DSHGEM-LIB　通信エンジンライブラリ(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-８／１５

３．１４　キャリア情報アクセスサービス関数

２００９年6月

株式会社データマップ

文書番号　DSHGEM-LIB-09-30328-02

[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】

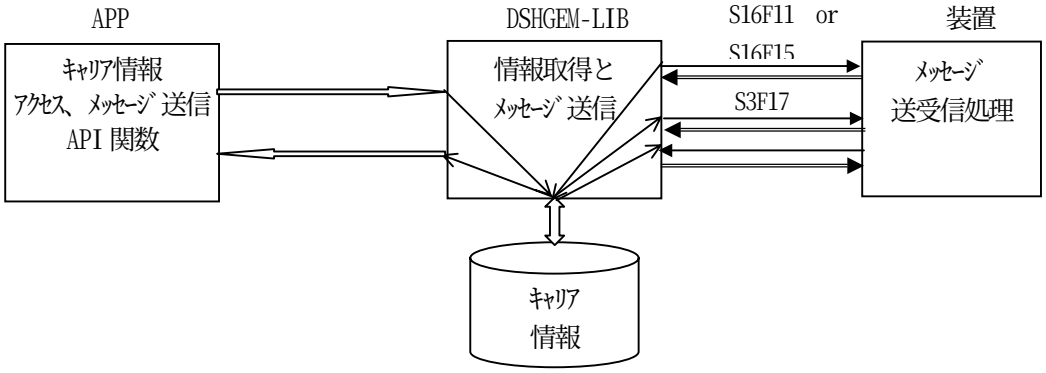
番号	改訂日付	項 目	概 略
1.	2009.6	改訂版	以前の DSHGEM-LIB-07-3032x-00 を全面改訂 .Net VB2008, C#2008 対応関数の説明を追加した。

目　次

(VOL－９に続く)

3．14 CAR キャリア情報アクセスサービス関数

ここで述べるキャリア情報は、DSHGEM-LIB 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 DSHGEM-LIB API 関数を使用します。



（1）情報アクセスと送信 API 関数

キャリア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	GemAllocCarInfo()	キャリア領域を割当て登録します。
2	GemSetCarInfo()	キャリア情報を設定・変更します。
3	GemGetCarInfo()	CARID 指定でキャリア情報を取得します。
4	GemGetCarInfoX()	キャリアインデックス指定でキャリア情報を取得します。
5	GemDelCarInfo()	CARID 指定でキャリア情報を削除します。
6	GemDelCarInfoX()	キャリアインデックス指定でキャリア情報を削除します。
7	GemGetCarId()	指定したキャリアインデックスの CARID を取得します。
8	GemGetCarIdIndex()	指定した CARID の情報インデックスを取得します。
9	GemSetCarIdStatus()	CARID 指定で CARID の状態を設定します。
10	GemSetCarIdStatusX()	キャリアインデックス指定で CARID の状態を設定します。
11	GemGetCarIdStatus()	CARID 指定で CARID の状態を取得します。
12	GemGetCarIdStatusX()	キャリアインデックス指定で CARID の状態を取得します。
13	GemSetCarMapStatus()	CARID 指定でキャリアマップの状態を設定します。
14	GemSetCarMapStatusX()	キャリアインデックス指定でキャリアマップの状態を設定します。
15	GemGetCarMapStatus()	CARID 指定でキャリアマップの状態を取得します。
16	GemGetCarMapStatusX()	キャリアインデックス指定でキャリアマップの状態を取得します。
17	GemSetCarLocation()	CARID 指定でキャリアロケーションを設定します。
18	GemSetCarLocationX()	キャリアインデックス指定でキャリアロケーションを設定します。
19	GemGetCarLocation ()	CARID 指定でキャリアロケーションを取得します。
20	GemGetCarLocationX()	キャリアインデックス指定でキャリアロケーションを取得します。
21	GemGetCarList()	キャリア ID の一覧リストを取得します。

CARID インデクスは、DSHGEM-LIB が管理する各 CARID 情報領域の番号です。このインデクスの値は、GemAllocCarInfo()関数実行時に DSHGEM-LIB によって割当てられ、APP に渡されます。また、キャリアの取得時に、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が使用できるキャリア情報処理用API関数として、以下の関数があります。

	API関数名	機能
1	DshFreeTCAR_INFO()	ｷｬﾘｱ情報が格納されている TCAR_INFO 構造体の内部で使用されているﾒﾓﾘを開放するための関数です。
2	DshCopyTCAR_INFO()	TCAR_INFO のｷｬﾘｱ情報を別の構造体にｺﾋﾟｰします。
3	DshInitCarInfo()	ｷｬﾘｱ情報 TCAR_INFO 構造体の初期設定をします。
4	DshInitCarSlotInfo()	ｽﾛｯﾄ情報 TSLOT_INFO 構造体の初期設定をします。
5	DshPutCarSlotInfo()	TCAR_INFO 構造体内にｽﾛｯﾄ情報を１個追加設定します。
6	DshPutCarSlotInfoCopy()	キャリア情報にスロット情報をコピーして設定します。

（3）ユーザ作成ライブラリ関数

	ﾗｲﾌﾞﾗﾘ関数名	機能
	なし	

3．14．1　使用する情報格納構造体

キャリア情報を操作する関数は、キャリア情報の格納のためにTCAR_INFO構造体を使用します。
キャリア情報とそのアクションなどに関連する構造体は下記のとおりです。

（1）TCAR_INFO – Carrier Information

```
typedef struct{
        int         index;              // 登録ｲﾝﾃﾞｸｽ値
        int         state;              // 状態値
        int         capacity;           // ｷｬﾘｱの容量
        char        *usage;             // ｷｬﾘｱの用途
        char        *carid;             // ｷｬﾘｱ ID
        int         map_status;         // ﾏｯﾌﾟ読込み状態
        int         id_status;          // ID 読込み状態
        int         acc_status;         // ｱｸｾｽ状態
        char        *location;          // 搬送位置
        int         slot_count;         // ｽﾛｯﾄ数
        TSLOT_INFO  **slot_list;        // ｽﾛｯﾄ情報のﾘｽﾄ
} TCAR_INFO;
```

（2）TSLOT_INFO – Carrier Slot Information

```
typedef struct{
        int         status;             // ｽﾛｯﾄ状態
        int         slotid;             // ｽﾛｯﾄ ID(U1)
        char        *mid;               // ﾏﾃﾘｱﾙ ID （ﾛｯﾄ ID）
} TSLOT_INFO;
```

（3）TCACT_INFO – Carrier Action Information - S3F17

```
typedef struct{
        TDATAID     dataid;
        char        *caction;           // car action cmd
        int         action_index;       // caction の index
        char        *carspec;           // carrier spec ( carid )
        int         ptn;                // port no.
        int         cp_count;           // parameter count
        TCACT_PARA  **cp_list;          // paramete list
}TCACT_INFO;
```

（4）TCACT_PARA – Carrier Action Parameter Information

```
typedef struct{
        char        *cattrid;           // cattrid
        int         attr_index;         // cattrid のｲﾝﾃﾞｸｽ
        void        *cattrdata;         // cattrdata
}TCACT_PARA;
```

（5）TCACT_ERR_INFO – Carrier Action Response Information – S3F18

```
typedef struct{
        int         caack;              // CAACK
        int         err_count;          // 含まれているｴﾗｰ情報の数
        TERR_INFO   **err_list;         // ｴﾗｰ情報ﾘｽﾄ
        } TCACT_ERR_INFO;
```

（6）TERR_INFO – Object 応答エラー情報

```
typedef struct{
        int     errcode;                // ｴﾗｰｺｰﾄﾞ (U1)
        char    *errtext;               // ｴﾗｰﾃｷｽﾄ
        } TERR_INFO;
```

(3.21 参照)

3．14．２　CAR キャリア情報アクセス関数

3．14．２．１　GemAllocCarInfo() - キャリアの登録

（1）呼出書式

[c,C++]

```
API int APIX GemAllocCarInfo(
        int   eqid,                 // 通信対象装置 ID(0,1,2,...)
        char *carid,                // ｷｬﾘｱ 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 GemAllocCarInfo (
          ByVal eqid As Int32,
          ByVal carid As String,
          ByRef index As Int32) As Int32
```

