

Interface programmatique de l'interface client de VIDÉOSMS

Version 1.2-18

Order Number: L-X044-XHM-FR-E

Mars 2005

Ce document décrit l'interface programmatique du logiciel client VIDÉOSMS, permettant de développer facilement et rapidement des applications SMS+ à valeur ajoutée ainsi que d'intégrer l'expédition de SMS dans les applications informatiques, sous différentes plates-formes.

Mars 2005

Copyright ©Mars 2005

Copyright © 2003-2005 MONACO TÉLÉMATIQUE MC-TEL

Ce manuel et les informations qu'il contient sont la propriété de MC-TEL et sont strictement confidentiels. Ce guide est mis à la disposition des clients de MC-TEL afin de leur permettre d'interfaçer leurs applications avec la passerelle SMS VidéoSMS/Gateway.

Il leur est strictement interdit de reproduire ce manuel, ni de le prêter, de le fournir, le vendre ou d'une façon générale le mettre à disposition d'autres personnes ou organisations. Ils sont responsables de la bonne application de cette règle et doivent veiller à son respect par leurs employés.

MCTEL prend toutes les mesures nécessaires afin que les informations contenues dans ce manuel soient correctes, mais n'assume aucune responsabilité en cas d'erreur ou d'omission.

MCTEL fait continuellement évoluer ses produits et ses services et l'information contenue dans ce manuel peut ne pas être à jour. Il est donc recommandé de vérifier qu'une nouvelle version de ce manuel n'est pas disponible sur les sites www.smsfax.com et www.mctel.fr.

Contents

PREFACE	v
---------	---

CHAPTER 1 INTRODUCTION	1-1
-------------------------------	------------

1.1 INTÉGRATION SMS DANS LES APPLICATIONS INFORMATIQUES	1-1
1.2 L'INTERFACE PROGRAMMATIQUE CLIENT DE VIDEOSMS	1-1

CHAPTER 2 INTERFACE PROGRAMMATIQUE	2-1
-------------------------------------------	------------

2.1 INTRODUCTION	2-1
2.2 DISTRIBUTION	2-1
2.2.1 Environnement Windows	2-1
2.2.2 Environnement Unix	2-2
2.2.3 Environnement OpenVMS	2-2
2.3 DÉFINITIONS ET LISTES DES ROUTINES	2-2
2.3.1 Fonctions de gestion	2-2
2.3.2 Fonctions de communication par SMS	2-3
2.4 PROCÉDURE DE LINK	2-4
2.5 AVERTISSEMENT	2-4

CHAPTER 3 STRUCTURES SMS	3-1
---------------------------------	------------

3.1 SMSMESSAGE	3-1
3.2 SMSSESSION	3-1

Contents

3.3	SMSREPORT	3-2
3.4	STRUCTURE SMSUSERLOCATION	3-2
3.5	WAPDATA	3-3
3.6	STRUCTURE SMSDATETIMETYPE	3-4
3.7	TYPE DE MESSAGES SMS (SMSMESSAGETYPE)	3-4
3.8	FORMAT DU CONTENU DU MESSAGE SMS	3-5
3.9	SMS ENCODING CONSTANTS	3-5
3.10	STATUS DE RETOUR ET ERREURS SMS	3-6
3.11	STATUTS DES ESSAIS DE TRANSMISSIONS	3-7
3.12	FORMAT TEMPS SMPP	3-8
<hr/>		
CHAPTER 4	ROUTINES DE GESTION DE LA COMMUNICATION	4-1
	SMS_CONNECTGATEWAY	4-2
	SMS_LOGINGATEWAY	4-4
	SMS_LOGOUTGATEWAY	4-7
	SMS_ENQUIRELINK	4-8
	SMS_DISCONNECTGATEWAY	4-9
<hr/>		
CHAPTER 5	ROUTINES DE VIDEOSMS HORS SESSION SMS+	5-1
	SMS_SEND	5-2
	SMS_WAPPUSH	5-7
	SMS_READSTATUS	5-12
	SMS_GETMESSAGE	5-15
	SMS_GETLOCATION	5-17

CHAPTER 6	ROUTINES DE GESTION DE SESSION SMS+		6-1
	SMS_READ	6-2	
	SMS_GETSESSIONINFO	6-4	
	SMS_GETSESSIONDATA	6-6	
	SMS_UPDATESESSIONDATA	6-8	
	SMS_REPLY	6-10	

CHAPTER 7	ROUTINES DE GESTION ET DE SUPERVISION		7-1
	SMS_GETPROCESSEDMESSAGE	7-2	
	SMS_GETACCOUNTRECORD	7-5	

CHAPTER 8	FICHER D'INCLUDE SMSAPI1.H		8-1
------------------	-----------------------------------	--	------------

CHAPTER 9	EXEMPLES		9-1
9.1	PROGRAMME D'ENVOI VISUAL BASIC		9-1

INDEX

EXEMPLES

3-1	Structure SMSMessage	3-1
3-2	Structure SMSSession	3-2
3-3	Structure SMSReport	3-2
3-4	Structure SMSUserLocation	3-3
3-5	Structure WapData	3-4
3-6	Structure SmsDateTimeType	3-4
8-1	Fichier d'include smsApi1.h	8-1
9-1	Programme d'envoi de SMS VB	9-1

Preface

Public

Ce manuel est destiné aux programmeurs qui désirent utiliser les fonctions de l'interface programmatique de VIDÉOSMS.

Structure de ce document

Ce guide est divisé en trois chapitres.

