DSHEng4　装置通信エンジン（GEM+GEM300)

ソフトウェア・パッケージ

APP インタフェース
ライブラリ関数説明書

(C, C++, .Net-Vb,C#)

VOL-１２／１５

３．20　PRJ プロセスジョブ情報アクセスサービス関数

２００９年7月

株式会社データマップ

文書番号　DSHEng4-09-3032C-00

[取り扱い注意]

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

【改訂履歴】

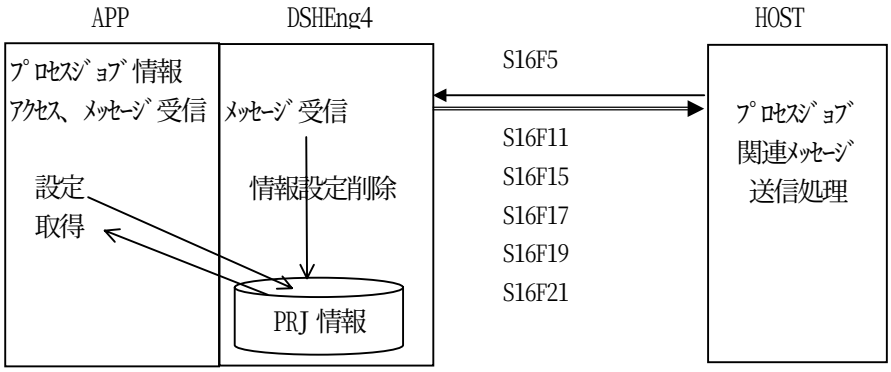
番号	改訂日付	項 目	概 略
1.	2009.7	初版	

目次

(VOL－１３に続く)

3．20　PRJ プロセスジョブ情報アクセス、送信サービス関数

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



（1）情報アクセスと送信 API 関数

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

	API 関数名	機能
1	EngAllocPrjInfo()	ﾌﾟﾛｾｽｼﾞｮﾌﾞ領域を割当て登録します。
2	EngSetPrjInfo()	ﾌﾟﾛｾｽｼﾞｮﾌﾞ情報を設定・変更します。
3	EngGetPrjInfo()	PRJID 指定でﾌﾟﾛｾｽｼﾞｮﾌﾞ情報を取得します。
4	EngGetPrjInfoX()	ﾌﾟﾛｾｽｼﾞｮﾌﾞｲﾝﾃﾞｸｽ指定でﾌﾟﾛｾｽｼﾞｮﾌﾞ情報を取得します。
5	EngDelPrjInfo()	PRJID 指定でﾌﾟﾛｾｽｼﾞｮﾌﾞ情報を削除します。
6	EngDelPrjInfoX()	ﾌﾟﾛｾｽｼﾞｮﾌﾞｲﾝﾃﾞｸｽ指定でﾌﾟﾛｾｽｼﾞｮﾌﾞ情報を削除します。
7	EngGetPrjId()	指定したﾌﾟﾛｾｽｼﾞｮﾌﾞｲﾝﾃﾞｸｽの PRJID を取得します。
8	EngGetPrjIdIndex()	指定した PRJID の情報ｲﾝﾃﾞｸｽを取得します。
9	EngSetPrjIdStatus()	PRJID 指定で PRJID の状態を設定します。
10	EngSetPrjIdStatusX()	ﾌﾟﾛｾｽｼﾞｮﾌﾞｲﾝﾃﾞｸｽ指定で PRJID の状態を設定します。
11	EngGetPrjIdStatus()	PRJID 指定で PRJID の状態を取得します。
12	EngGetPrjIdStatusX()	ﾌﾟﾛｾｽｼﾞｮﾌﾞｲﾝﾃﾞｸｽ指定で PRJID の状態を取得します。
13	EngGetPrjList()	ﾌﾟﾛｾｽｼﾞｮﾌﾞ ID の一覧ﾘｽﾄを取得します。

PRJID インデクスは、DSHEng4 が管理する各 PRJID 領域の番号です。このインデクスの値は、EngAllocPrjInfo()関数実行時に DSHEng4 によって割当てられ、APP に渡されます。また、ﾌﾟﾛｾｽｼﾞｮﾌﾞの取得時に、情報格納構造体のメンバー、 index に設定されます。

（2）ライブラリ関数

他に APP が使用できるプロセスジョブ情報処理用 API 関数として、以下の関数があります。

	API 関数名	機能
1	DshDecodeS16F5()	S16F5 に含まれるﾌﾟﾛｾｽｼﾞｮﾌﾞｺﾏﾝﾄﾞ情報を TPRJ_CMD_INFO 構造体内にﾃﾞｺｰﾄﾞするための関数です。
2	DshFreeTPRJ_CMD_INFO()	TPRJ_CMD_INFO 内に確保使用したﾒﾓﾘを開放します。
3	DshCopyTPRJ_CMD_INFO()	TPRJ_CMD_INFO 構造体を別の構造体にｺﾋﾟｰします。
4	DshInitTPRJ_CMD_ERR_INFO()	TPRJ_ERR_INFO 構造体内の初期設定を行います。S16F6 応答ﾒｯｾｰｼﾞ作成用)
5	DshFreeTPRJ_CMD_ERR_INFO()	TPRJ_ERR_INFO 構造体内の err_list に確保使用したﾒﾓﾘを開放します。
6	DshMakeS16F5Response()	S16F6 応答ﾒｯｾｰｼﾞを TPRJ_ERR_INFO 構造体内の応答情報から生成します。
7	DshDecodeS16F11()	S16F11 に含まれるﾌﾟﾛｾｽｼﾞｮﾌﾞ情報を TPRJ_INFO 構造体内にﾃﾞｺｰﾄﾞするための関数です。
8	DshFreeTPRJ_INFO()	TPRJ_INFO 構造体内に使用されているﾒﾓﾘを開放します。
9	DshCopyTPRJ_INFO()	TPRJ_INFO 構造体を別の構造体にｺﾋﾟｰします。
10	DshInitPrjInfo()	ﾌﾟﾛｾｽｼﾞｮﾌﾞ情報構造体 TPRJ_INFO を初期設定します。
11	DshPutPrjRcpInfo()	ﾌﾟﾛｾｽｼﾞｮﾌﾞ情報構造体 TPRJ_INFO にﾚｼﾋﾟ情報を追加設定します。
12	DshPutPrjCarInfo()	ﾌﾟﾛｾｽｼﾞｮﾌﾞ情報構造体 TPRJ_INFO にｷｬﾘｱ情報を 1 個追加設定します。
13	DshPutPrjMid()	ﾌﾟﾛｾｽｼﾞｮﾌﾞ情報構造体 TPRJ_INFO に MID(ﾏﾃﾘｱﾙ ID)を 1 個追加設定します。
14	DshPutPrjPauseCeid()	ﾌﾟﾛｾｽｼﾞｮﾌﾞ情報構造体 TPRJ_INFO に PAUSE CEID を 1 個追加設定します。
15	DshInitTPRJ_ERR_INFO()	ﾌﾟﾛｾｽｼﾞｮﾌﾞ生成応答情報のための TPRJ_ERR_INFO 構造体内の初期設定を行います。S16F12 応答ﾒｯｾｰｼﾞ用です。
16	DshPutTPRJ_ERR_PRJID()	ﾌﾟﾛｾｽｼﾞｮﾌﾞ生成応答情報の TPRJ_ERR_INFO 構造体内にﾌﾟﾛｾｽｼﾞｮﾌﾞ ID を設定します。
17	DshPutTPRJ_ERR_INFO()	ﾌﾟﾛｾｽｼﾞｮﾌﾞ生成応答情報の TPRJ_ERR_INFO 構造体内にｴﾗｰ情報を設定します。
18	DshFreeTPRJ_ERR_INFO()	TPRJ_ERR_INFO 構造体内の err_list に確保使用したﾒﾓﾘを開放します。
19	DshMakeS16F11Response()	S6F11 の応答ﾒｯｾｰｼﾞを生成します。 TPRJ_ERR_INFO を使用します。
20	DshDecodeS16F15 ()	S16F15 ﾒｯｾｰｼﾞから 1 個以上の PRJ 情報をﾌﾟﾛｾｽｼﾞｮﾌﾞ情報ﾘｽﾄの TPRJ_LIST 構造体にﾃﾞｺｰﾄﾞします。
21	DshFreeTPRJ_INFO_LIST()	ﾌﾟﾛｾｽｼﾞｮﾌﾞ情報ﾘｽﾄの TPRJ_LIST 構造体内に確保使用したﾒﾓﾘを開放します。

22	DshCopyTPRJ_LIST()	ﾌﾟﾛｾｽｼﾞｮﾌﾞ情報ﾘｽﾄの TPRJ_LIST 構造体を別の構造体にｺﾋﾟｰします。
23	DshInitTPRJ_LIST()	S16F15 ﾒｯｾｰｼﾞ送信のための TPRJ_LIST 構造体を初期化します。
24	DshAddTPRJ_LIST()	S16F15 ﾒｯｾｰｼﾞ送信のための TPRJ_LIST 構造体に TPRJ_INFO 構造体情報を１個追加します。(TPRJ_INFO は１個のﾌﾟﾛｾｽｼﾞｮﾌﾞ情報を含む構造体)
25	DshDecodeS16F17()	TPRJ_ DEQ_INFO 構造体のﾌﾟﾛｾｽｼﾞｮﾌﾞﾃﾞｷｭｰ情報を S16F17 ﾒｯｾｰｼﾞにｴﾝｺｰﾄﾞするための関数です。
26	DshMakeS16F15Response()	S16F15 に対する応答ﾒｯｾｰｼﾞ S16F16 を TPRJ_ERR_INFO 内の応答情報から生成します。
27	DshFreeTPRJ_DEQ_INFO()	TPRJ_DEQ_INFO 構造体に使用されているﾒﾓﾘを開放します。
28	DshInitTPRJ_DEQ_ERR_INFO()	ﾌﾟﾛｾｽｼﾞｮﾌﾞﾃﾞｷｭｰ応答情報のための TPRJ_DEQ_ERR_INFO 構造体内の初期設定を行います。S16F18 応答ﾒｯｾｰｼﾞ用です。
29	DshFreeTPRJ_DEQ_ERR_INFO()	TPRJ_DEQ_ERR_INFO 構造体に使用されているﾒﾓﾘを開放します。
30	DshMakeS16F17Response()	S16F17 に対する応答ﾒｯｾｰｼﾞ S16F1８を TPRJ_DEQ_ERR_INFO 内の応答情報から生成します。
31	DshFreeTPRJ_STATE_LIST()	S16F20 で得られたﾌﾟﾛｾｽｼﾞｮﾌﾞのﾘｽﾄ格納用構造体内で使用されたﾒﾓﾘを開放します。

