



DSHEng4 装置通信エンジン (GEM+GEM300)
ソフトウェア・パッケージ

APP インタフェース
ライブラリ関数説明書
(C, C++, .Net-Vb,C#)

VOL- 7 / 1 5

3 . 13 RCP レシビ情報アクセスサービス関数

2 0 0 9年7月

株式会社データマップ

文書番号 DSHEng4-09-30327-00



[取り扱い注意]

- ・ この資料ならびにソフトウェアの一部または全部を無断で使用、複製することはできません。
- ・ 本説明書に記述されている内容は予告なしで変更される可能性があります。
- ・ Windows は米国 Microsoft Corporation の登録商標です。
- ・ ユーザーが本ソフトウェアの使用によって生じた遺失履歴、(株)データマップの予見の有無を問わず発生した特別損害、付随的損害、間接損害およびその他の拡大損害に対して責任を負いません。

【改訂履歴】

番号	改訂日付	項目	概略
1.	2009. 7	初版	

目 次

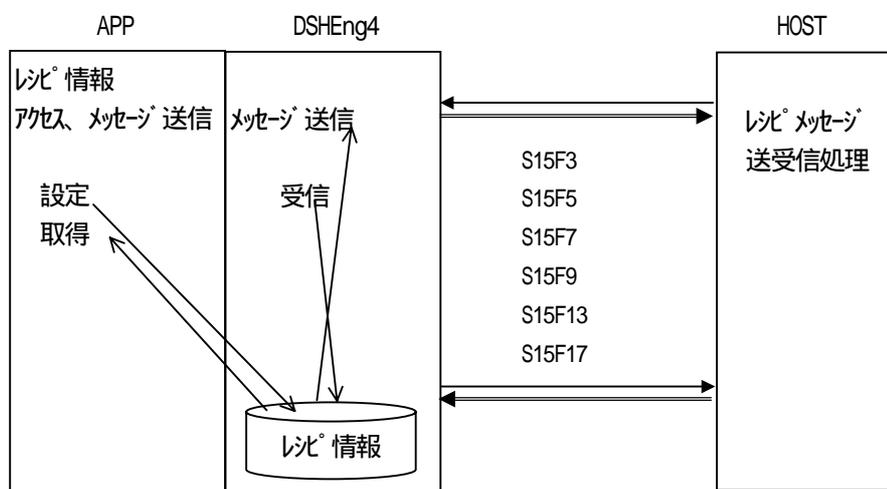
3.13 RCP レシピ情報アクセスサービス関数.....	1
3.13.1 使用する情報格納構造体.....	4
3.13.2 RCP レシピ情報アクセス関数.....	7
3.13.2.1 EngAllocRcpInfo() - レシピの登録.....	7
3.13.2.2 EngSetRcpInfo() - レシピ情報の設定.....	8
EngSetRcpInfoX() - インデクス指定でのレシピ情報の設定.....	8
3.13.2.3 EngGetRcpInfo() - レシピ情報の取得.....	10
EngGetRcpInfoX() - インデクス指定でのレシピ情報の取得.....	10
3.13.2.4 EngDelRcpInfo() - レシピの削除.....	12
EngDelRcpInfoX() - インデクスでのレシピの削除.....	12
3.13.2.5 EngSetRcpState() - レシピ状態の設定.....	13
EngSetRcpStateX() - インデクスでのレシピ状態の設定.....	13
3.13.2.6 EngGetRcpState() - レシピ状態の取得.....	15
EngGetRcpStateX() - インデクスでのレシピ状態の取得.....	15
3.13.2.7 EngGetRcpList() - 全登録レシピ ID 取得関数.....	17
3.13.2.8 EngGetRcpId() - インデクスから RCPID (レシピ ID) の取得.....	18
3.13.2.9 EngGetRcpIdIndex() - RCPID (レシピ ID) からインデクスの取得.....	19
3.13.2.10 EngSendS15F3() - レシピ・スペースアクション要求送信関数.....	20
3.13.2.11 EngSendS15F5() - レシピ・スペース・リネーム要求送信関数.....	23
3.13.2.12 EngSendS15F7() - レシピ・スペース取得メッセージ送信関数.....	26
3.13.2.13 EngSendS15F9() - レシピ・ステータス取得メッセージ送信関数.....	28
3.13.2.14 EngSendS15F13() - レシピ情報メッセージ送信関数.....	30
3.13.2.15 EngSendS15F17() - レシピ情報問合せ関数.....	33
3.13.3 RCP レシピ関連ライブラリ関数.....	35
3.13.3.1 DshDecodeS15F3() - S15F3 をレシピ・アクション情報にデコード.....	35
3.13.3.2 DshEncodeS15F3() - レシピ・アクション情報を S15F3 ヘンコード.....	36
3.13.3.3 DshDecodeS15FRsp() - S15F4, S15F6, S15F14, S15F16, S15F18 のデコード.....	37
3.13.3.4 DshFreeTRCP_ACT_INFO() - レシピ・アクション情報構造体メモリの開放.....	38
3.13.3.5 DshCopyTRCP_ACT_INFO() - レシピ・アクション情報構造体メモリのコピー.....	39
3.13.3.6 DshMakeS15F3Response() - S15F3 の応答メッセージの生成.....	40
3.13.3.7 DshDecodeS15F5() - S15F5 をレシピ・リネーム情報にデコード.....	42
3.13.3.8 DshFreeTRCP_RENAME_INFO() - レシピ・リネーム情報構造体メモリの開放.....	43
3.13.3.9 DshCopyTRCP_RENAME_INFO() - レシピ・リネーム情報構造体メモリのコピー.....	44
3.13.3.10 DshMakeS15F5Response() - S15F5 の応答メッセージの生成.....	45
3.13.3.11 DshDecodeS15F8() - S15F8 応答メッセージのデコード.....	47
3.13.3.12 DshFreeTRCP_S15F8_INFO() - S15F8 応答情報構造体メモリの開放.....	48
3.13.3.13 DshDecodeS15F10() - S15F10 応答メッセージのデコード.....	49
3.13.3.14 DshFreeTRCP_S15F10_INFO() - S15F10 応答情報構造体メモリの開放.....	50
3.13.3.15 DshDecodeS15F13() - S15F13 をレシピ情報にデコード.....	51
3.13.3.16 DshEncodeS15F13() - レシピ情報を S15F13 ヘンコード.....	52
3.13.3.17 DshFreeTRCP_INFO() - レシピ情報構造体メモリの開放.....	54
3.13.3.18 DshCopyTRCP_INFO() - レシピ情報構造体メモリのコピー.....	55
3.13.3.19 DshInitRcpInfo - レシピ TRCP_INFO の初期設定.....	56
3.13.3.20 DshPutRcpPara() - レシピパラメータ情報の追加.....	57
3.13.3.21 DshMakeS15F13Response() - S15F13 の応答メッセージの生成.....	59
3.13.3.22 DshFreeTRCP_ERR_INFO() - レシピ関連応答情報構造体メモリの開放.....	61

3 . 13 . 3 . 23	DshDecodeS15F180) - S15F18 応答メッセージのデコード	62
3 . 13 . 3 . 24	DshFreeTRCP_S15F18_INFO0) - S15F18 応答情報構造体メモリの開放	63
3 . 13 . 3 . 25	DshInitTRCP_ERR_INFO 0) - レシピ応答情報の初期化	64
3 . 13 . 3 . 26	DshPutTRCP_ERR_INFO 0) - レシピエラー情報の設定	65
3 . 13 . 4	ユーザ作成ライブラリ関数	66
3 . 13 . 4 . 1	DshResponseS15F40) - S15F4 レシピネームスペースアクション応答メッセージ	66
3 . 13 . 4 . 2	DshResponseS15F60) - S15F6 レシピネームスペースリネーム応答メッセージ	67
3 . 13 . 4 . 3	DshResponseS15F140) - S15F14 レシピ生成要求応答メッセージ	68

(VOL - 8に続く)

3.13 RCP レシビ情報アクセスサービス関数

ここで述べるレシビ情報は、DSHEng4 が管理します。従って、APP はこれらの情報をアクセスと関連メッセージを送信するために以下の DSHEng4 API 関数を使用します。



(1) 情報アクセスと送信 API 関数

レシビ情報のアクセスとホストへのメッセージ送信に関連するサービスのための API 関数名は一覧表のとおりです。

	API 関数名	機能
1	EngAllocRcpInfo()	レシビ情報領域を割当て登録します。。
2	EngSetRcpInfo()	レシビ情報を設定・変更します。。
3	EngSetRcpInfoX()	インデックス指定でレシビ情報を設定・変更します。。
4	EngGetRcpInfo()	RCPID 指定でレシビ情報を取得します。。
5	EngGetRcpInfoX()	レシビ インデックス指定でレシビ情報を取得します。。
6	EngDelRcpInfo()	RCPID 指定でレシビ情報を削除します。。
7	EngDelRcpInfoX()	レシビ インデックス指定でレシビ情報を削除します。。
8	EngSetRcpState()	RCPID 指定で RCPID の状態を設定します。。
9	EngSetRcpStateX()	レシビ インデックス指定で RCPID の状態を設定します。。
10	EngGetRcpState()	RCPID 指定で RCPID の状態を取得します。。
11	EngGetRcpStateX()	レシビ インデックス指定で RCPID の状態を取得します。。
12	EngGetRcpList()	レシビ ID の一覧リストを取得します。。
13	EngGetRcpId()	指定したレシビ インデックスの RCPID を取得します。。
14	EngGetRcpIdIndex()	指定した RCPID の情報インデックスを取得します。。
15	EngSendS15F3()	レシビ ネームスペースのアクション要求をします。。
16	EngSendS15F5()	レシビ ネームスペースのネーム要求をします。。
17	EngSendS15F7()	レシビ スペース要求をします。。
18	EngSendS15F9()	レシビ ステータスを要求します。。
19	EngSendS15F13()	レシビ 情報をホストに送信します。。
20	EngSendS15F17()	レシビ 情報をホストから取得します。。

RCPID インデックスは、DSHEng4 が管理する各 RCPID 領域の番号です。このインデックスの値は、

EngAllocRcpInfo()関数実行時に DSHeng4 によって割当てられ、APP に渡されます。また、レシピの取得時に、情報格納構造体のメンバー、 index に設定されます。

(2) ライブラリ関数

他に APP が使用できるレシピ情報処理用 API 関数として、以下の関数があります。

	API 関数名	機能
1	DshDecodeS15F3()	S15F3 のメッセージ内レシピネームアクション情報を TRCP_ACT_INFO 構造体にデコード収納するための関数です。
2	DshEncodeS15F3()	TRCP_ACT_INFO 構造体のレシピネームアクション情報を S15F3 メッセージにエンコードするための関数です。
3	DshDecodeS15FRsp()	S15F3, S15F5, S15F13, S15F15, S15F17 の応答メッセージを TRCP_ERR_INFO 構造体にデコードします。
4	DshFreeTRCP_ACT_INFO()	レシピネームアクション情報が格納されている TRCP_ACT_INFO 構造体の内部で使用されているメモリを開放するための関数です。
5	DshCopyTRCP_ACT_INFO()	TRCP_ACT_INFO のレシピネームアクション情報を別の構造体にコピーします。
6	DshMakeS15F3Response()	S15F4 応答メッセージを生成するための関数。u_s15f3.c で使用します。
7	DshDecodeS15F5()	S15F5 のメッセージ内レシピリネーム情報を TRCP_RENAME_INFO 構造体にデコード収納するための関数です。
8	DshFreeTRCP_RENAME_INFO()	レシピリネーム情報が格納されている TRCP_RENAME_INFO 構造体の内部で使用されているメモリを開放するための関数です。
9	DshCopyTRCP_RENAME_INFO()	TRCP_RENAME_INFO のレシピリネーム情報を別の構造体にコピーします。
10	DshMakeS15F5Response()	S15F6 応答メッセージを生成するための関数。u_s15f5.c で使用します。
11	DshDecodeS15F8()	S15F7 の応答メッセージを TRCP_S15F8_INFO 構造体にデコードします。
12	DshFreeTRCP_S15F8_INFO()	TRCP_S15F8_INFO 構造体内に使用されているメモリを開放します。
13	DshDecodeS15F10()	S15F9 の応答メッセージを TRCP_S15F10_INFO 構造体にデコードします。
14	DshFreeTRCP_S15F10_INFO()	TRCP_S15F10_INFO 構造体内に使用されているメモリを開放します。
15	DshDecodeS15F13()	S15F13 のメッセージ内レシピ情報を TRCP_INFO 構造体にデコード収納するための関数です。
16	DshEncodeS15F13()	TRCP_INFO 構造体のプロトタイプ名情報を S15F13 メッセージにエンコードするための関数です。
17	DshFreeTRCP_INFO()	レシピ情報が格納されている TRCP_INFO 構造体の内部で使用されているメモリを開放するための関数です。
18	DshCopyTRCP_INFO()	TRCP_INFO のレシピ情報を別の構造体にコピーします。
19	DshInitRcpInfo()	レシピ情報構造体 TRCP_INFO を初期設定します。。