Autres documents sur VIDÉOSMS et les SMS

Les autres manuels de la documentation VIDÉOSMS sont:

- *Guide d'installation et de configuration de VIDÉOSMS sous OpenVMS.*
- *Manuel de gestion et de supervision de VIDÉOSMS sous OpenVMS.*
- *Guide du développement d'applications clientes SMS+*
- *Utilisation de la commande DCL SMS*
- *Comment créer un service SMS+*

Les aspects techniques relatifs aux SMS, la norme GSM et diverses autres informations sont décrites dans des documents normatifs qu'il n'est pas nécessaire de connaître pour utiliser VIDÉOSMS. Pour information, ces documents sont les suivants:

- les documents internes à demander aux différents opérateurs de téléphonie mobile (Orange, Bouygues Télécom, SFR, etc...) (certains de ces documents sont confidentiels).
- Recommandation ETSI ETS 300 133-3: «*Electromagnetic compatibility and Radio Spectrum Matters (ERM); Enhanced Radio MESSAGE System (ERMES); Part 3: Network Aspects*».
- Recommandation ETSI GSM 03.40: «*Digital cellular telecommunications system (Phase 2+) Technical realization of the Short Message Service (SMS) Point-to-Point (PP)*».
- Recommandation ETSI GSM 07.05: «*Digital cellular telecommunications system (Phase 2+) Use of Data Terminal Equipment - Data Circuit terminating; Equipment (DTE-DCE) interface for Short Message Service (SMS) and Cell Broadcast Service (CBS)*».
- Recommandation ETSI ES 202 060-1: «*Short Message Service (SMS) for fixed networks: Network Based Solution (NBS); Part 1: Overview*».
- Recommandation ETSI TR 101 633: «*Digital cellular telecommunications system (Phase 2+): Support of Videotex*».
- Recommandation ITU E.132: «*Normalisation de certains éléments des procédures de commande de services téléphoniques supplémentaires*».

Équivalences et abréviations

Dans ce document, les noms suivants sont équivalents:

- SMS: short message service, minimessage
- SMS-MO (Mobile Originated): SMS envoyé par un utilisateur de mobile.
- SMS-MT (Mobile Terminated): SMS destiné à un utilisateur de mobile.
- téléfax: télécopie.
- fax: télécopie.

Marques

VIDEOSMS, SMSFAX, VIDÉONET, VIDÉONET/DEV, VIDÉOMAIL, VIDÉOCOM, VIDÉOMASK, VIDÉOTÉLEX, VIDÉOTÉLÉFAX, VIDÉOTÉLÉTEX, VIDÉOSCREEN, VIDÉOPA, VIDÉOSEC, TÉLEC, WEBACCESS, VIDÉOVALID, VIDÉOBASE, VIDÉOPAD, MONACO TÉLÉMATIQUE MC-TEL, le sigle de MC-TEL sont des marques déposées de MONACO TÉLÉMATIQUE MC-TEL.

Remarques et suggestions

Toutes remarques et suggestions concernant le logiciel VIDÉOSMS et les autres logiciels multimédia et de communication de Monaco Télématique MC-TEL doivent être adressés à:

MONACO TÉLÉMATIQUE MC-TEL,
B.P. 225,
98004 MONTE-CARLO CEDEX
MONACO

Téléphone: (+377)9216.8888

Télex: MC 489.288

Télécopie: (+377)9216.8865

Minitel: 3617 MCTEL1

Web: <http://www.mctel.fr> et <http://www.smsfax.com>

Email (support): sms@mctel.fr

Email (sales): sales@mctel.fr

Toute commande additionnelle de documentation doit être adressée à Monaco Télématique MC-TEL.

Les produits et services de Monaco Télématique MC-TEL comprennent:

- ensemble complet de solutions SMS pour les sociétés et les opérateurs télécoms: serveurs SMS, atelier de développement, logiciels clients, logiciels applicatifs standard, SMS-Centers.
- logiciels serveurs multimédia (Web et Wap, vidéotex, SMS+, vocal/fax, etc.) sur Unix et OpenVMS.
- intégration télécopie, télex, SMS, sur UNIX et OpenVMS.
- solution de sécurité sur Internet (WebAccess Advanced Firewall).

- réalisation et hébergement de services plurimédia (Web et Wap, vidéotex, SMS+, vocal/fax).
- réseaux et points d'accès multimédia internationaux avec fonction Kiosque.
- passerelles entre réseaux de communication hétérogènes (SMS, Web, Wap, vidéotex, téléfax, télex, X.400, RNIS, Alphapage).

1

Introduction

1.1 Intégration SMS dans les applications informatiques

VIDEOSMS offre tous les moyens permettant d'intégrer la gestion des SMS dans les applications informatiques.

Cette intégration peut concerner l'expédition automatique de SMS, mais elle peut surtout s'appliquer à la gestion de SMS entrants (requêtes) pour réaliser des applications SMS+ à forte valeur ajoutée, qui permettent de créer des services kiosques SMS+ avec paiement à l'acte et rémunération des fournisseurs de services par les opérateurs télécom mobiles (en France Orange, SFR et Bouygues Télécom).

1.2 L'interface programmatique client de VIDEOSMS

Les SMS sont transmis aux portables destinataires par l'intermédiaire d'une passerelle de gestion VIDEOSMS, elle-même raccordée aux différents opérateurs télécom.

La mise en place de cette passerelle nécessite une infrastructure lourde et coûteuse et des liaisons spécifiques (X.25, TCP/IP, frame relay, VPN, etc...) avec les différents opérateurs.

En conséquence, de nombreux sites trouveront plus rapide et commode d'utiliser la passerelle européenne de facilitation VIDEOSMS de Monaco Télématique MCTEL comme facilitateur pour accéder au réseau. L'interface programmatique cliente VIDEOSMS/Api/Client est alors fournie au développeur, qui pourra l'utiliser pour intégrer de façon transparente dans son application la gestion des SMS.

Cette API VIDEOSMS/Client API est fournie gratuitement aux développeurs. Ceux-ci disposent également d'autres outils:

- envoi de SMS au travers de scripts cgi appelés sur le Web.
- envoi de SMS par téléchargement ftp de fichiers XML.

