////////////////////////////////////////////////////////////////
// (1)  簡単な引数の内容チェック
//

// 共有フォルダチェック
if(!pSharedFolder || !(*pSharedFolder) ) {
    *piError = SENDFAX_ERR_PARAM_SHAREFOLDER;
    Goto    EXIT;
}

～

////////////////////////////////////////////////////////////////////////////
// (2)  共有フォルダパスのチェック
//

if(GetFileAttributes((char *)pSharedFolder ) == -1 ) {
    *piError = SENDFAX_ERR_NoShareFolder;
    Goto    EXIT;
}

////////////////////////////////////////////////////////////////////////////
// (3)  共有フォルダに送信原稿フォルダを作成
//

if(!GetTempFileName(pSharedFolder, "SFCS", 0, szDocFolder ) ) {
    *piError = SENDFAX_ERR_MakeShareDocFolder;
    Goto    EXIT;
}

DeleteFile(szDocFolder );

if(!CreateDirectory(szDocFolder, NULL ) ) {
    *piError = SENDFAX_ERR_MakeShareDocFolder;
    Goto    EXIT;
}

SUB_GetFileName(szDocFolder, szDocFolderName );

bDocFolder = TRUE;
```

～

```
////////////////////////////////////////////////////////////////////////////
// (5)  一時ファイルへ書き込み
//

～

// セクション名: [Doc]        ... 送信原稿
// ・ Num          ... 原稿数
if(pInfo->iDocNum ) {

   ～

   // ・ RelPath    ... 原稿ファイルパスを送信命令フォルダの相対パスとする
   //                   ("0":絶対パス, "1":相対パス)
   //                   (指定がない場合は "0":絶対パス です)
   WritePrivateProfileString("Doc", "RelPath", "1", szTempFile );
}

// セクション名: [Cover]     ... 送付状
// ・ Name         ... 送付状ファイルパス(.txt)
if(pInfo->pCoverName ) {

   ～

   // ・ RelPath    ... 送付状ファイルパスを送信命令フォルダの相対パスとする
   //                   ("0":絶対パス, "1":相対パス)
   //                   (指定がない場合は "0":絶対パス です)
   WritePrivateProfileString("Cover", "RelPath", "1", szTempFile );
}

～
```

```
    // セクション名: [Delete]    ... 削除要求
    //    ・ Folder      ... 命令ファイル処理後、削除フォルダ

    // 共有フォルダの相対パス
    WritePrivateProfileString("Delete", "Folder", szDocFolderName, szTempFile );

    // ・ RelPath    ... 削除フォルダを送信命令フォルダの相対パスとする
    //                   ("0":絶対パス, "1":相対パス)
    //                   (指定がない場合は "0":絶対パス です)
    WritePrivateProfileString("Delete", "RelPath", "1", szTempFile );


    ～


    return bRet;
}
```

2.2.3　TIFF ファイルの作成とＦＡＸ送信

TIFF ファイルのＦＡＸ送信プログラム【PrtCli2.exe】 は、STARFAX Server SDK 本体をセットアップしていないパソコンで TIFF ファイルの作成 と FAX 送信を行うサンプルプログラムです。本 CD-ROM の以下の位置に入っています。ユーザープログラムがプリンタドライバを制御して印刷結果(TIFF ファイル)を取得します。

¥サンプル¥応用サンプル¥クライアント¥共有フォルダ¥VC6¥PrtCli2.exe
... TIFF ファイルのＦＡＸ送信プログラム
¥サンプル¥応用サンプル¥クライアント¥共有フォルダ¥VC6¥PrtCli2¥
... TIFF ファイルのＦＡＸ送信プログラム 開発プロジェクト

なお、このサンプルプログラムには、元となるサンプルプログラムがあり、FAX 送信に関する部分を共有フォルダに出力する仕組みに変更して、クライアント動作するようにしています。共有フォルダ以外の処理については、「STARFAX Server SDK VC 開発向けプリンタドライバとビューア」の *2.2 TIFF ファイルの作成と FAX 送信* をご覧ください。

主な仕様、および操作方法は以下の通りです。
(以降、STARFAX Server SDK 本体をセットアップしているパソコンを [サーバー側]、STARFAX Server SDK 本体をセットアップしていないパソコンを [クライアント側] とします)

① [サーバー側]で STARFAX Server SDK を起動します。
(STARFAX Server SDK の起動は、「STARFAX Server SDK セットアップアニュアル」P21 参照)

② [サーバー側] の 送信命令フォルダ[制御関連インターフェイスフォルダ¥SENDMIS] を共有設定して、他のパソコンからアクセス(読み書き)できるようにします。
(※フォルダの共有設定方法は、ＯＳにより異なります)

③ [クライアント側]で PrtCli2.exe を起動します。

④ [共有設定した FAX 送信フォルダ(X)]を指定します。

⑤ 「操作１」の下欄に FAX 原稿に表示させる文字を入力します。

⑥ 「①FAX 原稿の作成」ボタンをクリックします。
(作成するファイル名を任意指定したい場合は、[指定]ラジオボタンをクリックして、ファイル名を入力してください。)

⑦ 「②FAX 原稿の表示」ボタンをクリックすると、作成された FAX 原稿がビューアで表示されます。

⑧ 「操作３」の下欄に送信先の FAX 番号を入力します。

⑨ 「FAX 送信」ボタンをクリックします。
この後、FAX 送信が正常に動作していない場合は、[サーバー側] の STARFAX Server SDK ログ管理プログラム で [イベント] の内容を確認して下さい。

ApiSend.cpp ：

```
//*********************************************************************
// 関数名　　：MakeSendMissionToSharedFolder
// 機能　　：共有フォルダ形式の送信命令ファイル作成
// 呼び出し　：MakeSendMissionToSharedFolder(LPCTSTR pSharedFolder, SENDFAX_MISSION *pInfo, int *piError )
// 入力　　：pSharedFolder = 共有フォルダ
//　　：pInfo　　= 送信命令ファイル作成情報ポインタ
//　　：piError　　= エラー時、エラー取得用ポインタ
// 出力　　：返値　TRUE　：正常終了
//　　：　　FALSE：失敗
//　　：　　*piError = エラーコード
//　　：
//　　：　【エラーコード】
//　　：　　SENDFAX_ERR_GetTempFolder　　... 一時フォルダの取得に失敗しました
//　　：　　SENDFAX_ERR_GetTempFile　　... 一時ファイルの取得に失敗しました
//　　：　　SENDFAX_ERR_MakeMisName　　... 送信命令ファイル名の作成に失敗しました
//　　：　　SENDFAX_ERR_NoShareFolder　　... 共有フォルダが存在しません
//　　：　　SENDFAX_ERR_MakeShareDocFolder ... 共有フォルダに送信原稿フォルダを作成できませんでした
//　　：　　SENDFAX_ERR_MakeShareDocFile　... 共有フォルダに送信原稿をコピーできませんでした
//　　：　　SENDFAX_ERR_MakeShareCvrFile　... 共有フォルダに送付状をコピーできませんでした
//　　：　　SENDFAX_ERR_PARAM_INFO　　... 関数引数エラー: 送信命令ファイル作成情報が指定されていません。
//　　：　　SENDFAX_ERR_PARAM_SENDNUM　　... 関数引数エラー: 相手先数の指定が 0 です。
//　　：　　SENDFAX_ERR_PARAM_SENDINFO　　... 関数引数エラー: 相手先情報が指定されていません。
//　　：　　SENDFAX_ERR_PARAM_FAX　　... 関数引数エラー: FAX 番号が指定されていません。
//　　：　　SENDFAX_ERR_PARAM_DOCorCOVER　... 関数引数エラー: 送信原稿ファイル、送付状ファイルが、
//　　：　　　　ともに指定されていません。
//　　：　　SENDFAX_ERR_PARAM_DOCNAME　　... 関数引数エラー: 送信原稿ファイル名が指定されていません。
//　　：　　SENDFAX_ERR_PARAM_SHAREFOLDER ... 関数引数エラー: 共有フォルダが指定されていません。
//　　：
// 特記事項　　：　共有フォルダ形式の送信命令ファイルを作成します。
// 作成者　　：
// 作成日　　：2010.07.20
//*********************************************************************
BOOL APIENTRY MakeSendMissionToSharedFolder(LPCTSTR pSharedFolder, SENDFAX_MISSION *pInfo, int *piError )
{

　～
```

```
////////////////////////////////////////////////////////////////////////////
// (1)  簡単な引数の内容チェック
//

// 共有フォルダチェック
if(!pSharedFolder || !(*pSharedFolder) ) {
    *piError = SENDFAX_ERR_PARAM_SHAREFOLDER;
    Goto    EXIT;
}

～

////////////////////////////////////////////////////////////////////////////
// (2)  共有フォルダパスのチェック
//

if(GetFileAttributes((char *)pSharedFolder ) == -1 ) {
    *piError = SENDFAX_ERR_NoShareFolder;
    Goto    EXIT;
}

////////////////////////////////////////////////////////////////////////////
// (3)  共有フォルダに送信原稿フォルダを作成
//

if(!GetTempFileName(pSharedFolder, "SFCS", 0, szDocFolder ) ) {
    *piError = SENDFAX_ERR_MakeShareDocFolder;
    Goto    EXIT;
}

DeleteFile(szDocFolder );

if(!CreateDirectory(szDocFolder, NULL ) ) {
    *piError = SENDFAX_ERR_MakeShareDocFolder;
    Goto    EXIT;
}

SUB_GetFileName(szDocFolder, szDocFolderName );

bDocFolder = TRUE;
```

～

```
////////////////////////////////////////////////////////////////////////////
// (5)  一時ファイルへ書き込み
//

～

// セクション名: [Doc]        ... 送信原稿
// ・ Num          ... 原稿数
if(pInfo->iDocNum ) {

   ～

   // ・ RelPath    ... 原稿ファイルパスを送信命令フォルダの相対パスとする
   //                   ("0":絶対パス, "1":相対パス)
   //                   (指定がない場合は "0":絶対パス です)
   WritePrivateProfileString("Doc", "RelPath", "1", szTempFile );
}

// セクション名: [Cover]     ... 送付状
// ・ Name         ... 送付状ファイルパス(.txt)
if(pInfo->pCoverName ) {

   ～

   // ・ RelPath    ... 送付状ファイルパスを送信命令フォルダの相対パスとする
   //                   ("0":絶対パス, "1":相対パス)
   //                   (指定がない場合は "0":絶対パス です)
   WritePrivateProfileString("Cover", "RelPath", "1", szTempFile );
}

～
```

```
    // セクション名: [Delete]    ... 削除要求
    //    ・ Folder      ... 命令ファイル処理後、削除フォルダ

    // 共有フォルダの相対パス
    WritePrivateProfileString("Delete", "Folder", szDocFolderName, szTempFile );

    // ・ RelPath    ... 削除フォルダを送信命令フォルダの相対パスとする
    //                   ("0":絶対パス, "1":相対パス)
    //                   (指定がない場合は "0":絶対パス です)
    WritePrivateProfileString("Delete", "RelPath", "1", szTempFile );