[.NET C#]

```
int GemAllocCarInfo(
          int eqid,
          byte[] carid,
          ref int index );
```

（2）引数

eqid

通信エンジンが通信する対象装置 ID を指定します。装置 ID は 0 から始まる番号です。

carid

登録したいキャリア ID が格納されているポインタです。

index

登録された情報領域のインデ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 CARID は既に登録されていた。
(-1)	登録できなかった。

（4）説明

キャリア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 carid で与えられるキャリア 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、carid に指定されたキャリア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。

３．14．２．２　GemSetCarInfo()　- キャリア情報の設定
.　　GemSetCarInfoX() - インデクスでのキャリア情報の設定

（1）呼出書式

[c,C++]

```
API int APIX GemSetCarInfo(
        int         eqid,               // 通信対象装置ID(0,1,2,...)
        TCAR_INFO *pinfo                // ｷｬﾘｱ情報格納構造体のﾎﾟｲﾝﾀ
);

API int APIX GemSetCarInfoX(
        int         eqid,               // 通信対象装置ID(0,1,2,...)
        int         index,              // GemAllocCarInfo()で得られたｷｬﾘｱ情報のインデクス
        TCAR_INFO *pinfo                // ｷｬﾘｱ情報格納構造体のﾎﾟｲﾝﾀ
);
```

[.NET VB]

```
Function GemSetCarInfo (
        ByVal eqid As Int32,
        ByRef cinfo As dsh_info.TCAR_INFO) As Int32

Function GemSetCarInfoX (
        ByVal eqid As Int32,
        ByVal index As Int32,
        ByRef cinfo As dsh_info.TCAR_INFO) As Int32
```

[.NET C#]

```
int GemSetCarInfo(
        int eqid,
        ref TCAR_INFO cinfo );

int GemSetCarInfoX(
        int eqid,
        int index,
        ref TCAR_INFO cinfo );
```

（2）引数

eqid

通信エンジンが通信する対象装置IDを指定します。装置IDは0から始まる番号です。

pinfo

設定したいキャリア情報が格納されている格納構造体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)	設定できなかった。

（4）説明

本関数は、pinfo に格納されているキャリア情報の設定・変更に使用します。

引数 pinfo 内の carid メンバーに指定されるキャリア ID の情報として設定されます。

pinfo 内には CARID の他、 スロット情報などの情報が含まれます。
指定した CARID が既に登録済みである場合には、pinfo 内の情報はすべて書き換えられます。

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

３．14．２．３　GemGetCarInfo() - キャリア情報の取得
.　　GemGetCarInfoX() - インデクス指定でのキャリア情報の取得

（1）呼出書式

[c,C++]

```
API int APIX GemGetCarInfo(
        int         eqid,                   // 通信対象装置 ID(0,1,2,...)
        char        *carid,                 // CARID が格納されている領域のﾎﾟｲﾝﾀ
        TCAR_INFO *pinfo                    // ｷｬﾘｱ情報を格納する構造体ﾎﾟｲﾝﾀを格納するﾎﾟｲﾝﾀ
);

API int APIX GemGetCarInfoX(
        int         eqid,                   // 通信対象装置 ID(0,1,2,...)
        int         index,                  // ｷｬﾘｱ情報のｲﾝﾃﾞｸｽ
        TCAR_INFO *pinfo                    // ｷｬﾘｱ情報を格納するﾎﾟｲﾝﾀ
);
```

[.NET VB]

```
Function GemGetCarInfo (
        ByVal eqid As Int32,
        ByVal carid As String,
        ByRef cinfo As dsh_info.TCAR_INFO) As Int32

Function GemGetCarInfoX (
        ByVal eqid As Int32,
        ByVal index As Int32,
        ByRef cinfo As dsh_info.TCAR_INFO) As I
```

[.NET C#]

```
int GemGetCarInfo(
        int eqid,
        byte[] carid,
        ref TCAR_INFO cinfo );

int GemGetCarInfoX(
        int eqid,
        int index,
        ref TCAR_INFO cinfo );
```

（2）引数

eqid

通信エンジンが通信する対象装置 ID を指定します。装置 ID は 0 から始まる番号です。

carid

キャリア ID が格納されている領域のポインタです。

pinfo

取得したキャリア情報を格納する構造体領域のポインタです。

index

キャリア ID 情報のインデクスです。登録時に GemAllocCarInfo()関数によって与えられます。

インデクスは、CARID から GemGetCarIdIndex()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。(CARID が未登録であった。)

（4）説明

carid または index に指定されているキャリアの情報を pinfo 構造体領域に格納します。

TCAR_INFO 構造体の中に情報を格納するために必要なメモリは、DSHGEM-LIB 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、carid，スロット情報などのためのメモリは DSHGEM-LIB が準備します。
これらのメモリは、使用後、ユーザが DSHGEM-LIB の API 関数を使って次のように開放してください。

```
TCAR_INFO  *pinfo;

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

３．14．２．４　GemDelCarInfo()　- キャリアの削除
.　　GemDelCarInfoX() - インデクスでのキャリアの削除

（1）呼出書式

[c,C++]

```
API int APIX GemDelCarInfo(
        int         eqid,                   // 通信対象装置 ID(0,1,2,...)
        char        *carid;                 // CARID が格納されている領域のポインタ
);

API int APIX GemDelCarInfoX(
        int         eqid,                   // 通信対象装置 ID(0,1,2,...)
        int         index,                  // ｲﾝﾃﾞｸｽ(0,1,2....)
);
```

[.NET VB]

```
Function GemDelCarInfo (
        ByVal eqid As Int32,
        ByVal carid As String) As Int32

Function GemDelCarInfoX (
        ByVal eqid As Int32,
        ByVal index As Int32) As Int32
```

[.NET C#]

```
int GemDelCarInfo(
        int eqid,
        byte[] carid );

int GemDelCarInfoX(
        int eqid,
        int index );
```

（2）引数

eqid

通信エンジンが通信する対象装置 ID を指定します。装置 ID は 0 から始まる番号です。

carid

キャリア ID が格納されている領域のポインタです。

index

削除したいキャリア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)	CARID が未登録であった。

（4）説明

carid または index に指定されているキャリア ID の情報をシステムの登録から削除します。

３．14．２．５　GemSetCarState() － キャリア情報状態の設定
.　　GemSetCarStateX() － インデクスでのキャリア情報状態の設定

（1）呼出書式

[c,C++]

```
API int APIX GemSetCarState(
        int       eqid,                 // 通信対象装置 ID(0,1,2,...)
        char      *carid,               // CARID 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 GemSetCarStateX(
        int       eqid,                 // 通信対象装置 ID(0,1,2,...)
        int       index,                // ｷｬﾘｱ情報のインデクス
        int       state                 // 設定したい状態値
);
```

[.NET VB]

```
Function GemSetCarState (
        ByVal eqid As Int32,
        ByVal carid As String,
        ByVal state As Int32) As Int32

Function GemSetCarStateX (
        ByVal eqid As Int32,
        ByVal index As Int32,
        ByVal state As Int32) As Int32
```

[.NET C#]

```
int GemSetCarState(
        int eqid,
        byte[] carid,
        int state);

int GemSetCarStateX(
        int eqid,
        int index,
        int state);
```

（2）引数

eqid

通信エンジンが通信する対象装置 ID を指定します。装置 ID は 0 から始まる番号です。

carid

キャリア ID が格納されている領域のポインタです。

state

設定したい CARID 情報の状態値です。
値=(-1)は使用不可を意味する値になります。

index

キャリア情報のインデクスです。登録時にGemAllocCarInfo()関数によって与えられます。
インデクスは、CARIDからGemGetCarIdIndex()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)	CARIDが未登録であった。

（4）説明

キャリア情報の状態値を設定します。
状態値の意味合いは値=(-1)以外についてはユーザが定義して使用します。

３．14．２．６　GemGetCarState() – キャリア情報状態の取得
.　　GemGetCarStateX() – インデクスでのキャリア情報状態の取得

（1）呼出書式

[c,C++]