（3）ユーザ作成ライブラリ関数

	ﾗｲﾌﾞﾗﾘ関数名	機能
1	DshResponseS16F6()	S16F6 プロセスジョブコマンド要求応答メッセージ
2	DshResponseS16F12()	S16F12 プロセスジョブ生成要求応答メッセージ
3	DshResponseS16F16()	S16F16 プロセスジョブマルチ生成要求応答メッセージ
4	DshResponseS16F18()	S16F18 プロセスジョブデキュー要求応答メッセージ

3．20．1　使用する情報格納構造体

プロセスジョブ情報を操作する関数は、情報格納のための TPRJ_INFO 構造体を使用します。
プロセスジョブに関連する構造体は下記のとおりです。

（1）TPRJ_LIST - Process Job List Information - S16F15

```
typedef struct{
        int         prj_count;          // ﾌﾟﾛｾｽｼﾞｮﾌﾞ情報数
        TPRJ_INFO   **prj_list;         // ﾌﾟﾛｾｽｼﾞｮﾌﾎﾟｲﾝﾀﾘｽﾄ
        int         err_count;          // ｴﾗｰ応答情報数
        TERR_INFO   **err_list;         // ｴﾗｰ情報ﾘｽﾄﾎﾟｲﾝﾀ(S16F16)
        } TPRJ_LIST;
```

（2）TPRJ_INFO - Process Job Information

```
typedef struct{
        int         index;              // Prj 情報 index
        int         state;
        char        *prjobid;           // process job id
        int         mf;
        int         cj_index;           // CJ の管理 index
        int         car_count;          // mf=13 のとき ｷｬﾘｱ数
        TCAR_INFO   **car_list;         // ｷｬﾘｱ情報ﾎﾟｲﾝﾀﾘｽﾄ
        int         mid_count;          // mf=14 のとき mid 数
        char        **mid_list;
        int         prrecipemethod;     // fmt=51(8) U1
        TRCP_INFO   *rcp_info;          // ﾚｼﾋﾟ情報
        int         prprocessstart;     // fmt 11(8) Bool 1=auto,0=man
        TCEID       pause_ceid;         // ｲﾍﾞﾝﾄ ID
        } TPRJ_INFO;
```

state の値

```
#define ST_PRJ_Pooled               0
#define ST_PRJ_Settingup            1
#define ST_PRJ_WaitingForHos        2
#define ST_PRJ_Processing           3
#define ST_PRJ_ProcessComplete      4
#define ST_PRJ_Reserved             5
#define ST_PRJ_Pausing              6
#define ST_PRJ_Paused               7
#define ST_PRJ_Stopping             8
#define ST_PRJ_Aborting             9
```

(3) TPRJ_ERR_INFO - S16F16 Response Information

```
typedef struct{
        int         prj_count;          // process job id count
        char        **prj_list;         // process job id list
        int         acka;               // Boolean
        int         err_count;          // number of errors
        TERR_INFO   **err_list;         // error information list
} TPRJ_ERR_INFO;
```

(4) TPRJ_CMD_INFO - Process Job Command Information - S16F5

```
typedef struct{
        char        *prjobid;
        char        *cmd;
        int         cmd_index;
        int         cp_count;
        TCMD_PARA   **cp_list;
        } TPRJ_CMD_INFO;
```

cmd_index

```
        #define CM_ABORT                    0
        #define CM_STOP                     1
        #define CM_CANCEL                   2
        #define CM_PAUSE                    3
        #define CM_RESUME                   4
```

3．20．２　PRJ プロセスジョブ情報アクセスとメッセージ送信関数

3．20．２．１　EngAllocPrjInfo() - プロセスジョブの登録

（1）呼出書式

[c,C++]

```
API int APIX EngAllocPrjInfo(
        char *prjid,                // ﾌﾟﾛｾｽｼﾞｮﾌﾞ ID 格納領域のﾎﾟｲﾝﾀ
        int  *index                 // 得られた情報領域ｲﾝﾃﾞｸｽ格納用ﾎﾟｲﾝﾀ
);
```

[.NET VB]

```
Function EngAllocPrjInfo (
          ByVal prjid As String,
          ByRef index As Int32) As Int32
```

[.NET C#]

```
int EngAllocPrjInfo(
          byte[] prjid,
          ref int index );
```

（2）引数

prjid

登録したいプロセスジョブ ID が格納されているポインタです。

index

DSHEng4 内に登録された PRJ 情報領域のインデクス値が格納される領域のポインタです。

（3）戻り値

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

（4）説明

プロセスジョブを新規にシステムに登録するための関数です。
登録は、引数 prjid で与えられるプロセスジョブ ID をシステムに登録します。

正常に登録できた場合は、index で指定される領域に登録された情報領域のインデクスが設定返却されます。
もし、prjid に指定されたプロセスジョブが既に登録済みであった場合には関数の戻り値 1 を返却します。
index には既に登録されている情報領域のインデクスが設定されます。

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

３．20．２．２　EngSetPrjInfo()　- プロセスジョブ情報の設定
.　　EngSetPrjInfoX() - インデクスでのプロセスジョブ情報の設定

（1）呼出書式

[c,C++]

```
API int APIX EngSetPrjInfo(
        TPRJ_INFO *pinfo                // プロセスジョブ情報格納構造体のﾎﾟｲﾝﾀ
);

API int APIX EngSetPrjInfoX(
        int         index,              // EngAllocPrjInfo()で得られたｲﾝﾃﾞｸｽ値
        TPRJ_INFO *pinfo                // プロセスジョブ情報格納構造体のﾎﾟｲﾝﾀ
);
```

[.NET VB]

```
Function EngSetPrjInfo (
        ByRef pinfo As dsh_info.TPRJ_INFO) As Int32

Function EngSetPrjInfoX (
        ByVal index As Int32,
        ByRef pinfo As dsh_info.TPRJ_INFO) As Int32
```

[.NET C#]

```
int EngSetPrjInfo(
       ref TPRJ_INFO pinfo );

int EngSetPrjInfoX(
        int index,
        ref TPRJ_INFO pinfo );
```

（2）引数

pinfo

設定したいプロセスジョブ情報が格納されている格納構造体領域のポインタです。

index

プロセスジョブ情報のインデクスです。登録時に EngAllocPrjInfo()関数によって与えられます。
インデクスは、PRJID から EngGetPrjIdIndex()関数で取得することができます。

（3）戻り値

戻り値	意味
0	正常に設定できた。
(-1)	設定できなかった。

（4）説明

本関数は、pinfo に格納されているプロセスジョブ情報の設定・変更に使用します。

引数 pinfo 内のメンバーprjid に指定されるプロセスジョブ ID の情報として設定されます。

pinfo 内には PRJID の他、キャリア、レシピ情報などの情報が含まれます。

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

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

３．20．２．３　EngGetPrjInfo()　- プロセスジョブ情報の取得
.　　EngGetPrjInfoX() - インデクス指定でのプロセスジョブ情報の取得

（1）呼出書式

[c,C++]

```
API int APIX EngGetPrjInfo(
        char        *prjid;              // PRJID が格納されている領域のポインタ
        TPRJ_INFO *pinfo                 // ﾌﾟﾛｾｽｼﾞｮﾌﾞ情報を格納する構造体のポインタ
);

API int APIX EngGetPrjInfoX(
        int         index;               // ﾌﾟﾛｾｽｼﾞｮﾌﾞ情報のｲﾝﾃﾞｸｽ
        TPRJ_INFO *pinfo                 // ﾌﾟﾛｾｽｼﾞｮﾌﾞ情報を格納する構造体のポインタ
);
```

[.NET VB]

```
Function EngGetPrjInfo (
        ByVal prjid As String,
        ByRef pinfo As dsh_info.TPRJ_INFO) As Int32

Function EngGetPrjInfoX (
        ByVal index As Int32,
        ByRef pinfo As dsh_info.TPRJ_INFO) As Int32
```

[.NET C#]

```
int EngGetPrjInfo(
        byte[] prjid,
        ref TPRJ_INFO pinfo );

int EngGetPrjInfoX(
        int index,
        ref TPRJ_INFO pinfo );
```

（2）引数

prjid

プロセスジョブ ID が格納されている領域のポインタです。

pinfo

取得したプロセスジョブ情報を格納する構造体領域のポインタです。

index

プロセスジョブ ID 情報のインデクスです。登録時に EngAllocPrjInfo()関数によって与えられます。
インデクスは、PRJID から EngGetPrjIdIndex()関数で取得することができます。

（3）戻り値

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

（4）説明

prjid または index に指定されているプロセスジョブの情報を pinfo に構造体に取得格納します。
TPRJ_INFO 構造体の中に情報を格納するために必要なメモリは、DSHEng4 が準備確保します。即ち、構造体のメンバーの中でポインタになっている情報の実体即ち、プロセスジョブ，キャリア情報などのためのメモリは DSHEng4 が準備します。
これらのメモリは、使用後、ユーザが DSHEng4 の API 関数を使って次のように開放してください。

```
TPRJ_INFO   *pinfo;

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

３．20．２．４　EngDelPrjInfo()　- プロセスジョブの削除
.　　EngDelPrjInfoX() - インデクスでのプロセスジョブの削除

（1）呼出書式

[c,C++]

```
API int APIX EngDelPrjInfo(
      char      *prjid;                  // PRJID が格納されている領域のﾎﾟｲﾝﾀ
);

API int APIX EngDelPrjInfo(
      int       index;                   // ｲﾝﾃﾞｸｽ(0,1,2....)
);
```

[.NET VB]

```
Function EngDelPrjInfo (
       ByVal prjid As String) As Int32

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

[.NET C#]

```
int EngDelPrjInfo(
       byte[] prjid );

int EngDelPrjInfoX(
       int index );
```

（2）引数

prjid

プロセスジョブ ID が格納されている領域のポインタです。

index

削除したいプロセスジョブ情報のインデクスです。

（3）戻り値

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

（4）説明

prjid または index に指定されているプロセスジョブ ID の情報をシステムの登録から削除します。

３．20．２．５　EngSetPrjState() － プロセスジョブ状態の設定
.　　EngSetPrjStateX() － インデクスでのプロセスジョブ状態の設定

（1）呼出書式

[c,C++]

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

API int APIX EngSetPrjStateX(
        int         index;                  // ﾌﾟﾛｾｽｼﾞｮﾌﾞ情報のインデクス
        int         state                   // 設定したい状態値
);
```

[.NET VB]

```
Function EngSetPrjState (
        ByVal prjid As String,
        ByVal state As Int32) As Int32