    ～


    return bRet;
}
```

第Ⅲ章

[メール de ＦＡＸ]でFAX送信

[メール de ＦＡＸ]機能で、STARFAX Server SDK本体をセットアップしていないパソコンからFAX送信する方法についてご説明します。

3.1 ［ メール de ＦＡＸ ］でFAX送信

［ メール de ＦＡＸ ］機能で、STARFAX Server SDK本体をセットアップしていないパソコンからFAX送信する方法は、以下の通りです。
（以降、STARFAX Server SDK本体をセットアップしているパソコンを［ サーバー側 ］、STARFAX Server SDK本体をセットアップしていないパソコンを［ クライアント側 ］とします）

① ［ サーバー側 ］で［ メール de ＦＡＸ ］の設定を有効にします。

(1) タスクトレイの STARFAX Server SDK のアイコンを右クリックして表示されるメニューから［ メール de ＦＡＸ ］を実行します。

(2) 最低限、以下の項目を設定します。

- ［ メール de ＦＡＸ を有効にする(Y) ］をチェックします。
- ［ メールサーバーの設定(S) ］を行います。

(3) 必要に応じて、以下の項目を設定します。

- ［ 件名に含まれる文字で識別(I) ］
- ［ メール受信間隔(N) ］
- ［ 対象外のメールをEMLファイルに保存(P) ］、及び、関連項目
- ［ FAX送信結果をメール通知する(N) ］、及び、関連項目

② ［ サーバー側 ］で STARFAX Server SDK を起動します。
(STARFAX Server SDKの起動は、「STARFAX Server SDK セットアップアニュアル」P21参照)

③ ［ クライアント側 ］から［ メール de ＦＡＸ ］の依頼メールを［ サーバー側 ］で設定した［ POPメールアドレス ］に送信します。

(［ メール de ＦＡＸ ］の依頼メールの詳細は *3.2 ［ メール de ＦＡＸ ］の依頼メールの仕様* をご覧下さい)

3.2 ［ メール de ＦＡＸ ］の依頼メールの仕様

STARFAX Server SDK の［ メール de ＦＡＸ ］機能が動作している場合、設定された［ POP メールアドレス ］に対しての依頼メールを送信すると、FAX 送信を行います。
この仕組みには、以下の特徴があります。

- **相手先を複数指定することができます。(同報送信)**
- **送信原稿ファイルを複数指定することができます。**
- **送付状を指定することができます。**
  (送付状ファイルは、テキストファイルで、差込内容を %? で指定できます。)
- **発信元情報を指定することができます。**

依頼メールの仕様についてご説明します。

■ 依頼メールの構成

- **宛先**

  ［ メール de ＦＡＸ 設定 ］の［ メールサーバー設定(S) ］で設定されている［ POP メールアドレス ］を指定します。

- **件名**

  件名には、［ メール de ＦＡＸ 設定 ］の［ 件名に含まれる文字で識別(I) ］で設定されている文字列を含んでいる必要があります。含んでいないと依頼メールとして扱われません。

  例: ″ Mail to FAX″

- **本文**

  ［ メール de ＦＡＸ ］では扱われません。
  (設定されていても FAX 送信されることはありません)

● **添付ファイル**

添付ファイルは以下のファイルを指定します。

- **送信命令ファイル**
- **送信原稿ファイル**
- **送付状ファイル**

詳細については後述をご覧ください。

■ 送信命令ファイル

送信命令ファイルは、INI ファイル形式のファイルです。
ファイル名は、必ず、 "Trans.txt" である必要があります。
全ての項目を指定する必要はありません。
最低限、相手先の FAX 番号を指定すれば FAX 送信できます。(※)
(※: 添付ファイルで送信原稿ファイルが設定されている必要があります)

```
[SendInfo]
Num=1
[Send1]
Fax=0663868894
```

その他の情報は、送付状への差込、発信元情報への差込、ログに情報を残したい、等の必要に応じて設定します。

● **セクション名: [SendInfo] ... 送信のための相手先情報**

- Num ... 送信相手先数

● **セクション名: [Send%d]　... 送信のための相手先内容 (1～)**

- Fax　... FAX 番号 (最大 128 バイト)
- Company　... 会社名 (最大 128 バイト)
- Division　... 所属名 (最大 128 バイト)
- Position　... 役職名 (最大 128 バイト)
- Name　... 氏名 (最大 128 バイト)
- Title　... 敬称 (最大 128 バイト)
- Telephone　... 電話番号 (最大 128 バイト)
- ZipCode　... 郵便番号 (最大 128 バイト)
- Address1　... 住所 1 (最大 128 バイト)
- Address2　... 住所 2 (最大 128 バイト)
- Speed　... 通信速度 ("0":自動, "1":高速, "2":中速, "3":低速)
- Comp　... 圧縮方式 ("0":自動, "1":MH, "2":MR, "3":MMR)
- Ecm　... エラー訂正 ("0":自動, "1":利用する, "2":利用しない)
- FCode　... Ｆコード番号 (最大 20 バイト)
- FreeArea　... ユーザが自由に利用できるエリア (最大 128 バイト)
- Line　... 送信回線指定 (0～)
  (指定がない場合は、空いている回線から送信されます)
- Priority　... 優先順位 (0～15)
  (指定がない場合の優先順位は 8 です。)
- Time　... 送信時刻 ("YYYYMMDDHHMMSS")

● **セクション名: [Cover]　... 送付状**

- Name　... 送付状ファイルパス(.txt)
  (さらに、添付ファイルで設定する必要があります)
- FontName　... 送付状フォント名
  (指定がない場合は "ＭＳ ゴシック" です)
- FontSize　... 送付状フォントサイズ (8 ～ 72 (ポイント))
  (指定がない場合は 10 ポイント です)

● **セクション名: [User]　... 発信元情報**

- UserInfo　... 発信元情報記録 ("記録位置,記録情報")
  ("0":記録しない, "1":原稿の内側に記録, "2":原稿の外側に記録)
- UserID　... 自局電話番号 (FAXID として、相手ファクシミリに通知されます。)
  (最大 20ﾊﾞｲﾄで、半角数字、スペース、+を設定してください。)

【 発信元情報の仕様 】

発信元情報とは、FAX 送信時に、各ページ毎、送信原稿の先頭に付加する情報です。
発信元情報は、[User]-UserInfo に、"記録位置,記録情報" の形式で指定します。

● **記録位置**

0: 記録しません。
1: 原稿の内側に記録します。原稿の内容によっては、原稿の先頭が少し消えてしまう可能性があります。
2: 原稿の外側に記録します。

1、または、2 を指定すると、記録情報を指定していなくも日付とページ数は記録されます。

```
[User]
UserInfo=1,
```

↓

```
                              2001 12/ 1 10:12      Page 01
```

● **記録情報**

自由に文字を指定できます。
以下の差込も使用できます。

- %S ... 会社名 ([Send%d]セクション-Company が差し込まれます。)
- %N ... 氏名 ([Send%d]セクション-Name と [Send%d]セクション-Title が差し込まれます。)
- %T ... FAX 番号 ([Send%d]セクション-Fax が差し込まれます。)

```
[Send1]
Company=山田サークル
Name=山田
[User]
UserInfo=1,メガ太郎 → %S %N
```

↓

```
      メガ太郎 → 山田サークル 山田様          2001 12/ 1 10:12      Page 01
```

■ 送信原稿ファイル

送信原稿ファイルは、以下の形式のファイルを複数指定可能です。

ファイル形式:
- TIFF 形式　圧縮なし
  - 修正 CCITT MH 圧縮
  - CCITT G3 MH 圧縮
  - CCITT G3 MR 圧縮
  - PackBits 圧縮
  - Class F 圧縮
  - G4 圧縮
  - JPEG 圧縮
- BMP ファイル
- PCX ファイル
- DCX ファイル
- JPEG ファイル
- テキストファイル
- FAX ファイル
- LNK ファイル

■　送付状ファイルの仕様

テキストファイル(*.txt)で、自由に作成したファイルを指定できます。
文字フォントは、″ＭＳ　ゴシック″、大きさは１０ポイントとして扱われ、FAX送信されます。

以下の差込が有効です。

- ● **%S　...　会社名　　([Send%d]セクションーCompany が差し込まれます。)**
- ● **%D　...　所属名　　([Send%d]セクションーDivision が差し込まれます。)**
- ● **%Y　...　役職名　　([Send%d]セクションーPosition が差し込まれます。)**
- ● **%N　...　氏名　　　([Send%d]セクションーName が差し込まれます。)**
- ● **%Z　...　郵便番号　([Send%d]セクションーZipCode が差し込まれます。)**
- ● **%A　...　住所１　　([Send%d]セクションーAddress1 が差し込まれます。)**
- ● **%B　...　住所２　　([Send%d]セクションーAddress2 が差し込まれます。)**
- ● **%H　...　電話番号　([Send%d]セクションーTelephone が差し込まれます。)**
- ● **%T　...　FAX番号　([Send%d]セクションーFax が差し込まれます。)**
- ● **%K　...　敬称　　　([Send%d]セクションーTitle が差し込まれます。)**
- ● **%P　...　ページ数　(自動的に計算され、差し込まれます。)**
- ● **%x　...　xxxx　([Send%d]セクションー が差し込まれます。)**

```
_/_/_/_/_/_/_/_/_/_/_/_/_/_/_/_/_/_/_/_/_/_/_/_/_/_/_/_/_/_/_/_/_/_/_/_/_/_/_/_/_/_/_/
                              ファクシミリ送付のご案内
_/_/_/_/_/_/_/_/_/_/_/_/_/_/_/_/_/_/_/_/_/_/_/_/_/_/_/_/_/_/_/_/_/_/_/_/_/_/_/_/_/_/_/

%S
%D
%Y
%N　%K

                                                               メガソフト株式会社
                                                                          メガ太郎