```
API int APIX GemGetCarState(
        int       eqid,                 // 通信対象装置 ID(0,1,2,...)
        char      *carid,               // CARID 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 GemGetCarStateX(
        int       eqid,                 // 通信対象装置 ID(0,1,2,...)
        int       index,                // ｷｬﾘｱのインデクス
        int       *state                // 取得した状態値格納用領域ポインタ
);
```

[.NET VB]

```
Function GemGetCarState (
        ByVal eqid As Int32,
        ByVal carid As String,
        ByRef state As Int32) As Int32

Function GemGetCarStateX (
        ByVal eqid As Int32,
        ByVal index As Int32,
        ByRef state As Int32) As Int32
```

[.NET C#]

```
int GemGetCarState(
        int eqid,
        byte[] carid,
        ref int state);

int GemGetCarStateX(
        int eqid,
        int index,
        ref int state);
```

（2）引数

eqid

通信エンジンが通信する対象装置 ID を指定します。装置 ID は 0 から始まる番号です。

carid

キャリア ID が格納されている領域のポインタです。

state

取得した CARID 情報の状態値を格納する領域のポインタです。

index

キャリア情報のインデクスです。登録時に GemAllocCarInfo()関数によって与えられます。

インデクスは、CARID から GemGetCarIdIndex()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)	CARID が未登録であった。

（4）説明

キャリア情報の状態値を取得します。

状態値の意味合いは値=(-1)以外についてはユーザが定義して使用します。

3．14．２．７　GemSetCarIdStatus() － キャリア ID 読取り状態の設定
.　　GemSetCarIdStatusX() － インデクスでのキャリア ID 読取り状態の設定

（1）呼出書式

[c,C++]

```
API int APIX GemSetCarIdStatus(
        int       eqid,                 // 通信対象装置 ID(0,1,2,...)
        char      *carid,               // CARID が格納されている領域のポインタ
        int       status                // 設定したい状態値
);

API int APIX GemSetCarIdStatusX(
        int       eqid,                 // 通信対象装置 ID(0,1,2,...)
        int       index,                // ｷｬﾘｱ情報のインデクス
        int       status                // 設定したい状態値
);
```

[.NET VB]

```
Function GemSetCarIdStatus (
        ByVal eqid As Int32,
        ByVal carid As String,
        ByVal state As Int32) As Int32

Function GemSetCarIdStatusX (
        ByVal eqid As Int32,
        ByVal index As Int32,
        ByVal state As Int32) As Int32
```

[.NET C#]

```
int GemSetCarIdStatus(
        int eqid,
        byte[] carid,
        int state );

int GemSetCarIdStatusX(
        int eqid,
        int index,
        int state );
```

（2）引数

eqid

通信エンジンが通信する対象装置 ID を指定します。装置 ID は 0 から始まる番号です。

carid

キャリア ID が格納されている領域のポインタです。

status

設定したい CARID 読取りの状態値です。

index

キャリア情報のインデクスです。登録時に GemAllocCarInfo()関数によって与えられます。

インデクスは、CARID から GemGetCarIdIndex()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)	CARID が未登録であった。

（4）説明

キャリア情報の ID 読み取り状態値を設定します。
デフォルトの値は次のとおりです。

キャリア ID 状態のデフォルト値

状態値記号	値
ST_CarIdNotRead	0
ST_CarIdWaitingForHost	1
ST_CarIdVerificationOK	2
ST_CarIdVerificationFail	3
ST_CarIdNotExist	4

3．14．2．8　GemGetCarIdStatus() - キャリア ID 読取り状態の取得
.　　GemGetCarIdStatusX() - インデクスでのキャリア ID 読取り状態の取得

（1）呼出書式

[c,C++]

```
API int APIX GemGetCarIdStatus(
        int       eqid,                 // 通信対象装置 ID(0,1,2,...)
        char      *carid,               // CARID が格納されている領域のポインタ
        int       *status               // 取得した状態値格納用領域ポインタ
);

API int APIX GemGetCarIdStatusX(
        int       eqid,                 // 通信対象装置 ID(0,1,2,...)
        int       index,                // ｷｬﾘｱのインデクス
        int       *status               // 取得した状態値格納用領域ポインタ
);
```

[.NET VB]

```
Function GemGetCarIdStatus (
        ByVal eqid As Int32,
        ByVal carid As String,
        ByRef state As Int32) As Int32

Function GemGetCarIdStatusX (
        ByVal eqid As Int32,
        ByVal index As Int32,
        ByRef state As Int32) As Int32
```

[.NET C#]

```
int GemGetCarIdStatus(
        int eqid,
        byte[] carid,
        ref int state );

int GemGetCarIdStatusX(
        int eqid,
        int index,
        ref int state );
```

（2）引数

eqid

通信エンジンが通信する対象装置 ID を指定します。装置 ID は 0 から始まる番号です。

carid

キャリア ID が格納されている領域のポインタです。

status

取得した CARID の読取り状態値を格納する領域のポインタです。

index

キャリア情報のインデクスです。登録時に GemAllocCarInfo()関数によって与えられます。

インデクスは、CARID から GemGetCarIdIndex()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)	CARID が未登録であった。

（4）説明

キャリア情報の ID 読み取り状態値を取得します。
デフォルトの値は次のとおりです。

キャリア ID 状態のデフォルト値

状態値記号	値
ST_CarIdNotRead	0
ST_CarIdWaitingForHost	1
ST_CarIdVerificationOK	2
ST_CarIdVerificationFail	3
ST_CarIdNotExist	4

3．14．2．9 GemSetCarMapStatus() – スロットマップ状態の設定
. GemSetCarMapStatusX() – インデクスでのスロットマップ状態の設定

（1）呼出書式

[c,C++]

```
API int APIX GemSetCarMapStatus(
        int       eqid,                 // 通信対象装置 ID(0,1,2,...)
        char      *carid,               // CARID が格納されている領域のポインタ
        int       status                // 設定したい状態値
);

API int APIX GemSetCarMapStatusX(
        int       eqid,                 // 通信対象装置 ID(0,1,2,...)
        int       index,                // ｷｬﾘｱ情報のインデクス
        int       status                // 設定したい状態値
);
```

[.NET VB]

```
Function GemSetCarMapStatus (
        ByVal eqid As Int32,
        ByVal carid As String,
        ByVal map_status As Int32) As Int32

Function GemSetCarMapStatusX (
        ByVal eqid As Int32,
        ByVal index As Int32,
        ByVal map_status As Int32) As Int32
```

[.NET C#]

```
int GemSetCarMapStatus(
        int eqid,
        byte[] carid,
        int map_status );

int GemSetCarMapStatusX(
        int eqid,
        int index,
        int map_status );
```

（2）引数

eqid

通信エンジンが通信する対象装置 ID を指定します。装置 ID は 0 から始まる番号です。

carid

キャリア ID が格納されている領域のポインタです。

status

設定したい CARID のスロットマップ状態値です。

index

キャリア情報のインデクスです。登録時に GemAllocCarInfo()関数によって与えられます。

インデクスは、CARID から GemGetCarIdIndex()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)	CARID が未登録であった。

（4）説明

キャリア情報のスロットマップ状態値を設定します。

スロットマップ状態のデフォルト値

状態値記号	値
ST_SlotMapNotRead	0
ST_SlotMapWaitingForHost	1
ST_SlotMapVerificationOK	2
ST_SlotMapVerificationFail	3
ST_SlotNotExist	4

3．14．2．10 GemGetCarMapStatus() － スロットマップ状態の取得
. GemGetCarMapStatusX() － インデクスでのスロットマップ状態の取得

（1）呼出書式

[c,C++]

```
API int APIX GemGetCarMapStatus(
        int       eqid,                 // 通信対象装置 ID(0,1,2,...)
        char      *carid,               // CARID が格納されている領域のポインタ
        int       *status               // 取得した状態値格納用領域ポインタ
);

API int APIX GemGetCarMapStatusX(
        int       eqid,                 // 通信対象装置 ID(0,1,2,...)
        int       index,                // ｷｬﾘｱのインデクス
        int       *status               // 取得した状態値格納用領域ポインタ
);
```

[.NET VB]