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

[.NET C#]

```
int EngSetPrjState(
        byte[] prjid,
        int state );

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

（2）引数

prjid

プロセスジョブ ID が格納されている領域のポインタです。

state

設定したい PRJ の状態値です。

index

プロセスジョブ情報のインデクスです。登録時に EngAllocPrjInfo()関数によって与えられます。
インデクスは、PRJID から EngGetPrjIdIndex()関数で取得することができます。

（3）戻り値

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

（4）説明

プロセスジョブ情報の状態値を設定します。
デフォルトの状態値は下表のとおりです。

プロセスジョブ状態のデフォルト値

状態値記号	値
ST_PRJ_Pooled	0
ST_PRJ_Settingup	1
ST_Prj_WaitingForHost	2
ST_PRJ_Processing	3
ST_PRJ_ProcessComplete	4
ST_PRJ_Pausing	6
ST_PRJ_Paused	7
ST_PRJ_Stopping	8
ST_PRJ_Aborting	9

３．20．２．６　EngGetPrjState() － プロセスジョブ状態の取得
.　　EngGetPrjStateX() － インデクスでのプロセスジョブ状態の取得

（1）呼出書式

[c,C++]

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

API int APIX EngGetPrjStateX(
        int       index,                  // ﾌﾟﾛｾｽｼﾞｮﾌﾞのインデクス
        int       *state                  // 取得した状態値格納用領域ポインタ
);
```

[.NET VB]

```
Function EngGetPrjState (
        ByVal prjid As String,
        ByRef state As Int32) As Int32

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

[.NET C#]

```
int EngGetPrjState(
        byte[] prjid,
        ref int state );

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

（2）引数

prjid

プロセスジョブ ID が格納されている領域のポインタです。

state

取得したプロセスジョブの状態値を格納する領域のポインタです。

index

プロセスジョブ ID 情報のインデクスです。登録時に EngAllocPrjInfo()関数によって与えられます。
インデクスは、PRJID から EngGetPrjIdIndex()関数で取得することができます。

（3）戻り値

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

（4）説明

プロセスジョブ情報の状態値を取得します。
デフォルトの状態値は下表のとおりです。

プロセスジョブ状態のデフォルト値

状態値記号	値
ST_PRJ_Pooled	0
ST_PRJ_Settingup	1
ST_Prj_WaitingForHost	2
ST_PRJ_Processing	3
ST_PRJ_ProcessComplete	4
ST_PRJ_Pausing	6
ST_PRJ_Paused	7
ST_PRJ_Stopping	8
ST_PRJ_Aborting	9

3．20．２．７　EngGetPrjList() – 全登録プロセスジョブ ID 取得関数

（1）呼出書式

[c,C++]

```
API int APIX EngGetPrjList(
        TTEXT_DLIST **list                  // 取得ﾘｽﾄ格納ﾎﾟｲﾝﾀの格納ﾎﾟｲﾝﾀ
);
```

[.NET VB]

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

[.NET C#]

```
int EngGetPrjList(
          IntPtr list );
```

（2）引数

list

取得できた PRJID を格納する TTEXT_DLIST 構造体のポインタを格納する領域のポインタです。

（3）戻り値

戻り値	意味
0	正常に取得できた。
(-1)	取得できなかった。

（4）説明

システムに登録されている PRJID(プロセスジョブ ID)を TTEXT_DLIST 構造体に取出すための関数です。
info->name_list には NULL が設定されます。(定義名がありません。)
取得した情報の処理が終了した後、DshFreeTText_DLIST()関数で list 内部の情報格納用に使用されているメモリを開放してください。

TTEXT_DLIST 構造体は次のとおりです。

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

3．20．２．８　EngGetPrjId() - インデクスから PRJID の取得

（1）呼出書式

[c,C++]

```
API int  APIX EgnGetPrjId(
        int     index,                  // プロセスジョブ情報のインデクス
        char    *prjid                  // 取得した PRJID を格納する領域のポインタ
);
```

[.NET VB]

```
Function EngGetPrjId (
        ByVal index As Int32,
        ByVal prjid As String) As Int32
```

[.NET C#]

```
int EngGetPrjId(
        int index,
        byte[] prjid );
```

（2）引数

index

プロセスジョブ情報のインデクスです。登録時に EngAllocPrjInfo()関数によって与えられます。
インデクスは、PRJID から EngGetPrjIdIndex()関数で取得することができます。

prjid

PRJID を格納するための領域のポインタです。

（3）戻り値

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

（4）説明

プロセスジョブ情報のインデクスから PRJID を取得し、prjid に格納します。
正常に取得できた場合は関数戻り値として 0 が返却されます。

3．20．２．９　EngGetPrjIdIndex() -PRJIDからインデクスの取得

（1）呼出書式

[c,C++]

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

[.NET VB]

```
Function EngGetPrjIdIndex (
          ByVal prjid As String,
          ByRef index As Int32) As Int32
```

[.NET C#]

```
int EngGetPrjIdIndex(
          byte[] prjid,
          ref int index );
```

（2）引数

prjid

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

index

　　取得したインデクスの値を格納するための領域のポインタです。

（3）戻り値

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

（4）説明

prjidに指定されるPRJIDからプロセスジョブ情報インデクスを取得するための関数です。
取得されたインデクスはindexで指定された領域に格納されます。
正常に取得できた場合は関数戻り値として0が返却されます。

3．20．3　PRJ プロセスジョブ関連ライブラリ関数

3．20．3．1　DshDecodeS16F5() - S16F5 をプロセスジョブコマンド情報にデコード

（1）呼出書式

[c,C++]

```
API int  APIX DshDecodeS16F5(
        DSHMSG *smsg,                   // SECS ﾒｯｾｰｼﾞ情報構造体のﾎﾟｲﾝﾀ
        TPRJ_CMD_INFO *info             // ﾃﾞｺｰﾄﾞしたｺﾏﾝﾄﾞ情報を格納する構造体のﾎﾟｲﾝﾀ
);
```

[.NET VB]

```
Function DshDecodeS16F5 (
        ByRef smsg As dshdr2.DSHMSG,
        ByRef info As dsh_info.TPRJ_CMD_INFO) As Int32
```

[.NET C#]

```
int DshDecodeS16F5(
        ref DSHMSG smsg,
        ref TPRJ_CMD_INFO info );
```

（2）引数

smsg

S16F5 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。

info

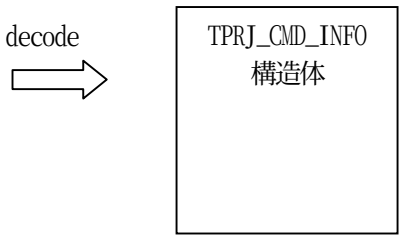
デコードしたプロセスジョブコマンド情報を格納するための構造体のポインタです。

（3）戻り値

戻り値	意味
0	正常にﾃﾞｺｰﾄﾞできた。
(-1)	smsg を正しくﾃﾞｺｰﾄﾞできなかった。

（4）説明

S16F5 はプロセスジョブに対するコマンド情報を含むメッセージです。
S16F5 メッセージに含まれるプロセスジョブコマンド情報を、ユーザプログラムが処理しやすい TPRJ_CMD_INFO 構造体の中にデコードします。
なお、構造体使用後は、構造体内部で使用されたメモリを DshFreeTPRJ_CMD_INFO()関数を使って開放してください。



３．20．３．２　DshFreeTPRJ_CMD_INFO() - プロセスジョブコマンド情報構造体メモリの開放

（1）呼出書式

[c,C++]

```
API void APIX DshFreeTPRJ_CMD_INFO(
      TPRJ_CMD_INFO *info              // ﾒﾓﾘを開放したい情報が格納されている構造体のﾎﾟｲﾝﾀ
);
```

[.NET VB]

```
Sub DshFreeTPRJ_CMD_INFO (
        ByRef info As dsh_info.TPRJ_CMD_INFO)
```

[.NET C#]

```
void DshFreeTPRJ_CMD_INFO(
        ref TPRJ_CMD_INFO info );
```

（2）引数

info

メモリを解放したいプロセスジョブコマンド情報構造体のポインタです。

（3）戻り値

なし。

（4）説明

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

３．20．３．３　DshCopyTPRJ_CMD_INFO() - プロセスジョブコマンド情報構造体のコピー

（1）呼出書式

[c,C++]

```
API int APIX   DshCopyTPRJ_CMD_INFO(
        TPRJ_CMD_INFO *dlist,                  // ｺﾋﾟｰ先のﾎﾟｲﾝﾀ
        TPRJ_CMD_INFO *slist                   // ｺﾋﾟｰ元のﾎﾟｲﾝﾀ
);
```

[.NET VB]

```
Function DshCopyTPRJ_CMD_INFO (
          ByRef dinfo As dsh_info.TPRJ_CMD_INFO,
          ByRef sinfo As dsh_info.TPRJ_CMD_INFO) As Int32
```

[.NET C#]

```
int DshCopyTPRJ_CMD_INFO(
          ref TPRJ_CMD_INFO dinfo,
          ref TPRJ_CMD_INFO sinfo );
```

（2）引数

dlist

プロセスジョブコマンド情報構造体のコピー先ポインタです。

slist

コピー元のプロセスジョブコマンド情報構造体が格納されている構造体メモリのポインタです。

（3）戻り値

戻り値	意味
0	正常にｺﾋﾟｰできた。
(-1)	sinfo または dinfo の値が NULL だったのでｺﾋﾟｰできなかった。

（4）説明

slist が指す TPRJ_CMD_INFO 構造体内に格納されているプロセスジョブコマンド情報を dinfo が指定する TPRJ_CMD_INFO 構造体にコピーします。

dinfo 内のメンバーで新しいメモリが必要なものは本関数が取得します。

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

３．20．３．４　DshInitTPRJ_CMD_ERR_INFO () - プロセスジョブ応答情報の初期化

（1）呼出書式

[c,C++]

```
API void APIX   DshInitTPRJ_CMD_ERR_INFO(
        TPRJ_CMD_ERR_INFO *errinfo,          // ｴﾗｰ情報格納構造体のﾎﾟｲﾝﾀ
        char        *prjobid,                // ﾌﾟﾛｾｽｼﾞｮﾌﾞ ID 格納ﾎﾟｲﾝﾀ
        int         acka,                    // ack ﾃﾞｰﾀ
        int         err_count                // ｴﾗｰ情報のﾘｽﾄｻｲｽﾞ （個数  0,1,2...）
);
```

[.NET VB]

```
Sub DshInitTPRJ_CMD_ERR_INFO (
        ByRef errinfo As dsh_info.TPRJ_CMD_ERR_INFO,
        ByVal prjobid As String,
        ByVal acka As Int32,
        ByVal errcount As Int32)
```

[.NET C#]

```
void DshInitTPRJ_CMD_ERR_INFO(
        ref TPRJ_CMD_ERR_INFO errinfo,
        byte[] prjobid,
        int acka,
        int errcount );
```