毎度格別のお引き立てにあずかり、まことにありがとうございます。
下記の書類を拝送しますので、よろしくご査収下さいますようお願い申し上げます。

                                     記

                                                                           以上
```

3.3　サンプルプログラム

サンプルプログラムは、３種類のプログラムを用意しています。それぞれ、元となるサンプルプログラムがあり、FAX 送信に関する部分を[メール de ＦＡＸ]の仕組みに変更して、クライアント動作するようにしています。
[メール de ＦＡＸ]以外の処理については、元の各種マニュアルをご覧ください。

- FAX 送信する (SendFAX.exe)

  元のプログラムは、「STARFAX Server SDK プログラミングマニュアル」の
  2.1 FAX 送信 をご覧ください。

- 印刷結果の FAX 送信 (PrtCli.exe)

  元のプログラムは、「STARFAX Server SDK VC 開発向けプリンタドライバとビューア」の
  2.1 印刷結果の FAX 送信 をご覧ください。

- TIFF ファイルの作成と FAX 送信 (PrtCli2.exe)

  元のプログラムは、「STARFAX Server SDK VC 開発向けプリンタドライバとビューア」の
  2.2 TIFF ファイルの作成と FAX 送信 をご覧ください。

3.3.1　FAX 送信

FAX 送信プログラム【SendFax.exe】 は、STARFAX Server SDK 本体をセットアップしていないパソコンから FAX の送信を行うサンプルプログラムです。本 CD-ROM の以下の位置に入っています。

¥Sample¥応用サンプル¥クライアント¥メール de ＦＡＸ¥VC6¥SendFax.exe
... FAX 送信プログラム
¥Sample¥応用サンプル¥クライアント¥メール de ＦＡＸ¥VC6¥SendFax¥
... FAX 送信プログラム 開発プロジェクト

なお、このサンプルプログラムには、元となるサンプルプログラムがあり、FAX 送信に関する部分を[メール de ＦＡＸ]の仕組みに変更して、クライアント動作するようにしています。
[メール de ＦＡＸ]以外の処理については、「STARFAX Server SDK プログラミングマニュアル」の *2.1 FAX 送信* をご覧ください。

主な仕様、および操作方法は以下の通りです。
(以降、STARFAX Server SDK 本体をセットアップしているパソコンを [サーバー側]、STARFAX Server SDK 本体をセットアップしていないパソコンを [クライアント側] とします)

① [サーバー側]で [メール de ＦＡＸ] の設定を有効にします。

(1) タスクトレイの STARFAX Server SDK のアイコンを右クリックして表示されるメニューから [メール de ＦＡＸ] を実行します。

(2) 最低限、以下の項目を設定します。

- [メール de ＦＡＸ を有効にする(Y)] をチェックします。
- [メールサーバーの設定(S)] を行います。

(3) 必要に応じて、以下の項目を設定します。

- [件名に含まれる文字で識別(I)]
- [メール受信間隔(N)]
- [対象外のメールを EML ファイルに保存(P)]、及び、関連項目
- [FAX 送信結果をメール通知する(N)] 、及び、関連項目

② [サーバー側]で STARFAX Server SDK を起動します。
(STARFAX Server SDK の起動は、「STARFAX Server SDK セットアップアニュアル」P21 参照)

③ [クライアント側]で FAX 送信プログラム【SendFax.exe】 を起動します。
(起動時の作業(カレント)フォルダの指定は特にありません。)

④ 各種送信内容を指定します。

- [相手先(S)]ボタンを押して、相手先を指定します。
- [原稿(D)]ボタンを押して、原稿を指定します。
- 必要であれば、[送付状(C)]ボタンを押して、送付状を指定します。
- 必要であれば、[発信元(U)]ボタンを押して、発信元情報を指定します。

⑤ [メール送信(G)]ボタンを押して、メール送信を行います。
この後、FAX 送信が正常に動作していない場合は、[サーバー側] の以下の表示をご確認下さい。

- STARFAX Server SDK ログ管理プログラム の [イベント]
  (主に、[コントロール] と [メール] をご確認下さい)
- [メール de ＦＡＸ 設定] の システムメニュー の [イベントログ]

ApiSend.cpp ：

```
//*********************************************************************
// 関数名　　：MakeTransFileForMailToFax
// 機能　　：[ メール de ＦＡＸ ]の送信命令ファイル作成
// 呼び出し　：MakeTransFileForMailToFax(LPCTSTR pTransFile, SENDFAX_MISSION *pInfo, int *piError )
// 入力　　：pTransFile = 送信命令ファイル
//　　：pInfo　　= 送信命令ファイル作成情報ポインタ
//　　：piError　= エラー時、エラー取得用ポインタ
// 出力　　：返値　TRUE　：正常終了
//　　：　　FALSE：失敗
//　　：　　　　*piError = エラーコード
//　　：
//　　：　【エラーコード】
//　　：　　SENDFAX_ERR_GetTempFolder　... 一時フォルダの取得に失敗しました
//　　：　　SENDFAX_ERR_GetTempFile　　... 一時ファイルの取得に失敗しました
//　　：　　SENDFAX_ERR_CreateTransFile ... 送信命令ファイルの作成に失敗しました
//　　：　　SENDFAX_ERR_PARAM_INFO　　... 関数引数エラー: 送信命令ファイル作成情報が指定されていません。
//　　：　　SENDFAX_ERR_PARAM_SENDNUM　... 関数引数エラー: 相手先数の指定が 0 です。
//　　：　　SENDFAX_ERR_PARAM_SENDINFO　... 関数引数エラー: 相手先情報が指定されていません。
//　　：　　SENDFAX_ERR_PARAM_FAX　　... 関数引数エラー: FAX 番号が指定されていません。
//　　：　　SENDFAX_ERR_PARAM_TRANSFILE ... 関数引数エラー: 送信命令ファイルが指定されていません。
//　　：
// 特記事項　　：　[ メール de ＦＡＸ ]の送信命令ファイルを作成します。
// 作成者　　：
// 作成日　　：2010.07.20
//*********************************************************************
BOOL APIENTRY　MakeTransFileForMailToFax(LPCTSTR pTransFile, MAIL2FAX_MISSION *pInfo, int *piError )
{
    BOOL　bRet = FALSE;　// 返値 ... 初期値は失敗

    int iErrorDummy;　// NULL 用ダミーエラーコード設定

    int i;

    char　szGetName[MAX_PATH+1];　　// ファイル名取得用

    SENDFAX_SENDINFO　*pSendInfo; // 相手先情報用

    char　szSection[32]; // セクション設定用
    char　szWork[512];　// 作業用

    if(!piError ) {
        // エラーコード取得用ポインタにダミー設定
        piError = &iErrorDummy;
```

```
}
*piError = SENDFAX_SUCCESS;      // エラーコードに初期値設定 ... 正常終了

////////////////////////////////////////////////////////////////////////////
// (1)  簡単な引数の内容チェック
//

    // 送信命令ファイルチェック
if(!pTransFile || !(*pTransFile) ) {
    *piError = SENDFAX_ERR_PARAM_TRANSFILE;
    goto    EXIT;
}

    // 送信命令ファイル作成情報チェック
if(!pInfo ) {
    *piError = SENDFAX_ERR_PARAM_INFO;
    goto    EXIT;
}
    // 相手先数チェック
if(!pInfo->iSendNum ) {
    *piError = SENDFAX_ERR_PARAM_SENDNUM;
    goto    EXIT;
}
    // 相手先情報チェック
for(i = 0 ;  i < pInfo->iSendNum ; i++ ) {
    pSendInfo = *(pInfo->ppSend + i );
    if(!pSendInfo ) {
        *piError = SENDFAX_ERR_PARAM_SENDINFO;
        goto    EXIT;
    }
        // FAX 番号チェック
    if(!pSendInfo->szFax[0] ) {
        *piError = SENDFAX_ERR_PARAM_FAX;
        goto    EXIT;
    }
}

////////////////////////////////////////////////////////////////////////////
// (2)  送信命令ファイルへ書き込み
//

DeleteFile(pTransFile );   // 念のため削除

// セクション名: [SendInfo]  ... 送信のための相手先情報
```

```
// ・ Num        ... 送信相手先数
wsprintf(szWork, "%d", pInfo->iSendNum );
WritePrivateProfileString("SendInfo", "Num", szWork, pTransFile );

// セクション名: [Send%d]    ... 送信のための相手先内容 (1～)
for(i = 0 ;  i < pInfo->iSendNum ; i++ ) {
    pSendInfo = *(pInfo->ppSend + i );

    wsprintf(szSection, "Send%d", i + 1 );

    // ・ Fax        ... FAX 番号 (最大 128 バイト)
    WritePrivateProfileString(szSection, "Fax", pSendInfo->szFax, pTransFile );
    // ・ Company    ... 会社名 (最大 128 バイト)
    if(pSendInfo->szCompany[0] ) {
        wsprintf(szWork, "¥"%s¥"", pSendInfo->szCompany );
        WritePrivateProfileString(szSection, "Company", szWork, pTransFile );
    }
    // ・ Division   ... 所属名 (最大 128 バイト)
    if(pSendInfo->szDivision[0] ) {
        wsprintf(szWork, "¥"%s¥"", pSendInfo->szDivision );
        WritePrivateProfileString(szSection, "Division", szWork, pTransFile );
    }
    // ・ Position   ... 役職名 (最大 128 バイト)
    if(pSendInfo->szPosition[0] ) {
        wsprintf(szWork, "¥"%s¥"", pSendInfo->szPosition );
        WritePrivateProfileString(szSection, "Position", szWork, pTransFile );
    }
    // ・ Name       ... 氏名 (最大 128 バイト)
    if(pSendInfo->szName[0] ) {
        wsprintf(szWork, "¥"%s¥"", pSendInfo->szName );
        WritePrivateProfileString(szSection, "Name", szWork, pTransFile );
    }
    // ・ Title      ... 敬称 (最大 128 バイト)
    if(pSendInfo->szTitle[0] ) {
        wsprintf(szWork, "¥"%s¥"", pSendInfo->szTitle );
        WritePrivateProfileString(szSection, "Title", szWork, pTransFile );
    }
    // ・ Telephone  ... 電話番号 (最大 128 バイト)
    if(pSendInfo->szTelephone[0] ) {
        WritePrivateProfileString(szSection, "Telephone", pSendInfo->szTelephone, pTransFile );
    }
    // ・ ZipCode    ... 郵便番号 (最大 128 バイト)
    if(pSendInfo->szZipCode[0] ) {
        WritePrivateProfileString(szSection, "ZipCode", pSendInfo->szZipCode, pTransFile );
    }
```

```
// ・ Address1   ... 住所1 (最大128バイト)
if(pSendInfo->szAddress1[0] ) {
    wsprintf(szWork, "¥"%s¥"", pSendInfo->szAddress1 );
    WritePrivateProfileString(szSection, "Address1", szWork, pTransFile );
}
// ・ Address2   ... 住所2 (最大128バイト)
if(pSendInfo->szAddress2[0] ) {
    wsprintf(szWork, "¥"%s¥"", pSendInfo->szAddress2 );
    WritePrivateProfileString(szSection, "Address2", szWork, pTransFile );
}
// ・ FCode      ... Fコード番号 (最大20バイト)
if(pSendInfo->szFCode[0] ) {
    wsprintf(szWork, "¥"%s¥"", pSendInfo->szFCode );
    WritePrivateProfileString(szSection, "FCode", szWork, pTransFile );
}
// ・ FreeArea   ... ユーザが自由に利用できるエリア (最大128バイト)
if(pSendInfo->szFreeArea[0] ) {
    wsprintf(szWork, "¥"%s¥"", pSendInfo->szFreeArea );
    WritePrivateProfileString(szSection, "FreeArea", szWork, pTransFile );
}
// ・ Speed      ... 通信速度 ("0":自動, "1":高速, "2":中速, "3":低速)
if(pSendInfo->iSpeed ) {
    wsprintf(szWork, "%d", pSendInfo->iSpeed );
    WritePrivateProfileString(szSection, "Speed", szWork, pTransFile );
}
// ・ Comp       ... 圧縮方式 ("0":自動, "1":MH, "2":MR, "3":MMR)
if(pSendInfo->iComp ) {
    wsprintf(szWork, "%d", pSendInfo->iComp );
    WritePrivateProfileString(szSection, "Comp", szWork, pTransFile );
}
// ・ Ecm        ... エラー訂正 ("0":自動, "1":利用する, "2":利用しない)
if(pSendInfo->iEcm ) {
    wsprintf(szWork, "%d", pSendInfo->iEcm );
    WritePrivateProfileString(szSection, "Ecm", szWork, pTransFile );
}
// ・ Line       ... 送信回線指定 (0～)
if(pSendInfo->bLine ) {
    wsprintf(szWork, "%d", pSendInfo->iLine );
    WritePrivateProfileString(szSection, "Line", szWork, pTransFile );
}
// ・ Priority   ... 優先順位 (0～15)
if(pSendInfo->bPriority ) {
    wsprintf(szWork, "%d", pSendInfo->iPriority );
    WritePrivateProfileString(szSection, "Priority", szWork, pTransFile );
}
```