```
Function GemGetCarMapStatus (
        ByVal eqid As Int32,
        ByVal carid As String,
        ByRef map_status As Int32) As Int32

Function GemGetCarMapStatusX (
        ByVal eqid As Int32,
        ByVal index As Int32,
        ByRef map_status As Int32) As Int32
```

[.NET C#]

```
int GemGetCarMapStatus(
        int eqid,
        byte[] carid,
        ref int map_status );

int GemGetCarMapStatusX(
        int eqid,
        int index,
        ref int map_status );
```

（2）引数

eqid

通信エンジンが通信する対象装置 ID を指定します。装置 ID は 0 から始まる番号です。

carid

キャリア ID が格納されている領域のポインタです。

status

取得した CARID のスロットマップ状態値を格納する領域のポインタです。

index

キャリア情報のインデクスです。登録時に GemAllocCarInfo()関数によって与えられます。

インデクスは、CARID から GemGetCarIdIndex()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)	CARID が未登録であった。

（4）説明

キャリア情報のスロットマップ状態値を取得します。

スロットマップ状態のデフォルト値

状態値記号	値
ST_SlotMapNotRead	0
ST_SlotMapWaitingForHost	1
ST_SlotMapVerificationOK	2
ST_SlotMapVerificationFail	3
ST_SlotNotExist	4

3．14．2．11 GemSetCarAccessStatus() － アクセス状態の設定
. GemSetCarAccessStatusX() － インデクスでのアクセス状態の設定

（1）呼出書式

[c,C++]

```
API int APIX GemSetCarAccessStatus(
        int       eqid,                 // 通信対象装置 ID(0,1,2,...)
        char      *carid,               // CARID が格納されている領域のポインタ
        int       status                // 設定したい状態値
);

API int APIX GemSetCarAccessStatusX(
        int       eqid,                 // 通信対象装置 ID(0,1,2,...)
        int       index,                // ｷｬﾘｱ情報のインデクス
        int       status                // 設定したい状態値
);
```

[.NET VB]

```
Function GemSetCarAccessStatus (
        ByVal eqid As Int32,
        ByVal carid As String,
        ByVal acc_status As Int32) As Int32

Function GemSetCarAccessStatusX (
        ByVal eqid As Int32,
        ByVal index As Int32,
        ByVal acc_status As Int32) As Int32
```

[.NET C#]

```
int GemSetCarAccessStatus(
        int eqid,
        byte[] carid,
        int acc_status );

int GemSetCarAccessStatusX(
        int eqid,
        int index,
        int acc_status );
```

（2）引数

carid

キャリア ID が格納されている領域のポインタです。

status

設定したい CARID のアクセス状態値です。

index

キャリア情報のインデクスです。登録時に GemAllocCarInfo()関数によって与えられます。
インデクスは、CARID から GemGetCarIdIndex()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)	CARID が未登録であった。

（4）説明

キャリア情報のアクセス状態値を設定します。

アクセス状態のデフォルト値

状態値記号	値
CAR_NOT_ACCESSED	0
CAR_IN_ACCESS	1
CAR_COMPLETE	2
CAR_STOPPED	3

3．14．2．12 GemGetCarAccessStatus() - アクセス状態の取得
. GemGetCarAccessStatusX() - インデクスでのアクセス状態の取得

（1）呼出書式

[c,C++]

```
API int APIX GemGetCarAccessStatus(
        int       eqid,                 // 通信対象装置 ID(0,1,2,...)
        char      *carid,               // CARID が格納されている領域のポインタ
        int       *status               // 取得した状態値格納用領域ポインタ
);

API int APIX GemGetCarAccessStatusX(
        int       eqid,                 // 通信対象装置 ID(0,1,2,...)
        int       index,                // ｷｬﾘｱのインデクス
        int       *status               // 取得した状態値格納用領域ポインタ
);
```

[.NET VB]

```
Function GemGetCarAccessStatus (
        ByVal eqid As Int32,
        ByVal carid As String,
        ByRef acc_status As Int32) As Int32

Function GemGetCarAccessStatusX (
        ByVal eqid As Int32,
        ByVal index As Int32,
        ByRef acc_status As Int32) As Int32
```

[.NET C#]

```
int GemGetCarAccessStatus(
        int eqid,
        byte[] carid,
        ref int acc_status );

int GemGetCarAccessStatusX(
        int eqid,
        int index,
        ref int acc_status );
```

（2）引数

eqid

通信エンジンが通信する対象装置 ID を指定します。装置 ID は 0 から始まる番号です。

carid

キャリア ID が格納されている領域のポインタです。

status

取得した CARID のアクセス状態値を格納する領域のポインタです。

index

キャリア情報のインデクスです。登録時に GemAllocCarInfo()関数によって与えられます。

インデクスは、CARID から GemGetCarIdIndex()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)	CARID が未登録であった。

（4）説明

キャリアのアクセス状態値を取得します。

アクセス状態のデフォルト値

状態値記号	値
CAR_NOT_ACCESSED	0
CAR_IN_ACCESS	1
CAR_COMPLETE	2
CAR_STOPPED	3

3．14．2．13 GemSetCarLocation() - ロケーションの設定
. GemSetCarLocationX() - インデクスでのロケーションの設定

（1）呼出書式

[c,C++]

```
API int    APIX GemSetCarLocation(
        int       eqid,                 // 通信対象装置 ID(0,1,2,...)
        char      *carid,               // CARID が格納されている領域のポインタ
        char      *location             // 設定したいロケーション格納ポインタ
);

API int    APIX GemSetCarLocationX(
        int       eqid,                 // 通信対象装置 ID(0,1,2,...)
        int       index,                // ｷｬﾘｱ情報のインデクス
        char      *location             // 設定したいロケーション格納ポインタ
);
```

[.NET VB]

```
Function GemSetCarLocation (
        ByVal eqid As Int32,
        ByVal carid As String,
        ByVal location As String) As Int32

Function GemSetCarLocationX (
        ByVal eqid As Int32,
        ByVal index As Int32,
        ByVal location As String) As Int32
```

[.NET C#]

```
int GemSetCarLocation(
        int eqid,
        byte[] carid,
        byte[] location );

int GemSetCarLocationX(
        int eqid,
        int index,
        byte[] location );
```

（2）引数

eqid

通信エンジンが通信する対象装置 ID を指定します。装置 ID は 0 から始まる番号です。

carid

キャリア ID が格納されている領域のポインタです。

location

設定したい CARID のロケーション値（文字列）です。

index

キャリア情報のインデクスです。登録時に GemAllocCarInfo()関数によって与えられます。

インデクスは、CARID から GemGetCarIdIndex()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)	CARID が未登録であった。

（4）説明

キャリア情報のロケーション(文字列)を設定します。

3．14．2．14 GemGetCarLocation() - ロケーションの取得
. GemGetCarLocationX() - インデクスでのロケーションの取得

（1）呼出書式

[c,C++]

```
API int    APIX GemGetCarLocation(
        int       eqid,                  // 通信対象装置ID(0,1,2,...)
        char      *carid,                // CARIDが格納されている領域のポインタ
        char      *location              // 取得したロケーション格納ポインタ
);

API int    APIX GemGetCarLocationX(
        int       eqid,                  // 通信対象装置ID(0,1,2,...)
        char      index,                 // ｷｬﾘｱのインデクス
        char      *location              // 取得したロケーション格納ポインタ
);
```

[.NET VB]

```
Function GemGetCarLocation (
        ByVal eqid As Int32,
        ByVal carid As String,
        ByVal location As String) As Int32

Function GemGetCarLocationX (
        ByVal eqid As Int32,
        ByVal index As Int32,
        ByVal location As String) As Int32
```

[.NET C#]

