DSHGEM-CLASS　GEM 通信エンジン・クラスライブラリ

ソフトウェア・パッケージ

クラス・ライブラリ説明書

Vol-２　メッセージ通信クラス 編

２０１１年6月（改－６）

株式会社データマップ

DSHGEM-07-30363-00-ClassLib-Comm

［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.	2010 年 1 月	初版	
2.	2010/03/03	(1) 応答ﾒｯｾｰｼﾞ送信ｸﾗｽの訂正(削除)	下記ﾒｯｾｰｼﾞの応答送信ｸﾗｽが表に記述されていますが、これらの応答はｴﾝｼﾞﾝが自動的に行うので存在しません。訂正しました。 S1F4, S1F12, S16F20, S16F22 関連箇所 4. 装置状態変数(SV).. 11. ﾌﾟﾛｾｽ･ｼﾞｮﾌﾞ関連ﾒｯｾｰｼﾞ
		(2) S6F1 の送信	S6F1 ﾄﾚｰｽﾃﾞｰﾀの送信に関する記述ﾐｽの訂正 (S6F1 はｴﾝｼﾞﾝが自動応答) ==>(S6F1 はｴﾝｼﾞﾝが自動送信) 関連箇所 6.ﾒｯｾｰｼﾞ表
		(3) S16F27,S16F28 ｸﾗｽ説明の追加	12.5 DshS16F27Send ｸﾗｽ 12.6 DshS16F28Rsponse ｸﾗｽ
3.	2010 年 7 月	send_sxfy() の overload ﾒｿｯﾄﾞ send()を追加 S2F353,S2F37,S2F33	1 次ﾒｯｾｰｼﾞ送信ｸﾗｽDshSxSySend ｸﾗｽのﾒｿｯﾄﾞに send()ﾒｿｯﾄﾞを追加しました。 7.1、7.2、7.3 count=0 のｹｰｽの説明を追加した。
4	2011/02/07	ｸﾗｽに Dispose ﾒｿｯﾄﾞﾄｸｸﾗｽのﾄﾚｰｽﾓﾆﾀｰと表示機能を実装した。	ｸﾗｽに Dispose ﾒｿｯﾄﾞを追加した。 また、ﾃﾞﾊﾞｯｸﾞ機能強化のためﾕｰｻﾞがｸﾗｽの生成、消滅の回数を管理し、それらのﾀｲﾐﾝｸﾞをﾄﾚｰｽ表示できるようにした。 Vol-1 Part-2 21. DshDebug クラス参照
5	2011/06/07	1 次ﾒｯｾｰｼﾞ送信に block ﾓｰﾄﾞを追加した。 send_request() send_request_wait() の説明を追加した。	1 次ﾒｯｾｰｼﾞを送信し、2 次ﾒｯｾｰｼﾞを受信するまで block ﾓｰﾄﾞで実行する。send_wait()ﾒｿｯｿを 1 次ﾒｯｾｰｼﾞ送信ｸﾗｽに機能追加した。 ﾕｰｻﾞ固有 1 次ﾒｯｾｰｼﾞの送受信のための DshEquipment ｸﾗｽの send_request(), send_request_wait()ﾒｿｯﾄﾞの説明を追加した。

目次

１．はじめに

本説明書は、DSHGEMCLASS がサポートする GEM, GEM300 関連 SECS-Ⅱ通信メッセージの送受信のためのクラスについて機能、構文、メンバー（プロパティ、メソッド）とその使用方法について説明します。

本説明書は Vol-2 になります。

DSHGEMCLASS が管理する基本的な GEM,GEM300 関連情報に関するクラスの説明については Vol-1 の説明書を参照してください。

以下、本クラスライブラリがサポートするメッセージの一覧表を示します。

[SECS-Ⅱメッセージ一覧表]

本ライブラリがサポートするメッセージの一覧表を示します。

この表に出ていないメッセージの送受信については、17. send_request(), send_request_wait()を参照してください。

	メッセージ	クラス名	機能概略
1	S1F3,4	DshS1F3Send	S1F3 送信　Selected Equipment Status Request
		－	S1F4 応答
2	S1F11,12	DshS1F11Send	S1F11 送信　Status Variable Namelist Request
		－	S1F12 応答
3	S1F15,16	DshS1F15Send	S1F15 送信　Request OFF-LINE
		DshS1F16Response	S1F16 応答
4	S1F17,18	DshS1F17Send	S1F17 送信　Request ON-LINE
		DshS1F18Response	S1F18 応答
5	S2F13,14	DshS2F13Send	S2F13 送信　Equipment Constant Request
		－	(S2F14 はｴﾝｼﾞﾝが自動応答)
6	S2F15,16	DshS2F15Send	S2F15 送信　New Equipment Constant Send
		－	(S2F14 はｴﾝｼﾞﾝが自動応答)
7	S2F23,24	DshS2F23Send	S2F23 送信　Trace Initialize Send
		DshS2F24Response	S2F24 応答
8	S2F29,30	DshS2F29eSend	S2F29 送信　Equipmeny Constant Namelist Request
		－	(S2F30 はｴﾝｼﾞﾝが自動応答)
9	S2F31,32	DshS2F31Send	S2F31 送信　Date and Time Set Requst
		－	(S2F32 はｴﾝｼﾞﾝが自動応答)
10	S2F33,34	DshS2F33Send	S2F33 送信　Define Report
		－	(S2F34 はｴﾝｼﾞﾝが自動応答)
11	S2F35,36	DshS2F35Send	S2F35 送信　Link Event Report
		－	(S2F36 はｴﾝｼﾞﾝが自動応答)
12	S2F37,38	DshS2F37Send	S2F37 送信　Enable/Disabel Event Report
		－	(S2F38 はｴﾝｼﾞﾝが自動応答)
13	S2F41,42	DshS2F41Send	S2F41 送信　Host Command Send
		DshS2F42Response	S2F42 応答
14	S2F43,44	DshS2F43Send	S2F43 送信　Reset Spooling Stream and Function
		DshS2F44Response	S2F44 応答

15	S2F45,46	DshS2F45Send	S2F45 送信　Define Variable Limit Attributes
		DshS2F46Response	S2F46 応答
16	S2F47,48	DshS2F47Send	S2F47 送信　Variable Limit Attributes Request
		－	(S2F48 はｴﾝｼﾞﾝが自動応答)
17	S2F49,50	DshS2F49Send	S2F49 送信　Enhanced Remote Command
		DshS2F50Response	S2F50 応答
18	S3F17,18	DshS3F17Send	S3F17 送信　Carrier Action Request
		DshS3F18Response	S3F18 応答
19	S3F23,24	DshS3F23Send	S3F23 送信　Port Group Action Request
		DshS3F24Response	S3F24 応答
20	S3F25,26	DshS3F25Send	S3F25 送信　Port Action Request
		DshS3F24Response	S3F26 応答
21	S3F27,28	DshS3F27Send	S3F27 送信　Change Access
		DshS3F28Response	S3F28 応答
22	S5F1,2	DshS5F1Send	S5F1 送信　Alarm Report Send
		DshS5F2Response	S5F2 応答
23	S5F3,4	DshS5F3Send	S5F3 送信　Enable/Disabel Alarm Send
		－	(S5F4 はｴﾝｼﾞﾝが自動応答)
24	S5F5,6	DshS5F5Send	S5F5 送信　List Alarm Request
		－	(S5F6 はｴﾝｼﾞﾝが自動応答)
25	S6F1,S6F2	－	(S6F1 はｴﾝｼﾞﾝが自動応答) Trace Data Send
		DshS6F2Response	S6F2 応答
26	S6F11,S6F12	DshS6F11Send	S6F11 送信　Event Report Send
		DshS6F12Response	S6F12 応答
27	S6F15,S6F16	DshS6F15Send	S6F15 送信　Event Report Request
		－	(S6F16 はｴﾝｼﾞﾝが自動応答)
28	S6F19,S6F20	DshS6F19Send	S6F19 送信　Indivisual Report Data
		－	(S6F20 はｴﾝｼﾞﾝが自動応答)
29	S6F23,S6F24	DshS6F23Send	S6F23 送信　Request Spooled Data
		－	(S6F24 はｴﾝｼﾞﾝが自動応答)
30	S7F1,S7F2	DshS7F1Send	S7F1 送信　Process Program Load Inquire
		DshS7F2Response	S7F2 応答
31	S7F3,S7F4	DshS7F3Send	S7F3 送信　Process Program Send
		DshS7F4Response	S7F4 応答
32	S7F5,S7F6	DshS7F5Send	S7F5 送信　Process Program Data
		－	(S7F6 はｴﾝｼﾞﾝが自動応答)
33	S7F17,S7F18	DshS7F17Send	S7F17 送信　Delete Process Program Send
		－	(S7F18 はｴﾝｼﾞﾝが自動応答)
34	S7F19,S7F20	DshS7F19Send	S7F19 送信　Current EPPD Request
		－	(S7F20 はｴﾝｼﾞﾝが自動応答)
35	S10F1,S10F2	DshS10F1Send	S10F1 送信　Terminal Request
		DshS10F2Response	S10F2 応答
36	S10F3,S10F4	DshS10F3Send	S10F3 送信　Treminal Display, Single
		DshS10F4Response	S10F4 応答
37	S10F5,S10F6	DshS10F5Send	S10F5 送信　Terminal Display, Multi-Block
		DshS10F6Response	S10F6 応答
38	S14F9,S14F10	DshS14F9Send	S14F9 送信　Create Object Request
		DshS14F10Response	S14F10 応答

39	S14F11,S14F12	DshS14F11Send	S14F11 送信　Delete Object Request
		DshS14F12Response	S14F12 応答
40	S15F3,S15F4	DshS15F3Send	S15F3 送信　Recipe Name Space Action Request
		DshS15F4Response	S15F4 応答
41	S15F5,S15F6	DshS15F5Send	S15F5 送信　Recipe Name space Rename Request
		DshS15F6Response	S15F6 応答
42	S15F7,S15F8	DshS15F7Send	S15F7 送信　Recipe Space Request
		-	(S15F8 はｴﾝｼﾞﾝが自動応答)
43	S15F9,S15F10	DshS15F9Send	S15F9 送信　Recipe Status Request
		-	(S15F10 はｴﾝｼﾞﾝが自動応答)
44	S15F13,S15F14	DshS15F13Send	S15F13 送信　Recipe Create Request
		DshS15F14Response	S15F14 応答
45	S15F17,S15F18	DshS15F17Send	S15F17 送信　Recipe Retrieve Request
		DshS15F18Response	S15F18 応答
46	S16F5,S16F6	DshS16F5Send	S16F5 送信　Process Job Command Request
		DshS16F6Response	S16F6 応答
47	S16F11,S16F12	DshS16F11Send	S16F11 送信　PrJobCreateEnh
		DshS16F12Response	S16F12 応答
48	S16F15,S16F16	DshS16F15Send	S16F15 送信　PrJobMultiCreate
		DshS16F16Response	S16F16 応答
49	S16F17,S16F18	DshS16F17Send	S16F17 送信　PrJobDeque
		DshS16F18Response	S16F18 応答
50	S16F19,S16F20	DshS16F19Send	S16F19 送信　PrGetAllJobs
		DshS16F20Response	S16F20 応答
51	S16F21,S16F22	DshS16F21Send	S16F21 送信　PrGetSpace
		DshS16F22Response	S16F22 応答
52	S16F27,S16F28	DshS16F27Send	S16F27 送信　Control Job Command Request
		DshS16F28Response	S16F28 応答

２．SECS-II メッセージ送受信の仕方

２．１　１次メッセージの送信と２次メッセージ受信

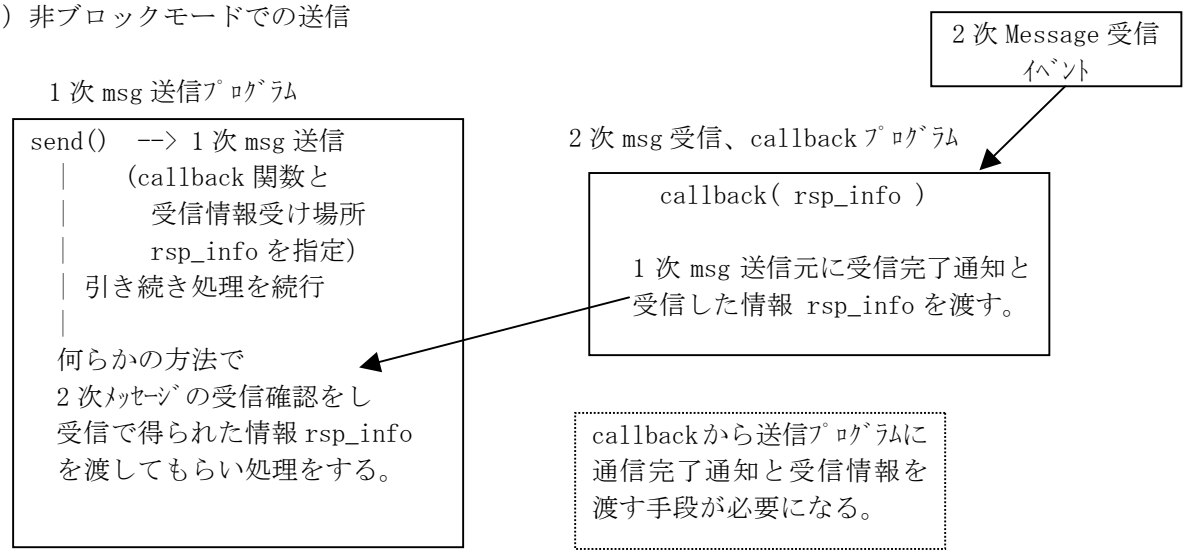
２０１１年６月に ブロックモードで送受信できるように機能を追加しました。

従来のクラスライブラリでは１次メッセージ送信後、プログラムはブロックされないで、２次メッセージの受信結果を send()メソッドの引数で指定したコールバック(callback)にイベント通知を受け、処理するような方法を取っていました。これは非ブロックモード(non-block)の送信になります。

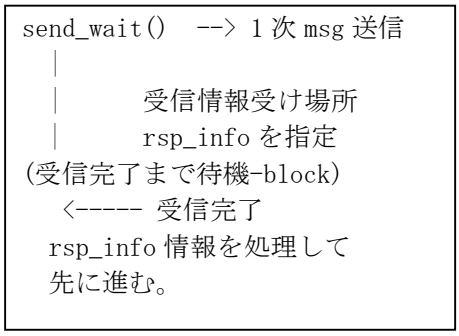
今回、新たに、１次メッセージを送信した後、プログラムを２次メッセージを受信するまでプログラムをブロックする機能、ブロックモード(block mode)による送受信機能を追加しました。メソッドとして send_wait()を使用します。
ブロックモードでは、callback に対する非同期に発生するイベントの処理をする必要がなくなり、プログラミングが非ブロックモードよりシンプルになります。

これによって、ユーザはプログラム設計、作成方針の中で、ブロック／非ブロック、どちらか組みやすい方法を選択することが可能になります。

（１）非ブロックモードでの送信



（２）ブロックモードでの送信



以下 S15F13 の送信を例に、送信処理の手順を説明します。

２．１．１　非ブロックモードでのプログラミング例

S1F13 の送信で、送信のためのクラスは、DshS2F13Send で、そのインスタンス名を send_class とします。
装置 ID は int eqid 変数に設定されているものとします。

（１）送信メッセージのために用意されているクラスのインスタンスを生成します。

```
DshS2F13Send send_class = new DshS2F13Send(eqid);
```

（２）メッセージに含むべき情報をインスタンスのプロパティのメンバーに設定します。
設定は、メンバーに直接またはクラスが提供するメソッドを使って行います。
例えば、装置定数 ECID = EC_Mdln の要求を行う場合は次のように設定します。

```
send_class.add_vid(eng_id.EC_Mdln);          // EC_Mdln は class eng_id で定義されている
```

（３）メッセージを送信するためのメソッドを実行します。

```
int ei = send_class.send_s2f13(ref rsp_info213, cback_s2f13, 213)
```

ここで、引数について説明します。

①rsp_info213
は受信する応答メッセージ S2F14 の内容を保存するためのクラス DshV_ValueList のインスタンスです。次のように static 指定で静的変数として準備しておきます。

```
private static DshV_ValueList rsp_info213 = new DshV_ValueList();
```

②cback_s2f13
エンジンから終了通知を受けるためのコールバック(イベント通知)関数を指定します。

```
private static DshCallback.callback_s2f13 cback_s2f13 = new
                                        DshCallback.callback_s2f13(callback_S2F13);
```

コールバック関数は DshCallback クラスで定義されています。
callback_S2F13 が実際のハンドラーです。これについては （４）で説明します。

③最後の 213
uint 型の引数で、ユーザが使用できるタグです。要求したプログラムが終了通知を受けたときに使用することができます。ここでは、S2F13 の 213 を指定しています。

送信要求が受け付けられたかどうは、このメソッドの返却値で判断します。
返却値=0 ならば受け付けられたことを意味し、(-1)ならば受け付けられなかったことを意味します。

正常に受け付けられたら、終了イベント通知を待ちます。終了イベントは、メッセージ送信メソッドの引数 callback 関数として与えられたコールバック関数の呼び出しで通知されます。

（４）送信メソッドの引数に与える callback 関数について説明します。
（３）－ ②で出てきました callback_S2F13 の例は次のようになります。

```
private static int callback_S2F13(int eqid, int end_status, ref DshV_ValueList rsp_info,
                                                                    uint upara)
{
      DshLog.log("  ! send_info send_S2F13 Callback() end_status = " +
                                                    end_status.ToString() + "¥n");
      DshLog.log("    upara = " + upara.ToString() + "¥n");
      if (end_status == 0)
      {
          disp_v_info.disp_DshV_ValueList("----- EC value list -----", ref rsp_info);
      }
      return 0;
}
```

callback_S2F13 関数には４つの引数が付いてきます。それぞれ次の意味になります。

①eqid	：装置 ID です。(3)の送信要求で指定したものです。
②end_status	：終了状態ｺｰﾄﾞです。 =0 で正常終了、 =(-1)で異常終了を示します。
③rsp_info	：S2F14 の情報が保存されているインスタンスで、(3)の rsp_info213 です。
④upara	：ユーザパラメータ（タグ）です。(3)で与えた 213 の値が戻されます。

（注）DshLog.log, DshV_ValueList は GemCsDemo デモプログラムに含まれる関数です。
ここでは説明を省略します。

この **callback 関数は、static** にしてください。そして、0 を返却してください。

以上、S2F13 を例に説明しましたが、送信要求に使用するクラス、送信関数の引数、コールバック関数の引数は、各メッセージごとに違います。詳しい内容については、３章以降に説明します。

２．１．２　ブロックモードによろプログラミング

送信のためのクラスは、DshS2F13Send で、そのインスタンス名を send_class とします。
装置 ID は int eqid 変数に設定されているものとします。

（１）送信メッセージのために用意されているクラスのインスタンスを生成します。

```
DshS2F13Send send_class = new DshS2F13Send(eqid);
```

（２）メッセージに含むべき情報をインスタンスのプロパティのメンバーに設定します。
設定は、メンバーに直接またはクラスが提供するメソッドを使って行います。
例えば、装置定数 ECID = EC_Mdln の要求を行う場合は次のように設定します。

```
send_class.add_vid(eng_id.EC_Mdln);          // EC_Mdln は class eng_id で定義されている
```

（３）メッセージを送信するためのメソッドを実行します。

```
int ei = send_class.send_wait(ref rsp_info213)
```

ここで、引数について説明します。
①rsp_info213
は受信する応答メッセージ S2F14 の内容を保存するためのクラス DshV_ValueList のインスタンスです。次のように static 指定で静的変数として準備しておきます。

```
private static DshV_ValueList rsp_info213 = new DshV_ValueList();
```

send_wait()によって、２次メッセージの受信、または通信エラーが発生し終了するまでプログラムはブロックされます。(先に進みません。)

通信が正常に終了すると ei= 0 が返却されます。そして、rsp_info213 に受信した S2F14 メッセージ内に含む情報が DshV_ValueList クラスのインスタンス rsp_info213 に渡されます。

この後、プログラムが rsp_info213 の処理を行い、先に進みます。

２．２　受信１次メッセージに対する２次メッセージの応答の仕方

ユーザプログラムでの SECS-Ⅱ１次メッセージの受信と処理の手順は以下のようになります。

（１）DshEngine クラスを使って DshGemClass エンジンを起動します。
そして、次に、通信相手の装置起動を DshEquipment クラスを使って行います。

（２）装置の起動を行った後、DshEquipment クラスの start_poll()メソッドの実行によってエンジンは相手装置から送信されてくる１次メッセージの受信ポーリングを開始します。

（３）ポーリングが開始後、ポーリングによって受信したとき、start_poll()メソッドで指定したコールバック関数(callback 関数)を呼び出します。受信したメッセージ情報はコールバック関数の引数として渡されます。

（４）コールバック関数では、引数に与えられた情報（装置 ID、メッセージ格納構造体のポインタ、トランザクション ID）に従ってメッセージを処理します。

（５）メッセージの処理は、まず、メッセージ ID(stream, function)の値によって、処理を行うことになります。

（６）メッセージ ID 毎の処理は次のような内容になります。
①SECS-Ⅱメッセージ情報は、DSHMSG 構造体の中に保存されて与えられます。これを mmsg とします。
②DshGemClass ライブラリには、mmsg に含まれる情報の保存が必要とされる全てのメッセージに対してクラスが提供されています。すなわち、メッセージ ID によってクラス名が決まります。
③ユーザは、まず、メッセージ情報保存用のクラスのインスタンスを生成します。
④次に、そのインスタンスの decode()メソッドに引数 mmsg を付けて呼び出します。
decode()メソッドは、mmsg に含まれる情報をインスタンスのプロパティメンバーに保存してくれます。decode()が成功した場合は、=0 が返却され、失敗すれば=(-1)が返却されます。
⑤ユーザはインスタンスのプロパティに保存されたメッセージの情報の処理です。
プロパティの内容をエンジンに登録設定する処理などの処理を行います。

（７）次に、２次メッセージの応答にの処理になります。
①DshGemClass ライブラリは、２次メッセージ応答のためのクラスもメッセージ毎に設けています。
②クラスは、例えば、DshS3F18Response クラスは、S3F18 メッセージ送信のためのクラスです。
③これら２次メッセージ応答用クラスには、全て、response()メソッドが定義されており、このメソッドを使って応答メッセージを送信します。
④response()メソッドによっては、応答メッセージを作成するために必要な情報を準備して引数として与えられるものもあります。

（８）最後に、コールバック関数は、mmsg メッセージ情報に使用されているメモリを DshLib クラスの DshFreeMessage()メソッドを使って開放します。

次ページに S2F41 の受信処理サンプルを示します。

プログラムサンプル　S2F41 受信処理

```
public static void s2f41(int eqid, uint trid, ref DSHMSG smsg)
{
    int hcack = 0;
    int err_count = 0;
    DshRCmd req_class = new DshRCmd();
    int ei = req_class.decode(ref smsg);                    // decode s2f41
    if (ei >= 0)
    {
        //  ここに処理
        hcack = proc_HostCommand( req_class );

        if (hcack != 0) err_count = 3;
    }
    else                                                    // decode error
    {
        hcack = 1;
    }
    DshRCmdRsp rsp_class = new DshRCmdRsp(hcack);
    for (int i = 0; i < err_count; i++)
    {
        rsp_class.add_ack(req_class.list[i].name, i + 1);
    }
    DshS2F42Response s2f42_class = new DshS2F42Response(eqid);
    s2f42_class.response(trid, ref rsp_class);              // send S2F42
}
```

３．装置定数(EC)関連メッセージの送受信

以下のメッセージがあります。

1	S2F13,14	DshS2F13Send	S2F13 送信　Equipment Constant Request
		－	(S2F14 はｴﾝｼﾞﾝが自動応答)
2	S2F15,16	DshS2F15Send	S2F15 送信　New Equipment Constant Send
		－	(S2F14 はｴﾝｼﾞﾝが自動応答)
3	S2F29,30	DshS2F29eSend	S2F29 送信　Equipmeny Constant Namelist Request
		－	(S2F30 はｴﾝｼﾞﾝが自動応答)

３．１　DshS2F13Send クラス

S2F13 メッセージを送信し、S2F14 応答メッセージを受信するためのクラスです。

３．１．１　コンストラクタ

オーバーロードの一覧を示します。

	名前	説明
1	public DshS2F13Send()	装置 ID=0 のｲﾝｽﾀﾝｽを生成します。
2	public DshS2F13Send(int eqid)	装置 ID を指定してｲﾝｽﾀﾝｽを生成します。

３．１．２　プロパティ

プロパティ一覧表に示します。

	名前	説明
1	public int eqid	装置 ID です。ﾃﾞﾌｫﾙﾄ値=0 です。
2	public int count	3 の vid_list 配列に保存されている ECID の数です。
3	public uint[] vid_list	ECID の配列ﾘｽﾄです。 S2F13 に設定する装置定数 ID 保存用です。

３．１．３　メソッド

本クラスのメソッドは次の通りです。

	名前	説明
1	public void clear()	vid_list を空にし、count = 0 にします。
2	public void add_vid()	vid_list に ECID を 1 個追加します。
3	public int send_s2f13() public int send()	S2F13 ﾒｯｾｰｼﾞを送信します。
4	public int send_wait()	S2F13 ﾒｯｾｰｼﾞを送信し、S2F14 を受信します。 ﾌﾟﾛｸﾞﾗﾑは応答受信までﾌﾞﾛｯｸされます。
5	public void Dispose()	ｸﾗｽ内部で使用された資源(ﾒﾓﾘ)ｼｽﾃﾑに返却します。 他のｸﾗｽをﾌﾟﾛﾊﾟﾃｨにしている場合、そのｸﾗｽの Dispose()も行います。

３．１．３．１　clear()

装置定数 ID 保存の vid_list を空にします。

【構文】
public void clear()

【引数】
なし

【戻り値】
なし。

【説明】
装置定数 ID 保存配列リスト vid_list を空にし、count = 0 にします。

３．１．３．２　add_vid()

装置定数 ID を保存する vid_list に ID を 1 個追加します。

【構文】
public void add_vid(uint vid)

【引数】
vid
　　追加する装置定数 ID です。

【戻り値】
なし。

【説明】
装置定数 ID 保存配列リスト vid_list に ID を 1 個追加します。
追加した後、count + 1 します。

３．１．３．３　send_s2f13(), send()

S2F13 メッセージの送信要求をします。

【構文】

```
public int send_s2f13(ref DshV_ValueList rsp_info,
                              DshCallback.callback_s2f13 callback, uint upara)
public int send (ref DshV_ValueList rsp_info,
                              DshCallback.callback_s2f13 callback, uint upara)
```

【引数】

rsp_info
　　S2F14 応答メッセージに含まれる装置定数情報を保存するためのクラスのインスタンスです。
　　コールバック関数の引数になります。

callback
　　S2F13 送信後、2 次メッセージ受信した時に呼び出される callback 関数です。

upara
　　ユーザパラメータ情報です。コールバックされる際に callback 関数の引数として渡されます。
　　要求とコールバック間のタグ情報として使用できます。

【戻り値】

返却値	意　味
0	要求が受け入れられた。
(-1)	要求が受け入れられなかった。

【説明】

当該インスタンスの vid_list に保存されている count 分の装置定数 ID を含む S2F13 メッセージを生成し、それを相手装置に送信するようエンジンに要求します。
要求がエンジンによって受け入れられたときは、0 を返却します。受け入れられなかった場合は(-1)を返却します。
送信後、受信した応答メッセージを rsp_info のインスタンスに設定し、callback 関数を呼び出します。
callback 関数からは 0 を返却してください。

【callback の構文】

```
public delegate int callback_s2f13(
    int eqid,                          // 装置 ID
    int end_status,                    // 終了状態ｺｰﾄﾞ
    ref DshV_ValueList vlist,          // S2F14 に含まれる装置定数情報    Vol-1 4.6 参照
    uint upara                         // ﾕｰｻﾞﾊﾟﾗﾒｰﾀ(送信要求ﾒｿｯﾄﾞで指定された upara)
)
```

end_status に返却される値は、=0 の場合は正常終了を意味します。

end_status	意　味
0	送受信が正常に完了した。
-14	T3 ﾀｲﾑｱｳﾄを検出し、応答ﾒｯｾｰｼﾞを受信できなかった。
(-1)	送信できなかった。

３．１．３．４　send_wait()

S2F13 メッセージを送信し、引き続き応答メッセージを受信も行います。

【構文】

```
public int send_wait(ref DshV_ValueList rsp_info)
public int send_wait()
```

【引数】

rsp_info

S2F14 応答メッセージに含まれる装置定数情報を保存するためのクラスのインスタンスです。

【戻り値】

返却値	意　味
0	正常に送受信した。
＜ 0	送受信ｴﾗｰが発生した。（送信ｴﾗｰ or T3 ﾀｲﾑｱｳﾄなど）

【説明】

当該インスタンスの vid_list に保存されている count 分の装置定数 ID を含む S2F13 メッセージを生成し、それを相手装置に送信するようエンジンに要求します。
本メソッドは送信の後、引き続き S2F14 応答メッセージを待機し、受信が終了したら、上の【戻り値】に示した値を返却します。
応答情報は、rsp_info に渡されます。

応答メッセージを受信するまでプログラムはブロック（待ち）状態になります。

送信エラー、受信エラーを検出した場合は、ei ＜ 0 の値を返却します。（T3 ﾀｲﾑｱｳﾄも含む）

T3 タイムアウトが発生した場合、DSHDR2 HSMS ドライバーに設定されている T3 プロトコルタイムアウト時間だけブロックされることになります。

３．１．３．５　Dispose()

本クラスが使用済になった際、クラス内で使用していて、開放すべきメモリがあれば、それを開放し、また Dispose すべきオブジェクトがあれば、それらを Dispose します。

【構文】

```
public void Dispose()
```

【引数】

なし。

【戻り値】

なし。

【説明】

生成された後、当該クラスが使用済みになった時点で、ユーザプログラムが明示的に使用していた資源を解放するためのメソッドです。

本メソッドが実行されるとシステムから本クラスに対する Finalizer は呼び出されません。

３．２　DshS2F15Send クラス

S2F15 メッセージを送信し、S2F16 応答メッセージを受信するためのクラスです。

３．２．１　コンストラクタ

オーバーロードの一覧を示します。

	名前	説明
1	public DshS2F15Send()	装置 ID=0 の空のｲﾝｽﾀﾝｽを生成します。
	public DshS2F15Send(int eqid)	装置 ID を指定してｲﾝｽﾀﾝｽを生成します。

３．２．２　プロパティ

プロパティ一覧表に示します。

	名前	説明
1	public int eqid	装置 ID です。ﾃﾞﾌｫﾙﾄ値=0 です。
2	public DshV_ValueList v_list	EC 変数 ID と値の配列ﾘｽﾄです。 送信情報になります。 DshV_ValueList ｸﾗｽは DshV_Value ｸﾗｽの配列ﾘｽﾄであり、 複数の変数 ID と変数値を保存することができます。 詳しくは、Vol-1 4.5, 4.6 を参照してください。

３．２．３　メソッド

本クラスのメソッドは次の通りです。

	名前	説明
1	public void clear()	v_list 配列の内容を空にします。
2	public void add_info()	装置定数 ID とその値を v_list に追加します。
3	public int send_s2f15() public int send()	S2F15 ﾒｯｾｰｼﾞを送信します。
4	public int send_wait()	S2F15 ﾒｯｾｰｼﾞを送信し、S2F16 を受信します。 ﾌﾟﾛｸﾞﾗﾑは応答受信までﾌﾞﾛｯｸされます。
5	public void Dispose()	ｸﾗｽ内部で使用された資源(ﾒﾓﾘ)ｼｽﾃﾑに返却します。 他のｸﾗｽをﾌﾟﾛﾊﾟﾃｨにしている場合、そのｸﾗｽの Dispose()も行います。 3.1.3.5 と同様です。そちらを参照ください。

３．２．３．１　clear()

v_list の配列リストを空にします。

【構文】

public void clear()

【引数】

なし

【戻り値】

なし。

【説明】

v_list を空にします。

３．２．３．２　add_v_info()

v_list に装置定数の ID と値を 1 個追加します。

【構文】

public void add_v_info(uint vid, int format, int size, IntPtr value)

【引数】

vid

追加する変数 ID です。

format

装置定数値のデータフォーマットです。(HSMS.ICODE_U1 など)

size

装置定数値の配列サイズです。

value

設定したい装置定数値が格納されているポインタです。

【戻り値】

なし。

【説明】

装置定数 ID 情報保存配列リスト v_list に ID を 1 個追加します。

３．２．３．３　send_s2f15(), send()

S2F15 メッセージの送信要求をします。

【構文】

```
public int send_s2f15(ref int eac, DshCallback.callback_s2f15 callback, uint upara)
public int send(ref int eac, DshCallback.callback_s2f15 callback, uint upara)
```

【引数】

eac

S2F16 応答メッセージに含まれる eac ACK 保存用です。コールバック関数の引数になります。

callback

S2F15 送信後、2 次メッセージを受信した時に呼び出される callback 関数です。

upara

ユーザパラメータ情報です。コールバックされる際に callback 関数の引数として渡されます。
要求とコールバック間のタグ情報として使用できます。

【戻り値】

返却値	意　味
0	要求が受け入れられた。
(-1)	要求が受け入れられなかった。

【説明】

当該インスタンスの vid_list に保存されている count 分の装置定数 ID を含む S2F15 メッセージを生成し、それを相手装置に送信するようエンジンに要求します。
要求がエンジンによって受け入れられたときは、0 を返却します。受け入れられなかった場合は(-1)を返却します。
送信後、受信した応答メッセージを rsp_info のインスタンスに設定し、callback 関数を呼び出します。
callback 関数からは 0 を返却してください。

【callback の構文】

```
public delegate int callback_s2f15(
    int eqid,                       // 装置 ID
    int end_status,                 // 終了状態ｺｰﾄﾞ
    ref int eac,                    // S2F16 に含まれる eac Ack です。
    uint upara                      // ﾕｰｻﾞﾊﾟﾗﾒｰﾀ(送信要求ﾒｿｯﾄﾞで指定された upara)
)
```

end_status に返却される値は、=0 の場合は正常終了を意味します。

end_status	意　味
0	送受信が正常に完了した。
-14	T3 ﾀｲﾑｱｳﾄを検出し、応答ﾒｯｾｰｼﾞを受信できなかった。
(-1)	送信できなかった。

３．３．４．４　send_wait()

S2F15 メッセージを送信し、引き続き応答メッセージを受信も行います。

【構文】

public int send_wait(ref int eac)

【引数】

eac

S2F16 応答メッセージの eac (ack)保存用です。

【戻り値】

返却値	意　味
0	正常に送受信した。
＜ 0	送受信ｴﾗｰが発生した。(送信ｴﾗｰ or T3 ﾀｲﾑｱｳﾄなど)

【説明】

当該インスタンスの vid_list に保存されている count 分の装置定数 ID を含む S2F15 メッセージを生成し、それを相手装置に送信するようエンジンに要求します。
本メソッドは送信の後、引き続き S2F16 応答メッセージを待機し、受信が終了したら、上の【戻り値】に示した値を返却します。
応答情報は、eac に渡されます。

応答メッセージを受信するまでプログラムはブロック（待ち）状態になります。

送信エラー、受信エラーを検出した場合は、ei ＜ 0 の値を返却します。(T3 ﾀｲﾑｱｳﾄも含む)

T3 タイムアウトが発生した場合、DSHDR2 HSMS ドライバーに設定されている T3 プロトコルタイムアウト時間だけブロックされることになります。

３．３　DshS2F29Send クラス

S2F29 メッセージを送信し、S2F30 応答メッセージを受信するためのクラスです。

３．３．１　コンストラクタ

オーバーロードの一覧を示します。

	名前	説明
1	public DshS2F29Send()	装置 ID=0 のｲﾝｽﾀﾝｽを生成します。
2	public DshS2F29Send(int eqid)	装置 ID を指定してｲﾝｽﾀﾝｽを生成します。

３．３．２　プロパティ

プロパティ一覧表に示します。

	名前	説明
1	public int eqid	装置 ID です。ﾃﾞﾌｫﾙﾄ値=0 です。
2	public int count	3 の vid_list 配列に保存されている ECID の数です。
3	public uint[] vid_list	ECID の配列ﾘｽﾄです。 S2F29 に設定する装置定数 ID 保存用です。

３．３．３　メソッド

本クラスのメソッドは次の通りです。

	名前	説明
1	public void clear()	vid_list を空にし、count = 0 にします。
2	public void add_vid()	vid_list に ECID を 1 個追加します。
3	public int send_s2f29() public int send()	S2F29 ﾒｯｾｰｼﾞを送信します。
4	public int send_wait()	S2F29 ﾒｯｾｰｼﾞを送信し、S2F30 を受信します。 ﾌﾟﾛｸﾞﾗﾑは応答受信までﾌﾞﾛｯｸされます。
5	public void Dispose()	ｸﾗｽ内部で使用された資源(ﾒﾓﾘ)ｼｽﾃﾑに返却します。 他のｸﾗｽをﾌﾟﾛﾊﾟﾃｨにしている場合、そのｸﾗｽの Dispose()も行います。 3.1.3.5 と同様です。そちらを参照ください。

３．３．３．１　clear()

装置定数 ID 保存の vid_list を空にします。

【構文】
public void clear()

【引数】
なし

【戻り値】
なし。

【説明】
装置定数 ID 保存配列リスト vid_list を空にし、count = 0 にします。

３．３．３．２　add_vid()

装置定数 ID を保存する vid_list に ID を 1 個追加します。

【構文】
public void add_vid(uint vid)

【引数】
vid
　　追加する装置定数 ID です。

【戻り値】
なし。

【説明】
装置定数 ID 保存配列リスト vid_list に ID を 1 個追加します。
追加した後、count + 1 します。

３．３．３．３　send_s2f29(), send()

S2F29 メッセージの送信要求をします。

【構文】

```
public int send_s2f29(ref DshEC_NameList rsp_list,
                              DshCallback.callback_s2f29 callback, uint upara)
public int send(ref DshEC_NameList rsp_list,
                              DshCallback.callback_s2f29 callback, uint upara)
```

【引数】

rsp_list
　　S2F30 応答メッセージに含まれる名前情報を保存するためのクラスのインスタンスです。
　　コールバック関数の引数になります。

callback
　　S2F29 送信後、2 次メッセージ受信した時に呼び出される callback 関数です。

upara
　　ユーザパラメータ情報です。コールバックされる際に callback 関数の引数として渡されます。
　　要求とコールバック間のタグ情報として使用できます。

【戻り値】

返却値	意　味
0	要求が受け入れられた。
(-1)	要求が受け入れられなかった。

【説明】

当該インスタンスの vid_list に保存されている count 分の装置定数 ID を含む S2F29 メッセージを生成し、それを相手装置に送信するようエンジンに要求します。
要求がエンジンによって受け入れられたときは、0 を返却します。受け入れられなかった場合は(-1)を返却します。
送信後、受信した応答メッセージを rsp_list のインスタンスに設定し、callback 関数を呼び出します。
callback 関数からは 0 を返却してください。

【callback の構文】

```
public delegate int callback_s2f29(
    int eqid,                          // 装置 ID
    int end_status,                    // 終了状態ｺｰﾄﾞ
    ref DshEC_NameList name_list,      // S2F30 に含まれる装置定数名情報　 Vol-4.8 参照
    uint upara                         // ﾕｰｻﾞﾊﾟﾗﾒｰﾀ(送信要求ﾒｿｯﾄﾞで指定された upara)
)
```

end_status に返却される値は、=0 の場合は正常終了を意味します。

end_status	意　味
0	送受信が正常に完了した。
-14	T3 ﾀｲﾑｱｳﾄを検出し、応答ﾒｯｾｰｼﾞを受信できなかった。
(-1)	送信できなかった。

３．３．３．４　send_wait()

S2F29 メッセージを送信し、引き続き応答メッセージを受信も行います。

【構文】

public int send_wait(ref DshEC_NameList rsp_list)

【引数】

rsp_list

S2F30 応答メッセージに含まれる装置定数情報を保存するためのクラスのインスタンスです。

【戻り値】

返却値	意　味
0	正常に送受信した。
＜ 0	送受信ｴﾗｰが発生した。（送信ｴﾗｰ or T3 ﾀｲﾑｱｳﾄなど）

【説明】

当該インスタンスの vid_list に保存されている count 分の装置定数 ID を含む S2F29 メッセージを生成し、それを相手装置に送信するようエンジンに要求します。
本メソッドは送信の後、引き続き S2F30 応答メッセージを待機し、受信が終了したら、上の【戻り値】に示した値を返却します。
応答情報は、rsp_list に渡されます。

応答メッセージを受信するまでプログラムはブロック（待ち）状態になります。

送信エラー、受信エラーを検出した場合は、ei ＜ 0 の値を返却します。(T3 ﾀｲﾑｱｳﾄも含む)

T3 タイムアウトが発生した場合、DSHDR2 HSMS ドライバーに設定されている T3 プロトコルタイムアウト時間だけブロックされることになります。

４．装置状態変数(SV)関連メッセージの送受信

以下のメッセージがあります。

1	S1F3,4	DshS1F3Send	S1F3 送信　　Selected Equipment Status Request
		－	S1F4 応答
2	S1F11,12	DshS1F11Send	S1F11 送信　　Status Variable Namelist Request
		－	S1F12 応答

４．１　DshS1F3Send クラス

S1F3 メッセージを送信し、S1F4 応答メッセージを受信するためのクラスです。

４．１．１　コンストラクタ

オーバーロードの一覧を示します。

	名前	説明
1	public DshS1F3Send()	装置 ID=0 のｲﾝｽﾀﾝｽを生成します。
2	public DshS1F3Send(int eqid)	装置 ID を指定してｲﾝｽﾀﾝｽを生成します。

４．１．２　プロパティ

プロパティ一覧表に示します。

	名前	説明
1	public int eqid	装置 ID です。ﾃﾞﾌｫﾙﾄ値=0 です。
2	public int count	3 の vid_list 配列に保存されている SVID の数です。
3	public uint[] vid_list	SVID の配列ﾘｽﾄです。 S1F3 に設定する装置状態変数 ID 保存用です。

４．１．３　メソッド

本クラスのメソッドは次の通りです。

	名前	説明
1	public void clear()	vid_list を空にし、count = 0 にします。
2	public void add_vid()	vid_list に SVID を 1 個追加します。
3	public int send_s1f3() public int send()	S1F3 ﾒｯｾｰｼﾞを送信します。
4	public int send_wait()	S1F3 ﾒｯｾｰｼﾞを送信し、S1F4 を受信します。 ﾌﾟﾛｸﾞﾗﾑは応答受信までﾌﾞﾛｯｸされます。
5	public void Dispose()	ｸﾗｽ内部で使用された資源(ﾒﾓﾘ)ｼｽﾃﾑに返却します。 他のｸﾗｽをﾌﾟﾛﾊﾟﾃｨにしている場合、そのｸﾗｽの Dispose()も行います。 3.1.3.5 と同様です。そちらを参照ください。

４．１．３．１　clear()

装置状態変数 ID 保存の vid_list を空にします。

【構文】
public void clear()

【引数】
なし

【戻り値】
なし。

【説明】
装置状態変数 ID 保存配列リスト vid_list を空にし、count = 0 にします。

４．１．３．２　add_vid()

装置状態変数 ID を保存する vid_list に ID を 1 個追加します。

【構文】
public void add_vid(uint vid)

【引数】
vid
　　追加する装置状態変数 ID です。

【戻り値】
なし。

【説明】
装置状態変数 ID 保存配列リスト vid_list に ID を 1 個追加します。
追加した後、count + 1 します。

４．１．３．３　send_s1f3(), send()

S1F3 メッセージの送信要求をします。

【構文】

```
public int send_s1f3(ref DshV_ValueList rsp_info,
                              DshCallback.callback_s1f3 callback, uint upara)
public int send(ref DshV_ValueList rsp_info,
                              DshCallback.callback_s1f3 callback, uint upara)
```

【引数】

rsp_info

S1F4 応答メッセージに含まれる装置状態変数情報を保存するためのクラスのインスタンスです。コールバック関数の引数になります。

callback

S1F3 送信後、2 次メッセージ受信した時に呼び出される callback 関数です。

upara

ユーザパラメータ情報です。コールバックされる際に callback 関数の引数として渡されます。要求とコールバック間のタグ情報として使用できます。

【戻り値】

返却値	意　味
0	要求が受け入れられた。
(-1)	要求が受け入れられなかった。

【説明】

当該インスタンスの vid_list に保存されている count 分の装置状態変数 ID を含む S1F3 メッセージを生成し、それを相手装置に送信するようエンジンに要求します。
要求がエンジンによって受け入れられたときは、0 を返却します。受け入れられなかった場合は(-1)を返却します。
送信後、受信した応答メッセージを rsp_info のインスタンスに設定し、callback 関数を呼び出します。
callback 関数からは 0 を返却してください。

【callback の構文】

```
public delegate int callback_s1f3(
    int eqid,                       // 装置 ID
    int end_status,                 // 終了状態ｺｰﾄﾞ
    ref DshV_ValueList vlist,       // S1F4 に含まれる装置状態変数情報    Vol-1 4.6 参照
    uint upara                      // ﾕｰｻﾞﾊﾟﾗﾒｰﾀ(送信要求ﾒｿｯﾄﾞで指定された upara)
)
```

end_status に返却される値は、=0 の場合は正常終了を意味します。

end_status	意　味
0	送受信が正常に完了した。
-14	T3 ﾀｲﾑｱｳﾄを検出し、応答ﾒｯｾｰｼﾞを受信できなかった。
(-1)	送信できなかった。

４．１．３．４　send_wait()

S1F3 メッセージの送信要求をし,引き続き応答メッセージも受信します。

【構文】

public int send_wait(ref DshV_ValueList rsp_info)

【引数】

rsp_info

S1F4 応答メッセージに含まれる装置状態変数情報を保存するためのクラスのインスタンスです。

【戻り値】

返却値	意　味
0	正常に送受信した。
＜ 0	送受信ｴﾗｰが発生した。（送信ｴﾗｰ or T3 ﾀｲﾑｱｳﾄなど）

【説明】

当該インスタンスの vid_list に保存されている count 分の装置状態変数 ID を含む S1F3 メッセージを生成し、それを相手装置に送信するようエンジンに要求します。
本メソッドは送信の後、引き続き S2F16 応答メッセージを待機し、受信が終了したら、上の【戻り値】に示した値を返却します。
応答情報は、DshV_ValueList クラスのｲﾝｽﾀﾝｽ rsp_info に渡されます。

応答メッセージを受信するまでプログラムはブロック（待ち）状態になります。

送信エラー、受信エラーを検出した場合は、ei ＜ 0 の値を返却します。(T3 ﾀｲﾑｱｳﾄも含む)

T3 タイムアウトが発生した場合、DSHDR2 HSMS ドライバーに設定されている T3 プロトコルタイムアウト時間だけブロックされることになります。

４．２　DshS1F11Send クラス

S1F11 メッセージを送信し、S1F12 応答メッセージを受信するためのクラスです。

４．２．１　コンストラクタ

オーバーロードの一覧を示します。

	名前	説明
1	public DshS1F11Send()	装置 ID=0 のｲﾝｽﾀﾝｽを生成します。
2	public DshS1F11Send(int eqid)	装置 ID を指定してｲﾝｽﾀﾝｽを生成します。

４．２．２　プロパティ

プロパティ一覧表に示します。

	名前	説明
1	public int eqid	装置 ID です。ﾃﾞﾌｫﾙﾄ値=0 です。
2	public int count	3 の vid_list 配列に保存されている SVID の数です。
3	public uint[] vid_list	SVID の配列ﾘｽﾄです。 S1F11 に設定する装置状態変数 ID 保存用です。

４．２．３　メソッド

本クラスのメソッドは次の通りです。

	名前	説明
1	public void clear()	vid_list を空にし、count = 0 にします。
2	public void add_vid()	vid_list に SVID を 1 個追加します。
3	public int send_s1f11() public int send()	S1F11 ﾒｯｾｰｼﾞを送信します。
4	public int send_wait()	S1F11 ﾒｯｾｰｼﾞを送信し、S1F12 を受信します。 ﾌﾟﾛｸﾞﾗﾑは応答受信までﾌﾞﾛｯｸされます。
5	public void Dispose()	ｸﾗｽ内部で使用された資源(ﾒﾓﾘ)ｼｽﾃﾑに返却します。 他のｸﾗｽをﾌﾟﾛﾊﾟﾃｨにしている場合、そのｸﾗｽの Dispose()も行います。 3.1.3.5 と同様です。そちらを参照ください。

４．２．３．１　clear()

装置状態変数 ID 保存の vid_list を空にします。

【構文】
public void clear()

【引数】
なし

【戻り値】
なし。

【説明】
装置状態変数 ID 保存配列リスト vid_list を空にし、count = 0 にします。

４．２．３．２　add_vid()

装置状態変数 ID を保存する vid_list に ID を 1 個追加します。

【構文】
public void add_vid(uint vid)

【引数】
vid
　　追加する装置状態変数 ID です。

【戻り値】
なし。

【説明】
装置状態変数 ID 保存配列リスト vid_list に ID を 1 個追加します。
追加した後、count + 1 します。

４．２．３．３　send_s1f11(), send()

S1F11 メッセージの送信要求をします。

【構文】

```
public int send_s1f11(ref DshSV_NameList rsp_list,
                              DshCallback.callback_s1f11 callback, uint upara)
public int send(ref DshSV_NameList rsp_list,
                              DshCallback.callback_s1f11 callback, uint upara)
```