（2）引数

errinfo

TPRJ_CMD_ERR_INFO 応答情報構造体のポインタです。

prjobid

プロセスジョブ ID が格納されている領域のポインタです。

acka

acka - ACK の値です。

err_count

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

（3）戻り値

なし。

（4）説明

本関数は、S16F5 に対する応答メッセージのための応答情報を TPRJ_CMD_ERR_INFO 構造体に初期設定するために使用します。

errinfo で指定された構造体の acka メンバーに引数 acka の値を設定し、prjobid メンバーに prjobid を、そして err_count メンバーに引数 err_count を設定します。

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

err_info へのエラー情報の設定には DshPutTERR_INFO_LIST ()関数を使用します。

３．20．３．５　DshFreeTPRJ_CMD_ERR_INFO() - プロセスジョブコマンドエラー構造体メモリの開放

（1）呼出書式

[c,C++]

```
API void APIX DshFreeTPRJ_CMD_ERR_INFO(
      TPRJ_CMD_ERR_INFO *info              // ﾒﾓﾘを開放したい情報が格納されている構造体のﾎﾟｲﾝﾀ
);
```

[.NET VB]

```
Sub DshFreeTPRJ_CMD_ERR_INFO (
        ByRef info As dsh_info.TPRJ_CMD_ERR_INFO)
```

[.NET C#]

```
void DshFreeTPRJ_CMD_ERR_INFO(
        ref TPRJ_CMD_ERR_INFO info );
```

（2）引数

info

メモリを解放したいプロセスジョブコマンドエラー情報構造体のポインタです。

（3）戻り値

なし。

（4）説明

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

3．20．3．6　DshMakeS16F5Response() - S16F5 の応答メッセージの生成

（1）呼出書式

[c,C++]

```
API int APIX DshMakeS16F5Response(
        TPRJ_CMD_INFO *info,            // ﾌﾟﾛｾｽｼﾞｮﾌﾞｺﾏﾝﾄﾞ情報格納領域ﾎﾟｲﾝﾀ
        TPRJ_CMD_ERR_INFO *erinfo,      // S16F6 に設定する応答情報格納領域のﾎﾟｲﾝﾀ
        DSHMSG *smsg,                   // S16F6 ﾒｯｾｰｼﾞを格納するﾒｯｾｰｼﾞ構造体のﾎﾟｲﾝﾀ
        BYTE   *buff,                   // S16F6 のﾃｷｽﾄ格納ﾊﾞｯﾌｧﾎﾟｲﾝﾀ
        int    buff_size                // buff のﾊﾞｲﾄｻｲｽﾞ
);
```

[.NET VB]

```
Function DshMakeS16F5Response (
        ByRef info As dsh_info.TPRJ_CMD_INFO,
        ByRef erinfo As dsh_info.TPRJ_CMD_ERR_INFO,
        ByRef smsg As dshdr2.DSHMSG,
        ByRef buff As Byte,
        ByVal buff_size As Int32) As Int32
```

[.NET C#]

```
int DshMakeS16F5Response(
        ref TPRJ_CMD_INFO info,
        ref TPRJ_CMD_ERR_INFO erinfo,
        ref DSHMSG smsg,
        byte[] buff,
        int buff_size );
```

（2）引数

info

プロセスジョブコマンド情報が格納されている領域のポインタです。

erinfo

S16F6 メッセージに設定する応答情報が格納されている領域のポインタです。

msg

S16F6 応答メッセージ情報を格納するためのメッセージ構造体のポインタです。

buff

S16F6 応答メッセージのテキストを格納するためのバッファポインタです。

buff_size

buff のバイトサイズです。

（3）戻り値

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

（4）説明

S16F5 に対する S16F6 応答メッセージを info に含まれるプロセスジョブ生成情報と応答情報に従って作成します。

応答情報内の、acka を S16F6 の ACKA として設定します。
acka はユーザが S16F5 プロセス生成メッセージを評価した結果です。

3．20．3．7　DshDecodeS16F11() - S16F11 をプロセスジョブ情報にデコード

（1）呼出書式

[c,C++]

```
API int  APIX DshDecodeS16F11(
        DSHMSG *smsg,                    // SECS ﾒｯｾｰｼﾞ情報構造体のﾎﾟｲﾝﾀ
        TPRJ_INFO *info                  // ﾃﾞｺｰﾄﾞした情報を格納する構造体のﾎﾟｲﾝﾀ
);
```

[.NET VB]

```
Function DshDecodeS16F11 (
        ByRef smsg As dshdr2.DSHMSG,
        ByRef info As dsh_info.TPRJ_INFO) As Int32
```

[.NET C#]

```
int DshDecodeS16F11(
        ref DSHMSG smsg,
        ref TPRJ_INFO info );
```

（2）引数

smsg

S16F11 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。

info

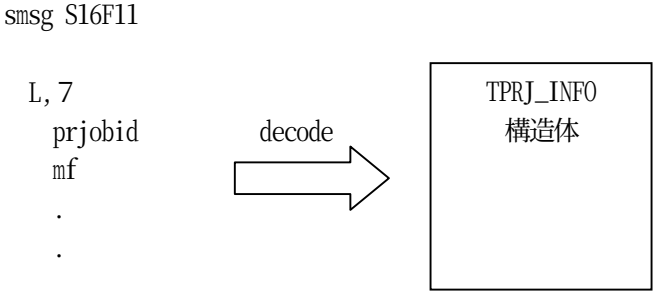
デコードしたプロセスジョブ情報を格納するための構造体のポインタです。

（3）戻り値

戻り値	意味
0	正常にﾃﾞｺｰﾄﾞできた。
(-1)	smsg を正しくﾃﾞｺｰﾄﾞできなかった。

（4）説明

S16F11 は 1 個のプロセスジョブの生成情報を含むメッセージです。
S16F11 メッセージに含まれるプロセスジョブ情報を、ユーザプログラムが処理しやすい TPRJ_INFO 構造体の中にデコードします。
なお、構造体使用後は、構造体内部で使用されたメモリを DshFreeTPRJ_INFO()関数を使って開放してください。



３．20．３．８　DshFreeTPRJ_INFO() - プロセスジョブ情報構造体メモリの開放

（1）呼出書式

[c,C++]

```
API void APIX DshFreeTPRJ_INFO(
      TPRJ_INFO *pinfo              // ﾒﾓﾘを開放したい情報が格納されている構造体のﾎﾟｲﾝﾀ
);
```

[.NET VB]

```
Sub DshFreeTPRJ_INFO (
        ByRef info As dsh_info.TPRJ_INFO )
```

[.NET C#]

```
void DshFreeTPRJ_INFO(
        ref TPRJ_INFO info );
```

（2）引数

pinfo

メモリを解放したいプロセスジョブ情報構造体のポインタです。

（3）戻り値

なし。

（4）説明

TPRJ_INFO 構造体内で情報格納用に使用されているメモリを全て解放します。
開放した後、TPRJ_INFO の内容を 全て 0 で初期設定します。
pinfo が NULL ならば、何も処理しません。

３．20．３．９　DshCopyTPRJ_INFO() - プロセスジョブ情報構造体のコピー

（1）呼出書式

[c,C++]

```
API int APIX   DshCopyTPRJ_INFO(
        TPRJ_INFO *dinfo,                  // ｺﾋﾟｰ先のﾎﾟｲﾝﾀ
        TPRJ_INFO *sinfo                   // ｺﾋﾟｰ元のﾎﾟｲﾝﾀ
);
```

[.NET VB]

```
Function DshCopyTPRJ_INFO (
          ByRef info As dsh_info.TPRJ_INFO,
          ByRef sinfo As dsh_info.TPRJ_INFO) As Int32
```

[.NET C#]

```
int DshCopyTPRJ_INFO(
          ref TPRJ_INFO info,
          ref TPRJ_INFO sinfo );
```

（2）引数

dlist

プロセスジョブ情報のコピー先ポインタです。

slist

コピー元のプロセスジョブ情報が格納されている構造体メモリのポインタです。

（3）戻り値

戻り値	意味
0	正常にｺﾋﾟｰできた。
(-1)	sinfo または dinfo の値が NULL だったのでｺﾋﾟｰできなかった。

（4）説明

sinfo が指す TPRJ_INFO 構造体内に格納されているプロセスジョブ情報を dinfo が指定する TPRJ_INFO 構造体にコピーします。

dinfo 内のメンバーで新しいメモリが必要なものは本関数が取得します。

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

3．20．3．10 DshInitPrjInfo - プロセスジョブTPRJ_INFOの初期設定

（1）呼出書式

[c,C++]

```
API int  APIX DshInitPrjInfo(
        TPRJ_INFO    *info,              // TPRJ_INFO構造体のﾎﾟｲﾝﾀ
        char         *prjobid,           // ﾌﾟﾛｾｽｼﾞｮﾌﾞ ID
        int          mf,                 // material format code
        int          cj_index,           // Control Job情報index
        int          m_count,            // ｷｬﾘｱ数または基板数
        int          prrecipemethod,     // ﾚｼﾋﾟ方式
        int          prprocessstart,     // 準備完了時ﾌﾟﾛｾｽ開始ﾌﾗｸﾞ
        int          ceid_count          // ﾌﾟﾛｾｽｼﾞｮﾌﾞへ送信できる収集ｲﾍﾞﾝﾄ数
);
```

[.NET VB]

```
Sub DshInitPrjInfo (
        ByRef info As dsh_info.TPRJ_INFO,
        ByVal prjobid As String,
        ByVal mf As Int32,
        ByVal cj_index As Int32,
        ByVal mid_count As Int32,
        ByVal prrecipemethod As Int32,
        ByVal prprocessstart As Int32,
        ByVal ceid_count As Int32)
```

[.NET C#]

```
void DshInitPrjInfo(
        ref TPRJ_INFO info,
        byte[] prjobid,
        int mf,
        int cj_index,
        int mid_count,
        int prrecipemethod,
        int prprocessstart,
        int ceid_count );
```

（2）引数

info

プロセスジョブTPRJ_INFO構造体のポインタです。このメンバーを初期設定します。

prjobid

設定するプロセスジョブIDです。

mf

material(材料)フォーマットコードで、13=ｷｬﾘｱ,14=基板単位,その他は指定なしの意味になります。

cj_index

LINKするCJ(Control Job)情報管理ﾃｰﾌﾞﾙのindex値です。((-1)で未定)

mid_count

処理対象のマテリアル数です。mf=13は、キャリア数、mf=14は基板数になります。mfがそれ以外の

の値の場合には、=0 を指定してください。

prrecipemethod

レシピの適用方法(1=ﾚｼﾋﾟのみ、2=可変ﾁｭｰﾆﾝｸﾞ付きﾚｼﾋﾟ)です。

prprocessstart

準備完了時のプロセス開始フラグ（TRUE=自動開始, FALSE=手動開始 ）

ceid_count

プロセスジョブへ送信できる収集イベントの数（プロセスジョブ停止用、CEID は別関数で設定）

（3）戻り値

なし。

（4）説明

本関数は APP が OFFLINE でプロセスジョブ情報を生成する際に使用することができます。
最初に info 内をクリアします。そして、引数で指定された情報を info 内に設定します。
メモリが必要なメンバーについてはメモリを確保し情報をコピーします。
レシピ情報の設定には DshPutPrjRcpInfo()、CEID の設定には DshPutPrjPauseCeid()関数を使用してください。