```
int GemGetCarLocation(
        int eqid,
        byte[] carid,
        byte[] location );

int GemGetCarLocationX(
        int eqid,
        int index,
        byte[] location );
```

（2）引数

eqid

通信エンジンが通信する対象装置IDを指定します。装置IDは0から始まる番号です。

carid

キャリアIDが格納されている領域のポインタです。

location

取得したCARIDのロケーション値を格納する領域のポインタです。

index

キャリア情報のインデクスです。登録時にGemAllocCarInfo()関数によって与えられます。

インデクスは、CARID から GemGetCarIdIndex()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)	CARID が未登録であった。

（4）説明

キャリア情報のロケーション(文字列)を取得します。

３．14．２．15 GemSetCarUsage() － Usage の設定
. GemSetCarUsageX() － インデクスでの Usage の設定

（1）呼出書式

[c,C++]

```
API int    APIX GemSetCarUsage(
        int       eqid,                 // 通信対象装置 ID(0,1,2,...)
        char      *carid,               // CARID が格納されている領域のポインタ
        char      *usage                // 設定したい Usage 格納ポインタ
);

API int    APIX GemSetCarUsageX(
        int       eqid,                 // 通信対象装置 ID(0,1,2,...)
        int       index,                // ｷｬﾘｱ情報のインデクス
        char      *usage                // 設定したい Usage 格納ポインタ
);
```

[.NET VB]

```
Function GemSetCarUsage (
        ByVal eqid As Int32,
        ByVal carid As String,
        ByVal usage As String) As Int32

Function GemSetCarUsageX (
        ByVal eqid As Int32,
        ByVal index As Int32,
        ByVal usage As String) As Int32
```

[.NET C#]

```
int GemSetCarUsage(
        int eqid,
        byte[] carid,
        byte[] usage );

int GemSetCarUsageX(
        int eqid,
        int index,
        byte[] usage );
```

（2）引数

eqid

通信エンジンが通信する対象装置 ID を指定します。装置 ID は 0 から始まる番号です。

carid

キャリア ID が格納されている領域のポインタです。

usage

設定したい CARID の usage 値（文字列）です。

index

キャリア情報のインデクスです。登録時に GemAllocCarInfo()関数によって与えられます。

インデクスは、CARID から GemGetCarIdIndex()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)	CARID が未登録であった。

（4）説明

キャリア情報の Usage(文字列)を設定します。

３．14．２．16 GemGetCarUsage() － Usage の取得
. GemGetCarUsageX() － インデクスでの Usage の取得

（1）呼出書式

[c,C++]

```
API int    APIX GemGetCarUsage(
        int       eqid,                 // 通信対象装置 ID(0,1,2,...)
        char      *carid,               // CARID が格納されている領域のポインタ
        char      *Usage                // 取得した Usage 格納ポインタ
);

API int    APIX GemGetCarUsageX(
        int       eqid,                 // 通信対象装置 ID(0,1,2,...)
        char      index,                // ｷｬﾘｱのインデクス
        char      *Usage                // 取得した Usage 格納ポインタ
);
```

[.NET VB]

```
Function GemGetCarUsage (
        ByVal eqid As Int32,
        ByVal carid As String,
        ByVal usage As String) As Int32

Function GemGetCarUsageX (
        ByVal eqid As Int32,
        ByVal index As Int32,
        ByVal usage As String) As Int32
```

[.NET C#]

```
int GemGetCarUsage(
        int eqid,
        byte[] carid,
        byte[] usage );

int GemGetCarUsageX(
        int eqid,
        int index,
        byte[] usage );
```

（2）引数

eqid

通信エンジンが通信する対象装置 ID を指定します。装置 ID は 0 から始まる番号です。

carid

キャリア ID が格納されている領域のポインタです。

Usage

取得した CARID の Usage 値を格納する領域のポインタです。

index

キャリア情報のインデクスです。登録時に GemAllocCarInfo()関数によって与えられます。

インデクスは、CARID から GemGetCarIdIndex()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)	CARID が未登録であった。

（4）説明

キャリア情報の Usage(文字列)を取得します。

3．14．2．17 GemSetCarLotid() - Lotidの設定
. GemSetCarLotidX() - インデクスでのLotidの設定

（1）呼出書式

[c,C++]

```
API int    APIX GemSetCarLotid(
        int       eqid,                 // 通信対象装置ID(0,1,2,...)
        char      *carid,               // CARIDが格納されている領域のポインタ
        int       order,                // スロット順位(0,1,2...)
        char      *lotid                // 設定したいlotid格納ポインタ
);

API int    APIX GemSetCarLotidX(
        int       eqid,                 // 通信対象装置ID(0,1,2,...)
        int       index,                // ｷｬﾘｱ情報のインデクス
        int       order,                // スロット順位(0,1,2...)
        char      *lotid                // 設定したいlotid格納ポインタ
);
```

[.NET VB]

```
Function GemSetCarLotid (
        ByVal eqid As Int32,
        ByVal carid As String,
        ByVal order As Int32,
        ByVal lotid As String) As Int32

Function GemSetCarLotidX (
        ByVal eqid As Int32,
        ByVal index As Int32,
        ByVal order As Int32,
        ByVal lotid As String) As Int32
```

[.NET C#]

```
int GemSetCarLotid(
        int eqid,
        byte[] carid,
        int order,
        byte[] lotid );

int GemSetCarLotidX(
        int eqid,
        int index,
        int order,
        byte[] lotid );
```

（2）引数

carid

キャリアIDが格納されている領域のポインタです。

order

アクセス対象キャリアスロットの順位です。(0,1,2...)

lotid

設定したいスロットの lotid 値（文字列）です。

index

キャリア情報のインデクスです。登録時に GemAllocCarInfo()関数によって与えられます。
インデクスは、CARID から GemGetCarIdIndex()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)	CARID が未登録であった。

（4）説明

指定キャリア ID の order 番目のスロットの lotid(文字列)を設定します。

3．14．2．18 GemGetCarLotid() - Lotid の取得
. GemGetCarLotidX() - インデクスでの Lotid の取得

（1）呼出書式

[c,C++]

```
API int    APIX GemGetCarLotid(
        int       eqid,                 // 通信対象装置 ID(0,1,2,...)
        char      *carid,               // CARID が格納されている領域のポインタ
        int       order,                // スロット順位(0,1,2...)
        char      *lotid                // 取得した lotid 格納ポインタ
);

API int    APIX GemGetCarLotidX(
        int       eqid,                 // 通信対象装置 ID(0,1,2,...)
        int       index,                // ｷｬﾘｱのインデクス
        int       order,                // スロット順位(0,1,2...)
        char      *lotid                // 取得した lotid 格納ポインタ
);
```

[.NET VB]

```
Function GemGetCarLotid (
        ByVal eqid As Int32,
        ByVal carid As String,
        ByVal order As Int32,
        ByVal lotid As String) As Int32

Function GemGetCarLotidX (
        ByVal eqid As Int32,
        ByVal index As Int32,
        ByVal order As Int32,
        ByVal lotid As String) As Int32
```

[.NET C#]

```
int GemGetCarLotid(
        int eqid,
        byte[] carid,
        int order,
        byte[] lotid );

int GemGetCarLotidX(
        int eqid,
        int index,
        int order,
        byte[] lotid );
```

（2）引数

eqid

通信エンジンが通信する対象装置 ID を指定します。装置 ID は 0 から始まる番号です。

carid

キャリア ID が格納されている領域のポインタです。

order

アクセス対象キャリアスロットの順位です。(0,1,2...)

lotid

取得したスロットの lotid 値を格納する領域のポインタです。

index

キャリア情報のインデクスです。登録時に GemAllocCarInfo()関数によって与えられます。
インデクスは、CARID から GemGetCarIdIndex()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)	CARID が未登録であった。

（4）説明

指定されたキャリア Id の order 番目の lotid(文字列)を取得します。

3．14．2．19 GemSetCarSubstid() - Substid の設定
. GemSetCarSubstidX() - インデクスでの Substid の設定

（1）呼出書式

[c,C++]

```
API int    APIX GemSetCarSubstid(
        int       eqid,                 // 通信対象装置 ID(0,1,2,...)
        char      *carid,               // CARID が格納されている領域のポインタ
        int       order,                // スロット順位(0,1,2…)
        char      *substid              // 設定したい substid 格納ポインタ
);