【引数】

rsp_list

S1F12 応答メッセージに含まれる名前情報を保存するためのクラスのインスタンスです。
コールバック関数の引数になります。

callback

S1F11 送信後、2 次メッセージ受信した時に呼び出される callback 関数です。

upara

ユーザパラメータ情報です。コールバックされる際に callback 関数の引数として渡されます。
要求とコールバック間のタグ情報として使用できます。

【戻り値】

返却値	意　味
0	要求が受け入れられた。
(-1)	要求が受け入れられなかった。

【説明】

当該インスタンスの vid_list に保存されている count 分の装置状態変数 ID を含む S1F11 メッセージを生成し、それを相手装置に送信するようエンジンに要求します。
要求がエンジンによって受け入れられたときは、0 を返却します。受け入れられなかった場合は(-1)を返却します。
送信後、受信した応答メッセージを rsp_list のインスタンスに設定し、callback 関数を呼び出します。
callback 関数からは 0 を返却してください。

【callback の構文】

```
public delegate int callback_s1f11(
    int eqid,                          // 装置 ID
    int end_status,                    // 終了状態ｺｰﾄﾞ
    ref DshSV_NameList name_list,      // S1F12 に含まれる装置状態変数名情報    Vol-4.9 参照
    uint upara                         // ﾕｰｻﾞﾊﾟﾗﾒｰﾀ(送信要求ﾒｿｯﾄﾞで指定された upara)
)
```

end_status に返却される値は、=0 の場合は正常終了を意味します。

end_status	意　味
0	送受信が正常に完了した。
-14	T3 ﾀｲﾑｱｳﾄを検出し、応答ﾒｯｾｰｼﾞを受信できなかった。
(-1)	送信できなかった。

４．２．３．４　send_wait()

S1F11 メッセージの送信要求をし,引き続き応答メッセージも受信します。

【構文】

public int send_wait(ref DshSV_NameList rsp_list)

【引数】

rsp_list

S1F12 応答メッセージに含まれる名前情報を保存するためのクラスのインスタンスです

【戻り値】

返却値	意　味
0	正常に送受信した。
＜ 0	送受信ｴﾗｰが発生した。（送信ｴﾗｰ or T3 ﾀｲﾑｱｳﾄなど）

【説明】

当該インスタンスの vid_list に保存されている count 分の装置状態変数 ID を含む S1F11 メッセージを生成し、それを相手装置に送信するようエンジンに要求します
本メソッドは送信の後、引き続き S1F12 応答メッセージを待機し、受信が終了したら、上の【戻り値】に示した値を返却します。
応答情報は、DshSV_NameList クラスのｲﾝｽﾀﾝｽ rsp_list に渡されます。

応答メッセージを受信するまでプログラムはブロック（待ち）状態になります。

送信エラー、受信エラーを検出した場合は、ei ＜ 0 の値を返却します。(T3 ﾀｲﾑｱｳﾄも含む)

T3 タイムアウトが発生した場合、DSHDR2 HSMS ドライバーに設定されている T3 プロトコルタイムアウト時間だけブロックされることになります。

５．変数リミット(LIMIT)関連メッセージの送受信

以下のメッセージがあります。

1	S2F45,46	DshS2F45Send	S2F45 送信　Define Variable Limit Attributes
		DshS2F46Response	S2F46 応答
2	S2F47,48	DshS2F47Send	S2F47 送信　Variable Limit Attributes Request
		－	(S2F48 はｴﾝｼﾞﾝが自動応答)

５．１　DshS2F45Send クラス

S2F45 メッセージを送信し、S2F46 応答メッセージを受信するためのクラスです。

５．１．１　コンストラクタ

オーバーロードの一覧を示します。

	名前	説明
1	public DshS2F45Send()	装置 ID=0 のｲﾝｽﾀﾝｽを生成します。
2	public DshS2F45Send(int eqid)	装置 ID を指定してｲﾝｽﾀﾝｽを生成します。

５．１．２　プロパティ

プロパティ一覧表に示します。

	名前	説明
1	public int eqid	装置 ID です。ﾃﾞﾌｫﾙﾄ値=0 です。
2	public DshLimitList limit_list	複数のﾘﾐｯﾄ情報を保存するｸﾗｽです。 DshLimitList は DshLimit ｸﾗｽの配列ﾘｽﾄです。

５．１．３　メソッド

本クラスのメソッドは次の通りです。

	名前	説明
1	public void set_list()	準備した DshLimitList のｲﾝｽﾀﾝｽの内容を limit_list に設定します。 Vol-1 の 5.1, 5.2 参照
2	public int send_s2f45() public int send()	S2F45 ﾒｯｾｰｼﾞを送信します。
3	public int send_wait()	S2F45 ﾒｯｾｰｼﾞを送信し、S2F46 を受信します。 ﾌﾟﾛｸﾞﾗﾑは応答受信までﾌﾞﾛｯｸされます。
4	public void Dispose()	ｸﾗｽ内部で使用された資源(ﾒﾓﾘ)ｼｽﾃﾑに返却します。 他のｸﾗｽをﾌﾟﾛﾊﾟﾃｨにしている場合、そのｸﾗｽの Dispose()も行います。 3.1.3.5 と同様です。そちらを参照ください。

５．１．３．１　set_list()

ユーザが準備した DshLimitList のクラスのインスタンスの内容を当該インスタンスの limit_list に設定します。

【構文】

public void set_list(ref DshLimitList info)

【引数】

info

設定したい情報が保存されている DshLimitList のインスタンスです。

【戻り値】

なし。

【説明】

info の引数のリミット情報リストの内容を当該インスタンスの limit_list にコピーします。

５．１．３．２　send_s2f45(), send()

S2F45 メッセージの送信要求をします。

【構文】

```
public int send_s2f45(ref DshLimitRspList rsp_list,
                              DshCallback.callback_s2f45 callback, uint upara)
public int send(ref DshLimitRspList rsp_list,
                              DshCallback.callback_s2f45 callback, uint upara)
```

【引数】

rsp_list

S2F46 応答メッセージに含まれる変数リミット設定応答情報を保存するためのクラスのインスタンスです。コールバック関数の引数になります。

callback

S2F45 送信後、2 次メッセージ受信した時に呼び出される callback 関数です。

upara

ユーザパラメータ情報です。コールバックされる際に callback 関数の引数として渡されます。要求とコールバック間のタグ情報として使用できます。

【戻り値】

返却値	意　味
0	要求が受け入れられた。
(-1)	要求が受け入れられなかった。

【説明】

当該インスタンスの limit_list に保存されている count 分の vid の変数リミットから S2F45 メッセージを生成し、それを相手装置に送信するようエンジンに要求します。
要求がエンジンによって受け入れられたときは、0 を返却します。受け入れられなかった場合は(-1)を返却します。
送信後、受信した応答メッセージを rsp_list のインスタンスに設定し、callback 関数を呼び出します。
callback 関数からは 0 を返却してください。

【callback の構文】

```
public delegate int callback_s2f45(
    int eqid,                          // 装置 ID
    int end_status,                    // 終了状態ｺｰﾄﾞ
    ref DshLimitRspList rsp_info,      // S2F46 に含まれる変数ﾘﾐｯﾄ設定応答情報 Vol-1 5.4 参照
    uint upara                         // ﾕｰｻﾞﾊﾟﾗﾒｰﾀ(送信要求ﾒｿｯﾄﾞで指定された upara)
)
```

end_status に返却される値は、=0 の場合は正常終了を意味します。

end_status	意　味
0	送受信が正常に完了した。
-14	T3 ﾀｲﾑｱｳﾄを検出し、応答ﾒｯｾｰｼﾞを受信できなかった。
(-1)	送信できなかった。

５．１．３．３　send_wait()

S2F45 メッセージの送信要求をし,引き続き応答メッセージも受信します。

【構文】

public int send_wait(ref DshLimitRspList rsp_list)

【引数】

rsp_list

S2F46 応答メッセージに含まれる変数リミット設定応答情報を保存するためのクラスのインスタンスです。

【戻り値】

返却値	意　味
0	正常に送受信した。
＜ 0	送受信ｴﾗｰが発生した。(送信ｴﾗｰ or T3 ﾀｲﾑｱｳﾄなど)

【説明】

当該インスタンスの limit_list に保存されている count 分の vid の変数リミットから S2F45 メッセージを生成し、それを相手装置に送信するようエンジンに要求します。
本メソッドは送信の後、引き続き S2F46 応答メッセージを待機し、受信が終了したら、上の【戻り値】に示した値を返却します。
応答情報は、DshLimitRspList クラスのｲﾝｽﾀﾝｽ rsp_list に渡されます。

応答メッセージを受信するまでプログラムはブロック（待ち）状態になります。

送信エラー、受信エラーを検出した場合は、ei ＜ 0 の値を返却します。(T3 ﾀｲﾑｱｳﾄも含む)

T3 タイムアウトが発生した場合、DSHDR2 HSMS ドライバーに設定されている T3 プロトコルタイムアウト時間だけブロックされることになります。

５．２　S2F46Response クラス

S2F45 メッセージを送信し、S2F46 応答メッセージを受信するためのクラスです。

５．２．１　コンストラクタ

オーバーロードの一覧を示します。

	名前	説明
1	public S2F46Response()	装置 ID=0 のｲﾝｽﾀﾝｽを生成します。
2	public S2F46Response(int eqid)	装置 ID を指定してｲﾝｽﾀﾝｽを生成します。

５．２．２　プロパティ

プロパティ一覧表に示します。

	名前	説明
1	public int eqid	装置 ID です。ﾃﾞﾌｫﾙﾄ値=0 です。

５．２．３　メソッド

本クラスのメソッドは次の通りです。

	名前	説明
1	public int response()	S2F46 応答ﾒｯｾｰｼﾞを送信します。
2	public void Dispose()	ｸﾗｽ内部で使用された資源(ﾒﾓﾘ)ｼｽﾃﾑに返却します。 他のｸﾗｽをﾌﾟﾛﾊﾟﾃｨにしている場合、そのｸﾗｽの Dispose()も行います。 3.1.3.5 と同様です。そちらを参照ください。

５．２．３．１　response()

S2F46 メッセージの応答送信要求をします。

【構文】

public int response(uint trid, ref DshLimitRspList rsp_class)

【引数】

trid

エンジンからポーリング時に与えられたトランザクション ID です。
処理した１次メッセージとの対応を取ります。

rsp_class

S2F46 メッセージのための応答情報が保存されている DshLimitRspList クラスのインスタンスです。Vol 5.4 参照

【戻り値】

返却値	意　味
0	要求が受け入れられた。
(-1)	要求が受け入れられなかった。(ﾄﾗﾝｻﾞｸｼｮﾝ ID が正しくなかった。)

【説明】

受信した S2F45 に対する S2F46 メッセージを送信します。

応答メッセージに含める情報は rsp_class に指定した DshLimitRspList クラスのインスタンスです。
trid は、この応答メッセージに対する S2F45 受信時にエンジンから与えられたトランザクション ID です。
trid がエンジンの管理の中に見つからなかった場合には、(-1)を返却します。
要求が受け入れられたら 0 を返却します。この後、応答メッセージ送信結果は特に通知されません。

５．３　DshS2F47Send クラス

S2F47 メッセージを送信し、S2F48 応答メッセージを受信するためのクラスです。

５．３．１　コンストラクタ

オーバーロードの一覧を示します。

	名前	説明
1	public DshS2F47Send()	装置 ID=0 のｲﾝｽﾀﾝｽを生成します。
2	public DshS2F47Send(int eqid)	装置 ID を指定してｲﾝｽﾀﾝｽを生成します。

５．３．２　プロパティ

プロパティ一覧表に示します。

	名前	説明
1	public int eqid	装置 ID です。ﾃﾞﾌｫﾙﾄ値=0 です。
2	public int count	3 の vid_list 配列に保存されている変数 ID の数です。
3	public uint[] vid_list	変数 ID の配列ﾘｽﾄです。 S2F47 に設定する変数 ID 保存用です。

５．３．３　メソッド

本クラスのメソッドは次の通りです。

	名前	説明
1	public void clear()	vid_list を空にし、count = 0 にします。
2	public void add_vid()	vid_list に変数 ID を 1 個追加します。
3	public int send_s2f47() public int send()	S2F47 ﾒｯｾｰｼﾞを送信します。
4	public int send_wait()	S2F47 ﾒｯｾｰｼﾞを送信し、S2F48 を受信します。 ﾌﾟﾛｸﾞﾗﾑは応答受信までﾌﾞﾛｯｸされます。
5	public void Dispose()	ｸﾗｽ内部で使用された資源(ﾒﾓﾘ)ｼｽﾃﾑに返却します。 他のｸﾗｽをﾌﾟﾛﾊﾟﾃｨにしている場合、そのｸﾗｽの Dispose()も行います。 3.1.3.5 と同様です。そちらを参照ください。

５．３．３．１　clear()

変数 ID 保存の vid_list を空にします。

【構文】
public void clear()

【引数】
なし

【戻り値】
なし。

【説明】
変数 ID 保存配列リスト vid_list を空にし、count = 0 にします。

５．３．３．２　add_vid()

変数 ID を保存する vid_list に ID を 1 個追加します。

【構文】
public void add_vid(uint vid)

【引数】
vid
　　追加する変数 ID です。

【戻り値】
なし。

【説明】
変数 ID 保存配列リスト vid_list に ID を 1 個追加します。
追加した後、count + 1 します。

５．３．３．３　send_s2f47(), send()

S2F47 メッセージの送信要求をします。

【構文】

```
public int send_s2f47(DshS2F48LimitRspList rsp_info,
                              DshCallback.callback_s2f47 callback, uint upara)
public int send(DshS2F48LimitRspList rsp_info,
                              DshCallback.callback_s2f47 callback, uint upara)
```

【引数】

rsp_info
　　S2F48 応答メッセージに含まれる変数リミット情報を保存するためのクラスのインスタンスです。
　　コールバック関数の引数になります。

callback
　　S2F47 送信後、2 次メッセージ受信した時に呼び出される callback 関数です。

upara
　　ユーザパラメータ情報です。コールバックされる際に callback 関数の引数として渡されます。
　　要求とコールバック間のタグ情報として使用できます。

【戻り値】

返却値	意　味
0	要求が受け入れられた。
(-1)	要求が受け入れられなかった。

【説明】

当該インスタンスの vid_list に保存されている count 分の変数 ID を含む S2F47 メッセージを生成し、それを相手装置に送信するようエンジンに要求します。
要求がエンジンによって受け入れられたときは、0 を返却します。受け入れられなかった場合は(-1)を返却します。
送信後、受信した応答メッセージを rsp_info のインスタンスに設定し、callback 関数を呼び出します。
callback 関数からは 0 を返却してください。

【callback の構文】

```
public delegate int callback_s2f47(
    int eqid,                          // 装置 ID
    int end_status,                    // 終了状態ｺｰﾄﾞ
    ref DshS2F48LimitRspList rsp_info, // S2F48 に含まれる変数リミット名情報　Vol-5.5 参照
    uint upara                         // ﾕｰｻﾞﾊﾟﾗﾒｰﾀ(送信要求ﾒｿｯﾄﾞで指定された upara)
)
```

end_status に返却される値は、=0 の場合は正常終了を意味します。

end_status	意　味
0	送受信が正常に完了した。
-14	T3 ﾀｲﾑｱｳﾄを検出し、応答ﾒｯｾｰｼﾞを受信できなかった。
(-1)	送信できなかった。

５．３．３．４　send_wait()

S2F47 メッセージの送信要求をし,引き続き応答メッセージも受信します。

【構文】

public int send_wait(DshS2F48LimitRspList rsp_info)

【引数】

rsp_info

S2F48 応答メッセージに含まれる変数リミット情報を保存するためのクラスのインスタンスです。

【戻り値】

返却値	意　味
0	正常に送受信した。
＜ 0	送受信ｴﾗｰが発生した。（送信ｴﾗｰ or T3 ﾀｲﾑｱｳﾄなど）

【説明】

当該インスタンスの vid_list に保存されている count 分の変数 ID を含む S2F47 メッセージを生成し、それを相手装置に送信するようエンジンに要求します。
本メソッドは送信の後、引き続き S2F48 応答メッセージを待機し、受信が終了したら、上の【戻り値】に示した値を返却します。
応答情報は、DshS2F48LimitRspList クラスのｲﾝｽﾀﾝｽ rsp_info に渡されます。

応答メッセージを受信するまでプログラムはブロック（待ち）状態になります。

送信エラー、受信エラーを検出した場合は、ei ＜ 0 の値を返却します。(T3 ﾀｲﾑｱｳﾄも含む)

T3 タイムアウトが発生した場合、DSHDR2 HSMS ドライバーに設定されている T3 プロトコルタイムアウト時間だけブロックされることになります。

６．トレース(TRACE)関連メッセージの送受信

以下のメッセージがあります。

1	S2F23,24	DshS2F23Send	S2F23 送信　 Trace Initialize Send
		DshS2F24Response	S2F24 応答
2	S6F1,S6F2	－	(S6F1 はｴﾝｼﾞﾝが自動送信) Trace Data Send
		DshS6F2Response	S6F2 応答

６．１　DshS2F23Send クラス

S2F23 メッセージを送信し、S2F24 応答メッセージを受信するためのクラスです。

６．１．１　コンストラクタ

オーバーロードの一覧を示します。

	名前	説明
1	public DshS2F23Send()	装置 ID=0 のｲﾝｽﾀﾝｽを生成します。
2	public DshS2F23Send(int eqid)	装置 ID を指定してｲﾝｽﾀﾝｽを生成します。

６．１．２　プロパティ

プロパティ一覧表に示します。

	名前	説明
1	public int eqid	装置 ID です。ﾃﾞﾌｫﾙﾄ値=0 です。
2	public DshTrace tr_info	ﾄﾚｰｽ情報を保存するｸﾗｽです。

６．１．３　メソッド

本クラスのメソッドは次の通りです。

	名前	説明
1	public void set_info()	準備した DshTrace のｲﾝｽﾀﾝｽの内容を tr_info に設定します。 Vol-1 の 6.1 参照
2	public int send_s2f23() public int send()	S2F23 ﾒｯｾｰｼﾞを送信します。
3	public int send_wait()	S2F23 ﾒｯｾｰｼﾞを送信し、S2F24 を受信します。 ﾌﾟﾛｸﾞﾗﾑは応答受信までﾌﾞﾛｯｸされます。
4	public void Dispose()	ｸﾗｽ内部で使用された資源(ﾒﾓﾘ)ｼｽﾃﾑに返却します。 他のｸﾗｽをﾌﾟﾛﾊﾟﾃｨにしている場合、そのｸﾗｽの Dispose()も行います。 3.1.3.5 と同様です。そちらを参照ください。

６．１．３．１　set_info()

ユーザが準備したDshTraceのクラスのインスタンスの内容を当該インスタンスのtr_infoに設定します。

【構文】

public void set_info(ref DshTrace info)

【引数】

info

設定したい情報が保存されているDshTraceのインスタンスです。
ユーザが準備します。

【戻り値】

なし。

【説明】

infoの引数のトレースの内容を当該インスタンスのtr_infoにコピーします。

６．１．３．２　send_s2f23(), send()

S2F23 メッセージの送信要求をします。

【構文】

```
public int send_s2f23(ref int tiaack,
                              DshCallback.callback_s2f23 callback, uint upara)
public int send(ref int tiaack,
                              DshCallback.callback_s2f23 callback, uint upara)
```

【引数】

tiaack

S2F24 応答メッセージの ACK 情報保存用です。コールバック関数の引数になります。

callback

S2F23 送信後、2 次メッセージ受信した時に呼び出される callback 関数です。

upara

ユーザパラメータ情報です。コールバックされる際に callback 関数の引数として渡されます。
要求とコールバック間のタグ情報として使用できます。

【戻り値】

返却値	意　味
0	要求が受け入れられた。
(-1)	要求が受け入れられなかった。

【説明】

当該インスタンスの tr_info に保存されているトレース情報から S2F23 メッセージを生成し、それを相手装置に送信するようエンジンに要求します。
要求がエンジンによって受け入れられたときは、0 を返却します。受け入れられなかった場合は(-1)を返却します。
送信後、受信した応答メッセージの tiaack を引数にして callback 関数を呼び出します。
callback 関数からは 0 を返却してください。

【callback の構文】

```
public delegate int callback_s2f23(
    int eqid,                       // 装置 ID
    int end_status,                 // 終了状態ｺｰﾄﾞ
    ref int tiaack,                 // S2F24 に含まれる tiaack です。
    uint upara                      // ﾕｰｻﾞﾊﾟﾗﾒｰﾀ(送信要求ﾒｿｯﾄﾞで指定された upara)
)
```

end_status に返却される値は、=0 の場合は正常終了を意味します。

end_status	意　味
0	送受信が正常に完了した。
-14	T3 ﾀｲﾑｱｳﾄを検出し、応答ﾒｯｾｰｼﾞを受信できなかった。
(-1)	送信できなかった。

６．１．３．３　send_wait()

S2F23 メッセージの送信要求をし,引き続き応答メッセージも受信します。

【構文】

public int send_wait(ref int tiaack)

【引数】

tiaack

S2F24 応答メッセージに含まれる ACK 保存用です。

【戻り値】

返却値	意　味
0	正常に送受信した。
＜ 0	送受信ｴﾗｰが発生した。（送信ｴﾗｰ or T3 ﾀｲﾑｱｳﾄなど）

【説明】

当該インスタンスの tr_info に保存されているトレース情報から S2F23 メッセージを生成し、それを相手装置に送信するようエンジンに要求します。
本メソッドは送信の後、引き続き S2F24 応答メッセージを待機し、受信が終了したら、上の【戻り値】に示した値を返却します。
応答情報は、ACK を tiaack に渡されます。

応答メッセージを受信するまでプログラムはブロック（待ち）状態になります。

送信エラー、受信エラーを検出した場合は、ei ＜ 0 の値を返却します。(T3 ﾀｲﾑｱｳﾄも含む)

T3 タイムアウトが発生した場合、DSHDR2 HSMS ドライバーに設定されている T3 プロトコルタイムアウト時間だけブロックされることになります。

６．２　DshS2F24Response クラス

S2F23 メッセージを送信し、S2F24 応答メッセージを受信するためのクラスです。

６．２．１　コンストラクタ

オーバーロードの一覧を示します。

	名前	説明
1	public DshS2F24Response()	装置 ID=0 のｲﾝｽﾀﾝｽを生成します。
2	public DshS2F24Response(int eqid)	装置 ID を指定してｲﾝｽﾀﾝｽを生成します。

６．２．２　プロパティ

プロパティ一覧表に示します。

	名前	説明
1	public int eqid	装置 ID です。ﾃﾞﾌｫﾙﾄ値=0 です。

６．２．３　メソッド

本クラスのメソッドは次の通りです。

	名前	説明
1	public int response()	S2F24 応答ﾒｯｾｰｼﾞを送信します。
2	public void Dispose()	ｸﾗｽ内部で使用された資源(ﾒﾓﾘ)ｼｽﾃﾑに返却します。 他のｸﾗｽをﾌﾟﾛﾊﾟﾃｨにしている場合、そのｸﾗｽの Dispose()も行います。 3.1.3.5 と同様です。そちらを参照ください。

６．２．３．１　response()

S2F24 メッセージの応答送信要求をします。

【構文】

public int response(uint trid, int tiaack)

【引数】

trid

エンジンからポーリング時に与えられたトランザクション ID です。
処理した１次メッセージとの対応を取ります。

tiaack

S2F24 メッセージのための応答 ACK です。

【戻り値】

返却値	意　味
0	要求が受け入れられた。
(-1)	要求が受け入れられなかった。(ﾄﾗﾝｻﾞｸｼｮﾝ ID が正しくなかった。)

【説明】

受信した S2F23 に対する S2F24 メッセージを送信します。

tiaack は S2F24 に設定する ACK です。
trid は、この応答メッセージに対する S2F45 受信時にエンジンから与えられたトランザクション ID です。
trid がエンジンの管理の中に見つからなかった場合には、(-1)を返却します。
要求が受け入れられたら 0 を返却します。この後、応答メッセージ送信結果は特に通知されません。

６．３　DshS6F2Response クラス

S6F1 メッセージを受信した後、S6F2 応答メッセージを送信するためのクラスです。

６．３．１　コンストラクタ

オーバーロードの一覧を示します。

	名前	説明
1	public DshS6F2Response()	装置 ID=0 のｲﾝｽﾀﾝｽを生成します。
2	public DshS6F2Response(int eqid)	装置 ID を指定してｲﾝｽﾀﾝｽを生成します。

６．３．２　プロパティ

プロパティ一覧表に示します。

	名前	説明
1	public int eqid	装置 ID です。ﾃﾞﾌｫﾙﾄ値=0 です。

６．３．３　メソッド

本クラスのメソッドは次の通りです。

	名前	説明
1	public int response()	S6F2 応答ﾒｯｾｰｼﾞを送信します。
2	public void Dispose()	ｸﾗｽ内部で使用された資源(ﾒﾓﾘ)ｼｽﾃﾑに返却します。 他のｸﾗｽをﾌﾟﾛﾊﾟﾃｨにしている場合、そのｸﾗｽの Dispose()も行います。 3.1.3.5 と同様です。そちらを参照ください。

６．３．３．１　response()

S6F2 メッセージの応答送信要求をします。

【構文】

public int response(uint trid, int ackc6)

【引数】

trid

エンジンからポーリング時に与えられたトランザクション ID です。
処理した１次メッセージとの対応を取ります。

ackc6

S6F2 メッセージのための ACK です。

【戻り値】

返却値	意　味
0	要求が受け入れられた。
(-1)	要求が受け入れられなかった。(ﾄﾗﾝｻﾞｸｼｮﾝ ID が正しくなかった。)

【説明】

受信した S6F1 に対する S6F2 メッセージを送信します。

応答メッセージに含める情報は ackc6 です。
trid は、この応答メッセージに対する S6F1 受信時にエンジンから与えられたトランザクション ID です。
trid がエンジンの管理の中に見つからなかった場合には、(-1)を返却します。
要求が受け入れられたら 0 を返却します。この後、応答メッセージ送信結果は特に通知されません。

７．収集イベントとレポート関連メッセージの送受信

以下のメッセージがあります。

1	S2F35,36	DshS2F35Send	S2F35 送信　Link Event Report
		－	(S2F36 はｴﾝｼﾞﾝが自動応答)
2	S2F37,38	DshS2F37Send	S2F37 送信　Enable/Disabel Event Report
		－	(S2F38 はｴﾝｼﾞﾝが自動応答)
3	S2F33,34	DshS2F33Send	S2F33 送信　Define Report
		－	(S2F34 はｴﾝｼﾞﾝが自動応答)
4	S6F11,S6F12	DshS6F11Send	S6F11 送信　Event Report Send
		DshS6F12Response	S6F12 応答
5	S6F15,S6F16	DshS6F15Send	S6F15 送信　Event Report Request
		－	(S6F16 はｴﾝｼﾞﾝが自動応答)
6	S6F19,S6F20	DshS6F19Send	S6F19 送信　Indivisual Report Data
		－	(S6F20 はｴﾝｼﾞﾝが自動応答)

７．１　DshS2F35Send クラス

S2F35 メッセージを送信し、S2F36 応答メッセージを受信するためのクラスです。

７．１．１　コンストラクタ

オーバーロードの一覧を示します。

	名前	説明
1	public DshS2F35Send()	装置 ID=0 のｲﾝｽﾀﾝｽを生成します。
2	public DshS2F35Send(int eqid)	装置 ID を指定してｲﾝｽﾀﾝｽを生成します。

７．１．２　プロパティ

プロパティ一覧表に示します。

	名前	説明
1	public int eqid	装置 ID です。ﾃﾞﾌｫﾙﾄ値=0 です。
2	public int count	3 の list 配列に保存されているﾘﾝｸ情報の数です。 count=0 で send_s2f35()を実行すると、全 CEID についてﾘﾝｸされているﾚﾎﾟｰﾄ ID の解除指定になります。
3	public DshCeRpList[] list	ｲﾍﾞﾝﾄにﾘﾝｸされているﾚﾎﾟｰﾄ情報を保存する DshCeRpList ｸﾗｽの配列です。 DshCeRpList については、Vol-1 7.5 参照

７．１．３　メソッド

本クラスのメソッドは次の通りです。

	名前	説明
1	public void clear()	list を空にし、count = 0 にします。
2	public void add_ceid()	list 配列に、引数に指定された ceid を与えて DshCeRpList ｸﾗｽのｲﾝｽﾀﾝｽを生成し、追加します。
3	public void add_rpid()	list 配列の指定位置の DshCeRpList ｸﾗｽのｲﾝｽﾀﾝｽに rpid を追加します。
4	public int send_s2f35() public int send()	S2F35 ﾒｯｾｰｼﾞを送信します。
5	public int send_wait()	S2F35 ﾒｯｾｰｼﾞを送信し、S2F36 を受信します。 ﾌﾟﾛｸﾞﾗﾑは応答受信までﾌﾞﾛｯｸされます。
6	public void Dispose()	ｸﾗｽ内部で使用された資源(ﾒﾓﾘ)ｼｽﾃﾑに返却します。 他のｸﾗｽをﾌﾟﾛﾊﾟﾃｨにしている場合、そのｸﾗｽの Dispose()も行います。 3.1.3.5 と同様です。そちらを参照ください。

７．１．３．１　clear()

イベントリンク情報が保存されている list 配列を空にします。

【構文】

public void clear()

【引数】

なし

【戻り値】

なし。

【説明】

イベントリンク情報が保存されている list 配列を空にし、count = 0 にします。

７．１．３．２　add_ceid()

list 配列に、引数に指定された ceid を与えて DshCeRpList クラスのインスタンスを生成し、追加します。

【構文】

public void add_ceid(uint ceid)

【引数】

ceid

DshCeRpList クラスのインスタンスに設定する CEID、イベント ID です。

【戻り値】

なし。

【説明】

イベントリンク情報を保存する DshCeRpList クラスのインスタンスを生成し、引数で与えられた ceid を設定します。そして、それを list 配列に追加します。そして、count+1 します。

DshCeRpList クラスにレポート ID を設定するためのメソッドとして、add_rpid()メソッドがあります。

７．１．３．３　add_rpid()

list 配列の指定位置の DshCeRpList クラスのインスタンスに 1 個レポート ID を追加します。

【構文】

public int add_rpid(int index, uint rpid)

【引数】

index

レポート ID を追加したい list 配列の位置を指定します。

rpid

DshCeRpList クラスのインスタンスに追加したいレポート ID です。

【戻り値】

返却値	意　味
0	正常に追加できた。
(-1)	追加できなかった。(index >=count であった)

【説明】

イベントリンク情報を保存する list 配列の index 位置のクラスのインスタンスに rpid で与えられたレポート ID を追加します。

もし、index で指定された位置にインスタンスが存在しなかった場合（index >= count）には、(-1)を返却します。

７．１．３．４　send_s2f35(), send()

S2F35 メッセージの送信要求をします。

【構文】

```
public int send_s2f35(ref int lrack, DshCallback.callback_s2f35 callback, uint upara)
public int send(ref int lrack, DshCallback.callback_s2f35 callback, uint upara)
```

【引数】

lrack

S2F36 応答メッセージの ACK 情報保存用です。コールバック関数の引数になります。

callback

S2F35 送信後、2 次メッセージ受信した時に呼び出される callback 関数です。

upara

ユーザパラメータ情報です。コールバックされる際に callback 関数の引数として渡されます。
要求とコールバック間のタグ情報として使用できます。

【戻り値】

返却値	意　味
0	要求が受け入れられた。
(-1)	要求が受け入れられなかった。

【説明】

当該インスタンスの list 配列に保存されているイベントリンク情報から S2F35 メッセージを生成し、それを相手装置に送信するようエンジンに要求します。
プロパティ count=0 の場合、全 CEID のレポートの解除になります。

要求がエンジンによって受け入れられたときは、0 を返却します。受け入れられなかった場合は(-1)を返却します。
送信後、受信した応答メッセージの lrack を引数にして callback 関数を呼び出します。
callback 関数からは 0 を返却してください。

【callback の構文】

```
public delegate int callback_s2f35(
    int eqid,                       // 装置 ID
    int end_status,                 // 終了状態ｺｰﾄﾞ
    ref int lrack,                  // S2F36 に含まれる lrack です。
    uint upara                      // ﾕｰｻﾞﾊﾟﾗﾒｰﾀ(送信要求ﾒｿｯﾄﾞで指定された upara)
)
```

end_status に返却される値は、=0 の場合は正常終了を意味します。

end_status	意　味
0	送受信が正常に完了した。
-14	T3 ﾀｲﾑｱｳﾄを検出し、応答ﾒｯｾｰｼﾞを受信できなかった。
(-1)	送信できなかった。

７．１．３．５　send_wait()

S2F35 メッセージの送信要求をし,引き続き応答メッセージも受信します。

【構文】

public int send_wait(ref int lrack)

【引数】

lrack

S2F36 応答メッセージの ACK 情報保存用です。

【戻り値】

返却値	意　味
0	正常に送受信した。
＜ 0	送受信ｴﾗｰが発生した。（送信ｴﾗｰ or T3 ﾀｲﾑｱｳﾄなど）

【説明】

当該インスタンスの list 配列に保存されているイベントリンク情報から S2F35 メッセージを生成し、それを相手装置に送信するようエンジンに要求します。
本メソッドは送信の後、引き続き S2F36 応答メッセージを待機し、受信が終了したら、上の【戻り値】に示した値を返却します。
応答情報は、ACK を lrack に渡されます。

応答メッセージを受信するまでプログラムはブロック（待ち）状態になります。

送信エラー、受信エラーを検出した場合は、ei ＜ 0 の値を返却します。(T3 ﾀｲﾑｱｳﾄも含む)

T3 タイムアウトが発生した場合、DSHDR2 HSMS ドライバーに設定されている T3 プロトコルタイムアウト時間だけブロックされることになります。

７．２　DshS2F37Send クラス

S2F37 メッセージを送信し、S2F38 応答メッセージを受信するためのクラスです。
CEID を指定して Enable/Disable の設定を行います。

７．２．１　コンストラクタ

オーバーロードの一覧を示します。

	名前	説明
1	public DshS2F37Send()	装置 ID=0 のｲﾝｽﾀﾝｽを生成します。
2	public DshS2F37Send(int eqid)	装置 ID を指定してｲﾝｽﾀﾝｽを生成します。

７．２．２　プロパティ

プロパティ一覧表に示します。

	名前	説明
1	public int eqid	装置 ID です。ﾃﾞﾌｫﾙﾄ値=0 です。
2	public int count	3 の id_list 配列に保存されている CEID の数です。 count=0 で send_s2f37()関数を実行すると全 CEID の指定になります。
3	public uint[] id_list	Enable/Disable する対象の CEID を保存する配列です。
4	**public void Dispose()**	**ｸﾗｽ内部で使用された資源(ﾒﾓﾘ)ｼｽﾃﾑに返却します。 他のｸﾗｽをﾌﾟﾛﾊﾟﾃｨにしている場合、そのｸﾗｽの Dispose()も行います。 3.1.3.5 と同様です。そちらを参照ください。**

７．２．３　メソッド

本クラスのメソッドは次の通りです。

	名前	説明
1	public void clear()	id_list を空にし、count = 0 にします。
2	public void add_ceid()	id_list 配列に、引数に指定された ceid を追加します。
3	public int send_s2f37() public int send()	S2F37 ﾒｯｾｰｼﾞを送信します。
4	public int send_wait()	S2F37 ﾒｯｾｰｼﾞを送信し、S2F38 を受信します。 ﾌﾟﾛｸﾞﾗﾑは応答受信までﾌﾞﾛｯｸされます。
5	public void Dispose()	ｸﾗｽ内部で使用された資源(ﾒﾓﾘ)ｼｽﾃﾑに返却します。 他のｸﾗｽをﾌﾟﾛﾊﾟﾃｨにしている場合、そのｸﾗｽの Dispose()も行います。 3.1.3.5 と同様です。そちらを参照ください。

７．２．３．１　clear()

イベントリンク情報が保存されている id_list 配列を空にします。

【構文】

public void clear()

【引数】

なし

【戻り値】

なし。

【説明】

イベント ID が保存されている id_list 配列を空にし、count = 0 にします。

７．２．３．２　add_ceid()

id_list 配列に、引数に指定された ceid を追加します。

【構文】

public void add_ceid(uint ceid)

【引数】

ceid

イベント ID です。

【戻り値】

なし。

【説明】

id_list 配列に引数で与えられた ceid を追加します。そして、count+1 します。

７．２．３．３　send_s2f37(), send()

S2F37 メッセージの送信要求をします。

【構文】

public int send_s2f37(int ceed, ref int erack, DshCallback.callback_s2f37 callback, uint upara)
public int send(int ceed, ref int erack, DshCallback.callback_s2f37 callback, uint upara)

【引数】

ceed
　　イベント ID を Enable(有効)にするか Disable(無効)にするかを指定します。
　　=1 は Enable を意味し、 =0 は Disable を意味します。
erack
　　S2F38 応答メッセージの ACK 情報保存用です。コールバック関数の引数になります。
callback
　　S2F37 送信後、2 次メッセージ受信した時に呼び出される callback 関数です。
upara
　　ユーザパラメータ情報です。コールバックされる際に callback 関数の引数として渡されます。
　　要求とコールバック間のタグ情報として使用できます。

【戻り値】

返却値	意　味
0	要求が受け入れられた。
(-1)	要求が受け入れられなかった。

【説明】

当該インスタンスの id_list 配列に保存されているイベント ID 情報から S2F37 メッセージを生成し、それを相手装置に送信するようエンジンに要求します。
プロパティ count=0 の場合は、全 CEID が対象になります。

要求がエンジンによって受け入れられたときは、0 を返却します。受け入れられなかった場合は(-1)を返却します。
送信後、受信した応答メッセージの erack を引数にして callback 関数を呼び出します。
callback 関数からは 0 を返却してください。

【callback の構文】

```
public delegate int callback_s2f37(
    int eqid,                       // 装置 ID
    int end_status,                 // 終了状態ｺｰﾄﾞ
    ref int erack,                  // S2F38 に含まれる erack です。
    uint upara                      // ﾕｰｻﾞﾊﾟﾗﾒｰﾀ(送信要求ﾒｿｯﾄﾞで指定された upara)
)
```

end_status に返却される値は、=0 の場合は正常終了を意味します。

end_status	意　味
0	送受信が正常に完了した。
-14	T3 ﾀｲﾑｱｳﾄを検出し、応答ﾒｯｾｰｼﾞを受信できなかった。
(-1)	送信できなかった。

７．２．３．４　send_wait()

S2F37 メッセージの送信要求をし,引き続き応答メッセージも受信します。

【構文】

public int send_wait(int ceed, ref int erack)

【引数】

ceed

イベント ID を Enable(有効)にするか Disable(無効)にするかを指定します。
=1 は Enable を意味し、 =0 は Disable を意味します。

erack

S2F38 応答メッセージの ACK 情報保存用です。

【戻り値】

返却値	意　味
0	正常に送受信した。
＜ 0	送受信ｴﾗｰが発生した。(送信ｴﾗｰ or T3 ﾀｲﾑｱｳﾄなど)

【説明】

当該インスタンスの id_list 配列に保存されているイベント ID 情報から S2F37 メッセージを生成し、それを相手装置に送信するようエンジンに要求します。
プロパティ count=0 の場合は、全 CEID が対象になります。

本メソッドは送信の後、引き続き S2F38 応答メッセージを待機し、受信が終了したら、上の【戻り値】に示した値を返却します。
応答情報は、ACK を erack に渡されます。

応答メッセージを受信するまでプログラムはブロック（待ち）状態になります。

送信エラー、受信エラーを検出した場合は、ei ＜ 0 の値を返却します。(T3 ﾀｲﾑｱｳﾄも含む)

T3 タイムアウトが発生した場合、DSHDR2 HSMS ドライバーに設定されている T3 プロトコルタイムアウト時間だけブロックされることになります。

７．３　DshS2F33Send クラス

S2F33 メッセージを送信し、S2F34 応答メッセージを受信するためのクラスです。

７．３．１　コンストラクタ

オーバーロードの一覧を示します。

	名前	説明
1	public DshS2F33Send()	装置 ID=0 のｲﾝｽﾀﾝｽを生成します。
2	public DshS2F33Send(int eqid)	装置 ID を指定してｲﾝｽﾀﾝｽを生成します。

７．３．２　プロパティ

プロパティ一覧表に示します。

	名前	説明
1	public int eqid	装置 ID です。ﾃﾞﾌｫﾙﾄ値=0 です。
2	public int count	3 の list 配列に保存されているﾘﾝｸ情報の数です。 count=0 で send_s2f33()を実行すると、全ﾚﾎﾟｰﾄ ID の削除になります。
3	public DshRpVList[] list	ﾚﾎﾟｰﾄにﾘﾝｸされている変数情報を保存する DshRpVList ｸﾗｽの配列です。 DshRpVList については、Vol-1 7.6 参照 list.count = 0 の場合は、当該ﾚﾎﾟｰﾄ ID のﾘﾝｸが解除されます。

７．３．３　メソッド

本クラスのメソッドは次の通りです。

	名前	説明
1	public void clear()	list を空にし、count = 0 にします。
2	public void add_rpid()	list 配列に、引数に指定された rpid を与えて DshRpVList ｸﾗｽのｲﾝｽﾀﾝｽを生成し、追加します。
3	public void add_vid()	list 配列の指定位置の DshRpVList ｸﾗｽのｲﾝｽﾀﾝｽに vid を追加します。
4	public int send_s2f33() public int send()	S2F33 ﾒｯｾｰｼﾞを送信します。
5	public int send_wait()	S2F33 ﾒｯｾｰｼﾞを送信し、S2F34 を受信します。 ﾌﾟﾛｸﾞﾗﾑは応答受信までﾌﾞﾛｯｸされます。
6	public void Dispose()	ｸﾗｽ内部で使用された資源(ﾒﾓﾘ)ｼｽﾃﾑに返却します。 他のｸﾗｽをﾌﾟﾛﾊﾟﾃｨにしている場合、そのｸﾗｽの Dispose()も行います。 3.1.3.5 と同様です。そちらを参照ください。

７．３．３．１　clear()

レポートリンク情報が保存されている list 配列を空にします。

【構文】

public void clear()

【引数】

なし

【戻り値】

なし。

【説明】

レポートリンク情報が保存されている list 配列を空にし、count = 0 にします。

７．３．３．２　add_rpid()

list 配列に、引数に指定された rpid を与えて DshRpVList クラスのインスタンスを生成し、追加します。

【構文】

public void add_rpid(uint rpid)

【引数】

rpid

DshRpVList クラスのインスタンスに設定するレポート ID です。

【戻り値】

なし。

【説明】

レポートリンク情報を保存する DshRpVList クラスのインスタンスを生成し、引数で与えられた rpid を設定します。そして、それを list 配列に追加します。そして、count+1 します。

DshRpVList クラスに変数 ID を設定するためのメソッドとして、add_vid()メソッドがあります。

７．３．３．３　add_vid()

list 配列の指定位置の DshRpVList クラスのインスタンスに 1 個変数 ID を追加します。

【構文】

public int add_vid(int index, uint vid)

【引数】

index

変数 ID を追加したい list 配列の位置を指定します。

vid

DshRpVList クラスのインスタンスに追加したい変数 ID です。

【戻り値】

返却値	意　味
0	正常に追加できた。
(-1)	追加できなかった。(index >=count であった)

【説明】

レポートリンク情報を保存する list 配列の index 位置のクラスのインスタンスに vid で与えられた変数 ID を追加します。

もし、index で指定された位置にインスタンスが存在しなかった場合（index >= count）には、(-1)を返却します。

７．３．３．４　send_s2f33()

S2F33 メッセージの送信要求をします。

【構文】

```
public int send_s2f33(ref int drack, DshCallback.callback_s2f33 callback, uint upara)
public int send(ref int drack, DshCallback.callback_s2f33 callback, uint upara)
```

【引数】

drack

S2F34 応答メッセージの ACK 情報保存用です。コールバック関数の引数になります。

callback

S2F33 送信後、2 次メッセージ受信した時に呼び出される callback 関数です。

upara

ユーザパラメータ情報です。コールバックされる際に callback 関数の引数として渡されます。
要求とコールバック間のタグ情報として使用できます。

【戻り値】

返却値	意　味
0	要求が受け入れられた。
(-1)	要求が受け入れられなかった。

【説明】

当該インスタンスの list 配列に保存されているレポートリンク情報から S2F33 メッセージを生成し、それを相手装置に送信するようエンジンに要求します。
プロパティ count=0 の場合は、全レポート ID の削除指定になります。

要求がエンジンによって受け入れられたときは、0 を返却します。受け入れられなかった場合は(-1)を返却します。
送信後、受信した応答メッセージの drack を引数にして callback 関数を呼び出します。
callback 関数からは 0 を返却してください。

【callback の構文】

```
public delegate int callback_s2f33(
    int eqid,                       // 装置 ID
    int end_status,                 // 終了状態ｺｰﾄﾞ
    ref int drack,                  // S2F34 に含まれる drack です。
    uint upara                      // ﾕｰｻﾞﾊﾟﾗﾒｰﾀ(送信要求ﾒｿｯﾄﾞで指定された upara)
)
```

end_status に返却される値は、=0 の場合は正常終了を意味します。

end_status	意　味
0	送受信が正常に完了した。
-14	T3 ﾀｲﾑｱｳﾄを検出し、応答ﾒｯｾｰｼﾞを受信できなかった。
(-1)	送信できなかった。

７．３．３．５　send_wait()

S2F33 メッセージの送信要求をし,引き続き応答メッセージも受信します。

【構文】

public int send_wait(ref int drack)

【引数】

drack

S2F34 応答メッセージの ACK 情報保存用です。

【戻り値】

返却値	意　味
0	正常に送受信した。
＜ 0	送受信ｴﾗｰが発生した。（送信ｴﾗｰ or T3 ﾀｲﾑｱｳﾄなど）

【説明】

当該インスタンスの list 配列に保存されているレポートリンク情報から S2F33 メッセージを生成し、それを相手装置に送信するようエンジンに要求します。
プロパティ count=0 の場合は、全レポート ID の削除指定になります。

本メソッドは送信の後、引き続き S2F34 応答メッセージを待機し、受信が終了したら、上の【戻り値】に示した値を返却します。
応答情報は、ACK を drack に渡されます。

応答メッセージを受信するまでプログラムはブロック（待ち）状態になります。

送信エラー、受信エラーを検出した場合は、ei ＜ 0 の値を返却します。(T3 ﾀｲﾑｱｳﾄも含む)

T3 タイムアウトが発生した場合、DSHDR2 HSMS ドライバーに設定されている T3 プロトコルタイムアウト時間だけブロックされることになります。

７．４　DshS6F11Send クラス

S6F11 メッセージを送信し、S6F12 応答メッセージを受信するためのクラスです。
イベントレポートを送信します。

７．４．１　コンストラクタ

オーバーロードの一覧を示します。

	名前	説明
1	public DshS6F11Send()	装置 ID=0 のｲﾝｽﾀﾝｽを生成します。
2	public DshS6F11Send(uint ceid)	ｲﾍﾞﾝﾄ ID を指定して装置 ID=0 のｲﾝｽﾀﾝｽを生成します。
3	public DshS6F11Send(int eqid)	装置 ID を指定してｲﾝｽﾀﾝｽを生成します。
4	public DshS6F11Send(int eqid, uint ceid)	装置 ID とｲﾍﾞﾝﾄ ID を指定してｲﾝｽﾀﾝｽを生成します。

７．４．２　プロパティ

プロパティ一覧表に示します。

	名前	説明
1	public int eqid	装置 ID です。ﾃﾞﾌｫﾙﾄ値=0 です。
2	public uint ceid	送信するｲﾍﾞﾝﾄ ID(CEID)です。

７．４．３　メソッド

本クラスのメソッドは次の通りです。

	名前	説明
1	public void set_ceid()	ｲﾍﾞﾝﾄ ID(CEID)を設定します。
2	public int send_s6f11() public int send()	S6F11 ﾒｯｾｰｼﾞを送信します。
3	public int send_wait()	S6F11 ﾒｯｾｰｼﾞを送信し、S6F12 を受信します。 ﾌﾟﾛｸﾞﾗﾑは応答受信までﾌﾞﾛｯｸされます。
4	public void Dispose()	ｸﾗｽ内部で使用された資源(ﾒﾓﾘ)ｼｽﾃﾑに返却します。 他のｸﾗｽをﾌﾟﾛﾊﾟﾃｨにしている場合、そのｸﾗｽの Dispose()も行います。 3.1.3.5 と同様です。そちらを参照ください。

７．４．３．１　set_ceid()

イベント ID を設定します。

【構文】

public void set_ceid(uint ceid)

【引数】

ceid

イベント ID (CEID) です。
このイベント ID にリンクされているレポート情報を S6F11 で送信します。

【戻り値】

なし。

【説明】

S6F11 に設定したいイベント ID を設定します。

７．４．３．２　send_s6f11(), send()

S6F11 メッセージの送信要求をします。

【構文】

```
public int send_s6f11(uint ceid, DshCallback.callback_s6f11 callback, uint upara)
public int send(uint ceid, DshCallback.callback_s6f11 callback, uint upara)
public int send_s6f11(DshCallback.callback_s6f11 callback, uint upara)
public int send(DshCallback.callback_s6f11 callback, uint upara)
```

【引数】

ceid
　　送信するイベント ID です。

callback
　　S6F11 送信後、2 次メッセージ受信した時に呼び出される callback 関数です。

upara
　　ユーザパラメータ情報です。コールバックされる際に callback 関数の引数として渡されます。
　　要求とコールバック間のタグ情報として使用できます。