```
    // ・ Time        ... 送信時刻 ("YYYYMMDDHHMMSS")
    if(pSendInfo->bTime ) {
        wsprintf(szWork, "%s", pSendInfo->szTime );
        WritePrivateProfileString(szSection, "Time", szWork, pTransFile );
    }
}

// セクション名: [Cover]     ... 送付状
// ・ Name        ... 送付状ファイルパス(.txt)
if(pInfo->pCoverName ) {

    SUB_GetFileName(pInfo->pCoverName, szGetName );

    wsprintf(szWork, "¥"%s¥"", szGetName );
    WritePrivateProfileString("Cover", "Name", szWork, pTransFile );
}
// ・ FontName   ... 送付状フォント名
if(pInfo->pCoverFontName ) {
    wsprintf(szWork, "¥"%s¥"", pInfo->pCoverFontName );
    WritePrivateProfileString("Cover", "FontName", szWork, pTransFile );
}
// ・ FontSize   ... 送付状フォントサイズ (8 ～ 72 (ポイント ) )
if(pInfo->iCoverFontSize ) {
    wsprintf(szWork, "%d", pInfo->iCoverFontSize );
    WritePrivateProfileString("Cover", "FontSize", szWork, pTransFile );
}

// セクション名: [User]       ... 発信元情報
// ・ UserInfo   ... 発信元情報記録 ("記録位置,記録情報") (最大 140 バイト)
//                    ("0":記録しない, "1":原稿の内側に記録, "2":原稿の外側に記録)
if(pInfo->szUserInfo[0] ) {
    wsprintf(szWork, "¥"%s¥"", pInfo->szUserInfo );
    WritePrivateProfileString("User", "UserInfo", szWork, pTransFile );
}
// ・ UserID     ... 自局電話番号 (FAXID として、相手ファクシミリに通知されます。) (最大 20 バイト)
if(pInfo->szUserID[0] ) {
    wsprintf(szWork, "¥"%s¥"", pInfo->szUserID );
    WritePrivateProfileString("User", "UserID", szWork, pTransFile );
}

////////////////////////////////////////////////////////////////////////////
// (3) 送信命令ファイルのフラッシュ 【重要】
//
//      OS により、WritePrivateProfileString() 【Win32API】 を使用しての
//    実際のファイルへの書き出しのタイミングが異なる為、必ずこのフラッシュの
```

```
    //    ステップが必要です。
    //

    WritePrivateProfileString(NULL, NULL, NULL, pTransFile );

    bRet = TRUE;    // 正常終了

EXIT:

    // エラー時は、送信命令ファイルを削除
    if(!bRet ) {
        DeleteFile(pTransFile );
    }

    return bRet;
}
```

SendFaxDlg.cpp ：

```
void CSendFaxDlg::OnOK()
{

    ～

    ////////////////////////////////////////////////////////////////////////////
    // (2) [ メール de ＦＡＸ ]の送信命令ファイル作成関数 呼び出し

    if(!MakeTransFileForMailToFax(szTrans, &MailToFaxInfo, &iError ) ) {

        switch(iError ) {
        case    SENDFAX_ERR_GetTempFolder:
            MessageBox("一時フォルダの取得に失敗しました", "FAX 送信", MB_ICONEXCLAMATION | MB_OK );
            break;
        case    SENDFAX_ERR_GetTempFile:
            MessageBox("一時ファイルの取得に失敗しました", "FAX 送信", MB_ICONEXCLAMATION | MB_OK );
            break;
        case    SENDFAX_ERR_CreateTransFile:
            MessageBox("送信命令ファイルの作成に失敗しました", "FAX 送信", MB_ICONEXCLAMATION | MB_OK );
            break;
        case    SENDFAX_ERR_PARAM_INFO:
            MessageBox("関数引数エラー: 送信命令ファイル作成情報が指定されていません。","FAX 送信",MB_ICONEXCLAMATION |
MB_OK );
            break;
        case    SENDFAX_ERR_PARAM_SENDNUM:
            MessageBox(" 関数引数エラー: 相手先数の指定が 0 です。", "FAX 送信", MB_ICONEXCLAMATION | MB_OK );
            break;
        case    SENDFAX_ERR_PARAM_SENDINFO:
            MessageBox("関数引数エラー: 相手先情報が指定されていません。", "FAX 送信", MB_ICONEXCLAMATION | MB_OK );
            break;
        case    SENDFAX_ERR_PARAM_FAX:
            MessageBox("関数引数エラー: FAX 番号が指定されていません。", "FAX 送信", MB_ICONEXCLAMATION | MB_OK );
            break;
        case    SENDFAX_ERR_PARAM_TRANSFILE:
            MessageBox("関数引数エラー: 送信命令ファイルが指定されていません。","FAX 送信",MB_ICONEXCLAMATION|MB_OK);
            break;
        default:
            break;
        }

    }
    else {
```

```
MessageBox(
"このサンプルプログラムは MAPI を利用してメール送信を行います。¥n 宛先 に STARFAX Server SDK の [ メール de ＦＡＸ ] で設定した¥nPOP メールアドレス を指定して送信してください。¥n¥n なお、送信後、OUTLOOK 等メーラーを起動して送信を行わないと¥n 実際に送信されないことがあります。", "FAX 送信", MB_ICONINFORMATION | MB_OK );

////////////////////////////////////////////////////////////////////////////
//
// メール送信
//
////////////////////////////////////////////////////////////////////////////

////////////////////////////////////////////////////////////////////////////
// MAPI 初期化

HINSTANCE hInstMail = ::LoadLibraryA("MAPI32.DLL" );

if(hInstMail == NULL ) {
    AfxMessageBox(AFX_IDP_FAILED_MAPI_LOAD );
    goto    EXIT;
}

ULONG (PASCAL *lpfnSendMail )(ULONG, ULONG, MapiMessage*, FLAGS, ULONG );
(FARPROC&)lpfnSendMail = GetProcAddress(hInstMail, "MAPISendMail" );
if(lpfnSendMail == NULL ) {
    AfxMessageBox(AFX_IDP_INVALID_MAPI_DLL );
    ::FreeLibrary(hInstMail );
    goto    EXIT;
}

////////////////////////////////////////////////////////////////////////////
// 添付ファイル設定

// prepare the file description (for the attachment)
MapiFileDesc    fileDesc[6];

int iAttach = 0;

// 送信命令ファイル
ZeroMemory(&fileDesc[iAttach], sizeof(MapiFileDesc) );
fileDesc[iAttach].nPosition = iAttach;
fileDesc[iAttach].lpszPathName = szTrans;
fileDesc[iAttach].lpszFileName = GetFileNamePtr(szTrans );
iAttach++;
```

```
// 送信原稿
if(m_szDocName1[0] ) {
    ZeroMemory(&fileDesc[iAttach], sizeof(MapiFileDesc) );
    fileDesc[iAttach].nPosition = iAttach;
    fileDesc[iAttach].lpszPathName = m_szDocName1;
    fileDesc[iAttach].lpszFileName = GetFileNamePtr(m_szDocName1 );
    iAttach++;
}
if(m_szDocName2[0] ) {
    ZeroMemory(&fileDesc[iAttach], sizeof(MapiFileDesc) );
    fileDesc[iAttach].nPosition = iAttach;
    fileDesc[iAttach].lpszPathName = m_szDocName2;
    fileDesc[iAttach].lpszFileName = GetFileNamePtr(m_szDocName2 );
    iAttach++;
}
if(m_szDocName3[0] ) {
    ZeroMemory(&fileDesc[iAttach], sizeof(MapiFileDesc) );
    fileDesc[iAttach].nPosition = iAttach;
    fileDesc[iAttach].lpszPathName = m_szDocName3;
    fileDesc[iAttach].lpszFileName = GetFileNamePtr(m_szDocName3 );
    iAttach++;
}
if(m_szDocName4[0] ) {
    ZeroMemory(&fileDesc[iAttach], sizeof(MapiFileDesc) );
    fileDesc[iAttach].nPosition = iAttach;
    fileDesc[iAttach].lpszPathName = m_szDocName4;
    fileDesc[iAttach].lpszFileName = GetFileNamePtr(m_szDocName4 );
    iAttach++;
}

// 送付状
if(MailToFaxInfo.pCoverName ) {
    ZeroMemory(&fileDesc[iAttach], sizeof(MapiFileDesc) );
    fileDesc[iAttach].nPosition = iAttach;
    fileDesc[iAttach].lpszPathName = MailToFaxInfo.pCoverName;
    fileDesc[iAttach].lpszFileName = GetFileNamePtr(MailToFaxInfo.pCoverName );
    iAttach++;
}

////////////////////////////////////////////////////////////////////////////
// その他メール設定

MapiMessage message;
```

```
ZeroMemory(&message, sizeof(message) );
message.nFileCount = iAttach;
message.lpFiles = fileDesc;
message.lpszNoteText = "Mail to FAX";
message.lpszSubject = "Mail to FAX";

////////////////////////////////////////////////////////////////////////////
// メール送信

AfxGetApp()->EnableModeless(FALSE );
HWND hWndTop;
CWnd* pParentWnd = CWnd::GetSafeOwner(NULL, &hWndTop );

pParentWnd->SetCapture();
::SetFocus(NULL );

pParentWnd->m_nFlags |= WF_STAYDISABLED;

int nError = lpfnSendMail(
    0,
    (ULONG)pParentWnd->GetSafeHwnd(),
    &message,
    MAPI_LOGON_UI | MAPI_DIALOG,
    0 );

Sleep(500 );

::ReleaseCapture();
pParentWnd->m_nFlags &= ~WF_STAYDISABLED;

pParentWnd->EnableWindow(TRUE );
::SetActiveWindow(NULL );
pParentWnd->SetActiveWindow();
pParentWnd->SetFocus();

if(hWndTop != NULL ) {
    ::EnableWindow(hWndTop, TRUE );
}
AfxGetApp()->EnableModeless(TRUE );

if(nError != SUCCESS_SUCCESS &&
    nError != MAPI_USER_ABORT &&
    nError != MAPI_E_LOGIN_FAILURE ) {

    AfxMessageBox(AFX_IDP_FAILED_MAPI_SEND );
```

```
        }

        ////////////////////////////////////////////////////////////////////////
        // MAPI 後始末