API int    APIX GemSetCarSubstidX(
        int       eqid,                 // 通信対象装置 ID(0,1,2,...)
        int       index,                // ｷｬﾘｱ情報のインデクス
        int       order,                // スロット順位(0,1,2…)
        char      *substid              // 設定したい substid 格納ポインタ
);
```

[.NET VB]

```
Function GemSetCarSubstid (
        ByVal eqid As Int32,
        ByVal carid As String,
        ByVal order As Int32,
        ByVal substid As String) As Int32

Function GemSetCarSubstidX (
        ByVal eqid As Int32,
        ByVal index As Int32,
        ByVal order As Int32,
        ByVal substid As String) As Int32
```

[.NET C#]

```
int GemSetCarSubstid(
        int eqid,
        byte[] carid,
        int order,
        byte[] substid );

int GemSetCarSubstidX(
        int eqid,
        int index,
        int order,
        byte[] substid );
```

（2）引数

eqid

通信エンジンが通信する対象装置 ID を指定します。装置 ID は 0 から始まる番号です。

carid

キャリア ID が格納されている領域のポインタです。

order

アクセス対象キャリアスロットの順位です。(0,1,2...)

substid

設定したいスロットの substid 値（文字列）です。

index

キャリア情報のインデクスです。登録時に GemAllocCarInfo()関数によって与えられます。
インデクスは、CARID から GemGetCarIdIndex()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)	CARID が未登録であった。

（4）説明

指定キャリア ID の order 番目のスロットの substid(文字列)を設定します。

3．14．２．20 GemGetCarSubstid() － Substid の取得
. GemGetCarSubstidX() － インデクスでの Substid の取得

（1）呼出書式

[c,C++]

```
API int    APIX GemGetCarSubstid(
        int       eqid,                 // 通信対象装置 ID(0,1,2,...)
        char      *carid,               // CARID が格納されている領域のポインタ
        int       order,                // スロット順位(0,1,2…)
        char      *substid              // 取得した substid 格納ポインタ
);

API int    APIX GemGetCarSubstidX(
        int       eqid,                 // 通信対象装置 ID(0,1,2,...)
        char      index,                // ｷｬﾘｱのインデクス
        int       order,                // スロット順位(0,1,2…)
        char      *substid              // 取得した substid 格納ポインタ
);
```

[.NET VB]

```
Function GemGetCarSubstid (
        ByVal eqid As Int32,
        ByVal carid As String,
        ByVal order As Int32,
        ByVal substid As String) As Int32

Function GemGetCarSubstidX (
        ByVal eqid As Int32,
        ByVal index As Int32,
        ByVal order As Int32,
        ByVal substid As String) As Int32
```

[.NET C#]

```
int GemGetCarSubstid(
        int eqid,
        byte[] carid,
        int order,
        byte[] substid );

int GemGetCarSubstidX(
        int eqid,
        int index,
        int order,
        byte[] substid );
```

（2）引数

eqid

通信エンジンが通信する対象装置 ID を指定します。装置 ID は 0 から始まる番号です。

carid

キャリア ID が格納されている領域のポインタです。

order

アクセス対象キャリアスロットの順位です。(0,1,2...)

substid

取得したスロットの substid 値を格納する領域のポインタです。

index

キャリア情報のインデクスです。登録時に GemAllocCarInfo()関数によって与えられます。
インデクスは、CARID から GemGetCarIdIndex()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)	CARID が未登録であった。

（4）説明

指定されたキャリア Id の order 番目のスロットの基板 ID　substid(文字列)を取得します。

3．14．2．21 GemSetCarSlotmap() - Slotmap の設定
. GemSetCarSlotmapX() - インデクスでの Slotmap の設定

（1）呼出書式

[c,C++]

```
API int    APIX GemSetCarSlotmap(
        int       eqid,                 // 通信対象装置 ID(0,1,2,...)
        char      *carid,               // CARID が格納されている領域のポインタ
        int       order,                // スロット順位(0,1,2...)
        int       slotmap               // 設定したい slotmap 値
);

API int    APIX GemSetCarSlotmapX(
        int       eqid,                 // 通信対象装置 ID(0,1,2,...)
        int       index,                // ｷｬﾘｱ情報のインデクス
        int       order,                // スロット順位(0,1,2...)
        int       slotmap               // 設定したい slotmap 値
);
```

[.NET VB]

```
Function GemSetCarSlotmap (
        ByVal eqid As Int32,
        ByVal carid As String,
        ByVal order As Int32,
        ByVal slotmap As Int32) As Int32

Function GemSetCarSlotmapX (
        ByVal eqid As Int32,
        ByVal index As Int32,
        ByVal order As Int32,
        ByVal slotmap As Int32) As Int32
```

[.NET C#]

```
int GemSetCarSlotmap(
        int eqid,
        byte[] carid,
        int order,
        int slotmap );

int GemSetCarSlotmapX(
        int eqid,
        int index,
        int order,
        int slotmap );
```

（2）引数

eqid

通信エンジンが通信する対象装置 ID を指定します。装置 ID は 0 から始まる番号です。

carid

キャリア ID が格納されている領域のポインタです。

order

アクセス対象キャリアスロットの順位です。(0,1,2...)

slotmap

設定したいスロットの slotmap 値です。

index

キャリア情報のインデクスです。登録時に GemAllocCarInfo()関数によって与えられます。
インデクスは、CARID から GemGetCarIdIndex()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)	CARID が未登録であった。

（4）説明

指定キャリア ID の order 番目のスロットの slotmap を設定します。

3．14．2．22 GemGetCarSlotmap() - Slotmap の取得
. GemGetCarSlotmapX() - インデクスでの Slotmap の取得

（1）呼出書式

[c,C++]

```
API int    APIX GemGetCarSlotmap(
        int       eqid,                 // 通信対象装置 ID(0,1,2,...)
        char      *carid,               // CARID が格納されている領域のポインタ
        int       order,                // スロット順位(0,1,2...)
        int       *slotmap              // 取得した slotmap 値格納ポインタ
);

API int    APIX GemGetCarSlotmapX(
        int       eqid,                 // 通信対象装置 ID(0,1,2,...)
        char      index,                // ｷｬﾘｱのインデクス
        int       order,                // スロット順位(0,1,2...)
        int       *slotmap              // 取得した slotmap 値格納ポインタ
);
```

[.NET VB]

```
Function GemGetCarSlotmap (
        ByVal eqid As Int32,
        ByVal carid As String,
        ByVal order As Int32,
        ByRef slotmap As Int32) As Int32

Function GemGetCarSlotmapX (
        ByVal eqid As Int32,
        ByVal index As Int32,
        ByVal order As Int32,
        ByRef slotmap As Int32) As Int32
```

[.NET C#]

```
int GemGetCarSlotmap(
        int eqid,
        byte[] carid,
        int order,
        ref int slotmap );

int GemGetCarSlotmapX(
        int eqid,
        int index,
        int order,
        ref int slotmap );
```

（2）引数

eqid

通信エンジンが通信する対象装置 ID を指定します。装置 ID は 0 から始まる番号です。

carid

キャリア ID が格納されている領域のポインタです。

order

アクセス対象キャリアスロットの順位です。(0,1,2...)

slotmap

取得したスロットの slotmap 値を格納する領域のポインタです。

index

キャリア情報のインデクスです。登録時に GemAllocCarInfo()関数によって与えられます。
インデクスは、CARID から GemGetCarIdIndex()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)	CARID が未登録であった。

（4）説明

指定されたキャリア ID の order 番目のスロットの slotmap 値を取得します。

3．14．２．23 GemSetCarSlotCount() － スロット数の設定
. GemSetCarSlotCountX() － インデクスでのスロット数の設定

（1）呼出書式

[c,C++]

```
API int APIX GemSetCarSlotCount(
        int       eqid,                 // 通信対象装置 ID(0,1,2,...)
        char      *carid,               // CARID が格納されている領域のポインタ
        int       count                 // 設定したいスロット数
);

API int APIX GemSetCarSlotCountX(
        int       eqid,                 // 通信対象装置 ID(0,1,2,...)
        int       index,                // ｷｬﾘｱ情報のインデクス
        int       count                 // 設定したいスロット数
);
```

[.NET VB]

