TDC通信パッケージ(Linux)
開発ガイド

第1版

初版　2023年08月31日

目次

本書について

本書は、TDC通信パッケージ(Linux)を使用するユーザーの支援を目的として
書かれています。
本書には、TDC通信パッケージ(Linux)で提供される汎用プログラミング・
インターフェース、及び、それに関連する情報が掲載されています。

対象読者

本書は、TDC通信パッケージ開発環境(Linux)を導入し、それをC言語関数の呼び出しをサポートする
他の開発ツールとともに使用して、TDCのサービスを利用するアプリケーションの開発または
強化を計画している方を対象にしています。

表記上の規則

本書では、以下の表記上の規則を使用しています。

機能名
　機能名は、各ページの一番上に記され、その下に簡単な説明があります。

書式
　各書式にはパラメータ変数、パラメータ・タイプ、およびパラメータ・制限が付記されています。
　　・パラメータ変数はすべて小文字で記されています。
　　・パラメータのタイプには次の3種類があります。
　　入力 （I）
　　出力 （O）
　　入出力（I／O）
　　・パラメータの制限には次の2種類があります。
　　必須
　　任意

パラメーター
　各パラメーターにはC言語データ・タイプ、桁数、表記方法，および簡単な説明が付記されています。
　　・データ・タイプは文字または文字列で定義します。
　　文字 （char）
　　文字列（char*）
　　・桁数は各機能が有効とみなす桁数です。よって指定桁数を超えた値は無視されます。
　　・各パラメーターについて表記方法および簡単な説明があります。該当するものについては、
　　制約事項も付記されています。

戻り
　ここには、各機能の正常および異常の判定のみ戻されます。
　詳細なエラー内容については、result_val（処理結果パラメーター）に設定されます。

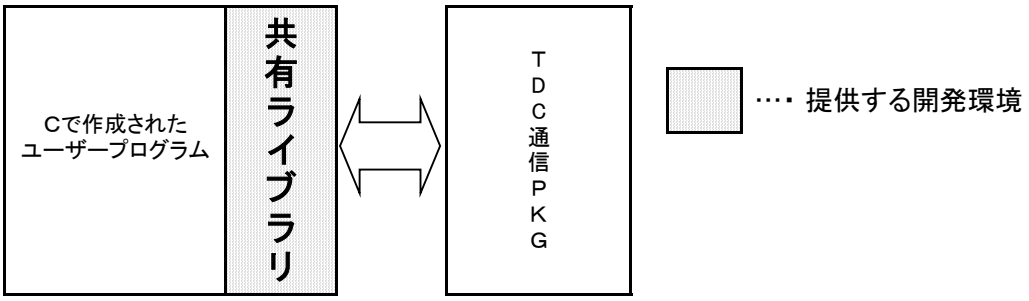
解説
　ここでは、必要に応じてその機能についての追加説明を行っています。

第1部　概説

第1章　概説

TDC通信パッケージ(Linux)には、取引先間のメッセージ転送を容易に行える開発環境を
備えており、それらを使用することによって、ユーザー独自のクライアント・サーバー・アプリケーションを
作成することができます。GCC などの開発言語を利用し、TDCシステムのサービスを容易に
組み込むことが出来ます。

TDC通信パッケージ開発環境(Linux)の位置付け



技術情報

TDC通信パッケージ開発環境(Linux)は、下記にリストされている製品のいづれかとともに
使用されます。

ソフトウェア環境

TDC通信パッケージ開発環境(Linux)に必要な操作環境は、次のとおりです。

・Red Hat Enterprise Linux release 8.6 (Ootpa)

TDC通信パッケージ開発環境(Linux)を使用してアプリケーションを開発する際に必要となる
開発ソフトウェアは、次のいづれかです。

・共有ライブラリへの呼出しをサポートする何らかのツール
・gcc バージョン 8.5.0 20210514 (Red Hat 8.5.0-10) (GCC)

第2部　TDC通信パッケージ(Linux) の 関数

第2章　関数の要約

関数の要約

TDC通信パッケージ開発環境(Linux)は、TDC通信パッケージ(Linux)の通信処理を制御し、
それと対話するための関数をサポートしています。次のリストは、カテゴリー別に全関数を示したものです。