【戻り値】

返却値	意　味
0	要求が受け入れられた。
300	ceed=0(Enable でない)のため送信できなかった。
(-1)	要求が受け入れられなかった。

【説明】

当該インスタンスの ceid のレポート情報をエンジンから取得し、S6F11 メッセージを生成し、それを相手装置に送信するようエンジンに要求します。
エンジンからは、ceid にリンクされたレポートにリンクされている変数 ID の現在の変数値情報を取得し、S6F11 に組み込んだ上で S6F11 メッセージを送信します。

要求がエンジンによって受け入れられたときは、0 を返却します。受け入れられなかった場合は(-1)を返却します。
送信後、受信した応答結果は引数 end_status に設定し、callback 関数を呼び出します。
callback 関数からは 0 を返却してください。

【callback の構文】

```
public delegate int callback_s6f11(
    int eqid,                       // 装置 ID
    int end_status,                 // 終了状態ｺｰﾄﾞ
    uint upara                      // ﾕｰｻﾞﾊﾟﾗﾒｰﾀ(送信要求ﾒｿｯﾄﾞで指定された upara)
)
```

end_status に返却される値は、=0 の場合は正常終了を意味します。

end_status	意　味
0	送受信が正常に完了し，S6F12 の ack=0 であった。
-14	T3 ﾀｲﾑｱｳﾄを検出し、応答ﾒｯｾｰｼﾞを受信できなかった。
(-1)	送信できなかった。(ackc6 が 0 でなかった。)

７．４．３．３　send_wait()

S6F11 メッセージの送信要求をし,引き続き応答メッセージも受信します。

【構文】

public int send_wait(uint ceid)

【引数】

ceid

送信するイベント ID です。

【戻り値】

返却値	意　味
0～255	正常に送受信した。 ackc6 の値になります。
300	ceed=0(Enable でない)のため送信できなかった。
＜ 0	送受信ｴﾗｰが発生した。(送信ｴﾗｰ or T3 ﾀｲﾑｱｳﾄなど)

【説明】

当該インスタンスの ceid のレポート情報をエンジンから取得し、S6F11 メッセージを生成し、それを相手装置に送信するようエンジンに要求します。
エンジンからは、ceid にリンクされたレポートにリンクされている変数 ID の現在の変数値情報を取得し、S6F11 に組み込んだ上で S6F11 メッセージを送信します。

本メソッドは送信の後、引き続き S6F12 応答メッセージを待機し、受信が終了したら、上の【戻り値】に示した値を返却します。ei >= 0 の場合は、ackc6 の値が渡されます。

応答メッセージを受信するまでプログラムはブロック（待ち）状態になります。

送信エラー、受信エラーを検出した場合は、ei ＜ 0 の値を返却します。(T3 ﾀｲﾑｱｳﾄも含む)

T3 タイムアウトが発生した場合、DSHDR2 HSMS ドライバーに設定されている T3 プロトコルタイムアウト時間だけブロックされることになります。

７．５　DshS6F12Response クラス

S6F11 メッセージを受信した後、S6F12 応答メッセージを送信するためのクラスです。

７．５．１　コンストラクタ

オーバーロードの一覧を示します。

	名前	説明
1	public DshS6F12Response()	装置 ID=0 のｲﾝｽﾀﾝｽを生成します。
2	public DshS6F12Response(int eqid)	装置 ID を指定してｲﾝｽﾀﾝｽを生成します。

７．５．２　プロパティ

プロパティ一覧表に示します。

	名前	説明
1	public int eqid	装置 ID です。ﾃﾞﾌｫﾙﾄ値=0 です。

７．５．３　メソッド

本クラスのメソッドは次の通りです。

	名前	説明
1	public int response()	S6F12 応答ﾒｯｾｰｼﾞを送信します。
2	public void Dispose()	ｸﾗｽ内部で使用された資源(ﾒﾓﾘ)ｼｽﾃﾑに返却します。 他のｸﾗｽをﾌﾟﾛﾊﾟﾃｨにしている場合、そのｸﾗｽの Dispose()も行います。 3.1.3.5 と同様です。そちらを参照ください。

７．５．３．１　response()

S6F12 メッセージの応答送信要求をします。

【構文】

public int response(uint trid, int ackc6)

【引数】

trid

エンジンからポーリング時に与えられたトランザクション ID です。
処理した１次メッセージとの対応を取ります。

ackc6

S6F12 メッセージのための ACK です。

【戻り値】

返却値	意　味
0	要求が受け入れられた。
(-1)	要求が受け入れられなかった。(ﾄﾗﾝｻﾞｸｼｮﾝ ID が正しくなかった。)

【説明】

受信した S6F11 に対する S6F12 メッセージを送信します。

応答メッセージに含める情報は ackc6 です。
trid は、この応答メッセージに対する S6F11 受信時にエンジンから与えられたトランザクション ID です。
trid がエンジンの管理の中に見つからなかった場合には、(-1)を返却します。
要求が受け入れられたら 0 を返却します。この後、応答メッセージ送信結果は特に通知されません。

７．６　DshS6F15Send クラス

S6F15 メッセージを送信し、S6F16 応答メッセージを受信するためのクラスです。
相手装置にイベントレポートの応答を要求します。

７．６．１　コンストラクタ

オーバーロードの一覧を示します。

	名前	説明
1	public DshS6F15Send()	装置 ID=0 のｲﾝｽﾀﾝｽを生成します。
2	public DshS6F15Send(uint ceid)	ｲﾍﾞﾝﾄ ID を指定して装置 ID=0 のｲﾝｽﾀﾝｽを生成します。
3	public DshS6F15Send(int eqid)	装置 ID を指定してｲﾝｽﾀﾝｽを生成します。
4	public DshS6F15Send(int eqid, uint ceid)	装置 ID とｲﾍﾞﾝﾄ ID を指定してｲﾝｽﾀﾝｽを生成します。

７．６．２　プロパティ

プロパティ一覧表に示します。

	名前	説明
1	public int eqid	装置 ID です。ﾃﾞﾌｫﾙﾄ値=0 です。
2	public uint ceid	送信するｲﾍﾞﾝﾄ ID(CEID)です。

７．６．３　メソッド

本クラスのメソッドは次の通りです。

	名前	説明
1	public void set_ceid()	ｲﾍﾞﾝﾄ ID(CEID)を設定します。
2	public int send_s6f15() public int send()	S6F15 ﾒｯｾｰｼﾞを送信します。
3	public int send_wait()	S6F15 ﾒｯｾｰｼﾞを送信し、S6F16 を受信します。 ﾌﾟﾛｸﾞﾗﾑは応答受信までﾌﾞﾛｯｸされます。
4	public void Dispose()	ｸﾗｽ内部で使用された資源(ﾒﾓﾘ)ｼｽﾃﾑに返却します。 他のｸﾗｽをﾌﾟﾛﾊﾟﾃｨにしている場合、そのｸﾗｽの Dispose()も行います。 3.1.3.5 と同様です。そちらを参照ください。

７．６．３．１　set_ceid()

イベント ID を設定します。

【構文】

public void set_ceid(uint ceid)

【引数】

ceid

イベント ID (CEID) です。
このイベント ID にリンクされているレポート情報を S6F15 で要求します。

【戻り値】

なし。

【説明】

S6F15 に設定したいイベント ID を設定します。

７．６．３．２　send_s6f15(), send()

S6F15 メッセージの送信要求をします。

【構文】

```
public int send_S6F15(uint ceid, ref DshCeContent rsp_info,
                              DshCallback.callback_s6f15 callback, uint upara)
public int send(uint ceid, ref DshCeContent rsp_info,
                              DshCallback.callback_s6f15 callback, uint upara)
public int send_S6F15( ref DshCeContent rsp_info,
                              DshCallback.callback_s6f15 callback, uint upara)
public int send( ref DshCeContent rsp_info,
                              DshCallback.callback_s6f15 callback, uint upara)
```

【引数】

ceid

レポート情報を求める対象のイベント ID です。

rsp_info

S6F16 応答メッセージに含まれるレポート情報保存用です。コールバック関数の引数になります。

callback

S6F15 送信後、2 次メッセージ受信した時に呼び出される callback 関数です。

upara

ユーザパラメータ情報です。コールバックされる際に callback 関数の引数として渡されます。
要求とコールバック間のタグ情報として使用できます。

【戻り値】

返却値	意　味
0	要求が受け入れられた。
(-1)	要求が受け入れられなかった。

【説明】

当該インスタンスの ceid のレポート情報の応答を S6F15 によって相手装置に要求します。
要求がエンジンによって受け入れられたときは、0 を返却します。受け入れられなかった場合は(-1)を返却します。
送信後、受信した応答メッセージのレポート情報格納用インスタンス rsp_info を引数にして callback 関数を呼び出します。callback 関数からは 0 を返却してください。

【callback の構文】

```
public delegate int callback_s6f15(
    int eqid,                          // 装置 ID
    int end_status,                    // 終了状態ｺｰﾄﾞ
    ref DshCeContent rsp_info,                 // S6F16 に含まれるﾚﾎﾟｰﾄ情報です。 Vol-1 7.3 参照
    uint upara                         // ﾕｰｻﾞﾊﾟﾗﾒｰﾀ(送信要求ﾒｿｯﾄﾞで指定された upara)
)
```

end_status に返却される値は、=0 の場合は正常終了を意味します。

end_status	意　味
0	送受信が正常に完了した。
-14	T3 ﾀｲﾑｱｳﾄを検出し、応答ﾒｯｾｰｼﾞを受信できなかった。
(-1)	送信できなかった。

７．６．３．３　send_wait()

S6F15 メッセージの送信要求をし,引き続き応答メッセージも受信します。

【構文】

```
public int send_wait(uint ceid, ref DshCeContent rsp_info)
public int send_wait(ref DshCeContent rsp_info)
```

【引数】

ceid

レポート情報を求める対象のイベント ID です。

rsp_info

S6F16 応答メッセージに含まれるレポート情報保存用です。

【戻り値】

返却値	意　味
0	正常に送受信した。
＜ 0	送受信ｴﾗｰが発生した。(送信ｴﾗｰ or T3 ﾀｲﾑｱｳﾄなど)

【説明】

当該インスタンスの ceid のレポート情報の応答を S6F15 によって相手装置に要求します。

本メソッドは送信の後、引き続き S6F16 応答メッセージを待機し、受信が終了したら、上の【戻り値】受信した応答メッセージのレポート情報はインスタンス rsp_info に渡されます。

応答メッセージを受信するまでプログラムはブロック（待ち）状態になります。

送信エラー、受信エラーを検出した場合は、ei ＜ 0 の値を返却します。(T3 ﾀｲﾑｱｳﾄも含む)

T3 タイムアウトが発生した場合、DSHDR2 HSMS ドライバーに設定されている T3 プロトコルタイムアウト時間だけブロックされることになります。

７．７　DshS6F19Send クラス

S6F19 メッセージを送信し、S6F20 応答メッセージを受信するためのクラスです。
相手装置に指定レポート ID に対するレポート情報の応答を要求します。

７．７．１　コンストラクタ

オーバーロードの一覧を示します。

	名前	説明
1	public DshS6F19Send()	装置 ID=0 のｲﾝｽﾀﾝｽを生成します。
2	public DshS6F19Send(uint rpid)	ﾚﾎﾟｰﾄ ID を指定して装置 ID=0 のｲﾝｽﾀﾝｽを生成します。
3	public DshS6F19Send(int eqid)	装置 ID を指定してｲﾝｽﾀﾝｽを生成します。
4	public DshS6F19Send(int eqid, uint rpid)	装置 ID とﾚﾎﾟｰﾄ ID を指定してｲﾝｽﾀﾝｽを生成します。

７．７．２　プロパティ

プロパティ一覧表に示します。

	名前	説明
1	public int eqid	装置 ID です。ﾃﾞﾌｫﾙﾄ値=0 です。
2	public uint rpid	送信するﾚﾎﾟｰﾄ ID(RPID)です。

７．７．３　メソッド

本クラスのメソッドは次の通りです。

	名前	説明
1	public void set_rpid()	ﾚﾎﾟｰﾄ ID(RPID)を設定します。
2	public int send_s6f19() public int send()	S6F19 ﾒｯｾｰｼﾞを送信します。
3	public int send_wait()	S6F19 ﾒｯｾｰｼﾞを送信し、S6F20 を受信します。 ﾌﾟﾛｸﾞﾗﾑは応答受信までﾌﾞﾛｯｸされます。
4	public void Dispose()	ｸﾗｽ内部で使用された資源(ﾒﾓﾘ)ｼｽﾃﾑに返却します。 他のｸﾗｽをﾌﾟﾛﾊﾟﾃｨにしている場合、そのｸﾗｽの Dispose()も行います。 3.1.3.5 と同様です。そちらを参照ください。

７．７．３．１　set_rpid()

レポート ID を設定します。

【構文】

public void set_rpid(uint rpid)

【引数】

rpid

レポート ID (RPID) です。
このレポート ID にリンクされているレポート情報を S6F19 で要求します。

【戻り値】

なし。

【説明】

S6F19 に設定したいレポート ID を設定します。

７．７．３．２　send_s6f19(), send()

S6F19 メッセージの送信要求をします。

【構文】

```
public int send_S6F19(uint rpid, ref DshRpContent rsp_info,
                              DshCallback.callback_s6f19 callback, uint upara)
public int send(uint rpid, ref DshRpContent rsp_info,
                              DshCallback.callback_s6f19 callback, uint upara)
public int send_S6F19( ref DshRpContent rsp_info,
                              DshCallback.callback_s6f19 callback, uint upara)
public int send( ref DshRpContent rsp_info,
                              DshCallback.callback_s6f19 callback, uint upara)
```

【引数】

rpid

レポート情報を要求する対象のレポート ID です。

rsp_info

S6F20 応答メッセージに含まれるレポート情報保存用です。コールバック関数の引数になります。

callback

S6F19 送信後、2 次メッセージ受信した時に呼び出される callback 関数です。

upara

ユーザパラメータ情報です。コールバックされる際に callback 関数の引数として渡されます。
要求とコールバック間のタグ情報として使用できます。

【戻り値】

返却値	意　味
0	要求が受け入れられた。
(-1)	要求が受け入れられなかった。

【説明】

当該インスタンスの rpid のレポート情報の応答を S6F19 で相手装置に要求します。
要求がエンジンによって受け入れられたときは、0 を返却します。受け入れられなかった場合は(-1)を返却します。
送信後、受信した応答メッセージのレポート情報格納用インスタンス rsp_info を引数にして callback 関数を呼び出します。callback 関数からは 0 を返却してください。

【callback の構文】

```
public delegate int callback_s6f19(
    int eqid,                          // 装置 ID
    int end_status,                    // 終了状態ｺｰﾄﾞ
    ref DshRpContent rsinfo,           // S6F20 に含まれるﾚﾎﾟｰﾄ情報です。 Vol-1 7.4 参照
    uint upara                         // ﾕｰｻﾞﾊﾟﾗﾒｰﾀ(送信要求ﾒｿｯﾄﾞで指定された upara)
)
```

end_status に返却される値は、=0 の場合は正常終了を意味します。

end_status	意　味
0	送受信が正常に完了した。
-14	T3 ﾀｲﾑｱｳﾄを検出し、応答ﾒｯｾｰｼﾞを受信できなかった。
(-1)	送信できなかった。

７．７．３．３　send_wait()

S6F19 メッセージの送信要求をし,引き続き応答メッセージも受信します。

【構文】

```
public int send_wait(uint rpid, ref DshRpContent rsp_info)
public int send_wait(ref DshRpContent rsp_info)
```

【引数】

rpid

レポート情報を要求する対象のレポート ID です。

rsp_info

S6F20 応答メッセージに含まれるレポート情報保存用です。

【戻り値】

返却値	意　味
0	正常に送受信した。
＜ 0	送受信ｴﾗｰが発生した。(送信ｴﾗｰ or T3 ﾀｲﾑｱｳﾄなど)

【説明】

当該インスタンスの rpid のレポート情報の応答を S6F19 によって相手装置に要求します。

本メソッドは送信の後、引き続き S6F20 応答メッセージを待機し、受信が終了したら、上の【戻り値】受信した応答メッセージのレポート情報はインスタンス rsp_info に渡されます。

応答メッセージを受信するまでプログラムはブロック（待ち）状態になります。

送信エラー、受信エラーを検出した場合は、ei ＜ 0 の値を返却します。(T3 ﾀｲﾑｱｳﾄも含む)

T3 タイムアウトが発生した場合、DSHDR2 HSMS ドライバーに設定されている T3 プロトコルタイムアウト時間だけブロックされることになります。

８．アラーム関連メッセージの送受信

以下のメッセージがあります。

1	S5F1,2	DshS5F1Send	S5F1 送信　　Alarm Report Send
		DshS5F2Response	S5F2 応答
2	S5F3,4	DshS5F3Send	S5F3 送信　　Enable/Disabel Alarm Send
		－	(S5F4 はｴﾝｼﾞﾝが自動応答)
3	S5F5,6	DshS5F5Send	S5F5 送信　　List Alarm Request
		－	(S5F6 はｴﾝｼﾞﾝが自動応答)

８．１　DshS5F1Send クラス

S5F1 メッセージを送信し、S5F2 応答メッセージを受信するためのクラスです。
アラームレポートを送信します。

８．１．１　コンストラクタ

オーバーロードの一覧を示します。

	名前	説明
1	public DshS5F1Send()	装置 ID=0 のｲﾝｽﾀﾝｽを生成します。
2	public DshS5F1Send(uint alid)	ｱﾗｰﾑ ID を指定して装置 ID=0 のｲﾝｽﾀﾝｽを生成します。
3	public DshS5F1Send(int eqid)	装置 ID を指定してｲﾝｽﾀﾝｽを生成します。
4	public DshS5F1Send(int eqid, uint alid)	装置 ID とｱﾗｰﾑ ID を指定してｲﾝｽﾀﾝｽを生成します。

８．１．２　プロパティ

プロパティ一覧表に示します。

	名前	説明
1	public int eqid	装置 ID です。ﾃﾞﾌｫﾙﾄ値=0 です。
2	public uint alid	送信するｱﾗｰﾑ ID(ALID)です。

８．１．３　メソッド

本クラスのメソッドは次の通りです。

	名前	説明
1	public void set_alid()	ｱﾗｰﾑ ID(ALID)を設定します。
2	public int send_s5f1() public int send()	S5F1 ﾒｯｾｰｼﾞを送信します。
3	public int send_wait()	S5F1 ﾒｯｾｰｼﾞを送信し、S5F2 を受信します。 ﾌﾟﾛｸﾞﾗﾑは応答受信までﾌﾞﾛｯｸされます。
4	public void Dispose()	ｸﾗｽ内部で使用された資源(ﾒﾓﾘ)ｼｽﾃﾑに返却します。 他のｸﾗｽをﾌﾟﾛﾊﾟﾃｨにしている場合、そのｸﾗｽの Dispose()も行います。 3.1.3.5 と同様です。そちらを参照ください。

８．１．３．１　set_alid()

アラーム ID を設定します。

【構文】

public void set_alid(uint alid)

【引数】

alid

アラーム ID (ALID) です。
このアラーム ID のアラーム情報を S5F1 で送信します。

【戻り値】

なし。

【説明】

S5F1 に設定したいアラーム ID を設定します。

８．１．３．２　send_s5f1(), send()

S5F1 メッセージの送信要求をします。

【構文】

```
public int send(uint alid, int alflag, DshCallback.callback_s5f1 callback, uint upara)
public int send_s5f1(uint alid, int alflag, DshCallback.callback_s5f1 callback, uint upara)
public int send(int alflag, DshCallback.callback_s5f1 callback, uint upara)
public int send_s5f1(int alflag, DshCallback.callback_s5f1 callback, uint upara)
```

【引数】

alid
　　アラーム ID です。
int alflag
　　アラーム発生／復旧の指定を行います。 1=発生、0=復旧です。
callback
　　S5F1 送信後、2 次メッセージ受信した時に呼び出される callback 関数です。
upara
　　ユーザパラメータ情報です。コールバックされる際に callback 関数の引数として渡されます。
　　要求とコールバック間のタグ情報として使用できます。

【戻り値】

返却値	意　味
0	要求が受け入れられた。
300	aled=0(Enable でない)のため送信できなかった。
(-1)	要求が受け入れられなかった。

【説明】

当該インスタンスの alid のアラーム情報をエンジンから取得し、S5F1 メッセージを生成し、それを相手装置に送信するようエンジンに要求します。
要求がエンジンによって受け入れられたときは、0 を返却します。受け入れられなかった場合は(-1)を返却します。
送信後、受信した応答結果は引数 end_status に設定し、callback 関数を呼び出します。
callback 関数からは 0 を返却してください。

【callback の構文】

```
public delegate int callback_s5f1(
    int eqid,                          // 装置 ID
    int end_status,                    // 終了状態ｺｰﾄﾞ
    uint upara                         // ﾕｰｻﾞﾊﾟﾗﾒｰﾀ(送信要求ﾒｿｯﾄﾞで指定された upara)
)
```

end_status に返却される値は、=0 の場合は正常終了を意味します。

end_status	意　味
0	送受信が正常に完了した。
-14	T3 ﾀｲﾑｱｳﾄを検出し、応答ﾒｯｾｰｼﾞを受信できなかった。
(-1)	送信できなかった。

８．１．３．３　send_wait()

S5F1 メッセージの送信要求をし,引き続き応答メッセージも受信します。

【構文】

```
public int send_wait(uint alid, int alflag)
public int send_wait(int alflag)
```

【引数】

alid

　　アラーム ID です。

alflag

　　アラーム発生／復旧の指定を行います。 1=発生、0=復旧です。

【戻り値】

返却値	意　味
0～255	正常に送受信した。 ackc5 の値になります。
300	aled=0(Enable でない)のため送信できなかった。
＜ 0	送受信ｴﾗｰが発生した。(送信ｴﾗｰ or T3 ﾀｲﾑｱｳﾄなど)

【説明】

当該インスタンスの alid のアラーム情報をエンジンから取得し、それに alflag を設定、S5F1 メッセージを生成し、それを相手装置に送信するようエンジンに要求します。

本メソッドは送信の後、引き続き S5F2 応答メッセージを待機し、受信が終了したら、上の【戻り値】に示した値を返却します。ei >= 0 の場合は、ackc5 の値が渡されます。

応答メッセージを受信するまでプログラムはブロック（待ち）状態になります。

送信エラー、受信エラーを検出した場合は、ei ＜ 0 の値を返却します。(T3 ﾀｲﾑｱｳﾄも含む)

T3 タイムアウトが発生した場合、DSHDR2 HSMS ドライバーに設定されている T3 プロトコルタイムアウト時間だけブロックされることになります。

８．２　DshS5F2Response クラス

S5F1 メッセージを受信した後、S5F2 応答メッセージを送信するためのクラスです。

８．２．１　コンストラクタ

オーバーロードの一覧を示します。

	名前	説明
1	public DshS5F2Response()	装置 ID=0 のｲﾝｽﾀﾝｽを生成します。
2	public DshS5F2Response(int eqid)	装置 ID を指定してｲﾝｽﾀﾝｽを生成します。

８．２．２　プロパティ

プロパティ一覧表に示します。

	名前	説明
1	public int eqid	装置 ID です。ﾃﾞﾌｫﾙﾄ値=0 です。

８．２．３　メソッド

本クラスのメソッドは次の通りです。

	名前	説明
1	public int response()	S5F2 応答ﾒｯｾｰｼﾞを送信します。
2	public void Dispose()	ｸﾗｽ内部で使用された資源(ﾒﾓﾘ)ｼｽﾃﾑに返却します。 他のｸﾗｽをﾌﾟﾛﾊﾟﾃｨにしている場合、そのｸﾗｽの Dispose()も行います。 3.1.3.5 と同様です。そちらを参照ください。

８．２．３．１　response()

S5F2 メッセージの応答送信要求をします。

【構文】

public int response(uint trid, int ackc5)

【引数】

trid

エンジンからポーリング時に与えられたトランザクション ID です。
処理した１次メッセージとの対応を取ります。

ackc5

S5F2 メッセージのための ACK です。

【戻り値】

返却値	意　味
0	要求が受け入れられた。
(-1)	要求が受け入れられなかった。(ﾄﾗﾝｻﾞｸｼｮﾝ ID が正しくなかった。)

【説明】

受信した S5F1 に対する S5F2 メッセージを送信します。

応答メッセージに含める情報は ackc5 です。
trid は、この応答メッセージに対する S5F1 受信時にエンジンから与えられたトランザクション ID です。
trid がエンジンの管理の中に見つからなかった場合には、(-1)を返却します。
要求が受け入れられたら 0 を返却します。この後、応答メッセージ送信結果は特に通知されません。

８．３　DshS5F3Send クラス

S5F3 メッセージを送信し、S5F4 応答メッセージを受信するためのクラスです。
ALID を指定して Enable/Disable の設定を行います。

８．３．１　コンストラクタ

オーバーロードの一覧を示します。

	名前	説明
1	public DshS5F3Send()	装置 ID=0 のｲﾝｽﾀﾝｽを生成します。
2	public DshS5F3Send(int eqid)	装置 ID を指定してｲﾝｽﾀﾝｽを生成します。

８．３．２　プロパティ

プロパティ一覧表に示します。

	名前	説明
1	public int eqid	装置 ID です。ﾃﾞﾌｫﾙﾄ値=0 です。
2	public uint alid	Enable/Disable 対象にする ALID です。
3	public int aled	Enable/Disable の指定ﾌﾗｸﾞです。 1=Enable（有効）, 0=Disable(無効)を意味します。
4	public int all_flag	Enable/Disableする対象ALIDを全てのｱﾗｰﾑIDに対して行うかどうかを指定します。 all_flag=0 であれば、対象になる ALID は上の alid で指定されたものだけになります。 all_flag=1 を指定した場合はｴﾝｼﾞﾝに登録されている全ｱﾗｰﾑ ID が対象になります。この場合、上の alid の意味はありません。

８．３．３　メソッド

本クラスのメソッドは次の通りです。

	名前	説明
1	public void set_info()	Enable/Disable する対象の alid, aled, all_flag を設定します。
2	public int send_s5f3() public int send()	S5F3 ﾒｯｾｰｼﾞを送信します。
3	public int send_wait()	S5F3 ﾒｯｾｰｼﾞを送信し、S5F4 を受信します。 ﾌﾟﾛｸﾞﾗﾑは応答受信までﾌﾞﾛｯｸされます。
4	public void Dispose()	ｸﾗｽ内部で使用された資源(ﾒﾓﾘ)ｼｽﾃﾑに返却します。 他のｸﾗｽをﾌﾟﾛﾊﾟﾃｨにしている場合、そのｸﾗｽの Dispose()も行います。 3.1.3.5 と同様です。そちらを参照ください。

８．３．３．１　set_info()

Enable/Disable する対象の alid, aled, all_flag を設定します。

【構文】

public void set_info(uint alid, int aled, int all_flag)

【引数】

alid

アラーム ID です。

aled

Enable(有効) / Disable(無効)を指定するフラグです。
1=Enable, 0=Disable の意味になります。

all_flag

全アラーム ID を対象とするかどうかを指定します。
0 の場合は、alid で指定したアラームだけが対象になります。
1 の場合は、エンジンが管理している全てのアラーム ID が対象になります。この場合、alid の意味はありません。

【戻り値】

なし。

【説明】

S5F3 メッセージ送信に必要なプロパティ情報を設定します。
この後、S5F3 の送信になります。

８．３．３．２　send_s5f3(), send()

S5F3 メッセージの送信要求をします。

【構文】

```
public int send_s5f3(ref int ackc5, DshCallback.callback_s5f3 callback, uint upara)
public int send(ref int ackc5, DshCallback.callback_s5f3 callback, uint upara)
```

【引数】

ackc5

S5F4 応答メッセージの ACK 情報保存用です。コールバック関数の引数になります。

callback

S5F3 送信後、2 次メッセージ受信した時に呼び出される callback 関数です。

upara

ユーザパラメータ情報です。コールバックされる際に callback 関数の引数として渡されます。
要求とコールバック間のタグ情報として使用できます。

【戻り値】

返却値	意　味
0	要求が受け入れられた。
(-1)	要求が受け入れられなかった。

【説明】

当該インスタンスの alid、all_flag, aled のプロパティ値に指定される情報に従って S5F3 メッセージを生成し、それを相手装置に送信するようエンジンに要求します。
要求がエンジンによって受け入れられたときは、0 を返却します。受け入れられなかった場合は(-1)を返却します。
送信後、受信した応答メッセージの ackc5 を引数にして callback 関数を呼び出します。
callback 関数からは 0 を返却してください。

【callback の構文】

```
public delegate int callback_s5f3(
    int eqid,                       // 装置 ID
    int end_status,                 // 終了状態ｺｰﾄﾞ
    ref int ackc5,                  // S5F4 に含まれる ackc5 です。
    uint upara                      // ﾕｰｻﾞﾊﾟﾗﾒｰﾀ(送信要求ﾒｿｯﾄﾞで指定された upara)
)
```

end_status に返却される値は、=0 の場合は正常終了を意味します。

end_status	意　味
0	送受信が正常に完了した。
-14	T3 ﾀｲﾑｱｳﾄを検出し、応答ﾒｯｾｰｼﾞを受信できなかった。
(-1)	送信できなかった。

８．３．３．３　send_wait()

S5F3 メッセージの送信要求をし,引き続き応答メッセージも受信します。

【構文】

public int send_wait(ref int ackc5)

【引数】

ackc5

S5F2 応答メッセージの ACK 情報保存用です。

【戻り値】

返却値	意　味
0	正常に送受信した。
＜ 0	送受信ｴﾗｰが発生した。（送信ｴﾗｰ or T3 ﾀｲﾑｱｳﾄなど）

【説明】

当該インスタンスの alid、all_flag, aled のプロパティ値に指定される情報に従って S5F3 メッセージを生成し、それを相手装置に送信するようエンジンに要求します。

本メソッドは送信の後、引き続き S5F4 応答メッセージを待機し、受信が終了したら、上の【戻り値】に示した値を返却します。ei=0 の場合は、ackc5 に ACK の値が渡されます。

応答メッセージを受信するまでプログラムはブロック（待ち）状態になります。

送信エラー、受信エラーを検出した場合は、ei ＜ 0 の値を返却します。(T3 ﾀｲﾑｱｳﾄも含む)

T3 タイムアウトが発生した場合、DSHDR2 HSMS ドライバーに設定されている T3 プロトコルタイムアウト時間だけブロックされることになります。

８．４　DshS5F5Send クラス

S5F5 メッセージを送信し、S5F6 応答メッセージを受信するためのクラスです。

８．４．１　コンストラクタ

オーバーロードの一覧を示します。

	名前	説明
1	public DshS5F5Send()	装置 ID=0 のｲﾝｽﾀﾝｽを生成します。
2	public DshS5F5Send(int eqid)	装置 ID を指定してｲﾝｽﾀﾝｽを生成します。

８．４．２　プロパティ

プロパティ一覧表に示します。

	名前	説明
1	public int eqid	装置 ID です。ﾃﾞﾌｫﾙﾄ値=0 です。
2	public int count	3 の alid_list 配列に保存されている ALID の数です。
3	public uint[] alid_list	ALID の配列ﾘｽﾄです。 S5F5 に設定するアラーム ID 保存用です。

８．４．３　メソッド

本クラスのメソッドは次の通りです。

	名前	説明
1	public void clear()	alid_list を空にし、count = 0 にします。
2	public void add_alid()	alid_list に ALID を 1 個追加します。
3	public int send_s5f5() public int send()	S5F5 ﾒｯｾｰｼﾞを送信します。
4	public int send_wait()	S5F5 ﾒｯｾｰｼﾞを送信し、S5F6 を受信します。 ﾌﾟﾛｸﾞﾗﾑは応答受信までﾌﾞﾛｯｸされます。
5	public void Dispose()	ｸﾗｽ内部で使用された資源(ﾒﾓﾘ)ｼｽﾃﾑに返却します。 他のｸﾗｽをﾌﾟﾛﾊﾟﾃｨにしている場合、そのｸﾗｽの Dispose()も行います。 3.1.3.5 と同様です。そちらを参照ください。

８．４．３．１　clear()

アラーム ID 保存の alid_list を空にします。

【構文】
public void clear()

【引数】
なし

【戻り値】
なし。

【説明】
アラーム ID 保存配列リスト alid_list を空にし、count = 0 にします。

８．４．３．２　add_alid()

アラーム ID を保存する alid_list に ID を 1 個追加します。

【構文】
public void add_alid(uint alid)

【引数】
alid
　　追加するアラーム ID です。

【戻り値】
なし。

【説明】
アラーム ID 保存配列リスト alid_list に ID を 1 個追加します。
追加した後、count + 1 します。

８．４．３．３　send_s5f5(), send()

S5F5 メッセージの送信要求をします。

【構文】

```
public int send_s5f5(ref DshAlarmList rsp_info,
                              DshCallback.callback_s5f5 callback, uint upara)
public int send(ref DshAlarmList rsp_info,
                              DshCallback.callback_s5f5 callback, uint upara)
```

【引数】

rsp_info

S5F6 応答メッセージに含まれる名前情報を保存するためのクラスのインスタンスです。
コールバック関数の引数になります。

callback

S5F5 送信後、2 次メッセージ受信した時に呼び出される callback 関数です。

upara

ユーザパラメータ情報です。コールバックされる際に callback 関数の引数として渡されます。
要求とコールバック間のタグ情報として使用できます。

【戻り値】

返却値	意　味
0	要求が受け入れられた。
(-1)	要求が受け入れられなかった。

【説明】

当該インスタンスの alid_list に保存されている count 分のアラーム ID を含む S5F5 メッセージを生成し、それを相手装置に送信するようエンジンに要求します。
要求がエンジンによって受け入れられたときは、0 を返却します。受け入れられなかった場合は(-1)を返却します。
送信後、受信した応答メッセージを rsp_info のインスタンスに設定し、callback 関数を呼び出します。
callback 関数からは 0 を返却してください。

【callback の構文】

```
public delegate int callback_s5f5(
    int eqid,                       // 装置 ID
    int end_status,                 // 終了状態ｺｰﾄﾞ
    ref DshAlarmList rsp_info,      // S5F6 に含まれるアラーム情報　 Vol-8.3 参照
    uint upara                      // ﾕｰｻﾞﾊﾟﾗﾒｰﾀ(送信要求ﾒｿｯﾄﾞで指定された upara)
)
```

end_status に返却される値は、=0 の場合は正常終了を意味します。

end_status	意　味
0	送受信が正常に完了した。
-14	T3 ﾀｲﾑｱｳﾄを検出し、応答ﾒｯｾｰｼﾞを受信できなかった。
(-1)	送信できなかった。

８．４．３．４　send_wait()

S5F5 メッセージの送信要求をし,引き続き応答メッセージも受信します。

【構文】

public int send_wait((ref DshAlarmList rsp_info)

【引数】

rsp_info

S5F6 応答メッセージに含まれる名前情報を保存するためのクラスのインスタンスです。

【戻り値】

返却値	意　味
0	正常に送受信した。
＜ 0	送受信ｴﾗｰが発生した。（送信ｴﾗｰ or T3 ﾀｲﾑｱｳﾄなど）

【説明】

当該インスタンスの alid_list に保存されている count 分のアラーム ID を含む S5F5 メッセージを生成し、それを相手装置に送信するようエンジンに要求します。

本メソッドは送信の後、引き続き S5F6 応答メッセージを待機し、受信が終了したら、上の【戻り値】に示した値を返却します。ei=0 の場合は、受信した応答メッセージを rsp_info のインスタンスに渡されます。

応答メッセージを受信するまでプログラムはブロック（待ち）状態になります。

送信エラー、受信エラーを検出した場合は、ei ＜ 0 の値を返却します。(T3 ﾀｲﾑｱｳﾄも含む)

T3 タイムアウトが発生した場合、DSHDR2 HSMS ドライバーに設定されている T3 プロトコルタイムアウト時間だけブロックされることになります。

９．プロセスプログラム関連メッセージの送受信

以下のメッセージとクラスがあります。

1	S7F1,S7F2	DshS7F1Send	S7F1 送信　　Process Program Load Inquire
		DshS7F2Response	S7F2 応答
2	S7F3,S7F4	DshS7F3Send	S7F3 送信　　Process Program Send
		DshS7F4Response	S7F4 応答
3	S7F5,S7F6	DshS7F5Send	S7F5 送信　　Process Program Data
		－	(S7F6 はｴﾝｼﾞﾝが自動応答)
4	S7F17,S7F18	DshS7F17Send	S7F17 送信　Delete Process Program Send
		－	(S7F18 はｴﾝｼﾞﾝが自動応答)
5	S7F19,S7F20	DshS7F19Send	S7F19 送信　Current EPPD Request
		－	(S7F20 はｴﾝｼﾞﾝが自動応答)

９．１　DshS7F1Send クラス

S7F1 メッセージを送信し、S7F2 応答メッセージを受信するためのクラスです。
プロセスプログラム（PP）のロード問合せ情報を送信します。

９．１．１　コンストラクタ

オーバーロードの一覧を示します。

	名前	説明
1	public DshS7F1Send()	装置 ID=0 の空のｲﾝｽﾀﾝｽを生成します。
2	public DshS7F1Send(int eqid)	装置 ID を指定してｲﾝｽﾀﾝｽを生成します。

９．１．２　プロパティ

プロパティ一覧表に示します。

	名前	説明
1	public int eqid	装置 ID です。ﾃﾞﾌｫﾙﾄ値=0 です。

９．１．３　メソッド

本クラスのメソッドは次の通りです。

	名前	説明
1	public int send_s7f1() public int send ()	S7F1 ﾒｯｾｰｼﾞを送信します。
2	public int send_wait()	S7F1 ﾒｯｾｰｼﾞを送信し、S7F2 を受信します。 ﾌﾟﾛｸﾞﾗﾑは応答受信までﾌﾞﾛｯｸされます。
3	public void Dispose()	ｸﾗｽ内部で使用された資源(ﾒﾓﾘ)ｼｽﾃﾑに返却します。 他のｸﾗｽをﾌﾟﾛﾊﾟﾃｨにしている場合、そのｸﾗｽの Dispose()も行います。 3.1.3.5 と同様です。そちらを参照ください。

９．１．３．１　send_s7f1(), send()

S7F1 メッセージの送信要求をします。

【構文】

```
public int send_S7F1(string ppid, uint length,
                     ref int ppgnt, DshCallback.callback_s7f1 callback, uint upara)
public int send(string ppid, uint length,
                     ref int ppgnt, DshCallback.callback_s7f1 callback, uint upara)
```

【引数】

ppid
　　プロセスプログラム ID です。
length
　　情報ロードに必要なメモリです。
ppgnt
　　S7F2 応答メッセージに含まれる ACK を保存します。
　　コールバック関数の引数になります。
callback
　　S7F1 送信後、2 次メッセージ受信した時に呼び出される callback 関数です。
upara
　　ユーザパラメータ情報です。コールバックされる際に callback 関数の引数として渡されます。
　　要求とコールバック間のタグ情報として使用できます。

【戻り値】

返却値	意　味
0	要求が受け入れられた。
(-1)	要求が受け入れられなかった。

【説明】

引数に指定された ppid, length 情報から S7F1 メッセージを生成し、それを相手装置に送信するようエンジンに要求します。
要求がエンジンによって受け入れられたときは、0 を返却します。受け入れられなかった場合は(-1)を返却します。
送信後、受信した応答メッセージの ppgnt を引数にして、callback 関数を呼び出します。
callback 関数からは 0 を返却してください。

【callback の構文】

```
public delegate int callback_s7f1(
    int eqid,                       // 装置 ID
    int end_status,                 // 終了状態ｺｰﾄﾞ
    ref int ppgnt,                  // S7F2 に含まれる ACK です。
    uint upara                      // ﾕｰｻﾞﾊﾟﾗﾒｰﾀ(送信要求ﾒｿｯﾄﾞで指定された upara)
)
```

end_status に返却される値は、=0 の場合は正常終了を意味します。

end_status	意　味
0	送受信が正常に完了した。
-14	T3 ﾀｲﾑｱｳﾄを検出し、応答ﾒｯｾｰｼﾞを受信できなかった。
(-1)	送信できなかった。

９．１．３．２　send_wait()

S7F1 メッセージの送信要求をし,引き続き応答メッセージも受信します。

【構文】

public int send_wait(string ppid, uint length, ref int ppgnt)

【引数】

ppid

　　プロセスプログラム ID です。

length

　　情報ロードに必要なメモリです。

ppgnt

　　S7F2 応答メッセージに含まれる ACK を保存します。

【戻り値】

返却値	意　味
0	正常に送受信した。
＜ 0	送受信ｴﾗｰが発生した。(送信ｴﾗｰ or T3 ﾀｲﾑｱｳﾄなど)

【説明】

当該インスタンスの alid_list に保存されている count 分のアラーム ID を含む S7F1 メッセージを生成し、それを相手装置に送信するようエンジンに要求します。

本メソッドは送信の後、引き続き S7F2 応答メッセージを待機し、受信が終了したら、上の【戻り値】に示した値を返却します。ei=0 の場合は、ACK を ppgnt に渡されます。

応答メッセージを受信するまでプログラムはブロック（待ち）状態になります。

送信エラー、受信エラーを検出した場合は、ei ＜ 0 の値を返却します。(T3 ﾀｲﾑｱｳﾄも含む)

T3 タイムアウトが発生した場合、DSHDR2 HSMS ドライバーに設定されている T3 プロトコルタイムアウト時間だけブロックされることになります。

９．２　DshS7F2Response クラス

S7F1 メッセージを受信した後、S7F2 応答メッセージを送信するためのクラスです。

９．２．１　コンストラクタ

オーバーロードの一覧を示します。

	名前	説明
1	public DshS7F2Response()	装置 ID=0 のｲﾝｽﾀﾝｽを生成します。
2	public DshS7F2Response(int eqid)	装置 ID を指定してｲﾝｽﾀﾝｽを生成します。

９．２．２　プロパティ

プロパティ一覧表に示します。

	名前	説明
1	public int eqid	装置 ID です。ﾃﾞﾌｫﾙﾄ値=0 です。

９．２．３　メソッド

本クラスのメソッドは次の通りです。

	名前	説明
1	public int response()	S7F2 応答ﾒｯｾｰｼﾞを送信します。
2	public void Dispose()	ｸﾗｽ内部で使用された資源(ﾒﾓﾘ)ｼｽﾃﾑに返却します。 他のｸﾗｽをﾌﾟﾛﾊﾟﾃｨにしている場合、そのｸﾗｽの Dispose()も行います。 3.1.3.5 と同様です。そちらを参照ください。

９．２．３．１　response()

S7F2 メッセージの応答送信要求をします。

【構文】

public int response(uint trid, int ppgnt)

【引数】

trid

エンジンからポーリング時に与えられたトランザクション ID です。
処理した１次メッセージとの対応を取ります。

ppgnt

S7F2 メッセージのための ACK です。

【戻り値】

返却値	意　味
0	要求が受け入れられた。
(-1)	要求が受け入れられなかった。(ﾄﾗﾝｻﾞｸｼｮﾝ ID が正しくなかった。)

【説明】

受信した S7F1 に対する S7F2 メッセージを送信します。

応答メッセージに含める情報は ppgnt です。
trid は、この応答メッセージに対する S7F1 受信時にエンジンから与えられたトランザクション ID です。
trid がエンジンの管理の中に見つからなかった場合には、(-1)を返却します。
要求が受け入れられたら 0 を返却します。この後、応答メッセージ送信結果は特に通知されません。

９．３　DshS7F3Send クラス

S7F3 メッセージを送信し、S7F4 応答メッセージを受信するためのクラスです。
プロセスプログラム（PP）情報を送信します。

９．３．１　コンストラクタ

オーバーロードの一覧を示します。

	名前	説明
1	public DshS7F3Send()	装置 ID=0 の空のｲﾝｽﾀﾝｽを生成します。
2	public DshS7F3Send(ref DshPP pclass)	装置 ID=0 のｲﾝｽﾀﾝｽをﾌﾟﾛｾｽﾌﾟﾛｸﾞﾗﾑ情報の DshPP ｸﾗｽのｲﾝｽﾀﾝｽを指定して生成します。
3	public DshS7F3Send(int eqid)	装置 ID を指定してｲﾝｽﾀﾝｽを生成します。
4	public DshS7F3Send(int eqid, ref DshPP pclass)	装置 ID とﾌﾟﾛｾｽﾌﾟﾛｸﾞﾗﾑ情報の DshPP ｸﾗｽのｲﾝｽﾀﾝｽを指定して生成します。

９．３．２　プロパティ

プロパティ一覧表に示します。

	名前	説明
1	public int eqid	装置 ID です。ﾃﾞﾌｫﾙﾄ値=0 です。
2	public DshPP pp_class	ﾌﾟﾛｾｽﾌﾟﾛｸﾞﾗﾑ情報(ppid, ppbody)の保存ｸﾗｽ DshPP のｲﾝｽﾀﾝｽです。

９．３．３　メソッド

本クラスのメソッドは次の通りです。

	名前	説明
1	public void set_ppinfo()	引数に与えられたﾌﾟﾛｾｽﾌﾟﾛｸﾞﾗﾑ情報を DshPP ｸﾗｽのｲﾝｽﾀﾝｽを使って設定します。
2	public int send_s7f3() public int send()	S7F3 ﾒｯｾｰｼﾞを送信します。
3	public int send_wait()	S7F3 ﾒｯｾｰｼﾞを送信し、S7F4 を受信します。 ﾌﾟﾛｸﾞﾗﾑは応答受信までﾌﾞﾛｯｸされます。
4	public void Dispose()	ｸﾗｽ内部で使用された資源(ﾒﾓﾘ)ｼｽﾃﾑに返却します。 他のｸﾗｽをﾌﾟﾛﾊﾟﾃｨにしている場合、そのｸﾗｽの Dispose()も行います。 3.1.3.5 と同様です。そちらを参照ください。

９．３．３．１　set_ppinfo()

ユーザが準備した DshTrace のクラスのインスタンスの内容を当該インスタンスの info に設定します。

【構文】

public void set_ppinfo(ref DshPP info)

【引数】

info

プロパティ pp_class に設定したいプロセスプログラム情報が保存されている DshPP のインスタンスです。
ユーザが準備します。

【戻り値】

なし。

【説明】

info の引数のプロセスプログラムの内容を当該インスタンスの pp_class にコピーします。

９．３．３．２　send_s7f3(), send()

S7F3 メッセージの送信要求をします。

【構文】

```
public int send_S7F3( ref int ackc7, DshCallback.callback_s7f3 callback, uint upara)
public int send( ref int ackc7, DshCallback.callback_s7f3 callback, uint upara)
```

【引数】

ackc7

S7F4 応答メッセージに含まれる ACK を保存します。
コールバック関数の引数になります。

callback

S7F3 送信後、2 次メッセージ受信した時に呼び出される callback 関数です。

upara

ユーザパラメータ情報です。コールバックされる際に callback 関数の引数として渡されます。
要求とコールバック間のタグ情報として使用できます。

【戻り値】

返却値	意　味
0	要求が受け入れられた。
(-1)	要求が受け入れられなかった。

【説明】

当該インスタンスの pp_class に保存されているプロセスプログラム情報から S7F3 メッセージを生成し、それを相手装置に送信するようエンジンに要求します。
要求がエンジンによって受け入れられたときは、0 を返却します。受け入れられなかった場合は(-1)を返却します。
送信後、受信した応答メッセージの ackc7 を引数にして、callback 関数を呼び出します。
callback 関数からは 0 を返却してください。

【callback の構文】

```
public delegate int callback_s7f3(
    int eqid,                       // 装置 ID
    int end_status,                 // 終了状態ｺｰﾄﾞ
    ref int ackc7,                  // S7F4 に含まれる ACK です。
    uint upara                      // ﾕｰｻﾞﾊﾟﾗﾒｰﾀ(送信要求ﾒｿｯﾄﾞで指定された upara)
)
```

end_status に返却される値は、=0 の場合は正常終了を意味します。

end_status	意　味
0	送受信が正常に完了した。
-14	T3 ﾀｲﾑｱｳﾄを検出し、応答ﾒｯｾｰｼﾞを受信できなかった。
(-1)	送信できなかった。

９．３．３．３　send_wait()

S7F3 メッセージの送信要求をし,引き続き応答メッセージも受信します。

【構文】

public int send_wait(ref int ackc7)

【引数】

ackc7

S7F4 応答メッセージに含まれる ACK を保存します。

【戻り値】

返却値	意　味
0	正常に送受信した。
＜ 0	送受信ｴﾗｰが発生した。（送信ｴﾗｰ or T3 ﾀｲﾑｱｳﾄなど）

【説明】

当該インスタンスの pp_class に保存されているプロセスプログラム情報から S7F3 メッセージを生成し、それを相手装置に送信するようエンジンに要求します。

本メソッドは送信の後、引き続き S7F4 応答メッセージを待機し、受信が終了したら、上の【戻り値】に示した値を返却します。ei=0 の場合は、ACK を ackc7 に渡されます。

応答メッセージを受信するまでプログラムはブロック（待ち）状態になります。

送信エラー、受信エラーを検出した場合は、ei ＜ 0 の値を返却します。(T3 ﾀｲﾑｱｳﾄも含む)

T3 タイムアウトが発生した場合、DSHDR2 HSMS ドライバーに設定されている T3 プロトコルタイムアウト時間だけブロックされることになります。

９．４　DshS7F4Response クラス

S7F3 メッセージを受信した後、S7F4 応答メッセージを送信するためのクラスです。

９．４．１　コンストラクタ

オーバーロードの一覧を示します。

	名前	説明
1	public DshS7F4Response()	装置 ID=0 のｲﾝｽﾀﾝｽを生成します。
2	public DshS7F4Response(int eqid)	装置 ID を指定してｲﾝｽﾀﾝｽを生成します。