また、mf の値によって、次の関数を使用してキャリアまたはマテリアル ID を設定してください。

mf = 13 ： DshPutPrjCarInfo()関数
mf = 14 ： DshPutPrjMid()関数
その他の mf の値の場合は、なにも設定しないでください。

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

３．20．３．11　DshPutPrjRcpInfo() -レシピ情報の追加

（1）呼出書式

[c,C++]

```
API void APIX DshPutPrjRcpInfo(
        TPRJ_INFO    *info,                 // ﾌﾟﾛｾｽｼﾞｮﾌﾞ情報構造体のﾎﾟｲﾝﾀ
        TRCP_INFO    *rinfo                 // ﾚｼﾋﾟ情報が格納されている構造体ﾎﾟｲﾝﾀ
);
```

[.NET VB]

```
Sub DshPutPrjRcpInfo (
          ByRef info As dsh_info.TPRJ_INFO,
          ByRef rcp_info As dsh_info.TRCP_INFO)
```

[.NET C#]

```
void DshPutPrjRcpInfo(
          ref TPRJ_INFO info,
          ref TRCP_INFO rcp_info );
```

（2）引数

info

プロセスジョブ情報構造体のポインタです。

rinfo

加えたいレシピ情報が格納されている構造体のポインタです。

（3）戻り値

なし。

（4）説明

先にDshInitPrjInfo()で初期設定された構造体メンバー rcp_info にレシピ情報rinfo の内容を加えます。実際には、rinfo の内容を DshCopyTRCP_INFO()関数を使って別メモリにコピーした上で rcp_info メンバーに設定します。

3．20．3．12 DshPutPrjCarInfo() - キャリア情報の追加

（1）呼出書式

[c,C++]

```
API int APIX DshPutPrjCarInfo(
        TPRJ_INFO    *info,                 // ﾌﾟﾛｾｽｼﾞｮﾌﾞ情報構造体のﾎﾟｲﾝﾀ
        TCAR_INFO    *cinfo                 // ｷｬﾘｱ情報が格納されている構造体ﾎﾟｲﾝﾀ
);
```

[.NET VB]

```
Function DshPutPrjCarInfo (
          ByRef info As dsh_info.TPRJ_INFO,
          ByRef cinfo As dsh_info.TCAR_INFO) As Int32
```

[.NET C#]

```
int DshPutPrjCarInfo(
          ref TPRJ_INFO info,
          ref TCAR_INFO cinfo );
```

（2）引数

info

プロセスジョブ情報構造体のポインタです。

cinfo

加えたいキャリア情報が格納されている構造体のポインタです。

（3）戻り値

戻り値	意味
0	正常に追加できた。
(-1)	ｷｬﾘｱ情報の数が既に指定数を超えている。

（4）説明

先にDshInitPrjInfo()で初期設定されたキャリア情報リストcar_listにキャリア情報TCAR_INFOを１個加えます。実際には、ｃinfoの内容をDshCopyTCAR_INFO()関数を使って別メモリにコピーした上でcar_listメンバーに設定します。

本関数が実行される順にキャリア情報リストcar_list上に、情報を追加していきます。
設定後 0 を返却します。
もし、info内のcar_countで指定された分の情報が既に設定済みであった場合は、(-1)を返却します。

本関数は、DshInitPrjInfo()関数の引数 mf の値を =13 で設定した場合に使用します。
mfは、マテリアルフォーマットを意味し、 =13がキャリア単位の指定になります。

TCAR_INFOの設定とそれに、スロットIDを設定するためには、以下の関数を使用できます。

DshInitCarInfo()　　：TCAR_INFOの初期設定
DshPutCarSlotInfo()：TCAR_INFOにスロットTSLOT_INFOを設定する。

TSLOT_INFOの設定には以下の関数を使用します。

DshInitCarSlotInfo()：TSLOT_INFOの初期設定
DshPutCarSlotInfo()：TSLOT_INFOにSlotIDを設定する。

3．20. 3．13 DshPutPrjMid() - マテリアル ID の追加

（1）呼出書式

[c,C++]

```
API int APIX DshPutPrjMid(
        TPRJ_INFO    *info,                 // ﾌﾟﾛｾｽｼﾞｮﾌﾞ情報構造体のﾎﾟｲﾝﾀ
        char         *mid                   // ﾏﾃﾘｱﾙ ID が格納されているﾎﾟｲﾝﾀ
);
```

[.NET VB]

```
Function DshPutPrjMid (
        ByRef info As dsh_info.TPRJ_INFO,
        ByVal mid As string) As Int32
```

[.NET C#]

```
int DshPutPrjMid(
        ref TPRJ_INFO info,
        byte[] mid );
```

（2）引数

info

プロセスジョブ情報構造体のポインタです。

mid

加えたいマテリアル ID が格納されている領域のポインタです。

（3）戻り値

戻り値	意味
0	正常に追加できた。
(-1)	ﾏﾃﾘｱﾙ ID の数が既に指定数を超えている。

（4）説明

先に DshInitPrjInfo()で初期設定されたマテリアル情報リスト mid_list にマテリアル ID を１個加えます。

本関数が実行される順にマテリアル ID リスト mid_list 上に、情報を追加していきます。
設定後 0 を返却します。
もし、info 内の mid_count で指定された分の MID 情報が既に設定済みであった場合は、(-1)を返却します。

本関数は、DshInitPrjInfo()関数の引数 mf の値を =14 で設定した場合に使用します。
mf は、マテリアルフォーマットを意味し、 =14 は基板（マテリアル）単位の指定になります。

3．20．3．14 DshPutPrjPauseCeid() - プロセスジョブ停止用イベント ID の追加

（1）呼出書式

[c,C++]

```
API int APIX DshPutPrjPauseCeid(
      TPRJ_INFO   *info,              // ﾌﾟﾛｾｽｼﾞｮﾌﾞ情報構造体のﾎﾟｲﾝﾀ
      int         ceid                // 加えたいｲﾍﾞﾝﾄ ID
);
```

[.NET VB]

```
Function DshPutPrjPauseCeid (
        ByRef info As dsh_info.TPRJ_INFO,
        ByVal ceid As Int32) As Int32
```

[.NET C#]

```
int DshPutPrjPauseCeid(
        ref TPRJ_INFO info,
        int ceid );
```

（2）引数

info

プロセスジョブ情報構造体のポインタです。

ceid

加えたいイベント ID です。

（3）戻り値

戻り値	意味
0	正常に追加できた。
(-1)	CEID の数が既に指定数を超えている。

（4）説明

先に DshInitPrjInfo()で初期設定されたイベント情報リスト pause_ceid_list に CEID を１個加えます。
本関数が実行される順に pause_ceid_list 内のイベント情報リスト上に、情報を追加していきます。
設定後 0 を返却します。
もし、info 内の ceid_count で指定された分の情報が既に設定済みであった場合は、(-1)を返却します。

(注) 本関数は ceid の値 (-1) は使用されないことを前提に処理します。

３．20．３．15　DshInitTPRJ_ERR_INFO () - プロセスジョブ生成応答情報の初期化

（1）呼出書式

[c,C++]

```
API void APIX   DshInitTPRJ_ERR_INFO(
        TPRJ_ERR_INFO *errinfo,          // ｴﾗｰ情報格納構造体ﾘｽﾄのﾎﾟｲﾝﾀ
        int         prj_count,           // ﾌﾟﾛｾｽｼﾞｮﾌﾞ ID 数
        int         acka,                // ack ﾃﾞｰﾀ
        int         err_count            // ｴﾗｰ情報のﾘｽﾄｻｲｽﾞ （個数  0,1,2...）
);
```

[.NET VB]

```
Sub DshInitTPRJ_ERR_INFO (
        ByRef errinfo As dsh_info.TPRJ_ERR_INFO,
        ByVal prj_count As Int32,
        ByVal acka As Int32,
        ByVal errcount As Int32)
```

[.NET C#]

```
void DshInitTPRJ_ERR_INFO(
        ref TPRJ_ERR_INFO errinfo,
        int prj_count,
        int acka,
        int errcount );
```

（2）引数

errinfo

TPRJ_ERR_INFO 応答情報構造体のポインタです。

prj_count

プロセスジョブ ID 数

S16F11、S16F17 の場合は = 1 にします。

acka

acka - ACK の値です。

err_count

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

（3）戻り値

なし。

（4）説明

本関数は、S16F11, 15, 17 に対する応答メッセージのための応答情報を TPRJ_ERR_INFO 構造体に初期設定するために使用します。

errinfo で指定された構造体の acka メンバーに引数 acka の値を設定し、prj_count メンバーに引数 prj_count を、err_count メンバーに引数 err_count の値を設定します。

また､errinfo 内の prj_list に prj_count 分のプロセスジョブ ID を格納するためのリスト領域を設けます。

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

err_info へのプロセスジョブ ID の設定には、DshPutTPRJ_ERR_PRJID()関数を使用し、エラー情報の設定には DshPutTPRJ_ERR_INFO()関数を使用します。

3．20．3．16 DshPutTPRJ_ERR_PRJID () - プロセスジョブ生成応答情報 - PRJID の設定

（1）呼出書式

[c,C++]

```
API void APIX   DshPutTPRJ_ERR_PRJID(
        TPRJ_ERR_INFO  *errinfo,            // ｴﾗｰ情報格納構造体ﾘｽﾄのﾎﾟｲﾝﾀ
        char        *prjid                  // ﾌﾟﾛｾｽｼﾞｮﾌﾞ ID が格納されている領域のﾎﾟｲﾝﾀ
);
```

[.NET VB]

```
Function DshPutTPRJ_ERR_PRJID (
        ByRef errinfo As dsh_info.TPRJ_ERR_INFO,
        ByVal prjid As String) As Int32
```

[.NET C#]

```
int DshPutTPRJ_ERR_PRJID(
        ref TPRJ_ERR_INFO errinfo,
        byte[] prjid );
```