※「TDC通信パッケージ(Linux)」ではファイル送受信時のファイルフォーマットの中で
「TNS拡張形式（可変長）」は保障対象外となりますのでご注意ください。

ファイル送信関数

これらの関数は、ユーザーアプリケーション・ファイルを指定宛先へ送信するために使用されます。

file_send_api
　センターへ指定ファイルを送信します。

file_overwrite_api
　センターへ指定ファイルを上書き送信します。

file_resend_api
　センターへ指定ファイルを再送信します。

ファイル受信関数

これらの関数は、自分宛に送られているファイルを受信するために使用されます。

file_recv_api
file_recv_api2
　センターに蓄積されているファイルを受信します。

file_rerecv_api
file_rerecv_api2
　センターに蓄積されているファイルを再受信します。

照会関数

これらの関数は、ファイルの蓄積状況などを取得するために使用されます。
取得するために使用されます。

file_list_api
　ｾﾝﾀｰに蓄積されている自分宛のﾌｧｲﾙ一覧の照会行います。

audit_list_api
　オーディットトレイルを受信します。

第3章　関数の使用

この章では、すべてのTDC通信パッケージ(Linux)関数を紹介しています。
各関数の説明、構文、パラメータ、戻りコード、および例が記載されています。

共有ライブラリのインストール

TDC通信パッケージ(Linux)の各種関数を提供する共有ライブラリー(.so)ファイルをプログラム
からアクセスするためには、共有ライブラリーをLinuxが検索するディレクトリーに入れておく必要があります。

導入時に、libtns.soファイルを次のディレクトリーに入れてください。

共有ライブラリディレクトリ：/usr/lib64

関数の実行形態

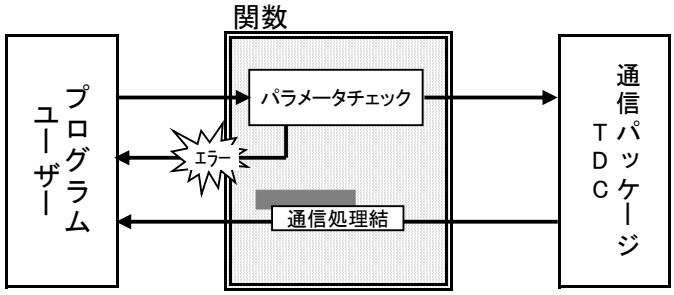
TDC通信パッケージ(Linux)は、複数のサービスを同時に受け付けることが可能です。
よって複数のユーザーアプリケーションから関数を同時に呼出すことができます。但し、実際に通信が
行えるセッション数に制限がありますので、そのセッション数を超えたサービス要求は、
TDC通信パッケージの処理待ち行列に入ります。

ここでは、関数をアプリケーションから使用する際の実行形態について説明されています。

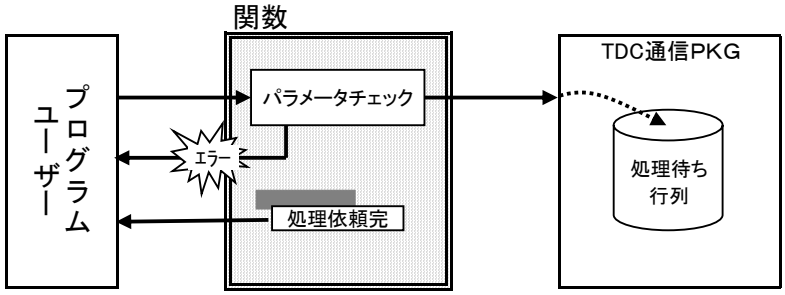
関数の同期／非同期

各関数は、次の実行形態の中から選択できます。
・関数からの通信処理結果を待つ(同期実行)
・関数からの通信処理結果を待たない(非同期実行)

・関数からの通信処理結果を待つ(同期実行)
ユーザープログラムが要求したサービスの通信処理が完了するまで関数から戻りません。



・関数からの通信処理結果を待たない(非同期実行)
ユーザープログラムから渡されたパラメータのシンタクスチェックを行い、TDC通信パッケージ(Linux)へ
処理依頼を通知したら、即座に関数から戻ります。



処理待ち行列の優先順位

各関数は、TDC通信パッケージ(Linux)が処理する優先順位を指定する
ことができます。下記に実行優先順を示します。

・実行優先順位

優先順位	指定パラメータ
1	同期／優先
2	非同期／優先
3	同期／通常
4	非同期／通常

同一優先順位の場合、先入れ先出しとなります。

関数の戻りコード

TDC通信パッケージ(Linux)関数はすべて、整数の状況コードを返します。戻りコードを照会
して、その関数が正常に完了したかエラーになったかを判別することができます。戻りコードには簡単な規則
がありますが、それによりエラーまたは特殊条件をプログラムによって処理することが容易になります。