９．４．２　プロパティ

プロパティ一覧表に示します。

	名前	説明
1	public int eqid	装置 ID です。ﾃﾞﾌｫﾙﾄ値=0 です。

９．４．３　メソッド

本クラスのメソッドは次の通りです。

	名前	説明
1	public int response()	S7F4 応答ﾒｯｾｰｼﾞを送信します。
2	public void Dispose()	ｸﾗｽ内部で使用された資源(ﾒﾓﾘ)ｼｽﾃﾑに返却します。 他のｸﾗｽをﾌﾟﾛﾊﾟﾃｨにしている場合、そのｸﾗｽの Dispose()も行います。 3.1.3.5 と同様です。そちらを参照ください。

９．４．３．１　response()

S7F4 メッセージの応答送信要求をします。

【構文】

public int response(uint trid, int ackc7)

【引数】

trid

エンジンからポーリング時に与えられたトランザクション ID です。
処理した１次メッセージとの対応を取ります。

ackc7

S7F4 メッセージに設定する応答 ACK です。

【戻り値】

返却値	意　味
0	要求が受け入れられた。
(-1)	要求が受け入れられなかった。(ﾄﾗﾝｻﾞｸｼｮﾝ ID が正しくなかった。)

【説明】

受信した S7F3 に対する S7F4 メッセージを送信します。

応答メッセージに含める情報は ackc7 です。
trid は、この応答メッセージに対する S7F3 受信時にエンジンから与えられたトランザクション ID です。
trid がエンジンの管理の中に見つからなかった場合には、(-1)を返却します。
要求が受け入れられたら 0 を返却します。この後、応答メッセージ送信結果は特に通知されません。

９．５　DshS7F5Send クラス

S7F5 メッセージを送信し、S7F6 応答メッセージを受信するためのクラスです。
相手装置にプロセスプログラム（PP）情報の応答を要求します。

９．５．１　コンストラクタ

オーバーロードの一覧を示します。

	名前	説明
1	public DshS7F5Send()	装置 ID=0 のｲﾝｽﾀﾝｽを生成します。
2	public DshS7F5Send(uint ppid)	ﾌﾟﾛｾｽﾌﾟﾛｸﾞﾗﾑ ID を指定して装置 ID=0 のｲﾝｽﾀﾝｽを生成します。
3	public DshS7F5Send(int eqid)	装置 ID を指定してｲﾝｽﾀﾝｽを生成します。
4	public DshS7F5Send(int eqid, uint ppid)	装置 ID とﾌﾟﾛｾｽﾌﾟﾛｸﾞﾗﾑ ID を指定してｲﾝｽﾀﾝｽを生成します。

９．５．２　プロパティ

プロパティ一覧表に示します。

	名前	説明
1	public int eqid	装置 ID です。ﾃﾞﾌｫﾙﾄ値=0 です。
2	public uint ppid	送信するﾌﾟﾛｾｽﾌﾟﾛｸﾞﾗﾑ ID(PPID)です。

９．５．３　メソッド

本クラスのメソッドは次の通りです。

	名前	説明
1	public void set_ppid()	ﾌﾟﾛｾｽﾌﾟﾛｸﾞﾗﾑ ID(PPID)を設定します。
2	public int send_s7f5() public int send()	S7F5 ﾒｯｾｰｼﾞを送信します。
3	public int send_wait()	S7F5 ﾒｯｾｰｼﾞを送信し、S7F6 を受信します。 ﾌﾟﾛｸﾞﾗﾑは応答受信までﾌﾞﾛｯｸされます。
4	public void Dispose()	ｸﾗｽ内部で使用された資源(ﾒﾓﾘ)ｼｽﾃﾑに返却します。 他のｸﾗｽをﾌﾟﾛﾊﾟﾃｨにしている場合、そのｸﾗｽの Dispose()も行います。 3.1.3.5 と同様です。そちらを参照ください。

９．５．３．１　set_ppid()

プロセスプログラム ID を設定します。

【構文】

public void set_ppid(uint ppid)

【引数】

ppid

プロセスプログラム ID (PPID) です。
このプロセスプログラム ID の情報を S7F5 で要求します。

【戻り値】

なし。

【説明】

S7F5 に設定したいプロセスプログラム ID を設定します。

９．５．３．２　send_s7f5(), send()

S7F5 メッセージの送信要求をします。

【構文】

```
public int send_S7F5(uint ppid, ref DshPP rsp_info,
                              DshCallback.callback_s7f5 callback, uint upara)
public int send(uint ppid, ref DshPP rsp_info,
                              DshCallback.callback_s7f5 callback, uint upara)
public int send_S7F5( ref DshPP rsp_info, DshCallback.callback_s7f5 callback, uint upara)
public int send( ref DshPP rsp_info, DshCallback.callback_s7f5 callback, uint upara)
```

【引数】

ppid
　　プロセスプログラム ID です。

rsp_info
　　S7F6 応答メッセージに含まれるプロセスプログラム情報保存用です。コールバック関数の引数になります。

callback
　　S7F5 送信後、2 次メッセージ受信した時に呼び出される callback 関数です。

upara
　　ユーザパラメータ情報です。コールバックされる際に callback 関数の引数として渡されます。要求とコールバック間のタグ情報として使用できます。

【戻り値】

返却値	意　味
0	要求が受け入れられた。
(-1)	要求が受け入れられなかった。

【説明】

指定 ppid の S7F5 を送信し、プロセスプログラム情報の応答を要求します。
要求がエンジンによって受け入れられたときは、0 を返却します。受け入れられなかった場合は(-1)を返却します。
送信後、受信した応答メッセージに含まれるプロセスプログラム情報を、格納用インスタンス rsp_info に保存し、それを引数にして callback 関数を呼び出します。
callback 関数からは 0 を返却してください。

【callback の構文】

```
public delegate int callback_s7f5(
    int eqid,                       // 装置 ID
    int end_status,                 // 終了状態ｺｰﾄﾞ
    ref DshPP rsp_info,             // S7F6 に含まれるﾚﾎﾟｰﾄ情報です。 Vol-1 11.1 参照
    uint upara                      // ﾕｰｻﾞﾊﾟﾗﾒｰﾀ(送信要求ﾒｿｯﾄﾞで指定された upara)
)
```

end_status に返却される値は、=0 の場合は正常終了を意味します。

end_status	意　味
0	送受信が正常に完了した。
-14	T3 ﾀｲﾑｱｳﾄを検出し、応答ﾒｯｾｰｼﾞを受信できなかった。
(-1)	送信できなかった。

９．５．３．３　send_wait()

S7F5 メッセージの送信要求をし,引き続き応答メッセージも受信します。

【構文】

```
public int send_wait(string ppid, ref DshPP rsp_info)
public int send_wait(ref DshPP rsp_info)
```

【引数】

ppid

　　プロセスプログラム ID です。

ackc7

　　S7F6 応答メッセージに含まれる ACK を保存します。

【戻り値】

返却値	意　味
0	正常に送受信した。
＜ 0	送受信ｴﾗｰが発生した。(送信ｴﾗｰ or T3 ﾀｲﾑｱｳﾄなど)

【説明】

指定された ppid のプロセスプログラム情報の応答を S7F5 の送信によって相手装置に要求します。

本メソッドは送信の後、引き続き S7F6 応答メッセージを待機し、受信が終了したら、上の【戻り値】に示した値を返却します。ei=0 の場合は、S7F6 に含まれるプロセスプログラム情報を rsp_info に設定し渡します。

応答メッセージを受信するまでプログラムはブロック（待ち）状態になります。

送信エラー、受信エラーを検出した場合は、ei ＜ 0 の値を返却します。(T3 ﾀｲﾑｱｳﾄも含む)

T3 タイムアウトが発生した場合、DSHDR2 HSMS ドライバーに設定されている T3 プロトコルタイムアウト時間だけブロックされることになります。

９．６　DshS7F17Send クラス

S7F17 メッセージを送信し、S7F18 応答メッセージを受信するためのクラスです。
S7F17 は相手装置にプロセスプログラム ID(PPID)を指定してそれらの削除を要求します。
そして応答メッセージを受信します。

９．６．１　コンストラクタ

オーバーロードの一覧を示します。

	名前	説明
1	public DshS7F17Send()	装置 ID=0 のｲﾝｽﾀﾝｽを生成します。
2	public DshS7F17Send(int eqid)	装置 ID を指定してｲﾝｽﾀﾝｽを生成します。

９．６．２　プロパティ

プロパティ一覧表に示します。

	名前	説明
1	public int eqid	装置 ID です。ﾃﾞﾌｫﾙﾄ値=0 です。
2	public int count	削除したい PPID の数です。 list 配列ﾘｽﾄに保存されます。
3	public string[] list	PPID を保存するための配列です。

９．６．３　メソッド

本クラスのメソッドは次の通りです。

	名前	説明
1	public void clear()	list 配列を空にします。
2	public void add_ppid()	ﾌﾟﾛｾｽﾌﾟﾛｸﾞﾗﾑ ID(PPID)を list に追加します。
3	public int send_s7f17() public int send()	S7F17 ﾒｯｾｰｼﾞを送信します。
4	public int send_wait()	S7F17 ﾒｯｾｰｼﾞを送信し、S7F18 を受信します。 ﾌﾟﾛｸﾞﾗﾑは応答受信までﾌﾞﾛｯｸされます。
5	public void Dispose()	ｸﾗｽ内部で使用された資源(ﾒﾓﾘ)ｼｽﾃﾑに返却します。 他のｸﾗｽをﾌﾟﾛﾊﾟﾃｨにしている場合、そのｸﾗｽの Dispose()も行います。 3.1.3.5 と同様です。そちらを参照ください。

９．６．３．１　clear()

レポートリンク情報が保存されている list 配列を空にします。

【構文】

public void clear()

【引数】

なし

【戻り値】

なし。

【説明】

PPID が保存されている list 配列を空にし、count = 0 にします。

９．６．３．２　add_ppid()

プロセスプログラム ID を設定します。

【構文】

public void add_ppid(uint ppid)

【引数】

ppid

プロセスプログラム ID (PPID) です。

【戻り値】
なし。

【説明】
S7F17 に設定したいプロセスプログラム ID を list 配列に追加します。そして、count + 1 する。

９．６．３．３　send_s7f17(), send()

S7F17 メッセージの送信要求をします。

【構文】

```
public int send_S7F17( ref int ackc7, DshCallback.callback_s7f17 callback, uint upara )
public int send( ref int ackc7, DshCallback.callback_s7f17 callback, uint upara )
```

【引数】

ackc7

S7F18 の ACK の保存をします。callback 時に引数として渡されます。

callback

S7F17 送信後、2 次メッセージ受信した時に呼び出される callback 関数です。

upara

ユーザパラメータ情報です。コールバックされる際に callback 関数の引数として渡されます。
要求とコールバック間のタグ情報として使用できます。

【戻り値】

返却値	意　味
0	要求が受け入れられた。
(-1)	要求が受け入れられなかった。

【説明】

当該インスタンスの list 配列に含まれる PPID から S7F17 を生成し相手装置に送信要求します。
要求がエンジンによって受け入れられたときは、0 を返却します。受け入れられなかった場合は(-1)を返却します。
送信後、受信した応答メッセージに含まれる ACKC7 を ackc7 に設定し、それを引数にして callback 関数を呼び出します。
callback 関数からは 0 を返却してください。

【callback の構文】

```
public delegate int callback_s7f17(
    int eqid,                       // 装置 ID
    int end_status,                 // 終了状態ｺｰﾄﾞ
    ref int ackc7,                  // S7F18 に含まれる ACKC7 情報です。
    uint upara                      // ﾕｰｻﾞﾊﾟﾗﾒｰﾀ(送信要求ﾒｿｯﾄﾞで指定された upara)
)
```

end_status に返却される値は、=0 の場合は正常終了を意味します。

end_status	意　味
0	送受信が正常に完了した。
-14	T3 ﾀｲﾑｱｳﾄを検出し、応答ﾒｯｾｰｼﾞを受信できなかった。
(-1)	送信できなかった。

９．６．３．４　send_wait()

S7F17 メッセージの送信要求をし,引き続き応答メッセージも受信します。

【構文】

public int send_wait(ref int ackc7)

【引数】

ackc7

S7F18 応答メッセージに含まれる ACK を保存します。

【戻り値】

返却値	意　味
0	正常に送受信した。
＜ 0	送受信ｴﾗｰが発生した。（送信ｴﾗｰ or T3 ﾀｲﾑｱｳﾄなど）

【説明】

当該インスタンスの list 配列に含まれる PPID から S7F17 を生成し相手装置に送信要求します。

本メソッドは送信の後、引き続き S7F18 応答メッセージを待機し、受信が終了したら、上の【戻り値】に示した値を返却します。ei=0 の場合は、ACK を ackc7 に渡されます。

応答メッセージを受信するまでプログラムはブロック（待ち）状態になります。

送信エラー、受信エラーを検出した場合は、ei ＜ 0 の値を返却します。(T3 ﾀｲﾑｱｳﾄも含む)

T3 タイムアウトが発生した場合、DSHDR2 HSMS ドライバーに設定されている T3 プロトコルタイムアウト時間だけブロックされることになります。

９．７　DshS7F19Send クラス

S7F19 メッセージを送信し、S7F20 応答メッセージを受信するためのクラスです。
S7F19 は相手装置から全プロセスプログラム ID(PPID)の送信を要求します。

９．７．１　コンストラクタ

オーバーロードの一覧を示します。

	名前	説明
1	public DshS7F19Send()	装置 ID=0 のｲﾝｽﾀﾝｽを生成します。
2	public DshS7F19Send(int eqid)	装置 ID を指定してｲﾝｽﾀﾝｽを生成します。

９．７．２　プロパティ

プロパティ一覧表に示します。

	名前	説明
1	public int eqid	装置 ID です。ﾃﾞﾌｫﾙﾄ値=0 です。

９．７．３　メソッド

本クラスのメソッドは次の通りです。

	名前	説明
1	public int send_s7f19()	S7F19 ﾒｯｾｰｼﾞを送信します。
2	public int send_wait()	S7F19 ﾒｯｾｰｼﾞを送信し、S7F20 を受信します。 ﾌﾟﾛｸﾞﾗﾑは応答受信までﾌﾞﾛｯｸされます。
3	public void Dispose()	ｸﾗｽ内部で使用された資源(ﾒﾓﾘ)ｼｽﾃﾑに返却します。 他のｸﾗｽをﾌﾟﾛﾊﾟﾃｨにしている場合、そのｸﾗｽの Dispose()も行います。 3.1.3.5 と同様です。そちらを参照ください。

９．７．３．１　send_s7f19()

S7F19 メッセージの送信要求をします。

【構文】

public int send_s7f19(ref DshStrList id_list, DshCallback.callback_s7f19 callback, uint upara)

【引数】

id_list

S7F20 応答メッセージで与えられる全 PPID を保存するための DshStrList クラスのインスタンスです。callback 時に引数として渡されます。Vol-1 17-5 参照

callback

S7F19 送信後、2 次メッセージ受信した時に呼び出される callback 関数です。

upara

ユーザパラメータ情報です。コールバックされる際に callback 関数の引数として渡されます。要求とコールバック間のタグ情報として使用できます。

【戻り値】

返却値	意　味
0	要求が受け入れられた。
(-1)	要求が受け入れられなかった。

【説明】

S7F19 送信を要求します。

要求がエンジンによって受け入れられたときは、0 を返却します。受け入れられなかった場合は(-1)を返却します。

送信後、受信した応答メッセージに含まれる全 PPID を id_list に設定し、それを引数にして callback 関数を呼び出します。

callback 関数からは 0 を返却してください。

【callback の構文】

```
public delegate int callback_s7f19(
    int eqid,                       // 装置 ID
    int end_status,                 // 終了状態ｺｰﾄﾞ
    ref DshStrList id_list,         // S7F20 に含まれる PPID 情報です。
    uint upara                      // ﾕｰｻﾞﾊﾟﾗﾒｰﾀ(送信要求ﾒｿｯﾄﾞで指定された upara)
)
```

end_status に返却される値は、=0 の場合は正常終了を意味します。

end_status	意　味
0	送受信が正常に完了した。
-14	T3 ﾀｲﾑｱｳﾄを検出し、応答ﾒｯｾｰｼﾞを受信できなかった。
(-1)	送信できなかった。

９．７．３．２　send_wait()

S7F19 メッセージの送信要求をし,引き続き応答メッセージも受信します。

【構文】

public int send_wait(ref DshStrList id_list)

【引数】

id_list

S7F20 応答メッセージで与えられる全 PPID を保存するための DshStrList クラスのインスタンスです。callback 時に引数として渡されます。Vol-1 17-5 参照

【戻り値】

返却値	意　味
0	正常に送受信した。
＜ 0	送受信ｴﾗｰが発生した。(送信ｴﾗｰ or T3 ﾀｲﾑｱｳﾄなど)

【説明】

S7F19 送信を要求します。

本メソッドは送信の後、引き続き S7F20 応答メッセージを待機し、受信が終了したら、上の【戻り値】に示した値を返却します。ei=0 の場合は、受信した応答メッセージに含まれる全 PPID を id_list に設定し渡されます。

応答メッセージを受信するまでプログラムはブロック（待ち）状態になります。

送信エラー、受信エラーを検出した場合は、ei ＜ 0 の値を返却します。(T3 ﾀｲﾑｱｳﾄも含む)

T3 タイムアウトが発生した場合、DSHDR2 HSMS ドライバーに設定されている T3 プロトコルタイムアウト時間だけブロックされることになります。

10．レシピ関連メッセージの送受信

以下のメッセージとクラスがあります。

	メッセージ	クラス名	機能概略
1	S15F3,S15F4	DshS15F3Send	S15F3 送信　Recipe Name Space Action Request
		DshS15F4Response	S15F4 応答
2	S15F5,S15F6	DshS15F5Send	S15F5 送信　Recipe Namespace Rename Request
		DshS15F6Response	S15F6 応答
3	S15F7,S15F8	DshS15F7Send	S15F7 送信　Recipe Space Request
		－	(S15F8 はｴﾝｼﾞﾝが自動応答)
4	S15F9,S15F10	DshS15F9Send	S15F9 送信　Recipe Status Request
		－	(S15F10 はｴﾝｼﾞﾝが自動応答)
5	S15F13,S15F14	DshS15F13Send	S15F13 送信　Recipe Create Request
		DshS15F14Response	S15F14 応答
6	S15F17,S15F18	DshS15F17Send	S15F17 送信　Recipe Retrieve Request
		DshS15F18Response	S15F18 応答

10. 1 DshS15F3Send クラス

S15F3 メッセージを送信し、S15F4 応答メッセージを受信するためのクラスです。
レシピ名スペースのアクション要求を行います。

10. 1. 1 コンストラクタ

オーバーロードの一覧を示します。

	名前	説明
1	public DshS15F3Send()	装置 ID=0 の空のｲﾝｽﾀﾝｽを生成します。
2	public DshS15F3Send(int eqid)	装置 ID を指定してｲﾝｽﾀﾝｽを生成します。

10. 1. 2 プロパティ

プロパティ一覧表に示します。

	名前	説明
1	public int eqid	装置 ID です。ﾃﾞﾌｫﾙﾄ値=0 です。
2	public string rcpid	ﾚｼﾋﾟ ID です。
3	public int rmnscmd	ｱｸｼｮﾝｺｰﾄﾞです。

10．1．3　メソッド

本クラスのメソッドは次の通りです。

	名前	説明
1	public void set_info()	引数に与えられたﾚｼﾋﾟ ID とｱｸｼｮﾝｺｰﾄﾞを設定します。
2	public int send_s15f3() public int send()	S15F3 ﾒｯｾｰｼﾞを送信します。
3	public int send_wait()	S15F3 ﾒｯｾｰｼﾞを送信し、S15F4 を受信します。 ﾌﾟﾛｸﾞﾗﾑは応答受信までﾌﾞﾛｯｸされます。
4	public void Dispose()	ｸﾗｽ内部で使用された資源(ﾒﾓﾘ)ｼｽﾃﾑに返却します。 他のｸﾗｽをﾌﾟﾛﾊﾟﾃｨにしている場合、そのｸﾗｽの Dispose()も行います。 3.1.3.5 と同様です。そちらを参照ください。

10．1．3．1　set_info()

【構文】

public void set_info(string rcpid, int rmnscmd)

【引数】

rcpid

　　アクション要求対象レシピ ID です。

rmnscmd

　　アクションコードです。

【戻り値】

なし。

【説明】

レシピ ID とアクションコードを当該インスタンスの rcpid, rmnscmd に設定します。

10. 1. 3. 2 send_s15f3(), send()

S15F3 メッセージの送信要求をします。

【構文】

```
public int send_S15F3(string rcpid, int rmnscmd,
            ref DshS15Rsp rsp_info, DshCallback.callback_s15f3 callback, uint upara)
public int send(string rcpid, int rmnscmd,
            ref DshS15Rsp rsp_info, DshCallback.callback_s15f3 callback, uint upara)
public int send_S15F3(ref DshS15Rsp rsp_info, DshCallback.callback_s15f3 callback, uint upara)
public int send(ref DshS15Rsp rsp_info, DshCallback.callback_s15f3 callback, uint upara)
```

【引数】

rcpid
　　レシピ ID です。
rmnscmd
　　アクションコードです。
rsp_info
　　S15F4 応答メッセージに含まれる情報を保存します。コールバック関数の引数になります。
callback
　　S15F3 送信後、2 次メッセージ受信した時に呼び出される callback 関数です。
upara
　　ユーザパラメータ情報です。コールバックされる際に callback 関数の引数として渡されます。
　　要求とコールバック間のタグ情報として使用できます。

【戻り値】

返却値	意　味
0	要求が受け入れられた。
(-1)	要求が受け入れられなかった。

【説明】

当該インスタンスの rcpid, rmnscmd から S15F3 メッセージを生成し、それを相手装置に送信するようエンジンに要求します。
要求がエンジンによって受け入れられたときは、0 を返却します。受け入れられなかった場合は(-1)を返却します。
送信後、受信した応答メッセージ情報を中に保存した rsp_info を引数にして、callback 関数を呼び出します。
callback 関数からは 0 を返却してください。

【callback の構文】

```
public delegate int callback_s15f3(
    int eqid,                       // 装置 ID
    int end_status,                 // 終了状態ｺｰﾄﾞ
    ref DshS15Rsp rsp_info,         // S15F4 に含まれる応答です。
    uint upara                      // ﾕｰｻﾞﾊﾟﾗﾒｰﾀ(送信要求ﾒｿｯﾄﾞで指定された upara)
)
```

10. 1. 3. 3 send_wait()

S15F3 メッセージの送信要求をし,引き続き応答メッセージも受信します。

【構文】

```
public int send_wait(string rcpid, int rmnscmd, ref DshS15Rsp rsp_info)
public int send_wait(ref DshS15Rsp rsp_info)
```

【引数】

rcpid
　　レシピ ID です。
rmnscmd
　　アクションコードです。
rsp_info
　　S15F4 応答メッセージに含まれる情報を保存します。

【戻り値】

返却値	意　味
0	正常に送受信した。
< 0	送受信ｴﾗｰが発生した。(送信ｴﾗｰ or T3 ﾀｲﾑｱｳﾄなど)

【説明】

指定された rcpid, rmnscmd から S15F3 メッセージを生成し、それを相手装置に送信するようエンジンに要求します。

本メソッドは送信の後、引き続き S15F4 応答メッセージを待機し、受信が終了したら、上の【戻り値】に示した値を返却します。ei=0 の場合は、受信した応答メッセージに含まれる応答情報を rsp_info に設定し渡されます。

応答メッセージを受信するまでプログラムはブロック（待ち）状態になります。

送信エラー、受信エラーを検出した場合は、ei < 0 の値を返却します。(T3 ﾀｲﾑｱｳﾄも含む)

T3 タイムアウトが発生した場合、DSHDR2 HSMS ドライバーに設定されている T3 プロトコルタイムアウト時間だけブロックされることになります。

end_status に返却される値は、=0 の場合は正常終了を意味します。

end_status	意　味
0	送受信が正常に完了した。
-14	T3 ﾀｲﾑｱｳﾄを検出し、応答ﾒｯｾｰｼﾞを受信できなかった。
(-1)	送信できなかった。

10. 2 DshS15F4Response クラス

S15F3 メッセージを受信した後、S15F4 応答メッセージを送信するためのクラスです。

10. 2. 1 コンストラクタ

オーバーロードの一覧を示します。

	名前	説明
1	public DshS15F4Response()	装置 ID=0 のｲﾝｽﾀﾝｽを生成します。
2	public DshS15F4Response(int eqid)	装置 ID を指定してｲﾝｽﾀﾝｽを生成します。

10. 2. 2 プロパティ

プロパティ一覧表に示します。

	名前	説明
1	public int eqid	装置 ID です。ﾃﾞﾌｫﾙﾄ値=0 です。

10．２．３　メソッド

本クラスのメソッドは次の通りです。

	名前	説明
1	public int response()	S15F4 応答ﾒｯｾｰｼﾞを送信します。
2	public void Dispose()	ｸﾗｽ内部で使用された資源(ﾒﾓﾘ)ｼｽﾃﾑに返却します。 他のｸﾗｽをﾌﾟﾛﾊﾟﾃｨにしている場合、そのｸﾗｽの Dispose()も行います。 3.1.3.5 と同様です。そちらを参照ください。

10．２．３．１　response()

S15F4 メッセージの応答送信要求をします。

【構文】

public int response(uint trid, ref DshS15Rsp rsp_class)

【引数】

trid

エンジンからポーリング時に与えられたトランザクション ID です。
処理した１次メッセージとの対応を取ります。

rsp_class

S15F4 メッセージに含める応答情報のインスタンスです。
DshS15Rsp クラスについては Vol-1 の 12.5 を参照してください。

【戻り値】

返却値	意　味
0	要求が受け入れられた。
(-1)	要求が受け入れられなかった。(ﾄﾗﾝｻﾞｸｼｮﾝ ID が正しくなかった。)

【説明】

受信した S15F3 に対する S15F4 メッセージを送信します。
DshS15RSP クラスのインスタンス rsp_class 内の応答情報から S15F4 を生成します。
trid は、この応答メッセージに対する S15F3 受信時にエンジンから与えられたトランザクション ID です。
trid がエンジンの管理の中に見つからなかった場合には、(-1)を返却します。
要求が受け入れられたら 0 を返却します。この後、応答メッセージ送信結果は特に通知されません。

10. 3 DshS15F5Send クラス

S15F5 メッセージを送信し、S15F6 応答メッセージを受信するためのクラスです。
S15F5 は、レシピ名の新しい名前への変更要求を行います。

10. 3. 1 コンストラクタ

オーバーロードの一覧を示します。

	名前	説明
1	public DshS15F5Send()	装置 ID=0 の空のｲﾝｽﾀﾝｽを生成します。
2	public DshS15F5Send(int eqid)	装置 ID を指定してｲﾝｽﾀﾝｽを生成します。

10. 3. 2 プロパティ

プロパティ一覧表に示します。

	名前	説明
1	public int eqid	装置 ID です。ﾃﾞﾌｫﾙﾄ値=0 です。
2	public string rcpid	ﾚｼﾋﾟ ID です。(現ﾚｼﾋﾟ名)
3	public int new_rcpid	名前を変えたい新しいﾚｼﾋﾟ ID です。

10．３．３　メソッド

本クラスのメソッドは次の通りです。

	名前	説明
1	public void set_info()	引数に与えられたﾚｼﾋﾟ ID とｱｸｼｮﾝｺｰﾄﾞを設定します。
2	public int send_s15f5() public int send()	S15F5 ﾒｯｾｰｼﾞを送信します。
3	public int send_wait()	S15F5 ﾒｯｾｰｼﾞを送信し、S15F6 を受信します。 ﾌﾟﾛｸﾞﾗﾑは応答受信までﾌﾞﾛｯｸされます。
4	public void Dispose()	ｸﾗｽ内部で使用された資源(ﾒﾓﾘ)ｼｽﾃﾑに返却します。 他のｸﾗｽをﾌﾟﾛﾊﾟﾃｨにしている場合、そのｸﾗｽの Dispose()も行います。 3.1.3.5 と同様です。そちらを参照ください。

10．３．３．１　set_info()

【構文】

public void set_info(string rcpid, string new_rcp)

【引数】

rcpid

　　現レシピ ID（名前）です。

new_rcp

　　新しいレシピ ID（名前）です。

【戻り値】

なし。

【説明】

現レシピ ID と新レシピ名を rcpid, new_rcpid に設定します。

10．３．３．２　send_s15f5(), send()

S15F5 メッセージの送信要求をします。

【構文】

```
public int send_S15F5(string rcpid, string new_rcp,
                      ref DshS15Rsp rsp_info, DshCallback.callback_s15f5 callback, uint upara)
public int send(string rcpid, string new_rcp,
                      ref DshS15Rsp rsp_info, DshCallback.callback_s15f5 callback, uint upara)
public int send_S15F5(ref DshS15Rsp rsp_info, DshCallback.callback_s15f5 callback, uint upara)
public int send(ref DshS15Rsp rsp_info, DshCallback.callback_s15f5 callback, uint upara)
```

【引数】

rcpid
　　現レシピ ID です。
new_rcp
　　新レシピ ID です。
rsp_info
　　S15F6 応答メッセージに含まれる情報を保存します。コールバック関数の引数になります。
callback
　　S15F5 送信後、2 次メッセージ受信した時に呼び出される callback 関数です。
upara
　　ユーザパラメータ情報です。コールバックされる際に callback 関数の引数として渡されます。
　　要求とコールバック間のタグ情報として使用できます。

【戻り値】

返却値	意　味
0	要求が受け入れられた。
(-1)	要求が受け入れられなかった。

【説明】

当該インスタンスの rcpid と new_rcpid から S15F5 メッセージを生成し、それを相手装置に送信するようエンジンに要求します。
要求がエンジンによって受け入れられたときは、0 を返却します。受け入れられなかった場合は(-1)を返却します。
送信後、受信した応答メッセージ情報を中に保存した rsp_info を引数にして、callback 関数を呼び出します。
callback 関数からは 0 を返却してください。

【callback の構文】

```
public delegate int callback_s15f5(
    int eqid,                       // 装置 ID
    int end_status,                 // 終了状態ｺｰﾄﾞ
    ref DshS15Rsp rsp_info,         // S15F6 に含まれる応答です。
    uint upara                      // ﾕｰｻﾞﾊﾟﾗﾒｰﾀ(送信要求ﾒｿｯﾄﾞで指定された upara)
)
```

10．３．３．３　send_wait()

S15F5 メッセージの送信要求をし,引き続き応答メッセージも受信します。

【構文】

```
public int send_wait(string rcpid, string new_rcp, ref DshS15Rsp rsp_info)
public int send_wait(ref DshS15Rsp rsp_info)
```

【引数】

rcpid
　　レシピ ID です。
new_rcp
　　新レシピ ID です。
rsp_info
　　S15F6 応答メッセージに含まれる情報を保存します。

【戻り値】

返却値	意　味
0	正常に送受信した。
＜ 0	送受信ｴﾗｰが発生した。(送信ｴﾗｰ or T3 ﾀｲﾑｱｳﾄなど)

【説明】

指定された rcpid と new_rcpid から S15F5 メッセージを生成し、それを相手装置に送信するようエンジンに要求します。

本メソッドは送信の後、引き続き S15F6 応答メッセージを待機し、受信が終了したら、上の【戻り値】に示した値を返却します。ei=0 の場合は、受信した応答メッセージに含まれる応答情報を rsp_info に設定し渡されます。

応答メッセージを受信するまでプログラムはブロック（待ち）状態になります。

送信エラー、受信エラーを検出した場合は、ei ＜ 0 の値を返却します。(T3 ﾀｲﾑｱｳﾄも含む)

T3 タイムアウトが発生した場合、DSHDR2 HSMS ドライバーに設定されている T3 プロトコルタイムアウト時間だけブロックされることになります。

end_status に返却される値は、=0 の場合は正常終了を意味します。

end_status	意　味
0	送受信が正常に完了した。
-14	T3 ﾀｲﾑｱｳﾄを検出し、応答ﾒｯｾｰｼﾞを受信できなかった。
(-1)	送信できなかった。

10. 4 DshS15F6Response クラス

S15F5 メッセージを受信した後、S15F6 応答メッセージを送信するためのクラスです。

10. 4. 1 コンストラクタ

オーバーロードの一覧を示します。

	名前	説明
1	public DshS15F6Response()	装置 ID=0 のｲﾝｽﾀﾝｽを生成します。
2	public DshS15F6Response(int eqid)	装置 ID を指定してｲﾝｽﾀﾝｽを生成します。

10. 4. 2 プロパティ

プロパティ一覧表に示します。

	名前	説明
1	public int eqid	装置 ID です。ﾃﾞﾌｫﾙﾄ値=0 です。

10．４．３　メソッド

本クラスのメソッドは次の通りです。

	名前	説明
1	public int response()	S15F6 応答ﾒｯｾｰｼﾞを送信します。
2	public void Dispose()	ｸﾗｽ内部で使用された資源(ﾒﾓﾘ)ｼｽﾃﾑに返却します。 他のｸﾗｽをﾌﾟﾛﾊﾟﾃｨにしている場合、そのｸﾗｽの Dispose()も行います。 3.1.3.5 と同様です。そちらを参照ください。

10．４．３．１　response()

S15F6 メッセージの応答送信要求をします。

【構文】

public int response(uint trid, ref DshS15Rsp rsp_class)

【引数】

trid

エンジンからポーリング時に与えられたトランザクション ID です。
処理した１次メッセージとの対応を取ります。

rsp_class

S15F6 メッセージに含める応答情報のインスタンスです。
DshS15Rsp クラスについては Vol-1 の 12.5 を参照してください。

【戻り値】

返却値	意　味
0	要求が受け入れられた。
(-1)	要求が受け入れられなかった。(ﾄﾗﾝｻﾞｸｼｮﾝ ID が正しくなかった。)

【説明】

受信した S15F5 に対する S15F6 メッセージを送信します。
DshS15RSP クラスのインスタンス rsp_class 内の応答情報から S15F6 を生成します。
trid は、この応答メッセージに対する S15F5 受信時にエンジンから与えられたトランザクション ID です。
trid がエンジンの管理の中に見つからなかった場合には、(-1)を返却します。
要求が受け入れられたら 0 を返却します。この後、応答メッセージ送信結果は特に通知されません。

10. 5 DshS15F7Send クラス

S15F7 メッセージを送信し、S15F8 応答メッセージを受信するためのクラスです。
S15F7 は、指定されたオブジェクトスペックの記憶装置内での記憶容量を取得するメッセージです。

10. 5. 1 コンストラクタ

オーバーロードの一覧を示します。

	名前	説明
1	public DshS15F7Send()	装置 ID=0 の空のｲﾝｽﾀﾝｽを生成します。
2	public DshS15F7Send(int eqid)	装置 ID を指定してｲﾝｽﾀﾝｽを生成します。

10. 5. 2 プロパティ

プロパティ一覧表に示します。

	名前	説明
1	public int eqid	装置 ID です。ﾃﾞﾌｫﾙﾄ値=0 です。
2	public string objspec	ｵﾌﾞｼﾞｪｸﾄｽﾍﾟｯｸです。

10．５．３　メソッド

本クラスのメソッドは次の通りです。

	名前	説明
1	public void set_objspec()	ｵﾌﾞｼﾞｪｸﾄｽﾍﾟｯｸを設定します。
2	public int send_s15f7() public int send()	S15F7 ﾒｯｾｰｼﾞを送信します。
3	public int send_wait()	S15F7 ﾒｯｾｰｼﾞを送信し、S15F8 を受信します。 ﾌﾟﾛｸﾞﾗﾑは応答受信までﾌﾞﾛｯｸされます。
4	public void Dispose()	ｸﾗｽ内部で使用された資源(ﾒﾓﾘ)ｼｽﾃﾑに返却します。 他のｸﾗｽをﾌﾟﾛﾊﾟﾃｨにしている場合、そのｸﾗｽの Dispose()も行います。 3.1.3.5 と同様です。そちらを参照ください。

10．５．３．１　set_objspec()

【構文】

public void set_objspec(string objspec)

【引数】

objspec

オブジェクトスペックです。

【戻り値】

なし。

【説明】

オブジェクトスペックを設定します。

10．５．３．２　send_s15f7(), send()

S15F7 メッセージの送信要求をします。

【構文】

```
public int send_S15F7(string objspec,
                      ref DshS15Rsp rsp_info, DshCallback.callback_s15f7 callback, uint upara)
public int send(string objspec,
                      ref DshS15Rsp rsp_info, DshCallback.callback_s15f7 callback, uint upara)
public int send_S15F7(ref DshS15Rsp rsp_info, DshCallback.callback_s15f7 callback, uint upara)
public int send(ref DshS15Rsp rsp_info, DshCallback.callback_s15f7 callback, uint upara)
```

【引数】

objspec
　　オブジェクトスペックです。

rsp_info
　　S15F8 応答メッセージに含まれる情報を保存します。コールバック関数の引数になります。

callback
　　S15F7 送信後、2 次メッセージ受信した時に呼び出される callback 関数です。

upara
　　ユーザパラメータ情報です。コールバックされる際に callback 関数の引数として渡されます。
　　要求とコールバック間のタグ情報として使用できます。

【戻り値】

返却値	意　味
0	要求が受け入れられた。
(-1)	要求が受け入れられなかった。

【説明】

当該インスタンスの objspec から S15F7 メッセージを生成し、それを相手装置に送信するようエンジンに要求します。
要求がエンジンによって受け入れられたときは、0 を返却します。受け入れられなかった場合は(-1)を返却します。
送信後、受信した応答メッセージ情報を中に保存した rsp_info を引数にして、callback 関数を呼び出します。
rsp_info の中の rmspace の値が記憶容量になります。
callback 関数からは 0 を返却してください。

【callback の構文】

```
public delegate int callback_s15f7(
    int eqid,                       // 装置 ID
    int end_status,                 // 終了状態ｺｰﾄﾞ
    ref DshS15Rsp rsp_info,         // S15F8 に含まれる応答です。
    uint upara                      // ﾕｰｻﾞﾊﾟﾗﾒｰﾀ(送信要求ﾒｿｯﾄﾞで指定された upara)
)
```

10．５．３．３　send_wait()

S15F7 メッセージの送信要求をし,引き続き応答メッセージも受信します。

【構文】

public int send_wait(string objspec, ref DshS15Rsp rsp_info)
public int send_wait(ref DshS15Rsp rsp_info)

【引数】

objspec
　　オブジェクトスペックです。
rsp_info
　　S15F8 応答メッセージに含まれる情報を保存します。

【戻り値】

返却値	意　味
0	正常に送受信した。
＜ 0	送受信ｴﾗｰが発生した。(送信ｴﾗｰ or T3 ﾀｲﾑｱｳﾄなど)

【説明】

指定された objspec から S15F7 メッセージを生成し、それを相手装置に送信するようエンジンに要求します。

本メソッドは送信の後、引き続き S15F8 応答メッセージを待機し、受信が終了したら、上の【戻り値】に示した値を返却します。ei=0 の場合は、受信した応答メッセージに含まれる応答情報を rsp_info に設定し渡されます。

応答メッセージを受信するまでプログラムはブロック（待ち）状態になります。

送信エラー、受信エラーを検出した場合は、ei ＜ 0 の値を返却します。(T3 ﾀｲﾑｱｳﾄも含む)

T3 タイムアウトが発生した場合、DSHDR2 HSMS ドライバーに設定されている T3 プロトコルタイムアウト時間だけブロックされることになります。

end_status に返却される値は、=0 の場合は正常終了を意味します。

end_status	意　味
0	送受信が正常に完了した。
-14	T3 ﾀｲﾑｱｳﾄを検出し、応答ﾒｯｾｰｼﾞを受信できなかった。
(-1)	送信できなかった。

10. 6 DshS15F9Send クラス

S15F9 メッセージを送信し、S15F10 応答メッセージを受信するためのクラスです。
S15F9 は、レシピの状態(state)とバージョン情報の取得を行います。

10. 6. 1 コンストラクタ

オーバーロードの一覧を示します。

	名前	説明
1	public DshS15F9Send()	装置 ID=0 の空のｲﾝｽﾀﾝｽを生成します。
2	public DshS15F9Send(int eqid)	装置 ID を指定してｲﾝｽﾀﾝｽを生成します。

10. 6. 2 プロパティ

プロパティ一覧表に示します。

	名前	説明
1	public int eqid	装置 ID です。ﾃﾞﾌｫﾙﾄ値=0 です。
2	public string rcpid	ﾚｼﾋﾟ ID です。

10．６．３　メソッド

本クラスのメソッドは次の通りです。

	名前	説明
1	public void set_rcpid()	ﾚｼﾋﾟ ID を設定します。
2	public int send_s15f9() public int send()	S15F9 ﾒｯｾｰｼﾞを送信します。
3	public int send_wait()	S15F9 ﾒｯｾｰｼﾞを送信し、S15F10 を受信します。 ﾌﾟﾛｸﾞﾗﾑは応答受信までﾌﾞﾛｯｸされます。
4	public void Dispose()	ｸﾗｽ内部で使用された資源(ﾒﾓﾘ)ｼｽﾃﾑに返却します。 他のｸﾗｽをﾌﾟﾛﾊﾟﾃｨにしている場合、そのｸﾗｽの Dispose()も行います。 3.1.3.5 と同様です。そちらを参照ください。

10．６．３．１　set_rcpid()

【構文】

public void set_rcpid(string rcpid)

【引数】

rcpid

レシピ ID です。

【戻り値】

なし。

【説明】

レシピ ID をプロパティの rcpid に設定します。

10．６．３．２　send_s15f9(), send()

S15F9 メッセージの送信要求をします。

【構文】

```
public int send_S15F9(string rcpid,
                      ref DshS15Rsp rsp_info, DshCallback.callback_s15f9 callback, uint upara)
public int send_S15F9(ref DshS15Rsp rsp_info, DshCallback.callback_s15f9 callback, uint upara)
```

【引数】

rcpid
　　レシピ ID です。
rsp_info
　　S15F10 応答メッセージに含まれる情報を保存します。コールバック関数の引数になります。
callback
　　S15F9 送信後、2 次メッセージ受信した時に呼び出される callback 関数です。
upara
　　ユーザパラメータ情報です。コールバックされる際に callback 関数の引数として渡されます。
　　要求とコールバック間のタグ情報として使用できます。

【戻り値】

返却値	意　味
0	要求が受け入れられた。
(-1)	要求が受け入れられなかった。

【説明】

当該インスタンスの rcpid から S15F9 メッセージを生成し、それを相手装置に送信するようエンジンに要求します。
要求がエンジンによって受け入れられたときは、0 を返却します。受け入れられなかった場合は(-1)を返却します。
送信後、受信した応答メッセージ情報を中に保存した rsp_info を引数にして、callback 関数を呼び出します。
状態値とバージョン情報は、rsp_info の rcpstate と rcpver プロパティに設定されます。
callback 関数からは 0 を返却してください。

【callback の構文】

```
public delegate int callback_s15f9(
    int eqid,                       // 装置 ID
    int end_status,                 // 終了状態ｺｰﾄﾞ
    ref DshS15Rsp rsp_info,         // S15F10 に含まれる応答です。
    uint upara                      // ﾕｰｻﾞﾊﾟﾗﾒｰﾀ(送信要求ﾒｿｯﾄﾞで指定された upara)
)
```

end_status に返却される値は、=0 の場合は正常終了を意味します。

end_status	意　味
0	送受信が正常に完了した。
-14	T3 ﾀｲﾑｱｳﾄを検出し、応答ﾒｯｾｰｼﾞを受信できなかった。
(-1)	送信できなかった。

10．６．３．３　send_wait()

S15F9 メッセージの送信要求をし,引き続き応答メッセージも受信します。

【構文】

public int send_wait(string rcpid, ref DshS15Rsp rsp_info)
public int send_wait(ref DshS15Rsp rsp_info)

【引数】

rcpid
　　レシピ ID です。
rsp_info
　　S15F10 応答メッセージに含まれる情報を保存します。

【戻り値】

返却値	意　味
0	正常に送受信した。
＜ 0	送受信ｴﾗｰが発生した。(送信ｴﾗｰ or T3 ﾀｲﾑｱｳﾄなど)

【説明】

指定された rcpid から S15F9 メッセージを生成し、それを相手装置に送信するようエンジンに要求します。

本メソッドは送信の後、引き続き S15F10 応答メッセージを待機し、受信が終了したら、上の【戻り値】に示した値を返却します。ei=0 の場合は、受信した応答メッセージに含まれる応答情報が rsp_info に設定され渡されます。

応答メッセージを受信するまでプログラムはブロック（待ち）状態になります。

送信エラー、受信エラーを検出した場合は、ei ＜ 0 の値を返却します。(T3 ﾀｲﾑｱｳﾄも含む)

T3 タイムアウトが発生した場合、DSHDR2 HSMS ドライバーに設定されている T3 プロトコルタイムアウト時間だけブロックされることになります。

10. 7 DshS15F13Send クラス

S15F13 メッセージを送信し、S15F14 応答メッセージを受信するためのクラスです。
S15F13 は、レシピ情報を送信するためのメッセージです。

10. 7. 1 コンストラクタ

オーバーロードの一覧を示します。

	名前	説明
1	public DshS15F13Send()	装置 ID=0 の空のｲﾝｽﾀﾝｽを生成します。
2	public DshS15F13Send(int update_flag, ref DshRecipe rclass)	生成/修正ﾌﾗｸﾞとﾚｼﾋﾟ情報を保存している DshRecipe ｸﾗｽのｲﾝｽﾀﾝｽを指定して装置 ID=0 のｲﾝｽﾀﾝｽを生成します。
3	public DshS15F13Send(int eqid)	装置 ID を指定してｲﾝｽﾀﾝｽを生成します。
4	public DshS15F13Send(int eqid, int update_flag, ref DshRecipe rclass)	装置 ID,生成/修正ﾌﾗｸﾞとﾚｼﾋﾟ情報を保存している DshRecipe ｸﾗｽのｲﾝｽﾀﾝｽを指定してのｲﾝｽﾀﾝｽを生成します。

10. 7. 2 プロパティ

プロパティ一覧表に示します。

	名前	説明
1	public int eqid	装置 ID です。ﾃﾞﾌｫﾙﾄ値=0 です。
2	public int update_flag	生成/修正ﾌﾗｸﾞです。 0=生成、1=修正を意味します。
3	public DshRecipe rcp_class	ﾚｼﾋﾟ情報が保存されている DshRecipe ｸﾗｽのｲﾝｽﾀﾝｽです。

10．７．３　メソッド

本クラスのメソッドは次の通りです。

	名前	説明
1	public void set_update_flag()	生成/修正ﾌﾗｸﾞを設定します。 0=生成、1=修正を意味します。
2	public void set_recipe()	DshRecipe ｸﾗｽのｲﾝｽﾀﾝｽを設定します。
3	public int send_s15f13() public int send()	S15F13 ﾒｯｾｰｼﾞを送信します。
4	public int send_wait()	S15F13 ﾒｯｾｰｼﾞを送信し、S15F14 を受信します。 ﾌﾟﾛｸﾞﾗﾑは応答受信までﾌﾞﾛｯｸされます。
5	public void Dispose()	ｸﾗｽ内部で使用された資源(ﾒﾓﾘ)ｼｽﾃﾑに返却します。 他のｸﾗｽをﾌﾟﾛﾊﾟﾃｨにしている場合、そのｸﾗｽの Dispose()も行います。 3.1.3.5 と同様です。そちらを参照ください。

10．７．３．１　set_update_flag()

【構文】

public void set_update_flag(int flag)

【引数】

flag

生成/修正フラグを指定します。(0=生成、1=修正を意味します)

【戻り値】

なし。

【説明】

flag を update_flag に設定します。

10．７．３．２　set_recipe()

【構文】

public void set_recipe(ref DshRecipe rclass)

【引数】

rclass

レシピ情報が保存されている DshRecipe クラスのインスタンスです。

【戻り値】

なし。

【説明】

rclass のレシピ情報を rcp_class に設定します。（コピーします。）

10．７．３．３　send_s15f13()、send()

S15F13 メッセージの送信要求をします。

【構文】

```
public int send_s15f13(ref DshS15Rsp rsp_info,
                                    DshCallback.callback_s15f13 callback, uint upara)
public int send(ref DshS15Rsp rsp_info,
                                    DshCallback.callback_s15f13 callback, uint upara)
```

【引数】

rsp_info
　　S15F14 応答メッセージに含まれる情報を保存します。コールバック関数の引数になります。
callback
　　S15F13 送信後、2 次メッセージ受信した時に呼び出される callback 関数です。
upara
　　ユーザパラメータ情報です。コールバックされる際に callback 関数の引数として渡されます。
　　要求とコールバック間のタグ情報として使用できます。

【戻り値】

返却値	意　味
0	要求が受け入れられた。
(-1)	要求が受け入れられなかった。

【説明】

当該インスタンスの rcp_class のレシピ情報から S15F13 メッセージを生成し、それを相手装置に送信するようエンジンに要求します。
要求がエンジンによって受け入れられたときは、0 を返却します。受け入れられなかった場合は(-1)を返却します。
送信後、受信した応答メッセージ情報を中に保存した rsp_info を引数にして、callback 関数を呼び出します。
状態値とバージョン情報は、rsp_info の rcpstate と rcpver プロパティに設定されます。
callback 関数からは 0 を返却してください。

【callback の構文】

```
public delegate int callback_s15f13(
    int eqid,                       // 装置 ID
    int end_status,                 // 終了状態ｺｰﾄﾞ
    ref DshS15Rsp rsp_info,         // S15F14 に含まれる応答です。
    uint upara                      // ﾕｰｻﾞﾊﾟﾗﾒｰﾀ(送信要求ﾒｿｯﾄﾞで指定された upara)
)
```

end_status に返却される値は、=0 の場合は正常終了を意味します。

end_status	意　味
0	送受信が正常に完了した。
-14	T3 ﾀｲﾑｱｳﾄを検出し、応答ﾒｯｾｰｼﾞを受信できなかった。
(-1)	送信できなかった。