Les applications ainsi développées en client/serveur pourront à tout moment être utilisées sur un serveur VIDEOSMS fourni à la société cliente par Monaco Télématique MCTEL, sans modification ni recompilation.

2

Interface Programmatique

2.1 Introduction

Les fonctions de gestion de SMS peuvent aisément être interfacées avec n'importe quel applicatif écrit en langage compilé, permettant l'émission automatique de messages SMS et la réalisation d'applications client/serveur SMS+ à forte valeur ajoutée.

Un ensemble de routines permettent d'avoir accès aux diverses fonctionnalités de VIDÉOSMS à partir de nombreuses plates-formes:

- Unix:
 - ◊ UNIX-SCO
 - ◊ Linux
 - ◊ Sun Solaris
 - ◊ Compaq/Digital Unix (Alpha)
 - ◊ HP-UX
 - ◊ IBM AIX
- Windows
- OpenVMS
- PalmOS

Ces routines sont documentées de façon extensive dans le chapitre suivant, auquel les programmeurs sont priés de se reporter. Une brève liste des routines est donnée ci-dessous afin de permettre de visualiser rapidement les principales fonctions à la disposition des programmeurs.

2.2 Distribution

Les routines sont livrées sur CD-ROM ou téléchargeables sur nos sites: www.mctel.fr et www.smsfax.com

2.2.1 Environnement Windows

Sous Windows, l'API VIDEOSMS/Client comprend les fichiers suivants:

- la DLL *smsApi.dll*
- le fichier de définitions C/C++ *smsApi.h*
- le fichier de link C/C++ *smsApi.lib*
- fichier de définitions de type pour Visual Basic *smsApi.tlb*

- des fichiers d'exemples (DemoSMSAPI.vbp, DemoSMSAPI.vbw, Form1.frm, DemoSMSAPI.exe)

2.2.2 Environnement Unix

Sous les plates-formes UNIX, l'API VIDEOSMS/Client comprend les fichiers suivants:

- la librairie smsApi.o
- le fichier de définitions *smsApi.h*
- des fichiers d'exemples.

2.2.3 Environnement OpenVMS

Sous OpenVMS, la solution est disponible sous tous les environnements matériels:

- VAX.
- Alpha.
- Itanium.

Elle comprend:

- la librairie VIDEOSMSCLIENT.OLB
- des programmes d'exemples: TEST_SMSCLIENT.EXE, TEST_SMSCLIENT.BAS
- la procédure de link LINK_VIDEOSMSCLIENT.COM

2.3 Définitions et listes des routines

Les définitions de certaines constantes (messages d'erreurs, etc...) se trouvent dans les fichiers .h.

Des fichiers de link sont fournis.

2.3.1 Fonctions de gestion

Ces fonctions sont utilisées pour établir et gérer la communication avec le serveur VIDEOSMS distant:

- SMS_ConnectGateway: appel du serveur SMS distant.
- SMS_LoginGateway: identification sur le serveur SMS distant.
- SMS_LogoutGateway: logout du serveur SMS distant.
- SMS_DisconnectGateway: déconnexion de la passerelle SMS distante.
- SMS_EnquireLink: vérification de l'état du lien avec la passerelle SMS distante.

Il est à noter que la passerelle SMS distante déconnecte la liaison TCP/IP en cas d'inactivité excédant un timeout déterminé (configuré lors de la création du compte client, par défaut de 1 minute). Il est de la responsabilité de l'application cliente d'émettre des requêtes *SMS_EnquireLink* régulières en cas d'inactivité approchant du timeout pour éviter une déconnexion par la passerelle distante.

2.3.2 Fonctions de communication par SMS

Il existe deux types de routines gérant la communication par SMS avec les mobiles, qui peuvent être utilisées conjointement:

- les routines autonomes, utilisables en-dehors d'une session SMS+ initiée par un utilisateur SMS, par exemple pour envoyer des messages SMS-MT personnalisés en nombre:
 - ◇ *SMS_Send*: expédie un message SMS-MT non sollicité (push SMS) au destinataire spécifié. Cette routine ne doit pas être utilisée pour répondre à un SMS-MO de requête provenant d'un mobile lors d'une session SMS+, pour laquelle on répond avec la fonction *SMS_Reply*.
 - ◇ *SMS_PushWap*: envoie une URL Wap par SMS au destinataire, par exemple pour transmettre un contenu multimédia (logo, sonnerie, exécutable) par Wap, ou pour permettre à un mobile de se connecter automatiquement à un service Wap.
 - ◇ *SMS_GetMessage*: réception en temps réel sur un canal de communication ouvert de SMS-MO entrants ou des notifications d'expédition des SMS-MT précédemment soumis à la passerelle.
 - ◇ *SMS_ReadStatus*: recherche l'état d'un SMS dont l'expédition a été demandée.
 - ◇ *SMS_GetMessage*: retourne un SMS-MO reçu ou un accusé de réception d'un SMS-MT précédemment expédié.
 - ◇ *SMS_GetLocation*: effectue une demande de géolocalisation du numéro de mobile spécifié.
- les routines de gestion de la session avec l'utilisateur SMS+ distant, qui ne sont utilisables qu'en mode client/serveur, en réponse à une requête effectuée par un utilisateur distant et transmise par SMS au serveur:
 - ◇ *SMS_Read*: récupère un SMS-MO destiné à l'application, ainsi que les données fixes de la session: numéro de l'appelant, contenu du message SMS-MO destiné au serveur, etc.
 - ◇ *SMS_GetSessionInfo*: lecture des informations de la session courante.
 - ◇ *SMS_GetSessionData*: récupération d'informations spécifiques à cette session (échanges de SMS-MO et SMS-MT) précédemment sauvegardées par un appel à *SMS_GetSessionData* lors d'une requête précédente de l'utilisateur.