戻りコード	意味
ret ≠ 0	エラー状態が発生しました。
ret ＝ 0	関数が正常に完了しました。

※ 戻りコードが正常(0)を返したとしても、処理要求が正常終了したとは限りません。
必ず、処理結果コードの確認を行って下さい。

関数のパラメータ

各パラメータにはC言語データタイプ、パラメータタイプ、指定タイプ、および簡単な説明が付記されています。

・データタイプ
char*　　文字列の先頭ポインター

・パラメータタイプ
（I） 入力
プログラマーが指定します。
（O） 出力
TDC通信パッケージ(Linux)から戻されます。
（I/O） 入力／出力
プログラマーが指定し、TDC通信パッケージ(Linux)によって変更されます。

・指定タイプ
（必須）
入力省略できません。指定桁数に文字列が満たない場合は、残り桁数に空白をセットするか、
文字列の最後にNULL（0x00）をセットして下さい。
（任意）
入力省略可能です。省略時は全ての桁数に空白をセットして下さい。

※各関数の個別パラメータによっては指定に制限がありますので、各関数のパラメータの説明に
従って下さい。
※各関数の処理シーケンスパラメータに関しては、呼び出し元がセットする場合、
各関数を呼び出す都度、処理シーケンスコードがユニークなコードになるようにセットして下さい。
※スレッドにより並列でAPIを呼び出さないで下さい。

file_send_api（ファイル送信）

センターへ指定ファイルを送信します。

【書式】

```
int  ret  = file_send_api(uid, result_val,  tran_key,  async_flag,  make_head_flag,  msg_id,  send_name,  file_name,
                          file_ext,  file_type_flag,  compress_flag,  a_code_flag,  code_cnv_flag,  apl_length) ;
```

【パラメーター】

パラメータ	タイプ	桁数	表記形式	説明	入出力	必須
uid	char*	20	英数字推奨	使用する利用者ID（省略時はデフォルト使用）	IN	×
result_val	char*	8	数字	ライブラリでの処理結果をセット 前４桁が理由コード後４桁が内部ロジックコード	OUT	○
tran_key	char*	8	英数字	非同期モードで処理要求を発行した場合、 該当する処理の状況確認を行う際に利用する 呼び出し元が利用者IDごとにユニークな英数字を セットする 同期モード時でもセットすること	IN/ OUT	○
async_flag	char	1	'1':非同期＆通常 '2':同期＆通常 '3':非同期＆優先 '4':同期＆優先	同期／非同期、通常／優先の設定	IN	○
make_head_flag	char	1	'0':未処理 '1':ヘッダ作成	TNS形式、TNS拡張形式以外のフォーマットでは 無効になります。	IN	○
msg_id	char*	8	英数字	NMSメッセージ識別コード セットされていない場合はエラーとして処理を 中断する	IN	○
send_name	char*	24	英数字推奨	宛先	IN	○
file_name	char*	128	絶対パス付きのファイル名	送信するファイル名 ファイルが存在しない場合エラーとして処理を 中断する	IN	○
file_ext	char*	5	英数字	ファイル拡張子	IN	×
file_type_flag	char	1	'1':ＴＮＳ '2':ＴＮＳ拡張 '3':標準 '4':自由	フォーマット種別	IN	×
compress_flag	char	1	'0':圧縮なし '1':圧縮あり	データの圧縮方法	IN	×
a_code_flag	char	1	'0':暗号化なし '1':暗号化あり	データの暗号化方法	IN	×
code_cnv_flag	char	1	'0':コード変換なし '1':コード変換あり	コード変換方法を示す EBCDIC<->ASCII	IN	×
apl_length	chra*	5	数字	APL有効長	IN	×

【戻り値】

0	正常終了
0 以外	エラー

【解説】

任意のパラメータに値をセットしない場合は char の場合 NULL(0x00)をセットし char*の場合文字列の先頭バイトに NULLをセットすること
任意のパラメータの値がセットされていない項目は、msg_id パラメータにセットされた値で転送制御マスタを検索し値をセットする
転送制御マスタより値が得られなかった場合、転送制御マスター定義エラーとし処理を中断する。
データ圧縮・データ暗号化は、DIEX送信時のみ有効。

file_overwrite_api（ファイル上書き送信）

センターへ指定ファイルを上書き送信します。

【書式】

```
int  ret  = file_overwrite_api(uid, result_val,  tran_key,  async_flag,  make_head_flag,  msg_id,  send_name,  file_name,
                    file_ext,  file_type_flag,  compress_flag,  a_code_flag,  code_cnv_flag,  apl_length) ;
```