        ::FreeLibrary(hInstMail );
    }

EXIT:

    ～

}
```

3.3.2　印刷結果のＦＡＸ送信

印刷結果のＦＡＸ送信プログラム【PrtCli.exe】 は、STARFAX Server SDK 本体をセットアップしていないパソコンで 印刷結果の表示と FAX 送信を行うサンプルプログラムです。本 CD-ROM の以下の位置に入っています。ワード・エクセル等のアプリケーションから手動で印刷後、プリンタドライバからユーザープログラムが起動されます。

¥サンプル¥応用サンプル¥クライアント¥メール de ＦＡＸ¥VC6¥PrtCli.exe
... 印刷結果のＦＡＸ送信プログラム
¥サンプル¥応用サンプル¥クライアント¥メール de ＦＡＸ¥VC6¥PrtCli¥
... 印刷結果のＦＡＸプログラム 開発プロジェクト

なお、このサンプルプログラムには、元となるサンプルプログラムがあり、FAX 送信に関する部分を[メール de ＦＡＸ]の仕組みに変更して、クライアント動作するようにしています。[メール de ＦＡＸ] 以外の処理については、「STARFAX Server SDK VC 開発向けプリンタドライバとビューア」の *2.1 印刷結果の FAX 送信* をご覧ください。

主な仕様、および操作方法は以下の通りです。
(以降、STARFAX Server SDK 本体をセットアップしているパソコンを [サーバー側]、STARFAX Server SDK 本体をセットアップしていないパソコンを [クライアント側] とします)

① [サーバー側]で [メール de ＦＡＸ] の設定を有効にします。

(1) タスクトレイの STARFAX Server SDK のアイコンを右クリックして表示されるメニューから [メール de ＦＡＸ] を実行します。

(2) 最低限、以下の項目を設定します。

- [メール de ＦＡＸ を有効にする(Y)] をチェックします。
- [メールサーバーの設定(S)] を行います。

(3) 必要に応じて、以下の項目を設定します。

- [件名に含まれる文字で識別(I)]
- [メール受信間隔(N)]
- [対象外のメールを EML ファイルに保存(P)]、及び、関連項目
- [FAX 送信結果をメール通知する(N)] 、及び、関連項目

② [サーバー側]で STARFAX Server SDK を起動します。
(STARFAX Server SDK の起動は、「STARFAX Server SDK セットアップアニュアル」P21 参照)

③ [クライアント側]で、以下のプリンタドライバの動作に関するレジストリを指定します。

- HKEY_LOCAL_MACHINE¥Software¥MEGASOFT¥SfCs¥OutFolder … ファイル出力フォルダ
  文字列項目で、任意の作業フォルダを作成して指定します。
  (例: "C:¥Program Files¥SfCs¥Temp")
- HKEY_LOCAL_MACHINE¥Software¥MEGASOFT¥SfCs¥DocName …ドキュメント名
  文字列項目で、 このサンプルプログラムの場合、任意の文字列を指定します。
  (例: "SFCSPRN")
- HKEY_LOCAL_MACHINE¥Software¥MEGASOFT¥SfCs¥ExecFlag …プログラム実行フラグ
  DWORD 項目で、 1 を指定します。
- HKEY_LOCAL_MACHINE¥Software¥MEGASOFT¥SfCs¥ExecPath …プログラムパス
  文字列項目で、 PrtCli.exe をフルパスで指定します。
  (例: "C:¥Program Files¥SfCs¥PrtCli.exe")
- HKEY_LOCAL_MACHINE¥Software¥MEGASOFT¥SfCs¥ExecParam …追加パラメータ
  文字列項目で、 何も指定していない状態("") を設定します。

④ 印刷可能な適当なアプリケーション(ワード等)から プリンタ名 "MEGASOFT STARFAX Engine" に対して印刷を行うと 印刷結果のＦＡＸ送信プログラム【PrtCli.exe】 が起動して、印刷結果リスト に 印刷結果が 登録された状態になります。
(または、印刷結果のＦＡＸ送信プログラム【PrtCli.exe】 を起動して、[テスト印刷(T)]ボタンでテスト印刷 を行っても 同じ状態になります)

⑤ [表示(V)]ボタンを押して、印刷結果の内容を確認します。

⑥ [メール送信(S)]ボタンを押して、メール送信を行います。
この後、FAX 送信が正常に動作していない場合は、[サーバー側] の以下の表示をご確認下さい。

- STARFAX Server SDK ログ管理プログラム の [イベント]
  (主に、[コントロール] と [メール] をご確認下さい)
- [メール de ＦＡＸ 設定] の システムメニュー の [イベントログ]

ApiSend.cpp ：

```
//**********************************************************************
// 関数名　　：MakeTransFileForMailToFax
// 機能　　：[ メール de ＦＡＸ ]の送信命令ファイル作成
// 呼び出し　：MakeTransFileForMailToFax(LPCTSTR pTransFile, SENDFAX_MISSION *pInfo, int *piError )
// 入力　　：pTransFile = 送信命令ファイル
//　　：pInfo　　= 送信命令ファイル作成情報ポインタ
//　　：piError　= エラー時、エラー取得用ポインタ
// 出力　　：返値　TRUE　：正常終了
//　　：　　FALSE：失敗
//　　：　　　　*piError = エラーコード
//　　：
//　　：　【エラーコード】
//　　：　　SENDFAX_ERR_GetTempFolder　... 一時フォルダの取得に失敗しました
//　　：　　SENDFAX_ERR_GetTempFile　　... 一時ファイルの取得に失敗しました
//　　：　　SENDFAX_ERR_CreateTransFile ... 送信命令ファイルの作成に失敗しました
//　　：　　SENDFAX_ERR_PARAM_INFO　　... 関数引数エラー: 送信命令ファイル作成情報が指定されていません。
//　　：　　SENDFAX_ERR_PARAM_SENDNUM　... 関数引数エラー: 相手先数の指定が 0 です。
//　　：　　SENDFAX_ERR_PARAM_SENDINFO　... 関数引数エラー: 相手先情報が指定されていません。
//　　：　　SENDFAX_ERR_PARAM_FAX　　... 関数引数エラー: FAX 番号が指定されていません。
//　　：　　SENDFAX_ERR_PARAM_TRANSFILE ... 関数引数エラー: 送信命令ファイルが指定されていません。
//　　：
// 特記事項　　：　[ メール de ＦＡＸ ]の送信命令ファイルを作成します。
// 作成者　　：
// 作成日　　：2010.07.20
//**********************************************************************
BOOL APIENTRY　MakeTransFileForMailToFax(LPCTSTR pTransFile, MAIL2FAX_MISSION *pInfo, int *piError )
{
    BOOL　bRet = FALSE;　// 返値 ... 初期値は失敗

    int iErrorDummy;　// NULL 用ダミーエラーコード設定

    int i;

    char　szGetName[MAX_PATH+1];　　// ファイル名取得用

    SENDFAX_SENDINFO　*pSendInfo; // 相手先情報用

    char　szSection[32]; // セクション設定用
    char　szWork[512];　// 作業用

    if(!piError ) {
        // エラーコード取得用ポインタにダミー設定
        piError = &iErrorDummy;
```

```
}
*piError = SENDFAX_SUCCESS;      // エラーコードに初期値設定 ... 正常終了

////////////////////////////////////////////////////////////////////////////
// (1)  簡単な引数の内容チェック
//

    // 送信命令ファイルチェック
if(!pTransFile || !(*pTransFile) ) {
    *piError = SENDFAX_ERR_PARAM_TRANSFILE;
    goto    EXIT;
}

    // 送信命令ファイル作成情報チェック
if(!pInfo ) {
    *piError = SENDFAX_ERR_PARAM_INFO;
    goto    EXIT;
}
    // 相手先数チェック
if(!pInfo->iSendNum ) {
    *piError = SENDFAX_ERR_PARAM_SENDNUM;
    goto    EXIT;
}
    // 相手先情報チェック
for(i = 0 ;  i < pInfo->iSendNum ; i++ ) {
    pSendInfo = *(pInfo->ppSend + i );
    if(!pSendInfo ) {
        *piError = SENDFAX_ERR_PARAM_SENDINFO;
        goto    EXIT;
    }
        // FAX 番号チェック
    if(!pSendInfo->szFax[0] ) {
        *piError = SENDFAX_ERR_PARAM_FAX;
        goto    EXIT;
    }
}

////////////////////////////////////////////////////////////////////////////
// (2)  送信命令ファイルへ書き込み
//

DeleteFile(pTransFile );   // 念のため削除

// セクション名: [SendInfo]  ... 送信のための相手先情報
```

```
// ・ Num        ... 送信相手先数
wsprintf(szWork, "%d", pInfo->iSendNum );
WritePrivateProfileString("SendInfo", "Num", szWork, pTransFile );

// セクション名: [Send%d]    ... 送信のための相手先内容 (1～)
for(i = 0 ;  i < pInfo->iSendNum ; i++ ) {
    pSendInfo = *(pInfo->ppSend + i );

    wsprintf(szSection, "Send%d", i + 1 );

    // ・ Fax        ... FAX 番号 (最大 128 バイト)
    WritePrivateProfileString(szSection, "Fax", pSendInfo->szFax, pTransFile );
    // ・ Company    ... 会社名 (最大 128 バイト)
    if(pSendInfo->szCompany[0] ) {
        wsprintf(szWork, "¥"%s¥"", pSendInfo->szCompany );
        WritePrivateProfileString(szSection, "Company", szWork, pTransFile );
    }
    // ・ Division   ... 所属名 (最大 128 バイト)
    if(pSendInfo->szDivision[0] ) {
        wsprintf(szWork, "¥"%s¥"", pSendInfo->szDivision );
        WritePrivateProfileString(szSection, "Division", szWork, pTransFile );
    }
    // ・ Position   ... 役職名 (最大 128 バイト)
    if(pSendInfo->szPosition[0] ) {
        wsprintf(szWork, "¥"%s¥"", pSendInfo->szPosition );
        WritePrivateProfileString(szSection, "Position", szWork, pTransFile );
    }
    // ・ Name       ... 氏名 (最大 128 バイト)
    if(pSendInfo->szName[0] ) {
        wsprintf(szWork, "¥"%s¥"", pSendInfo->szName );
        WritePrivateProfileString(szSection, "Name", szWork, pTransFile );
    }
    // ・ Title      ... 敬称 (最大 128 バイト)
    if(pSendInfo->szTitle[0] ) {
        wsprintf(szWork, "¥"%s¥"", pSendInfo->szTitle );
        WritePrivateProfileString(szSection, "Title", szWork, pTransFile );
    }
    // ・ Telephone  ... 電話番号 (最大 128 バイト)
    if(pSendInfo->szTelephone[0] ) {
        WritePrivateProfileString(szSection, "Telephone", pSendInfo->szTelephone, pTransFile );
    }
    // ・ ZipCode    ... 郵便番号 (最大 128 バイト)
    if(pSendInfo->szZipCode[0] ) {
        WritePrivateProfileString(szSection, "ZipCode", pSendInfo->szZipCode, pTransFile );
    }
```