- ◇ *SMS_SaveSessionData*: sauvegarde d'informations applicatives spécifiques de cette session, qui pourront être récupérées ultérieurement lors de l'échange suivant de SMS avec cet utilisateur.
- ◇ *SMS_Reply*: renvoi à l'utilisateur d'un message SMS en réponse à sa demande, fermeture ou non de la session. Il reste possible d'envoyer des messages SMS non sollicités (push SMS) à cet utilisateur au moyen de la routine *SMS_Send*, mais les informations de session ne seront plus accessibles.

2.4 Procédure de link

Elle varie selon les plates-formes. Se reporter aux exemples livrés avec la distribution.

2.5 Avertissement

L'utilisation des routines de l'API VIDÉOSMS/Client peut permettre l'envoi de SMS en nombre et en conséquence générer des coûts de communication importants, à la charge de la société émettrice.

En conséquence, il est donc essentiel de ne donner accès à ces routines, aux systèmes sur lesquelles elles sont installées et aux applications SMS générées qu'aux utilisateurs possédant une habilitation suffisante et de mettre en place les procédures (firewall, etc...) permettant d'éviter un accès illégal au système et aux fonctions SMS par des personnes non autorisées.

Il est notamment recommandé d'utiliser lors de l'accès à la plateforme de facilitation SMS internationale de MCTEL une adresse TCP/IP fixe et de transmettre à MCTEL cette adresse afin que nos firewalls soient configurés pour autoriser uniquement cette adresse à accéder au compte SMS de la société.

3

Structures SMS

3.1 SMSMessage

La structure SMSMessage définit la structure d'un SMS émis ou reçu.

Example 3-1 Structure SMSMessage

```
typedef struct
{
    SMSMessageType messagetype;
    Char          msg_reference_number[60];
    Char   called_number[20];
    Char   caller_number[20];
    Char   alternate_number[20];
    UInt16 destPort;
    UInt16 srcPort;
    UInt32 msg_length;
    UInt32 msg_format;
    UInt32 msg_class;
    UInt32 msg_encoding;
    UInt32 msg_action;
    UInt32 msg_gsmflags;
    UInt32 msg_options;
    UInt32 country;
    UInt32 carrier;
    UInt32 sms_state;
    UInt32 sms_error;
    UInt32 credit_cost;
    SMSDateTime posted_date;
    char   transmitted_date[18];
    SMSDateTime deferred_date;
    SMSDateTime expiration_date;
    Char   sender_address[50];
    Char   msg_data[500];
} SMSMessage
```

3.2 SMSSession

La structure SMSSession stocke des informations de session.

Example 3–2 Structure SMSSession

```
typedef struct
{
    char    sessionId[20];
    Char    called_number[20];
    Char    caller_number[20];
    char    client_name[20];
    char    application_name[50];
    UInt32  taxationRate;
    UInt32  current_exchange;
    UInt32  total_exchange;
    SMSDateTimeType Openeddatetime;
    SMSDateTimeType LastExchangedatetime;
    char    latitude[8];
    char    longitude[9];
    char    price[8];
} SMSSession
```

Example 3–3 Structure SMSReport

```
typedef struct
{
    UInt32  report_type;
    UInt32  msg_reference_number;
    Char    called_number[20];
    Char    messageID[50];
    SmsDateTimeType timeStamp;
} SMSReport
```

3.3 SMSReport

La structure SMSReport stocke les informations sur les rapports de transmission: accusé d'envoi correct, erreur de transmission, etc.

3.4 Structure SMSUserLocation

Cette structure stocke les informations de géopositionnement de l'utilisateur.

La variable *type* peut être:

- 0: non applicable (pas de localisation effectuée).
- 1: position actuelle.
- 2: positioncurrent or last known location;
- 3: positionnement initial pour un appel à un service d'urgence.
- 4: demande de positionnement différé.

La forme de la localisation de l'utilisateur est stockée dans *shape*:

- 0: non applicable (pas de localisation effectuée).

Example 3–4 Structure SMSUserLocation

```
typedef struct
{
    SmsDateTimeType age; /* age of location estimate */
    Uint32 type; /* location type */
    Uint32 shape; /* location shape */
    Uint32 cellid; /* Global Cell Identifier the user is attached to */
    char latitude[8]; /* latitude */
    char longitude[9]; /* longitude */
    Uint32 privacycheck; /* LCS Privacy check */
    Uint32 Qos; /* requested quality of service */
    Uint32 priority; /* priority of the request */
    Char NA-ESRD; /* NA Emergency Services Routing Digits (USA) */
    Char NA-ESRK; /* NA Emergency Services Routing Key (USA) */
} SMSUserLocation
```

- 1: point ellipsoïde avec un cercle d'incertitude.
- 2: point ellipsoïde avec une ellipse d'incertitude.
- 3: point ellipsoïde avec une altitude et une ellipse d'incertitude.
- 4: arc ellipsoïde.
- 5: point ellipsoïde.

Les paramètres NA-ESRD et NA-ESRK ne sont utilisés qu'aux Etats-Unis.

Le paramètre *privacycheck* renseigne le type d'action liée à la demande de positionnement:

- 1: positionnement autorisé sans notification de l'utilisateur.
- 2: positionnement autorisé avec notification de l'utilisateur.
- 3: la demande de positionnement exige la notification de l'utilisateur et son acceptation, le positionnement est retourné seulement si l'utilisateur l'accepte ou s'il ne répond pas à la notification.
- 4: la demande de positionnement exige la notification de l'utilisateur et son acceptation, le positionnement est retourné seulement si l'utilisateur l'accepte.
- 5: demande de positionnement géographique non autorisé.

3.5 WapData

La structure WapData stocke des données Wap, notamment utilisées pour des opérations de Wap Push avec la fonction *SMS_WapPush*.