10．７．３．４　send_wait()

S15F13 メッセージの送信要求をし,引き続き応答メッセージも受信します。

【構文】

public int send_wait(ref DshS15Rsp rsp_info)

【引数】

rsp_info

S15F14 応答メッセージに含まれる情報を保存します。

【戻り値】

返却値	意　味
0	正常に送受信した。
＜ 0	送受信ｴﾗｰが発生した。（送信ｴﾗｰ or T3 ﾀｲﾑｱｳﾄなど）

【説明】

当該インスタンスの rcp_class のレシピ情報から S15F13 メッセージを生成し、それを相手装置に送信するようエンジンに要求します。

本メソッドは送信の後、引き続き S15F14 応答メッセージを待機し、受信が終了したら、上の【戻り値】に示した値を返却します。ei=0 の場合は、受信した応答メッセージに含まれる応答情報が rsp_info に設定され渡されます。

応答メッセージを受信するまでプログラムはブロック（待ち）状態になります。

送信エラー、受信エラーを検出した場合は、ei ＜ 0 の値を返却します。(T3 ﾀｲﾑｱｳﾄも含む)

T3 タイムアウトが発生した場合、DSHDR2 HSMS ドライバーに設定されている T3 プロトコルタイムアウト時間だけブロックされることになります。

10. 8 DshS15F14Response クラス

S15F13 メッセージを受信した後、S15F14 応答メッセージを送信するためのクラスです。

10. 8. 1 コンストラクタ

オーバーロードの一覧を示します。

	名前	説明
1	public DshS15F14Response()	装置 ID=0 のｲﾝｽﾀﾝｽを生成します。
2	public DshS15F14Response(int eqid)	装置 ID を指定してｲﾝｽﾀﾝｽを生成します。

10. 8. 2 プロパティ

プロパティ一覧表に示します。

	名前	説明
1	public int eqid	装置 ID です。ﾃﾞﾌｫﾙﾄ値=0 です。

10．８．３　メソッド

本クラスのメソッドは次の通りです。

	名前	説明
1	public int response()	S15F14 応答ﾒｯｾｰｼﾞを送信します。
2	public void Dispose()	ｸﾗｽ内部で使用された資源(ﾒﾓﾘ)ｼｽﾃﾑに返却します。 他のｸﾗｽをﾌﾟﾛﾊﾟﾃｨにしている場合、そのｸﾗｽの Dispose()も行います。 3.1.3.5 と同様です。そちらを参照ください。

10．８．３．１　response()

S15F14 メッセージの応答送信要求をします。

【構文】

public int response(uint trid, ref DshS15Rsp rsp_class)

【引数】

trid

エンジンからポーリング時に与えられたトランザクション ID です。
処理した１次メッセージとの対応を取ります。

rsp_class

S15F14 メッセージに含める応答情報のインスタンスです。
DshS15Rsp クラスについては Vol-1 の 12.5 を参照してください。

【戻り値】

返却値	意　味
0	要求が受け入れられた。
(-1)	要求が受け入れられなかった。(ﾄﾗﾝｻﾞｸｼｮﾝ ID が正しくなかった。)

【説明】

受信した S15F13 に対する S15F14 メッセージを送信します。
DshS15RSP クラスのインスタンス rsp_class 内の応答情報から S15F14 を生成します。
trid は、この応答メッセージに対する S15F13 受信時にエンジンから与えられたトランザクション ID です。
trid がエンジンの管理の中に見つからなかった場合には、(-1)を返却します。
要求が受け入れられたら 0 を返却します。この後、応答メッセージ送信結果は特に通知されません。

10. 9 DshS15F17Send クラス

S15F17 メッセージを送信し、S15F18 応答メッセージを受信するためのクラスです。
S15F17 は、レシピ検索要求を送信するためのメッセージです。

10. 9. 1 コンストラクタ

オーバーロードの一覧を示します。

	名前	説明
1	public DshS15F17Send()	装置 ID=0 の空のｲﾝｽﾀﾝｽを生成します。
2	public DshS15F17Send(string rcpid, int rcpseccode)	ﾚｼﾋﾟ ID とﾚｼﾋﾟｾｸｼｮﾝを指定して装置 ID=0 のｲﾝｽﾀﾝｽを生成します。
3	public DshS15F17Send(int eqid)	装置 ID を指定してｲﾝｽﾀﾝｽを生成します。
4	public DshS15F17Send(int eqid, int update_flag, ref DshRecipe rclass)	装置 ID, ﾚｼﾋﾟ ID とﾚｼﾋﾟｾｸｼｮﾝを指定してのｲﾝｽﾀﾝｽを生成します。

10. 9. 2 プロパティ

プロパティ一覧表に示します。

	名前	説明
1	public int eqid	装置 ID です。ﾃﾞﾌｫﾙﾄ値=0 です。
2	public string rcpid	検索対象ﾚｼﾋﾟ ID です。
3	public int rcpseccode	検索対象のﾚｼﾋﾟｾｸｼｮﾝｺｰﾄﾞです。

10．9．3　メソッド

本クラスのメソッドは次の通りです。

	名前	説明
1	public void set_info()	ﾚｼﾋﾟ ID とﾚｼﾋﾟｾｸｼｮﾝｺｰﾄﾞを設定します。
2	public int send_s15f17() public int send()	S15F17 ﾒｯｾｰｼﾞを送信します。
3	public int send_wait()	S15F17 ﾒｯｾｰｼﾞを送信し、S15F18 を受信します。 ﾌﾟﾛｸﾞﾗﾑは応答受信までﾌﾞﾛｯｸされます。
4	public void Dispose()	ｸﾗｽ内部で使用された資源(ﾒﾓﾘ)ｼｽﾃﾑに返却します。 他のｸﾗｽをﾌﾟﾛﾊﾟﾃｨにしている場合、そのｸﾗｽの Dispose()も行います。 3.1.3.5 と同様です。そちらを参照ください。

10．9．3．1　set_info()

【構文】

public void set_info(string rcpid, int rcpseccode)

【引数】

rcpid

検索するレシピ ID です。

rcpseccode

検索するレシピセクションコードです。

【戻り値】

なし。

【説明】

レシピ ID とレシピセクションコードをプロパティの rcpid, rcpseccode に設定します。

10．9．3．2　send_s15f17(), send()

S15F17 メッセージの送信要求をします。

【構文】

```
public int send_S15F17(string rcpid, int rcpseccode,
            ref DshRcpS15F18Rsp rsp_info, DshCallback.callback_s15f17 callback, uint upara)
public int send(string rcpid, int rcpseccode,
            ref DshRcpS15F18Rsp rsp_info, DshCallback.callback_s15f17 callback, uint upara)
public int send_S15F17(
            ref DshRcpS15F18Rsp rsp_info, DshCallback.callback_s15f17 callback, uint upara)
public int send(
            ref DshRcpS15F18Rsp rsp_info, DshCallback.callback_s15f17 callback, uint upara)
```

【引数】

rcpid
　　検索するレシピ ID です。
rcpseccode
　　検索するレシピセクションコードです。
rsp_info
　　S15F18 応答メッセージに含まれる情報を保存します。コールバック関数の引数になります。
callback
　　S15F17 送信後、2 次メッセージ受信した時に呼び出される callback 関数です。
upara
　　ユーザパラメータ情報です。コールバックされる際に callback 関数の引数として渡されます。
　　要求とコールバック間のタグ情報として使用できます。

【戻り値】

返却値	意　味
0	要求が受け入れられた。
(-1)	要求が受け入れられなかった。

【説明】

当該インスタンスの rcpid と rcpseccode の情報から S15F17 メッセージを生成し、それを相手装置に送信するようエンジンに要求します。
要求がエンジンによって受け入れられたときは、0 を返却します。受け入れられなかった場合は(-1)を返却します。
送信後、受信した応答メッセージ情報を中に保存した rsp_info を引数にして、callback 関数を呼び出します。
rsp_info に検索結果情報が保存されます。
callback 関数からは 0 を返却してください。

【callback の構文】

```
public delegate int callback_s15f17(
    int eqid,                          // 装置 ID
    int end_status,                    // 終了状態ｺｰﾄﾞ
    ref DshS15F18Rsp rsp_info,         // S15F18 に含まれる応答です。   Vol-1 12.6 参照
    uint upara                         // ﾕｰｻﾞﾊﾟﾗﾒｰﾀ(送信要求ﾒｿｯﾄﾞで指定された upara)
)
```

end_status に返却される値は、=0 の場合は正常終了を意味します。

end_status	意　味
0	送受信が正常に完了した。
-18	T3 ﾀｲﾑｱｳﾄを検出し、応答ﾒｯｾｰｼﾞを受信できなかった。
(-1)	送信できなかった。

10．９．３．３　send_wait()

S15F17 メッセージの送信要求をし,引き続き応答メッセージも受信します。

【構文】

```
public int send_wait(ref DshS15Rsp rsp_info)
public int send_wait(string rcpid, int rcpseccode, ref DshS15Rsp rsp_info)
```

【引数】

rcpid
　　検索するレシピ ID です。
rcpseccode
　　検索するレシピセクションコードです。
rsp_info
　　S15F18 応答メッセージに含まれる情報を保存します。

【戻り値】

返却値	意　味
0	正常に送受信した。
＜ 0	送受信ｴﾗｰが発生した。(送信ｴﾗｰ or T3 ﾀｲﾑｱｳﾄなど)

【説明】

指定された rcpid と rcpseccode の情報から S15F17 メッセージを生成し、それを相手装置に送信するようエンジンに要求します。

本メソッドは送信の後、引き続き S15F18 応答メッセージを待機し、受信が終了したら、上の【戻り値】に示した値を返却します。ei=0 の場合は、受信した応答メッセージに含まれる応答情報が rsp_info に設定され渡されます。

応答メッセージを受信するまでプログラムはブロック（待ち）状態になります。

送信エラー、受信エラーを検出した場合は、ei ＜ 0 の値を返却します。(T3 ﾀｲﾑｱｳﾄも含む)

T3 タイムアウトが発生した場合、DSHDR2 HSMS ドライバーに設定されている T3 プロトコルタイムアウト時間だけブロックされることになります。

10. 10 DshS15F18Response クラス

S15F17 メッセージを受信した後、S15F18 応答メッセージを送信するためのクラスです。

10. 10. 1 コンストラクタ

オーバーロードの一覧を示します。

	名前	説明
1	public DshS15F18Response()	装置 ID=0 のｲﾝｽﾀﾝｽを生成します。
2	public DshS15F18Response(int eqid)	装置 ID を指定してｲﾝｽﾀﾝｽを生成します。

10. 10. 2 プロパティ

プロパティ一覧表に示します。

	名前	説明
1	public int eqid	装置 ID です。ﾃﾞﾌｫﾙﾄ値=0 です。

10. 10. 3　メソッド

本クラスのメソッドは次の通りです。

	名前	説明
1	public int response()	S15F18 応答ﾒｯｾｰｼﾞを送信します。
2	public void Dispose()	ｸﾗｽ内部で使用された資源(ﾒﾓﾘ)ｼｽﾃﾑに返却します。 他のｸﾗｽをﾌﾟﾛﾊﾟﾃｨにしている場合、そのｸﾗｽの Dispose()も行います。 3.1.3.5 と同様です。そちらを参照ください。

10. 10. 3. 1　response()

S15F18 メッセージの応答送信要求をします。

【構文】

public int response(uint trid, ref DshS15F18Rsp rsp_class)

【引数】

trid

エンジンからポーリング時に与えられたトランザクション ID です。
処理した１次メッセージとの対応を取ります。

rsp_class

S15F18 メッセージに含める応答情報のインスタンスです。
DshS15F18Rsp クラスについては Vol-1 の 12.6 を参照してください。

【戻り値】

返却値	意　味
0	要求が受け入れられた。
(-1)	要求が受け入れられなかった。(ﾄﾗﾝｻﾞｸｼｮﾝ ID が正しくなかった。)

【説明】

受信した S15F17 に対する S15F18 メッセージを送信します。
DshS15RSP クラスのインスタンス rsp_class 内の応答情報から S15F18 を生成します。
trid は、この応答メッセージに対する S15F17 受信時にエンジンから与えられたトランザクション ID です。
trid がエンジンの管理の中に見つからなかった場合には、(-1)を返却します。
要求が受け入れられたら 0 を返却します。この後、応答メッセージ送信結果は特に通知されません。

11. プロセス・ジョブ関連メッセージの送受信

以下のメッセージとクラスがあります。

	メッセージ	クラス名	機能概略
1	S16F5,S16F6	DshS16F5Send	S16F5 送信　Process Job Command Request
		DshS16F6Response	S16F6 応答
2	S16F11,S16F12	DshS16F11Send	S16F11 送信　PrJobCreateEnh
		DshS16F12Response	S16F12 応答
3	S16F15,S16F16	DshS16F15Send	S16F15 送信　PrJobMultiCreate
		DshS16F16Response	S16F16 応答
4	S16F17,S16F18	DshS16F17Send	S16F17 送信　PrJobDeque
		DshS16F18Response	S16F18 応答
5	S16F19,S16F20	DshS16F19Send	S16F19 送信　PrGetAllJobs
		-	(S16F20 はｴﾝｼﾞﾝが自動応答)
6	S16F21,S16F22	DshS16F21Send	S16F21 送信　PrGetSpace
		-	(S16F22 はｴﾝｼﾞﾝが自動応答)

11. 1 DshS16F5Send クラス

S16F5 メッセージを送信し、S16F6 応答メッセージを受信するためのクラスです。
プロセスジョブのコマンド要求を行います。

11. 1. 1 コンストラクタ

オーバーロードの一覧を示します。

	名前	説明
1	public DshS16F5Send()	装置 ID=0 の空のｲﾝｽﾀﾝｽを生成します。
2	public DshS16F5Send(string prjid, string cmd)	ﾌﾟﾛｾｽｼﾞｮﾌﾞ ID とｺﾏﾝﾄﾞ値を指定して装置 ID=0 のｲﾝｽﾀﾝｽを生成します。
3	public DshS16F5Send(int eqid)	装置 ID を指定してｲﾝｽﾀﾝｽを生成します。
4	public DshS16F5Send(int eqid, string prjid, string cmd)	装置 ID, ﾌﾟﾛｾｽｼﾞｮﾌﾞ ID とｺﾏﾝﾄﾞ値を指定してｲﾝｽﾀﾝｽを生成します。

11. 1. 2 プロパティ

プロパティ一覧表に示します。

	名前	説明
1	public int eqid	装置 ID です。ﾃﾞﾌｫﾙﾄ値=0 です。
2	public DshPrjCmd cmd_class	ﾌﾟﾛｾｽｼﾞｮﾌﾞｺﾏﾝﾄﾞ情報を保存するための DshPrjCmd ｸﾗｽのｲﾝｽﾀﾝｽです。Vol-1 13-4 参照

11. 1. 3 メソッド

本クラスのメソッドは次の通りです。

	名前	説明
1	public void set_info()	cmd_class に DshPrjCmd ｸﾗｽのｲﾝｽﾀﾝｽを設定します。
2	public void add_para()	cmd_class(DshPrjCmd ｸﾗｽのｲﾝｽﾀﾝｽ)に 1 個の付属ﾊﾟﾗﾒｰﾀを追加します。
3	public int send_s16f5() public int send()	S16F5 ﾒｯｾｰｼﾞを送信します。
4	public int send_wait()	S16F5 ﾒｯｾｰｼﾞを送信し、S16F6 を受信します。 ﾌﾟﾛｸﾞﾗﾑは応答受信までﾌﾞﾛｯｸされます。
5	public void Dispose()	ｸﾗｽ内部で使用された資源(ﾒﾓﾘ)ｼｽﾃﾑに返却します。 他のｸﾗｽをﾌﾟﾛﾊﾟﾃｨにしている場合、そのｸﾗｽの Dispose()も行います。 3.1.3.5 と同様です。そちらを参照ください。

11. 1. 3. 1 set_info()

【構文】

public void set_info(ref DshPrjCmd info)

【引数】

info

設定したい DshPrjCmd クラスのインスタンスです。

【戻り値】

なし。

【説明】

info に与えられた DshPrjCmd クラスのインスタンスの内容を cmd_class に設定します。

11. １. ３. ２　add_para()

【構文】

public void add_para(string name, int format, int size, IntPtr value)
public void add_para(string name, string value)

【引数】

pname
　　コマンドパラメータ名です。
format
　　パラメータ値のフォーマットです。(HSMS.ICODE_U1 など)
size
　　パラメータ値の配列サイズです。(HSMS.ICODE_A 以外は 1 です)
value
　　パラメータ値です。(値が格納されているポインタまたは文字列です。)

【戻り値】

なし。

【説明】

cmd_class のプロパティである list 配列に 1 個のパラメータ情報を追加します。
value が string で与えられた場合は、HSMS.ICODE_A のフォーマットを意味します。

11. 1. 3. 3 send_s16f5(), send()

S16F5 メッセージの送信要求をします。

【構文】

```
public int send_S16F5( ref DshS16Rsp rsp_info, DshCallback.callback_s16f5 callback, uint upara)
public int send( ref DshS16Rsp rsp_info, DshCallback.callback_s16f5 callback, uint upara)
```

【引数】

rsp_info

S16F6 応答メッセージに含まれる情報を保存します。コールバック関数の引数になります。

callback

S16F5 送信後、2 次メッセージ受信した時に呼び出される callback 関数です。

upara

ユーザパラメータ情報です。コールバックされる際に callback 関数の引数として渡されます。
要求とコールバック間のタグ情報として使用できます。

【戻り値】

返却値	意　味
0	要求が受け入れられた。
(-1)	要求が受け入れられなかった。

【説明】

当該インスタンスの cmd_class から S16F5 メッセージを生成し、それを相手装置に送信するようエンジンに要求します。
要求がエンジンによって受け入れられたときは、0 を返却します。受け入れられなかった場合は(-1)を返却します。
送信後、受信した応答メッセージ情報を中に保存した rsp_info を引数にして、callback 関数を呼び出します。
callback 関数からは 0 を返却してください。

【callback の構文】

```
public delegate int callback_s16f5(
    int eqid,                       // 装置 ID
    int end_status,                 // 終了状態ｺｰﾄﾞ
    ref DshS16Rsp rsp_info,         // S16F6 に含まれる応答です。 Vol-1 13.5 参照
    uint upara                      // ﾕｰｻﾞﾊﾟﾗﾒｰﾀ(送信要求ﾒｿｯﾄﾞで指定された upara)
)
```

end_status に返却される値は、=0 の場合は正常終了を意味します。

end_status	意　味
0	送受信が正常に完了した。
-14	T3 ﾀｲﾑｱｳﾄを検出し、応答ﾒｯｾｰｼﾞを受信できなかった。
(-1)	送信できなかった。

11. １．３．４　send_wait()

S16F5 メッセージの送信要求をし,引き続き応答メッセージも受信します。

【構文】

public int send_wait(ref DshS16Rsp rsp_info)

【引数】

rsp_info

S16F6 応答メッセージに含まれる情報を保存します。

【戻り値】

返却値	意　味
0	正常に送受信した。
＜ 0	送受信ｴﾗｰが発生した。（送信ｴﾗｰ or T3 ﾀｲﾑｱｳﾄなど）

【説明】

当該インスタンスの cmd_class から S16F5 メッセージを生成し、それを相手装置に送信するようエンジンに要求します。

本メソッドは送信の後、引き続き S16F6 応答メッセージを待機し、受信が終了したら、上の【戻り値】に示した値を返却します。ei=0 の場合は、受信した応答メッセージに含まれる応答情報が rsp_info に設定され渡されます。

応答メッセージを受信するまでプログラムはブロック（待ち）状態になります。

送信エラー、受信エラーを検出した場合は、ei ＜ 0 の値を返却します。(T3 ﾀｲﾑｱｳﾄも含む)

T3 タイムアウトが発生した場合、DSHDR2 HSMS ドライバーに設定されている T3 プロトコルタイムアウト時間だけブロックされることになります。

11. 2 DshS16F6Response クラス

S16F5 メッセージを受信した後、S16F6 応答メッセージを送信するためのクラスです。

11. 2. 1 コンストラクタ

オーバーロードの一覧を示します。

	名前	説明
1	public DshS16F6Response()	装置 ID=0 のｲﾝｽﾀﾝｽを生成します。
2	public DshS16F6Response(int eqid)	装置 ID を指定してｲﾝｽﾀﾝｽを生成します。

11. 2. 2 プロパティ

プロパティ一覧表に示します。

	名前	説明
1	public int eqid	装置 ID です。ﾃﾞﾌｫﾙﾄ値=0 です。

11．２．３　メソッド

本クラスのメソッドは次の通りです。

	名前	説明
1	public int response()	S16F6 応答ﾒｯｾｰｼﾞを送信します。
2	public void Dispose()	ｸﾗｽ内部で使用された資源(ﾒﾓﾘ)ｼｽﾃﾑに返却します。 他のｸﾗｽをﾌﾟﾛﾊﾟﾃｨにしている場合、そのｸﾗｽの Dispose()も行います。 3.1.3.5 と同様です。そちらを参照ください。

11．２．３．１　response()

S16F6 メッセージの応答送信要求をします。

【構文】

public int response(uint trid, ref DshS16Rsp rsp_class)

【引数】

trid

エンジンからポーリング時に与えられたトランザクション ID です。
処理した１次メッセージとの対応を取ります。

rsp_class

S16F6 メッセージに含める応答情報のインスタンスです。
DshS16Rsp クラスについては Vol-1 の 12.5 を参照してください。

【戻り値】

返却値	意　味
0	要求が受け入れられた。
(-1)	要求が受け入れられなかった。(ﾄﾗﾝｻﾞｸｼｮﾝ ID が正しくなかった。)

【説明】

受信した S16F5 に対する S16F6 メッセージを送信します。
DshS16Rsp クラスのインスタンス rsp_class 内の応答情報から S16F6 を生成します。
trid は、この応答メッセージに対する S16F5 受信時にエンジンから与えられたトランザクション ID です。
trid がエンジンの管理の中に見つからなかった場合には、(-1)を返却します。
要求が受け入れられたら 0 を返却します。この後、応答メッセージ送信結果は特に通知されません。

11. 3 DshS16F11Send クラス

S16F11 メッセージを送信し、S16F12 応答メッセージを受信するためのクラスです。
プロセスジョブ情報の送信を行います。

11. 3. 1 コンストラクタ

オーバーロードの一覧を示します。

	名前	説明
1	public DshS16F11Send()	装置 ID=0 の空のｲﾝｽﾀﾝｽを生成します。
2	public DshS16F11Send(ref DshPrj prj_class)	DshPrj ｸﾗｽのｲﾝｽﾀﾝｽを指定して装置 ID=0 のｲﾝｽﾀﾝｽを生成します。
3	public DshS16F11Send(int eqid)	装置 ID を指定してｲﾝｽﾀﾝｽを生成します。
4	public DshS16F11Send(int eqid, ref DshPrj prj_class)	装置 ID, DshPrj ｸﾗｽのｲﾝｽﾀﾝｽを指定してｲﾝｽﾀﾝｽを生成します。

11. 3. 2 プロパティ

プロパティ一覧表に示します。

	名前	説明
1	public int eqid	装置 ID です。ﾃﾞﾌｫﾙﾄ値=0 です。
2	public DshPrj prj_class	ﾌﾟﾛｾｽｼﾞｮﾌﾞ情報を保存するための DshPrj ｸﾗｽのｲﾝｽﾀﾝｽです。Vol-1 13-1 参照

11．３．３　メソッド

本クラスのメソッドは次の通りです。

	名前	説明
1	public void set_info()	prj_class に DshPrj ｸﾗｽのｲﾝｽﾀﾝｽを設定します。
2	public int send_s16f11() public int send()	S16F11 ﾒｯｾｰｼﾞを送信します。
3	public int send_wait()	S16F11 ﾒｯｾｰｼﾞを送信し、S16F12 を受信します。 ﾌﾟﾛｸﾞﾗﾑは応答受信までﾌﾞﾛｯｸされます。
4	public void Dispose()	ｸﾗｽ内部で使用された資源(ﾒﾓﾘ)ｼｽﾃﾑに返却します。 他のｸﾗｽをﾌﾟﾛﾊﾟﾃｨにしている場合、そのｸﾗｽの Dispose()も行います。 3.1.3.5 と同様です。そちらを参照ください。

11．３．３．１　set_info()

【構文】

public void set_info(ref DshPrj info)

【引数】

info

設定したい DshPrj クラスのインスタンスです。

【戻り値】

なし。

【説明】

info に与えられた DshPrj クラスのインスタンスの内容を prj_class に設定します。
実際にはコピーします。

11．３．３．２　send_s16f11(), send()

S16F11 メッセージの送信要求をします。

【構文】

```
public int send_s16f11(string prjid,
                       ref DshS16Rsp rsp_info, DshCallback.callback_s16f11 callback, uint upara)
public int send(string prjid,
                       ref DshS16Rsp rsp_info, DshCallback.callback_s16f11 callback, uint upara)
public int send_s16f11(ref DshPrj prj_class,
                       ref DshS16Rsp rsp_info, DshCallback.callback_s16f11 callback, uint upara)
public int send(ref DshPrj prj_class,
                       ref DshS16Rsp rsp_info, DshCallback.callback_s16f11 callback, uint upara)
public int send_s16f11(ref DshS16Rsp rsp_info, DshCallback.callback_s16f11 callback, uint upara)
public int send(ref DshS16Rsp rsp_info, DshCallback.callback_s16f11 callback, uint upara)
```

【引数】

prjid
　　プロセスジョブ ID を指定します。エンジンから情報を prj_class に取得します。

prj_class
　　プロパティの prj_class にコピーします。

rsp_info
　　S16F12 応答メッセージに含まれる情報を保存します。コールバック関数の引数になります。

callback
　　S16F11 送信後、2 次メッセージ受信した時に呼び出される callback 関数です。

upara
　　ユーザパラメータ情報です。コールバックされる際に callback 関数の引数として渡されます。
　　要求とコールバック間のタグ情報として使用できます。

【戻り値】

返却値	意　味
0	要求が受け入れられた。
(-1)	要求が受け入れられなかった。または prjid が管理外であった。

【説明】

当該インスタンスの prj_class から S16F11 メッセージを生成し、それを相手装置に送信するようエンジンに要求します。

プロセスジョブ ID が引数に指定されていた場合にはそのプロセスジョブ情報をエンジンの管理から prj_class プロパティに取得した上で送信処理をします。
また、DshPrj クラスのインスタンスが引数に与えられた場合には、それを prj_class プロパティにコピーした上で送信処理を行います。

要求がエンジンによって受け入れられたときは、0 を返却します。受け入れられなかった場合は(-1)を返却します。
送信後、受信した応答メッセージ情報を保存した rsp_info 引数にして、callback 関数を呼び出します。
callback 関数からは 0 を返却してください。

【callback の構文】

```
public delegate int callback_s16f11(
    int eqid,                          // 装置 ID
    int end_status,                    // 終了状態ｺｰﾄﾞ
    ref DshS16Rsp rsp_info,            // S16F12 に含まれる応答です。 Vol-1 13.5 参照
    uint upara                         // ﾕｰｻﾞﾊﾟﾗﾒｰﾀ(送信要求ﾒｿｯﾄﾞで指定された upara)
)
```

end_status に返却される値は、=0 の場合は正常終了を意味します。

end_status	意　味
0	送受信が正常に完了した。
-14	T3 ﾀｲﾑｱｳﾄを検出し、応答ﾒｯｾｰｼﾞを受信できなかった。
(-1)	送信できなかった。

11．３．３．３　send_wait()

S16F11 メッセージの送信要求をし,引き続き応答メッセージも受信します。

【構文】

```
public int send_wait(string prjid, ref DshS16Rsp rsp_info)
public int send_wait(ref DshPrj prj_class, ref DshS16Rsp rsp_info)
public int send_wait(ref DshS16Rsp rsp_info)
```

【引数】

prjid
　　プロセスジョブ ID を指定します。エンジンから情報を prj_class に取得します。

prj_class
　　プロパティの prj_class にコピーします。

rsp_info
　　S16F12 応答メッセージに含まれる情報を保存します。

【戻り値】

返却値	意　味
0	正常に送受信した。
＜ 0	送受信ｴﾗｰが発生した。(送信ｴﾗｰ or T3 ﾀｲﾑｱｳﾄなど)

【説明】

当該インスタンスの prj_class から S16F11 メッセージを生成し、それを相手装置に送信するようエンジンに要求します。

プロセスジョブ ID が引数に指定されていた場合にはそのプロセスジョブ情報をエンジンの管理から prj_class プロパティに取得した上で送信処理をします。
また、DshPrj クラスのインスタンスが引数に与えられた場合には、それを prj_class プロパティにコピーした上で送信処理を行います。

本メソッドは送信の後、引き続き S16F12 応答メッセージを待機し、受信が終了したら、上の【戻り値】に示した値を返却します。ei=0 の場合は、受信した応答メッセージに含まれる応答情報が rsp_info に設定され渡されます。

応答メッセージを受信するまでプログラムはブロック（待ち）状態になります。

送信エラー、受信エラーを検出した場合は、ei ＜ 0 の値を返却します。(T3 ﾀｲﾑｱｳﾄも含む)

T3 タイムアウトが発生した場合、DSHDR2 HSMS ドライバーに設定されている T3 プロトコルタイムアウト時間だけブロックされることになります。

11. 4 DshS16F12Response クラス

S16F11 メッセージを受信した後、S16F12 応答メッセージを送信するためのクラスです。

11. 4. 1 コンストラクタ

オーバーロードの一覧を示します。

	名前	説明
1	public DshS16F12Response()	装置 ID=0 のｲﾝｽﾀﾝｽを生成します。
2	public DshS16F12Response(int eqid)	装置 ID を指定してｲﾝｽﾀﾝｽを生成します。

11. 4. 2 プロパティ

プロパティ一覧表に示します。

	名前	説明
1	public int eqid	装置 ID です。ﾃﾞﾌｫﾙﾄ値=0 です。

11．４．３　メソッド

本クラスのメソッドは次の通りです。

	名前	説明
1	public int response()	S16F12 応答ﾒｯｾｰｼﾞを送信します。
2	public void Dispose()	ｸﾗｽ内部で使用された資源(ﾒﾓﾘ)ｼｽﾃﾑに返却します。 他のｸﾗｽをﾌﾟﾛﾊﾟﾃｨにしている場合、そのｸﾗｽの Dispose()も行います。 3.1.3.5 と同様です。そちらを参照ください。

11．４．３．１　response()

S16F12 メッセージの応答送信要求をします。

【構文】

public int response(uint trid, ref DshS16Rsp rsp_class)

【引数】

trid

エンジンからポーリング時に与えられたトランザクション ID です。
処理した１次メッセージとの対応を取ります。

rsp_class

S16F12 メッセージに含める応答情報のインスタンスです。
DshS16Rsp クラスについては Vol-1 の 12.5 を参照してください。

【戻り値】

返却値	意　味
0	要求が受け入れられた。
(-1)	要求が受け入れられなかった。(ﾄﾗﾝｻﾞｸｼｮﾝ ID が正しくなかった。)

【説明】

受信した S16F11 に対する S16F12 メッセージを送信します。
DshS16Rsp クラスのインスタンス rsp_class 内の応答情報から S16F12 を生成します。
trid は、この応答メッセージに対する S16F11 受信時にエンジンから与えられたトランザクション ID です。
trid がエンジンの管理の中に見つからなかった場合には、(-1)を返却します。
要求が受け入れられたら 0 を返却します。この後、応答メッセージ送信結果は特に通知されません。

11. 5 DshS16F15Send クラス

S16F15 メッセージを送信し、S16F16 応答メッセージを受信するためのクラスです。
複数個のプロセスジョブ ID 分のプロセスジョブ情報の送信を行います。

11. 5. 1 コンストラクタ

オーバーロードの一覧を示します。

	名前	説明
1	public DshS16F15Send()	装置 ID=0 の空のｲﾝｽﾀﾝｽを生成します。
2	public DshS16F15Send(int eqid)	装置 ID を指定してｲﾝｽﾀﾝｽを生成します。

11. 5. 2 プロパティ

プロパティ一覧表に示します。

	名前	説明
1	public int eqid	装置 ID です。ﾃﾞﾌｫﾙﾄ値=0 です。
2	public string[] id_list	複数のﾌﾟﾛｾｽｼﾞｮﾌﾞ ID を保存するための配列です。

11．５．３　メソッド

本クラスのメソッドは次の通りです。

	名前	説明
1	public void add_id()	id_list 配列に 1 個のﾌﾟﾛｾｽｼﾞｮﾌﾞ ID を追加します。
2	public int send_s16f15() public int send()	S16F15 ﾒｯｾｰｼﾞを送信します。
3	public int send_wait()	S16F15 ﾒｯｾｰｼﾞを送信し、S16F16 を受信します。 ﾌﾟﾛｸﾞﾗﾑは応答受信までﾌﾞﾛｯｸされます。
4	public void Dispose()	ｸﾗｽ内部で使用された資源(ﾒﾓﾘ)ｼｽﾃﾑに返却します。 他のｸﾗｽをﾌﾟﾛﾊﾟﾃｨにしている場合、そのｸﾗｽの Dispose()も行います。 3.1.3.5 と同様です。そちらを参照ください。

11．５．３．１　add_id()

【構文】

public void add_id(string prjid)

【引数】

prjid

プロセスジョブ ID です。

【戻り値】

なし。

【説明】

prjid のプロセスジョブ ID を id_list 配列に追加します。

その後、count+1 します。

11．５．３．２　send_s16f15(), send()

S16F15 メッセージの送信要求をします。

【構文】

```
public int send_s16f15(ref DshS16MultiPrjRsp rsp_info,
                                    DshCallback.callback_s16f15 callback, uint upara)
public int send(ref DshS16MultiPrjRsp rsp_info,
                                    DshCallback.callback_s16f15 callback, uint upara)
```

【引数】

rsp_info

S16F16 応答メッセージに含まれる情報を保存します。コールバック関数の引数になります。

callback

S16F15 送信後、2 次メッセージ受信した時に呼び出される callback 関数です。

upara

ユーザパラメータ情報です。コールバックされる際に callback 関数の引数として渡されます。要求とコールバック間のタグ情報として使用できます。

【戻り値】

返却値	意　味
0	要求が受け入れられた。
(-1)	要求が受け入れられなかった。または id が管理外であった。

【説明】

当該インスタンスの id_list 配列に含まれる 1 個以上のプロセスジョブ ID のプロセスジョブ情報をエンジンの管理情報から取得し、 S16F15 メッセージを生成し、それを相手装置に送信するようエンジンに要求します。

要求がエンジンによって受け入れられたときは、0 を返却します。受け入れられなかった場合は(-1)を返却します。
送信後、受信した応答メッセージ情報を中に保存した rsp_info を引数にして、callback 関数を呼び出します。
callback 関数からは 0 を返却してください。

【callback の構文】

```
public delegate int callback_s16f15(
    int eqid,                          // 装置 ID
    int end_status,                    // 終了状態ｺｰﾄﾞ
    ref DshS16MultiPrjRsp rsp_info,    // S16F16 に含まれる応答です。 Vol-1 13.6 参照
    uint upara                         // ﾕｰｻﾞﾊﾟﾗﾒｰﾀ(送信要求ﾒｿｯﾄﾞで指定された upara)
)
```

end_status に返却される値は、=0 の場合は正常終了を意味します。

end_status	意　味
0	送受信が正常に完了した。
-14	T3 ﾀｲﾑｱｳﾄを検出し、応答ﾒｯｾｰｼﾞを受信できなかった。
(-1)	送信できなかった。

11．５．３．３　send_wait()

S16F15 メッセージの送信要求をし,引き続き応答メッセージも受信します。

【構文】

public int send_wait(ref DshS16MultiPrjRsp rsp_info)

【引数】

rsp_info

S16F16 応答メッセージに含まれる情報を保存します。

【戻り値】

返却値	意　味
0	正常に送受信した。
＜ 0	送受信ｴﾗｰが発生した。（送信ｴﾗｰ or T3 ﾀｲﾑｱｳﾄなど）

【説明】

当該インスタンスの id_list 配列に含まれる 1 個以上のプロセスジョブ ID のプロセスジョブ情報をエンジンの管理情報から取得し、 S16F15 メッセージを生成し、それを相手装置に送信するようエンジンに要求します。

本メソッドは送信の後、引き続き S16F16 応答メッセージを待機し、受信が終了したら、上の【戻り値】に示した値を返却します。ei=0 の場合は、受信した応答メッセージに含まれる応答情報が rsp_info に設定され渡されます。

応答メッセージを受信するまでプログラムはブロック（待ち）状態になります。

送信エラー、受信エラーを検出した場合は、ei ＜ 0 の値を返却します。（T3 ﾀｲﾑｱｳﾄも含む）

T3 タイムアウトが発生した場合、DSHDR2 HSMS ドライバーに設定されている T3 プロトコルタイムアウト時間だけブロックされることになります。

11. 6 DshS16F16Response クラス

S16F15 メッセージを受信した後、S16F16 応答メッセージを送信するためのクラスです。

11. 6. 1 コンストラクタ

オーバーロードの一覧を示します。

	名前	説明
1	public DshS16F16Response()	装置 ID=0 のｲﾝｽﾀﾝｽを生成します。
2	public DshS16F16Response(int eqid)	装置 ID を指定してｲﾝｽﾀﾝｽを生成します。

11. 6. 2 プロパティ

プロパティ一覧表に示します。

	名前	説明
1	public int eqid	装置 ID です。ﾃﾞﾌｫﾙﾄ値=0 です。

11．６．３　メソッド

本クラスのメソッドは次の通りです。

	名前	説明
1	public int response()	S16F16 応答ﾒｯｾｰｼﾞを送信します。
2	public void Dispose()	ｸﾗｽ内部で使用された資源(ﾒﾓﾘ)ｼｽﾃﾑに返却します。 他のｸﾗｽをﾌﾟﾛﾊﾟﾃｨにしている場合、そのｸﾗｽの Dispose()も行います。 3.1.3.5 と同様です。そちらを参照ください。

11．６．３．１　response()

S16F16 メッセージの応答送信要求をします。

【構文】

public int response(uint trid, ref DshS16MultiPrjRsp rsp_class)

【引数】

trid

エンジンからポーリング時に与えられたトランザクション ID です。
処理した１次メッセージとの対応を取ります。

rsp_class

S16F16 メッセージに含める応答情報のインスタンスです。
DshS16MultiPrjRsp クラスについては Vol-1 の 13.6 を参照してください。

【戻り値】

返却値	意　味
0	要求が受け入れられた。
(-1)	要求が受け入れられなかった。(ﾄﾗﾝｻﾞｸｼｮﾝ ID が正しくなかった。)

【説明】

受信した S16F15 に対する S16F16 メッセージを送信します。
DshS16MultiPrjRsp クラスのインスタンス rsp_class 内の応答情報から S16F16 を生成します。
trid は、この応答メッセージに対する S16F15 受信時にエンジンから与えられたトランザクション ID です。
trid がエンジンの管理の中に見つからなかった場合には、(-1)を返却します。
要求が受け入れられたら 0 を返却します。この後、応答メッセージ送信結果は特に通知されません。

11. 7　DshS16F17Send クラス

S16F17 メッセージを送信し、S16F18 応答メッセージを受信するためのクラスです。
複数個のプロセスジョブ ID 分のプロセスジョブ削除のための送信を行います。

11. 7. 1　コンストラクタ

オーバーロードの一覧を示します。

	名前	説明
1	public DshS16F17Send()	装置 ID=0 の空のｲﾝｽﾀﾝｽを生成します。
2	public DshS16F17Send(int eqid)	装置 ID を指定してｲﾝｽﾀﾝｽを生成します。

11. 7. 2　プロパティ

プロパティ一覧表に示します。

	名前	説明
1	public int eqid	装置 ID です。ﾃﾞﾌｫﾙﾄ値=0 です。
2	public string[] list	複数のﾌﾟﾛｾｽｼﾞｮﾌﾞ ID を保存するための配列です。

11．７．３　メソッド

本クラスのメソッドは次の通りです。

	名前	説明
1	public void add_prjid()	list 配列に 1 個のﾌﾟﾛｾｽｼﾞｮﾌﾞ ID を追加します。
2	public int send_s16f17() public int send()	S16F17 ﾒｯｾｰｼﾞを送信します。
3	public int send_wait()	S16F17 ﾒｯｾｰｼﾞを送信し、S16F18 を受信します。 ﾌﾟﾛｸﾞﾗﾑは応答受信までﾌﾞﾛｯｸされます。
4	public void Dispose()	ｸﾗｽ内部で使用された資源(ﾒﾓﾘ)ｼｽﾃﾑに返却します。 他のｸﾗｽをﾌﾟﾛﾊﾟﾃｨにしている場合、そのｸﾗｽの Dispose()も行います。 3.1.3.5 と同様です。そちらを参照ください。

11．７．３．１　add_prjid()

【構文】

public void add_prjid(string prjid)

【引数】

prjid

プロセスジョブ ID です。

【戻り値】

なし。

【説明】

prjid のプロセスジョブ ID を list 配列に追加します。

その後、count+1 します。

11．７．３．２　send_s16f17(), send()

S16F17 メッセージの送信要求をします。

【構文】

```
public int send_S16F17(ref DshS16MultiPrjRsp rsp_info,
                                        DshCallback.callback_s16f17 callback, uint upara)
public int send(ref DshS16MultiPrjRsp rsp_info,
                                        DshCallback.callback_s16f17 callback, uint upara)
```

【引数】

rsp_info

S16F18 応答メッセージに含まれる情報を保存します。コールバック関数の引数になります。

callback

S16F17 送信後、2 次メッセージ受信した時に呼び出される callback 関数です。

upara

ユーザパラメータ情報です。コールバックされる際に callback 関数の引数として渡されます。要求とコールバック間のタグ情報として使用できます。

【戻り値】

返却値	意　味
0	要求が受け入れられた。
(-1)	要求が受け入れられなかった。または id が管理外であった。

【説明】

当該インスタンスの list 配列に含まれる 1 個以上のプロセスジョブ ID から S16F17 メッセージを生成し、それを相手装置に送信するようエンジンに要求します。

要求がエンジンによって受け入れられたときは、0 を返却します。受け入れられなかった場合は(-1)を返却します。
送信後、受信した応答メッセージ情報を中に保存した rsp_info を引数にして、callback 関数を呼び出します。
callback 関数からは 0 を返却してください。

【callback の構文】

```
public delegate int callback_s16f17(
    int eqid,                          // 装置 ID
    int end_status,                    // 終了状態ｺｰﾄﾞ
    ref DshS16MultiPrjRsp rsp_info,    // S16F18 に含まれる応答です。 Vol-1 13.6 参照
    uint upara                         // ﾕｰｻﾞﾊﾟﾗﾒｰﾀ(送信要求ﾒｿｯﾄﾞで指定された upara)
)
```

end_status に返却される値は、=0 の場合は正常終了を意味します。

end_status	意　味
0	送受信が正常に完了した。
-14	T3 ﾀｲﾑｱｳﾄを検出し、応答ﾒｯｾｰｼﾞを受信できなかった。
(-1)	送信できなかった。

11. 7. 3. 3 send_wait()

S16F17 メッセージの送信要求をし,引き続き応答メッセージも受信します。

【構文】

public int send_wait(ref DshS16MultiPrjRsp rsp_info)

【引数】

rsp_info

S16F18 応答メッセージに含まれる情報を保存します。

【戻り値】

返却値	意　味
0	正常に送受信した。
< 0	送受信ｴﾗｰが発生した。(送信ｴﾗｰ or T3 ﾀｲﾑｱｳﾄなど)

【説明】

当該インスタンスの list 配列に含まれる 1 個以上のプロセスジョブ ID から S16F17 メッセージを生成し、それを相手装置に送信するようエンジンに要求します。

本メソッドは送信の後、引き続き S16F18 応答メッセージを待機し、受信が終了したら、上の【戻り値】に示した値を返却します。ei=0 の場合は、受信した応答メッセージに含まれる応答情報が rsp_info に設定され渡されます。

応答メッセージを受信するまでプログラムはブロック（待ち）状態になります。

送信エラー、受信エラーを検出した場合は、ei < 0 の値を返却します。(T3 ﾀｲﾑｱｳﾄも含む)

T3 タイムアウトが発生した場合、DSHDR2 HSMS ドライバーに設定されている T3 プロトコルタイムアウト時間だけブロックされることになります。

11. 8 DshS16F18Response クラス

S16F17 メッセージを受信した後、S16F18 応答メッセージを送信するためのクラスです。

11. 8. 1 コンストラクタ

オーバーロードの一覧を示します。

	名前	説明
1	public DshS16F18Response()	装置 ID=0 のｲﾝｽﾀﾝｽを生成します。
2	public DshS16F18Response(int eqid)	装置 ID を指定してｲﾝｽﾀﾝｽを生成します。

11. 8. 2 プロパティ

プロパティ一覧表に示します。

	名前	説明
1	public int eqid	装置 ID です。ﾃﾞﾌｫﾙﾄ値=0 です。

11．８．３　メソッド

本クラスのメソッドは次の通りです。

	名前	説明
1	public int response()	S16F18 応答ﾒｯｾｰｼﾞを送信します。
2	public void Dispose()	ｸﾗｽ内部で使用された資源(ﾒﾓﾘ)ｼｽﾃﾑに返却します。 他のｸﾗｽをﾌﾟﾛﾊﾟﾃｨにしている場合、そのｸﾗｽの Dispose()も行います。 3.1.3.5 と同様です。そちらを参照ください。

11．８．３．１　response()

S16F18 メッセージの応答送信要求をします。

【構文】

public int response(uint trid, ref DshS16MultiPrjRsp rsp_class)

【引数】

trid

エンジンからポーリング時に与えられたトランザクション ID です。
処理した１次メッセージとの対応を取ります。

rsp_class

S16F18 メッセージに含める応答情報のインスタンスです。
DshS16MultiPrjRsp クラスについては Vol-1 の 13.6 を参照してください。

【戻り値】

返却値	意　味
0	要求が受け入れられた。
(-1)	要求が受け入れられなかった。(ﾄﾗﾝｻﾞｸｼｮﾝ ID が正しくなかった。)

【説明】

受信した S16F17 に対する S16F18 メッセージを送信します。
DshS16MultiPrjRsp クラスのインスタンス rsp_class 内の応答情報から S16F18 を生成します。
trid は、この応答メッセージに対する S16F17 受信時にエンジンから与えられたトランザクション ID です。
trid がエンジンの管理の中に見つからなかった場合には、(-1)を返却します。
要求が受け入れられたら 0 を返却します。この後、応答メッセージ送信結果は特に通知されません。

11. 9 DshS16F19Send クラス

S16F19 メッセージを送信し、S16S20 応答メッセージを受信するためのクラスです。
相手装置から現存するプロセスジョブ ID と状態を取得します。

11. 9. 1 コンストラクタ

オーバーロードの一覧を示します。

	名前	説明
1	public DshS16F19Send()	装置 ID=0 の空のｲﾝｽﾀﾝｽを生成します。
2	public DshS16F19Send(int eqid)	装置 ID を指定してｲﾝｽﾀﾝｽを生成します。

11. 9. 2 プロパティ

プロパティ一覧表に示します。

	名前	説明
1	public int eqid	装置 ID です。ﾃﾞﾌｫﾙﾄ値=0 です。

11. 9. 3 メソッド

本クラスのメソッドは次の通りです。

	名前	説明
1	public int send_s16f19() public int send()	S16F19 ﾒｯｾｰｼﾞを送信します。
2	public int send_wait()	S16F19 ﾒｯｾｰｼﾞを送信し、S16F20 を受信します。 ﾌﾟﾛｸﾞﾗﾑは応答受信までﾌﾞﾛｯｸされます。
3	public void Dispose()	ｸﾗｽ内部で使用された資源(ﾒﾓﾘ)ｼｽﾃﾑに返却します。 他のｸﾗｽをﾌﾟﾛﾊﾟﾃｨにしている場合、そのｸﾗｽの Dispose()も行います。 3.1.3.5 と同様です。そちらを参照ください。

11．９．３．１　send_s16f19(), send()

S16F19 メッセージの送信要求をします。

【構文】

```
public int send_S16F19(ref DshPrjStateList rsp_info,
                                    DshCallback.callback_s16f19 callback, uint upara)
public int send(ref DshPrjStateList rsp_info,
                                    DshCallback.callback_s16f19 callback, uint upara)
```

【引数】

rsp_info

S16S20 応答メッセージに含まれる情報を保存します。コールバック関数の引数になります。

callback

S16F19 送信後、2 次メッセージ受信した時に呼び出される callback 関数です。

upara

ユーザパラメータ情報です。コールバックされる際に callback 関数の引数として渡されます。要求とコールバック間のタグ情報として使用できます。

【戻り値】

返却値	意　味
0	要求が受け入れられた。
(-1)	要求が受け入れられなかった。または id が管理外であった。

【説明】

S16F19 メッセージを生成し、それを相手装置に送信するようエンジンに要求します。

要求がエンジンによって受け入れられたときは、0 を返却します。受け入れられなかった場合は(-1)を返却します。
送信後、受信した応答メッセージ情報を中に保存した rsp_info を引数にして、callback 関数を呼び出します。
callback 関数からは 0 を返却してください。

【callback の構文】

```
public delegate int callback_s16f19(
    int eqid,                          // 装置 ID
    int end_status,                    // 終了状態ｺｰﾄﾞ
    ref DshPrjStateList rsp_info,      // S16S20 に含まれる応答です。 Vol-1 13.7 参照
    uint upara                         // ﾕｰｻﾞﾊﾟﾗﾒｰﾀ(送信要求ﾒｿｯﾄﾞで指定された upara)
)
```

end_status に返却される値は、=0 の場合は正常終了を意味します。

end_status	意　味
0	送受信が正常に完了した。
-14	T3 ﾀｲﾑｱｳﾄを検出し、応答ﾒｯｾｰｼﾞを受信できなかった。
(-1)	送信できなかった。