20	DshPutRcpPara()	レシピパラメータ情報を TRCP_INFO 内に設定します。。
21	DshMakeS15F13Response()	S15F14 応答メッセージを生成するための関数。u_s15f13.c で使用します。
22	DshFreeTRCP_ERR_INFO()	TRCP_ERR_INFO 構造体内に使用されているメモリを開放します。
23	DshDecodeS15F18()	S15F17 の応答メッセージを TRCP_S15F18_INFO 構造体にデコードします。
24	DshFreeTRCP_S15F18_INFO()	TRCP_S15F18_INFO 構造体内に使用されているメモリを開放します。
25	DshInitTRCP_ERR_INFO()	TRCP_ERR_INFO 内に rmax とエラー情報の個数を設定します。
26	DshPutTRCP_ERR_INFO()	TRCP_ERR_INFO 内の err_list 上に 1 個のエラー情報(errcode, errtext)を追加します。

(3) ユーザ作成ライブラリ関数

	ライブラリ関数名	機能
1	DshResponseS15F4()	S15F3 レシピネームスペースアクション応答
2	DshResponseS15F6()	S15F5 レシピネームスペースリネーム応答
3	DshResponseS15F14()	S15F13 レシピ生成要求応答

3.13.1 使用する情報格納構造体

レシピ情報を操作する関数は、共通の情報格納のための TRCP_INFO 構造体を使用します。
 レシピ情報関連の構造体は下記のとおりです。

(1) TRCP_INFO Recipe Information S15F13

```
typedef struct{
    int      index;
    int      state;           // レジ° 状態
    char     *name;          // name
    char     *rcpspec;       // rcpid
    int      para_count;     // # of pparameter
    TRCP_PARA **para_list;   // parameter list
    char     *rcpbody;
}TRCP_INFO;                // Recipe Information
```

(注) ppara_count = 0 の場合、パラメータがないことを意味します。

(2) TRCP_PARA Recipe Parameter Information - S15F13

```
typedef struct{
    char     *rcpparm;       // para name
    int      par_fmt;       // format
    int      par_size;      // array size
    void     *rcpparval;    // para value;
}TRCP_PARA;                // Recipe Parameter
```

(3) TRCP_ERR_INFO Response Information S15F14

```
typedef struct{
    int      rmack;         // U1
    int      err_count;
    TERR_INFO **err_list;
} TRCP_ERR_INFO;
```

(注) TRCP_ERR_INFO の構造は TOBJ_ERR_INFO と同じなので、初期化、設定には TOBJ_ERR_INFO 用の関数を使用します。

(4) TERR_INFO - Response Error Information - S15F14

```
typedef struct{
    int      errcode;
    char     *errtext;
} TERR_INFO;
```

(5) TRCP_ACT_INFO - Recipe Action(Create / Delete) Information - S15f3

```
typedef struct{
    char      *rmnsspec;          // rcpid
    int       rmnscmd;           // action command 1=create,5=delete
}TRCP_ACT_INFO;                // Recipe Action
```

(6) TRCP_RENAME_INFO - Recipe Rename Information - S15F5

```
typedef struct{
    char      *rmnsspec;          // rcpid
    char      *rmnews;           // new rcpid name
}TRCP_RENAME_INFO;            // Recipe Rename
```

(7) TRCP_S15F8_INFO - S15F8 Ack 情報

```
typedef struct{
    ULONG     rmspace;           // レシピネームスペースにある1個のレシピ保存容量(バイト)
    int       rmack;            // ack
    int       err_count;        // 検出エラー数
    TERR_INFO **err_list;       // 検出したエラー情報リスト
} TRCP_S15F8_INFO;
```

(8) TRCP_S15F10_INFO - S15F10 Ack 情報

```
typedef struct{
    int       rcpstat;          // status
    char      *rcpver;          // version
    int       rmack;            // ack
    int       err_count;        // 検出エラー数
    TERR_INFO **err_list;       // 検出したエラー情報リスト
} TRCP_S15F10_INFO;
```

(9) TRCP_ATTR - レシピ属性情報 S15F18

```
typedef struct{
    char      *attrid;          // 属性 ID
    int       format;           // 属性データのフォーマット
    int       asize;            // 属性データの配列サイズ
    void      *attrdata;        // 属性データの格納ポインタ
} TRCP_ATTR;
```

(10) TRCP_SECNM - レシビセクション属性情報 S15F18

```
typedef struct{
    char      *rcpsecnm;      // レシビセクションネーム
    int       attr_count;    // 属性情報の数
    TRCP_ATTR **attr_list;   // 属性情報のリスト
} TRCP_SECNM;
```

(11) TRCP_S15F18_INFO - S15F18 ack 情報

```
typedef struct{
    TRCP_SECNM *m_secnm;     // レシビセクション属性情報のポインタ
    char      *rcpbody;     // レシビボディ
    int       dset_count;   // secnm_list のデータセット数
    TRCP_SECNM **secnm_list; // エージェント固有の属性情報リスト
    int       rmack;        // S15F18 の ack
    int       err_count;    // 検出したエラー数
    TERR_INFO **err_list;   // 検出したエラー情報リスト
} TRCP_S15F18_INFO;
```

3.13.2 RCP レシピ情報アクセス関数

3.13.2.1 EngAllocRcpInfo() - レシピの登録

(1) 呼出書式

[C, C++]

```
API int APIX EngAllocRcpInfo(
    char *rcpid,          // レシピ ID 格納領域のポインタ
    int *index           // 得られた情報領域のインデックス格納用ポインタ
);
```

[.NET VB]

```
Function EngAllocRcpInfo (
    ByVal rcpid As String,
    ByRef index As Int32) As Int32
```

[.NET C#]

```
int EngAllocRcpInfo(
    byte[] rcpid,
    ref int index);
```

(2) 引数

rcpid

登録したいレシピ ID が格納されているポインタです。

index

登録された情報領域のインデックス値が格納される領域のポインタです。

(3) 戻り値

戻り値	意味
0	正常に登録できた。
1	指定された RCPID は既に登録されていた。
(-1)	登録できなかった。

(4) 説明

レシピを新規にシステムに登録するための関数です。

登録は、引数 rcpid で与えられるレシピ ID をシステムに登録します。

正常に登録できた場合は、index で指定される領域に登録された情報領域のインデックスが設定返却されます。もし、rcpid に指定されたレシピが既に登録済みであった場合には関数の戻り値 1 を返却します。index には既に登録されている情報領域のインデックスが設定されます。

得られたインデックスを使って、情報の設定、取得、削除などのアクセスを行うことができます。

3.13.2.2 EngSetRcpInfo() - レシピ情報の設定 EngSetRcpInfoX() インデクス指定でのレシピ情報の設定

(1) 呼出書式

[C, C++]

```
API int APIX EngSetRcpInfo(
    TRCP_INFO *pinfo           // レシピ情報格納構造体のポインタ
);

API int APIX EngSetRcpInfoX(
    int index,                // RCPID が格納されている領域のインデクス
    TRCP_INFO *pinfo         // レシピ情報格納構造体のポインタ
);
```

[.NET VB]

```
Function EngSetRcpInfo (
    ByRef pinfo As dsh_info.TRCP_INFO) As Int32

Function EngSetRcpInfoX (
    ByVal index As Int32,
    ByRef pinfo As dsh_info.TRCP_INFO) As Int32
```

[.NET C#]

```
int EngSetRcpInfo(
    ref TRCP_INFO pinfo );

int EngSetRcpInfoX(
    int index,
    ref TRCP_INFO pinfo );
```

(2) 引数

pinfo

設定したいレシピ情報が格納されている格納構造体領域のポインタです。

index

レシピ ID 情報のインデクスです。登録時に EngAllocRcpInfo()関数によって与えられます。インデクスは、RCPID から EngGetRcpIdIndex()関数で取得することができます。

(3) 戻り値

戻り値	意味
0	正常に設定できた。
(-1)	設定できなかった。

(4) 説明

本関数は、pinfo に格納されているレシピ情報の設定・変更に使います。

引数 pinfo 内の構造体メンバー rcpspec に指定されるレシピ ID の情報として設定されます。

pinfo 内には RCPID の他、 コマンドコード、 パラメータなどの情報が含まれます。



指定した RCPID が既に登録済みである場合には、その情報は pinfo 内の情報にすべて書き換えられます。

pinfo 内の RCPID が未登録であった場合は、登録手続きをしてからレシピ情報を設定します。
(EngAllocRcpInfo()関数で行われる登録と同じ登録が行われます。)

3.13.2.3 EngGetRcpInfo() - レシビ情報の取得 EngGetRcpInfoX() インデクス指定でのレシビ情報の取得

(1) 呼出書式

[C, C++]

```
API int APIX EngGetRcpInfo(
    char *rcpid; // RCPID が格納されている領域のポインタ
    TRCP_INFO *pinfo // レシビ情報を格納する構造体ポインタ
);

API int APIX EngGetRcpInfoX(
    int index; // RCPID が格納されている領域のインデクス
    TRCP_INFO *pinfo // レシビ情報を格納する構造体ポインタ
);
```

[.NET VB]

```
Function EngGetRcpInfo (
    ByVal rcpid As String,
    ByRef pinfo As dsh_info.TRCP_INFO) As Int32

Function EngGetRcpInfoX (
    ByVal index As Int32,
    ByRef pinfo As dsh_info.TRCP_INFO) As Int32
```

[.NET C#]

```
int EngGetRcpInfo(
    byte[] rcpid,
    ref TRCP_INFO pinfo);

int EngGetRcpInfoX(
    int index,
    ref TRCP_INFO pinfo );
```

(2) 引数

rcpid

レシビ ID が格納されている領域のポインタです。

pinfo

取得したレシビ情報を格納する構造体領域のポインタを格納するポインタです。

index

レシビ ID 情報のインデクスです。登録時に EngAllLocRcpInfo()関数によって与えられます。
インデクスは、RCPID から EngGetRcpIdIndex()関数で取得することができます。

(3) 戻り値

戻り値	意味
0	正常に取得できた。
(-1)	取得できなかった。(RCPID が未登録であった。)

(4) 説明

rcpid または index に指定されている RCPID のレシピ情報を pinfo が指す構造体に格納します。

TRCP_INFO 構造体の中に情報を格納するために必要なメモリは、DSHEng4 が準備確保します。即ち、構造体のメンバーの中でポインタになっている情報の実体即ち、rcpid、パラメータなどのためのメモリはDSHEng4 が準備します。

これらのメモリは、使用后、ユーザがDSHEng4 のAPI 関数を使って次のように開放してください。

```
TRCP_INFO *pinfo;

if ( EngGetRcpInfo( rcpid, pinfo ) == 0 ){
    pinfo の処理
    処理終了後
    DshFreeTRCP_INFO( pinfo );           // pinfo 内に使用されているメモリの開放
}
```

3.13.2.4 EngDelRcpInfo() - レシピの削除 EngDelRcpInfoX() - インデクスでのレシピの削除

(1) 呼出書式

[C, C++]

```
API int APIX EngDelRcpInfo(
    char *rcpid // RCPID が格納されている領域のポインタ
);

API int APIX EngDelRcpInfo(
    int index // インデクス(0,1,2,...)
);
```

[.NET VB]

```
Function EngDelRcpInfo (
    ByVal rcpid As String) As Int32

Function EngDelRcpInfoX (
    ByVal index As Int32) As Int32
```

[.NET C#]

```
int EngDelRcpInfo(
    byte[] rcpid);

int EngDelRcpInfoX(
    int index );
```

(2) 引数

rcpid
レシピ ID が格納されている領域のポインタです。

index
削除したいレシピ情報のインデクスです。

(3) 戻り値

戻り値	意味
0	正常に削除できた。
(-1)	RCPID が未登録であった。

(4) 説明

rcpid または index に指定されているレシピ ID の情報をシステムの登録から削除します。

3.13.2.5 EngSetRcpState() レシビ状態の設定 EngSetRcpStateX() インデクスでのレシビ状態の設定

(1) 呼出書式

[C, C++]

```
API int APIX EngSetRcpState(
    char    *rcpid,           // RCPID が格納されている領域のポインタ
    int     state            // 設定したい状態値
);
```

```
API int APIX EngSetRcpStateX(
    int     index,          // レシビ情報のインデクス
    int     state            // 設定したい状態値
);
```

[.NET VB]

```
Function EngSetRcpState (
    ByVal rcpid As String,
    ByVal state As Int32) As Int32
```

```
Function EngSetRcpStateX (
    ByVal index As Int32,
    ByVal state As Int32) As Int32
```