【パラメーター】

パラメータ	タイプ	桁数	タイプ	説明	入出力	必須
uid	char*	20	英数字推奨	使用する利用者ID（省略時はデフォルト使用）	IN	×
result_val	char*	8	数字	ライブラリでの処理結果をセット 前４桁が理由コード後４桁が内部ロジックコード	OUT	○
tran_key	char*	8	英数字	非同期モードで処理要求を発行した場合、 該当する処理の状況確認を行う際に利用する 過去にファイル送信を行った際に指定した英数字 を設定します 同期モード時でもセットすること	IN	○
async_flag	char	1	'1':非同期＆通常 '2':同期＆通常 '3':非同期＆優先 '4':同期＆優先	同期／非同期、通常／優先の設定	IN	○
make_head_flag	char	1	'0':未処理 '1':ヘッダ作成	TNS形式、TNS拡張形式以外のフォーマットでは 無効になります。	IN	○
msg_id	char*	8	英数字	NMSメッセージ識別コード セットされていない場合はエラーとして処理を 中断する	IN	○
send_name	char*	24	英数字推奨	宛先	IN	○
file_name	char*	128	絶対パス付きのファイル名	送信するファイル名 ファイルが存在しない場合エラーとして処理を 中断する	IN	○
file_ext	char*	5	英数字	ファイル拡張子	IN	×
file_type_flag	char	1	'1':ＴＮＳ '2':ＴＮＳ拡張 '3':標準 '4':自由	フォーマット種別	IN	×
compress_flag	char	1	'0':圧縮なし '1':圧縮あり	データの圧縮方法	IN	×
a_code_flag	char	1	'0':暗号化なし '1':暗号化あり	データの暗号化方法	IN	×
code_cnv_flag	char	1	'0':コード変換なし '1':コード変換あり	コード変換方法を示す EBCDIC<->ASCII	IN	×
apl_length	chra*	5	数字	APL有効長	IN	×

【戻り値】

0	正常終了
0 以外	エラー

【解説】

任意のパラメータに値をセットしない場合は char の場合 NULL(0x00)をセットし char*の場合文字列の先頭バイトに NULLをセットすること
任意のパラメータの値がセットされていない項目は、msg_id パラメータにセットされた値で転送制御マスタを検索し値をセットする
転送制御マスタより値が得られなかった場合、転送制御マスター定義エラーとし処理を中断する。

file_resend_api(ファイル再送信)

センターへ指定ファイルを再送信します。

【書式】

```
int  ret  = file_resend_api(uid, result_val,  tran_key,  async_flag,  msg_id,  file_name,  file_ext) ;
```

【パラメーター】

パラメータ	タイプ	桁数	タイプ	説明	入出力	必須
uid	char*	20	英数字推奨	使用する利用者ID（省略時はデフォルト使用）	IN	×
result_val	char*	8	数字	ライブラリでの処理結果をセット 前４桁が理由コード後４桁が内部ロジックコード	OUT	○
tran_key	char*	8	英数字	非同期モードで処理要求を発行した場合、 該当する処理の状況確認を行う際に利用する 過去にファイル送信を行った際に指定した英数字 を設定します 同期モード時でもセットすること	IN	○
async_flag	char	1	'1':非同期＆通常 '2':同期＆通常 '3':非同期＆優先 '4':同期＆優先	同期／非同期、通常／優先の設定	IN	○
file_name	char*	128	絶対パス付きのファイル名	送信するファイル名 ファイルが存在しない場合エラーとして処理を 中断する	IN	○
file_ext	char*	5	英数字	ファイル拡張子	IN	×

【戻り値】

0	正常終了
0 以外	エラー

【解説】

任意のパラメータに値をセットしない場合は char の場合 NULL(0x00)をセットし char*の場合文字列の先頭バイトに NULLをセットすること
任意のパラメータの値がセットされていない項目は、msg_id パラメータにセットされた値で転送制御マスタを検索し値をセットする
転送制御マスタより値が得られなかった場合、転送制御マスター定義エラーとし処理を中断する。
別センターへの再送信は不可。

file_recv_api(ファイル受信)

センターに蓄積されているファイルを受信します。

【書式】

```
int  ret  = file_recv_api(uid, result_val, tran_key, async_flag, file_merge_flag, make_head_flag,
                          msg_id, recv_name, file_name, file_ext, file_type_flag, compress_flag,
```