（2）引数

errinfo

プロセスジョブエラー情報構造体のポインタです。

prjid

プロセスジョブ ID が格納されている領域のポインタです。

（3）戻り値

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

（4）説明

本関数は、errinfo 内の prj_list リストの先頭から空きリストを探します。
もし、空きリストがなければ、(-1)を返却します。
もし、空きリストがあれば、その空きリストに prjid で指定されたプロセスジョブ ID を加え、0 を返却します。

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

3．20．3．17 DshPutTPRJ_ERR_INFO () - プロセスジョブ生成応答情報の設定

（1）呼出書式

[c,C++]

```
API void APIX   DshPutTPRJ_ERR_INFO (
        TPRJ_ERR_INFO  *errinfo,            // ｴﾗｰ情報格納構造体ﾘｽﾄのﾎﾟｲﾝﾀ
        int         errcode,                // error code
        char        *errtext                // error text
);
```

[.NET VB]

```
Function DshPutTPRJ_ERR_INFO (
        ByRef errinfo As dsh_info.TPRJ_ERR_INFO,
        ByVal errcode As Int32,
        ByVal errtext As String) As Int32
```

[.NET C#]

```
int DshPutTPRJ_ERR_INFO(
        ref TPRJ_ERR_INFO errinfo,
        int errcode,
        byte[] errtext );
```

（2）引数

errinfo

プロセスジョブエラー情報構造体のポインタです。

errcode

設定するエラーコードです。(メッセージ内のアイテムは U1(51)です。)

errtext

設定するエラーテキストが格納されている領域のポインタです。

（3）戻り値

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

（4）説明

本関数は、errinfo 内の errlist リストの先頭から空きリストを探します。
もし、空きリストがなければ、(-1)を返却します。
もし、空きリストがあれば、その空きリストに１個 TERR_INFO 構造体領域を設け、その構造体に ercode と errtext を格納し、0 を返却します。TERR_INFO と内部メンバーのメモリは本関数が取得します。

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

３．20．３．18　DshFreeTPRJ_ERR_INFO() - プロセスジョブ応答情報メモリの開放

（1）呼出書式

[c,C++]

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

[.NET VB]

```
Sub DshFreeTPRJ_ERR_INFO (
        ByRef info As dsh_info.TPRJ_ERR_INFO)
```

[.NET C#]

```
void DshFreeTPRJ_ERR_INFO(
        ref TPRJ_ERR_INFO info );
```

（2）引数

erinfo

メモリを解放したいプロセスジョブ応答情報構造体のポインタです。

（3）戻り値

なし。

（4）説明

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

3．20. 3．19 DshMakeS16F11Response() - S16F11 の応答メッセージの生成

（1）呼出書式

[c,C++]

```
API int APIX DshMakeS16F11Response(
        TPRJ_INFO *info,                // ﾌﾟﾛｾｽｼﾞｮﾌﾞ情報格納領域ﾎﾟｲﾝﾀ
        TPRJ_ERR_INFO *erinfo,          // S16F12 に設定する応答情報格納領域のﾎﾟｲﾝﾀ
        DSHMSG *smsg,                   // S16F12 ﾒｯｾｰｼﾞを格納するﾒｯｾｰｼﾞ構造体のﾎﾟｲﾝﾀ
        BYTE   *buff,                   // S16F12 のﾃｷｽﾄ格納ﾊﾞｯﾌｧﾎﾟｲﾝﾀ
        int    buff_size                // buff のﾊﾞｲﾄｻｲｽﾞ
);
```

[.NET VB]

```
Function DshMakeS16F11Response (
          ByRef pinfo As dsh_info.TPRJ_INFO,
          ByRef erinfo As dsh_info.TPRJ_ERR_INFO,
          ByRef smsg As dshdr2.DSHMSG,
          ByRef buff As Byte,
          ByVal buff_size As Int32) As Int32
```

[.NET C#]

```
int DshMakeS16F11Response(
          ref TPRJ_INFO pinfo,
          ref TPRJ_ERR_INFO erinfo,
          ref DSHMSG smsg,
          byte[] buff,
          int buff_size );
```

（2）引数

info

プロセスジョブ情報が格納されている領域のポインタです。

erinfo

S16F12 メッセージに設定する応答情報が格納されている領域のポインタです。

msg

S16F12 応答メッセージ情報を格納するためのメッセージ構造体のポインタです。

buff

S16F12 応答メッセージのテキストを格納するためのバッファポインタです。

buff_size

buff のバイトサイズです。

（3）戻り値

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

（4）説明

S16F11 に対する S16F12 応答メッセージを info に含まれるプロセスジョブ生成情報と応答情報に従って作成します。

応答情報内の、acka を S16F12 の ACKA として設定します。
acka はユーザが S16F11 プロセスジョブ生成メッセージを評価した結果です。

erinfo 情報の生成、設定に DshInitTPRJ_ERR_INFO()、DshPutTPRJ_ERR_PRJID()、DshPutTPRJ_ERR_INFO() 関数を使用することができます。

３．20．３．20　DshDecodeS16F15() - S16F15 をプロセスジョブ情報リストにデコード

（1）呼出書式

[c,C++]

```
API int  APIX DshDecodeS16F15(
        DSHMSG *smsg,                    // SECS ﾒｯｾｰｼﾞ情報構造体のﾎﾟｲﾝﾀ
        TPRJ_LIST *list                  // ﾃﾞｺｰﾄﾞした情報を格納する構造体のﾎﾟｲﾝﾀ
);
```

[.NET VB]

```
Function DshDecodeS16F15 (
         ByRef smsg As dshdr2.DSHMSG,
         ByRef tlist As dsh_info.TPRJ_LIST) As Int32
```

[.NET C#]

```
int DshDecodeS16F15(
         ref DSHMSG smsg,
         ref TPRJ_LIST tlist );
```

（2）引数

smsg

S16F15 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。

list

デコードした１個以上のプロセスジョブ情報を格納するためのリスト構造体のポインタです。

（3）戻り値

戻り値	意味
0	正常にﾃﾞｺｰﾄﾞできた。
(-1)	smsg を正しくﾃﾞｺｰﾄﾞできなかった。

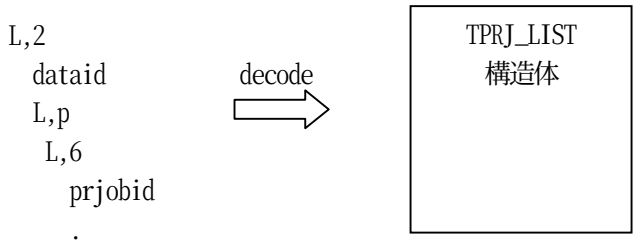
（4）説明

S16F15 メッセージに含まれる１個以上のプロセスジョブ情報を、ユーザプログラムが処理しやすい TPRJ_LIST 構造体の中にデコードします。

TPRJ_LIST は TPRJ_INFO 情報を複数個格納するためのリストです。

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

smsg S16F15



３．20. ３．21 DshFreeTPRJ_LIST() - プロセスジョブ情報構造体リストメモリの開放

（1）呼出書式

[c,C++]

```
API void APIX DshFreeTPRJ_LIST(
      TPRJ_LIST *plist              // ﾒﾓﾘを開放したい情報ﾘｽﾄが格納されている構造体のﾎﾟｲﾝﾀ
);
```

[.NET VB]

```
Sub DshFreeTPRJ_LIST (
        ByRef list As dsh_info.TPRJ_LIST)
```

[.NET C#]

```
void DshFreeTPRJ_LIST(
        ref TPRJ_LIST list );
```

（2）引数

plist

メモリを解放したいプロセスジョブ情報構造体リストのポインタです。

（3）戻り値

なし。

（4）説明

TPRJ_LIST 構造体内で情報格納用に使用されているメモリを全て解放します。
開放した後、TPRJ_LIST の内容を 全て 0 で初期設定します。
plist が NULL ならば、何も処理しません。

３．20. ３．22 DshCopyTPRJ_LIST() - プロセスジョブ情報構造体リストメモリのコピー

（1）呼出書式

[c,C++]

```
API int APIX   DshCopyTPRJ_LIST(
        TPRJ_LIST *dlist,                  // ｺﾋﾟｰ先のﾎﾟｲﾝﾀ
        TPRJ_LIST *slist                   // ｺﾋﾟｰ元のﾎﾟｲﾝﾀ
);
```

[.NET VB]

```
Function DshCopyTPRJ_LIST (
          ByRef list As dsh_info.TPRJ_LIST,
          ByRef slist As dsh_info.TPRJ_LIST) As Int32
```

[.NET C#]

```
int DshCopyTPRJ_LIST(
          ref TPRJ_LIST list,
          ref TPRJ_LIST slist );
```

（2）引数

dlist

プロセスジョブ情報リストのコピー先ポインタです。

slist

コピー元のプロセスジョブ情報リストが格納されている構造体メモリのポインタです。

（3）戻り値

戻り値	意味
0	正常にｺﾋﾟｰできた。
(-1)	sinfo または dinfo の値が NULL だったのでｺﾋﾟｰできなかった。

（4）説明

slist が指す TPRJ_LIST 構造体内に格納されているプロセスジョブ情報を dlist が指定する TPRJ_LIST 構造体にコピーします。

slist 内のメンバーで新しいメモリが必要なものは本関数が取得します。

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

３．20. ３．23 DshInitTPRJ_LIST - プロセスジョブリスト TPRJ_LIST の初期設定

（1）呼出書式

[c,C++]

```
API void  APIX DshInitTPRJ_LIST(
      TPRJ_LIST    *list,              // ﾌﾟﾛｾｽｼﾞｮﾌﾞﾘｽﾄ TPRJ_LIST 構造体のﾎﾟｲﾝﾀ
      int          prj_count           // ﾘｽﾄに設定できるﾌﾟﾛｾｽｼﾞｮﾌﾞ数
);
```

[.NET VB]

```
Sub DshInitTPRJ_LIST (
        ByRef list As dsh_info.TPRJ_LIST,
        ByVal prj_count As Int32)
```

[.NET C#]

```
void DshInitTPRJ_LIST(
        ref TPRJ_LIST list,
        int prj_count );
```

（2）引数

list

プロセスジョブ TPRJ_LIST 構造体のポインタです。このメンバーを初期設定します。

prj_count

プロセスジョブリストに設定できるプロセスジョブの数です。

（3）戻り値

なし。

（4）説明

list で指定された TPRJ_LIST 構造体に prj_count 分の TPRJ_INFO 構造体のポインタを格納できるリストを生成します。
TPRJ_INFO は１個のプロセスジョブ情報を格納するための構造体です。

list にプロセスジョブ情報（TPRJ_INFO 構造体）を加えるためには DshAddTPRJ_LIST()関数を使用してください。

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

3．20. 3．24 DshAddTPRJ_LIST() - プロセスジョブ情報をリストに追加

（1）呼出書式

[c,C++]

```
API int APIX DshAddTRPJ_LIST(
        TPRJ_LIST   *list,              // ﾌﾟﾛｾｽｼﾞｮﾌﾞﾘｽﾄ情報構造体のﾎﾟｲﾝﾀ
        TPRJ_INFO   *info               // ﾌﾟﾛｾｽｼﾞｮﾌﾞ情報が格納されている構造体ﾎﾟｲﾝﾀ
);
```