[.NET C#]

```
int EngSetRcpState(
    byte[] rcpid,
    int state);
```

```
int EngSetRcpStateX(
    int index,
    int state );
```

(2) 引数

rcpid

レシビ ID が格納されている領域のポインタです。

state

設定したい RCPID の状態値です。

値=(-1)は使用不可を意味する値になります。

index

レシビ情報のインデクスです。登録時に EngAllLocRcpInfo()関数によって与えられます。

インデクスは、RCPID から EngGetRcpIdIndex()関数で取得することができます。

(3) 戻り値

戻り値	意味
0	正常に設定できた。
(-1)	RCPID が未登録であった。

(4) 説明

レシビ情報の状態値を設定します。

状態値の意味合いは値=(-1)以外についてはユーザが定義して使用します。

3.13.2.6 EngGetRcpState() レシビ状態の取得 EngGetRcpStateX() インデクスでのレシビ状態の取得

(1) 呼出書式

[C, C++]

```
API int WINAPI EngGetRcpState(
    char    *rcpid,           // RCPID が格納されている領域のポインタ
    int     *state           // 取得した状態値格納用領域ポインタ
);
```

```
API int WINAPI EngGetRcpStateX(
    int     index,          // レシビのインデクス
    int     *state         // 取得した状態値格納用領域ポインタ
);
```

[.NET VB]

```
Function EngGetRcpState (
    ByVal rcpid As String,
    ByRef state As Int32) As Int32
```

```
Function EngGetRcpStateX (
    ByVal index As Int32,
    ByRef state As Int32) As Int32
```

[.NET C#]

```
int EngGetRcpState(
    byte[] rcpid,
    ref int state);
```

```
int EngGetRcpStateX(
    int index,
    ref int state );
```

(2) 引数

rcpid

レシビ ID が格納されている領域のポインタです。

state

取得した RCPID の状態値を格納する領域のポインタです。

index

レシビ ID 情報のインデクスです。登録時に EngAllLocRcpInfo()関数によって与えられます。インデクスは、RCPID から EngGetRcpIdIndex()関数で取得することができます。

(3) 戻り値

戻り値	意味
0	正常に取得できた。
(-1)	RCPID が未登録であった。

(4) 説明

レシビ情報の状態値を取得します。
状態値の意味合いは値=(-1)以外についてはユーザが定義して使用します。

3.13.2.7 EngGetRcpList() 全登録レシピ ID 取得関数

(1) 呼出書式

[C, C++]

```
API int APIX EngGetRcpList(
    TTEXT_DLIST **list           // 取得リスト格納ポインタの格納ポインタ
);
```

[.NET VB]

```
Function EngGetRcpList (
    ByRef list As IntPtr) As Int32
```

[.NET C#]

```
int EngGetRcpList(
    IntPtr list );
```

(2) 引数

list

取得できた RCPID が格納されている TTEXT_DLIST 構造体のポインタを格納する領域のポインタです。

(3) 戻り値

戻り値	意味
0	正常に取得できた。
(-1)	取得できなかった。

(4) 説明

システムに登録されている RCPID(レシピ ID)とその名前を TTEXT_DLIST 構造体に取り出すための関数です。

取出す名前は、装置管理情報定義ファイルで RCPID 定義時に与えられた名前です。

取得した情報の処理が終了した後、DshFreeTText_DLIST()関数で list 内部の情報格納用に使用されているメモリを開放してください。

TTEXT_DLIST 構造体は次のとおりです。

```
typedef struct{
    int         count;           // 取得できた ID 数
    char        **id_list;      // 取得できた ID 格納用配列
    char        **name_list;    // 取得できた名前格納ポインタ配列
}TTEXT_DLIST;
```

3.13.2.8 EngGetRcpId() インデクスから RCPID (レシピ ID) の取得

(1) 呼出書式

[C, C++]

```
API int APIX EngGetRcpId(
    int index,           // RCP 情報のインデクス
    char *rcpid         // 取得した RCPID を格納する領域のポインタ
);
```

[.NET VB]

```
Function EngGetRcpId (
    ByVal index As Int32,
    ByVal rcpid As String) As Int32
```

[.NET C#]

```
int EngGetRcpId(
    int index,
    byte[] rcpid);
```

(2) 引数

index

RCP 情報のインデクスです。登録時に EngAllocRcpInfo()関数によって与えられます。インデクスは、RCPID から EngGetRcpIdIndex()関数で取得することができます。

rcpid

RCPID を格納するための領域のポインタです。

(3) 戻り値

戻り値	意味
0	正常に取得できた。
(-1)	取得できなかった。(未登録)

(4) 説明

RCP 情報のインデクスから RCPID (レシピ ID) を取得し、rcpid に格納します。正常に取得できた場合は関数戻り値として 0 が返却されます。

3.13.2.9 EngGetRcpIdIndex() RCPID (レシピ ID) からインデクスの取得

(1) 呼出書式

[C, C++]

```
API int APIX EgnGetRcpIdIndex(
    char *rcpid,           // RCPID が格納されている領域のポインタ
    int *index            // 取得したインデクスを格納するための領域のポインタ
);
```

[.NET VB]

```
Function EngGetRcpIdIndex (
    ByVal rcpid As String,
    ByRef index As Int32) As Int32
```

[.NET C#]

```
int EngGetRcpIdIndex(
    byte[] rcpid,
    ref int index);
```

(2) 引数

rcpid

インデクスを取得したい対象の RCPID が格納されている領域のポインタです。

index

取得したインデクスの値を格納するための領域のポインタです。

(3) 戻り値

戻り値	意味
0	正常に取得できた。
(-1)	取得できなかった。(未登録)

(4) 説明

rcpid に指定される RCPID (レシピ ID) からレシピ情報インデクスを取得するための関数です。

取得されたインデクスは index で指定された領域に格納されます。

正常に取得できた場合は関数戻り値として 0 が返却されます。

3.13.2.10 EngSendS15F3() レシピ・スペースアクション要求送信関数

(1) 呼出書式

[C, C++]

```
API int APIX EngSendS15F3(
    char *rcpid,                // レシピ ID 格納領域のポインタ
    int rmns cmd,              // アクションコマンド
    TRCP_ERR_INFO *erinfo,     // 応答情報格納用構造体のポインタ
    int (WINAPI *s15f3Callback)(), // 実行終了時のコールバック関数
    ULONG upara                // callback 時のパラメータ
);
```

[.NET VB]

```
Function EngSendS15F3 (
    ByVal rmns spec As String,
    ByVal rmns cmd As Int32,
    ByRef erinfo As dsh_info.TRCP_ERR_INFO,
    ByVal callback As vcallback.callback_S15F3,
    ByVal upara As Int32) As Int32
```

[.NET C#]

```
int EngSendS15F3(
    byte[] rmns spec,
    int rmns cmd,
    ref TRCP_ERR_INFO erinfo,
    CallbackS15F3 callback,
    uint upara );
```

(2) 引数

rcpid

アクション要求をしたいレシピ ID が格納されているポインタです。

rmns cmd

レシピネームスペースで実行されるコマンドです。(1=生成、5=削除)

erinfo

受信した応答メッセージ S15F4 に含まれる情報を格納するための構造体領域のポインタを指定します。

s15f3Callback

DSHEng4 によるメッセージ送信処理が終了したときに呼出されるコールバック関数を指定します。

ユーザは任意の関数名を指定できます。

コールバックの指定が=0 の場合はブロックモードになります。

upara

コールバックされたときに引数で渡してもらうためのパラメータです。関数実行終了時にコールバックされた際、何かの判別情報として使用したい値、構造体ポインタなどを設定します。

(3) 戻り値

戻り値	意味
0	(1) ブロックモード : 正常に送信できた。 erinfo に S15F4 応答情報が返却されます。

	(2) 非「リターン」：要求が受け付けられた。
(-1)	送信できなかった。
(-14)	T3タイムアウトを検出した。

(4) 説明

rcpid で指定されたレシピネームスペースについて rmnsrmd で指定したアクションの実行要求のために S15F3 メッセージを送信します。

S15F4 応答メッセージ受信で得られた情報はデコードされて erinfo で指定された構造体領域に返却されま
す。

送信要求から S15F4 応答メッセージ受信までの制御は引数のコールバック関数指定の有無によって次のよ
うになります。

s15f3Callback の指定	制御の流れ
なし (=0)	S15F3 送信後、応答メッセージ S15F4 が受信されるか、またはエラーを検出するまで制 御が要求元に戻ってきません。 正常終了であれば、erinfo に応答情報が格納されているポインタが返却されます。
あり	送信要求後、S15F3 の送信前に制御が戻されます。実行結果はコールバック関数で与 えられます。 コールバック関数の end_status が=0 ならば erinfo に応答情報が格納されているポイ ンタが返却されます。 エラーが検出された場合、(-1)が end_status にセットされます。

正常に応答メッセージを受信した場合、その中に含まれている応答情報を erinfo で指定された
TRCP_ERR_INFO 構造体の中に格納され返却されます。ユーザ側で erinfo 内の情報の処理を終えた後、その
構造体に使用されているメモリを解放してください。

解放は次のように DshFreeTRCP_ERR_INFO()関数を使って行ってください。

```
DshFreeTRCP_ERR_INFO (erinfo)
```

(5) コールバック関数

[C, C++]

```
API int APIX s15f3Callback(  
    int end_status,           // 実行結果  
    TRCP_ERR_INFO *erinfo,  // 応答情報が格納されている構造体のポインタ  
    ULONG upara              // 呼出時に指定したパラメータ  
);
```

[.NET VB]

```
Function callback_S15F3(ByVal end_status As Integer, ByRef erinfo As dsh_info.TRCP_ERR_INFO, ByVal  
upara As Integer) As Integer
```

[.NET C#]

```
int CallbackS15F3(int end_status, ref TRCP_ERR_INFO errinfo, uint upara);
```

end_status には以下の値が設定されます。

結果	意味
----	----

0	正常に送信できた。erinfo に応答情報が格納されているポインタが返却されます。
(-1)	送信エラーを検出した。
(-14)	T3タイムアウトを検出した。

3.13.2.11 EngSendS15F5() レシピ・スペース・リネーム要求送信関数

(1) 呼出書式

[C, C++]

```
API int APIX EngSendS15F5(
    char *cur_rcpid,           // 現レシピ ID 格納領域のポインタ
    char *new_rcpid,          // 新レシピ ID 格納領域のポインタ
    TRCP_ERR_INFO *erinfo,    // 応答情報格納用構造体のポインタ
    int (WINAPI *s15f5Callback)(), // 実行終了時のコールバック関数
    ULONG upara               // callback 時のパラメータ
);
```

[.NET VB]

```
Function EngSendS15F5 (
    ByVal rmnsspec As String,
    ByVal rmnews As String,
    ByRef erinfo As dsh_info.TRCR_ERR_INFO,
    ByVal callback As vcallback.callback_S15F5,
    ByVal upara As Int32) As Int32
```

[.NET C#]

```
int EngSendS15F5(
    byte[] rmnsspec,
    byte[] rmnews,
    ref TRCP_ERR_INFO erinfo,
    CallbackS15F5 callback,
    uint upara );
```

(2) 引数

cur_rcpid

現在のレシピ ID(=レシピネーム)が格納されているポインタです。

rmnscmd

新しく変えたいレシピ ID(=レシピネーム)で格納されているポインタです。

erinfo

受信した応答メッセージ S15F6 に含まれる情報を格納するための構造体領域のポインタを指定します。

s15f5Callback

DSHEng4 によるメッセージ送信処理が終了したときに呼出されるコールバック関数を指定します。ユーザは任意の関数名を指定できます。コールバックの指定が=0 の場合はブロックモードになります。

upara

コールバックされたときに引数で渡してもらうためのパラメータです。関数実行終了時にコールバックされた際、何かの判別情報として使用したい値、構造体ポインタなどを設定します。

(3) 戻り値

戻り値	意味
0	(1) ブロックモード : 正常に送信できた。 erinfo に S15F6 応答情報が返却されます。

	(2) 非「リクエスト」：要求が受け付けられた。
(-1)	送信できなかった。
(-14)	T3タイムアウトを検出した。

(4) 説明

cur_rcpid で指定されたレシピネームを new_rcpid に指定されたレシピネームにリネームするために S15F5 メッセージを送信します。

S15F6 応答メッセージ受信で得られた情報はデコードされて erinfo で指定された構造体領域に返却されます。

送信要求から S15F6 応答メッセージ受信までの制御は引数のコールバック関数指定の有無によって次のようになります。

s15f5Callback の指定	制御の流れ
なし (=0)	S15F5 送信後、応答メッセージ S15F6 が受信されるか、またはエラーを検出するまで制御が要求元に戻ってきません。 正常終了であれば、erinfo に応答情報が格納されているポインタが返却されます。
あり	送信要求後、S15F5 の送信前に制御が戻されます。実行結果はコールバック関数で与えられます。 コールバック関数の end_status が=0 ならば erinfo に応答情報が格納されているポインタが返却されます。 エラーが検出された場合、(-1)が end_status にセットされます。