```
// ・ Address1   ... 住所1 (最大128バイト)
if(pSendInfo->szAddress1[0] ) {
    wsprintf(szWork, "¥"%s¥"", pSendInfo->szAddress1 );
    WritePrivateProfileString(szSection, "Address1", szWork, pTransFile );
}
// ・ Address2   ... 住所2 (最大128バイト)
if(pSendInfo->szAddress2[0] ) {
    wsprintf(szWork, "¥"%s¥"", pSendInfo->szAddress2 );
    WritePrivateProfileString(szSection, "Address2", szWork, pTransFile );
}
// ・ FCode      ... Fコード番号 (最大20バイト)
if(pSendInfo->szFCode[0] ) {
    wsprintf(szWork, "¥"%s¥"", pSendInfo->szFCode );
    WritePrivateProfileString(szSection, "FCode", szWork, pTransFile );
}
// ・ FreeArea   ... ユーザが自由に利用できるエリア (最大128バイト)
if(pSendInfo->szFreeArea[0] ) {
    wsprintf(szWork, "¥"%s¥"", pSendInfo->szFreeArea );
    WritePrivateProfileString(szSection, "FreeArea", szWork, pTransFile );
}
// ・ Speed      ... 通信速度 ("0":自動, "1":高速, "2":中速, "3":低速)
if(pSendInfo->iSpeed ) {
    wsprintf(szWork, "%d", pSendInfo->iSpeed );
    WritePrivateProfileString(szSection, "Speed", szWork, pTransFile );
}
// ・ Comp       ... 圧縮方式 ("0":自動, "1":MH, "2":MR, "3":MMR)
if(pSendInfo->iComp ) {
    wsprintf(szWork, "%d", pSendInfo->iComp );
    WritePrivateProfileString(szSection, "Comp", szWork, pTransFile );
}
// ・ Ecm        ... エラー訂正 ("0":自動, "1":利用する, "2":利用しない)
if(pSendInfo->iEcm ) {
    wsprintf(szWork, "%d", pSendInfo->iEcm );
    WritePrivateProfileString(szSection, "Ecm", szWork, pTransFile );
}
// ・ Line       ... 送信回線指定 (0～)
if(pSendInfo->bLine ) {
    wsprintf(szWork, "%d", pSendInfo->iLine );
    WritePrivateProfileString(szSection, "Line", szWork, pTransFile );
}
// ・ Priority   ... 優先順位 (0～15)
if(pSendInfo->bPriority ) {
    wsprintf(szWork, "%d", pSendInfo->iPriority );
    WritePrivateProfileString(szSection, "Priority", szWork, pTransFile );
}
```

```
    // ・ Time        ... 送信時刻 ("YYYYMMDDHHMMSS")
    if(pSendInfo->bTime ) {
        wsprintf(szWork, "%s", pSendInfo->szTime );
        WritePrivateProfileString(szSection, "Time", szWork, pTransFile );
    }
}

// セクション名: [Cover]     ... 送付状
// ・ Name        ... 送付状ファイルパス(.txt)
if(pInfo->pCoverName ) {

    SUB_GetFileName(pInfo->pCoverName, szGetName );

    wsprintf(szWork, "¥"%s¥"", szGetName );
    WritePrivateProfileString("Cover", "Name", szWork, pTransFile );
}
// ・ FontName   ... 送付状フォント名
if(pInfo->pCoverFontName ) {
    wsprintf(szWork, "¥"%s¥"", pInfo->pCoverFontName );
    WritePrivateProfileString("Cover", "FontName", szWork, pTransFile );
}
// ・ FontSize   ... 送付状フォントサイズ (8 ～ 72 (ポイント ) )
if(pInfo->iCoverFontSize ) {
    wsprintf(szWork, "%d", pInfo->iCoverFontSize );
    WritePrivateProfileString("Cover", "FontSize", szWork, pTransFile );
}

// セクション名: [User]       ... 発信元情報
// ・ UserInfo   ... 発信元情報記録 ("記録位置,記録情報") (最大 140 バイト)
//                    ("0":記録しない, "1":原稿の内側に記録, "2":原稿の外側に記録)
if(pInfo->szUserInfo[0] ) {
    wsprintf(szWork, "¥"%s¥"", pInfo->szUserInfo );
    WritePrivateProfileString("User", "UserInfo", szWork, pTransFile );
}
// ・ UserID     ... 自局電話番号 (FAXID として、相手ファクシミリに通知されます。) (最大 20 バイト)
if(pInfo->szUserID[0] ) {
    wsprintf(szWork, "¥"%s¥"", pInfo->szUserID );
    WritePrivateProfileString("User", "UserID", szWork, pTransFile );
}

////////////////////////////////////////////////////////////////////////////
// (3) 送信命令ファイルのフラッシュ 【重要】
//
//      OS により、WritePrivateProfileString() 【Win32API】 を使用しての
//    実際のファイルへの書き出しのタイミングが異なる為、必ずこのフラッシュの
```

```
    //    ステップが必要です。
    //

    WritePrivateProfileString(NULL, NULL, NULL, pTransFile );

    bRet = TRUE;    // 正常終了

EXIT:

    // エラー時は、送信命令ファイルを削除
    if(!bRet ) {
        DeleteFile(pTransFile );
    }

    return bRet;
}
```

PrtCliDlg.cpp ：

```
////////////////////////////////////////////////////////////////////////////
// メール送信ボタン

void CPrtCliDlg::OnOK()
{

    ～

    //――――――――――――――――――――――――――――――――――――――――
    // (2) [ メール de ＦＡＸ ]の送信命令ファイル作成関数 呼び出し

    if(!(bRet = MakeTransFileForMailToFax(szTrans, &MailToFaxInfo, &iError ) ) ) {
        switch(iError ) {
        case    SENDFAX_ERR_GetTempFolder:
            MessageBox("一時フォルダの取得に失敗しました", CAP_APR_PRTCLI, MB_ICONEXCLAMATION | MB_OK );
            break;
        case    SENDFAX_ERR_GetTempFile:
            MessageBox("一時ファイルの取得に失敗しました", CAP_APR_PRTCLI, MB_ICONEXCLAMATION | MB_OK );
            break;
        case    SENDFAX_ERR_CreateTransFile:
            MessageBox("送信命令ファイルの作成に失敗しました", CAP_APR_PRTCLI, MB_ICONEXCLAMATION | MB_OK );
            break;
        case    SENDFAX_ERR_PARAM_INFO:
            MessageBox("関数引数エラー: 送信命令ファイル作成情報が指定されていません。", CAP_APR_PRTCLI,
MB_ICONEXCLAMATION | MB_OK );
            break;
        case    SENDFAX_ERR_PARAM_SENDNUM:
            MessageBox(" 関数引数エラー: 相手先数の指定が 0 です。", CAP_APR_PRTCLI, MB_ICONEXCLAMATION | MB_OK );
            break;
        case    SENDFAX_ERR_PARAM_SENDINFO:
            MessageBox("関数引数エラー: 相手先情報が指定されていません。", CAP_APR_PRTCLI, MB_ICONEXCLAMATION | MB_OK );
            break;
        case    SENDFAX_ERR_PARAM_FAX:
            MessageBox("関数引数エラー: FAX 番号が指定されていません。", CAP_APR_PRTCLI, MB_ICONEXCLAMATION | MB_OK );
            break;
        case    SENDFAX_ERR_PARAM_TRANSFILE:
            MessageBox("関数引数エラー: 送信命令ファイルが指定されていません。", CAP_APR_PRTCLI, MB_ICONEXCLAMATION |
MB_OK );
            break;
        default:
            break;
        }
```

```
}
else {
```

MessageBox("このサンプルプログラムは MAPI を利用してメール送信を行います。¥n 宛先 に STARFAX Server SDK の [メール de ＦＡＸ] で設定した¥nPOP メールアドレス を指定して送信してください。¥n¥n なお、送信後、OUTLOOK 等メーラーを起動して送信を行わないと¥n 実際に送信されないことがあります。", CAP_APR_PRTCLI, MB_ICONINFORMATION | MB_OK);

```
    ////////////////////////////////////////////////////////////////////////////
    //
    // メール送信
    //
    ////////////////////////////////////////////////////////////////////////////

    ////////////////////////////////////////////////////////////////////////////
    // MAPI 初期化

    HINSTANCE hInstMail = ::LoadLibraryA("MAPI32.DLL" );

    if(hInstMail == NULL ) {
        AfxMessageBox(AFX_IDP_FAILED_MAPI_LOAD );
        goto    EXIT;
    }

    ULONG (PASCAL *lpfnSendMail )(ULONG, ULONG, MapiMessage*, FLAGS, ULONG );
    (FARPROC&)lpfnSendMail = GetProcAddress(hInstMail, "MAPISendMail" );
    if(lpfnSendMail == NULL ) {
        AfxMessageBox(AFX_IDP_INVALID_MAPI_DLL );
        ::FreeLibrary(hInstMail );
        goto    EXIT;
    }

    ////////////////////////////////////////////////////////////////////////////
    // 添付ファイル設定

    MapiFileDesc    *pFileDesc;

    pFileDesc = new MapiFileDesc[m_clPRT.GetSelectedCount() + 1];

    int iAttach = 0;

    // 送信命令ファイル
    ZeroMemory(&pFileDesc[iAttach], sizeof(MapiFileDesc) );
    pFileDesc[iAttach].nPosition = iAttach;
    pFileDesc[iAttach].lpszPathName = szTrans;
```

```
pFileDesc[iAttach].lpszFileName = pSUB_GetFileNamePtr(szTrans );
iAttach++;

POSITION pos = m_clPRT.GetFirstSelectedItemPosition();
while(pos ) {
    int nItem = m_clPRT.GetNextSelectedItem(pos );
    PRES_DATINFO    *pItem = (PRES_DATINFO *)(m_clPRT.GetItemData(nItem )) ;
    if(pItem ) {
        ZeroMemory(&pFileDesc[iAttach], sizeof(MapiFileDesc) );
        pFileDesc[iAttach].nPosition = iAttach;
        pFileDesc[iAttach].lpszPathName = pItem->pTiffFile;
        pFileDesc[iAttach].lpszFileName = pSUB_GetFileNamePtr(pItem->pTiffFile );

        iAttach++;
    }
}

////////////////////////////////////////////////////////////////////////////
// その他メール設定

MapiMessage message;

ZeroMemory(&message, sizeof(message) );
message.nFileCount = iAttach;
message.lpFiles = pFileDesc;
message.lpszNoteText = "Mail to FAX";
message.lpszSubject = "Mail to FAX";

////////////////////////////////////////////////////////////////////////////
// メール送信

AfxGetApp()->EnableModeless(FALSE );
HWND hWndTop;
CWnd* pParentWnd = CWnd::GetSafeOwner(NULL, &hWndTop );

pParentWnd->SetCapture();
::SetFocus(NULL );

pParentWnd->m_nFlags |= WF_STAYDISABLED;

int nError = lpfnSendMail(
    0,
    (ULONG)pParentWnd->GetSafeHwnd(),
    &message,
    MAPI_LOGON_UI | MAPI_DIALOG,
```