11. ９. ３. ２　send_wait()

S16F19 メッセージの送信要求をし,引き続き応答メッセージも受信します。

【構文】

public int send_wait(ref DshPrjStateList rsp_info)

【引数】

rsp_info

S16F20 応答メッセージに含まれる情報を保存します。

【戻り値】

返却値	意　味
0	正常に送受信した。
＜ 0	送受信ｴﾗｰが発生した。(送信ｴﾗｰ or T3 ﾀｲﾑｱｳﾄなど)

【説明】

S16F19 メッセージを生成し、それを相手装置に送信するようエンジンに要求します。

本メソッドは送信の後、引き続き S16F20 応答メッセージを待機し、受信が終了したら、上の【戻り値】に示した値を返却します。ei=0 の場合は、受信した応答メッセージに含まれる応答情報が rsp_info に設定され渡されます。

応答メッセージを受信するまでプログラムはブロック（待ち）状態になります。

送信エラー、受信エラーを検出した場合は、ei ＜ 0 の値を返却します。(T3 ﾀｲﾑｱｳﾄも含む)

T3 タイムアウトが発生した場合、DSHDR2 HSMS ドライバーに設定されている T3 プロトコルタイムアウト時間だけブロックされることになります。

11. 10 DshS16F21Send クラス

S16F21 メッセージを送信し、S16S20 応答メッセージを受信するためのクラスです。
相手装置からプロセスジョブのための生成可能スペース数を取得します。

11. 10. 1 コンストラクタ

オーバーロードの一覧を示します。

	名前	説明
1	public DshS16F21Send()	装置 ID=0 の空のｲﾝｽﾀﾝｽを生成します。
2	public DshS16F21Send(int eqid)	装置 ID を指定してｲﾝｽﾀﾝｽを生成します。

11. 10. 2 プロパティ

プロパティ一覧表に示します。

	名前	説明
1	public int eqid	装置 ID です。ﾃﾞﾌｫﾙﾄ値=0 です。

11. 10. 3 メソッド

本クラスのメソッドは次の通りです。

	名前	説明
1	public int send_s16f21() public int send()	S16F21 ﾒｯｾｰｼﾞを送信します。
2	public int send_wait()	S16F21 ﾒｯｾｰｼﾞを送信し、S16F22 を受信します。 ﾌﾟﾛｸﾞﾗﾑは応答受信までﾌﾞﾛｯｸされます。
3	public void Dispose()	ｸﾗｽ内部で使用された資源(ﾒﾓﾘ)ｼｽﾃﾑに返却します。 他のｸﾗｽをﾌﾟﾛﾊﾟﾃｨにしている場合、そのｸﾗｽの Dispose()も行います。 3.1.3.5 と同様です。そちらを参照ください。

11. 10. 3. 1 send_s16f21(), send()

S16F21 メッセージの送信要求をします。

【構文】

```
public int send_S16F21(ref int prjspace, DshCallback.callback_s16f21 callback, uint upara)
public int send(ref int prjspace, DshCallback.callback_s16f21 callback, uint upara)
```

【引数】

prjspace

S16S20 応答メッセージに含まれるスペース数を保存します。コールバック関数の引数になります。

callback

S16F21 送信後、2 次メッセージ受信した時に呼び出される callback 関数です。

upara

ユーザパラメータ情報です。コールバックされる際に callback 関数の引数として渡されます。
要求とコールバック間のタグ情報として使用できます。

【戻り値】

返却値	意　味
0	要求が受け入れられた。
(-1)	要求が受け入れられなかった。または id が管理外であった。

【説明】

S16F21 メッセージを生成し、それを相手装置に送信するようエンジンに要求します。

要求がエンジンによって受け入れられたときは、0 を返却します。受け入れられなかった場合は(-1)を返却します。
送信後、受信した応答メッセージ情報を中に保存した rsp_info を引数にして、callback 関数を呼び出します。
callback 関数からは 0 を返却してください。

【callback の構文】

```
public delegate int callback_s16f19(
    int eqid,                       // 装置 ID
    int end_status,                 // 終了状態ｺｰﾄﾞ
    ref int prjspace,               // S16S20 に含まれるﾌﾟﾛｾｽｼﾞｮﾌﾞの生成可能ｽﾍﾟｰｽ数です。
    uint upara                      // ﾕｰｻﾞﾊﾟﾗﾒｰﾀ(送信要求ﾒｿｯﾄﾞで指定された upara)
)
```

end_status に返却される値は、=0 の場合は正常終了を意味します。

end_status	意　味
0	送受信が正常に完了した。
-14	T3 ﾀｲﾑｱｳﾄを検出し、応答ﾒｯｾｰｼﾞを受信できなかった。
(-1)	送信できなかった。

11. 10. 3. 2 send_wait()

S16F21 メッセージの送信要求をし,引き続き応答メッセージも受信します。

【構文】

public int send_wait(ref int prjspace)

【引数】

prjspace

S16S20 応答メッセージに含まれるスペース数を保存します。

【戻り値】

返却値	意　味
0	正常に送受信した。
＜ 0	送受信ｴﾗｰが発生した。（送信ｴﾗｰ or T3 ﾀｲﾑｱｳﾄなど）

【説明】

S16F21 メッセージを生成し、それを相手装置に送信するようエンジンに要求します。

本メソッドは送信の後、引き続き S16F22 応答メッセージを待機し、受信が終了したら、上の【戻り値】に示した値を返却します。ei=0 の場合は、受信した応答メッセージに含まれる PRJ スペース数が prjspace に設定され渡されます。

応答メッセージを受信するまでプログラムはブロック（待ち）状態になります。

送信エラー、受信エラーを検出した場合は、ei ＜ 0 の値を返却します。(T3 ﾀｲﾑｱｳﾄも含む)

T3 タイムアウトが発生した場合、DSHDR2 HSMS ドライバーに設定されている T3 プロトコルタイムアウト時間だけブロックされることになります。

12．コントロール・ジョブ関連メッセージの送受信

以下のメッセージとクラスがあります。

	メッセージ	クラス名	機能概略
1	S14F9,S14F10	DshS14F9Send	S14F9 送信　Create Object Request
		DshS14F10Response	S14F10 応答
2	S14F11,S14F12	DshS14F11Send	S14F11 送信　Delete Object Request
		DshS14F12Response	S14F12 応答
3	S16F27,S16F28	DshS16F27Send	S16F27 送信　Control Job Command Request
		DshS16F28Response	S16F28 応答

12. 1 DshS14F9Send クラス

S14F9 メッセージを送信し、S14F10 応答メッセージを受信するためのクラスです。
コントロールジョブ生成情報の送信を行います。

12. 1. 1 コンストラクタ

オーバーロードの一覧を示します。

	名前	説明
1	public DshS14F9Send()	装置 ID=0 の空のｲﾝｽﾀﾝｽを生成します。
2	public DshS14F9Send(ref DshCj cj_class)	DshCj ｸﾗｽのｲﾝｽﾀﾝｽを指定して装置 ID=0 のｲﾝｽﾀﾝｽを生成します。
3	public DshS14F9Send(int eqid)	装置 ID を指定してｲﾝｽﾀﾝｽを生成します。
4	public DshS14F9Send(int eqid, ref DshCj cj_class)	装置 ID, DshCj ｸﾗｽのｲﾝｽﾀﾝｽを指定してｲﾝｽﾀﾝｽを生成します。

12. 1. 2 プロパティ

プロパティ一覧表に示します。

	名前	説明
1	public int eqid	装置 ID です。ﾃﾞﾌｫﾙﾄ値=0 です。
2	public DshCj cj_class	ｺﾝﾄﾛｰﾙｼﾞｮﾌﾞ情報を保存するための DshCj ｸﾗｽのｲﾝｽﾀﾝｽです。Vol-1 14-1 参照

12. 1. 3 メソッド

本クラスのメソッドは次の通りです。

	名前	説明
1	public void set_info()	cj_class に DshCj ｸﾗｽのｲﾝｽﾀﾝｽを設定します。
2	public int send_s14f9() public int send()	S14F9 ﾒｯｾｰｼﾞを送信します。
3	public int send_wait()	S14F9 ﾒｯｾｰｼﾞを送信し、S14F10 を受信します。 ﾌﾟﾛｸﾞﾗﾑは応答受信までﾌﾞﾛｯｸされます。
4	public void Dispose()	ｸﾗｽ内部で使用された資源(ﾒﾓﾘ)ｼｽﾃﾑに返却します。 他のｸﾗｽをﾌﾟﾛﾊﾟﾃｨにしている場合、そのｸﾗｽの Dispose()も行います。 3.1.3.5 と同様です。そちらを参照ください。

12. 1. 3. 1 set_info()

【構文】

public void set_info(ref DshCj info)

【引数】

info

設定したい DshCj クラスのインスタンスです。

【戻り値】

なし。

【説明】

info に与えられた DshCj クラスのインスタンスの内容を cj_class に設定します。
実際にはコピーします。

12. 1. 3. 2 send_s14f9(), send

S14F9 メッセージの送信要求をします。

【構文】

```
public int send_s14f9(string cjid,
                      ref DshS14Rsp rsp_info, DshCallback.callback_s14f9 callback, uint upara)
public int send(string cjid,
                      ref DshS14Rsp rsp_info, DshCallback.callback_s14f9 callback, uint upara)
public int send_s14f9(ref DshCj cj_class,
                      ref DshS14Rsp rsp_info, DshCallback.callback_s14f9 callback, uint upara)
public int send(ref DshCj cj_class,
                      ref DshS14Rsp rsp_info, DshCallback.callback_s14f9 callback, uint upara)
public int send_s14f9(ref DshS14Rsp rsp_info, DshCallback.callback_s14f9 callback, uint upara)
public int send(ref DshS14Rsp rsp_info, DshCallback.callback_s14f9 callback, uint upara)
```

【引数】

cjid
　　コントロールジョブ ID を指定します。エンジンから情報を cj_class に取得します。

cj_class
　　プロパティの cj_class にコピーします。

rsp_info
　　S14F10 応答メッセージに含まれる情報を保存します。コールバック関数の引数になります。

callback
　　S14F9 送信後、2 次メッセージ受信した時に呼び出される callback 関数です。

upara
　　ユーザパラメータ情報です。コールバックされる際に callback 関数の引数として渡されます。
　　要求とコールバック間のタグ情報として使用できます。

【戻り値】

返却値	意　味
0	要求が受け入れられた。
(-1)	要求が受け入れられなかった。または cjid が管理外であった。

【説明】

当該インスタンスの cj_class から S14F9 メッセージを生成し、それを相手装置に送信するようエンジンに要求します。

引数にコントロールジョブ ID が指定されていた場合にはそのコントロールジョブ情報をエンジンの管理から cj_class プロパティに取得した上で送信処理をします。
また、DshCj クラスのインスタンスが引数に与えられた場合には、それを cj_class プロパティにコピーした上で送信処理を行います。

要求がエンジンによって受け入れられたときは、0 を返却します。受け入れられなかった場合は(-1)を返却します。
送信後、受信した応答メッセージ情報を中に保存した rsp_info を引数にして、callback 関数を呼び出します。
callback 関数からは 0 を返却してください。

【callback の構文】

```
public delegate int callback_s14f9(
    int eqid,                          // 装置 ID
    int end_status,                    // 終了状態ｺｰﾄﾞ
    ref DshS14Rsp rsp_info,            // S14F10 に含まれる応答です。 Vol-1 14.14 参照
    uint upara                         // ﾕｰｻﾞﾊﾟﾗﾒｰﾀ(送信要求ﾒｿｯﾄﾞで指定された upara)
)
```

end_status に返却される値は、=0 の場合は正常終了を意味します。

end_status	意　味
0	送受信が正常に完了した。
-14	T3 ﾀｲﾑｱｳﾄを検出し、応答ﾒｯｾｰｼﾞを受信できなかった。
(-1)	送信できなかった。

12. 1. 3. 3 send_wait()

S14F9 メッセージの送信要求をし,引き続き応答メッセージも受信します。

【構文】

```
public int send_wait(string cjid, ref DshS14Rsp rsp_info)
public int send_wait(ref DshCj cj_class, ref DshS14Rsp rsp_info)
public int send_wait(ref DshS14Rsp rsp_info)
```

【引数】

cjid
　　コントロールジョブ ID を指定します。エンジンから情報を cj_class に取得します。

cj_class
　　プロパティの cj_class にコピーします。

rsp_info
　　S14F10 応答メッセージに含まれる情報を保存します。

【戻り値】

返却値	意　味
0	正常に送受信した。
＜ 0	送受信ｴﾗｰが発生した。(送信ｴﾗｰ or T3 ﾀｲﾑｱｳﾄなど)

【説明】

当該インスタンスの cj_class から S14F9 メッセージを生成し、それを相手装置に送信するようエンジンに要求します。

引数にコントロールジョブ ID が指定されていた場合にはそのコントロールジョブ情報をエンジンの管理から cj_class プロパティに取得した上で送信処理をします。
また、DshCj クラスのインスタンスが引数に与えられた場合には、それを cj_class プロパティにコピーした上で送信処理を行います。

本メソッドは送信の後、引き続き S14F10 応答メッセージを待機し、受信が終了したら、上の【戻り値】に示した値を返却します。ei=0 の場合は、受信した応答メッセージ情報が rsp_info に設定され渡されます。

応答メッセージを受信するまでプログラムはブロック（待ち）状態になります。

送信エラー、受信エラーを検出した場合は、ei ＜ 0 の値を返却します。(T3 ﾀｲﾑｱｳﾄも含む)

T3 タイムアウトが発生した場合、DSHDR2 HSMS ドライバーに設定されている T3 プロトコルタイムアウト時間だけブロックされることになります。

12. 2 DshS14F10Response クラス

S14F9 メッセージを受信した後、S14F10 応答メッセージを送信するためのクラスです。

12. 2. 1 コンストラクタ

オーバーロードの一覧を示します。

	名前	説明
1	public DshS14F10Response()	装置 ID=0 のｲﾝｽﾀﾝｽを生成します。
2	public DshS14F10Response(int eqid)	装置 ID を指定してｲﾝｽﾀﾝｽを生成します。

12. 2. 2 プロパティ

プロパティ一覧表に示します。

	名前	説明
1	public int eqid	装置 ID です。ﾃﾞﾌｫﾙﾄ値=0 です。

12. 2. 3 メソッド

本クラスのメソッドは次の通りです。

	名前	説明
1	public int response()	S14F10 応答ﾒｯｾｰｼﾞを送信します。
2	public void Dispose()	ｸﾗｽ内部で使用された資源(ﾒﾓﾘ)ｼｽﾃﾑに返却します。 他のｸﾗｽをﾌﾟﾛﾊﾟﾃｨにしている場合、そのｸﾗｽの Dispose()も行います。 3.1.3.5 と同様です。そちらを参照ください。

12. 2. 3. 1 response()

S14F10 メッセージの応答送信要求をします。

【構文】

public int response(uint trid, ref DshS14Rsp rsp_class)

【引数】

trid

エンジンからポーリング時に与えられたトランザクション ID です。
処理した１次メッセージとの対応を取ります。

rsp_class

S14F10 メッセージに含める応答情報のインスタンスです。
DshS14Rsp クラスについては Vol-1 の 12.5 を参照してください。

【戻り値】

返却値	意　味
0	要求が受け入れられた。
(-1)	要求が受け入れられなかった。(ﾄﾗﾝｻﾞｸｼｮﾝ ID が正しくなかった。)

【説明】

受信した S14F9 に対する S14F10 メッセージを送信します。
DshS16Rsp クラスのインスタンス rsp_class 内の応答情報から S14F10 を生成します。
trid は、この応答メッセージに対する S14F9 受信時にエンジンから与えられたトランザクション ID です。
trid がエンジンの管理の中に見つからなかった場合には、(-1)を返却します。
要求が受け入れられたら 0 を返却します。この後、応答メッセージ送信結果は特に通知されません。

12. 3 DshS14F11Send クラス

S14F11 メッセージを送信し、S14F12 応答メッセージを受信するためのクラスです。
コントロールジョブ削除情報の送信を行います。

12. 3. 1 コンストラクタ

オーバーロードの一覧を示します。

	名前	説明
1	public DshS14F11Send()	装置 ID=0 の空のｲﾝｽﾀﾝｽを生成します。
2	public DshS14F11Send(ref DshCj cj_class)	DshCj ｸﾗｽのｲﾝｽﾀﾝｽを指定して装置 ID=0 のｲﾝｽﾀﾝｽを生成します。
3	public DshS14F11Send(int eqid)	装置 ID を指定してｲﾝｽﾀﾝｽを生成します。
4	public DshS14F11Send(int eqid, ref DshCj cj_class)	装置 ID, DshCj ｸﾗｽのｲﾝｽﾀﾝｽを指定してｲﾝｽﾀﾝｽを生成します。

12. 3. 2 プロパティ

プロパティ一覧表に示します。

	名前	説明
1	public int eqid	装置 ID です。ﾃﾞﾌｫﾙﾄ値=0 です。
2	public DshCj cj_class	ｺﾝﾄﾛｰﾙｼﾞｮﾌﾞ情報を保存するための DshCj ｸﾗｽのｲﾝｽﾀﾝｽです。Vol-1 14-1 参照

12. 3. 3 メソッド

本クラスのメソッドは次の通りです。

	名前	説明
1	public void set_info()	cj_class に DshCj ｸﾗｽのｲﾝｽﾀﾝｽを設定します。
2	public int send_s14f11() public int send()	S14F11 ﾒｯｾｰｼﾞを送信します。
3	public int send_wait()	S14F11 ﾒｯｾｰｼﾞを送信し、S14F12 を受信します。 ﾌﾟﾛｸﾞﾗﾑは応答受信までﾌﾞﾛｯｸされます。
4	public void Dispose()	ｸﾗｽ内部で使用された資源(ﾒﾓﾘ)ｼｽﾃﾑに返却します。 他のｸﾗｽをﾌﾟﾛﾊﾟﾃｨにしている場合、そのｸﾗｽの Dispose()も行います。 3.1.3.5 と同様です。そちらを参照ください。

12. 3. 3. 1 set_info()

【構文】

public void set_info(ref DshCj info)

【引数】

info

設定したい DshCj クラスのインスタンスです。

【戻り値】

なし。

【説明】

info に与えられた DshCj クラスのインスタンスの内容を cj_class に設定します。
実際にはコピーします。

12. 3. 3. 2 send_s14f11(), send()

S14F11 メッセージの送信要求をします。

【構文】

```
public int send_s14f11(string cjid,
                       ref DshS14Rsp rsp_info, DshCallback.callback_s14f11 callback, uint upara)
public int send(string cjid,
                       ref DshS14Rsp rsp_info, DshCallback.callback_s14f11 callback, uint upara)
public int send_s14f11(ref DshCj cj_class,
                       ref DshS14Rsp rsp_info, DshCallback.callback_s14f11 callback, uint upara)
public int send(ref DshCj cj_class,
                       ref DshS14Rsp rsp_info, DshCallback.callback_s14f11 callback, uint upara)
public int send_s14f11(ref DshS14Rsp rsp_info, DshCallback.callback_s14f11 callback, uint upara)
public int send (ref DshS14Rsp rsp_info, DshCallback.callback_s14f11 callback, uint upara)
```

【引数】

cjid
　　コントロールジョブ ID を指定します。エンジンから情報を cj_class に取得します。

cj_class
　　プロパティの cj_class にコピーします。

rsp_info
　　S14F12 応答メッセージに含まれる情報を保存します。コールバック関数の引数になります。

callback
　　S14F11 送信後、2 次メッセージ受信した時に呼び出される callback 関数です。

upara
　　ユーザパラメータ情報です。コールバックされる際に callback 関数の引数として渡されます。
　　要求とコールバック間のタグ情報として使用できます。

【戻り値】

返却値	意　味
0	要求が受け入れられた。
(-1)	要求が受け入れられなかった。または cjid が管理外であった。

【説明】

当該インスタンスの cj_class から S14F11 メッセージを生成し、それを相手装置に送信するようエンジンに要求します。

コントロールジョブ ID が引数に指定されていた場合にはそのコントロールジョブ情報をエンジンの管理から cj_class プロパティに取得した上で送信処理をします。
また、DshCj クラスのインスタンスが引数に与えられた場合には、それを cj_class プロパティにコピーした上で送信処理を行います。

要求がエンジンによって受け入れられたときは、0 を返却します。受け入れられなかった場合は(-1)を返却します。
送信後、受信した応答メッセージ情報を中に保存した rsp_info を引数にして、callback 関数を呼び出します。
callback 関数からは 0 を返却してください。

【callback の構文】

```
public delegate int callback_s14f11(
    int eqid,                          // 装置 ID
    int end_status,                    // 終了状態ｺｰﾄﾞ
    ref DshS14Rsp rsp_info,            // S14F12 に含まれる応答です。 Vol-1 14.14 参照
    uint upara                         // ﾕｰｻﾞﾊﾟﾗﾒｰﾀ(送信要求ﾒｿｯﾄﾞで指定された upara)
)
```

end_status に返却される値は、=0 の場合は正常終了を意味します。

end_status	意　味
0	送受信が正常に完了した。
-14	T3 ﾀｲﾑｱｳﾄを検出し、応答ﾒｯｾｰｼﾞを受信できなかった。
(-1)	送信できなかった。

12. 3. 3. 3 send_wait()

S14F11 メッセージの送信要求をし,引き続き応答メッセージも受信します。

【構文】

```
public int send_wait(string cjid, ref DshS14Rsp rsp_info)
public int send_wait(ref DshCj cj_class, ref DshS14Rsp rsp_info)
public int send_wait(ref DshS14Rsp rsp_info)
```

【引数】

cjid
　　コントロールジョブ ID を指定します。エンジンから情報を cj_class に取得します。

cj_class
　　プロパティの cj_class にコピーします。

rsp_info
　　S14F12 応答メッセージに含まれる情報を保存します。

【戻り値】

返却値	意　味
0	正常に送受信した。
＜ 0	送受信ｴﾗｰが発生した。(送信ｴﾗｰ or T3 ﾀｲﾑｱｳﾄなど)

【説明】

当該インスタンスの cj_class から S14F11 メッセージを生成し、それを相手装置に送信するようエンジンに要求します。

引数にコントロールジョブ ID が指定されていた場合にはそのコントロールジョブ情報をエンジンの管理から cj_class プロパティに取得した上で送信処理をします。
また、DshCj クラスのインスタンスが引数に与えられた場合には、それを cj_class プロパティにコピーした上で送信処理を行います。

本メソッドは送信の後、引き続き S14F12 応答メッセージを待機し、受信が終了したら、上の【戻り値】に示した値を返却します。ei=0 の場合は、受信した応答メッセージ情報が rsp_info に設定され渡されます。

応答メッセージを受信するまでプログラムはブロック（待ち）状態になります。

送信エラー、受信エラーを検出した場合は、ei ＜ 0 の値を返却します。(T3 ﾀｲﾑｱｳﾄも含む)

T3 タイムアウトが発生した場合、DSHDR2 HSMS ドライバーに設定されている T3 プロトコルタイムアウト時間だけブロックされることになります。

12. 4 DshS14F12Response クラス

S14F11 メッセージを受信した後、S14F12 応答メッセージを送信するためのクラスです。

12. 4. 1 コンストラクタ

オーバーロードの一覧を示します。

	名前	説明
1	public DshS14F12Response()	装置 ID=0 のｲﾝｽﾀﾝｽを生成します。
2	public DshS14F12Response(int eqid)	装置 ID を指定してｲﾝｽﾀﾝｽを生成します。

12. 4. 2 プロパティ

プロパティ一覧表に示します。

	名前	説明
1	public int eqid	装置 ID です。ﾃﾞﾌｫﾙﾄ値=0 です。

12．４．３　メソッド

本クラスのメソッドは次の通りです。

	名前	説明
1	public int response()	S14F12 応答ﾒｯｾｰｼﾞを送信します。
2	public void Dispose()	ｸﾗｽ内部で使用された資源(ﾒﾓﾘ)ｼｽﾃﾑに返却します。 他のｸﾗｽをﾌﾟﾛﾊﾟﾃｨにしている場合、そのｸﾗｽの Dispose()も行います。 3.1.3.5 と同様です。そちらを参照ください。

12．４．３．１　response()

S14F12 メッセージの応答送信要求をします。

【構文】

public int response(uint trid, ref DshS14Rsp rsp_class)

【引数】

trid

エンジンからポーリング時に与えられたトランザクション ID です。
処理した１次メッセージとの対応を取ります。

rsp_class

S14F12 メッセージに含める応答情報のインスタンスです。
DshS14Rsp クラスについては Vol-1 の 12.5 を参照してください。

【戻り値】

返却値	意　味
0	要求が受け入れられた。
(-1)	要求が受け入れられなかった。(ﾄﾗﾝｻﾞｸｼｮﾝ ID が正しくなかった。)

【説明】

受信した S14F11 に対する S14F12 メッセージを送信します。
DshS16Rsp クラスのインスタンス rsp_class 内の応答情報から S14F12 を生成します。
trid は、この応答メッセージに対する S14F11 受信時にエンジンから与えられたトランザクション ID です。
trid がエンジンの管理の中に見つからなかった場合には、(-1)を返却します。
要求が受け入れられたら 0 を返却します。この後、応答メッセージ送信結果は特に通知されません。

12. 5 DshS16F27Send クラス

S16F27 メッセージを送信し、S16F28 応答メッセージを受信するためのクラスです。
コントロールジョブのコマンド要求を行います。

本クラスは DshCjCmd クラスを継承します。従って、送信するコマンド情報は DshCjCmd のプロパティに設定するか、DshCjCmd クラスのメソッドを使用して設定します。

12. 5. 1 コンストラクタ

オーバーロードの一覧を示します。

	名前	説明
1	public DshS16F27Send()	装置 ID=0 の空のｲﾝｽﾀﾝｽを生成します。
2	public DshS16F27Send(int eqid)	装置 ID を指定してｲﾝｽﾀﾝｽを生成します。

12. 5. 2 プロパティ

プロパティ一覧表に示します。

	名前	説明
1	public int eqid	装置 ID です。ﾃﾞﾌｫﾙﾄ値=0 です。
2	(DshCjCmd のﾌﾟﾛﾊﾟﾃｨを継承します。)	ｺﾝﾄﾛｰﾙｼﾞｮﾌﾞｺﾏﾝﾄﾞ情報を保存するための DshCjCmd ｸﾗｽのﾌﾟﾛﾊﾟﾃｨです。Vol-1 14-2 参照

12．５．３　メソッド

本クラスのメソッドは次の通りです。

	名前	説明
1	public int send_s16f27() public int send()	S16F27 ﾒｯｾｰｼﾞを送信します。
2	public int send_wait()	S16F27 ﾒｯｾｰｼﾞを送信し、S16F28 を受信します。 ﾌﾟﾛｸﾞﾗﾑは応答受信までﾌﾞﾛｯｸされます。
3	(DshChCmd ｸﾗｽのﾒｿｯﾄﾞ等)	Vol-1 14-2 参照
4	public void Dispose()	ｸﾗｽ内部で使用された資源(ﾒﾓﾘ)ｼｽﾃﾑに返却します。 他のｸﾗｽをﾌﾟﾛﾊﾟﾃｨにしている場合、そのｸﾗｽの Dispose()も行います。 3.1.3.5 と同様です。そちらを参照ください。

12．５．３．１　send_s16f27(), send()

S16F27 メッセージの送信要求をします。

【構文】

```
public int send_S16F27( ref DshS16F27Rsp rsp_info,
                                    DshCallback.callback_s16f27 callback, uint upara)
public int send( ref DshS16F27Rsp rsp_info,
                                    DshCallback.callback_s16f27 callback, uint upara)
```

【引数】

rsp_info

S16F28 応答メッセージに含まれる情報を保存します。コールバック関数の引数になります。

callback

S16F27 送信後、2 次メッセージ受信した時に呼び出される callback 関数です。

upara

ユーザパラメータ情報です。コールバックされる際に callback 関数の引数として渡されます。要求とコールバック間のタグ情報として使用できます。

【戻り値】

返却値	意　味
0	要求が受け入れられた。
(-1)	要求が受け入れられなかった。

【説明】

当該インスタンスが継承する DshCjCmd に設定されたプロパティ情報から､S16F27 メッセージを生成し、それを相手装置に送信するようエンジンに要求します。
要求がエンジンによって受け入れられたときは、0 を返却します。受け入れられなかった場合は(-1)を返却します。
送信後、受信した応答メッセージ情報を中に保存した rsp_info を引数にして、callback 関数を呼び出します。
callback 関数からは 0 を返却してください。

【callback の構文】

```
public delegate int callback_s16f27(
    int eqid,                       // 装置 ID
    int end_status,                 // 終了状態ｺｰﾄﾞ
    ref DshS16F27Rsp rsp_info,      // S16F28 に含まれる応答です。 Vol-1 14.15 参照
    uint upara                      // ﾕｰｻﾞﾊﾟﾗﾒｰﾀ(送信要求ﾒｿｯﾄﾞで指定された upara)
)
```

end_status に返却される値は、=0 の場合は正常終了を意味します。

end_status	意　味
0	送受信が正常に完了した。
-14	T3 ﾀｲﾑｱｳﾄを検出し、応答ﾒｯｾｰｼﾞを受信できなかった。
(-1)	送信できなかった。

12．５．３．２　send_wait()

S16F27 メッセージの送信要求をし,引き続き応答メッセージも受信します。

【構文】

public int send_wait(ref DshS16F27Rsp rsp_info)

【引数】

rsp_info

S16F28 応答メッセージに含まれる情報を保存します。

【戻り値】

返却値	意　味
0	正常に送受信した。
＜ 0	送受信ｴﾗｰが発生した。（送信ｴﾗｰ or T3 ﾀｲﾑｱｳﾄなど）

【説明】

当該インスタンスが継承する DshCjCmd に設定されたプロパティ情報から､S16F27 メッセージを生成し、それを相手装置に送信するようエンジンに要求します。

本メソッドは送信の後、引き続き S16F28 応答メッセージを待機し、受信が終了したら、上の【戻り値】に示した値を返却します。ei=0 の場合は、受信した応答メッセージ情報が rsp_info に設定され渡されます。

応答メッセージを受信するまでプログラムはブロック（待ち）状態になります。

送信エラー、受信エラーを検出した場合は、ei ＜ 0 の値を返却します。(T3 ﾀｲﾑｱｳﾄも含む)

T3 タイムアウトが発生した場合、DSHDR2 HSMS ドライバーに設定されている T3 プロトコルタイムアウト時間だけブロックされることになります。

12. 6 DshS16F28Response クラス

S16F27 メッセージを受信した後、S16F28 応答メッセージを送信するためのクラスです。

12. 6. 1 コンストラクタ

オーバーロードの一覧を示します。

	名前	説明
1	public DshS16F28Response()	装置 ID=0 のｲﾝｽﾀﾝｽを生成します。
2	public DshS16F28Response(int eqid)	装置 ID を指定してｲﾝｽﾀﾝｽを生成します。

12. 6. 2 プロパティ

プロパティ一覧表に示します。

	名前	説明
1	public int eqid	装置 ID です。ﾃﾞﾌｫﾙﾄ値=0 です。

12．６．３　メソッド

本クラスのメソッドは次の通りです。

	名前	説明
1	public int response()	S16F28 応答ﾒｯｾｰｼﾞを送信します。
2	public void Dispose()	ｸﾗｽ内部で使用された資源(ﾒﾓﾘ)ｼｽﾃﾑに返却します。 他のｸﾗｽをﾌﾟﾛﾊﾟﾃｨにしている場合、そのｸﾗｽの Dispose()も行います。 3.1.3.5 と同様です。そちらを参照ください。

12．６．３．１　response()

S16F28 メッセージの応答送信要求をします。

【構文】

public int response(uint trid, ref DshS16F27Rsp rsp_class)

【引数】

trid

エンジンからポーリング時に与えられたトランザクション ID です。
処理した１次メッセージとの対応を取ります。

rsp_class

S16F28 メッセージに含める応答情報のインスタンスです。
DshS16F27Rsp クラスについては Vol-1 の 14.15 を参照してください。

【戻り値】

返却値	意　味
0	要求が受け入れられた。
(-1)	要求が受け入れられなかった。(ﾄﾗﾝｻﾞｸｼｮﾝ ID が正しくなかった。)

【説明】

受信した S16F27 に対する S16F28 メッセージを送信します。
DshS16F27Rsp クラスのインスタンス rsp_class 内の応答情報から S16F28 を生成します。
trid は、この応答メッセージに対する S16F27 受信時にエンジンから与えられたトランザクション ID です。
trid がエンジンの管理の中に見つからなかった場合には、(-1)を返却します。
要求が受け入れられたら 0 を返却します。この後、応答メッセージ送信結果は特に通知されません。

13．スプール関連メッセージの送受信

以下のメッセージとクラスがあります。

	メッセージ	クラス名	機能概略
1	S2F43,44	DshS2F43Send	S2F43 送信　Reset Spooling Stream and Function
		DshS2F44Response	S2F44 応答
2	S6F23,S6F24	DshS6F23Send	S6F23 送信　Request Spooled Data
		－	(S6F24 はｴﾝｼﾞﾝが自動応答)

13. 1 DshS2F43Send クラス

S2F43 メッセージを送信し、S2F44 応答メッセージを受信するためのクラスです。
スプール情報の送信を行います。

13. 1. 1 コンストラクタ

オーバーロードの一覧を示します。

	名前	説明
1	public DshS2F43Send()	装置 ID=0 の空のｲﾝｽﾀﾝｽを生成します。
2	public DshS2F43Send(ref DshSpoolSend info)	ｽﾌﾟｰﾙ情報を指定して装置 ID=0 のｲﾝｽﾀﾝｽを生成します。
3	public DshS2F43Send(int eqid)	装置 ID を指定してｲﾝｽﾀﾝｽを生成します。
4	public DshS2F43Send(int eqid, ref DshSpoolSend info)	装置 ID とｽﾌﾟｰﾙ情報を指定してｲﾝｽﾀﾝｽを生成します。

13. 1. 2 プロパティ

プロパティ一覧表に示します。

	名前	説明
1	public int eqid	装置 ID です。ﾃﾞﾌｫﾙﾄ値=0 です。
2	public DshSpoolSend sp_info	ｽﾌﾟｰﾙ情報が保存される DshSpoolSend ｸﾗｽのｲﾝｽﾀﾝｽです。 DshSpoolSend ｸﾗｽについては、Vol-1 15.4 参照

13．1．3　メソッド

本クラスのメソッドは次の通りです。

	名前	説明
1	public void set_info()	sp_info に情報を設定します。 DshSpoolSend ｸﾗｽのｲﾝｽﾀﾝｽを設定します。
2	public int send_s2f43() public int send()	S2F43 ﾒｯｾｰｼﾞを送信します。
3	public int send_wait()	S2F43 ﾒｯｾｰｼﾞを送信し、S2F44 を受信します。 ﾌﾟﾛｸﾞﾗﾑは応答受信までﾌﾞﾛｯｸされます。
4	public void Dispose()	ｸﾗｽ内部で使用された資源(ﾒﾓﾘ)ｼｽﾃﾑに返却します。 他のｸﾗｽをﾌﾟﾛﾊﾟﾃｨにしている場合、そのｸﾗｽの Dispose()も行います。 3.1.3.5 と同様です。そちらを参照ください。

13．1．3．1　set_info()

【構文】

public void set_info(ref　DshSpoolSend info)

【引数】

info

設定したい DshSpoolSend クラスのインスタンスです。

【戻り値】

なし。

【説明】

info に与えられた DshSpoolSend クラスのインスタンスの内容を sp_info に設定します。
実際にはコピーします。

13. 1. 3. 2 send_s2f43(), send()

S2F43 メッセージの送信要求をします。

【構文】

```
public int send_s2f43(ref DshSpoolRsp rsp_info,
                              DshCallback.callback_s2f43 callback, uint upara)
public int send(ref DshSpoolRsp rsp_info,
                              DshCallback.callback_s2f43 callback, uint upara)
```

【引数】

rsp_info
　　S2F44 応答メッセージに含まれる情報を保存します。コールバック関数の引数になります。
callback
　　S2F43 送信後、2 次メッセージ受信した時に呼び出される callback 関数です。
upara
　　ユーザパラメータ情報です。コールバックされる際に callback 関数の引数として渡されます。
　　要求とコールバック間のタグ情報として使用できます。

【戻り値】

返却値	意　味
0	要求が受け入れられた。
(-1)	要求が受け入れられなかった。または cjid が管理外であった。

【説明】

当該インスタンスのプロパティ sp_info 情報から S2F43 メッセージを生成し、それを相手装置に送信するようエンジンに要求します。

要求がエンジンによって受け入れられたときは、0 を返却します。受け入れられなかった場合は(-1)を返却します。
送信後、受信した応答メッセージ情報を中に保存した rsp_info を引数にして、callback 関数を呼び出します。
callback 関数からは 0 を返却してください。

【callback の構文】

```
public delegate int callback_s2f43(
    int eqid,                          // 装置 ID
    int end_status,                    // 終了状態ｺｰﾄﾞ
    ref DshSpoolRsp rsp_info,          // S2F44 に含まれる応答です。 Vol-1 15.5 参照
    uint upara                         // ﾕｰｻﾞﾊﾟﾗﾒｰﾀ(送信要求ﾒｿｯﾄﾞで指定された upara)
)
```

end_status に返却される値は、=0 の場合は正常終了を意味します。

end_status	意　味
0	送受信が正常に完了した。
-13	T3 ﾀｲﾑｱｳﾄを検出し、応答ﾒｯｾｰｼﾞを受信できなかった。
(-1)	送信できなかった。

13. 1. 3. 3 send_wait()

S2F43 メッセージの送信要求をし,引き続き応答メッセージも受信します。

【構文】

public int send_wait(ref DshSpoolRsp rsp_info)

【引数】

rsp_info

S2F44 応答メッセージに含まれる情報を保存します。

【戻り値】

返却値	意　味
0	正常に送受信した。
< 0	送受信ｴﾗｰが発生した。(送信ｴﾗｰ or T3 ﾀｲﾑｱｳﾄなど)

【説明】

当該インスタンスのプロパティ sp_info 情報から S2F43 メッセージを生成し、それを相手装置に送信するようエンジンに要求します。

本メソッドは送信の後、引き続き S2F44 応答メッセージを待機し、受信が終了したら、上の【戻り値】に示した値を返却します。ei=0 の場合は、受信した応答メッセージ情報が rsp_info に設定され渡されます。

応答メッセージを受信するまでプログラムはブロック（待ち）状態になります。

送信エラー、受信エラーを検出した場合は、ei < 0 の値を返却します。(T3 ﾀｲﾑｱｳﾄも含む)

T3 タイムアウトが発生した場合、DSHDR2 HSMS ドライバーに設定されている T3 プロトコルタイムアウト時間だけブロックされることになります。

13. 2 DshS2F44Response クラス

S2F43 メッセージを受信した後、S2F44 応答メッセージを送信するためのクラスです。

13. 2. 1 コンストラクタ

オーバーロードの一覧を示します。

	名前	説明
1	public DshS2F44Response()	装置 ID=0 のｲﾝｽﾀﾝｽを生成します。
2	public DshS2F44Response(int eqid)	装置 ID を指定してｲﾝｽﾀﾝｽを生成します。

13. 2. 2 プロパティ

プロパティ一覧表に示します。

	名前	説明
1	public int eqid	装置 ID です。ﾃﾞﾌｫﾙﾄ値=0 です。

13. ２. ３　メソッド

本クラスのメソッドは次の通りです。

	名前	説明
1	public int response()	S2F44 応答ﾒｯｾｰｼﾞを送信します。
2	public void Dispose()	ｸﾗｽ内部で使用された資源(ﾒﾓﾘ)ｼｽﾃﾑに返却します。 他のｸﾗｽをﾌﾟﾛﾊﾟﾃｨにしている場合、そのｸﾗｽのDispose()も行います。 3.1.3.5 と同様です。そちらを参照ください。

13. ２. ３. １　response()

S2F44 メッセージの応答送信要求をします。

【構文】

public int response(uint trid, ref DshSpoolRsp rsp_class)

【引数】

trid

エンジンからポーリング時に与えられたトランザクション ID です。
処理した１次メッセージとの対応を取ります。

rsp_class

S2F44 メッセージに含める応答情報のインスタンスです。
DshSpoolRsp クラスについては Vol-1 の 15.5 を参照してください。

【戻り値】

返却値	意　味
0	要求が受け入れられた。
(-1)	要求が受け入れられなかった。(ﾄﾗﾝｻﾞｸｼｮﾝ ID が正しくなかった。)

【説明】

受信した S2F43 に対する S2F44 メッセージを送信します。
DshSpoolRsp クラスのインスタンス rsp_class 内の応答情報から S2F44 を生成します。
trid は、この応答メッセージに対する S2F43 受信時にエンジンから与えられたトランザクション ID です。
trid がエンジンの管理の中に見つからなかった場合には、(-1)を返却します。
要求が受け入れられたら 0 を返却します。この後、応答メッセージ送信結果は特に通知されません。

13. 3 DshS6F23Send クラス

S6F23 メッセージを送信し、S6F24 応答メッセージを受信するためのクラスです。
スプールデータの転送または削除要求の送信を行います。

13. 3. 1 コンストラクタ

オーバーロードの一覧を示します。

	名前	説明
1	public DshS6F23Send()	装置 ID=0 の空のｲﾝｽﾀﾝｽを生成します。
2	public DshS6F23Send(int rsdc)	rsdc を指定して装置 ID=0 のｲﾝｽﾀﾝｽを生成します。 rsdc は、0=転送開始、1=削除を意味します。
3	public DshS6F23Send(int eqid)	装置 ID を指定してｲﾝｽﾀﾝｽを生成します。
4	public DshS6F23Send(int eqid, int rsdc)	装置 ID と rsdc を指定してｲﾝｽﾀﾝｽを生成します。 rsdc は、0=転送開始、1=削除を意味します。

13. 3. 2 プロパティ

プロパティ一覧表に示します。

	名前	説明
1	public int eqid	装置 ID です。ﾃﾞﾌｫﾙﾄ値=0 です。
2	public int rsdc	rsdc は、ｽﾌﾟｰﾙﾃﾞｰﾀに対する指令です。 0=転送開始、1=削除を意味します。

13. 3. 3 メソッド

本クラスのメソッドは次の通りです。

	名前	説明
1	public int send_s6f23() public int send()	S6F23 ﾒｯｾｰｼﾞを送信します。
2	public int send_wait()	S2F23 ﾒｯｾｰｼﾞを送信し、S2F24 を受信します。 ﾌﾟﾛｸﾞﾗﾑは応答受信までﾌﾞﾛｯｸされます。
3	public void Dispose()	ｸﾗｽ内部で使用された資源(ﾒﾓﾘ)ｼｽﾃﾑに返却します。 他のｸﾗｽをﾌﾟﾛﾊﾟﾃｨにしている場合、そのｸﾗｽの Dispose()も行います。 3.1.3.5 と同様です。そちらを参照ください。

13．３．３．１　send_s6f23(), send()

S6F23 メッセージの送信要求をします。

【構文】

```
public int send_s6f23(int rsdc, ref int rsda, DshCallback.callback_s6f23 callback, uint upara)
public int send(int rsdc, ref int rsda, DshCallback.callback_s6f23 callback, uint upara)
public int send_s6f23(ref int rsda, DshCallback.callback_s6f23 callback, uint upara)
public int send(ref int rsda, DshCallback.callback_s6f23 callback, uint upara)
```

【引数】

rsdc
　　スプールデータの転送、削除を指定します。(0=転送、1=削除)

rsda
　　S6F24 応答メッセージに含まれる ack 情報を保存します。コールバック関数の引数になります。

callback
　　S6F23 送信後、2 次メッセージ受信した時に呼び出される callback 関数です。

upara
　　ユーザパラメータ情報です。コールバックされる際に callback 関数の引数として渡されます。
　　要求とコールバック間のタグ情報として使用できます。

【戻り値】

返却値	意　味
0	要求が受け入れられた。
(-1)	要求が受け入れられなかった。または cjid が管理外であった。

【説明】

当該インスタンスのプロパティ rsdc を含めた S6F23 メッセージを生成し、それを相手装置に送信するようエンジンに要求します。
要求がエンジンによって受け入れられたときは、0 を返却します。受け入れられなかった場合は(-1)を返却します。
送信後、受信した応答メッセージの ACK を rsda に保存し、それを引数にして、callback 関数を呼び出します。
callback 関数からは 0 を返却してください。

【callback の構文】

```
public delegate int callback_s6f23(
    int eqid,                       // 装置 ID
    int end_status,                 // 終了状態ｺｰﾄﾞ
    ref int rsda,                   // S6F24 に含まれる応答 ack です。
    uint upara                      // ﾕｰｻﾞﾊﾟﾗﾒｰﾀ(送信要求ﾒｿｯﾄﾞで指定された upara)
)
```

end_status に返却される値は、=0 の場合は正常終了を意味します。

end_status	意　味
0	送受信が正常に完了した。
-13	T3 ﾀｲﾑｱｳﾄを検出し、応答ﾒｯｾｰｼﾞを受信できなかった。
(-1)	送信できなかった。

13．３．３．２　send_wait()

S6F23 メッセージの送信要求をし,引き続き応答メッセージも受信します。

【構文】

public int send_wait(ref DshSpoolRsp rsp_info)

【引数】

rsdc

スプールデータの転送、削除を指定します。(0=転送、1=削除)

rsda

S6F24 応答メッセージに含まれる ack 情報を保存します。

【戻り値】

返却値	意　味
0	正常に送受信した。
＜ 0	送受信ｴﾗｰが発生した。(送信ｴﾗｰ or T3 ﾀｲﾑｱｳﾄなど)

【説明】

当該インスタンスのプロパティ rsdc を含めた S6F23 メッセージを生成し、それを相手装置に送信するようエンジンに要求します。

本メソッドは送信の後、引き続き S6F24 応答メッセージを待機し、受信が終了したら、上の【戻り値】に示した値を返却します。ei=0 の場合は、受信した応答メッセージの ACK が rsda に保存され渡されます。

応答メッセージを受信するまでプログラムはブロック（待ち）状態になります。

送信エラー、受信エラーを検出した場合は、ei ＜ 0 の値を返却します。(T3 ﾀｲﾑｱｳﾄも含む)

T3 タイムアウトが発生した場合、DSHDR2 HSMS ドライバーに設定されている T3 プロトコルタイムアウト時間だけブロックされることになります。

14．ホストコマンド、キャリアアクション関連メッセージの送受信

以下のメッセージとクラスがあります。

	メッセージ	クラス名	機能概略
1	S2F41,42	DshS2F41Send	S2F41 送信　Host Command Send
		DshS2F42Response	S2F42 応答
2	S2F49,50	DshS2F49Send	S2F49 送信　Enhanced Remote Command
		DshS2F50Response	S2F50 応答
3	S3F17,18	DshS3F17Send	S3F17 送信　Carrier Action Request
		DshS3F18Response	S3F18 応答
4	S3F23,24	DshS3F23Send	S3F23 送信　Port Group Action Request
		DshS3F24Response	S3F24 応答
5	S3F25,26	DshS3F25Send	S3F25 送信　Port Action Request
		DshS3F26Response	S3F26 応答
6	S3F27,28	DshS3F27Send	S3F27 送信　Change Access
		DshS3F28Response	S3F28 応答

14. 1 DshS2F41Send クラス

S2F41 メッセージを送信し、S2F42 応答メッセージを受信するためのクラスです。
ホストリモートコマンド情報の送信を行います。

14. 1. 1 コンストラクタ

オーバーロードの一覧を示します。

	名前	説明
1	public DshS2F41Send()	装置 ID=0 の空のｲﾝｽﾀﾝｽを生成します。
2	public DshS2F41Send(ref DshRCmd info)	ﾘﾓｰﾄｺﾏﾝﾄﾞ情報を指定して装置 ID=0 のｲﾝｽﾀﾝｽを生成します。
3	public DshS2F41Send(int eqid)	装置 ID を指定してｲﾝｽﾀﾝｽを生成します。
4	public DshS2F41Send(int eqid, ref DshRCmd info)	装置 ID とﾘﾓｰﾄｺﾏﾝﾄﾞ情報を指定してｲﾝｽﾀﾝｽを生成します。

14. 1. 2 プロパティ

プロパティ一覧表に示します。

	名前	説明
1	public int eqid	装置 ID です。ﾃﾞﾌｫﾙﾄ値=0 です。
2	public DshRCmd cmd_info	ﾘﾓｰﾄｺﾏﾝﾄﾞ情報が保存される DshRcmd ｸﾗｽのｲﾝｽﾀﾝｽです。 DshRCmd ｸﾗｽについては、Vol-1 16.1 参照

14. 1. 3 メソッド

本クラスのメソッドは次の通りです。

	名前	説明
1	public void set_cmd_info()	cmd_info の情報を設定します。 DshRCmd ｸﾗｽのｲﾝｽﾀﾝｽを設定します。
2	public int send_s2f41() public int send()	S2F41 ﾒｯｾｰｼﾞを送信します。
3	public int send_wait()	S2F41 ﾒｯｾｰｼﾞを送信し、S2F42 を受信します。 ﾌﾟﾛｸﾞﾗﾑは応答受信までﾌﾞﾛｯｸされます。
4	public void Dispose()	ｸﾗｽ内部で使用された資源(ﾒﾓﾘ)ｼｽﾃﾑに返却します。 他のｸﾗｽをﾌﾟﾛﾊﾟﾃｨにしている場合、そのｸﾗｽの Dispose()も行います。 3.1.3.5 と同様です。そちらを参照ください。