正常に応答メッセージを受信した場合、その中に含まれている応答情報を erinfo で指定された TRCP_ERR_INFO 構造体の中に格納され返却されます。ユーザ側で erinfo 内の情報の処理を終えた後、その構造体に使用されているメモリを解放してください。

解放は次のように DshFreeTRCP_ERR_INFO()関数を使って行ってください。

```
DshFreeTRCP_ERR_INFO (erinfo)
```

(5) コールバック関数

[C,C++]

```
API int APIX s15f5Callback(
    int end_status,           // 実行結果
    TRCP_ERR_INFO *erinfo,   // 応答情報が格納されている構造体のポインタ
    ULONG upara              // 呼出時に指定したパラメータ
);
```

[.NET VB]

```
Function callback_S15F5(ByVal end_status As Integer, ByRef erinfo As dsh_info.TRCP_ERR_INFO, ByVal upara As Integer) As Integer
```

[.NET C#]

```
int CallbackS15F5(int status, ref TRCP_ERR_INFO errinfo, uint upara);
```

end_status には以下の値が設定されます。

結果	意味
----	----

0	正常に送信できた。erinfo に応答情報が格納されているポインタが返却されます。
(-1)	送信エラーを検出した。
(-14)	T3タイムアウトを検出した。

3.13.2.12 EngSendS15F7() レシピ・スペース取得メッセージ送信関数

(1) 呼出書式

[C, C++]

```
API int APIX EngSendS15F7(
    char *rcpid,                // レシピ ID 格納領域のポインタ
    TRCP_S15F8_INFO *erinfo,    // 応答情報格納用構造体のポインタ
    int (WINAPI *s15f7Callback)(), // 実行終了時のコールバック関数
    ULONG upara                // callback 時のパラメータ
);
```

[.NET VB]

```
Function EngSendS15F7 (
    ByVal objspec As String,
    ByRef erinfo As dsh_info.TRCP_S15F8_INFO,
    ByVal callback As vcallback.callback_S15F7,
    ByVal upara As Int32) As Int32
```

[.NET C#]

```
int EngSendS15F7(
    byte[] objspec,
    ref TRCP_S15F8_INFO erinfo,
    CallbackS15F7 callback,
    uint upara );
```

(2) 引数

rcpid

問合せたいレシピ ID が格納されているポインタです。

erinfo

受信した応答メッセージ S15F8 に含まれる情報を格納するための構造体領域のポインタを指定します。

s15f7Callback

DSEng4 によるメッセージ送信処理が終了したときに呼出されるコールバック関数を指定します。ユーザは任意の関数名を指定できます。コールバックの指定が=0 の場合はブロックモードになります。

upara

コールバックされたときに引数で渡してもらうためのパラメータです。関数実行終了時にコールバックされた際、何かの判別情報として使用したい値、構造体ポインタなどを設定します。

(3) 戻り値

戻り値	意味
0	(1) ブロックモード : 正常に送信できた。 erinfo に S15F8 応答情報が返却されます。 (2) 非ブロックモード : 要求が受け付けられた。
(-1)	送信できなかった。
(-14)	T3タイムアウトを検出した。

(4) 説明

rcpid で指定されたレシピア情報を格納するために必要な記憶容量の問い合わせをホストに対して行います。S15F7 メッセージで送信します。

S15F8 応答メッセージ受信で得られた情報はデコードされて erinfo で指定された構造体領域に返却されません。

送信要求から S15F8 応答メッセージ受信までの制御は引数のコールバック関数指定の有無によって次のようになります。

s15f7Callback の指定	制御の流れ
なし (=0)	S15F7 送信後、応答メッセージ S15F8 が受信されるか、またはエラーを検出するまで制御が要求元に戻ってきません。 正常終了であれば、erinfo に応答情報が格納されているポインタが返却されます。
あり	送信要求後、S15F7 の送信前に制御が戻されます。実行結果はコールバック関数で与えられます。 コールバック関数の end_status が=0 ならば erinfo に応答情報が格納されているポインタが返却されます。 エラーが検出された場合、(-1)が end_status にセットされます。

正常に応答メッセージを受信した場合、その中に含まれている応答情報を erinfo で指定された TRCP_S15F8_INFO 構造体の中に格納され返却されます。ユーザ側で erinfo 内の情報の処理を終えた後、その構造体で使用されているメモリを解放してください。

解放は次のように DshFreeTRCP_S15F8_INFO()関数を使って行ってください。

```
DshFreeTRCP_S15F8_INFO (erinfo)
```

(5) コールバック関数

[C, C++]

```
API int APIX s15f7Callback(
    int end_status,           // 実行結果
    TRCP_S15F8_INFO *erinfo, // 応答情報が格納されている構造体のポインタ
    ULONG upara              // 呼出時に指定したパラメータ
);
```

[.NET VB]

```
Function callback_S15F7(ByVal end_status As Integer, ByRef erinfo As dsh_info.TRCP_S15F8_INFO,
    ByVal upara As Integer) As Integer
```

[.NET C#]

```
int CallbackS15F7(int end_status, ref TRCP_S15F8_INFO errinfo, uint upara);
```

end_status には以下の値が設定されます。

結果	意味
0	正常に送信できた。erinfo に応答情報が格納されているポインタが返却されます。
(-1)	送信エラーを検出した。
(-14)	T3タイムアウトを検出した。

3.13.2.13 EngSendS15F9() レシピ・ステータス取得メッセージ送信関数

(1) 呼出書式

[C, C++]

```
API int APIX EngSendS15F9(
    char *rcpid, // レシピ ID 格納領域のポインタ
    TRCP_S15F10_INFO *erinfo, // 応答情報格納用構造体のポインタ
    int (WINAPI *s15f9Callback)(), // 実行終了時のコールバック関数
    ULONG upara // callback 時のパラメータ
);
```

[.NET VB]

```
Function EngSendS15F9 (
    ByVal rcpspec As String,
    ByRef erinfo As dsh_info.TRCPS15F10_INFO,
    ByVal callback As vcallback.callback_S15F9,
    ByVal upara As Int32) As Int32
```

[.NET C#]

```
int EngSendS15F9(
    byte[] rcpspec,
    ref TRCP_S15F10_INFO erinfo,
    CallbackS15F9 callback,
    uint upara );
```

(2) 引数

rcpid

ステータスを受信したいレシピ ID が格納されているポインタです。

erinfo

受信した応答メッセージ S15F10 に含まれる情報を格納するための構造体領域のポインタを指定します。

s15f9Callback

DSEng4 によるメッセージ送信処理が終了したときに呼出されるコールバック関数を指定します。ユーザは任意の関数名を指定できます。コールバックの指定が=0 の場合はブロックモードになります。

upara

コールバックされたときに引数で渡してもらうためのパラメータです。関数実行終了時にコールバックされた際、何かの判別情報として使用したい値、構造体ポインタなどを設定します。

(3) 戻り値

戻り値	意味
0	(1) ブロックモード : 正常に送信できた。 erinfo に S15F10 応答情報が返却されます。 (2) 非ブロックモード : 要求が受け付けられた。
(-1)	送信できなかった。
(-14)	T3タイムアウトを検出した。

(4) 説明

rcpid で指定されたレシピの状態情報をホストから取得するための関数です。S15F9 メッセージで送信します。

S15F10 応答メッセージ受信で得られた情報はデコードされて erinfo で指定された構造体領域に返却されません。

送信要求から S15F10 応答メッセージ受信までの制御は引数のコールバック関数指定の有無によって次のようになります。

s15f9Callback の指定	制御の流れ
なし	S15F9 送信後、応答メッセージ S15F10 が受信されるか、またはエラーを検出するまで制御が要求元に戻ってきません。 正常終了であれば、erinfo に S15F10 応答情報が返却されます。
あり	送信要求後、S15F9 の送信前に制御が戻されます。実行結果はコールバック関数で与えられます。 コールバック関数の end_status が=0 ならば erinfo に S15F10 応答情報が返却されます。 エラーが検出された場合、(-1)が end_status にセットされます。

正常に応答メッセージを受信した場合、その中に含まれている応答情報を erinfo で指定された TRCP_S15F10_INFO 構造体の中に格納され返却されます。ユーザ側で erinfo 内の情報の処理を終えた後、その構造体に使用されているメモリを解放してください。

解放は次のように DshFreeTRCP_S15F10_INFO()関数を使って行ってください。

```
DshFreeTRCP_S15F10_INFO (erinfo)
```

(5) コールバック関数

[C, C++]

```
API int APIX s15f9Callback(
    int end_status,           // 実行結果
    TRCP_S15F10_INFO *erinfo, // 応答情報が格納されている構造体のポインタ
    ULONG upara              // 呼出時に指定したパラメータ
);
```

[.NET VB]

```
Function callback_S15F9(ByVal end_status As Integer, ByRef erinfo As dsh_info.TRCP_S15F10_INFO,
    ByVal upara As Integer) As Integer
```

[.NET C#]

```
int CallbackS15F9(int end_status, ref TRCP_S15F10_INFO errinfo, uint upara);
```

end_status には以下の値が設定されます。

結果	意味
0	正常に送信できた。erinfo に S15F10 応答情報が返却されます。
(-1)	送信できなかった。
(-14)	T3タイムアウトを検出した。

3.13.2.14 EngSendS15F13() レシピ情報メッセージ送信関数

(1) 呼出書式

[C, C++]

```
API int WINAPI EngSendS15F13(
    TRCP_INFO *info,           // レシピ情報格納領域のポインタ
    TRCP_ERR_INFO *erinfo,    // S15F14 応答情報格納用構造体のポインタ
    int (WINAPI *s15f13Callback)(), // 実行終了時のコールバック関数
    ULONG upara               // callback 時のパラメータ
);
```

[.NET VB]

```
Function EngSendS15F13 (
    ByRef info As dsh_info.TRCP_INFO,
    ByVal updt_flag As Int32,
    ByRef erinfo As dsh_info.TRCP_ERR_INFO,
    ByVal callback As vcallback.callback_S15F13,
    ByVal upara As Int32) As Int32
```

[.NET C#]

```
int EngSendS15F13(
    ref TRCP_INFO info,
    int updt_flag,
    ref TRCP_ERR_INFO erinfo,
    CallbackS15F13 callback,
    uint upara );
```

(2) 引数

info

送信したいレシピ情報が格納されている構造体のポインタです。

erinfo

受信した応答メッセージS15F14に含まれる情報を格納するための構造体領域のポインタを指定します。

s15f13Callback

DSHEng4 によるメッセージ送信処理が終了したときに呼出されるコールバック関数を指定します。
ユーザは任意の関数名を指定できます。
コールバックの指定が=0 の場合はブロックモードになります。

upara

コールバックされたときに引数で渡してもらうためのパラメータです。関数実行終了時にコールバックされた際、何かの判別情報として使用したい値、構造体ポインタなどを設定します。

(3) 戻り値

戻り値	意味
0	(1) ブロックモード : 正常に送信できた。 erinfo に S15F14 応答情報が返却されます。 (2) 非ブロックモード : 要求が受け付けられた。
(-1)	送信できなかった。
(-14)	T3タイムアウトを検出した。

(4) 説明

レスピ情報の送信要求を行います。S15F13 メッセージで送信します。

要求を受けた DSHeng4 は、info に格納されているレスピ情報を S15F13 メッセージにエンコードし相手に送信します。

S15F14 応答メッセージ受信で得られた情報はデコードされて erinfo で指定された構造体領域に返却されま

す。

送信要求から S15F14 応答メッセージ受信までの制御は引数の S15F13 コールバック関数指定の有無によっ

s15f13Callback の指定	制御の流れ
なし (=0)	S15F13 送信後、応答メッセージ S15F14 が受信されるか、またはエラーを検出するまで制御が要求元に戻ってきません。 正常終了であれば、erinfo に S15F14 応答情報が返却されます。
あり	送信要求後、S15F13 の送信前に制御が戻されます。実行結果はコールバック関数で与えられます。 コールバック関数の end_status が=0 ならば erinfo に S15F14 応答情報が返却されます。 エラーが検出された場合、(-1) が end_status にセットされます。

正常に 応答メッセージを受信した場合、その中に含まれている 応答情報がデコードされ erinfo で指定された TRCP_ERR_INFO 構造体の中に格納され返却されます。ユーザ側で erinfo 内の情報の処理を終えた後、その構造体

に使用されているメモリを解放してください。

解放は次のように DshFreeTRCP_ERR_INFO() 関数を使って行ってください。

```
DshFreeTRCP_ERR_INFO (erinfo)
```