【パラメーター】

パラメータ	タイプ	桁数	タイプ	説明	入出力	必須
uid	char*	20	英数字推奨	使用する利用者ID（省略時はデフォルト使用）	IN	×
result_val	char*	8	数字	ライブラリでの処理結果をセット 前４桁が理由コード後４桁が内部ロジックコード	OUT	○
tran_key	char*	8	英数字	非同期モードで処理要求を発行した場合、 該当する処理の状況確認を行う際に利用する 呼び出し元が利用者IDごとにユニークな英数字 をセットする 同期モード時でもセットすること	IN/ OUT	○
async_flag	char	1	'1':非同期＆通常 '2':同期＆通常 '3':非同期＆優先 '4':同期＆優先	同期／非同期、通常／優先の設定	IN	○
file_merge_flag	char	1	'0':結合を行わない '1':結合を行う	複数ファイル受信時の結合の有無	IN	○
make_head_flag	char	1	'0':未処理 '1':ヘッダ作成付加	TNS形式、TNS拡張形式以外のフォーマットでは 無効になります。	IN	○
msg_id	char*	8	英数字	NMSメッセージ識別コード セットされていない場合はエラーとして処理を 中断する	IN	○
recv_name	char*	24	英数字推奨	差し出し元	IN	○
file_name	char*	128	絶対パス	受信ファイルを書き込むパス名 パスが存在しない場合エラーとして処理を中断する	IN	○
file_ext	char*	5	英数字	ファイル拡張子	IN	×
file_type_flag	char	1	'1':ＴＮＳ '2':ＴＮＳ拡張 '3':標準 '4':自由	フォーマット種別	IN	×
compress_flag	char	1	'0':圧縮なし '1':圧縮あり	データの圧縮方法	IN	×
a_code_flag	char	1	'0':暗号化なし '1':暗号化あり	データの暗号化方法	IN	×
code_cnv_flag	char	1	'0':コード変換なし '1':コード変換あり	コード変換方法を示す EBCDIC<->ASCII	IN	×
apl_length	chra*	5	数字	APL有効長	IN	×

【戻り値】

0	正常終了
0 以外	エラー

【解説】

任意のパラメータに値をセットしない場合は char の場合 NULL(0x00)をセットし char*の場合文字列の先頭バイトに NULLをセットすること
任意のパラメータの値がセットされていない項目は、msg_id パラメータにセットされた値で転送制御マスタを検索し値をセットする
転送制御マスタより値が得られなかった場合、転送制御マスター定義エラーとし処理を中断する。

file_recv_api2(拡張ファイル受信)

センターに蓄積されているファイルを受信します。

【書式】

```
int  ret  = file_recv_api2(uid, result_val, tran_key, async_flag, file_merge_flag, make_head_flag,
                           msg_id, recv_name, file_name, recv_ctl_file, recv_cnt_tbl, file_ext,
```

【パラメーター】

パラメータ	タイプ	桁数	タイプ	説明	入出力	必須
uid	char*	20	英数字推奨	使用する利用者ID（省略時はデフォルト使用）	IN	×
result_val	char*	8	数字	ライブラリでの処理結果をセット 前４桁が理由コード後４桁が内部ロジックコード	OUT	○
tran_key	char*	8	英数字	非同期モードで処理要求を発行した場合、 該当する処理の状況確認を行う際に利用する 呼び出し元が利用者IDごとにユニークな英数字 をセットする 同期モード時でもセットすること	IN/ OUT	○
async_flag	char	1	'1':非同期＆通常 '2':同期＆通常 '3':非同期＆優先 '4':同期＆優先	同期／非同期、通常／優先の設定	IN	○
file_merge_flag	char	1	'0':結合を行わない '1':結合を行う	複数ファイル受信時の結合の有無	IN	○
make_head_flag	char	1	'0':未処理 '1':ヘッダ作成付加	TNS形式、TNS拡張形式以外のフォーマットでは 無効になります。	IN	○
msg_id	char*	8	英数字	NMSメッセージ識別コード セットされていない場合はエラーとして処理を 中断する	IN	○
recv_name	char*	24	英数字推奨	差し出し元	IN	○
file_name	char*	128	絶対パス	受信ファイルを書き込むパス名 パスが存在しない場合エラーとして処理を中断する	IN	○
recv_ctl_file	char*	128	絶対パス	受信した各ファイルのヘッダー情報を 書込むファイル名をセットします。	IN	○
recv_cnt_tbl	char*	12	数字	受信したファイル件数を返します。	OUT	○
file_ext	char*	5	英数字	ファイル拡張子	IN	×
file_type_flag	char	1	'1':ＴＮＳ '2':ＴＮＳ拡張 '3':標準 '4':自由	フォーマット種別	IN	×
compress_flag	char	1	'0':圧縮なし '1':圧縮あり	データの圧縮方法	IN	×
a_code_flag	char	1	'0':暗号化なし '1':暗号化あり	データの暗号化方法	IN	×
code_cnv_flag	char	1	'0':コード変換なし '1':コード変換あり	コード変換方法を示す EBCDIC<->ASCII	IN	×
apl_length	chra*	5	数字	APL有効長	IN	×