```
Function GemSetCarSlotCount (
        ByVal eqid As Int32,
        ByVal carid As String,
        ByVal count As Int32) As Int32

Function GemSetCarSlotCountX (
        ByVal eqid As Int32,
        ByVal index As Int32,
        ByVal count As Int32) As Int32
```

[.NET C#]

```
int GemSetCarSlotCount(
        int eqid,
        byte[] carid,
        int count);

int GemSetCarSlotCountX(
        int eqid,
        int index,
        int count);
```

（2）引数

eqid

通信エンジンが通信する対象装置 ID を指定します。装置 ID は 0 から始まる番号です。

carid

キャリア ID が格納されている領域のポインタです。

count

設定したい CARID のスロット数です。

index

キャリア情報のインデクスです。登録時に GemAllocCarInfo()関数によって与えられます。

インデクスは、CARID から GemGetCarIdIndex()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)	CARID が未登録であった。

（4）説明

キャリア情報のスロット数を設定します。

3．14．2．24 GemGetCarSlotCount() － スロット数の取得
. GemGetCarSlotCountX() － インデクスでのスロット数の取得

（1）呼出書式

[c,C++]

```
API int APIX GemGetCarSlotCount(
        int       eqid,                 // 通信対象装置ID(0,1,2,...)
        char      *carid,               // CARIDが格納されている領域のポインタ
        int       *count                // 取得したスロット数格納用領域ポインタ
);

API int APIX GemGetCarSlotCountX(
        int       eqid,                 // 通信対象装置ID(0,1,2,...)
        int       index,                // ｷｬﾘｱのインデクス
        int       *count                // 取得したスロット数格納用領域ポインタ
);
```

[.NET VB]

```
Function GemGetCarSlotCount (
        ByVal eqid As Int32,
        ByVal carid As String,
        ByRef count As Int32) As Int32

Function GemGetCarSlotCountX (
        ByVal eqid As Int32,
        ByVal index As Int32,
        ByRef count As Int32) As Int32
```

[.NET C#]

```
int GemGetCarSlotCount(
        int eqid,
        byte[] carid,
        ref int count );

int GemGetCarSlotCountX(
        int eqid,
        int index,
        ref int count );
```

（2）引数

eqid

通信エンジンが通信する対象装置IDを指定します。装置IDは0から始まる番号です。

carid

キャリアIDが格納されている領域のポインタです。

count

取得したCARIDのスロット数を格納する領域のポインタです。

index

キャリア情報のインデクスです。登録時にGemAllocCarInfo()関数によって与えられます。
インデクスは、CARIDからGemGetCarIdIndex()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)	CARIDが未登録であった。

（4）説明

キャリア情報のスロット数を取得します。

3．14．2．25 GemGetCarList() - 全登録キャリア ID 取得関数

（1）呼出書式

[c,C++]

```
API int APIX GemGetCarList(
      int          eqid,                  // 通信対象装置 ID(0,1,2,...)
      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 GemGetCarList (
        ByVal eqid As Int32,
        ByRef list As IntPtr) As Int32
```

[.NET C#]

```
int GemGetCarList(
        int eqid,
        IntPtr list );
```

（2）引数

eqid

通信エンジンが通信する対象装置 ID を指定します。装置 ID は 0 から始まる番号です。

list

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

（3）戻り値

戻り値	意味
0	正常に取得できた。
(-1)	取得できなかった。

（4）説明

システムに登録されている全 CARID(キャリア ID)リストを 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。(本情報には定義名がないため)
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;
```

3．14．２．26 GemGetCarId() - インデクスからCARID（キャリアID）の取得

（1）呼出書式

[c,C++]

```
API int  APIX EgnGetCarId(
        int     eqid,                   // 通信対象装置ID(0,1,2,...)
        int     index,                  // キャリア情報のインデクス
        char    *carid                  // 取得したCARIDを格納する領域のポインタ
);
```

[.NET VB]

```
Function GemGetCarId (
        ByVal eqid As Int32,
        ByVal index As Int32,
        ByVal carid As String) As Int32
```

[.NET C#]

```
int GemGetCarId(
        int eqid,
        int index,
        byte[] carid );
```

（2）引数

eqid

通信エンジンが通信する対象装置IDを指定します。装置IDは0から始まる番号です。

index

キャリア情報のインデクスです。登録時にGemAllocCarInfo()関数によって与えられます。
インデクスは、CARIDからGemGetCarIdIndex()関数で取得することができます。

carid

CARID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

キャリア情報のインデクスからCARID（キャリアID）を取得し、carid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。

3．14．２．27　GemGetCarIdIndex() –CARID（キャリア ID）からインデクスの取得

（1）呼出書式

[c,C++]

```
API int  APIX EgnGetCarIdIndex(
        int     eqid,                   // 通信対象装置 ID(0,1,2,...)
        char    *carid,                 // CARID 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 GemGetCarIdIndex (
        ByVal eqid As Int32,
        ByVal carid As String,
        ByRef index As Int32) As Int32
```

[.NET C#]

```
int GemGetCarIdIndex(
        int eqid,
        byte[] carid,
        ref int index );
```

（2）引数

eqid

通信エンジンが通信する対象装置 ID を指定します。装置 ID は 0 から始まる番号です。

carid

インデクスを取得したい対象の CARID 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

carid に指定される CARID（キャリア ID）からキャリア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。

３．14．３　CAR キャリア関連ライブラリ関数

３．14．３．１　DshFreeTCAR_INFO() - キャリア情報構造体メモリの開放

（1）呼出書式

[c,C++]

```
API void APIX DshFreeTCAR_INFO(
      TCAR_INFO *pinfo          // ﾒﾓﾘを開放したいｷｬﾘｱ情報が格納されている構造体のﾎﾟｲﾝﾀ
);
```

[.NET VB]

```
Sub DshFreeTCAR_INFO (
        ByRef info As dsh_info.TCAR_INFO)
```

[.NET C#]

```
void DshFreeTCAR_INFO(
        ref TCAR_INFO info );
```

（2）引数

pinfo

メモリを解放したいキャリア情報構造体のポインタです。

（3）戻り値

なし。

（4）説明

TCAR_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、TCAR_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。

3．14．3．2　DshCopyTCAR_INFO() - キャリア情報構造体メモリのコピー

（1）呼出書式

[c,C++]

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

[.NET VB]

```
Function DshCopyTCAR_INFO (
          ByRef dinfo As dsh_info.TCAR_INFO,
          ByRef sinfo As dsh_info.TCAR_INFO) As Int32
```

[.NET C#]

```
int DshCopyTCAR_INFO(
          ref TCAR_INFO dinfo,
          ref TCAR_INFO sinfo );
```

（2）引数

dinfo

キャリア情報のコピー先構造体メモリのポインタです。

sinfo

コピー元のキャリア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TCAR_INFO 構造体内に格納されているキャリア情報をdinfo が指定するTCAR_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、DshFreeTCAR_INFO()関数を使って開放してください。

3．14．3．3　DshInitCarInfo() - TCAR_INFO の初期設定

（1）呼出書式

[c,C++]

```
API int  APIX DshInitCarInfo(
        TCAR_INFO    *info,              // TCAR_INFO 構造体のﾎﾟｲﾝﾀ
        char         *carid,             // ｷｬﾘｱ ID
        char         *usage,             // Usage 名
        int          map_status,         // map_status
        int          id_status,          // id_status
        int          acc_status,         // acc_status
        char         *location,          // location
        int          slot_count          // slot count
);
```

[.NET VB]

```
Sub DshInitCarInfo (
        ByRef info As dsh_info.TCAR_INFO,
        ByVal carid As String,
        ByVal usage As String,
        ByVal map_status As Int32,
        ByVal id_status As Int32,
        ByVal acc_status As Int32,
        ByVal location As String,
        ByVal slot_count As Int32)
```

[.NET C#]

```
void DshInitCarInfo(
        ref TCAR_INFO info,
        byte[] carid,
        byte[] usage,
        int map_status,
        int id_status,
        int acc_status,
        byte[] location,
        int slot_count );
```