(5) コールバック関数

[C, C++]

```
API int APIX s15f13Callback(
    int end_status,           // 実行結果
    TRCP_ERR_INFO *erinfo,   // S15F14 応答情報格納用構造体のポインタ
    ULONG upara              // 呼出時に指定したパラメータ
);
```

[.NET VB]

```
Function callback_S15F13(ByVal end_status As Integer, ByRef erinfo As dsh_info.TRCP_ERR_INFO, ByVal upara As Integer) As Integer
```

[.NET C#]

```
int CallbackS15F13(int status, ref TRCP_ERR_INFO errinfo, uint upara);
```

end_status には以下の値が設定されます。

結果	意味
0	正常に送信できた。erinfo に S15F14 応答情報が返却されます。

(-1)	送信できなかった。
(-14)	T3タイムアウトを検出した。

3.13.2.15 EngSendS15F17() - レシビ情報問合せ関数

(1) 呼出書式

[C, C++]

```
API int WINAPI EngSendS15F17(
    char *rcpid,                // レシビ情報格納領域のポインタ
    int rcpscocode,            // レシビセクションコード
    TRCP_S15F18_INFO *erinfo,   // S15F18 応答情報格納用構造体のポインタ
    int (WINAPI *s15f17Callback)(), // 実行終了時のコールバック関数
    ULONG upara                // callback 時のパラメータ
);
```

[.NET VB]

```
Function EngSendS15F17 (
    ByVal rcpspec As String,
    ByVal rcpscocode As Int32,
    ByRef erinfo As dsh_info.TRCP_S15F18_INFO,
    ByVal callback As vcallback.callback_S15F17,
    ByVal upara As Int32) As Int32
```

[.NET C#]

```
int EngSendS15F17(
    byte[] rcpspec,
    int rcpscocode,
    ref TRCP_S15F18_INFO erinfo,
    CallbackS15F17 callback,
    uint upara );
```

(2) 引数

rcpid

ホストに問合せたいレシビ ID が格納されているポインタです。

rcpscocode

レシビのセクションを指定します。

erinfo

受信した応答メッセージ S15F18 に含まれる情報を格納するための構造体領域のポインタを指定します。

s15f17Callback

DSHEng4 によるメッセージ送信処理が終了したときに呼出されるコールバック関数を指定します。

ユーザは任意の関数名を指定できます。

コールバックの指定が=0 の場合はブロックモードになります。

upara

コールバックされたときに引数で渡してもらうためのパラメータです。関数実行終了時にコールバックされた際、何かの判別情報として使用したい値、構造体ポインタなどを設定します。

(3) 戻り値

戻り値	意味
0	(1) ブロックモード : 正常に送信できた。 erinfo に S15F18 応答情報が返却されます。

	(2) 非ブロックド：要求が受け付けられた。
(-1)	送信できなかった。
(-14)	T3タイムアウトを検出した。

(4) 説明

ホストにレシピ情報の問合せ要求を行います。S15F17 メッセージを使います。
 問合せ指定はレシピ ID で行います。
 要求を受けた DSHeng4 は、S15F17 メッセージにエンコードし、ホストに送信します。
 送信要求から S15F18 応答メッセージ受信までの制御は引数のコールバック関数指定の有無によって次のようになります。

s15f17Callback の指定	制御の流れ
なし (=0)	S15F17 送信後、応答メッセージ S15F18 が受信されるか、またはエラーを検出するまで制御が要求元に戻ってきません。 正常終了であれば、erinfo に S15F18 応答情報が返却されます。
あり	送信要求後、S15F17 の送信前に制御が戻されます。実行結果はコールバック関数で与えられます。 コールバック関数の end_status が=0 ならば erinfo に S15F18 応答情報が返却されます。 エラーが検出された場合、(-1) が end_status にセットされます。

正常に応答メッセージを受信した場合、その中に含まれている応答情報がデコードされ erinfo で指定された TRCP_S15F18_INFO 構造体の中に格納され返却されます。ユーザ側で erinfo 内の情報の処理を終えた後、その構造体で使用されているメモリを解放してください。
 解放は次のように DshFreeTRCP_S15F18_INFO() 関数を使って行ってください。

```
DshFreeTRCP_S15F18_INFO (erinfo)
```

(5) コールバック関数

[C, C++]

```
API int APIX s15f17Callback(
    int end_status,           // 実行結果
    TRCP_S15F18_INFO *erinfo, // S15F18 応答情報格納用構造体のポインタ
    ULONG upara              // 呼出時に指定したパラメータ
);
```

[.NET VB]

```
Function callback_S15F17(ByVal end_status As Integer, ByRef erinfo As dsh_info.TRCP_S15F18_INFO,
    ByVal upara As Integer) As Integer
```

[.NET C#]

```
int CallbackS15F17(int end_status, ref TRCP_S15F18_INFO errinfo, uint upara);
```

end_status には以下の値が設定されます。

結果	意味
0	正常に送信できた。erinfo に S15F18 応答情報が返却されます。
(-1)	送信エラーを検出した。
(-14)	T3タイムアウトを検出した。

3.13.3 RCP レシビ関連ライブラリ関数

3.13.3.1 DshDecodeS15F3() - S15F3 をレシビ・アクション情報にデコード

(1) 呼出書式

[C, C++]

```
API int APIX DshDecodeS15F3(
    DSHMSG *smsg, // SECS メッセージ 情報構造体のポインタ
    TRCP_ACT_INFO *pinfo // デコードした情報を格納する構造体のポインタ
);
```

[.NET VB]

```
Function DshDecodeS15F3 (
    ByRef smsg As dshdr2.DSHMSG,
    ByRef info As dsh_info.TRCP_ACT_INFO) As Int32
```

[.NET C#]

```
int DshDecodeS15F3(
    ref DSHMSG smsg,
    ref TRCP_ACT_INFO info );
```

(2) 引数

msg

S15F3 の SECS メッセージ 情報が格納されている構造体のポインタです。

pinfo

デコードしたレシビアクション情報を格納する構造体のポインタです。

(3) 戻り値

戻り値	意味
0	正常にデコードできた。
(-1)	msg を正しくデコードできなかった。

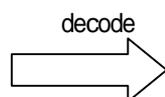
(4) 説明

S15F3 メッセージに含まれるレシビアクション情報を、ユーザプログラムが処理しやすい TRCP_ACT_INFO 構造体の中にデコードします。

なお、構造体使用後は、構造体内部で使用されたメモリを DshFreeTACT_INFO() 関数を使って開放してください。

msg S15F3

L,2
rmnspec
rmnscmd
.



3.13.3.2 DshEncodeS15F3() - レシピ・アクション情報を S15F3 へエンコード

(1) 呼出書式

[C, C++]

```
API int APIX DshEncodeS15F3(
    DSHMSG *smsg,           // SECS メッセージ 情報構造体のポインタ
    BYTE *buffer,          // S15F3 を格納するバッファポインタ
    int buflen,            // buffer のバイトサイズ
    TRCP_ACT_INFO *pinfo   // エンコードしたいアクション情報がされている構造体のポインタ
);
```

[.NET VB]

```
Function DshEncodeS15F3 (
    ByRef smsg As dshdr2.DSHMSG,
    ByRef buff As Byte,
    ByVal buflen As Int32,
    ByRef pinfo As dsh_info.TRPC_ACT_INFO) As Int32
```

[.NET C#]

```
int DshEncodeS15F3(
    ref DSHMSG smsg,
    byte[] buff,
    int buflen,
    ref TRCP_ACT_INFO info);
```

(2) 引数

smsg

エンコードした S15F3 メッセージを格納するメッセージ情報構造体のポインタです。

buffer

エンコードした S15F3 のテキストを格納するバッファポインタです。

buflen

buffer のバイトサイズです。

pinfo

エンコードしたいレシピアクション情報が格納されている構造体のポインタです。

(3) 戻り値

戻り値	意味
0	正常にエンコードできた。
(-1)	smsg を正しくエンコードできなかった。

(4) 説明

TRCP_ACT_INFO 構造体に格納されているレシピアクション情報を、S15F3 の SECS メッセージにエンコードします。

smsg S15F3

L,2

rmnsspec

rmnscmd

← encode



3.13.3.3 DshDecodeS15FRsp() - S15F4, S15F6, S15F14, S15F16, S15F18 のデコード

(1) 呼出書式

[C, C++]

```
API int APIX DshDecodeS15FRsp(
    DSHMSG *smsg, // SECS 応答メッセージ情報構造体のポインタ
    TRCP_ERR_INFO *erinfo // デコードした情報を格納する構造体のポインタ
);
```

[.NET VB]

```
Function DshDecodeS15FRsp (
    ByRef smsg As dshdr2.DSHMSG,
    ByRef erinfo As dsh_info.TRCP_ERR_INFO) As Int32
```

[.NET C#]

```
int DshDecodeS15FRsp(
    ref DSHMSG smsg,
    ref TRCP_ERR_INFO erinfo );
```

(2) 引数

smsg

S15F4, F6, F14, F16, または F18 の SECS 応答メッセージ情報が格納されている構造体のポインタです。

erinfo

デコードしたレスピ情報を格納する TRCP_ERR_INFO 構造体のポインタです。

(3) 戻り値

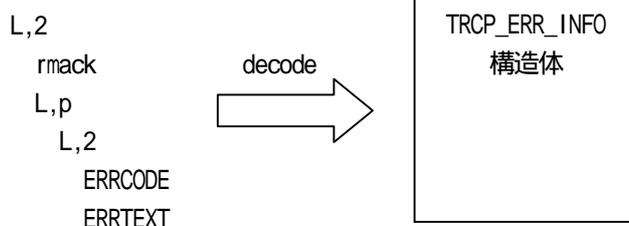
戻り値	意味
0	正常にデコードできた。
(-1)	smsg を正しくデコードできなかった。

(4) 説明

S15F3, S15F5, S15F13, S15F15, S15F17 の応答メッセージに含まれるレスピ関連応答情報を、ユーザプログラムが処理しやすい TRCP_ERR_INFO 構造体の中にデコードします。

なお、構造体使用後は、構造体内部で使用されたメモリを DshFreeTRCP_ERR_INFO()関数を使って開放してください。

smsg S15F4, F6, F14, F16, F18



3.13.3.4 DshFreeTRCP_ACT_INFO() - レシピ・アクション情報構造体メモリの開放

(1) 呼出書式

[C, C++]

```
API void APIX DshFreeTRCP_ACT_INFO(  
    TRCP_ACT_INFO *pinfo // メモリを開放したいAction情報が格納されている構造体のポインタ  
);
```

[.NET VB]

```
Sub DshFreeTRCP_ACT_INFO (  
    ByRef info As dsh_info.TRCP_ACT_INFO)
```

[.NET C#]

```
void DshFreeTRCP_ACT_INFO(  
    ref TRCP_ACT_INFO info );
```

(2) 引数

pinfo

メモリを解放したいレシピアクション情報構造体のポインタです。

(3) 戻り値

なし。

(4) 説明

TRCP_ACT_INFO 構造体内で情報格納用に使用されているメモリを全て解放します。

開放した後、TLIMIT_INFO の内容を全て0で初期設定します。

pinfo が NULL ならば、何も処理しません。

開放した後、TTRCP_ACT_INFO の内容を全て0で初期設定します。

pinfo が NULL ならば、何も処理しません。

3.13.3.5 DshCopyTRCP_ACT_INFO() - レシピ・アクション情報構造体メモリのコピー

(1) 呼出書式

[C, C++]

```
API int APIX DshCopyTRCP_ACT_INFO(
    TRCP_ACT_INFO *dinfo,           // 北°-先のポインタ
    TRCP_ACT_INFO *sinfo           // 北°-元のポインタ
);
```

[.NET VB]

```
Function DshCopyTRCP_ACT_INFO (
    ByRef dinfo As dsh_info.TRCP_ACT_INFO,
    ByRef sinfo As dsh_info.TRCP_ACT_INFO) As Int32
```

[.NET C#]

```
int DshCopyTRCP_ACT_INFO(
    ref TRCP_ACT_INFO dinfo,
    ref TRCP_ACT_INFO sinfo);
```

(2) 引数

pinfo

レシピアクション情報のコピー先構造体メモリのポインタです。

sinfo

コピー元のレシピアクション情報が格納されている構造体メモリのポインタです。

(3) 戻り値

戻り値	意味
0	正常に北°-できた。
(-1)	sinfo または dinfo の値が NULL だったので北°-できなかった。

(4) 説明

sinfo が指す TRCP_ACT_INFO 構造体内に格納されているレシピアクション情報を dinfo で指定された TRCP_ACT_INFO 構造体にコピーします。

dinfo 内のメンバーで新しいメモリが必要なものは本関数が取得します。

dinfo 内メンバーで確保されたメモリは、使用后、DshFreeTRCP_ACT_INFO() 関数を使って開放してください。

3.13.3.6 DshMakeS15F3Response() - S15F3 の応答メッセージの生成

(1) 呼出書式

[C, C++]

```
API int APIX DshMakeS15F3Response(
    TRCP_ACTION_INFO *info,           // レシビ・アクション情報格納領域のポインタ
    TRCP_ERR_INFO *erinfo,           // S15F4 に設定する応答情報格納領域のポインタ
    DSHMSG *msg,                      // S15F4 メッセージを格納するメッセージ構造体のポインタ
    BYTE *buff,                       // S15F4 のテキスト格納バッファポインタ
    int buff_size                      // buff のバイトサイズ
);
```