【戻り値】

0	正常終了
0 以外	エラー

【解説】

任意のパラメータに値をセットしない場合は char の場合 NULL(0x00)をセットし char*の場合文字列の先頭バイトに NULLをセットすること
任意のパラメータの値がセットされていない項目は、msg_id パラメータにセットされた値で転送制御マスタを検索し値をセットする
転送制御マスタより値が得られなかった場合、転送制御マスター定義エラーとし処理を中断する。

file_rerecv_api(ファイル再受信(保管取出し))

センターに蓄積されているファイルを再受信します。

【書式】

```
int  ret  = file_rerecv_api(uid, result_val,  tran_key,  async_flag, file_path, fext) ;
```

【パラメーター】

パラメータ	タイプ	桁数	タイプ	説明	入出力	必須
uid	char*	20	英数字推奨	使用する利用者ID（省略時はデフォルト使用）	IN	×
result_val	char*	8	数字	ライブラリでの処理結果をセット 前４桁が理由コード後４桁が内部ロジックコード	OUT	○
tran_key	char*	8	英数字	非同期モードで処理要求を発行した場合、 該当する処理の状況確認を行う際に利用する 過去にファイル受信を行った際に指定した英数字 を設定します 同期モード時でもセットすること	IN	○
async_flag	char	1	'1':非同期＆通常 '2':同期＆通常 '3':非同期＆優先 '4':同期＆優先	同期／非同期、通常／優先の設定	IN	○
file_path	char*	128	絶対パス	受信ファイルを書き込むパス名 パス名が存在しない場合エラーとして処理を 中断する	IN	○
file_ext	char*	5	英数字	ファイル拡張子	IN	×

【戻り値】

0	正常終了
0 以外	エラー

【解説】

過去に指定したtran_keyが'受信'処理以外の場合エラーになります。

file_rerecv_api2(拡張ファイル再受信(保管取出し))

センターに蓄積されているファイルを再受信します。

【書式】

```
int  ret  = file_rerecv_api2(uid, result_val, hokan_key, async_flag, file_path, recv_ctl_file , file_ext) ;
```

【パラメーター】

パラメータ	タイプ	桁数	タイプ	説明	入出力	必須
uid	char*	20	英数字推奨	使用する利用者ID（省略時はデフォルト使用）	IN	×
result_val	char*	8	数字	ライブラリでの処理結果をセット 前４桁が理由コード後４桁が内部ロジックコード	OUT	○
hokan_key	char*	12	英数字	拡張ファイル受信で返された個別再受信キーを セットします。	IN	○
async_flag	char	1	'1':非同期＆通常 '2':同期＆通常 '3':非同期＆優先 '4':同期＆優先	同期／非同期、通常／優先の設定	IN	○
file_path	char*	128	絶対パス	受信ファイルを書き込むパス名 パス名が存在しない場合エラーとして処理を 中断する	IN	○
recv_ctl_file	char*	128	絶対パス	受信した各ファイルのヘッダー情報を書込む ファイル名をセットします。	IN	○
file_ext	char*	5	英数字	ファイル拡張子	IN	×

【戻り値】

0	正常終了
0 以外	エラー

【解説】

過去に指定したhokan_keyが'拡張受信'処理以外の場合エラーになります。

【補足説明】

拡張ファイル受信で受信したファイルのみ、拡張ファイル再受信が行えます。
拡張ファイル受信で受信した際に、受信ファイル統合有りで受信処理が行われた受信指示は、拡張ファイル再受信は行えません。

file_list_api(受信ファイル照会(蓄積状況))

センターに蓄積されている自分宛のファイル一覧の照会行います。

【書式】

```
int  ret  = file_list_api(uid, result_val,  tran_key,  async_flag,  msg_id,  recv_name,  recv_file_name);
```

【パラメーター】

パラメータ	タイプ	桁数	タイプ	説明	入出力	必須
uid	char*	20	英数字推奨	使用する利用者ID（省略時はデフォルト使用）	IN	×
result_val	char*	8	数字	ライブラリでの処理結果をセット 前４桁が理由コード後４桁が内部ロジックコード	OUT	○
tran_key	char*	8	英数字	非同期モードで処理要求を発行した場合、 該当する処理の状況確認を行う際に利用する 呼び出し元が利用者IDごとにユニークな英数字 をセットする 同期モード時でもセットすること	IN/ OUT	○
async_flag	char	1	'1':非同期＆通常 '2':同期＆通常 '3':非同期＆優先 '4':同期＆優先	同期／非同期、通常／優先の設定	IN	○
msg_id	char*	8	英数字	NMSメッセージ識別コード セットされていない場合はエラーとして処理を 中断する ワイルドカードで指定する場合、 先頭１バイト目に'*'をセットする	IN	×
recv_name	char*	24	英数字推奨	差し出し元	IN	×
recv_file_name	char*	128	絶対パス付きのファイル名	受信するファイル名 ファイル名の設定がない場合、パス名が存在しない 場合エラーとして処理を中断する	IN	○