（2）引数

info

TCAR_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。

carid

設定するキャリア ID（文字列）です。

usage

材料の種類です。

map_status

キャリアの MAP STATUS です。

id_status

キャリアの ID STATUS です。

acc_status
　　キャリアの ACCESSING STATUS です。
location
　　現在の location ID です。
slot_count
　　Slot 数です。

（3）戻り値
なし。

（4）説明
本関数は APP が OFFLINE でキャリア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。

3．14．3．4　DshInitCarSlotInfo() - キャリアスロット情報の初期設定

（1）呼出書式

[c,C++]

```
API void APIX   DshInitCarSlotInfo(
        TSLOT_INFO   *info,                 // ｽﾛｯﾄ情報構造体のﾎﾟｲﾝﾀ
        int          slotid,                // ｽﾛｯﾄ ID
        char         *mid,                  // mid(material id)
        char         *substid,              // substrate id
        char         *substloc              // 現在の基板ﾛｹｰｼｮﾝ
);
```

[.NET VB]

```
Sub DshInitCarSlotInfo (
        ByRef sinfo As dsh_info.TSLOT_INFO,
        ByVal slotid As Int32,
        ByVal mid As String,
        ByVal substid As String,
        ByVal substloc As String)
```

[.NET C#]

```
void DshInitCarSlotInfo(
        ref TSLOT_INFO sinfo,
        int slotid,
        byte[] mid,
        byte[] substid,
        byte[] substloc );
```

（2）引数

info

初期設定するキャリアスロット情報の構造体ポインタです。

slotid

このスロットに与えられたスロット ID です。

mid

このスロットに含まれる基板の material(ロット)ID です。

substid

このスロットに含まれる基板の ID です。

substloc

このスロットに含まれる基板の location 位置です。

（3）戻り値

なし。

（4）説明

最初に TSLOT_INFO 構造体のメモリを確保し、info に設定します。そして引数に与えられた情報を info 内の各メンバーに設定します。

info 自体のメモリを確保するとともに、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。

ここで確保されたメモリはDshFreeTCAR_INFO()関数でTCAR_INFO構造体内メモリが開放されるときに同時に開放されます。

3．14．3．5　DshPutCarSlotInfo() - キャリア情報にスロット情報を設定

（1）呼出書式

[c,C++]

```
API int APIX DshPutCarSlotInfo(
        TCAR_INFO    *info,                 // ｷｬﾘｱ情報構造体のﾎﾟｲﾝﾀ
        TSLOT_INFO   *sinfo                 // ｽﾛｯﾄ情報構造体のﾎﾟｲﾝﾀ
);
```

[.NET VB]

```
Function DshPutCarSlotInfo (
          ByRef info As dsh_info.TCAR_INFO,
          ByRef sinfo As IntPtr) As Int32
```

[.NET C#]

```
int DshPutCarSlotInfo(
          ref TCAR_INFO info,
          ref TSLOT_INFO sinfo );
```

（2）引数

info

スロット情報を設定するキャリア情報構造体のポインタです。

sinfo

設定したいスロット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	正常に設定できた。
(-1)	予約されているｶｳﾝﾄ数を超えていたので設定できなかった。

（4）説明

先にDshInitCarInfo()で初期設定されたキャリア情報構造体のslot_list上に１個のスロット情報を加えます。

slot_countで予約された数のslot_list上には本関数が実行される順にスロット情報を追加していきます。

slot_count以内の追加であれば設定後 0 を返却します。

もし、slot_counte数を超える場合は、設定しないで(-1)を返却します。

3．14．3．6　DshPutCarSlotInfoCopy() – キャリア情報にスロット情報をコピーして設定

（1）呼出書式

[c,C++]

```
API int APIX DshPutCarSlotInfoCopy(
        TCAR_INFO    *info,              // ｷｬﾘｱ情報構造体のﾎﾟｲﾝﾀ
        TSLOT_INFO   *sinfo              // ｽﾛｯﾄ情報構造体のﾎﾟｲﾝﾀ
);
```

[.NET VB]

```
Function DshPutCarSlotInfoCopy (
          ByRef info As dsh_info.TCAR_INFO,
          ByRef sinfo As dsh_info.TSLOT_INFO) As Int32
```

[.NET C#]

```
int DshPutCarSlotInfoCopy(
          ref TCAR_INFO info,
          ref TSLOT_INFO sinfo);
```

（2）引数

info

スロット情報を設定するキャリア情報構造体のポインタです。

sinfo

設定したいスロット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	正常に設定できた。
(-1)	予約されているｶｳﾝﾄ数を超えていたので設定できなかった。

（4）説明

先に DshInitCarInfo()で初期設定されたキャリア情報構造体の slot_list 上に１個のスロット情報を加えます。

本関数は、sinfo の情報を別メモリにコピーした上で追加します。

（3.14.4.13 の DshPutCarSlotInfo()の場合は sinfo の情報をそのまま追加します。

slot_count で予約された数の slot_list 上には本関数が実行される順にスロット情報を追加していきます。

slot_count 以内の追加であれば設定後 0 を返却します。

もし、slot_counte 数を超える場合は、設定しないで(-1)を返却します。

3．14．3．7　DshInitCarContent() - キャリアコンテントマップ情報の初期設定

（1）呼出書式

[c,C++]

```
API void APIX DshInitCarContent(
      TCACT_CONTENT   *info,            // ｷｬﾘｱ情報構造体のﾎﾟｲﾝﾀ
      int    count                      // ｺﾝﾃﾝﾄﾏｯﾌﾟ(lotid, substid)数
);
```

[.NET VB]

```
Sub DshInitCarContent (
        ByRef info As dsh_info.TCACT_CONTENT,
        ByVal count As Int32)
```

[.NET C#]

```
void DshInitCarContent(
        ref TCACT_CONTENT info,
        int count);
```

（2）引数

info

キャリアコンテントマップ情報を格納する構造体のポインタです。

count

格納するコンテントマップ数です。(後で設定する lotid,構造体情報の数)

（3）戻り値

なし。

（4）説明

info で指定される TCACT_CONTENT 構造体を初期設定します。

info 内に格納するロット ID と基板 ID の数を count で指定します。本関数は、info の lotid, substid メンバーに count 数分のポインタ配列領域を設けます。

TCACT_INFO 構造体内に対するロット、基板情報の設定は DshPutCarContent()関数を使用してください。

本 ContentMap 情報は S3F17 メッセージの"ContentMap"属性に関連する情報です。

3．14．3．8　DshPutCarContent() - キャリアコンテントマップ情報の設定

（1）呼出書式

[c,C++]

```
API int APIX DshPutCarContent(
        TCACT_CONTENT   *info,          // ｷｬﾘｱ情報構造体のﾎﾟｲﾝﾀ
        int    order,                   // 設定情報の順番(0,1,2....)
        char   *lotid,                  // ﾛｯﾄ ID(文字列)の格納ﾎﾟｲﾝﾀ
        char   *substid                 // 基板 ID(文字列)の格納ﾎﾟｲﾝﾀ
);
```

[.NET VB]

```
Function DshPutCarContent (
        ByRef info As dsh_info.TCACT_CONTENT,
        ByVal order As Int32,
        ByVal mid As String,
        ByVal substid As String) As Int32
```

[.NET C#]

```
int DshPutCarContent(
        ref TCACT_CONTENT info,
        int order,
        byte[] lotid,
        byte[] substid);
```

（2）引数

info

キャリアコンテントマップ情報を格納する構造体のポインタです。

order

情報を設定する配列の順位(0,1,2,...)です。

lotid

設定したいロット ID（文字列）です。

substid

設定したい基板 ID（文字列）です。

（3）戻り値

戻り値	意味
0	正常に設定できた。
(-1)	予約されているｶｳﾝﾄ数を超えていたので設定できなかった。

（4）説明

info で指定される TCACT_CONTENT 構造体内の lotid[], substid[]配列の order 番目の配列にロット ID,基板 ID を設定します。

lotid、substid 何れも NULL の場合は設定しません。

指定配列順位 order が info 内の count 以上の値の場合は設定しないで(-1)を返却します。