[.NET VB]

```
Function DshMakeS15F3Response (
    ByRef pinfo As dsh_info.TRCP_ACT_INFO,
    ByRef erinfo As dsh_info.TRCP_ERR_INFO,
    ByRef msg As dshdr2.DSHMSG,
    ByRef buff As Byte,
    ByVal buff_size As Int32) As Int32
```

[.NET C#]

```
int DshMakeS15F3Response(
    ref TRCP_ACT_INFO pinfo,
    ref TRCP_ERR_INFO erinfo,
    ref DSHMSG msg,
    byte[] buff,
    int buff_size );
```

(2) 引数

info

レシビ・アクション情報が格納されている領域のポインタです。

erinfo

S15F4 メッセージに設定する応答情報が格納されている領域のポインタです。

msg

S15F4 応答メッセージ情報を格納するためのメッセージ構造体のポインタです。

buff

S15F4 応答メッセージのテキストを格納するためのバッファポインタです。

buff_size

buff のバイトサイズです。

(3) 戻り値

戻り値	意味
0	正常に生成できた。
(-1)	生成できなかった。(buff 領域不足)

(4) 説明

S15F3 に対する S15F4 応答メッセージを sinfo に含まれるレシビアクション情報と応答情報に従って作成します。

応答情報内の、rmack を S15F4 の RMACK として設定します。
rmack はユーザがレシピ・アクション情報を評価した結果です。

erinfo 応答情報の初期設定と情報設定にはそれぞれ DshInitTOBJ_ERR_INFO()、DshPutTOBJ_ERR_INFO()関数を使用することができます。(3.21 を参照してください)

3.13.3.7 DshDecodeS15F5() - S15F5 をレシビ・リネーム情報にデコード

(1) 呼出書式

[C, C++]

```
API int APIX DshDecodeS15F5(
    DSHMSG *msg, // SECS メッセージ 情報構造体のポインタ
    TRCP_RENAME_INFO *pinfo // デコードした情報を格納する構造体のポインタ
);
```

[.NET VB]

```
Function DshDecodeS15F5 (
    ByRef msg As dshdr2.DSHMSG,
    ByRef info As dsh_info.TRCP_RENAME_INFO) As Int32
```

[.NET C#]

```
int DshDecodeS15F5(
    ref DSHMSG msg,
    ref TRCP_RENAME_INFO info );
```

(2) 引数

msg

S15F5 の SECS メッセージ 情報が格納されている構造体のポインタです。

pinfo

デコードしたレシビ・リネーム情報を格納する構造体のポインタです。

(3) 戻り値

戻り値	意味
0	正常にデコードできた。
(-1)	msg を正しくデコードできなかった。

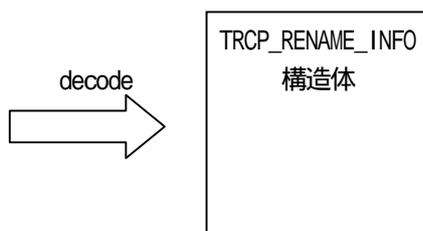
(4) 説明

S15F5 メッセージに含まれるレシビ・リネーム情報を、ユーザプログラムが処理しやすい TRCP_RENAME_INFO 構造体の中にデコードします。

なお、構造体使用後は、構造体内部で使用されたメモリを DshFreeTRCP_RENAME_INFO()関数を使って開放してください。

msg S15F5

```
L,2
rmnspec
rmnscmd
.
```



3.13.3.8 DshFreeTRCP_RENAME_INFO() - レシピ・リネーム情報構造体メモリの開放

(1) 呼出書式

[C, C++]

```
API void APIX DshFreeTRCP_RENAME_INFO(
    TRCP_RENAME_INFO *pinfo // メモリを開放したいリネーム情報が格納されている構造体のポインタ
);
```

[.NET VB]

```
Sub DshFreeTRCP_RENAME_INFO (
    ByRef info As dsh_info.TRCP_RENAME_INFO)
```

[.NET C#]

```
void DshFreeTRCP_RENAME_INFO(
    ref TRCP_RENAME_INFO info );
```

(2) 引数

pinfo

メモリを解放したいレシピ・リネーム情報構造体のポインタです。

(3) 戻り値

なし。

(4) 説明

TRCP_RENAME_INFO 構造体内で情報格納用に使用されているメモリを全て解放します。

開放した後、TRCP_RENAME_INFO の内容を全て 0 で初期設定します。

pinfo が NULL ならば、何も処理しません。

3.13.3.9 DshCopyTRCP_RENAME_INFO() - レシピ・リネーム情報構造体メモリのコピー

(1) 呼出書式

[C, C++]

```
API int APIX DshCopyTRCP_RENAME_INFO(
    TRCP_RENAME_INFO *dinfo,           // 北°-先のポインタ
    TRCP_RENAME_INFO *sinfo           // 北°-元のポインタ
);
```

[.NET VB]

```
Function DshCopyTRCP_RENAME_INFO Lib (
    ByRef dinfo As dsh_info.TRCP_RENAME_INFO,
    ByRef sinfo As dsh_info.TRCP_RENAME_INFO) As Int32
```

[.NET C#]

```
int DshCopyTRCP_RENAME_INFO(
    ref TRCP_RENAME_INFO dinfo,
    ref TRCP_RENAME_INFO sinfo);
```

(2) 引数

pinfo

レシピ・リネーム情報のコピー先構造体メモリのポインタです。

sinfo

コピー元のレシピ・リネーム情報が格納されている構造体メモリのポインタです。

(3) 戻り値

戻り値	意味
0	正常に北°-できた。
(-1)	sinfo または dinfo の値が NULL だったので北°-できなかった。

(4) 説明

sinfo が指す TRCP_RENAME_INFO 構造体内に格納されているレシピ・リネーム情報を dinfo で指定された TRCP_RENAME_INFO 構造体にコピーします。

dinfo 内のメンバーで新しいメモリが必要なものは本関数が取得します。

dinfo 内メンバーで確保されたメモリは、使用后、DshFreeTRCP_RENAME_INFO()関数を使って開放してください。

3.13.3.10 DshMakeS15F5Response() - S15F5 の応答メッセージの生成

(1) 呼出書式

[C, C++]

```
API int APIX DshMakeS15F5Response(
    TRCP_RENAME_INFO *info,           // レシビ・リネーム情報格納領域のポインタ
    TRCP_ERR_INFO *erinfo,           // S15F6 に設定する応答情報が格納領域のポインタ
    DSHMSG *msg,                      // S15F6 メッセージを格納するメッセージ構造体のポインタ
    BYTE *buff,                       // S15F6 のテキスト格納バッファポインタ
    int buff_size                     // buff のバイトサイズ
);
```

[.NET VB]

```
Function DshMakeS15F5Response (
    ByRef pinfo As dsh_info.TRCP_RENAME_INFO,
    ByRef erinfo As dsh_info.TRCP_ERR_INFO,
    ByRef msg As dshdr2.DSHMSG,
    ByRef buff As Byte,
    ByVal buff_size As Int32) As Int32
```

[.NET C#]

```
int DshMakeS15F5Response(
    ref TRCP_RENAME_INFO pinfo,
    ref TRCP_ERR_INFO erinfo,
    ref DSHMSG msg,
    byte[] buff,
    int buff_size );
```

(2) 引数

info

レシビ・リネーム情報が格納されている領域のポインタです。

erinfo

S15F6 メッセージに設定する応答情報が格納されている領域のポインタです。

msg

S15F6 応答メッセージ情報を格納するためのメッセージ構造体のポインタです。

buff

S15F6 応答メッセージのテキストを格納するためのバッファポインタです。

buff_size

buff のバイトサイズです。

(3) 戻り値

戻り値	意味
0	正常に生成できた。
(-1)	生成できなかった。(buff 領域不足)

(4) 説明

S15F5 に対する S15F6 応答メッセージを info に含まれるレシビ・リネーム情報と erinfo に含まれる応答情報に従って作成します。



応答情報内の rmack を S15F6 の RMACK として設定します。
rmack はユーザがレシピ・リネーム情報を評価した結果です。

erinfo 応答情報の初期設定と情報設定にはそれぞれ DshInitTOBJ_ERR_INFO()、DshPutTOBJ_ERR_INFO()関数を使用することができます。

3.13.3.11 DshDecodeS15F8() - S15F8 応答メッセージのデコード

(1) 呼出書式

[C, C++]

```
API int APIX DshDecodeS15F8(
    DSHMSG *smsg, // SECS 応答メッセージ情報構造体のポインタ
    TRCP_S15F8_INFO *erinfo // デコードした情報を格納する構造体のポインタ
);
```

[.NET VB]

```
Function DshDecodeS15F8 (
    ByRef smsg As dshdr2.DSHMSG,
    ByRef erinfo As dsh_info.TRCP_S15F8_INFO) As Int32
```

[.NET C#]

```
int DshDecodeS15F8(
    ref DSHMSG smsg,
    ref TRCP_S15F8_INFO erinfo );
```

(2) 引数

smsg

S15F8 の SECS 応答メッセージ情報が格納されている構造体のポインタです。

erinfo

デコードしたレシプスペースデータを格納する TRCP_S15F8_INFO 構造体のポインタです。

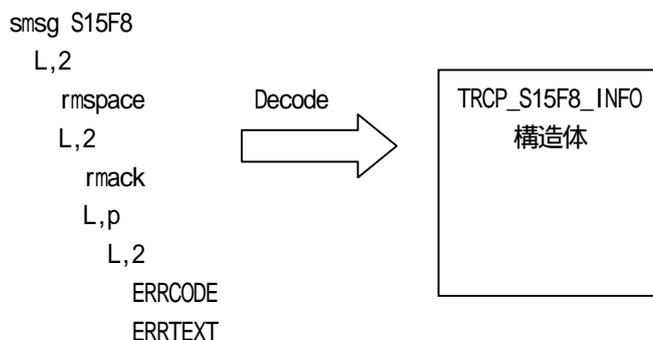
(3) 戻り値

戻り値	意味
0	正常にデコードできた。
(-1)	smsg を正しくデコードできなかった。

(4) 説明

S15F7 の応答メッセージに含まれるレシプスペースデータの情報を、ユーザプログラムが処理しやすい TRCP_S15F8_INFO 構造体の中にデコードします。

なお、構造体使用後は、構造体内部で使用されたメモリを DshFreeTRCP_S15F8_INFO()関数を使って開放してください。



3 . 13 . 3 . 12 DshFreeTRCP_S15F8_INFO() S15F8 応答情報構造体メモリの開放

(1) 呼出書式

[C, C++]

```
API void APIX DshFreeTRCP_S15F8_INFO(
    TRCP_S15F8_INFO *erinfo // メモリを開放したい応答情報が格納されている構造体のポインタ
);
```

[.NET VB]

```
Sub DshFreeTRCP_S15F8_INFO (
    ByRef info As dsh_info.TRCP_S15F8_INFO)
```

[.NET C#]

```
void DshFreeTRCP_S15F8_INFO(
    ref TRCP_S15F8_INFO info );
```

(2) 引数

erinfo

メモリを解放したいレシビ関連応答情報構造体のポインタです。

(3) 戻り値

なし。

(4) 説明

S15F7 の応答メッセージデコード時に TRCP_S15F8_INFO 構造体内で情報格納用に割当て使用されているメモリを全て解放します。

開放した後、TRCP_S15F8_INFO の内容を全て 0 で初期設定します。

erinfo が NULL ならば、何も処理しません。

3.13.3.13 DshDecodeS15F10() - S15F10 応答メッセージのデコード

(1) 呼出書式

[C, C++]

```
API int APIX DshDecodeS15F10(
    DSHMSG *smsg, // SECS 応答メッセージ情報構造体のポインタ
    TRCP_S15F10_INFO *erinfo // デコードした情報を格納する構造体のポインタ
);
```

[.NET VB]

```
Function DshDecodeS15F10 (
    ByRef smsg As dshdr2.DSHMSG,
    ByRef info As dsh_info.TRCP_S15F10_INFO) As Int32
```

[.NET C#]

```
int DshDecodeS15F10(
    ref DSHMSG smsg,
    ref TRCP_S15F10_INFO info );
```