14. 1. 3. 1 set_info()

【構文】

public void set_info(ref DshRCmd info)

【引数】

info

設定したい DshRCmd クラスのインスタンスです。

【戻り値】

なし。

【説明】

info に与えられた DshRCmd クラスのインスタンスの内容を cmd_info に設定します。
実際にはコピーします。

14. 1. 3. 2 send_s2f41(), send()

S2F41 メッセージの送信要求をします。

【構文】

public int send_s2f41(ref DshRCmdRsp rsp_info, DshCallback.callback_s2f41 callback, uint upara)
public int send(ref DshRCmdRsp rsp_info, DshCallback.callback_s2f41 callback, uint upara)

【引数】

rsp_info
　　S2F42 応答メッセージに含まれる情報を保存します。コールバック関数の引数になります。
callback
　　S2F41 送信後、2 次メッセージ受信した時に呼び出される callback 関数です。
upara
　　ユーザパラメータ情報です。コールバックされる際に callback 関数の引数として渡されます。
　　要求とコールバック間のタグ情報として使用できます。

【戻り値】

返却値	意　味
0	要求が受け入れられた。
(-1)	要求が受け入れられなかった。または cjid が管理外であった。

【説明】

当該インスタンスのプロパティ cmd_info メンバーから S2F41 メッセージを生成し、それを相手装置に送信するようエンジンに要求します。

要求がエンジンによって受け入れられたときは、0 を返却します。受け入れられなかった場合は(-1)を返却します。
送信後、受信した応答メッセージ情報を中に保存した rsp_info を引数にして、callback 関数を呼び出します。
callback 関数からは 0 を返却してください。

【callback の構文】

```
public delegate int callback_s2f41(
    int eqid,                       // 装置 ID
    int end_status,                 // 終了状態ｺｰﾄﾞ
    ref DshRCmdRsp rsp_info,        // S2F42 に含まれる応答です。 Vol-1 16.10 参照
    uint upara                      // ﾕｰｻﾞﾊﾟﾗﾒｰﾀ(送信要求ﾒｿｯﾄﾞで指定された upara)
)
```

end_status に返却される値は、=0 の場合は正常終了を意味します。

end_status	意　味
0	送受信が正常に完了した。
-14	T3 ﾀｲﾑｱｳﾄを検出し、応答ﾒｯｾｰｼﾞを受信できなかった。
(-1)	送信できなかった。

14. 1. 3. 3 send_wait()

S2F41 メッセージの送信要求をし,引き続き応答メッセージも受信します。

【構文】

public int send_wait(ref DshRCmdRsp rsp_info)

【引数】

rsp_info

S2F42 応答メッセージに含まれる情報を保存します。

【戻り値】

返却値	意　味
0	正常に送受信した。
< 0	送受信ｴﾗｰが発生した。(送信ｴﾗｰ or T3 ﾀｲﾑｱｳﾄなど)

【説明】

当該インスタンスのプロパティ cmd_info メンバーから S2F41 メッセージを生成し、それを相手装置に送信するようエンジンに要求します。

本メソッドは送信の後、引き続き S2F42 応答メッセージを待機し、受信が終了したら、上の【戻り値】に示した値を返却します。ei=0 の場合は、受信した応答メッセージ情報が rsp_info に保存され渡されます。

応答メッセージを受信するまでプログラムはブロック（待ち）状態になります。

送信エラー、受信エラーを検出した場合は、ei < 0 の値を返却します。(T3 ﾀｲﾑｱｳﾄも含む)

T3 タイムアウトが発生した場合、DSHDR2 HSMS ドライバーに設定されている T3 プロトコルタイムアウト時間だけブロックされることになります。

14. 2 DshS2F42Response クラス

S2F41 メッセージを受信した後、S2F42 応答メッセージを送信するためのクラスです。

14. 2. 1 コンストラクタ

オーバーロードの一覧を示します。

	名前	説明
1	public DshS2F42Response()	装置 ID=0 のｲﾝｽﾀﾝｽを生成します。
2	public DshS2F42Response(int eqid)	装置 ID を指定してｲﾝｽﾀﾝｽを生成します。

14. 2. 2 プロパティ

プロパティ一覧表に示します。

	名前	説明
1	public int eqid	装置 ID です。ﾃﾞﾌｫﾙﾄ値=0 です。

14. 2. 3 メソッド

本クラスのメソッドは次の通りです。

	名前	説明
1	public int response()	S2F42 応答ﾒｯｾｰｼﾞを送信します。
2	public void Dispose()	ｸﾗｽ内部で使用された資源(ﾒﾓﾘ)ｼｽﾃﾑに返却します。 他のｸﾗｽをﾌﾟﾛﾊﾟﾃｨにしている場合、そのｸﾗｽのDispose()も行います。 3.1.3.5 と同様です。そちらを参照ください。

14. 2. 3. 1 response()

S2F42 メッセージの応答送信要求をします。

【構文】

public int response(uint trid, ref DshRCmdRsp rsp_class)

【引数】

trid

エンジンからポーリング時に与えられたトランザクション ID です。
処理した１次メッセージとの対応を取ります。

rsp_class

S2F42 メッセージに含める応答情報のインスタンスです。
DshRCmdRsp クラスについては Vol-1 の 16.10 を参照してください。

【戻り値】

返却値	意　味
0	要求が受け入れられた。
(-1)	要求が受け入れられなかった。(ﾄﾗﾝｻﾞｸｼｮﾝ ID が正しくなかった。)

【説明】

受信した S2F41 に対する S2F42 メッセージを送信します。
DshRCmdRsp クラスのインスタンス rsp_class 内の応答情報から S2F42 を生成します。
trid は、この応答メッセージに対する S2F41 受信時にエンジンから与えられたトランザクション ID です。
trid がエンジンの管理の中に見つからなかった場合には、(-1)を返却します。
要求が受け入れられたら 0 を返却します。この後、応答メッセージ送信結果は特に通知されません。

14. 3 DshS2F49Send クラス

S2F49 メッセージを送信し、S2F50 応答メッセージを受信するためのクラスです。
ホストリモートコマンド情報の送信を行います。

14. 3. 1 コンストラクタ

オーバーロードの一覧を示します。

	名前	説明
1	public DshS2F49Send()	装置 ID=0 の空のｲﾝｽﾀﾝｽを生成します。
2	public DshS2F49Send(ref DshERCmd info)	ﾘﾓｰﾄｺﾏﾝﾄﾞ情報を指定して装置 ID=0 のｲﾝｽﾀﾝｽを生成します。
3	public DshS2F49Send(int eqid)	装置 ID を指定してｲﾝｽﾀﾝｽを生成します。
4	public DshS2F49Send(int eqid, ref DshERCmd info)	装置 ID とﾘﾓｰﾄｺﾏﾝﾄﾞ情報を指定してｲﾝｽﾀﾝｽを生成します。

14. 3. 2 プロパティ

プロパティ一覧表に示します。

	名前	説明
1	public int eqid	装置 ID です。ﾃﾞﾌｫﾙﾄ値=0 です。
2	public DshERCmd cmd_info	ﾘﾓｰﾄｺﾏﾝﾄﾞ情報が保存される DshERcmd ｸﾗｽのｲﾝｽﾀﾝｽです。 DshERCmd ｸﾗｽについては、Vol-1 16.2 参照

14.. 3　メソッド

本クラスのメソッドは次の通りです。

	名前	説明
1	public void set_cmd_info()	cmd_info の情報を設定します。 DshERCmd ｸﾗｽのｲﾝｽﾀﾝｽを設定します。
2	public int send_s2f49() public int send()	S2F49 ﾒｯｾｰｼﾞを送信します。
3	public int send_wait()	S2F49 ﾒｯｾｰｼﾞを送信し、S2F50 を受信します。 ﾌﾟﾛｸﾞﾗﾑは応答受信までﾌﾞﾛｯｸされます。
4	public void Dispose()	ｸﾗｽ内部で使用された資源(ﾒﾓﾘ)ｼｽﾃﾑに返却します。 他のｸﾗｽをﾌﾟﾛﾊﾟﾃｨにしている場合、そのｸﾗｽの Dispose()も行います。 3.1.3.5 と同様です。そちらを参照ください。

14. 3. 3. 1　set_info()

【構文】

public void set_info(ref　DshERCmd info)

【引数】

info

設定したい DshERCmd クラスのインスタンスです。

【戻り値】

なし。

【説明】

info に与えられた DshERCmd クラスのインスタンスの内容を cmd_info に設定します。
実際にはコピーします。

14. 3. 3. 2 send_s2f49(), send()

S2F49 メッセージの送信要求をします。

【構文】

public int send_s2f49(ref DshRCmdRsp rsp_info, DshCallback.callback_s2f49 callback, uint upara)
public int send(ref DshRCmdRsp rsp_info, DshCallback.callback_s2f49 callback, uint upara)

【引数】

rsp_info
　　S2F50 応答メッセージに含まれる情報を保存します。コールバック関数の引数になります。
callback
　　S2F49 送信後、2 次メッセージ受信した時に呼び出される callback 関数です。
upara
　　ユーザパラメータ情報です。コールバックされる際に callback 関数の引数として渡されます。
　　要求とコールバック間のタグ情報として使用できます。

【戻り値】

返却値	意　味
0	要求が受け入れられた。
(-1)	要求が受け入れられなかった。または cjid が管理外であった。

【説明】

当該インスタンスのプロパティ cmd_info メンバーから S2F49 メッセージを生成し、それを相手装置に送信するようエンジンに要求します。

要求がエンジンによって受け入れられたときは、0 を返却します。受け入れられなかった場合は(-1)を返却します。
送信後、受信した応答メッセージ情報を中に保存した rsp_info を引数にして、callback 関数を呼び出します。
callback 関数からは 0 を返却してください。

【callback の構文】

```
public delegate int callback_s2f49(
    int eqid,                       // 装置 ID
    int end_status,                 // 終了状態ｺｰﾄﾞ
    ref DshRCmdRsp rsp_info,        // S2F50 に含まれる応答です。 Vol-1 16.10 参照
    uint upara                      // ﾕｰｻﾞﾊﾟﾗﾒｰﾀ(送信要求ﾒｿｯﾄﾞで指定された upara)
)
```

end_status に返却される値は、=0 の場合は正常終了を意味します。

end_status	意　味
0	送受信が正常に完了した。
-14	T3 ﾀｲﾑｱｳﾄを検出し、応答ﾒｯｾｰｼﾞを受信できなかった。
(-1)	送信できなかった。

14．３．３．３　send_wait()

S2F49 メッセージの送信要求をし,引き続き応答メッセージも受信します。

【構文】

public int send_wait(ref DshRCmdRsp rsp_info)

【引数】

rsp_info

S2F50 応答メッセージに含まれる情報を保存します。

【戻り値】

返却値	意　味
0	正常に送受信した。
＜ 0	送受信ｴﾗｰが発生した。（送信ｴﾗｰ or T3 ﾀｲﾑｱｳﾄなど）

【説明】

当該インスタンスのプロパティ cmd_info メンバーから S2F49 メッセージを生成し、それを相手装置に送信するようエンジンに要求します。

本メソッドは送信の後、引き続き S2F50 応答メッセージを待機し、受信が終了したら、上の【戻り値】に示した値を返却します。ei=0 の場合は、受信した応答メッセージ情報が rsp_info に保存され渡されます。

応答メッセージを受信するまでプログラムはブロック（待ち）状態になります。

送信エラー、受信エラーを検出した場合は、ei ＜ 0 の値を返却します。（T3 ﾀｲﾑｱｳﾄも含む）

T3 タイムアウトが発生した場合、DSHDR2 HSMS ドライバーに設定されている T3 プロトコルタイムアウト時間だけブロックされることになります。

14. 4 DshS2F50Response クラス

S2F49 メッセージを受信した後、S2F50 応答メッセージを送信するためのクラスです。

14. 4. 1 コンストラクタ

オーバーロードの一覧を示します。

	名前	説明
1	public DshS2F50Response()	装置 ID=0 のｲﾝｽﾀﾝｽを生成します。
2	public DshS2F50Response(int eqid)	装置 ID を指定してｲﾝｽﾀﾝｽを生成します。

14. 4. 2 プロパティ

プロパティ一覧表に示します。

	名前	説明
1	public int eqid	装置 ID です。ﾃﾞﾌｫﾙﾄ値=0 です。

14. ４. ３　メソッド

本クラスのメソッドは次の通りです。

	名前	説明
1	public int response()	S2F50 応答ﾒｯｾｰｼﾞを送信します。
2	public void Dispose()	ｸﾗｽ内部で使用された資源(ﾒﾓﾘ)ｼｽﾃﾑに返却します。 他のｸﾗｽをﾌﾟﾛﾊﾟﾃｨにしている場合、そのｸﾗｽの Dispose()も行います。 3.1.3.5 と同様です。そちらを参照ください。

14. ４. ３. １　response()

S2F50 メッセージの応答送信要求をします。

【構文】

public int response(uint trid, ref DshRCmdRsp rsp_class)

【引数】

trid

エンジンからポーリング時に与えられたトランザクション ID です。
処理した１次メッセージとの対応を取ります。

rsp_class

S2F50 メッセージに含める応答情報のインスタンスです。
DshRCmdRsp クラスについては Vol-1 の 16.10 を参照してください。

【戻り値】

返却値	意　味
0	要求が受け入れられた。
(-1)	要求が受け入れられなかった。(ﾄﾗﾝｻﾞｸｼｮﾝ ID が正しくなかった。)

【説明】

受信した S2F49 に対する S2F50 メッセージを送信します。
DshRCmdRsp クラスのインスタンス rsp_class 内の応答情報から S2F50 を生成します。
trid は、この応答メッセージに対する S2F49 受信時にエンジンから与えられたトランザクション ID です。
trid がエンジンの管理の中に見つからなかった場合には、(-1)を返却します。
要求が受け入れられたら 0 を返却します。この後、応答メッセージ送信結果は特に通知されません。

14. 5 DshS3F17Send クラス

S3F17 メッセージを送信し、S3F18 応答メッセージを受信するためのクラスです。
キャリアアクション情報の送信を行います。

14. 5. 1 コンストラクタ

オーバーロードの一覧を示します。

	名前	説明
1	public DshS3F17Send()	装置 ID=0 の空のｲﾝｽﾀﾝｽを生成します。
2	public DshS3F17Send(ref DshCarAction info)	ｷｬﾘｱｱｸｼｮﾝ情報を指定して装置 ID=0 のｲﾝｽﾀﾝｽを生成します。
3	public DshS3F17Send(int eqid)	装置 ID を指定してｲﾝｽﾀﾝｽを生成します。
4	public DshS3F17Send(int eqid, ref DshCarAction info)	装置 ID とｷｬﾘｱｱｸｼｮﾝ情報を指定してｲﾝｽﾀﾝｽを生成します。

14. 5. 2 プロパティ

プロパティ一覧表に示します。

	名前	説明
1	public int eqid	装置 ID です。ﾃﾞﾌｫﾙﾄ値=0 です。
2	public DshCarAction ca_info	ｷｬﾘｱｱｸｼｮﾝ情報が保存される DshCarAction ｸﾗｽのｲﾝｽﾀﾝｽです。 DshCarAction ｸﾗｽについては、Vol-1 16.8 参照

14. 5 メソッド

本クラスのメソッドは次の通りです。

	名前	説明
1	public void set_info()	ca_info に情報を設定します。 DshCarAction ｸﾗｽのｲﾝｽﾀﾝｽを設定します。
2	public int send_s3f17() public int send()	S3F17 ﾒｯｾｰｼﾞを送信します。
3	public int send_wait()	S3F17 ﾒｯｾｰｼﾞを送信し、S3F18 を受信します。 ﾌﾟﾛｸﾞﾗﾑは応答受信までﾌﾞﾛｯｸされます。
4	public void Dispose()	ｸﾗｽ内部で使用された資源(ﾒﾓﾘ)ｼｽﾃﾑに返却します。 他のｸﾗｽをﾌﾟﾛﾊﾟﾃｨにしている場合、そのｸﾗｽの Dispose()も行います。 3.1.3.5 と同様です。そちらを参照ください。

14. 5. 3. 1 set_info()

【構文】

public void set_info(ref DshCarAction info)

【引数】

info

設定したい DshCarAction クラスのインスタンスです。

【戻り値】

なし。

【説明】

info に与えられた DshCarAction クラスのインスタンスの内容を ca_info に設定します。
実際にはコピーします。

14．５．３．２　send_s3f17(), send()

S3F17 メッセージの送信要求をします。

【構文】

```
public int send_s3f17(ref DshCarActionRsp rsp_info,
                              DshCallback.callback_s3f17 callback, uint upara)
public int send(ref DshCarActionRsp rsp_info,
                              DshCallback.callback_s3f17 callback, uint upara)
```

【引数】

rsp_info
　　S3F18 応答メッセージに含まれる情報を保存します。コールバック関数の引数になります。
callback
　　S3F17 送信後、2 次メッセージ受信した時に呼び出される callback 関数です。
upara
　　ユーザパラメータ情報です。コールバックされる際に callback 関数の引数として渡されます。
　　要求とコールバック間のタグ情報として使用できます。

【戻り値】

返却値	意　味
0	要求が受け入れられた。
(-1)	要求が受け入れられなかった。または cjid が管理外であった。

【説明】

当該インスタンスのプロパティ ca_info メンバーから S3F17 メッセージを生成し、それを相手装置に送信するようエンジンに要求します。

要求がエンジンによって受け入れられたときは、0 を返却します。受け入れられなかった場合は(-1)を返却します。
送信後、受信した応答メッセージ情報を中に保存した rsp_info を引数にして、callback 関数を呼び出します。
callback 関数からは 0 を返却してください。

【callback の構文】

```
public delegate int callback_s3f17(
    int eqid,                          // 装置 ID
    int end_status,                    // 終了状態ｺｰﾄﾞ
    ref DshCarActionRsp rsp_info,      // S3F18 に含まれる応答です。 Vol-1 16.12 参照
    uint upara                         // ﾕｰｻﾞﾊﾟﾗﾒｰﾀ(送信要求ﾒｿｯﾄﾞで指定された upara)
)
```

end_status に返却される値は、=0 の場合は正常終了を意味します。

end_status	意　味
0	送受信が正常に完了した。
-14	T3 ﾀｲﾑｱｳﾄを検出し、応答ﾒｯｾｰｼﾞを受信できなかった。
(-1)	送信できなかった。

14. 5. 3. 3 send_wait()

S3F17 メッセージの送信要求をし,引き続き応答メッセージも受信します。

【構文】

public int send_wait(ref DshCarActionRsp rsp_info)

【引数】

rsp_info

S3F18 応答メッセージに含まれる情報を保存します。

【戻り値】

返却値	意　味
0	正常に送受信した。
< 0	送受信ｴﾗｰが発生した。(送信ｴﾗｰ or T3 ﾀｲﾑｱｳﾄなど)

【説明】

当該インスタンスのプロパティ ca_info メンバーから S3F17 メッセージを生成し、それを相手装置に送信するようエンジンに要求します。

本メソッドは送信の後、引き続き S3F18 応答メッセージを待機し、受信が終了したら、上の【戻り値】に示した値を返却します。ei=0 の場合は、受信した応答メッセージ情報が rsp_info に保存され渡されます。

応答メッセージを受信するまでプログラムはブロック（待ち）状態になります。

送信エラー、受信エラーを検出した場合は、ei < 0 の値を返却します。(T3 ﾀｲﾑｱｳﾄも含む)

T3 タイムアウトが発生した場合、DSHDR2 HSMS ドライバーに設定されている T3 プロトコルタイムアウト時間だけブロックされることになります。

14. 6 DshS3F18Response クラス

S3F17 メッセージを受信した後、S3F18 応答メッセージを送信するためのクラスです。

14. 6. 1 コンストラクタ

オーバーロードの一覧を示します。

	名前	説明
1	public DshS3F18Response()	装置 ID=0 のｲﾝｽﾀﾝｽを生成します。
2	public DshS3F18Response(int eqid)	装置 ID を指定してｲﾝｽﾀﾝｽを生成します。

14. 6. 2 プロパティ

プロパティ一覧表に示します。

	名前	説明
1	public int eqid	装置 ID です。ﾃﾞﾌｫﾙﾄ値=0 です。

14．６．３　メソッド

本クラスのメソッドは次の通りです。

	名前	説明
1	public int response()	S3F18 応答ﾒｯｾｰｼﾞを送信します。
2	public void Dispose()	ｸﾗｽ内部で使用された資源(ﾒﾓﾘ)ｼｽﾃﾑに返却します。 他のｸﾗｽをﾌﾟﾛﾊﾟﾃｨにしている場合、そのｸﾗｽの Dispose()も行います。 3.1.3.5 と同様です。そちらを参照ください。

14．６．３．１　response()

S3F18 メッセージの応答送信要求をします。

【構文】

public int response(uint trid, ref DshCarActionRsp rsp_class)

【引数】

trid

エンジンからポーリング時に与えられたトランザクション ID です。
処理した１次メッセージとの対応を取ります。

rsp_class

S3F18 メッセージに含める応答情報のインスタンスです。
DshCarActionRsp クラスについては Vol-1 の 16.12 を参照してください。

【戻り値】

返却値	意　味
0	要求が受け入れられた。
(-1)	要求が受け入れられなかった。(ﾄﾗﾝｻﾞｸｼｮﾝ ID が正しくなかった。)

【説明】

受信した S3F17 に対する S3F18 メッセージを送信します。
DshCarActionRsp クラスのインスタンス rsp_class 内の応答情報から S3F18 を生成します。
trid は、この応答メッセージに対する S3F17 受信時にエンジンから与えられたトランザクション ID です。
trid がエンジンの管理の中に見つからなかった場合には、(-1)を返却します。
要求が受け入れられたら 0 を返却します。この後、応答メッセージ送信結果は特に通知されません。

14. 7 DshS3F23Send クラス

S3F23 メッセージを送信し、S3F24 応答メッセージを受信するためのクラスです。
ポートグループアクション情報の送信を行います。

14. 7. 1 コンストラクタ

オーバーロードの一覧を示します。

	名前	説明
1	public DshS3F23Send()	装置 ID=0 の空のｲﾝｽﾀﾝｽを生成します。
2	public DshS3F23Send(ref DshPortGroupAction info)	ﾎﾟｰﾄｸﾞﾙｰﾌﾟｱｸｼｮﾝ情報を指定して装置 ID=0 のｲﾝｽﾀﾝｽを生成します。
3	public DshS3F23Send(int eqid)	装置 ID を指定してｲﾝｽﾀﾝｽを生成します。
4	public DshS3F23Send(int eqid, ref DshPortGroupAction info)	装置 ID とﾎﾟｰﾄｸﾞﾙｰﾌﾟｱｸｼｮﾝ情報を指定してｲﾝｽﾀﾝｽを生成します。

14. 7. 2 プロパティ

プロパティ一覧表に示します。

	名前	説明
1	public int eqid	装置 ID です。ﾃﾞﾌｫﾙﾄ値=0 です。
2	public DshPortGroupAction action_info	ﾎﾟｰﾄｸﾞﾙｰﾌﾟｱｸｼｮﾝ情報が保存される DshPortGroupAction ｸﾗｽのｲﾝｽﾀﾝｽです。 DshPortGroupAction ｸﾗｽについては、Vol-1 16.7 参照

14. 7　メソッド

本クラスのメソッドは次の通りです。

	名前	説明
1	public void set_info()	action_info に情報を設定します。 DshPortGroupAction ｸﾗｽのｲﾝｽﾀﾝｽを設定します。
2	public int send_s3f23() public int send()	S3F23 ﾒｯｾｰｼﾞを送信します。
3	public int send_wait()	S3F23 ﾒｯｾｰｼﾞを送信し、S3F24 を受信します。 ﾌﾟﾛｸﾞﾗﾑは応答受信までﾌﾞﾛｯｸされます。
4	public void Dispose()	ｸﾗｽ内部で使用された資源(ﾒﾓﾘ)ｼｽﾃﾑに返却します。 他のｸﾗｽをﾌﾟﾛﾊﾟﾃｨにしている場合、そのｸﾗｽの Dispose()も行います。 3.1.3.5 と同様です。そちらを参照ください。

14. 7. 3. 1　set_info()

【構文】

public void set_info(ref　DshPortGroupAction info)

【引数】

info

設定したい DshPortGroupAction クラスのインスタンスです。

【戻り値】

なし。

【説明】

info に与えられた DshPortGroupAction クラスのインスタンスの内容を action_info に設定します。
実際にはコピーします。

14．７．３．２　send_s3f23(), send()

S3F23 メッセージの送信要求をします。

【構文】

```
public int send_s3f23(ref DshPortGroupActionRsp rsp_info,
                              DshCallback.callback_s3f23 callback, uint upara)
public int send(ref DshPortGroupActionRsp rsp_info,
                              DshCallback.callback_s3f23 callback, uint upara)
```

【引数】

rsp_info
　　S3F24 応答メッセージに含まれる情報を保存します。コールバック関数の引数になります。
callback
　　S3F23 送信後、2 次メッセージ受信した時に呼び出される callback 関数です。
upara
　　ユーザパラメータ情報です。コールバックされる際に callback 関数の引数として渡されます。
　　要求とコールバック間のタグ情報として使用できます。

【戻り値】

返却値	意　味
0	要求が受け入れられた。
(-1)	要求が受け入れられなかった。または cjid が管理外であった。

【説明】

当該インスタンスのプロパティ action_info メンバーから S3F23 メッセージを生成し、それを相手装置に送信するようエンジンに要求します。

要求がエンジンによって受け入れられたときは、0 を返却します。受け入れられなかった場合は(-1)を返却します。
送信後、受信した応答メッセージ情報を中に保存した rsp_info を引数にして、callback 関数を呼び出します。
callback 関数からは 0 を返却してください。

【callback の構文】

```
public delegate int callback_s3f23(
    int eqid,                          // 装置 ID
    int end_status,                    // 終了状態ｺｰﾄﾞ
    ref DshPortGroupActionRsp rsp_info, // S3F24 に含まれる応答です。 Vol-1 16.13 参照
    uint upara                         // ﾕｰｻﾞﾊﾟﾗﾒｰﾀ(送信要求ﾒｿｯﾄﾞで指定された upara)
)
```

end_status に返却される値は、=0 の場合は正常終了を意味します。

end_status	意　味
0	送受信が正常に完了した。
-14	T3 ﾀｲﾑｱｳﾄを検出し、応答ﾒｯｾｰｼﾞを受信できなかった。
(-1)	送信できなかった。

14. 7. 3. 3 send_wait()

S3F23 メッセージの送信要求をし,引き続き応答メッセージも受信します。

【構文】

public int send_wait(ref DshPortGroupActionRsp rsp_info)

【引数】

rsp_info

S3F24 応答メッセージに含まれる情報を保存します。

【戻り値】

返却値	意　味
0	正常に送受信した。
< 0	送受信ｴﾗｰが発生した。(送信ｴﾗｰ or T3 ﾀｲﾑｱｳﾄなど)

【説明】

当該インスタンスのプロパティ action_info メンバーから S3F23 メッセージを生成し、それを相手装置に送信するようエンジンに要求します。

本メソッドは送信の後、引き続き S3F24 応答メッセージを待機し、受信が終了したら、上の【戻り値】に示した値を返却します。ei=0 の場合は、受信した応答メッセージ情報が rsp_info に保存され渡されます。

応答メッセージを受信するまでプログラムはブロック（待ち）状態になります。

送信エラー、受信エラーを検出した場合は、ei < 0 の値を返却します。(T3 ﾀｲﾑｱｳﾄも含む)

T3 タイムアウトが発生した場合、DSHDR2 HSMS ドライバーに設定されている T3 プロトコルタイムアウト時間だけブロックされることになります。

14. 8 DshS3F24Response クラス

S3F23 メッセージを受信した後、S3F24 応答メッセージを送信するためのクラスです。

14. 8. 1 コンストラクタ

オーバーロードの一覧を示します。

	名前	説明
1	public DshS3F24Response()	装置 ID=0 のｲﾝｽﾀﾝｽを生成します。
2	public DshS3F24Response(int eqid)	装置 ID を指定してｲﾝｽﾀﾝｽを生成します。

14. 8. 2 プロパティ

プロパティ一覧表に示します。

	名前	説明
1	public int eqid	装置 ID です。ﾃﾞﾌｫﾙﾄ値=0 です。

14．８．３　メソッド

本クラスのメソッドは次の通りです。

	名前	説明
1	public int response()	S3F24 応答ﾒｯｾｰｼﾞを送信します。
2	public void Dispose()	ｸﾗｽ内部で使用された資源(ﾒﾓﾘ)ｼｽﾃﾑに返却します。 他のｸﾗｽをﾌﾟﾛﾊﾟﾃｨにしている場合、そのｸﾗｽの Dispose()も行います。 3.1.3.5 と同様です。そちらを参照ください。

14．８．３．１　response()

S3F24 メッセージの応答送信要求をします。

【構文】

public int response(uint trid, ref DshPortGroupActionRsp rsp_class)

【引数】

trid

エンジンからポーリング時に与えられたトランザクション ID です。
処理した１次メッセージとの対応を取ります。

rsp_class

S3F24 メッセージに含める応答情報のインスタンスです。
DshPortGroupActionRsp クラスについては Vol-1 の 16.13 を参照してください。

【戻り値】

返却値	意　味
0	要求が受け入れられた。
(-1)	要求が受け入れられなかった。(ﾄﾗﾝｻﾞｸｼｮﾝ ID が正しくなかった。)

【説明】

受信した S3F23 に対する S3F24 メッセージを送信します。
DshPortGroupActionRsp クラスのインスタンス rsp_class 内の応答情報から S3F24 を生成します。
trid は、この応答メッセージに対する S3F23 受信時にエンジンから与えられたトランザクション ID です。
trid がエンジンの管理の中に見つからなかった場合には、(-1)を返却します。
要求が受け入れられたら 0 を返却します。この後、応答メッセージ送信結果は特に通知されません。

14. 9 DshS3F25Send クラス

S3F25 メッセージを送信し、S3F26 応答メッセージを受信するためのクラスです。
ポートアクション情報の送信を行います。

14. 9. 1 コンストラクタ

オーバーロードの一覧を示します。

	名前	説明
1	public DshS3F25Send()	装置 ID=0 の空のｲﾝｽﾀﾝｽを生成します。
2	public DshS3F25Send(ref DshPortAction info)	ﾎﾟｰﾄｱｸｼｮﾝ情報を指定して装置 ID=0 のｲﾝｽﾀﾝｽを生成します。
3	public DshS3F25Send(int eqid)	装置 ID を指定してｲﾝｽﾀﾝｽを生成します。
4	public DshS3F25Send(int eqid, ref DshPortAction info)	装置 ID とﾎﾟｰﾄｱｸｼｮﾝ情報を指定してｲﾝｽﾀﾝｽを生成します。

14. 9. 2 プロパティ

プロパティ一覧表に示します。

	名前	説明
1	public int eqid	装置 ID です。ﾃﾞﾌｫﾙﾄ値=0 です。
2	public DshPortAction action_info	ﾎﾟｰﾄｱｸｼｮﾝ情報が保存される DshPortAction ｸﾗｽのｲﾝｽﾀﾝｽです。 DshPortAction ｸﾗｽについては、Vol-1 16.8 参照

14. 9. 3 メソッド

本クラスのメソッドは次の通りです。

	名前	説明
1	public void set_info()	action_info に情報を設定します。 DshPortAction ｸﾗｽのｲﾝｽﾀﾝｽを設定します。
2	public int send_s3f25() public int send()	S3F25 ﾒｯｾｰｼﾞを送信します。
3	public int send_wait()	S3F25 ﾒｯｾｰｼﾞを送信し、S3F26 を受信します。 ﾌﾟﾛｸﾞﾗﾑは応答受信までﾌﾞﾛｯｸされます。
4	public void Dispose()	ｸﾗｽ内部で使用された資源(ﾒﾓﾘ)ｼｽﾃﾑに返却します。 他のｸﾗｽをﾌﾟﾛﾊﾟﾃｨにしている場合、そのｸﾗｽの Dispose()も行います。 3.1.3.5 と同様です。そちらを参照ください。

14. 9. 3. 1 set_info()

【構文】

public void set_info(ref DshPortAction info)

【引数】

info

設定したい DshPortAction クラスのインスタンスです。

【戻り値】

なし。

【説明】

info に与えられた DshPortAction クラスのインスタンスの内容を action_info に設定します。
実際にはコピーします。

14. 9. 3. 2 send_s3f25(), send()

S3F25 メッセージの送信要求をします。

【構文】

```
public int send_s3f25(ref DshPortActionRsp rsp_info,
                              DshCallback.callback_s3f25 callback, uint upara)
public int send(ref DshPortActionRsp rsp_info,
                              DshCallback.callback_s3f25 callback, uint upara)
```

【引数】

rsp_info

S3F26 応答メッセージに含まれる情報を保存します。コールバック関数の引数になります。

callback

S3F25 送信後、2 次メッセージ受信した時に呼び出される callback 関数です。

upara

ユーザパラメータ情報です。コールバックされる際に callback 関数の引数として渡されます。
要求とコールバック間のタグ情報として使用できます。

【戻り値】

返却値	意　味
0	要求が受け入れられた。
(-1)	要求が受け入れられなかった。または cjid が管理外であった。

【説明】

当該インスタンスのプロパティ action_info メンバーから S3F25 メッセージを生成し、それを相手装置に送信するようエンジンに要求します。

要求がエンジンによって受け入れられたときは、0 を返却します。受け入れられなかった場合は(-1)を返却します。
送信後、受信した応答メッセージ情報を中に保存した rsp_info を引数にして、callback 関数を呼び出します。
callback 関数からは 0 を返却してください。

【callback の構文】

```
public delegate int callback_s3f25(
    int eqid,                          // 装置 ID
    int end_status,                    // 終了状態ｺｰﾄﾞ
    ref DshPortActionRsp rsp_info,  // S3F26 に含まれる応答です。 Vol-1 16.13 参照
    uint upara                         // ﾕｰｻﾞﾊﾟﾗﾒｰﾀ(送信要求ﾒｿｯﾄﾞで指定された upara)
)
```

end_status に返却される値は、=0 の場合は正常終了を意味します。

end_status	意　味
0	送受信が正常に完了した。
-14	T3 ﾀｲﾑｱｳﾄを検出し、応答ﾒｯｾｰｼﾞを受信できなかった。
(-1)	送信できなかった。

14. 9. 3. 3 send_wait()

S3F25 メッセージの送信要求をし,引き続き応答メッセージも受信します。

【構文】

public int send_wait(ref DshPortActionRsp rsp_info)

【引数】

rsp_info

S3F26 応答メッセージに含まれる情報を保存します。

【戻り値】

返却値	意　味
0	正常に送受信した。
< 0	送受信ｴﾗｰが発生した。(送信ｴﾗｰ or T3 ﾀｲﾑｱｳﾄなど)

【説明】

当該インスタンスのプロパティ action_info メンバーから S3F25 メッセージを生成し、それを相手装置に送信するようエンジンに要求します。

本メソッドは送信の後、引き続き S3F26 応答メッセージを待機し、受信が終了したら、上の【戻り値】に示した値を返却します。ei=0 の場合は、受信した応答メッセージ情報が rsp_info に保存され渡されます。

応答メッセージを受信するまでプログラムはブロック（待ち）状態になります。

送信エラー、受信エラーを検出した場合は、ei < 0 の値を返却します。(T3 ﾀｲﾑｱｳﾄも含む)

T3 タイムアウトが発生した場合、DSHDR2 HSMS ドライバーに設定されている T3 プロトコルタイムアウト時間だけブロックされることになります。

14. 10 DshS3F26Response クラス

S3F25 メッセージを受信した後、S3F26 応答メッセージを送信するためのクラスです。

14. 10. 1 コンストラクタ

オーバーロードの一覧を示します。

	名前	説明
1	public DshS3F26Response()	装置 ID=0 のｲﾝｽﾀﾝｽを生成します。
2	public DshS3F26Response(int eqid)	装置 ID を指定してｲﾝｽﾀﾝｽを生成します。

14. 10. 2 プロパティ

プロパティ一覧表に示します。

	名前	説明
1	public int eqid	装置 ID です。ﾃﾞﾌｫﾙﾄ値=0 です。

14. 10. 3　メソッド

本クラスのメソッドは次の通りです。

	名前	説明
1	public int response()	S3F26 応答ﾒｯｾｰｼﾞを送信します。
2	public void Dispose()	ｸﾗｽ内部で使用された資源(ﾒﾓﾘ)ｼｽﾃﾑに返却します。 他のｸﾗｽをﾌﾟﾛﾊﾟﾃｨにしている場合、そのｸﾗｽの Dispose()も行います。 3.1.3.5 と同様です。そちらを参照ください。

14. 10. 3. 1　response()

S3F26 メッセージの応答送信要求をします。

【構文】

public int response(uint trid, ref DshPortActionRsp rsp_class)

【引数】

trid

エンジンからポーリング時に与えられたトランザクション ID です。
処理した１次メッセージとの対応を取ります。

rsp_class

S3F26 メッセージに含める応答情報のインスタンスです。
DshPortActionRsp クラスについては Vol-1 の 16.13 を参照してください。

【戻り値】

返却値	意　味
0	要求が受け入れられた。
(-1)	要求が受け入れられなかった。(ﾄﾗﾝｻﾞｸｼｮﾝ ID が正しくなかった。)

【説明】

受信した S3F25 に対する S3F26 メッセージを送信します。
DshPortActionRsp クラスのインスタンス rsp_class 内の応答情報から S3F26 を生成します。
trid は、この応答メッセージに対する S3F25 受信時にエンジンから与えられたトランザクション ID です。
trid がエンジンの管理の中に見つからなかった場合には、(-1)を返却します。
要求が受け入れられたら 0 を返却します。この後、応答メッセージ送信結果は特に通知されません。

14. 11 DshS3F27Send クラス

S3F27 メッセージを送信し、S3F28 応答メッセージを受信するためのクラスです。
ポートアクセス変更情報の送信を行います。

14. 11. 1 コンストラクタ

オーバーロードの一覧を示します。

	名前	説明
1	public DshS3F27Send()	装置 ID=0 の空のｲﾝｽﾀﾝｽを生成します。
2	public DshS3F27Send(ref DshPortAccess info)	ﾎﾟｰﾄｱｸｾｽ情報を指定して装置 ID=0 のｲﾝｽﾀﾝｽを生成します。
3	public DshS3F27Send(int eqid)	装置 ID を指定してｲﾝｽﾀﾝｽを生成します。
4	public DshS3F27Send(int eqid, ref DshPortAccess info)	装置 ID とﾎﾟｰﾄｱｸｾｽ情報を指定してｲﾝｽﾀﾝｽを生成します。

14. 11. 2 プロパティ

プロパティ一覧表に示します。

	名前	説明
1	public int eqid	装置 ID です。ﾃﾞﾌｫﾙﾄ値=0 です。
2	public DshPortAccess access_info	ﾎﾟｰﾄｱｸｾｽ情報が保存される DshPortAccess ｸﾗｽのｲﾝｽﾀﾝｽです。 DshPortAccess ｸﾗｽについては、Vol-1 16.9 参照

14. 11. 3　メソッド

本クラスのメソッドは次の通りです。

	名前	説明
1	public void set_info()	access_info に情報を設定します。 DshPortAccess ｸﾗｽのｲﾝｽﾀﾝｽを設定します。
2	public int send_s3f27() public int send()	S3F27 ﾒｯｾｰｼﾞを送信します。
3	public int send_wait()	S3F27 ﾒｯｾｰｼﾞを送信し、S3F28 を受信します。 ﾌﾟﾛｸﾞﾗﾑは応答受信までﾌﾞﾛｯｸされます。
4	public void Dispose()	ｸﾗｽ内部で使用された資源(ﾒﾓﾘ)ｼｽﾃﾑに返却します。 他のｸﾗｽをﾌﾟﾛﾊﾟﾃｨにしている場合、そのｸﾗｽの Dispose()も行います。 3.1.3.5 と同様です。そちらを参照ください。

14. 11. 3. 1　set_info()

【構文】

public void set_info(ref　DshPortAccess info)

【引数】

info

設定したい DshPortAccess クラスのインスタンスです。

【戻り値】

なし。

【説明】

info に与えられた DshPortAccess クラスのインスタンスの内容を access_info に設定します。
実際にはコピーします。

14. 11. 3. 2 send_s3f27(), send()

S3F27 メッセージの送信要求をします。

【構文】

```
public int send_s3f27(ref DshPortAccessRsp rsp_info,
                      DshCallback.callback_s3f27 callback, uint upara)
public int send(ref DshPortAccessRsp rsp_info,
                      DshCallback.callback_s3f27 callback, uint upara)
```

【引数】

rsp_info

S3F28 応答メッセージに含まれる情報を保存します。コールバック関数の引数になります。

callback

S3F27 送信後、2 次メッセージ受信した時に呼び出される callback 関数です。

upara

ユーザパラメータ情報です。コールバックされる際に callback 関数の引数として渡されます。
要求とコールバック間のタグ情報として使用できます。

【戻り値】

返却値	意　味
0	要求が受け入れられた。
(-1)	要求が受け入れられなかった。または cjid が管理外であった。

【説明】

当該インスタンスのプロパティ access_info メンバーから S3F27 メッセージを生成し、それを相手装置に送信するようエンジンに要求します。

要求がエンジンによって受け入れられたときは、0 を返却します。受け入れられなかった場合は(-1)を返却します。
送信後、受信した応答メッセージ情報を中に保存した rsp_info を引数にして、callback 関数を呼び出します。
callback 関数からは 0 を返却してください。

【callback の構文】

```
public delegate int callback_s3f27(
    int eqid,                          // 装置 ID
    int end_status,                    // 終了状態ｺｰﾄﾞ
    ref DshPortAccessRsp rsp_info,     // S3F28 に含まれる応答です。 Vol-1 16.15 参照
    uint upara                         // ﾕｰｻﾞﾊﾟﾗﾒｰﾀ(送信要求ﾒｿｯﾄﾞで指定された upara)
)
```

end_status に返却される値は、=0 の場合は正常終了を意味します。

end_status	意　味
0	送受信が正常に完了した。
-14	T3 ﾀｲﾑｱｳﾄを検出し、応答ﾒｯｾｰｼﾞを受信できなかった。
(-1)	送信できなかった。

14. 11. 3. 3 send_wait()

S3F27 メッセージの送信要求をし,引き続き応答メッセージも受信します。

【構文】

public int send_wait(ref DshPortAccessRsp rsp_info)

【引数】

rsp_info

S3F28 応答メッセージに含まれる情報を保存します。

【戻り値】

返却値	意　味
0	正常に送受信した。
< 0	送受信ｴﾗｰが発生した。(送信ｴﾗｰ or T3 ﾀｲﾑｱｳﾄなど)

【説明】

当該インスタンスのプロパティ access_info メンバーから S3F27 メッセージを生成し、それを相手装置に送信するようエンジンに要求します。

本メソッドは送信の後、引き続き S3F28 応答メッセージを待機し、受信が終了したら、上の【戻り値】に示した値を返却します。ei=0 の場合は、受信した応答メッセージ情報が rsp_info に保存され渡されます。

応答メッセージを受信するまでプログラムはブロック（待ち）状態になります。

送信エラー、受信エラーを検出した場合は、ei < 0 の値を返却します。(T3 ﾀｲﾑｱｳﾄも含む)

T3 タイムアウトが発生した場合、DSHDR2 HSMS ドライバーに設定されている T3 プロトコルタイムアウト時間だけブロックされることになります。

14. 12 DshS3F28Response クラス

S3F27 メッセージを受信した後、S3F28 応答メッセージを送信するためのクラスです。

14. 12. 1 コンストラクタ

オーバーロードの一覧を示します。

	名前	説明
1	public DshS3F28Response()	装置 ID=0 のｲﾝｽﾀﾝｽを生成します。
2	public DshS3F28Response(int eqid)	装置 ID を指定してｲﾝｽﾀﾝｽを生成します。

14. 12. 2 プロパティ

プロパティ一覧表に示します。

	名前	説明
1	public int eqid	装置 ID です。ﾃﾞﾌｫﾙﾄ値=0 です。

14. 12. 3　メソッド

本クラスのメソッドは次の通りです。

	名前	説明
1	public int response()	S3F28 応答ﾒｯｾｰｼﾞを送信します。
2	public void Dispose()	ｸﾗｽ内部で使用された資源(ﾒﾓﾘ)ｼｽﾃﾑに返却します。 他のｸﾗｽをﾌﾟﾛﾊﾟﾃｨにしている場合、そのｸﾗｽの Dispose()も行います。 3.1.3.5 と同様です。そちらを参照ください。

14. 12. 3. 1　response()

S3F28 メッセージの応答送信要求をします。

【構文】

public int response(uint trid, ref DshPortAccessRsp rsp_class)

【引数】

trid

エンジンからポーリング時に与えられたトランザクション ID です。
処理した１次メッセージとの対応を取ります。

rsp_class

S3F28 メッセージに含める応答情報のインスタンスです。
DshPortAccessRsp クラスについては Vol-1 の 16.15 を参照してください。

【戻り値】

返却値	意　味
0	要求が受け入れられた。
(-1)	要求が受け入れられなかった。(ﾄﾗﾝｻﾞｸｼｮﾝ ID が正しくなかった。)

【説明】

受信した S3F27 に対する S3F28 メッセージを送信します。
DshPortAccessRsp クラスのインスタンス rsp_class 内の応答情報から S3F28 を生成します。
trid は、この応答メッセージに対する S3F27 受信時にエンジンから与えられたトランザクション ID です。
trid がエンジンの管理の中に見つからなかった場合には、(-1)を返却します。
要求が受け入れられたら 0 を返却します。この後、応答メッセージ送信結果は特に通知されません。

15．端末表示関連メッセージの送受信

以下のメッセージとクラスがあります。

	メッセージ	クラス名	機能概略
1	S10F1,S10F2	DshS10F1Send	S10F1 送信　Terminal Request
		DshS10F2Response	S10F2 応答
2	S10F3,S10F4	DshS10F3Send	S10F3 送信　Treminal Display, Single
		DshS10F4Response	S10F4 応答
3	S10F5,S10F6	DshS10F5Send	S10F5 送信　Terminal Display, Multi-Block
		DshS10F6Response	S10F6 応答

15. 1 DshS10F1Send クラス

S10F1 メッセージを送信し、S10F2 応答メッセージを受信するためのクラスです。

15. 1. 1 コンストラクタ

オーバーロードの一覧を示します。

	名前	説明
1	public DshS10F1Send()	装置 ID=0 のｲﾝｽﾀﾝｽを生成します。
2	public DshS10F1Send(DshTermMsg info)	表示情報保存ｸﾗｽ DshTermMsg のｲﾝｽﾀﾝｽを指定して装置 ID=0 のｲﾝｽﾀﾝｽを生成します。
3	public DshS10F1Send(int eqid)	装置 ID を指定してｲﾝｽﾀﾝｽを生成します。
4	public DshS10F1Send(int eqid, DshTermMsg info))	装置 ID と表示情報保存ｸﾗｽ DshTermMsg のｲﾝｽﾀﾝｽを指定してｲﾝｽﾀﾝｽを生成します。

15. 1. 2 プロパティ

プロパティ一覧表に示します。

	名前	説明
1	public int eqid	装置 ID です。ﾃﾞﾌｫﾙﾄ値=0 です。
2	public DshTermMsg t_info	端末表示情報を保存するｸﾗｽです。 Vol-1 17.1 参照

15. 1. 3 メソッド

本クラスのメソッドは次の通りです。

	名前	説明
1	public int send_s10f1() public int send()	S10F1 ﾒｯｾｰｼﾞを送信します。
2	public int send_wait()	S10F1 ﾒｯｾｰｼﾞを送信し、S10F2 を受信します。 ﾌﾟﾛｸﾞﾗﾑは応答受信までﾌﾞﾛｯｸされます。
3	public void Dispose()	ｸﾗｽ内部で使用された資源(ﾒﾓﾘ)ｼｽﾃﾑに返却します。 他のｸﾗｽをﾌﾟﾛﾊﾟﾃｨにしている場合、そのｸﾗｽの Dispose()も行います。 3.1.3.5 と同様です。そちらを参照ください。

15. 1. 3. 1 send_s10f1(), send()

S10F1 メッセージの送信要求をします。

【構文】

```
public int send_s10f1(ref int ackc10, DshCallback.callback_s10f1 callback, uint upara)
public int send(ref int ackc10, DshCallback.callback_s10f1 callback, uint upara)
```

【引数】

ackc10

S10F2 応答メッセージの ACK 情報保存用です。コールバック関数の引数になります。

callback

S10F1 送信後、2 次メッセージ受信した時に呼び出される callback 関数です。

upara

ユーザパラメータ情報です。コールバックされる際に callback 関数の引数として渡されます。
要求とコールバック間のタグ情報として使用できます。

【戻り値】

返却値	意　味
0	要求が受け入れられた。
(-1)	要求が受け入れられなかった。

【説明】

当該インスタンスの t_info に保存されている端末表示情報から S10F1 メッセージを生成し、それを相手装置に送信するようエンジンに要求します。
要求がエンジンによって受け入れられたときは、0 を返却します。受け入れられなかった場合は(-1)を返却します。
送信後、受信した応答メッセージの ackc10 を引数にして callback 関数を呼び出します。
callback 関数からは 0 を返却してください。

【callback の構文】

```
public delegate int callback_s10f1(
    int eqid,                       // 装置 ID
    int end_status,                 // 終了状態ｺｰﾄﾞ
    ref int ackc10,                 // S10F2 に含まれる ackc10 です。
    uint upara                      // ﾕｰｻﾞﾊﾟﾗﾒｰﾀ(送信要求ﾒｿｯﾄﾞで指定された upara)
)
```

end_status に返却される値は、=0 の場合は正常終了を意味します。

end_status	意　味
0	送受信が正常に完了した。
-14	T3 ﾀｲﾑｱｳﾄを検出し、応答ﾒｯｾｰｼﾞを受信できなかった。
(-1)	送信できなかった。