[.NET VB]

```
Function DshAddTPRJ_LIST (
          ByRef list As dsh_info.TPRJ_LIST,
          ByRef prj_info As dsh_info.TPRJ_INFO) As Int32
```

[.NET C#]

```
int DshAddTPRJ_LIST(
          ref TPRJ_LIST list,
          ref TPRJ_INFO prj_info );
```

（2）引数

list

プロセスジョブリスト情報構造体のポインタです。

info

加えたいプロセスジョブ情報が格納されている構造体のポインタです。

（3）戻り値

戻り値	意味
0	正常に追加できた。
(-1)	ﾌﾟﾛｾｽｼﾞｮﾌﾞの数が既に指定数に達している。

（4）説明

先に DshInitPrjInfo()で初期設定されたプロセスジョブリスト構造体 list の prj_list メンバーリストにプロセスジョブ情報 info の内容を加えます。

実際には、info の内容を DshCopyTPRJ_INFO()関数を使って別メモリにコピーした上でそのポインタを prj_list メンバーに加えます。

３．20. ３．25 DshMakeS16F15Response() - S16F15 の応答メッセージの生成

（1）呼出書式

[c,C++]

```
API int APIX DshMakeS16F15Response(
        TPRJ_LIST *list,                // ﾌﾟﾛｾｽｼﾞｮﾌﾞﾘｽﾄ情報格納領域ﾎﾟｲﾝﾀ
        TPRJ_ERR_INFO *erinfo,          // S16F16 に設定する応答情報格納領域のﾎﾟｲﾝﾀ
        DSHMSG *smsg,                   // S16F16 ﾒｯｾｰｼﾞを格納するﾒｯｾｰｼﾞ構造体のﾎﾟｲﾝﾀ
        BYTE   *buff,                   // S16F16 のﾃｷｽﾄ格納ﾊﾞｯﾌｧﾎﾟｲﾝﾀ
        int    buff_size                // buff のﾊﾞｲﾄｻｲｽﾞ
);
```

[.NET VB]

```
Function DshMakeS16F15Response (
          ByRef pinfo As dsh_info.TPRJ_LIST,
          ByRef erinfo As dsh_info.TPRJ_ERR_INFO,
          ByRef smsg As dshdr2.DSHMSG,
          ByRef buff As Byte,
          ByVal buff_size As Int32) As Int32
```

[.NET C#]

```
int DshMakeS16F15Response(
          ref TPRJ_LIST pinfo,
          ref TPRJ_ERR_INFO erinfo,
          ref DSHMSG smsg,
          byte[] buff,
          int buff_size );
```

（2）引数

list

プロセスジョブリスト情報が格納されている領域のポインタです。

erinfo

S16F16 メッセージに設定する応答情報が格納されている領域のポインタです。

msg

S16F16 応答メッセージ情報を格納するためのメッセージ構造体のポインタです。

buff

S16F16 応答メッセージのテキストを格納するためのバッファポインタです。

buff_size

buff のバイトサイズです。

（3）戻り値

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

（4）説明

S16F15 に対する S16F16 応答メッセージを list に含まれる 1 個以上のプロセスジョブ生成情報リストと応答情報に従って作成します。

応答情報内の、acka を S16F16 の ACKA として設定します。
acka はユーザが S16F15 プロセス生成メッセージを評価した結果です。

３．20. ３．26 DshDecodeS16F17() - S16F17 をプロセスジョブデキュー情報にデコード

（1）呼出書式

[c,C++]

```
API int  APIX DshDecodeS16F17(
        DSHMSG *smsg,                    // SECS ﾒｯｾｰｼﾞ情報構造体のﾎﾟｲﾝﾀ
        TPRJ_DEQ_INFO *info              // ﾃﾞｺｰﾄﾞした情報を格納する構造体のﾎﾟｲﾝﾀ
);
```

[.NET VB]

```
Function DshDecodeS16F17 (
        ByRef smsg As dshdr2.DSHMSG,
        ByRef info As dsh_info.TPRJ_DEQ_INFO) As Int32
```

[.NET C#]

```
int DshDecodeS16F17(
        ref DSHMSG smsg,
        ref TPRJ_DEQ_INFO info );
```

（2）引数

smsg

S16F17 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。

info

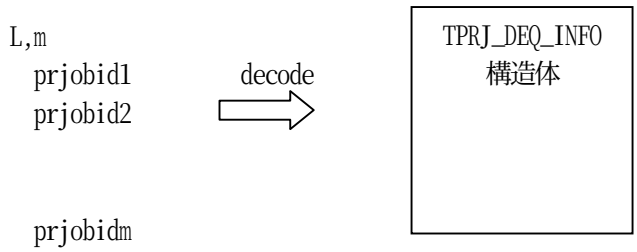
デコードしたプロセスジョブデキュー情報を格納するための構造体のポインタです。

（3）戻り値

戻り値	意味
0	正常にﾃﾞｺｰﾄﾞできた。
(-1)	smsg を正しくﾃﾞｺｰﾄﾞできなかった。

（4）説明

S16F17 は１個以上のデキューしたいプロセスジョブ ID 情報を含むメッセージです。
プロセスジョブ数が０個の場合は、全プロセスジョブ情報のデキューを意味します。
S16F17 メッセージに含まれるプロセスジョブ ID 情報を、ユーザプログラムが処理しやすい TPRJ_DEQ_INFO 構造体の中にデコードします。
なお、構造体使用後は、構造体内部で使用されたメモリを DshFreeTPRJ_DEQ_INFO()関数を使って開放してください。



３．20．３．27　DshFreeTPRJ_DEQ_INFO() - プロセスジョブDEQUE情報構造体メモリの開放

（1）呼出書式

[c,C++]

```
API void APIX DshFreeTPRJ_DEQ_INFO(
      TPRJ_DEQ_INFO *info              // ﾒﾓﾘを開放したい情報が格納されている構造体のﾎﾟｲﾝﾀ
);
```

[.NET VB]

```
Sub DshFreeTPRJ_DEQ_INFO (
        ByRef info As dsh_info.TPRJ_DEQ_INFO)
```

[.NET C#]

```
void DshFreeTPRJ_DEQ_INFO(
        ref TPRJ_DEQ_INFO info );
```

（2）引数

info

メモリを解放したいプロセスジョブDEQUE情報構造体のポインタです。

（3）戻り値

なし。

（4）説明

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

３．20．３．28 DshInitTPRJ_DEQ_ERR_INFO () - プロセスジョブデキュー応答情報の初期化

（1）呼出書式

[c,C++]

```
API void APIX   DshInitTPRJ_DEQ_ERR_INFO(
        TPRJ_DEQ_ERR_INFO *errinfo,          // ｴﾗｰ情報格納構造体のﾎﾟｲﾝﾀ
        TPRJ_DEQ_INFO *list,                 // ﾌﾟﾛｾｽｼﾞｮﾌﾞﾃﾞｷｭｰ情報ﾘｽﾄ格納構造体のﾎﾟｲﾝﾀ
        int         acka,                    // ackﾃﾞｰﾀ
        int         err_count                // ｴﾗｰ情報のﾘｽﾄｻｲｽﾞ（個数  0,1,2...）
);
```

[.NET VB]

```
Sub DshInitTPRJ_DEQ_ERR_INFO (
        ByRef errinfo As dsh_info.TPRJ_DEQ_ERR_INFO,
        ByRef info As dsh_info.TPRJ_DEQ_INFO,
        ByVal acka As Int32,
        ByVal errcount As Int32)
```

[.NET C#]

```
void DshInitTPRJ_DEQ_ERR_INFO(
        ref TPRJ_DEQ_ERR_INFO errinfo,
        ref TPRJ_DEQ_INFO info,
        int acka,
        int errcount );
```

（2）引数

errinfo

TPRJ_DEQ_ERR_INFO 応答情報構造体のポインタです。

list

プロセスジョブデキュー情報リスト格納構造体のポインタです。(S16F17 のデコードで得られた)

acka

acka - ACK の値です。

err_count

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

（3）戻り値

なし。

（4）説明

本関数は、S16F17 に対する応答メッセージのための応答情報を TPRJ_DEQ_ERR_INFO 構造体に初期設定するために使用します。

errinfo で指定された構造体の acka メンバーに引数 acka の値を設定し、prj_count, prj_list メンバーには list に含まれるプロセスジョブ ID を設定します。そして err_count メンバーには引数 err_count を設定します。

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

err_info へのエラー情報の設定には DshPutTERR_INFO_LIST ()関数を使用します。

３．20. ３．29 DshFreeTPRJ_DEQ_ERR_INFO() - プロセスジョブDEQUE応答情報メモリの開放

（1）呼出書式

[c,C++]

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

[.NET VB]

```
Sub DshFreeTPRJ_DEQ_ERR_INFO (
        ByRef info As dsh_info.TPRJ_DEQ_ERR_INFO)
```

[.NET C#]

```
void DshFreeTPRJ_DEQ_ERR_INFO(
        ref TPRJ_DEQ_ERR_INFO info );
```

（2）引数

erinfo

メモリを解放したいプロセスジョブ応答情報構造体のポインタです。

（3）戻り値

なし。

（4）説明

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

prj_list, err_listに使用されているメモリです。

3．20. 3．30 DshMakeS16F17Response() - S16F17 の応答メッセージの生成

（1）呼出書式

e[c,C++]

```
API int APIX DshMakeS16F17Response(
        TPRJ_DEQ_INFO *info,            // ﾌﾟﾛｾｽｼﾞｮﾌﾞﾃﾞｷｭｰ情報格納領域ﾎﾟｲﾝﾀ
        TPRJ_DEQ_ERR_INFO *erinfo,      // S16F18 に設定する応答情報格納領域のﾎﾟｲﾝﾀ
        DSHMSG *smsg,                   // S16F18 ﾒｯｾｰｼﾞを格納するﾒｯｾｰｼﾞ構造体のﾎﾟｲﾝﾀ
        BYTE   *buff,                   // S16F18 のﾃｷｽﾄ格納ﾊﾞｯﾌｧﾎﾟｲﾝﾀ
        int    buff_size                // buff のﾊﾞｲﾄｻｲｽﾞ
);
```