(2) 引数

smsg

S15F10 の SECS 応答メッセージ情報が格納されている構造体のポインタです。

erinfo

デコードしたレスピ情報を格納する TRCP_S15F10_INFO 構造体のポインタです。

(3) 戻り値

戻り値	意味
0	正常にデコードできた。
(-1)	smsg を正しくデコードできなかった。

(4) 説明

S15F9 の応答メッセージに含まれるレスピ関連メッセージの情報を、ユーザプログラムが処理しやすい TRCP_S15F10_INFO 構造体の中にデコードします。

なお、構造体使用後は、構造体内部で使用されたメモリを DshFreeTRCP_S15F10_INFO() 関数を使って開放してください。

smsg S15F10

L,3

rcpstat

rcpvers

L,2

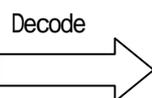
rmack

L,p

L,2

ERRCODE

ERRTEXT



Decode

TRCP_S15F10_INFO
構造体

3 . 13 . 3 . 14 DshFreeTRCP_S15F10_INFO() S15F10 応答情報構造体メモリの開放

(1) 呼出書式

[C, C++]

```
API void APIX DshFreeTRCP_S15F10_INFO(
    TRCP_S15F10_INFO *erinfo // メモリを開放したい応答情報が格納されている構造体のポインタ
);
```

[.NET VB]

```
Sub DshFreeTRCP_S15F10_INFO (
    ByRef info As dsh_info.TRCP_S15F10_INFO)
```

[.NET C#]

```
void DshFreeTRCP_S15F10_INFO(
    ref TRCP_S15F10_INFO info );
```

(2) 引数

erinfo

メモリを解放したいレシビ関連応答情報構造体のポインタです。

(3) 戻り値

なし。

(4) 説明

S15F9 の応答メッセージデコード時に TRCP_S15F10_INFO 構造体内で情報格納用に割当て使用されているメモリを全て解放します。

開放した後、TRCP_S15F10_INFO の内容を全て 0 で初期設定します。

erinfo が NULL ならば、何も処理しません。

3.13.3.15 DshDecodeS15F13() - S15F13 をレシビ情報にデコード

(1) 呼出書式

[C, C++]

```
API int APIX DshDecodeS15F13(
    DSHMSG *msg, // SECS メッセージ 情報構造体のポインタ
    TRCP_INFO *pinfo // デコードした情報を格納する構造体のポインタ
);
```

[.NET VB]

```
Function DshDecodeS15F13 (
    ByRef msg As dshdr2.DSHMSG,
    ByRef info As dsh_info.TRCP_INFO,
    ByRef updt_flag As Int32) As Int32
```

[.NET C#]

```
int DshDecodeS15F13Ext(
    ref DSHMSG msg,
    ref TRCP_INFO tinfo,
    ref int updt_flag );
```

(2) 引数

msg

S15F13 の SECS メッセージ 情報が格納されている構造体のポインタです。

pinfo

デコードしたレシビ情報を格納する構造体のポインタです。

(3) 戻り値

戻り値	意味
0	正常にデコードできた。
(-1)	msg を正しくデコードできなかった。

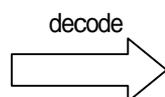
(4) 説明

S15F13 メッセージに含まれるレシビ情報を、ユーザプログラムが処理しやすい TRCP_INFO 構造体の中にデコードします。

なお、構造体使用後は、構造体内部で使用されたメモリを DshFreeTRCP_INFO()関数を使って開放してください。

msg S15F13

```
L,5
dataid
rcpupdt
rcpspec
.
```



3.13.3.16 DshEncodeS15F13() - レシビ情報を S15F13 へエンコード

(1) 呼出書式

[C, C++]

```
API int APIX DshEncodeS15F13(
    DSHMSG *smsg,           // SECS メッセージ 情報構造体のポインタ
    BYTE *buffer,         // S15F13 を格納するバッファポインタ
    int buflen,           // buffer のバイトサイズ
    TRCP_INFO *pinfo       // エンコードしたいレシビ情報が格納されている構造体のポインタ
);
```

[.NET VB]

```
Function DshEncodeS15F13 (
    ByRef smsg As dshdr2.DSHMSG,
    ByRef buff As Byte,
    ByVal buflen As Int32,
    ByRef rinfo As dsh_info.TRCP_INFO,
    ByVal updt_flag As Int32) As Int32
```

[.NET C#]

```
int DshEncodeS15F13(
    ref DSHMSG smsg,
    byte[] buff,
    int buflen,
    ref TRCP_INFO rinfo,
    int updt_flag );
```

(2) 引数

smsg
エンコードした S15F13 メッセージを格納するメッセージ情報構造体のポインタです。

buffer
エンコードした S15F13 のテキストを格納するバッファポインタです。

buflen
buffer のバイトサイズです。

pinfo
エンコードしたいレシビ情報が格納されている構造体のポインタです。

(3) 戻り値

戻り値	意味
0	正常にエンコードできた。
(-1)	smsg を正しくエンコードできなかった。

(4) 説明

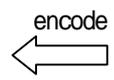
TRCP_INFO 構造体に格納されているレシビ情報を、S15F13 の SECS メッセージにエンコードします。

smsg S15F13

L,4



rcpid
mdlIn
.



3.13.3.17 DshFreeTRCP_INFO() - レシピ情報構造体メモリの開放

(1) 呼出書式

[C, C++]

```
API void APIX DshFreeTRCP_INFO(  
    TRCP_INFO *pinfo           // メモリを開放したい RCP 情報が格納されている構造体のポインタ  
);
```

[.NET VB]

```
Sub DshFreeTRCP_INFO (  
    ByRef info As dsh_info.TRCP_INFO)
```

[.NET C#]

```
void DshFreeTRCP_INFO(  
    ref TRCP_INFO info );
```

(2) 引数

pinfo

メモリを解放したいレシピ情報構造体のポインタです。

(3) 戻り値

なし。

(4) 説明

TRCP_INFO 構造体内で情報格納用に使用されているメモリを全て解放します。

開放した後、TRCP_INFO の内容を全て 0 で初期設定します。

pinfo が NULL ならば、何も処理しません。

3.13.3.18 DshCopyTRCP_INFO() - レシピ情報構造体メモリのコピー

(1) 呼出書式

[C, C++]

```
API int APIX DshCopyTRCP_INFO(
    TRCP_INFO *dinfo,           // 北°-先のポインタ
    TRCP_INFO *sinfo           // 北°-元のポインタ
);
```

[.NET VB]

```
Function DshCopyTRCP_INFO (
    ByRef info As dsh_info.TRCP_INFO,
    ByRef sinfo As dsh_info.TRCP_INFO) As Int32
```

[.NET C#]

```
int DshCopyTRCP_INFO(
    ref TRCP_INFO info,
    ref TRCP_INFO sinfo );
```

(2) 引数

dinfo

レシピ情報のコピー先構造体メモリのポインタです。

sinfo

コピー元のレシピ情報が格納されている構造体メモリのポインタです。

(3) 戻り値

戻り値	意味
0	正常に北°-できた。
(-1)	sinfo または dinfo の値が NULL だったので北°-できなかった。

(4) 説明

sinfo が指す TRCP_INFO 構造体内に格納されているレシピ情報を dinfo が指定する TRCP_INFO 構造体にコピーします。

dinfo 内のメンバーで新しいメモリが必要なものは本関数が取得します。

dinfo 内メンバーで確保されたメモリは、使用后、DshFreeTRCP_INFO()関数を使って開放してください。

3.13.3.19 DshInitRcpInfo レシピ TRCP_INFO の初期設定

(1) 呼出書式

[C, C++]

```
API int APIX DshInitRcpInfo(
    TRCP_INFO *info,           // TRCP_INFO 構造体のポインタ
    char *rcpspec,            // レシピ ID
    char *rcpbody,            // RCPBODY 名
    int para_count            // コマンドコード情報リストのサイズ
);
```

[.NET VB]

```
Sub DshInitRcpInfo (
    ByRef info As dsh_info.TRCP_INFO,
    ByVal rcpspec As String,
    ByVal rcpbody As String,
    ByVal para_count As Int32)
```

[.NET C#]

```
void DshInitRcpInfo(
    ref TRCP_INFO info,
    byte[] rcpspec,
    byte[] rcpbody,
    int para_count );
```

(2) 引数

info
レシピ TRCP_INFO 構造体のポインタです。このメンバーを初期設定します。

rcpspec
設定するレシピ ID (文字列) です。

rcpbody
レシピの RCPBODY 情報です。

para_count
TRCP_INFO に含まれるパラメータの最大数です。

(3) 戻り値

なし。

(4) 説明

本関数は APP が OFFLINE でレシピ情報を生成する際に使用することができます。最初に info 内をクリアします。そして、引数で指定された情報を info 内に設定します。メモリが必要なメンバーについてはメモリを確保し情報をコピーします。

また、レシピのパラメータ情報の設定については次の DshPutRcpPara()関数を使ってください。

TRCP_INFO 構造体の使用後、DshFreeTRCP_INFO()関数を使って構造体内部で使用したメモリを開放してください。

3.13.3.20 DshPutRcpPara() レシピパラメータ情報の追加

(1) 呼出書式

[C, C++]

```
API int APIX DshPutRcpPara(
    TRCP_INFO *info,           // レシピ情報構造体のポインタ
    char *rcpparm,           // レシピパラメータ名
    int para_fmt,           // para 値の format
    int para_size,         // para 値の配列サイズ
    void *rcpparval         // para 値が格納されている領域のポインタ
);
```

[.NET VB]

```
Function DshPutRcpPara (
    ByRef info As dsh_info.TRCP_INFO,
    ByVal rcpparm As String,
    ByVal para_fmt As Int32,
    ByVal para_size As Int32,
    ByVal rcpparval As IntPtr) As Int32
```

[.NET C#]

```
int DshPutRcpPara(
    ref TRCP_INFO info,
    byte[] rcpparm,
    int para_fmt,
    int para_size,
    byte[] rcpparval );
```

(2) 引数

info

レシピ情報構造体のポインタです。

rcpparm

レシピパラメータの名前です。

para_fmt

パラメータ値のフォーマットです。

para_size

パラメータ値の配列サイズです。

rcpparval

パラメータ値が格納されている領域のポインタです。

(3) 戻り値

戻り値	意味
0	正常に追加できた。
(-1)	パラメータの数が指定数を超えている。

(4) 説明

先に DshInitRcpInfo() で初期設定されたレシピパラメータ情報リスト para_list にパラメータ情報 TRCP_PARA を 1 個加えます。

para_list 内のパラメータ情報リスト上に、本関数が実行される順に情報を追加していきます。
設定後 0 を返却します。

もし、info 内の para_count で指定された分のパラメータが既に設定済みであった場合は、(-1)を返却します。

3.13.3.21 DshMakeS15F13Response() - S15F13 の応答メッセージの生成

(1) 呼出書式

[C, C++]

```
API int APIX DshMakeS15F13Response(
    TRCP_INFO *info,           // レシ情報格納領域ポインタ
    TRCP_ERR_INFO *erinfo,    // S15F14 に設定する応答情報格納領域のポインタ
    DSHMSG *smsg,             // S15F14 メッセージを格納するメッセージ構造体のポインタ
    BYTE *buff,               // S15F14 のテキスト格納バッファポインタ
    int buff_size             // buff のバイトサイズ
);
```

[.NET VB]

```
Function DshMakeS15F13Response (
    ByRef pinfo As dsh_info.TRCR_INFO,
    ByRef erinfo As dsh_info.TRCR_ERR_INFO,
    ByRef smsg As dshdr2.DSHMSG,
    ByRef buff As Byte,
    ByVal buff_size As Int32) As Int32
```

[.NET C#]

```
int DshMakeS15F13Response(
    ref TRCP_INFO pinfo,
    ref TRCP_ERR_INFO erinfo,
    ref DSHMSG smsg,
    byte[] buff,
    int buff_size );
```

(2) 引数

info

レシ情報が格納されている領域のポインタです。

erinfo

S15F14 メッセージに設定する応答情報が格納されている領域のポインタです。

msg

S15F14 応答メッセージ情報を格納するためのメッセージ構造体のポインタです。

buff

S15F14 応答メッセージのテキストを格納するためのバッファポインタです。

buff_size

buff のバイトサイズです。

(3) 戻り値

戻り値	意味
0	正常に生成できた。
(-1)	生成できなかった。(buff 領域不足)

(4) 説明

S15F13 に対する S15F14 応答メッセージを info に含まれるレシ情報と erinfo に含まれる応答情報に従って作成します。

応答情報に含まれている rmack を S15F14 の RMACK として設定します。
rmack はユーザがレシビ情報を評価した結果です。

erinfo 応答情報の初期設定と情報設定にはそれぞれ DshInitTOBJ_ERR_INFO()、DshPutTOBJ_ERR_INFO()関数を使用することができます。(3.21 を参照してください)