【戻り値】

0	正常終了
0 以外	エラー

【補足説明】

受信ファイル照会を行った際、蓄積ファイルが0件だった場合、異常とせず正常終了とする

audit_list_api(オーディットトレイル受信)

オーディットトレイルの受信処理を行います。

【書式】

```
int  ret  = audit_list_api(uid, result_val,  tran_key,  async_flag,  from_date, to_date,
```

【パラメーター】

パラメータ	タイプ	桁数	タイプ	説明	入出力	必須
uid	char*	20	英数字推奨	使用する利用者ID（省略時はデフォルト使用）	IN	×
result_val	char*	8	数字	ライブラリでの処理結果をセット 前４桁が理由コード後４桁が内部ロジックコード	OUT	○
tran_key	char*	8	英数字	非同期モードで処理要求を発行した場合、 該当する処理の状況確認を行う際に利用する 呼び出し元が利用者IDごとにユニークな英数字 をセットする 同期モード時でもセットすること	IN/ OUT	○
async_flag	char	1	'1':非同期＆通常 '2':同期＆通常 '3':非同期＆優先 '4':同期＆優先	同期／非同期、通常／優先の設定	IN	○
from_date	char*	6	"YYMMDD"	メッセージのオーディットを取り出す開始日付 指定がない場合は 970101	IN	×
to_date	char*	6	"YYMMDD"	メッセージのオーディットを取り出す終了日付 指定がない場合は 000101(内部では210101)	IN	×
message_class	char	1	'1':送信メッセージ '2':受信メッセージ '*':すべてのメッセージ	取り出すメッセージの種類 '1', '2' 以外はすべてのメッセージを対象とする	IN	×
message_status	char	1	'1':未受信状態 '2':受信状態 '3':受信前に削除された 状態 '*':すべての状態	取り出すメッセージの状態 '1','2','3' 以外はすべてのメッセージを 対象とする	IN	×
recv_name	char*	128	絶対パス付きのファイル名	パス名が存在しない場合、ファイル名の指定がない場合はエラーとして処理を中断する	IN	○