Example 3–5 Structure WapData

```
typedef struct
{
    UInt16  destPort;
    UInt16  srcPort;
    UInt32  wap_length;
    UInt32  wap_action;
    UInt32  wap_encoding;
    UInt32  wap_options;
    SMSDateTime  created_date;
    SMSDateTime  expiration_date;
    Char  wap_url[100];
    Char  wap_text[50];
    Char  MSISDN[20];
    Char  wap_content[500];
} SMSMessage
```

Example 3–6 Structure SmsDateTimeType

```
{
typedef
    UInt32  dateTime;
} SMSDateTimeType
```

3.6 Structure SMSDateTimeType

Elle stocke les dates et heures utilisés pour les horodatages.

3.7 Type de messages SMS (SMSMessageType)

Constante:	Val	Signification
SMSMessageTypeReceived	0	Message SMS-MO reçu
SMSMessageTypeReport	1	Message de notification SMS
SMSMessageTypeSubmitted	2	Message SMS-MT non sollicité (push SMS-MT) à transmettre au mobile spécifié
SMSMessageTypeDelivered	3	A SMS-MT bien délivré au destinataire distant
SMSMessageTypeReply	4	Message SMS-MT envoyé en réponse à une requête préalablement reçue par un SMS-MO.
FAXMessageTypeReceived	10	Message fax reçu
FAXMessageTypeReport	11	Message de notification fax
FAXMessageTypeSubmitted	12	Message d'expédition d'un fax
VoiceMessageTypeReport	21	Message de notification concernant un message vocal
VoiceMessageTypeSubmitted	22	Message d'expédition d'un message vocal vocalisé en Text To Speech

3.8 Format du contenu du message SMS

Constante:	Val	Signification
SMSFormatAlphabetic	3	Données alphanumériques encodées selon l'encodage spécifié
SMSFormatNumeric	2	Données uniquement numériques (chiffres de 0 à 9)
SMSFormatBinary	4	Données binaires (de 00h à FFh)
SMSFormatHexa	5	Données binaires encodées en hexadécimal (2 caractères hexadécimaux par octet). Dans ce cas <i>msg_length</i> encode le nombre de caractères hexadécimaux, le nombre d'octets binaires du message étant deux fois plus faible

3.9 SMS encoding constants

Les constantes d'encodage (DCS data coding scheme) décrivent le type d'encodage des données SMS à envoyer ou reçues.

Constante:	Val	Signification
SMSEncodingIA5	1	Encodage CCITT T.50/ISO 8859-1 (défaut)
SMSEncodingGSM	2	Encodage GSM

Constante:	Val	Signification
SMSEncodingUnicode	3	Encodage Unicode

3.10 Status de retour et erreurs SMS

Constante:	Val	Signification
SMSNormal	1	Exécution correcte de la fonction
SMSError	2	Erreur sans précision
SMSNoVideosms	4	VIDEOSMS non installé sur ce système
SMSInvalidParameter	6	Un ou plusieurs paramètres spécifiés sont invalides
SMSNoNetwork	8	Pas de liaison TCP/IP ou X.25 utilisable pour accéder à la passerelle distante
SMSCannotConnect	10	Connexion avec la passerelle SMS distante impossible
SMSNoAnswer	12	Pas de réponse de la passerelle distante jusqu'au timeout de connexion
SMSInvalidIPAddress	14	Adresse IP appelante non autorisée à utiliser la passerelle SMS pour le compte spécifié
SMSInvalidLoginInfo	16	Username et/ou mot de passe de la société invalides
SMSInvalidMessageId	18	Format ou numéro du message spécifié invalide
SMSNoMessage	20	Pas de message en attente pour cette société/application
SMSLinkLost	22	Liaison TCP/IP ou X.25 avec la passerelle SMS distante interrompue
SMSNoCarrier	24	Pas d'opérateur réseau SMS pour transmettre le message à ce numéro ou dans ce pays
SMSMissingParameter	26	Un paramètre obligatoire ou un champ obligatoire d'une structure est manquant
SMSNoPrivilege	28	Les droits d'accès de l'utilisateur ne permettent pas d'effectuer cette opération
SMSInvalidUserid	30	Userid invalide ou non retrouvé
SMSCompanyLocked	32	Le compte de la société est bloqué
SMSUserIdLocked	34	Le compte de cet utilisateur est bloqué
SMSAccountNoCredit	36	Le compte de cet utilisateur est en prépaiement et n'a plus de crédit

Constante:	Val	Signification
SMSObsoleteVersion	38	La version du logiciel VIDEOSMS/Client est trop ancienne, installer une version plus récente
SMSAccountAlreadyExist	40	Ce compte utilisateur existe déjà
SMSNotIdentifier	42	L'utilisateur n'est pas authentifié
SMSAccountNotExist	44	Le compte de l'utilisateur n'existe pas
SMSInvalidDate	46	La date spécifiée est invalide
SMSFileNotFound	48	Fichier spécifié non trouvé
SMSUpdateCfgFilees	50	Mettre à jour les fichiers de configuration
SMSMobileAlreadyExist	52	Ce numéro de mobile est déjà utilisé par un autre compte
SMSTooManyResendPwd	54	Trop de renvois de mots de passe ont été effectués, renvoi interdit
SMSExpiredPromoCode	56	Le code promotionnel spécifié a expiré, la création d'un compte avec ce code promotionnel est interdit
SMSVoucherInvalid	58	Le numéro de voucher spécifié est invalide
SMSVoucherAlreadySpent	60	Ce voucher a déjà été dépensé
SMSNotAvailableInCountry	62	La fonction demandée (ex: création d'un compte) n'est pas offerte dans le pays du demandeur
SMSOpenPayingAccount	64	L'ouverture gratuite d'un compte n'est pas possible dans le pays du demandeur, il doit acheter des unités pour ouvrir son compte

3.11 Statuts des essais de transmissions

La variable *error* de la structure *SMSMessage* stocke le code d'erreur détaillé, qui peut être:

Constante:	Val	Signification
SMSUnknownSubscriber	51	Ce numéro de mobile est inconnu ou non attribué
SMSMobileUnreacheable	54	Le mobile est éteint ou hors couverture
SMSNumberPorted	58	Ce numéro de mobile est porté par un autre opérateur
SMSUnknownError	99	Erreur non précisée

D'autres erreurs spécifiques à certains réseaux peuvent également être retournées (codes supérieurs à 100).