3.13.3.22 DshFreeTRCP_ERR_INFO() - レシビ関連応答情報構造体メモリの開放

(1) 呼出書式

[C, C++]

```
API void APIX DshFreeTRCP_ERR_INFO(
    TRCP_ERR_INFO *erinfo // メモリを開放したいPP情報が格納されている構造体のポインタ
);
```

[.NET VB]

```
Sub DshFreeTRCP_ERR_INFO (
    ByRef info As dsh_info.TRCP_ERR_INFO)
```

[.NET C#]

```
void DshFreeTRCP_ERR_INFO(
    ref TRCP_ERR_INFO info );
```

(2) 引数

erinfo

メモリを解放したいレシビ関連応答情報構造体のポインタです。

(3) 戻り値

なし。

(4) 説明

S15F13 などの応答メッセージデコード時に TRCP_ERR_INFO 構造体内で情報格納用に割当て使用されているメモリを全て解放します。

開放した後、TRCP_ERR_INFO の内容を全て 0 で初期設定します。

erinfo が NULL ならば、何も処理しません。

3.13.3.23 DshDecodeS15F18() - S15F18 応答メッセージのデコード

(1) 呼出書式

[C, C++]

```
API int APIX DshDecodeS15F18(
    DSHMSG *smsg, // SECS 応答メッセージ情報構造体のポインタ
    TRCP_S15F18_INFO *erinfo // デコードした情報を格納する構造体のポインタ
);
```

[.NET VB]

```
Function DshDecodeS15F18 (
    ByRef smsg As dshdr2.DSHMSG,
    ByRef info As dsh_info.TRPCP_S15F18_INFO) As Int32
```

[.NET C#]

```
int DshDecodeS15F18(
    ref DSHMSG smsg,
    ref TRCP_S15F18_INFO info );
```

(2) 引数

smsg

S15F18 の SECS 応答メッセージ情報が格納されている構造体のポインタです。

erinfo

デコードしたレシピ情報を格納する TRCP_S15F18_INFO 構造体のポインタです。

(3) 戻り値

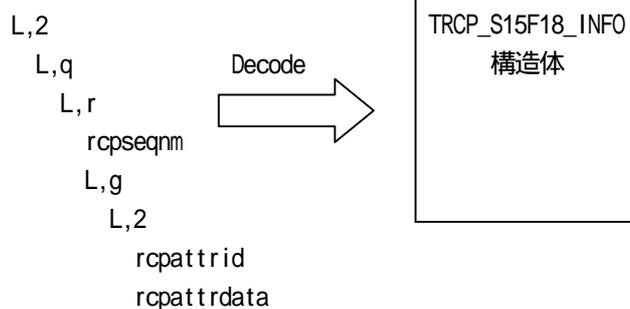
戻り値	意味
0	正常にデコードできた。
(-1)	smsg を正しくデコードできなかった。

(4) 説明

S15F17 の応答メッセージに含まれるレシピ関連メッセージの情報を、ユーザプログラムが処理しやすい TRCP_S15F18_INFO 構造体の中にデコードします。

なお、構造体使用後は、構造体内部で使用されたメモリを DshFreeTRCP_S15F18_INFO() 関数を使って開放してください。

smsg S15F18



3.13.3.24 DshFreeTRCP_S15F18_INFO() S15F18 応答情報構造体メモリの開放

(1) 呼出書式

[C, C++]

```
API void APIX DshFreeTRCP_S15F18_INFO(
    TRCP_S15F18_INFO *erinfo // メモリを開放したい応答情報が格納されている構造体のポインタ
);
```

[.NET VB]

```
Sub DshFreeTRCP_S15F18_INFO (
    ByRef info As dsh_info.TRCP_S15F18_INFO)
```

[.NET C#]

```
void DshFreeTRCP_S15F18_INFO(
    ref TRCP_S15F18_INFO info );
```

(2) 引数

erinfo

メモリを解放したいレシビ関連応答情報構造体のポインタです。

(3) 戻り値

なし。

(4) 説明

S15F17の応答メッセージデコード時にTRCP_S15F18_INFO構造体内で情報格納用に割当て使用されているメモリを全て解放します。

開放した後、TRCP_S15F18_INFOの内容を全て0で初期設定します。

erinfoがNULLならば、何も処理しません。

3.13.3.25 DshInitTRCP_ERR_INFO () レシピア答情報の初期化

(1) 呼出書式

[C, C++]

```
API void APIX DshInitTRCP_ERR_INFO(
    TRCP_ERR_INFO *info,           // エラ-情報格納構造体リストのポインタ
    int rmack,                     // ack データ
    int err_count                  // エラ-情報のリストサイズ (個数 0,1,2...)
);
```

[.NET VB]

```
Sub DshInitTRCP_ERR_INFO (
    ByRef rcp_errinfo As dsh_info.TRCP_ERR_INFO,
    ByVal rmack As Int32,
    ByVal errcount As Int32)
```

[.NET C#]

```
void DshInitTRCP_ERR_INFO(
    ref TRCP_ERR_INFO rcp_errinfo,
    int rmack,
    int errcount );
```

(2) 引数

info

TRCP_ERR_INFO 応答情報構造体のポインタです。

rmack

rmack - ACK の値です。

err_count

エラー情報構造体の数です。 = 0 の場合はエラー情報がないことになります。

(3) 戻り値

なし。

(4) 説明

本関数は、レシピア関連応答メッセージ TRCP_ERR_INFO 構造体に初期設定を行います。

info で指定された構造体の rmack メンバーに引数 rmack の値を設定し、err_count メンバーにも引数 err_count の値を設定します。

もし、err_count > 0 の場合は、err_list に err_count だけの TRCP_ERR_INFO エラー情報構造体のポインタリストを設けます。

3.13.3.26 DshPutTRCP_ERR_INFO () レシピエラー情報の設定

(1) 呼出書式

[C, C++]

```
API void APIX DshPutTRCP_ERR_INFO (
    TRCP_ERR_INFO *errinfo,          // エラー情報格納構造体リストのポインタ
    int errcode,                    // error code
    char *errtext                    // error text
);
```

[.NET VB]

```
Function DshPutTRCP_ERR_INFO (
    ByRef errinfo As dsh_info.TRCP_ERR_INFO,
    ByVal errcode As Int32,
    ByVal errtext As String) As Int32
```

[.NET C#]

```
int DshPutTRCP_ERR_INFO(
    ref TRCP_ERR_INFO errinfo,
    int errcode,
    byte[] errtext );
```

(2) 引数

errinfo

レシピエラー情報構造体のポインタです。

errcode

設定するエラーコードです。(メッセージ内のアイテムは U1(51)です。)

errtext

設定するエラーテキストが格納されている領域のポインタです。

(3) 戻り値

戻り値	意味
0	正常に設定できた。
(-1)	リストが満杯で設定できなかった。

(4) 説明

本関数は、errinfo 内の errlist リストの先頭から空きリストを探します。

もし、空きリストがなければ、(-1)を返却します。

もし、空きリストがあれば、その空きリストに1個 TRCP_ERR_INFO 構造体領域を設け、その構造体に errcode と errtext を格納し、0 を返却します。TRCP_ERR_INFO と内部メンバーのメモリは本関数が取得します。

本関数の実行前に DshInitTRCP_ERR_INFO()関数を使って errinfo を初期化しておく必要があります。

3.13.4 ユーザ作成ライブラリ関数

3.13.4.1 DshResponseS15F4() S15F4 レシビネームスペースアクション応答メッセージ

(1) 呼出書式

[C, C++]

```
API int APIX DshResponseS15F4(
    ID_TR trid, // DSHDR2 のトランザクション ID
    TRCP_ACT_INFO *info, // レシビネームスペースアクションメッセージ格納ポインタ
    TRCP_ERR_INFO *erinfo // S15F4 応答情報格納用構造体のポインタ
);
```

[.NET VB]

```
Function DshResponseS15F4 (
    ByVal trid As Int32,
    ByRef info As dsh_info.TRCP_ACT_INFO,
    ByRef erinfo As dsh_info.TRCP_ERR_INFO) As Int32
```

[.NET C#]

```
int DshResponseS15F4(
    uint trid,
    ref TRCP_ACT_INFO info,
    ref TRCP_ERR_INFO erinfo );
```

(2) 引数

trid

S15F3 信時に DSHEng4 から渡される DSHDR2 通信ドライバーのトランザクション管理のための ID です。

info

レシビネームスペースアクションの表示メッセージ情報が格納されているポインタです。

erinfo

送信する応答メッセージのための応答情報が格納されている構造体のポインタです。

(3) 戻り値

戻り値	意味
0	正常に送信できた。
(-1)	送信できなかった。

(4) 説明

レシビネームスペースアクションメッセージ S15F3 に対する応答メッセージを送信します。

本関数はユーザ作成ライブラリ DLL(dsh_ulib.dll)に含まれる関数ですが、ここでは DSHEng4 パッケージに標準的な関数として付属されているものです。(ユーザ独自による作成も可能です)

引数に指定されている erinfo 情報から S15F4 メッセージを組み立て、その後、S15F4 メッセージを送信します。

なお、S15F4 メッセージの組み立てに、DshMakeS15F3Response()関数を使用できます。

3.13.4.2 DshResponseS15F6() S15F6 レシビネームスペースリネーム応答メッセージ

(1) 呼出書式

[C, C++]

```
API int APIX DshResponseS15F6(
    ID_TR trid,                // DSHDR2 のトランザクション ID
    TRCP_RENAME_INFO *info,   // レシビネームスペースリネームメッセージ格納ポインタ
    TRCP_ERR_INFO *erinfo     // S15F6 応答情報格納用構造体のポインタ
);
```

[.NET VB]

```
Function DshResponseS15F6 (
    ByVal trid As Int32,
    ByRef info As dsh_info.TRCP_RENAME_INFO,
    ByRef erinfo As dsh_info.TRCP_ERR_INFO) As Int32
```

[.NET C#]

```
int DshResponseS15F6(
    uint trid,
    ref TRCP_RENAME_INFO info,
    ref TRCP_ERR_INFO erinfo );
```

(2) 引数

trid

S15F5 信時に DSHEng4 から渡される DSHDR2 通信ドライバーのトランザクション管理のための ID です。

info

レシビネームスペースリネームの表示メッセージ情報が格納されているポインタです。

erinfo

送信する応答メッセージのための応答情報が格納されている構造体のポインタです。

(3) 戻り値

戻り値	意味
0	正常に送信できた。
(-1)	送信できなかった。

(4) 説明

レシビネームスペースリネームメッセージ S15F5 に対する応答メッセージを送信します。

本関数はユーザ作成ライブラリ DLL (dsh_ulib.dll) に含まれる関数ですが、ここでは DSHEng4 パッケージに標準的な関数として付属されているものです。(ユーザ独自による作成も可能です)

引数に指定されている erinfo 情報から S15F6 メッセージを組み立て、その後、S15F6 メッセージを送信します。

なお、S15F6 メッセージの組み立てに、DshMakeS15F5Response() 関数を使用できます。

3.13.4.3 DshResponseS15F14() S15F14 レシピ生成要求応答メッセージ

(1) 呼出書式

[C, C++]

```
API int APIX DshResponseS15F14(
    ID_TR trid,                // DSHDR2 のトランザクション ID
    TRCP_INFO *info,          // レシピ生成要求メッセージ格納ポインタ
    TRCP_ERR_INFO *erinfo     // S15F14 応答情報格納用構造体のポインタ
);
```

[.NET VB]

```
Function DshResponseS15F14 (
    ByVal trid As Int32,
    ByRef info As dsh_info.TRCP_INFO,
    ByRef erinfo As dsh_info.TRCP_ERR_INFO) As Int32
```

[.NET C#]

```
int DshResponseS15F14(
    uint trid,
    ref TRCP_INFO info,
    ref TRCP_ERR_INFO erinfo );
```

(2) 引数

trid

S15F13 信時に DSHEng4 から渡される DSHDR2 通信ドライバーのトランザクション管理のための ID です。

info

レシピ生成要求の表示メッセージ情報が格納されているポインタです。

erinfo

送信する応答メッセージのための応答情報が格納されている構造体のポインタです。

(3) 戻り値

戻り値	意味
0	正常に送信できた。
(-1)	送信できなかった。

(4) 説明

レシピ生成要求メッセージ S15F13 に対する応答メッセージを送信します。

本関数はユーザ作成ライブラリ DLL(dsh_ulib.dll)に含まれる関数ですが、ここでは DSHEng4 パッケージに標準的な関数として付属されているものです。(ユーザ独自による作成も可能です)

引数に指定されている erinfo 情報から S15F14 メッセージを組み立て、その後、S15F14 メッセージを送信します。

なお、S15F14 メッセージの組み立てに、DshMakeS15F13Response()関数を使用できます。