【戻り値】

0	正常終了
0 以外	エラー

【補足説明】

オーディットトレイル照会を行った際、オーディットトレイルが0件だった場合、異常とせず正常終了とする

ファイル命名規則

・送信保管ファイル

メッセージ識別子＋”．”＋システム日付＋SeqNo＋”．”＋ファイル拡張子

※SeqNoは、日付内システム連番

注：ファイル拡張子が省略(転送制御マスタでも省略されている場合)は "." + ファイル拡張子の部分が無くなります。

・受信保管ファイル

＜ＴＮＳ標準フォーマット（ファイルタイプ＝”１”）＞

１．ファイル統合＝”あり”の場合

差出し元先頭１Byte＝”＊”の場合

メッセージ識別子＋”．”＋システム日付＋SeqNo＋”．”＋ファイル拡張子

差出し元先頭１Byte≠”＊”の場合

メッセージ識別子＋”．”＋システム日付＋SeqNo＋”．”＋差出し元＋”．”＋ファイル拡張子

※SeqNoは、日付内システム連番

２．ファイル統合＝”なし”の場合

受信ファイルのＨヘッダーの情報より

ファイル＋”．”＋ファイル作成日付＋当日SeqNo＋”．”＋宛先取引先コード＋”．”＋発信元取引先コード＋”．”＋ファイル拡張子

＜ＴＮＳ拡張フォーマット（ファイルタイプ＝”２”）＞

１．ファイル統合＝”あり”の場合

差出し元先頭１Byte＝”＊”の場合

メッセージ識別子＋”．”＋システム日付＋SeqNo＋”．”＋ファイル拡張子

差出し元先頭１Byte≠”＊”の場合

メッセージ識別子＋”．”＋システム日付＋SeqNo＋”．”＋差出し元＋”．”＋ファイル拡張子

※SeqNoは、日付内システム連番

２．ファイル統合＝”なし”の場合

受信ファイルのＨヘッダーの情報より

ファイルコード＋ファイル詳細コード＋”．”＋作成日付世紀＋作成日付＋SeqNo＋SEQ拡張＋

宛先取引先コード＋宛先取引先サブアドレス＋”．”＋発信元ｺｰﾄﾞ＋発信元ｻﾌﾞｱﾄﾞﾚｽ＋ﾌｧｲﾙ拡張子

＜標準形式（ファイルタイプ＝”３”）＞

差出し元先頭１Byte＝”＊”の場合

メッセージ識別子＋”．”＋システム日付＋SeqNo＋”．”＋ファイル拡張子

差出し元先頭１Byte≠”＊”の場合

メッセージ識別子＋”．”＋システム日付＋SeqNo＋”．”＋差出し元取引先コード(ファイル内)＋”．”＋ファイル拡張子

※SeqNoは、日付内システム連番

＜自由形式（ファイルタイプ＝４”）＞

差出し元先頭１Byte＝”＊”の場合

メッセージ識別子＋”．”＋システム日付＋SeqNo＋”．”＋ファイル拡張子

差出し元先頭１Byte≠”＊”の場合

メッセージ識別子＋”．”＋システム日付＋SeqNo＋”．”＋差出し元取引先コード(パラメータ)＋”．”＋ファイル拡張子

※SeqNoは、日付内システム連番

注：ファイル拡張子が省略(転送制御マスタでも省略されている場合)は "." + ファイル拡張子の部分が無くなります。

注２：ホストから受信後、エラーが発生したファイルはエラー保管領域に保存されます。ファイル名はホストのファイル名と同じです。
（システム環境設定「ローカル保管パス名」で指定したディレクトリ内の「error」ディレクトリに保管されます。）

・受信ファイル

一時保管ファイル名と同名

付録A. 適用業務プログラムの開発

ここでは、ユーザーがプログラミング作業を行うときに役立つ指針とプログラミング上の注意事項を示します。
TDC通信パッケージ(Linux)の導入処理中には、開発に必要ないくつかのファイルが導入ディレクトリーにコピーされます。そのようなファイルには、以下のものがあります。

（Cプログラミング・サポートファイル）

・libtns.so,libtns.aファイル
　このファイルは、C言語のTDC通信パッケージ(Linux)・共有ライブラリーです。

・tnaapi.h,tnsdefine.h ヘッダーファイル
　このファイルは、C言語のヘッダーファイルです。TDC通信パッケージ(Linux)で使用される関数および戻りコードのプロトタイプを提供します。

Cプログラミング・サポート

インクルードファイル

TDC通信パッケージ(Linux)のインクルードファイルを使用するアプリケーションプログラムは、
tnsapi.hのインクルードすることをお勧めします。

適用業務のコンパイルとリンク

適用業務はそれぞれのプロセッサーで実行される前に、コンパイルとリンクがなされなければなりません。

gccを使用する場合：

コンパイルするには：	gcc −c myprog.c
リンクするには：	gcc −o myprog myprog.o −ltns

上記において：

myprog	システムユニット上の適用業務名
tns	TDC通信パッケージ(Linux)・共有ライブラリ

付録B. 受信ファイル情報書込みファイルのファイルレイアウト

通知する項目／レイアウトは次の通り。

ﾌｫｰﾏｯﾄ区分	ヘッダレコードのファイルタイプ	X(01)
発信元	ヘッダレコードの発信元	X(16)
宛先	ヘッダレコードの宛先	X(16)
ﾒｯｾｰｼﾞ識別子	受信したﾒｯｾｰｼﾞ識別子	X(08)
日付	ヘッダの日付	X(08)
SEQ	ヘッダのSEQ	X(03)
件数	トレーラの件数	X(07)
個別再受信キー	システム側で割り振ったユニークな値	X(12)
ファイル名	受信したファイルの物理ファイル名	X(45)

	ﾌｧｲﾙﾀｲﾌﾟ	発信元	宛先	ﾒｯｾｰｼﾞ識別子	日付	SEQ	件数	個別再受信キー	ﾌｧｲﾙ名
TNS標準	1	T1000 (残りﾌﾞﾗﾝｸ11)	03101 (残りﾌﾞﾗﾝｸ11)	ITNSM01△	000306△△	03△	00023△△	000000010001	AAAA.DAT (残りﾌﾞﾗﾝｸ)
TNS拡張	2	T10000003 (残りﾌﾞﾗﾝｸ8)	03101001 (残りﾌﾞﾗﾝｸ8)	ITNSM01△	20000306	003	0000023	000000010001	AAAA.DAT (残りﾌﾞﾗﾝｸ)
TDC標準	3	OTNS△△△△ OTNS001△	OTNSD△△△ OTNS038△	ITNSAAZ1	ﾌﾞﾗﾝｸ	ﾌﾞﾗﾝｸ	ﾌﾞﾗﾝｸ	000000010001	AAAA.DAT (残りﾌﾞﾗﾝｸ)
自由	4	OTNS△△△△ OTNS001△	OTNSD△△△ OTNS038△	ITNSAAZ1	ﾌﾞﾗﾝｸ	ﾌﾞﾗﾝｸ	ﾌﾞﾗﾝｸ	000000010001	AAAA.DAT (残りﾌﾞﾗﾝｸ)