15. 1. 3. 2 send_wait()

S10F1 メッセージの送信要求をし,引き続き応答メッセージも受信します。

【構文】

public int send_wait(ref int ackc10)

【引数】

ackc10

S10F2 応答メッセージの ACK 情報保存用です。

【戻り値】

返却値	意　味
0	正常に送受信した。
< 0	送受信ｴﾗｰが発生した。(送信ｴﾗｰ or T3 ﾀｲﾑｱｳﾄなど)

【説明】

当該インスタンスの t_info に保存されている端末表示情報から S10F1 メッセージを生成し、それを相手装置に送信するようエンジンに要求します。

本メソッドは送信の後、引き続き S10F2 応答メッセージを待機し、受信が終了したら、上の【戻り値】に示した値を返却します。ei=0 の場合は、受信した応答 ACK が ackc10 に保存され渡されます。

応答メッセージを受信するまでプログラムはブロック（待ち）状態になります。

送信エラー、受信エラーを検出した場合は、ei < 0 の値を返却します。(T3 ﾀｲﾑｱｳﾄも含む)

T3 タイムアウトが発生した場合、DSHDR2 HSMS ドライバーに設定されている T3 プロトコルタイムアウト時間だけブロックされることになります。

15. 2 DshS10F2Response クラス

S10F1 メッセージを送信し、S10F2 応答メッセージを受信するためのクラスです。

15. 2. 1 コンストラクタ

オーバーロードの一覧を示します。

	名前	説明
1	public DshS10F2Response()	装置 ID=0 のｲﾝｽﾀﾝｽを生成します。
2	public DshS10F2Response(int eqid)	装置 ID を指定してｲﾝｽﾀﾝｽを生成します。

15. 2. 2 プロパティ

プロパティ一覧表に示します。

	名前	説明
1	public int eqid	装置 ID です。ﾃﾞﾌｫﾙﾄ値=0 です。

15. 2. 3 メソッド

本クラスのメソッドは次の通りです。

	名前	説明
1	public int response()	S10F2 応答ﾒｯｾｰｼﾞを送信します。
2	public void Dispose()	ｸﾗｽ内部で使用された資源(ﾒﾓﾘ)ｼｽﾃﾑに返却します。 他のｸﾗｽをﾌﾟﾛﾊﾟﾃｨにしている場合、そのｸﾗｽのDispose()も行います。 3.1.3.5 と同様です。そちらを参照ください。

15. 2. 3. 1 response()

S10F2 メッセージの応答送信要求をします。

【構文】

public int response(uint trid, int ackc10)

【引数】

trid

エンジンからポーリング時に与えられたトランザクション ID です。
処理した１次メッセージとの対応を取ります。

ackc10

S10F2 メッセージのための応答 ACK です。

【戻り値】

返却値	意　味
0	要求が受け入れられた。
(-1)	要求が受け入れられなかった。(ﾄﾗﾝｻﾞｸｼｮﾝ ID が正しくなかった。)

【説明】

受信した S10F1 に対する S10F2 メッセージを送信します。

ackc10 は S10F2 に設定する ACK です。
trid は、この応答メッセージに対する S2F45 受信時にエンジンから与えられたトランザクション ID です。
trid がエンジンの管理の中に見つからなかった場合には、(-1)を返却します。
要求が受け入れられたら 0 を返却します。この後、応答メッセージ送信結果は特に通知されません。

15. 3 DshS10F3Send クラス

S10F3 メッセージを送信し、S10F4 応答メッセージを受信するためのクラスです。

15. 3. 1 コンストラクタ

オーバーロードの一覧を示します。

	名前	説明
1	public DshS10F3Send()	装置 ID=0 のｲﾝｽﾀﾝｽを生成します。
2	public DshS10F3Send(DshTermMsg info)	表示情報保存ｸﾗｽ DshTermMsg のｲﾝｽﾀﾝｽを指定して装置 ID=0 のｲﾝｽﾀﾝｽを生成します。
3	public DshS10F3Send(int eqid)	装置 ID を指定してｲﾝｽﾀﾝｽを生成します。
4	public DshS10F3Send(int eqid, DshTermMsg info))	装置 ID と表示情報保存ｸﾗｽ DshTermMsg のｲﾝｽﾀﾝｽを指定してｲﾝｽﾀﾝｽを生成します。

15. 3. 2 プロパティ

プロパティ一覧表に示します。

	名前	説明
1	public int eqid	装置 ID です。ﾃﾞﾌｫﾙﾄ値=0 です。
2	public DshTermMsg t_info	端末表示情報を保存するｸﾗｽです。 Vol-1 17.1 参照

15. 3. 3 メソッド

本クラスのメソッドは次の通りです。

	名前	説明
1	public int send_s10f3() public int send()	S10F3 ﾒｯｾｰｼﾞを送信します。
2	public int send_wait()	S10F3 ﾒｯｾｰｼﾞを送信し、S10F4 を受信します。 ﾌﾟﾛｸﾞﾗﾑは応答受信までﾌﾞﾛｯｸされます。
3	public void Dispose()	ｸﾗｽ内部で使用された資源(ﾒﾓﾘ)ｼｽﾃﾑに返却します。 他のｸﾗｽをﾌﾟﾛﾊﾟﾃｨにしている場合、そのｸﾗｽの Dispose()も行います。 3.1.3.5 と同様です。そちらを参照ください。

15. 3. 3. 1 send_s10f3(), send()

S10F3 メッセージの送信要求をします。

【構文】

```
public int send_s10f3(ref int ackc10, DshCallback.callback_s10f3 callback, uint upara)
public int send(ref int ackc10, DshCallback.callback_s10f3 callback, uint upara)
```

【引数】

ackc10

S10F4 応答メッセージの ACK 情報保存用です。コールバック関数の引数になります。

callback

S10F3 送信後、2 次メッセージ受信した時に呼び出される callback 関数です。

upara

ユーザパラメータ情報です。コールバックされる際に callback 関数の引数として渡されます。
要求とコールバック間のタグ情報として使用できます。

【戻り値】

返却値	意　味
0	要求が受け入れられた。
(-1)	要求が受け入れられなかった。

【説明】

当該インスタンスの t_info に保存されている端末表示情報から S10F3 メッセージを生成し、それを相手装置に送信するようエンジンに要求します。
要求がエンジンによって受け入れられたときは、0 を返却します。受け入れられなかった場合は(-1)を返却します。
送信後、受信した応答メッセージの ackc10 を引数にして callback 関数を呼び出します。
callback 関数からは 0 を返却してください。

【callback の構文】

```
public delegate int callback_s10f3(
    int eqid,                       // 装置 ID
    int end_status,                 // 終了状態ｺｰﾄﾞ
    ref int ackc10,                 // S10F4 に含まれる ackc10 です。
    uint upara                      // ﾕｰｻﾞﾊﾟﾗﾒｰﾀ(送信要求ﾒｿｯﾄﾞで指定された upara)
)
```

end_status に返却される値は、=0 の場合は正常終了を意味します。

end_status	意　味
0	送受信が正常に完了した。
-14	T3 ﾀｲﾑｱｳﾄを検出し、応答ﾒｯｾｰｼﾞを受信できなかった。
(-1)	送信できなかった。

15．３．３．２　send_wait()

S10F3 メッセージの送信要求をし,引き続き応答メッセージも受信します。

【構文】

public int send_wait(ref int ackc10)

【引数】

ackc10

S10F4 応答メッセージの ACK 情報保存用です。

【戻り値】

返却値	意　味
0	正常に送受信した。
＜ 0	送受信ｴﾗｰが発生した。（送信ｴﾗｰ or T3 ﾀｲﾑｱｳﾄなど）

【説明】

当該インスタンスの t_info に保存されている端末表示情報から S10F3 メッセージを生成し、それを相手装置に送信するようエンジンに要求します。

本メソッドは送信の後、引き続き S10F4 応答メッセージを待機し、受信が終了したら、上の【戻り値】に示した値を返却します。ei=0 の場合は、受信した応答 ACK が ackc10 に保存され渡されます。

受信するまではプログラムはブロック（停止）状態になります。

送信エラー、受信エラーを検出した場合は、ei ＜ 0 の値を返却します。(T3 ﾀｲﾑｱｳﾄも含む)

T3 タイムアウトが発生した場合、DSHDR2 HSMS ドライバーに設定されている T3 プロトコルタイムアウト時間だけブロックされることになります。

15. 4 DshS10F4Response クラス

S10F3 メッセージを送信し、S10F4 応答メッセージを受信するためのクラスです。

15. 4. 1 コンストラクタ

オーバーロードの一覧を示します。

	名前	説明
1	public DshS10F4Response()	装置 ID=0 のｲﾝｽﾀﾝｽを生成します。
2	public DshS10F4Response(int eqid)	装置 ID を指定してｲﾝｽﾀﾝｽを生成します。

15. 4. 2 プロパティ

プロパティ一覧表に示します。

	名前	説明
1	public int eqid	装置 ID です。ﾃﾞﾌｫﾙﾄ値=0 です。

15．４．３　メソッド

本クラスのメソッドは次の通りです。

	名前	説明
1	public int response()	S10F4 応答ﾒｯｾｰｼﾞを送信します。
2	public void Dispose()	ｸﾗｽ内部で使用された資源(ﾒﾓﾘ)ｼｽﾃﾑに返却します。 他のｸﾗｽをﾌﾟﾛﾊﾟﾃｨにしている場合、そのｸﾗｽの Dispose()も行います。 3.1.3.5 と同様です。そちらを参照ください。

15．４．３．１　response()

S10F4 メッセージの応答送信要求をします。

【構文】

public int response(uint trid, int ackc10)

【引数】

trid

エンジンからポーリング時に与えられたトランザクション ID です。
処理した１次メッセージとの対応を取ります。

ackc10

S10F4 メッセージのための応答 ACK です。

【戻り値】

返却値	意　味
0	要求が受け入れられた。
(-1)	要求が受け入れられなかった。(ﾄﾗﾝｻﾞｸｼｮﾝ ID が正しくなかった。)

【説明】

受信した S10F3 に対する S10F4 メッセージを送信します。

ackc10 は S10F4 に設定する ACK です。
trid は、この応答メッセージに対する S2F45 受信時にエンジンから与えられたトランザクション ID です。
trid がエンジンの管理の中に見つからなかった場合には、(-1)を返却します。
要求が受け入れられたら 0 を返却します。この後、応答メッセージ送信結果は特に通知されません。

15. 5 DshS10F5Send クラス

S10F5 メッセージを送信し、S10F6 応答メッセージを受信するためのクラスです。

15. 5. 1 コンストラクタ

オーバーロードの一覧を示します。

	名前	説明
1	public DshS10F5Send()	装置 ID=0 のｲﾝｽﾀﾝｽを生成します。
2	public DshS10F5Send(DshTermMultiMsg info)	表示情報保存ｸﾗｽ DshTermMultiMsg のｲﾝｽﾀﾝｽを指定して装置 ID=0 のｲﾝｽﾀﾝｽを生成します。
3	public DshS10F5Send(int eqid)	装置 ID を指定してｲﾝｽﾀﾝｽを生成します。
4	public DshS10F5Send(int eqid, DshTermMultiMsg info))	装置 ID と表示情報保存ｸﾗｽ DshTermMultiMsg のｲﾝｽﾀﾝｽを指定してｲﾝｽﾀﾝｽを生成します。

15. 5. 2 プロパティ

プロパティ一覧表に示します。

	名前	説明
1	public int eqid	装置 ID です。ﾃﾞﾌｫﾙﾄ値=0 です。
2	public DshTermMultiMsg t_info	端末表示情報を保存するｸﾗｽです。 Vol-1 17.2 参照

15. 5. 3 メソッド

本クラスのメソッドは次の通りです。

	名前	説明
1	public int send_s10f5() public int send()	S10F5 ﾒｯｾｰｼﾞを送信します。
2	public int send_wait()	S10F5 ﾒｯｾｰｼﾞを送信し、S10F6 を受信します。 ﾌﾟﾛｸﾞﾗﾑは応答受信までﾌﾞﾛｯｸされます。
3	public void Dispose()	ｸﾗｽ内部で使用された資源(ﾒﾓﾘ)ｼｽﾃﾑに返却します。 他のｸﾗｽをﾌﾟﾛﾊﾟﾃｨにしている場合、そのｸﾗｽの Dispose()も行います。 3.1.3.5 と同様です。そちらを参照ください。

15. 5. 3. 1 send_s10f5(), send()

S10F5 メッセージの送信要求をします。

【構文】

```
public int send_s10f5(ref int ackc10, DshCallback.callback_s10f5 callback, uint upara)
public int send(ref int ackc10, DshCallback.callback_s10f5 callback, uint upara)
```

【引数】

ackc10

S10F6 応答メッセージの ACK 情報保存用です。コールバック関数の引数になります。

callback

S10F5 送信後、2 次メッセージ受信した時に呼び出される callback 関数です。

upara

ユーザパラメータ情報です。コールバックされる際に callback 関数の引数として渡されます。
要求とコールバック間のタグ情報として使用できます。

【戻り値】

返却値	意　味
0	要求が受け入れられた。
(-1)	要求が受け入れられなかった。

【説明】

当該インスタンスの t_info に保存されている端末表示情報から S10F5 メッセージを生成し、それを相手装置に送信するようエンジンに要求します。
要求がエンジンによって受け入れられたときは、0 を返却します。受け入れられなかった場合は(-1)を返却します。
送信後、受信した応答メッセージの ackc10 を引数にして callback 関数を呼び出します。
callback 関数からは 0 を返却してください。

【callback の構文】

```
public delegate int callback_s10f5(
    int eqid,                       // 装置 ID
    int end_status,                 // 終了状態ｺｰﾄﾞ
    ref int ackc10,                 // S10F6 に含まれる ackc10 です。
    uint upara                      // ﾕｰｻﾞﾊﾟﾗﾒｰﾀ(送信要求ﾒｿｯﾄﾞで指定された upara)
)
```

end_status に返却される値は、=0 の場合は正常終了を意味します。

end_status	意　味
0	送受信が正常に完了した。
-14	T3 ﾀｲﾑｱｳﾄを検出し、応答ﾒｯｾｰｼﾞを受信できなかった。
(-1)	送信できなかった。

15．５．３．２　send_wait()

S10F5 メッセージの送信要求をし,引き続き応答メッセージも受信します。

【構文】

public int send_wait(ref int ackc10)

【引数】

ackc10

S10F6 応答メッセージの ACK 情報保存用です。

【戻り値】

返却値	意　味
0	正常に送受信した。
＜ 0	送受信ｴﾗｰが発生した。（送信ｴﾗｰ or T3 ﾀｲﾑｱｳﾄなど）

【説明】

当該インスタンスの t_info に保存されている端末表示情報から S10F5 メッセージを生成し、それを相手装置に送信するようエンジンに要求します。

本メソッドは送信の後、引き続き S10F6 応答メッセージを待機し、受信が終了したら、上の【戻り値】に示した値を返却します。ei=0 の場合は、受信した応答 ACK が ackc10 に保存され渡されます。

応答メッセージを受信するまでプログラムはブロック（待ち）状態になります。

送信エラー、受信エラーを検出した場合は、ei ＜ 0 の値を返却します。(T3 ﾀｲﾑｱｳﾄも含む)

T3 タイムアウトが発生した場合、DSHDR2 HSMS ドライバーに設定されている T3 プロトコルタイムアウト時間だけブロックされることになります。

15. 6 DshS10F6Response クラス

S10F5 メッセージを送信し、S10F6 応答メッセージを受信するためのクラスです。

15. 6. 1 コンストラクタ

オーバーロードの一覧を示します。

	名前	説明
1	public DshS10F6Response()	装置 ID=0 のｲﾝｽﾀﾝｽを生成します。
2	public DshS10F6Response(int eqid)	装置 ID を指定してｲﾝｽﾀﾝｽを生成します。

15. 6. 2 プロパティ

プロパティ一覧表に示します。

	名前	説明
1	public int eqid	装置 ID です。ﾃﾞﾌｫﾙﾄ値=0 です。

15．６．３　メソッド

本クラスのメソッドは次の通りです。

	名前	説明
1	public int response()	S10F6 応答ﾒｯｾｰｼﾞを送信します。
2	public void Dispose()	ｸﾗｽ内部で使用された資源(ﾒﾓﾘ)ｼｽﾃﾑに返却します。 他のｸﾗｽをﾌﾟﾛﾊﾟﾃｨにしている場合、そのｸﾗｽの Dispose()も行います。 3.1.3.5 と同様です。そちらを参照ください。

15．６．３．１　response()

S10F6 メッセージの応答送信要求をします。

【構文】

public int response(uint trid, int ackc10)

【引数】

trid

エンジンからポーリング時に与えられたトランザクション ID です。
処理した１次メッセージとの対応を取ります。

ackc10

S10F6 メッセージのための応答 ACK です。

【戻り値】

返却値	意　味
0	要求が受け入れられた。
(-1)	要求が受け入れられなかった。(ﾄﾗﾝｻﾞｸｼｮﾝ ID が正しくなかった。)

【説明】

受信した S10F5 に対する S10F6 メッセージを送信します。

ackc10 は S10F6 に設定する ACK です。
trid は、この応答メッセージに対する S10F5 受信時にエンジンから与えられたトランザクション ID です。
trid がエンジンの管理の中に見つからなかった場合には、(-1)を返却します。
要求が受け入れられたら 0 を返却します。この後、応答メッセージ送信結果は特に通知されません。

16．オンライン／オフライン、日付時刻設定メッセージ送受信

以下のメッセージとクラスがあります。

	メッセージ	クラス名	機能概略
1	S1F15,16	DshS1F15Send	S1F15 送信　Request OFF-LINE
		DshS1F16Response	S1F16 応答
2	S1F17,18	DshS1F17Send	S1F17 送信　Request ON-LINE
		DshS1F18Response	S1F18 応答
3	S2F31,32	DshS2F31Send	S2F31 送信　Date and Time Set Requst
		－	(S2F32 はｴﾝｼﾞﾝが自動応答)

16. 1 DshS1F15Send クラス

S1F15 メッセージを送信し、S1F16 応答メッセージを受信するためのクラスです。
S1F15 はオフライン要求のメッセージです。

16. 1. 1 コンストラクタ

オーバーロードの一覧を示します。

	名前	説明
1	public DshS1F15Send()	装置 ID=0 のｲﾝｽﾀﾝｽを生成します。
2	public DshS1F15Send(int eqid)	装置 ID を指定してｲﾝｽﾀﾝｽを生成します。

16. 1. 2 プロパティ

プロパティ一覧表に示します。

	名前	説明
1	public int eqid	装置 ID です。ﾃﾞﾌｫﾙﾄ値=0 です。

16. 1. 3 メソッド

本クラスのメソッドは次の通りです。

	名前	説明
1	public int send_s1f15() public int send()	S1F15 ﾒｯｾｰｼﾞを送信します。
2	public int send_wait()	S1F15 ﾒｯｾｰｼﾞを送信し、S1F16 を受信します。 ﾌﾟﾛｸﾞﾗﾑは応答受信までﾌﾞﾛｯｸされます。
3	public void Dispose()	ｸﾗｽ内部で使用された資源(ﾒﾓﾘ)ｼｽﾃﾑに返却します。 他のｸﾗｽをﾌﾟﾛﾊﾟﾃｨにしている場合、そのｸﾗｽの Dispose()も行います。 3.1.3.5 と同様です。そちらを参照ください。

16. 1. 3. 1 send_s1f15(), send()

S1F15 メッセージの送信要求をします。

【構文】

```
public int send_s1f15(ref int oflack, DshCallback.callback_s1f15 callback, uint upara)
public int send(ref int oflack, DshCallback.callback_s1f15 callback, uint upara)
```

【引数】

oflack

S1F16 応答メッセージの ACK 情報保存用です。コールバック関数の引数になります。

callback

S1F15 送信後、2 次メッセージ受信した時に呼び出される callback 関数です。

upara

ユーザパラメータ情報です。コールバックされる際に callback 関数の引数として渡されます。
要求とコールバック間のタグ情報として使用できます。

【戻り値】

返却値	意　味
0	要求が受け入れられた。
(-1)	要求が受け入れられなかった。

【説明】

S1F15 メッセージを生成し、それを相手装置に送信するようエンジンに要求します。
要求がエンジンによって受け入れられたときは、0 を返却します。受け入れられなかった場合は(-1)を返却します。
送信後、受信した応答メッセージの oflack を引数にして callback 関数を呼び出します。
callback 関数からは 0 を返却してください。

【callback の構文】

```
public delegate int callback_s1f15(
    int eqid,                       // 装置 ID
    int end_status,                 // 終了状態ｺｰﾄﾞ
    ref int oflack,                 // S1F16 に含まれる oflack です。
    uint upara                      // ﾕｰｻﾞﾊﾟﾗﾒｰﾀ(送信要求ﾒｿｯﾄﾞで指定された upara)
)
```

end_status に返却される値は、=0 の場合は正常終了を意味します。

end_status	意　味
0	送受信が正常に完了した。
-14	T3 ﾀｲﾑｱｳﾄを検出し、応答ﾒｯｾｰｼﾞを受信できなかった。
(-1)	送信できなかった。

16. 1. 3. 2 send_wait()

S1F15 メッセージの送信要求をし,引き続き応答メッセージも受信します。

【構文】

public int send_wait(ref int oflack)

【引数】

oflack

S1F16 応答メッセージの ACK 情報保存用です。

【戻り値】

返却値	意　味
0	正常に送受信した。
＜ 0	送受信ｴﾗｰが発生した。(送信ｴﾗｰ or T3 ﾀｲﾑｱｳﾄなど)

【説明】

S1F15 メッセージを生成し、それを相手装置に送信するようエンジンに要求します。

本メソッドは送信の後、引き続き S1F16 応答メッセージを待機し、受信が終了したら、上の【戻り値】に示した値を返却します。ei=0 の場合は、受信した応答 ACK が oflack に保存され渡されます。

応答メッセージを受信するまでプログラムはブロック（待ち）状態になります。

送信エラー、受信エラーを検出した場合は、ei ＜ 0 の値を返却します。(T3 ﾀｲﾑｱｳﾄも含む)

T3 タイムアウトが発生した場合、DSHDR2 HSMS ドライバーに設定されている T3 プロトコルタイムアウト時間だけブロックされることになります。

16. 2 DshS1F16Response クラス

S1F15 メッセージを送信し、S1F16 応答メッセージを受信するためのクラスです。

16. 2. 1 コンストラクタ

オーバーロードの一覧を示します。

	名前	説明
1	public DshS1F16Response()	装置 ID=0 のｲﾝｽﾀﾝｽを生成します。
2	public DshS1F16Response(int eqid)	装置 ID を指定してｲﾝｽﾀﾝｽを生成します。

16. 2. 2 プロパティ

プロパティ一覧表に示します。

	名前	説明
1	public int eqid	装置 ID です。ﾃﾞﾌｫﾙﾄ値=0 です。

16. ２. ３　メソッド

本クラスのメソッドは次の通りです。

	名前	説明
1	public int response()	S1F16 応答ﾒｯｾｰｼﾞを送信します。
2	public void Dispose()	ｸﾗｽ内部で使用された資源(ﾒﾓﾘ)ｼｽﾃﾑに返却します。 他のｸﾗｽをﾌﾟﾛﾊﾟﾃｨにしている場合、そのｸﾗｽの Dispose()も行います。 3.1.3.5 と同様です。そちらを参照ください。

16. ２. ３. １　response()

S1F16 メッセージの応答送信要求をします。

【構文】

public int response(uint trid, int oflack)

【引数】

trid

エンジンからポーリング時に与えられたトランザクション ID です。
処理した１次メッセージとの対応を取ります。

oflack

S1F16 メッセージのための応答 ACK です。

【戻り値】

返却値	意　味
0	要求が受け入れられた。
(-1)	要求が受け入れられなかった。(ﾄﾗﾝｻﾞｸｼｮﾝ ID が正しくなかった。)

【説明】

受信した S1F15 に対する S1F16 メッセージを送信します。

oflack は S1F16 に設定する ACK です。
trid は、この応答メッセージに対する S1F15 受信時にエンジンから与えられたトランザクション ID です。
trid がエンジンの管理の中に見つからなかった場合には、(-1)を返却します。
要求が受け入れられたら 0 を返却します。この後、応答メッセージ送信結果は特に通知されません。

16. 3 DshS1F17Send クラス

S1F17 メッセージを送信し、S1F18 応答メッセージを受信するためのクラスです。
S1F17 はオンライン要求のメッセージです。

16. 3. 1 コンストラクタ

オーバーロードの一覧を示します。

	名前	説明
1	public DshS1F17Send()	装置 ID=0 のｲﾝｽﾀﾝｽを生成します。
2	public DshS1F17Send(int eqid)	装置 ID を指定してｲﾝｽﾀﾝｽを生成します。

16. 3. 2 プロパティ

プロパティ一覧表に示します。

	名前	説明
1	public int eqid	装置 ID です。ﾃﾞﾌｫﾙﾄ値=0 です。

16. 3. 3 メソッド

本クラスのメソッドは次の通りです。

	名前	説明
1	public int send_s1f17() public int send()	S1F17 ﾒｯｾｰｼﾞを送信します。
2	public int send_wait()	S1F17 ﾒｯｾｰｼﾞを送信し、S1F18 を受信します。 ﾌﾟﾛｸﾞﾗﾑは応答受信までﾌﾞﾛｯｸされます。
3	public void Dispose()	ｸﾗｽ内部で使用された資源(ﾒﾓﾘ)ｼｽﾃﾑに返却します。 他のｸﾗｽをﾌﾟﾛﾊﾟﾃｨにしている場合、そのｸﾗｽの Dispose()も行います。 3.1.3.5 と同様です。そちらを参照ください。

16．３．３．１　send_s1f17(), send()

S1F17 メッセージの送信要求をします。

【構文】

```
public int send_s1f17(ref int onlack, DshCallback.callback_s1f17 callback, uint upara)
public int send(ref int onlack, DshCallback.callback_s1f17 callback, uint upara)
```

【引数】

onlack

S1F18 応答メッセージの ACK 情報保存用です。コールバック関数の引数になります。

callback

S1F17 送信後、2 次メッセージ受信した時に呼び出される callback 関数です。

upara

ユーザパラメータ情報です。コールバックされる際に callback 関数の引数として渡されます。
要求とコールバック間のタグ情報として使用できます。

【戻り値】

返却値	意　味
0	要求が受け入れられた。
(-1)	要求が受け入れられなかった。

【説明】

S1F17 メッセージを生成し、それを相手装置に送信するようエンジンに要求します。
要求がエンジンによって受け入れられたときは、0 を返却します。受け入れられなかった場合は(-1)を返却します。
送信後、受信した応答メッセージの onlack を引数にして callback 関数を呼び出します。
callback 関数からは 0 を返却してください。

【callback の構文】

```
public delegate int callback_s1f17(
    int eqid,                       // 装置 ID
    int end_status,                 // 終了状態ｺｰﾄﾞ
    ref int onlack,                 // S1F18 に含まれる onlack です。
    uint upara                      // ﾕｰｻﾞﾊﾟﾗﾒｰﾀ(送信要求ﾒｿｯﾄﾞで指定された upara)
)
```

end_status に返却される値は、=0 の場合は正常終了を意味します。

end_status	意　味
0	送受信が正常に完了した。
-14	T3 ﾀｲﾑｱｳﾄを検出し、応答ﾒｯｾｰｼﾞを受信できなかった。
(-1)	送信できなかった。

16. 3. 3. 2 send_wait()

S1F17 メッセージの送信要求をし,引き続き応答メッセージも受信します。

【構文】

public int send_wait(ref int onlack)

【引数】

onlack

S1F18 応答メッセージの ACK 情報保存用です。

【戻り値】

返却値	意　味
0	正常に送受信した。
< 0	送受信ｴﾗｰが発生した。(送信ｴﾗｰ or T3 ﾀｲﾑｱｳﾄなど)

【説明】

S1F17 メッセージを生成し、それを相手装置に送信するようエンジンに要求します。

本メソッドは送信の後、引き続き S1F18 応答メッセージを待機し、受信が終了したら、上の【戻り値】に示した値を返却します。ei=0 の場合は、受信した応答 ACK が onlack に保存され渡されます。

応答メッセージを受信するまでプログラムはブロック（待ち）状態になります。

送信エラー、受信エラーを検出した場合は、ei < 0 の値を返却します。(T3 ﾀｲﾑｱｳﾄも含む)

T3 タイムアウトが発生した場合、DSHDR2 HSMS ドライバーに設定されている T3 プロトコルタイムアウト時間だけブロックされることになります。

16. 4 DshS1F18Response クラス

S1F17 メッセージを送信し、S1F18 応答メッセージを受信するためのクラスです。

16. 4. 1 コンストラクタ

オーバーロードの一覧を示します。

	名前	説明
1	public DshS1F18Response()	装置 ID=0 のｲﾝｽﾀﾝｽを生成します。
2	public DshS1F18Response(int eqid)	装置 ID を指定してｲﾝｽﾀﾝｽを生成します。

16. 4. 2 プロパティ

プロパティ一覧表に示します。

	名前	説明
1	public int eqid	装置 ID です。ﾃﾞﾌｫﾙﾄ値=0 です。

16．４．３　メソッド

本クラスのメソッドは次の通りです。

	名前	説明
1	public int response()	S1F18 応答ﾒｯｾｰｼﾞを送信します。
2	public void Dispose()	ｸﾗｽ内部で使用された資源(ﾒﾓﾘ)ｼｽﾃﾑに返却します。 他のｸﾗｽをﾌﾟﾛﾊﾟﾃｨにしている場合、そのｸﾗｽの Dispose()も行います。 3.1.3.5 と同様です。そちらを参照ください。

16．４．３．１　response()

S1F18 メッセージの応答送信要求をします。

【構文】

public int response(uint trid, int onlack)

【引数】

trid

エンジンからポーリング時に与えられたトランザクション ID です。
処理した１次メッセージとの対応を取ります。

onlack

S1F18 メッセージのための応答 ACK です。

【戻り値】

返却値	意　味
0	要求が受け入れられた。
(-1)	要求が受け入れられなかった。(ﾄﾗﾝｻﾞｸｼｮﾝ ID が正しくなかった。)

【説明】

受信した S1F17 に対する S1F18 メッセージを送信します。

onlack は S1F18 に設定する ACK です。
trid は、この応答メッセージに対する S1F17 受信時にエンジンから与えられたトランザクション ID です。
trid がエンジンの管理の中に見つからなかった場合には、(-1)を返却します。
要求が受け入れられたら 0 を返却します。この後、応答メッセージ送信結果は特に通知されません。

16. 5 DshS2F31Send クラス

S2F31 メッセージを送信し、S2F32 応答メッセージを受信するためのクラスです。
S2F31 は日付時刻設定要求のメッセージです。

16. 5. 1 コンストラクタ

オーバーロードの一覧を示します。

	名前	説明
1	public DshS2F31Send()	装置 ID=0 のｲﾝｽﾀﾝｽを生成します。
	public DshS2F31Send(string dtime)	日付時刻を指定して装置 ID=0 のｲﾝｽﾀﾝｽを生成します。 dtime のﾌｫｰﾏｯﾄは次の通りです。 YYYYMMDDhhmmsscc です、 年　月日時分秒 10ms 単位 桁数　4　2 2 2 2 2 2 cc は 10ms 単位です。
2	public DshS2F31Send(int eqid)	装置 ID を指定してｲﾝｽﾀﾝｽを生成します。
	public DshS2F31Send(int eqid, string dtime)	装置 ID と日付時刻を指定してｲﾝｽﾀﾝｽを生成します。

16. 5. 2 プロパティ

プロパティ一覧表に示します。

	名前	説明
1	public int eqid	装置 ID です。ﾃﾞﾌｫﾙﾄ値=0 です。
2	public string date_time	日付時刻ﾃﾞｰﾀです。 dtime のﾌｫｰﾏｯﾄは次の通りです。 YYYYMMDDhhmmsscc です、 年　月日時分秒 10ms 単位 桁数　4　2 2 2 2 2 2 cc は 10ms 単位です。

16. ５. ３　メソッド

本クラスのメソッドは次の通りです。

	名前	説明
1	public int send_s2f31() public int send()	S2F31 ﾒｯｾｰｼﾞを送信します。
2	public int send_wait()	S2F31 ﾒｯｾｰｼﾞを送信し、S2F32 を受信します。 ﾌﾟﾛｸﾞﾗﾑは応答受信までﾌﾞﾛｯｸされます。
3	public void Dispose()	ｸﾗｽ内部で使用された資源(ﾒﾓﾘ)ｼｽﾃﾑに返却します。 他のｸﾗｽをﾌﾟﾛﾊﾟﾃｨにしている場合、そのｸﾗｽの Dispose()も行います。 3.1.3.5 と同様です。そちらを参照ください。

16. ５. ３. １　set_info()

【構文】

public void set_info(string dtime)

【引数】

dtime

日付時刻を設定します。

dtime のﾌｫｰﾏｯﾄは次の通りです。

YYYYMMDDhhmmsscc です、

年　月日時分秒 10ms 単位

桁数　4　２２２２２２

cc は 10ms 単位です。

【戻り値】

なし。

【説明】

dtime を date_time プロパティに設定します。

16. 5. 3. 2 send_s2f31(), send()

S2F31 メッセージの送信要求をします。

【構文】

```
public int send_s2f31(ref int tiack, DshCallback.callback_s2f31 callback, uint upara)
public int send(ref int tiack, DshCallback.callback_s2f31 callback, uint upara)
```

【引数】

tiack

S2F32 応答メッセージの ACK 情報保存用です。コールバック関数の引数になります。

callback

S2F31 送信後、2 次メッセージ受信した時に呼び出される callback 関数です。

upara

ユーザパラメータ情報です。コールバックされる際に callback 関数の引数として渡されます。
要求とコールバック間のタグ情報として使用できます。

【戻り値】

返却値	意　味
0	要求が受け入れられた。
(-1)	要求が受け入れられなかった。

【説明】

当該インスタンスの date_time（日付時刻データ）から S2F31 メッセージを生成し、それを相手装置に送信するようエンジンに要求します。
要求がエンジンによって受け入れられたときは、0 を返却します。受け入れられなかった場合は(-1)を返却します。
送信後、受信した応答メッセージの tiack を引数にして callback 関数を呼び出します。
callback 関数からは 0 を返却してください。

【callback の構文】

```
public delegate int callback_s2f31(
    int eqid,                       // 装置 ID
    int end_status,                 // 終了状態ｺｰﾄﾞ
    ref int tiack,                  // S2F32 に含まれる tiack です。
    uint upara                      // ﾕｰｻﾞﾊﾟﾗﾒｰﾀ(送信要求ﾒｿｯﾄﾞで指定された upara)
)
```

end_status に返却される値は、=0 の場合は正常終了を意味します。

end_status	意　味
0	送受信が正常に完了した。
-14	T3 ﾀｲﾑｱｳﾄを検出し、応答ﾒｯｾｰｼﾞを受信できなかった。
(-1)	送信できなかった。

16. 5. 3. 3 send_wait()

S2F31 メッセージの送信要求をし,引き続き応答メッセージも受信します。

【構文】

public int send_wait(ref int tiack)

【引数】

tiack

S2F32 応答メッセージの ACK 情報保存用です。

【戻り値】

返却値	意　味
0	正常に送受信した。
＜ 0	送受信ｴﾗｰが発生した。(送信ｴﾗｰ or T3 ﾀｲﾑｱｳﾄなど)

【説明】

当該インスタンスの date_time（日付時刻データ）から S2F31 メッセージを生成し、それを相手装置に送信するようエンジンに要求します。

本メソッドは送信の後、引き続き S2F32 応答メッセージを待機し、受信が終了したら、上の【戻り値】に示した値を返却します。ei=0 の場合は、受信した応答 ACK が tiack に保存され渡されます。

応答メッセージを受信するまでプログラムはブロック（待ち）状態になります。

送信エラー、受信エラーを検出した場合は、ei ＜ 0 の値を返却します。(T3 ﾀｲﾑｱｳﾄも含む)

T3 タイムアウトが発生した場合、DSHDR2 HSMS ドライバーに設定されている T3 プロトコルタイムアウト時間だけブロックされることになります。

17. ユーザ固有メッセージの送受信

ユーザ固有メッセージの送受信のために次の３つのメソッドがDshEngineに準備されています。

- send_request() メソッド　　(非ブロックモードでの送受信)
- send_request_wait()メソッド(ブロックモードでの送受信)
- send_response()メソッド

これらは、static なメソッドになっていますので、DshEquipment.send_request(..)のように呼び出してください。

17. 1　send_request()

GEM でサポートされていない、あるいは、エンジンで標準サポートされていないユーザ独自の 1 次メッセージの送信を行います。

構文】

```
public static int send_request(ref DSHMSG smsg, ref DSHMSG rmsg,
                               DshCallback.callback_send_request callback, uint upara)
```

【引数】

smsg

SECS-Ⅱ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。
メッセージの組立ては、ユーザが行います。

rmsg

応答 2 次メッセージ情報を格納するための構造体のポインタです。

callback

send_request()が終了したときに呼び出されるコールバック関数(イベントハンドラー)です。

upara

ユーザが callback で指定した関数が呼び出された際に、引数で渡して欲しいデータです。

【戻り値】

返却値	意　味
0	正常に要求が受付された。
< 0	send_request 要求に失敗した。

【説明】

本メソッドはユーザが組み立てた smsg に保存されている 1 次メッセージの送信を行います。そして、受信した応答メッセージを rmsg に格納し、callback でユーザに報せます。

本メソッドは、static の関数として準備されていますので、インスタンスを生成しないで次のコーディングで使用できます。

```
DshEngine.send_request(.... )
```

send_request()メソッドがエンジンに受け付けられた場合、返却値 0 で戻ります。受け入れられなかった場合は(-1)が返却されます。

そして、送信でき、応答メッセージを受信できたら、callback によって指定された関数を呼び出します。その際、送信結果と受信メッセージを引数として与えます。

ユーザは、本メソッドを実行する前に、smsg 構造体内に 1 次メッセージを組立てセットしなければなりません。ストリーム、ファンクション、そしてテキストなどです。詳しくは、プログラミング例を参考にしてください。

【終了通知関数】

send_request()メソッドに対する callback の書式は、DshCallback クラスに次のように定義されています。

```
public delegate int callback_send_request(ref DSHMSG rmsg, int end_status, uint upara);
```

rmsg　　：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。
　　send_request()メソッドの引数で与えられた構造体のポインタです。
end_status ：send_request()の処理結果です。 0 であれば正常に終了です。
　　(-1)であればエラー終了です。
upara　　：send_request()メソッドで与えられた upara の値が渡されます。

【例】 S7F5 を送信し、S7F6 を受信する処理

```
private void send_s7f5( string ppid )
    int ei = 0;
    DSHMSG smsg = new DSHMSG();                          // 送信用構造体
    DSHMSG rmsg = new DSHMSG();                          // 受信用構造体
    IntPtr buff = Marshal.AllocCoTaskMem(1024);          // ﾒｯｾｰｼﾞ Text 用ﾊﾞｯﾌｧ
    smsg.wbit = 1;                                       // Wait bit=1
    smsg.stream = 7;                                     // S7
    smsg.function = 5;                                   // F5
    while (true)
    {
        smsg.buffer = buff;                              // buff ptr 設定
        smsg.length = 1024;                              // buff size 設定

        HSMS.D_InitItemPut(ref smsg);                    // smsg 構造体の put のための初期化

        ei = HSMS.D_PutItem(ref smsg, HSMS.ICODE_L, IntPtr.Zero, 1);     // L-1 ｾｯﾄ
        if (ei < 0) break;

        ei = HSMS.D_PutItem(ref smsg, HSMS.ICODE_A, .ppid, ppid.Length); // PPID  ｾｯﾄ
        break;
    }
    if (ei < 0)
    {
        DshLog.log("  !! Message setup error¥n");       // 組立てｴﾗｰ
        return;
    }
    ei = DshEquipment.send_request(ref smsg, ref rmsg, cback_request_s7f5, 705); // 送信
    if (ei < 0)
    {
        DshLog.log("  !! send_request() error¥n");
    }
    Marshal.FreeCoTaskMem(buff);                         // buff ﾒﾓﾘ開放
}
```

送受信完了で呼び出される callback 関数

```
private static int callback_send_request_s7f5(int end_status, ref DSHMSG rmsg, uint upara)
{
    string ppid ="";
    string ppbody ="";
    IntPtr ptr = Marshal.AllocCoTaskMem( 1024 );                          // dataitem 値取得用ﾊﾞｯﾌｧ

    DshLog.log("  ! send_request callback() end_status = " + end_status.ToString() + "¥r¥n");
    if (end_status == 0)
    {
        formid.fm.OutLog(" APP S" + rmsg.stream.ToString() + "F" + rmsg.function.ToString() + " rcvd");
        formid.fm.OutLog("     length=" + rmsg.length.ToString());
        HSMS.D_InitItemGet(ref rmsg);                                     // rmsg 初期化
        int n = 0;
        int ei = 0;
        while( true )
        {
            n = HSMS.D_GetItem(ref rmsg, HSMS.ICODE_L, IntPtr.Zero, 0);    / L-2
            if ( n != 2 )
            {
                ei = (-1); break;
            }
            n = HSMS.D_GetItem( ref rmsg, HSMS.ICODE_A, ptr, 1024 );       // PPID
            if ( n < 0 )
            {
                ei = (-1); break;
            }
            ppid = DshLib.DshPtrToString(ptr, 1024, n);
            n = HSMS.D_GetItem(ref rmsg, HSMS.ICODE_A, ptr, 1024);         // PPBODY
            if (n < 0)
            {
                ei = (-1); break;
            }
            ppbody = DshLib.DshPtrToString(ptr, 1024, n);
            break;        }
        if (ei == 0)
        {
            DshLog.log("    ppid     = " + ppid + "¥n");
            DshLog.log("    ppbody   = " + ppbody + "¥n");
        }
        else
        {
            DshLog.log("  !! Message format error" + "¥n");
        }
        break;
    }
    Marshal.FreeCoTaskMem(ptr);                                           // ptr ﾒﾓﾘ開放
    return 0;
}
//----- callback 用 instance
private static DshCallback.callback_send_request cback_request_s7f5 =
                              new DshCallback.callback_send_request(callback_send_request_s7f5);
```

17. 2　send_request_wait()

GEM でサポートされていない、あるいは、エンジンで標準サポートされていないユーザ独自の 1 次メッセージをブロックモードで送信を行います。

構文】

public static int send_request_wait(ref DSHMSG smsg, ref DSHMSG rmsg)

【引数】

smsg

SECS-Ⅱ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。
メッセージの組立ては、ユーザが行います。

rmsg

応答 2 次メッセージ情報を格納するための構造体のポインタです。

【戻り値】

返却値	意　味
0	正常に要求が受付された。
＜ 0	send_request_wait 送受信に失敗した。

【説明】

本メソッドはユーザが組み立てた smsg に保存されている 1 次メッセージの送信を行います。そして、受信した応答メッセージを rmsg に格納します。

本メソッドは、send_request()との違いは、send_request()は非ブロックモードの送受信であり、send_request_wait()は、ブロックモードでの送受信になります。
ブロックモードでは、応答メッセージの受信までプログラムはブロック（待ち）状態になります。

本メソッドは、static の関数として準備されていますので、インスタンスを生成しないで次のコーディングで使用できます。

```
DshEngine.send_request_wait(.... )
```

send_request_wait()メソッドが正常に完了したときは返却値 0 で戻ります。異常を検出した場合は負の値(＜ 0)が返却されます。

ユーザは、本メソッドを実行する前に、smsg 構造体内に 1 次メッセージを組立てセットしなければなりません。ストリーム、ファンクション、そしてテキストなどです。詳しくは、次ページのプログラミング例を参考にしてください。

それから、受信メッセージの処理が終了したら、rmsg の中にメッセージ格納用に使用されたメモリの開放を必ず実行してください。実行しないとメモリリークが発生します。

DshEquipment.dsh_free_buffer() メソッドを使って開放します。

```
DshEquipment.dsh_free_buffer( ref rmsg );   // rmsg 内のﾊﾞｯﾌｧﾒﾓﾘを開放する。
```

【例】　S7F5 を送信し、S7F6 を受信する処理

```
private void send_s7f5_wait( string ppid )
     int ei = 0;
     DSHMSG smsg = new DSHMSG();                          // 送信用構造体
     DSHMSG rmsg = new DSHMSG();                          // 受信用構造体
     IntPtr buff = Marshal.AllocCoTaskMem(1024);                  // ﾒｯｾｰｼﾞ Text 用ﾊﾞｯﾌｧ
     smsg.wbit = 1;                                               // Wait bit=1
     smsg.stream = 7;                                     // S7
     smsg.function = 5;                                   // F5
     while (true){
         smsg.buffer = buff;                                      // buff ptr 設定
         smsg.length = 1024;                                      // buff size 設定
         HSMS.D_InitItemPut(ref smsg);                            // smsg 構造体の put のための初期化
         ei = HSMS.D_PutItem(ref smsg, HSMS.ICODE_L, IntPtr.Zero, 1);     // L-1 ｾｯﾄ
         if (ei < 0) break;
         ei = HSMS.D_PutItem(ref smsg, HSMS.ICODE_A, .ppid, ppid.Length); // PPID ｾｯﾄ
         break;
     }
     if (ei < 0){
         DshLog.log("  !! Message setup error¥n");                        // 組立てｴﾗｰ
         return;
     }
     ei = DshEquipment.send_request_wait(ref smsg, ref rmsg);             // 送受信
     if (ei < 0){
         DshLog.log("  !! send_request_wait() error¥n");
     }else{
        IntPtr ptr = Marshal.AllocCoTaskMem(1024);
        HSMS.D_InitItemGet(ref rmsg);                                     // rmsg 初期化
        int n = 0; int ei = 0;
        while( true ){
            n = HSMS.D_GetItem(ref rmsg, HSMS.ICODE_L, IntPtr.Zero, 0);    // L-2
            if ( n != 2 ){
                ei = (-1); break;
            }
            n = HSMS.D_GetItem( ref rmsg, HSMS.ICODE_A, ptr, 1024 );       // PPID
            if ( n < 0 ){
                ei = (-1); break;
            }
            ppid = DshLib.DshPtrToString(ptr, 1024, n);
            n = HSMS.D_GetItem(ref rmsg, HSMS.ICODE_A, ptr, 1024);         // PPBODY
            if (n < 0){
                ei = (-1); break;
            }
            ppbody = DshLib.DshPtrToString(ptr, 1024, n);
            break;
        }
        if (ei == 0) {
            DshLog.log("    ppid     = " + ppid + "¥n");
            DshLog.log("    ppbody   = " + ppbody + "¥n");
        }else{
            DshLog.log("  !! Message format error" + "¥n");
        }
        DshEquipment.dsh_free_msg_buffer( ref rmsg );  // !!  これを必ず実行すること。
    }
    Marshal.FreeCoTaskMem(ptr);                                           // ptr ﾒﾓﾘ開放
    Marshal.FreeCoTaskMem(buff);                                          // buff ﾒﾓﾘ開放
}
```

17. 3　send_response()

GEM でサポートされていない、あるいは、エンジンで標準サポートされていないユーザ独自の 2 次メッセージの応答送信を行います。

構文】

```
public static int send_response(uint trid, ref DSHMSG rmsg)
```

【引数】

rrmsg

SECS-Ⅱ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。
メッセージの組立ては、ユーザが行います。

【戻り値】

返却値	意　味
0	正常に要求が受付された。
＜ 0	send_response 要求に失敗した。

【説明】

本メソッドはユーザが組み立てた rmsg に保存されている 2 次メッセージの送信を行います。

本メソッドは、static の関数として準備されていますので、インスタンスを生成いないで次のコーディングで使用できます。

```
DshEngine.send_response(.... )
```

send_response()メソッドがエンジンに受け付けられた場合、返却値 0 で戻ります。受け入れられなかった場合は(-1)が返却されます。

ユーザは、本メソッドを実行する前に、rmsg 構造体内に 2 次メッセージを組立てセットしなければなりません。ストリーム、ファンクション、そしてテキストなどです。詳しくは、プログラミング例を参考にしてください。

本メソッドは、エンジンに応答メッセージの送信を要求し、それが受付されてから後、送信が終了しても特に通知はありません。

【例】　S10F3 を受信した後、S10F4 を send_response を使って送信します。

```
public static void s10f1(int eqid, uint trid, ref DSHMSG smsg)
{

   // ＜ここで、S10F3 の処理を行う。
   // 以下、S10F4 ﾒｯｾｰｼﾞを準備し、応答送信します。

    int ackc10 = 0;

    DSHMSG rmsg = new DSHMSG();                                    // 2 ｼﾞﾒｯｾｰｼﾞ情報格納構造体
    rmsg.stream = 10;                                              // S10
    rmsg.function = 4;                                             // F4
    rmsg.wbit = 0;                                                 // w-bit = 0
    IntPtr buff = Marshal.AllocCoTaskMem(128);                     // text 用ﾊﾞｯﾌｧﾒﾓﾘ確保
    rmsg.buffer = buff;
    rmsg.length = 128;
    HSMS.D_InitItemPut(ref rmsg);                                  // rmsg 構造体初期化
    HSMS.D_PutItem(ref rmsg, HSMS.ICODE_B, ref ackc10, 1);                 // ackc10 を設定

    DshEngine.send_response(trid, ref rmsg);                        // 応答送信

    Marshal.FreeCoTaskMem(buff);                                   //text 用ﾊﾞｯﾌｧﾒﾓﾘ開放
    return;
}
```