```
            0 );

        Sleep(500 );

        ::ReleaseCapture();
        pParentWnd->m_nFlags &= ~WF_STAYDISABLED;

        pParentWnd->EnableWindow(TRUE );
        ::SetActiveWindow(NULL );
        pParentWnd->SetActiveWindow();
        pParentWnd->SetFocus();

        if(hWndTop != NULL ) {
            ::EnableWindow(hWndTop, TRUE );
        }
        AfxGetApp()->EnableModeless(TRUE );

        if(nError != SUCCESS_SUCCESS &&
            nError != MAPI_USER_ABORT &&
            nError != MAPI_E_LOGIN_FAILURE ) {

            AfxMessageBox(AFX_IDP_FAILED_MAPI_SEND );
        }

        delete pFileDesc;

        ////////////////////////////////////////////////////////////////////////////
        // MAPI 後始末

        ::FreeLibrary(hInstMail );
    }

EXIT:

    ～

}
```

3.3.3 TIFF ファイルの作成とＦＡＸ送信

TIFF ファイルのＦＡＸ送信プログラム【PrtCli2.exe】 は、STARFAX Server SDK 本体をセットアップしていないパソコンで TIFF ファイルの作成 と FAX 送信を行うサンプルプログラムです。本 CD-ROM の以下の位置に入っています。ユーザープログラムがプリンタドライバを制御して印刷結果(TIFF ファイル)を取得します。

¥サンプル¥応用サンプル¥クライアント¥メール de ＦＡＸ¥VC6¥PrtCli2.exe
... TIFF ファイルのＦＡＸ送信プログラム
¥サンプル¥応用サンプル¥クライアント¥メール de ＦＡＸ¥VC6¥PrtCli2¥
... TIFF ファイルのＦＡＸ送信プログラム 開発プロジェクト

なお、このサンプルプログラムには、元となるサンプルプログラムがあり、FAX 送信に関する部分を[メール de ＦＡＸ]の仕組みに変更して、クライアント動作するようにしています。[メール to ＦＸ] 以外の処理については、「STARFAX Server SDK VC 開発向けプリンタドライバとビューア」の *2.2 TIFF ファイルの作成と FAX 送信* をご覧ください。

主な仕様、および操作方法は以下の通りです。
(以降、STARFAX Server SDK 本体をセットアップしているパソコンを [サーバー側]、STARFAX Server SDK 本体をセットアップしていないパソコンを [クライアント側] とします)

① [サーバー側]で [メール de ＦＡＸ] の設定を有効にします。

(1) タスクトレイの STARFAX Server SDK のアイコンを右クリックして表示されるメニューから [メール de ＦＡＸ] を実行します。

(2) 最低限、以下の項目を設定します。

- [メール de ＦＡＸ を有効にする(Y)] をチェックします。
- [メールサーバーの設定(S)] を行います。

(3) 必要に応じて、以下の項目を設定します。

- [件名に含まれる文字で識別(I)]
- [メール受信間隔(N)]
- [対象外のメールを EML ファイルに保存(P)]、及び、関連項目
- [FAX 送信結果をメール通知する(N)] 、及び、関連項目

② [サーバー側]で STARFAX Server SDK を起動します。
(STARFAX Server SDK の起動は、「STARFAX Server SDK セットアップアニュアル」P21 参照)

③ [クライアント側]で PrtCli2.exe を起動します。

④ 「操作１」の下欄に FAX 原稿に表示させる文字を入力します。

⑤ 「①FAX 原稿の作成」ボタンをクリックします。
(作成するファイル名を任意指定したい場合は、[指定]ラジオボタンをクリックして、ファイル名を入力してください。)

⑥ 「②FAX 原稿の表示」ボタンをクリックすると、作成された FAX 原稿がビューアで表示されます。

⑦ 「操作３」の下欄に送信先の FAX 番号を入力します。

⑧ 「メール送信」ボタンをクリックします。
この後、FAX 送信が正常に動作していない場合は、[サーバー側] の以下の表示をご確認下さい。

- STARFAX Server SDK ログ管理プログラム の [イベント]
  (主に、[コントロール] と [メール] をご確認下さい)
- [メール de ＦＡＸ 設定] の システムメニュー の [イベントログ]

ApiSend.cpp ：

```
//*********************************************************************
// 関数名　　：MakeTransFileForMailToFax
// 機能　　：[ メール de ＦＡＸ ]の送信命令ファイル作成
// 呼び出し　：MakeTransFileForMailToFax(LPCTSTR pTransFile, SENDFAX_MISSION *pInfo, int *piError )
// 入力　　：pTransFile = 送信命令ファイル
//　　：pInfo　= 送信命令ファイル作成情報ポインタ
//　　：piError　= エラー時、エラー取得用ポインタ
// 出力　　：返値　TRUE　：正常終了
//　　：　FALSE：失敗
//　　：　　*piError = エラーコード
//　　：
//　　：　【エラーコード】
//　　：　　SENDFAX_ERR_GetTempFolder　... 一時フォルダの取得に失敗しました
//　　：　　SENDFAX_ERR_GetTempFile　... 一時ファイルの取得に失敗しました
//　　：　　SENDFAX_ERR_CreateTransFile ... 送信命令ファイルの作成に失敗しました
//　　：　　SENDFAX_ERR_PARAM_INFO　... 関数引数エラー: 送信命令ファイル作成情報が指定されていません。
//　　：　　SENDFAX_ERR_PARAM_SENDNUM　... 関数引数エラー: 相手先数の指定が 0 です。
//　　：　　SENDFAX_ERR_PARAM_SENDINFO　... 関数引数エラー: 相手先情報が指定されていません。
//　　：　　SENDFAX_ERR_PARAM_FAX　... 関数引数エラー: FAX 番号が指定されていません。
//　　：　　SENDFAX_ERR_PARAM_TRANSFILE ... 関数引数エラー: 送信命令ファイルが指定されていません。
//　　：
// 特記事項　　：　[ メール de ＦＡＸ ]の送信命令ファイルを作成します。
// 作成者　　：
// 作成日　　：2010.07.20
//*********************************************************************
BOOL APIENTRY　MakeTransFileForMailToFax(LPCTSTR pTransFile, MAIL2FAX_MISSION *pInfo, int *piError )
{
    BOOL　bRet = FALSE;　// 返値 ... 初期値は失敗

    int iErrorDummy;　// NULL 用ダミーエラーコード設定

    int i;

    char　szGetName[MAX_PATH+1];　　// ファイル名取得用

    SENDFAX_SENDINFO　*pSendInfo; // 相手先情報用

    char　szSection[32]; // セクション設定用
    char　szWork[512];　// 作業用

    if(!piError ) {
        // エラーコード取得用ポインタにダミー設定
        piError = &iErrorDummy;
```

```
}
*piError = SENDFAX_SUCCESS;      // エラーコードに初期値設定 ... 正常終了

////////////////////////////////////////////////////////////////////////////
// (1)  簡単な引数の内容チェック
//

    // 送信命令ファイルチェック
if(!pTransFile || !(*pTransFile) ) {
    *piError = SENDFAX_ERR_PARAM_TRANSFILE;
    goto    EXIT;
}

    // 送信命令ファイル作成情報チェック
if(!pInfo ) {
    *piError = SENDFAX_ERR_PARAM_INFO;
    goto    EXIT;
}
    // 相手先数チェック
if(!pInfo->iSendNum ) {
    *piError = SENDFAX_ERR_PARAM_SENDNUM;
    goto    EXIT;
}
    // 相手先情報チェック
for(i = 0 ;  i < pInfo->iSendNum ; i++ ) {
    pSendInfo = *(pInfo->ppSend + i );
    if(!pSendInfo ) {
        *piError = SENDFAX_ERR_PARAM_SENDINFO;
        goto    EXIT;
    }
        // FAX 番号チェック
    if(!pSendInfo->szFax[0] ) {
        *piError = SENDFAX_ERR_PARAM_FAX;
        goto    EXIT;
    }
}

////////////////////////////////////////////////////////////////////////////
// (2)  送信命令ファイルへ書き込み
//

DeleteFile(pTransFile );   // 念のため削除

// セクション名: [SendInfo]  ... 送信のための相手先情報
```

```
// ・ Num        ... 送信相手先数
wsprintf(szWork, "%d", pInfo->iSendNum );
WritePrivateProfileString("SendInfo", "Num", szWork, pTransFile );

// セクション名: [Send%d]    ... 送信のための相手先内容 (1～)
for(i = 0 ;  i < pInfo->iSendNum ; i++ ) {
    pSendInfo = *(pInfo->ppSend + i );

    wsprintf(szSection, "Send%d", i + 1 );

    // ・ Fax        ... FAX 番号 (最大 128 バイト)
    WritePrivateProfileString(szSection, "Fax", pSendInfo->szFax, pTransFile );
    // ・ Company    ... 会社名 (最大 128 バイト)
    if(pSendInfo->szCompany[0] ) {
        wsprintf(szWork, "¥"%s¥"", pSendInfo->szCompany );
        WritePrivateProfileString(szSection, "Company", szWork, pTransFile );
    }
    // ・ Division   ... 所属名 (最大 128 バイト)
    if(pSendInfo->szDivision[0] ) {
        wsprintf(szWork, "¥"%s¥"", pSendInfo->szDivision );
        WritePrivateProfileString(szSection, "Division", szWork, pTransFile );
    }
    // ・ Position   ... 役職名 (最大 128 バイト)
    if(pSendInfo->szPosition[0] ) {
        wsprintf(szWork, "¥"%s¥"", pSendInfo->szPosition );
        WritePrivateProfileString(szSection, "Position", szWork, pTransFile );
    }
    // ・ Name       ... 氏名 (最大 128 バイト)
    if(pSendInfo->szName[0] ) {
        wsprintf(szWork, "¥"%s¥"", pSendInfo->szName );
        WritePrivateProfileString(szSection, "Name", szWork, pTransFile );
    }
    // ・ Title      ... 敬称 (最大 128 バイト)
    if(pSendInfo->szTitle[0] ) {
        wsprintf(szWork, "¥"%s¥"", pSendInfo->szTitle );
        WritePrivateProfileString(szSection, "Title", szWork, pTransFile );
    }
    // ・ Telephone  ... 電話番号 (最大 128 バイト)
    if(pSendInfo->szTelephone[0] ) {
        WritePrivateProfileString(szSection, "Telephone", pSendInfo->szTelephone, pTransFile );
    }
    // ・ ZipCode    ... 郵便番号 (最大 128 バイト)
    if(pSendInfo->szZipCode[0] ) {
        WritePrivateProfileString(szSection, "ZipCode", pSendInfo->szZipCode, pTransFile );
    }
```