Le code d'erreur n'est pas remis à zéro après la transmission correcte du SMS lors d'un essai suivant par l'opérateur, permettant ainsi de connaître a posteriori la cause d'une erreur transitoire (par exemple mobile éteint ou hors couverture).

3.12 Format temps SMPP

Le format temps SMPP est une chaîne de 16 caractères au format "YYMMDDhhmmssstnp" où:

- YY: deux derniers digits de l'année (99 à 99)
- MM: mois (01 à 12)
- DD: jour (01 à 31)
- hh: heure (00 à 23)
- mm: minutes (00 à 59)
- ss: secondes (00 à 59)
- t: dixièmes de secondes (0 à 9)
- nn: différence en 1/4 heure entre le temps local et l'Universal Time Constant (00 à 48).
- p: + ou - selon que le temps local est avancé (+) ou retardé (-) par rapport à l'UTC.

4 **Routines de gestion de la communication**

SMS_ConnectGateway—Connexion à la passerelle distante VIDEOSMS

Cette routine permet de se connecter à la passerelle distante VIDEOSMS

FORMAT **SMS_ConnectGateway** *UInt32 **ConnectId, char *GatewayAddress, UInt32 I_GatewayAddress, UInt32 GatewayPort*

RETURNS VMS Usage: **cond_values**
 type: **entier long**
 access: **lecture seule**
 mechanism: **par valeur**

ARGUMENTS **ConnectId**
VMS Usage: **pointeur**
type: **pointeur sur pointeur**
access: **lecture/écriture**
mechanism: **par référence**
Identifiant unique de connexion, initialisé lors de la connexion, à transmettre lors de l'appel des autres fonctions.

***GatewayAddress**
VMS Usage: **pointeur**
type: **pointeur sur chaîne de caractères**
access: **lecture seule**
mechanism: **par référence**
Pointeur sur une chaîne de caractères contenant l'adresse TCP/IP du serveur VIDEOSMS à appeler. Cette adresse doit être une adresse numérique sous la forme nnn.nnn.nnn.nnn (sous OpenVMS, cette chaîne est passée par descripteur).

I_GatewayAddress
VMS Usage: **longword**
type: **entier long**
access: **lecture seule**
mechanism: **par référence**
longueur utile de *GatewayAddress*.

GatewayPort
VMS Usage: **longword**
type: **entier long**
access: **lecture seule**
mechanism: **par référence**

Routines de gestion de la communication

SMS_ConnectGateway

Numéro de port TCP/IP sur lequel se connecter.

DESCRIPTION

Cette routine permet de se connecter à la passerelle distante VIDEOSMS, en spécifiant son adresse TCP/IP et son numéro de port. Elle initialise également les informations de session et les structures internes permettant la communication avec la passerelle (il est donc indispensable d'utiliser cette fonction pour se connecter, un appel TCP/IP par les sockets n'est pas possible).

Une routine équivalente existe pour permettre de se connecter au travers du réseau X.25 (SMS_ConnectX25Gateway) et peut être transmise sur demande aux clients souhaitant utiliser le réseau X.25.

STATUT DE RETOUR

SMSNormal	l'application s'est correctement connectée à la passerelle SMS distante.
SMSError	une erreur s'est produite.
SMSNoVideosms	VIDEOSMS n'est pas installé sur ce système.
SMSInvalidParameter	un des paramètres spécifiés est invalide.
SMSNoNetwork	le réseau TCP/IP est down ou la machine n'est pas raccordée au réseau.
SMSCannotConnect	la passerelle VIDEOSMS n'est pas accessible à partir du réseau sur lequel est connecté la machine appelante. Vérifier la configuration du firewall du réseau si besoin.
SMSAlreadyConnected	l'application est déjà connectée à la passerelle VIDEOSMS et son profil n'autorise pas des connexions multiples.
SMSNoAnswer	la gateway VIDEOSMS n'a pas accepté l'appel TCP/IP dans le temps spécifié.
SMSInvalidIPAddress	l'adresse IP de la machine appelante ne fait pas partie des adresses autorisées à appeler la gateway VIDEOSMS.

EXAMPLES

***userId**

VMS Usage: **pointeur**
type: **pointeur sur chaîne de caractères**
access: **lecture seule**
mechanism: **par référence**

Pointeur sur une chaîne de caractères contenant l'identification de l'utilisateur ou de l'application gestionnaire des SMS (terminé par un caractère nul sous Unix ou Windows, passée par descripteur sous OpenVMS).

***userPassword**

VMS Usage: **pointeur**
type: **pointeur sur chaîne de caractères**
access: **lecture seule**
mechanism: **par référence**

Pointeur sur une chaîne de caractères contenant le mot de passe de l'utilisateur ou de l'application appelante (terminé par un caractère nul sous Unix et Windows, passée par descripteur sous OpenVMS).

***SystemId**

VMS Usage: **pointeur**
type: **pointeur sur chaîne de caractères**
access: **lecture seule**
mechanism: **par référence**

Pointeur sur une chaîne de caractères contenant l'identifiant (SystemId) du système appelant, système et application spécifique (terminé par un caractère nul sous Unix et Windows, passée par descripteur sous OpenVMS).

LoginOptFlags

VMS Usage: **entier**
type: **SMSLoginOpt**
access: **lecture seule**
mechanism: **par valeur**

Entier long permettant de passer des paramètres complémentaires. Le flag *SMSLoginOpt_ReceiveOnly* peut être positionné pour spécifier que l'application se connecte comme un receiver allant seulement recevoir des notifications d'envoi des SMS-MT envoyés par ailleurs ou des SMS-MO reçus.