[.NET VB]

```
Function DshMakeS16F17Response (
        ByRef info As dsh_info.TPRJ_DEQ_INFO,
        ByRef erinfo As dsh_info.TPRJ_DEQ_ERR_INFO,
        ByRef smsg As dshdr2.DSHMSG,
        ByRef buff As Byte,
        ByVal buff_size As Int32) As Int32
```

[.NET C#]

```
int DshMakeS16F17Response(
        ref TPRJ_DEQ_INFO info,
        ref TPRJ_DEQ_ERR_INFO erinfo,
        ref DSHMSG smsg,
        byte[] buff,
        int buff_size );
```

（2）引数

info

プロセスジョブデキュー情報が格納されている領域のポインタです。

erinfo

S16F18 メッセージに設定する応答情報が格納されている領域のポインタです。

msg

S16F18 応答メッセージ情報を格納するためのメッセージ構造体のポインタです。

buff

S16F18 応答メッセージのテキストを格納するためのバッファポインタです。

buff_size

buff のバイトサイズです。

（3）戻り値

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

（4）説明

S16F17 に対する S16F18 応答メッセージを info に含まれるプロセスジョブデキュー情報と応答情報に従って作成します。

応答情報内の、acka を S16F18 の ACKA として設定します。
acka はユーザが S16F17 プロセスジョブデキューメッセージを評価した結果です。

erinfo 情報の生成、設定に DshInitTPRJ_DEQ_ERR_INFO()、DshPutTERR_INFO_LIST()関数を使用することができます。

３．20. ３．31 DshFreeTPRJ_STATE_LIST() - プロセスジョブDEQUE情報構造体メモリの開放

（1）呼出書式

[c,C++]

```
API void APIX DshFreeTPRJ_STATE_LIST(
      TPRJ_STATE_LIST *info              // ﾒﾓﾘを開放したい情報が格納されている構造体のﾎﾟｲﾝﾀ
);
```

[.NET VB]

```
Sub DshFreeTPRJ_STATE_LIST (
        ByRef list As dsh_info.TPRJ_STATE_LIST)
```

[.NET C#]

```
void DshFreeTPRJ_STATE_LIST(
        ref TPRJ_STATE_LIST list );
```

（2）引数

info

メモリを解放したいプロセスジョブ状態情報リスト構造体のポインタです。

（3）戻り値

なし。

（4）説明

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

３．20．４　ユーザ作成ライブラリ関数

３．20．４．１　DshResponseS16F6() – S16F6 プロセスジョブコマンド要求応答メッセージ

（1）呼出書式

[c,C++]

```
API int APIX DshResponseS16F6(
        ID_TR  trid,                    // DSHDR2 のﾄﾗﾝｻﾞｸｼｮﾝ ID
        TPRJ_CMD_INFO *info,            // ﾌﾟﾛｾｽｼﾞｮﾌﾞｺﾏﾝﾄﾞ要求ﾒｯｾｰｼﾞ情報格納領域のﾎﾟｲﾝ
        TPRJ_CMD_ERR_INFO *erinfo       // S16F6 応答情報格納用構造体のﾎﾟｲﾝﾀ
);
```

[.NET VB]

```
Function DshResponseS16F6 (
        ByVal trid As Int32,
        ByRef info As dsh_info.TPRJ_CMD_INFO,
        ByRef erinfo As dsh_info.TPRJ_CMD_ERR_INFO) As Int32
```

[.NET C#]

```
int DshResponseS16F6(
        uint trid,
        ref TPRJ_CMD_INFO info,
        ref TPRJ_CMD_ERR_INFO erinfo );
```

（2）引数

trid

S16F5 受信時に DSHEng4 から渡される DSHDR2 通信ドライバーのトランザクション管理のための ID です。

info

プロセスジョブコマンド要求情報が格納されている構造体のポインタです。

erinfo

送信する応答メッセージ S16F6 に含まれる情報を格納するための構造体領域のポインタを指定します。

（3）戻り値

戻り値	意味
0	正常に送信できた。
(-1)	送信できなかった。

（4）説明

プロセスジョブコマンド要求メッセージ S16F5 に対する応答メッセージを送信します。

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

引数に指定されている TPRJ_CMD_ERR_INFO 構造体に含まれている情報から S16F6 メッセージを組み立て、その後、S16F6 メッセージを送信します。

送信が終ったら、TPRJ_CMD_ERR_INFO の構造体に使用されたメモリを DshFreeTPRJ_CMD_ERR_INFO ()関数を使って開放します。

なお、S16F6 メッセージの組み立てに、DshMakeS16F6Response()関数を使用できます。

3．20．４．２　DshResponseS16F12() – S16F12 プロセスジョブ生成要求応答メッセージ

（1）呼出書式

[c,C++]

```
API int APIX DshResponseS16F12(
        ID_TR  trid,                    // DSHDR2 のﾄﾗﾝｻﾞｸｼｮﾝ ID
        TPRJ_INFO *info,                // ﾌﾟﾛｾｽｼﾞｮﾌﾞ生成要求ﾒｯｾｰｼﾞ情報格納領域のﾎﾟｲﾝﾀ
        TPRJ_ERR_INFO *erinfo           // S16F12 応答情報格納用構造体のﾎﾟｲﾝﾀ
);
```

[.NET VB]

```
Function DshResponseS16F12 (
        ByVal trid As Int32,
        ByRef info As dsh_info.TPRJ_INFO,
        ByRef erinfo As dsh_info.TPRJ_ERR_INFO) As Int32
```

[.NET C#]

```
int DshResponseS16F12(
        uint trid,
        ref TPRJ_INFO info,
        ref TPRJ_ERR_INFO erinfo );
```

（2）引数

trid

S16F11 受信時に DSHEng4 から渡される DSHDR2 通信ドライバーのトランザクション管理のための ID です。

info

プロセスジョブ生成要求情報が格納されている構造体のポインタです。

erinfo

送信する応答メッセージ S16F12 に含まれる情報を格納するための構造体領域のポインタを指定します。

（3）戻り値

戻り値	意味
0	正常に送信できた。
(-1)	送信できなかった。

（4）説明

プロセスジョブ生成要求メッセージ S16F11 に対する応答メッセージを送信します。

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

引数に指定されている TPRJ_ERR_INFO 構造体に含まれている情報から S16F12 メッセージを組み立て、その後、S16F12 メッセージを送信します。

送信が終ったら、TPRJ_ERR_INFO の構造体に使用されたメモリを DshFreeTPRJ_ERR_INFO ()関数を使って開

放します。

なお、S16F12 メッセージの組み立てに、DshMakeS16F12Response()関数を使用できます。

３．20．４．３　DshResponseS16F16() – S16F16 プロセスジョブマルチ生成要求応答メッセージ

（1）呼出書式

[c,C++]

```
API int APIX DshResponseS16F16(
        ID_TR  trid,                    // DSHDR2 のﾄﾗﾝｻﾞｸｼｮﾝ ID
        TPRJ_LIST *info,                // ﾌﾟﾛｾｽｼﾞｮﾌﾞ生成要求ﾒｯｾｰｼﾞ情報格納領域のﾎﾟｲﾝﾀ
        TPRJ_ERR_INFO *erinfo           // S16F16 応答情報格納用構造体のﾎﾟｲﾝﾀ
);
```

[.NET VB]

```
Function DshResponseS16F16 (
        ByVal trid As Int32,
        ByRef pinfo As dsh_info.TPRJ_LIST,
        ByRef erinfo As dsh_info.TPRJ_ERR_INFO) As Int32
```

[.NET C#]

```
int DshResponseS16F16(
        uint trid,
        ref TPRJ_LIST pinfo,
        ref TPRJ_ERR_INFO erinfo );
```

（2）引数

trid

S16F15 受信時に DSHEng4 から渡される DSHDR2 通信ドライバーのトランザクション管理のための ID です。

info

プロセスジョブ生成（MULTI）要求情報が格納されている構造体のポインタです。

erinfo

送信する応答メッセージ S16F16 に含まれる情報を格納するための構造体領域のポインタを指定します。

（3）戻り値

戻り値	意味
0	正常に送信できた。
(-1)	送信できなかった。

（4）説明

プロセスジョブ生成要求メッセージ S16F15 に対する応答メッセージを送信します。

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

引数に指定されている TPRJ_CMD_ERR_INFO 構造体に含まれている情報から S16F16 メッセージを組み立て、その後、S16F16 メッセージを送信します。

送信が終ったら、TPRJ_ERR_INFO の構造体に使用されたメモリを DshFreeTPRJ_ERR_INFO ()関数を使って開

放します。

なお、S16F16 メッセージの組み立てに、DshMakeS16F16Response()関数を使用できます。

３．20．４．４　DshResponseS16F18() – S16F18 プロセスジョブデキュー要求応答メッセージ

（1）呼出書式

[c,C++]

```
API int APIX DshResponseS16F18(
        ID_TR  trid,                    // DSHDR2 のﾄﾗﾝｻﾞｸｼｮﾝ ID
        TPRJ_DEQ_INFO *info,            // ﾌﾟﾛｾｽｼﾞｮﾌﾞﾃﾞｷｭｰ要求ﾒｯｾｰｼﾞ情報格納領域のﾎﾟｲﾝﾀ
        TPRJ_DEQ_ERR_INFO *erinfo       // S16F18 応答情報格納用構造体のﾎﾟｲﾝﾀ
);
```

[.NET VB]

```
Function DshResponseS16F18 (
        ByVal trid As Int32,
        ByRef info As dsh_info.TPRJ_DEQ_INFO,
        ByRef erinfo As dsh_info.TPRJ_DEQ_ERR_INFO) As Int32
```

[.NET C#]

```
int DshResponseS16F18(
        uint trid,
        ref TPRJ_DEQ_INFO info,
        ref TPRJ_DEQ_ERR_INFO erinfo );
```

（2）引数

trid

S16F17 受信時に DSHEng4 から渡される DSHDR2 通信ドライバーのトランザクション管理のための ID です。

info

プロセスジョブデキュー要求情報が格納されている構造体のポインタです。

erinfo

送信する応答メッセージ S16F18 に含まれる情報を格納するための構造体領域のポインタを指定します。

（3）戻り値

戻り値	意味
0	正常に送信できた。
(-1)	送信できなかった。

（4）説明

プロセスジョブデキュー要求メッセージ S16F17 に対する応答メッセージを送信します。

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

引数に指定されている TPRJ_DEQ_ERR_INFO 構造体に含まれている情報から S16F18 メッセージを組み立て、その後、S16F18 メッセージを送信します。

送信が終ったら、TPRJ_DEQ_ERR_INFO の構造体に使用されたメモリを DshFreeTPRJ_DEQ_ERR_INFO ()関数を

使って開放します。

なお、S16F18 メッセージの組み立てに、DshMakeS16F18Response()関数を使用できます。