```
// ・ Address1   ... 住所1 (最大128バイト)
if(pSendInfo->szAddress1[0] ) {
    wsprintf(szWork, "¥"%s¥"", pSendInfo->szAddress1 );
    WritePrivateProfileString(szSection, "Address1", szWork, pTransFile );
}
// ・ Address2   ... 住所2 (最大128バイト)
if(pSendInfo->szAddress2[0] ) {
    wsprintf(szWork, "¥"%s¥"", pSendInfo->szAddress2 );
    WritePrivateProfileString(szSection, "Address2", szWork, pTransFile );
}
// ・ FCode      ... Fコード番号 (最大20バイト)
if(pSendInfo->szFCode[0] ) {
    wsprintf(szWork, "¥"%s¥"", pSendInfo->szFCode );
    WritePrivateProfileString(szSection, "FCode", szWork, pTransFile );
}
// ・ FreeArea   ... ユーザが自由に利用できるエリア (最大128バイト)
if(pSendInfo->szFreeArea[0] ) {
    wsprintf(szWork, "¥"%s¥"", pSendInfo->szFreeArea );
    WritePrivateProfileString(szSection, "FreeArea", szWork, pTransFile );
}
// ・ Speed      ... 通信速度 ("0":自動, "1":高速, "2":中速, "3":低速)
if(pSendInfo->iSpeed ) {
    wsprintf(szWork, "%d", pSendInfo->iSpeed );
    WritePrivateProfileString(szSection, "Speed", szWork, pTransFile );
}
// ・ Comp       ... 圧縮方式 ("0":自動, "1":MH, "2":MR, "3":MMR)
if(pSendInfo->iComp ) {
    wsprintf(szWork, "%d", pSendInfo->iComp );
    WritePrivateProfileString(szSection, "Comp", szWork, pTransFile );
}
// ・ Ecm        ... エラー訂正 ("0":自動, "1":利用する, "2":利用しない)
if(pSendInfo->iEcm ) {
    wsprintf(szWork, "%d", pSendInfo->iEcm );
    WritePrivateProfileString(szSection, "Ecm", szWork, pTransFile );
}
// ・ Line       ... 送信回線指定 (0～)
if(pSendInfo->bLine ) {
    wsprintf(szWork, "%d", pSendInfo->iLine );
    WritePrivateProfileString(szSection, "Line", szWork, pTransFile );
}
// ・ Priority   ... 優先順位 (0～15)
if(pSendInfo->bPriority ) {
    wsprintf(szWork, "%d", pSendInfo->iPriority );
    WritePrivateProfileString(szSection, "Priority", szWork, pTransFile );
}
```

```
    // ・ Time        ... 送信時刻 ("YYYYMMDDHHMMSS")
    if(pSendInfo->bTime ) {
        wsprintf(szWork, "%s", pSendInfo->szTime );
        WritePrivateProfileString(szSection, "Time", szWork, pTransFile );
    }
}

// セクション名: [Cover]     ... 送付状
// ・ Name        ... 送付状ファイルパス(.txt)
if(pInfo->pCoverName ) {

    SUB_GetFileName(pInfo->pCoverName, szGetName );

    wsprintf(szWork, "¥"%s¥"", szGetName );
    WritePrivateProfileString("Cover", "Name", szWork, pTransFile );
}
// ・ FontName   ... 送付状フォント名
if(pInfo->pCoverFontName ) {
    wsprintf(szWork, "¥"%s¥"", pInfo->pCoverFontName );
    WritePrivateProfileString("Cover", "FontName", szWork, pTransFile );
}
// ・ FontSize   ... 送付状フォントサイズ (8 ～ 72 (ポイント ) )
if(pInfo->iCoverFontSize ) {
    wsprintf(szWork, "%d", pInfo->iCoverFontSize );
    WritePrivateProfileString("Cover", "FontSize", szWork, pTransFile );
}

// セクション名: [User]       ... 発信元情報
// ・ UserInfo   ... 発信元情報記録 ("記録位置,記録情報") (最大 140 バイト)
//                    ("0":記録しない, "1":原稿の内側に記録, "2":原稿の外側に記録)
if(pInfo->szUserInfo[0] ) {
    wsprintf(szWork, "¥"%s¥"", pInfo->szUserInfo );
    WritePrivateProfileString("User", "UserInfo", szWork, pTransFile );
}
// ・ UserID     ... 自局電話番号 (FAXID として、相手ファクシミリに通知されます。) (最大 20 バイト)
if(pInfo->szUserID[0] ) {
    wsprintf(szWork, "¥"%s¥"", pInfo->szUserID );
    WritePrivateProfileString("User", "UserID", szWork, pTransFile );
}

////////////////////////////////////////////////////////////////////////////
// (3) 送信命令ファイルのフラッシュ 【重要】
//
//      OS により、WritePrivateProfileString() 【Win32API】 を使用しての
//    実際のファイルへの書き出しのタイミングが異なる為、必ずこのフラッシュの
```

```
    //    ステップが必要です。
    //

    WritePrivateProfileString(NULL, NULL, NULL, pTransFile );

    bRet = TRUE;    // 正常終了

EXIT:

    // エラー時は、送信命令ファイルを削除
    if(!bRet ) {
        DeleteFile(pTransFile );
    }

    return bRet;
}
```

PrtCli2Dlg.cpp ：

```
//////////////////////////////////////////////////////////////////////////////
// [メール送信]ボタン

void CPrtCli2Dlg::OnButtonFax()
{

    ～

    //----------------------------------------------------------
    // (2) [ メール de ＦＡＸ ]の送信命令ファイル作成関数 呼び出し

    if(!(bRet = MakeTransFileForMailToFax(szTrans,  &MailToFaxInfo, &iError ) ) ) {
        switch(iError ) {
        case    SENDFAX_ERR_GetTempFolder:
            MessageBox("一時フォルダの取得に失敗しました", CAP_APR_PRTCLI2, MB_ICONEXCLAMATION | MB_OK );
            break;
        case    SENDFAX_ERR_GetTempFile:
            MessageBox("一時ファイルの取得に失敗しました", CAP_APR_PRTCLI2, MB_ICONEXCLAMATION | MB_OK );
            break;
        case    SENDFAX_ERR_CreateTransFile:
            MessageBox("送信命令ファイルの作成に失敗しました", CAP_APR_PRTCLI2, MB_ICONEXCLAMATION | MB_OK );
            break;
        case    SENDFAX_ERR_PARAM_INFO:
            MessageBox("関数引数エラー: 送信命令ファイル作成情報が指定されていません。", CAP_APR_PRTCLI2,
MB_ICONEXCLAMATION | MB_OK );
            break;
        case    SENDFAX_ERR_PARAM_SENDNUM:
            MessageBox(" 関数引数エラー: 相手先数の指定が 0 です。", CAP_APR_PRTCLI2, MB_ICONEXCLAMATION | MB_OK );
            break;
        case    SENDFAX_ERR_PARAM_SENDINFO:
            MessageBox("関数引数エラー: 相手先情報が指定されていません。", CAP_APR_PRTCLI2, MB_ICONEXCLAMATION | MB_OK );
            break;
        case    SENDFAX_ERR_PARAM_FAX:
            MessageBox("関数引数エラー: FAX 番号が指定されていません。", CAP_APR_PRTCLI2, MB_ICONEXCLAMATION | MB_OK );
            break;
        case    SENDFAX_ERR_PARAM_TRANSFILE:
            MessageBox("関数引数エラー: 送信命令ファイルが指定されていません。", CAP_APR_PRTCLI2, MB_ICONEXCLAMATION |
MB_OK );
            break;
        default:
            break;
        }
```

```
}
else {
```

MessageBox(″このサンプルプログラムは MAPI を利用してメール送信を行います。¥n 宛先 に STARFAX Server SDK の [メール de ＦＡＸ] で設定した¥nPOP メールアドレス を指定して送信してください。¥n¥n なお、送信後、OUTLOOK 等メーラーを起動して送信を行わないと¥n 実際に送信されないことがあります。″, CAP_APR_PRTCLI2, MB_ICONINFORMATION | MB_OK);

```
////////////////////////////////////////////////////////////////////////////
//
// メール送信
//
////////////////////////////////////////////////////////////////////////////

////////////////////////////////////////////////////////////////////////////
// MAPI 初期化

HINSTANCE hInstMail = ::LoadLibraryA("MAPI32.DLL" );

if(hInstMail == NULL ) {
    AfxMessageBox(AFX_IDP_FAILED_MAPI_LOAD );
    goto    EXIT;
}

ULONG (PASCAL *lpfnSendMail )(ULONG, ULONG, MapiMessage*, FLAGS, ULONG );
(FARPROC&)lpfnSendMail = GetProcAddress(hInstMail, "MAPISendMail" );
if(lpfnSendMail == NULL ) {
    AfxMessageBox(AFX_IDP_INVALID_MAPI_DLL );
    ::FreeLibrary(hInstMail );
    goto    EXIT;
}

////////////////////////////////////////////////////////////////////////////
// 添付ファイル設定

MapiFileDesc    fileDesc[2];

int iAttach = 0;

// 送信命令ファイル
ZeroMemory(&fileDesc[iAttach], sizeof(MapiFileDesc) );
fileDesc[iAttach].nPosition = iAttach;
fileDesc[iAttach].lpszPathName = szTrans;
fileDesc[iAttach].lpszFileName = pSUB_GetFileNamePtr(szTrans );
iAttach++;
```

```
ZeroMemory(&fileDesc[iAttach], sizeof(MapiFileDesc) );
fileDesc[iAttach].nPosition = iAttach;
fileDesc[iAttach].lpszPathName = m_szTIFFFILE;
fileDesc[iAttach].lpszFileName = pSUB_GetFileNamePtr(m_szTIFFFILE );
iAttach++;

////////////////////////////////////////////////////////////////////////////
// その他メール設定

MapiMessage message;

ZeroMemory(&message, sizeof(message) );
message.nFileCount = iAttach;
message.lpFiles = fileDesc;
message.lpszNoteText = "Mail to FAX";
message.lpszSubject = "Mail to FAX";

////////////////////////////////////////////////////////////////////////////
// メール送信

AfxGetApp()->EnableModeless(FALSE );
HWND hWndTop;
CWnd* pParentWnd = CWnd::GetSafeOwner(NULL, &hWndTop );

pParentWnd->SetCapture();
::SetFocus(NULL );

pParentWnd->m_nFlags |= WF_STAYDISABLED;

int nError = lpfnSendMail(
    0,
    (ULONG)pParentWnd->GetSafeHwnd(),
    &message,
    MAPI_LOGON_UI | MAPI_DIALOG,
    0 );

Sleep(500 );

::ReleaseCapture();
pParentWnd->m_nFlags &= ~WF_STAYDISABLED;

pParentWnd->EnableWindow(TRUE );
::SetActiveWindow(NULL );
pParentWnd->SetActiveWindow();
```

```
        pParentWnd->SetFocus();

        if(hWndTop != NULL ) {
            ::EnableWindow(hWndTop, TRUE );
        }
        AfxGetApp()->EnableModeless(TRUE );

        if(nError != SUCCESS_SUCCESS &&
            nError != MAPI_USER_ABORT &&
            nError != MAPI_E_LOGIN_FAILURE ) {

            AfxMessageBox(AFX_IDP_FAILED_MAPI_SEND );
        }

        ////////////////////////////////////////////////////////////////////////
        // MAPI 後始末

        ::FreeLibrary(hInstMail );
    }

EXIT:

    ～

}
```