NbrCredits

VMS Usage: **pointeur**
type: **pointeur sur entier long**
access: **écriture seule**
mechanism: **par référence**

Nombre de crédits sur le compte de l'utilisateur, si applicable (sinon 0).

DESCRIPTION

Cette routine permet à l'application de s'authentifier sur la passerelle distante VIDEOSMS, en spécifiant son identifiant de société et le mot de passe associé. Elle permet également de spécifier un identifiant d'utilisateur identifiant de façon unique l'usager ou l'application responsable du traitement des SMS subséquents.

Routines de gestion de la communication

SMS_LoginGateway

Lorsque le bit 0 de *Flags* est positionné, l'application se connectera uniquement comme un receiver. Dans ce cas, elle n'émettra pas de messages, mais se contentera de recevoir des SMS-MO destinés à l'application ou des accusés de réception des SMS-MT précédemment expédiés par ailleurs.

STATUT DE RETOUR

SMSNormal	l'application est correctement authentifiée sur la passerelle SMS distante.
SMSError	une erreur s'est produite.
SMSLinkLost	le lien avec la passerelle distante a été interrompu.
SMSNoVideosms	VIDÉOSMS n'est pas installé sur ce système.
SMSInvalidParameter	un des paramètres spécifiés est invalide.
SMSInvalidLoginInfo	le paramètre <i>companyId</i> ou le paramètre <i>password</i> est invalide, l'authentification est refusée.
SMSInvalidUserid	le paramètre <i>userid</i> spécifié est invalide.
SMSCompanyLocked	le compte de la société est bloqué.
SMSUserIdLocked	le compte de l'utilisateur ou de l'application spécifié est bloqué.
SMSAccountNoCredit	Ce compte n'a plus de crédit autorisé sur son compte.
SMSInvalidIPAddress	l'adresse IP de la machine appelante n'est pas autorisée pour ce compte.
SMSObsoleteVersion	la version de l'API client est obsolète et n'est plus supportée par le serveur, l'upgrade de l'API client est indispensable.

EXAMPLES

SMS_LogoutGateway—Identification sur la passerelle distante VIDEOSMS

Cette routine permet de clore la session sur la passerelle distante VIDEOSMS, sans mettre fin à la connexion.

FORMAT **SMS_LogoutGateway** *UInt32 *ConnectId*

RETURNS VMS Usage: **cond_values**
 type: **entier long**
 access: **lecture seule**
 mechanism: **par valeur**

ARGUMENTS **ConnectId**
 VMS Usage: **pointeur**
 type: **pointeur sur entier long**
 access: **lecture seule**
 mechanism: **par référence**
 Identifiant unique de connexion, initialisé par la fonction SMS_ConnectGateway

DESCRIPTION Cette routine permet à l'application de terminer la session sur la passerelle distante VIDEOSMS, sans mettre fin à la connexion. L'application peut ensuite soit se déconnecter avec la fonction *SMS_DisconnectGateway* ou se réidentifier sous un autre identifiant avec la fonction *SMS_LoginGateway*.

STATUT DE RETOUR

SMSNormal	la session applicative est correctement terminée, le lien reste établi avec la passerelle SMS distante.
SMSError	une erreur s'est produite.
SMSLinkLost	le lien avec la passerelle distante a été interrompu.
SMSNoVideosms	VIDÉOSMS n'est pas installé sur ce système.

EXAMPLES

SMS_EnquireLink—Vérification du statut de la liaison avec la passerelle distante VIDEOSMS

Cette routine permet de vérifier le bon fonctionnement du lien avec la passerelle distante VIDEOSMS, ainsi que d'éviter une déconnexion par timeout en cas d'inactivité.

FORMAT **SMS_EnquireLink** *UInt32 *ConnectId*

RETURNS VMS Usage: **cond_values**
 type: **entier long**
 access: **lecture seule**
 mechanism: **par valeur**

ARGUMENTS **ConnectId**
 VMS Usage: **pointeur**
 type: **pointeur sur entier long**
 access: **lecture seule**
 mechanism: **par référence**
 Identifiant unique de connection, initialisé par la fonction SMS_ConnectGateway

DESCRIPTION Cette routine permet de vérifier le bon fonctionnement du lien avec la passerelle distante VIDEOSMS, ainsi que d'éviter une déconnexion par timeout en cas d'inactivité.

STATUT DE RETOUR	SMSNormal	l'application est toujours en liaison avec la passerelle distante.
	SMSError	une erreur s'est produite.
	SMSLinkLost	le lien avec la passerelle distante a été interrompu.
	SMSNoVideosms	VIDEOSMS n'est pas installé sur ce système.

EXAMPLES

SMS_DisconnectGateway—Déconnexion de la passerelle distante VIDEOSMS

Cette routine se déconnecte de la passerelle distante VIDEOSMS

FORMAT **SMS_DisconnectGateway** *UInt32 *ConnectId*

RETURNS VMS Usage: **cond_values**
 type: **entier long**
 access: **lecture seule**
 mechanism: **par valeur**

ARGUMENTS **ConnectId**
 VMS Usage: **pointeur**
 type: **pointeur sur entier long**
 access: **lecture seule**
 mechanism: **par référence**
 Identifiant unique de connection, initialisé par la fonction SMS_ConnectGateway

DESCRIPTION Cette routine se déconnecte de la passerelle distante VIDEOSMS et désalloue les structures internes utilisées par VIDEOSMS.

Une routine équivalente existe pour se déconnecter lorsque la liaison a été établie au travers du réseau X.25 (SMS_DisconnectX25Gateway).

STATUT DE RETOUR

SMSNormal	l'application s'est déconnecté de la passerelle SMS distante.
SMSError	une erreur s'est produite.
SMSLinkLost	le lien avec la passerelle distante a été interrompu.
SMSNoVideosms	VIDÉOSMS n'est pas installé sur ce système.

EXAMPLES

5

Routines de VIDEOSMS hors session SMS+

Routines de VIDEOSMS hors session SMS+

SMS_Send

- ◇ des données alphanumériques (défaut).
- ◇ des données binaires à transmettre en mode transparent, soit sous forme de données binaires, soit sous forme de données hexadécimales.
- ◇ un nom de fichier contenant les données à expédier. Dans ce cas, le bit 0 de *msg_options* doit être positionné à 1.

En sortie, le champ *msg_reference_number* est rempli avec l'identifiant unique assigné par la passerelle au message SMS.

Il est possible de spécifier les informations suivantes optionnelles:

- *msg_encoding*: par défaut *SMSEncodingIA5*.
- *msg_class*: spécifie la classe du message:
 - ◇ 0: classe 0: affichage immédiat sans intervention de l'utilisateur, pas de stockage du SMS parmi les SMS reçus.
 - ◇ 1: classe 1: réception du SMS, l'utilisateur devant aller le consulter (défaut).
 - ◇ 2: classe 2: stockage dans carte SIM
 - ◇ 3: classe 3: spécifique de l'équipement.
- *msg_format*: format du message, si celui-ci est différent du défaut (le défaut est *SMSFormatAlphabetic*)
- *srcPort*: numéro de port émetteur du message, utilisé seulement pour les messages binaires NBS.
- *destPort*: numéro de port émetteur du message, utilisé seulement pour les messages binaires NBS.
- *alternate_number*: numéro de SMS secondaire, que la passerelle utilisera pour transmettre le message si le premier numéro spécifié n'est pas accessible et que le bit 4 de *msg_options* est positionné.
- *caller_number*: numéro SMS de l'appelant. Sur la plupart des réseaux, ce numéro ne peut être positionné, et le numéro SMS de l'appelant sera toujours le numéro court affecté à la passerelle par l'opérateur.
- *sender_address*: adresse email l'expéditeur du SMS. Si les bits 20 ou 21 de *msg_options* sont positionnés, un compte-rendu d'envoi ou d'erreur pourra être expédié à cette email lors de l'envoi du SMS ou d'une erreur durant la tentative d'expédition.
- *msg_options*: spécifications d'options. Les options sont spécifiées par le positionnement de bits du longword:
 - ◇ bit 0 mis: la chaîne *msg_data* ne contient pas les données à transmettre mais le nom du fichier contenant les données.
 - ◇ bit 1 mis: demande d'envoi du SMS à une date et heure donnée, spécifiée dans *deferredDate*.
 - ◇ bit 2 mis: demande d'envoi du SMS en priorité.
 - ◇ bit 3: date d'expiration de la validité du message spécifiée dans *expirationDate*.

Routines de VIDEOSMS hors session SMS+ SMS_Send

- ◇ bit 4: numéro de secours spécifié, à utiliser si le numéro principal ne peut être atteint.
- ◇ bit 5: demande de notification (pour tous les cas).
- ◇ bit 11: UDH spécifié par l'application et présent au début du champ *msg_data*, la longueur utile des données UDH est alors indiquée dans le champ *sms_error*.
- ◇ bit 16 mis: autorise le routage des messages en réponse vers l'application originatrice.
- ◇ bit 17 mis: le champ *application_name* est utilisé pour stocker des données spécifiques à l'application.
- ◇ bit 18 mis: l'application spécifie une référence interne (*private_reference*) qui sera stockée avec le message et retournée dans les notifications.
- ◇ bit 19 mis: spécification message EMS (ex: iMelody).
- ◇ bit 20 mis: demande d'envoi d'un message email de non-transmission à l'expéditeur en cas de non-transmission du SMS.
- ◇ bit 21 mis: demande d'envoi d'un message email de confirmation à l'expéditeur après la transmission correcte du message.
- ◇ bit 24 mis: le SMS devra être recyclé s'il n'a pu être expédié correctement: il restera donc en attente de recyclage dans la file d'attente.
- ◇ bit 26 mis: SMS envoyé en mode "replace", devant remplacer celui précédemment soumis pour le même destinataire s'il n'a pu être délivré.
- ◇ bit 29 mis: demande de notification HTTP/HTML sur le script spécifié dans la configuration du compte, ou dans le champ *notification_address*.
- ◇ bit 30 mis: le SMS est expédié en validation obligatoire, et restera dans la file d'attente jusqu'à ce qu'un responsable le valide pour expédition.

***private_ref**

VMS Usage: **pointeur**

type: **pointeur sur chaîne de 0 à 15 caractères**

access: **lecture seule**

mechanism: **par référence**

Pointeur sur une chaîne de 15 caractères maximum contenant un identifiant alphanumérique privé du message, spécifique de l'application envoyant le SMS. N'est utilisé que lorsque le bit 18 de *msg_options* est positionné.

***private_data**

VMS Usage: **pointeur**

type: **pointeur sur chaîne de 0 à 50 caractères**

access: **lecture seule**

mechanism: **par référence**

Pointeur sur une chaîne de caractères contenant des données spécifiques de l'application, à associer au message envoyé et à retourner dans les

notifications. N'est utilisé que lorsque le bit 17 de *msg_options* est positionné.

DESCRIPTION

Cette routine permet de demander l'expédition d'un SMS.

Le contenu du SMS à transmettre peut être passé à la routine ou stocké dans un fichier dont le nom complet est transmis à la routine. Le fichier doit être un fichier séquentiel.

Il est possible de demander l'envoi d'un SMS en priorité ou de demander l'envoi du SMS à une date et heure précise. Il est également possible de spécifier une date et heure d'expiration où le message sera détruit par l'opérateur s'il n'a pas été en mesure de le transmettre au mobile du destinataire.

STATUT DE RETOUR

SMSNormal	l'expédition du SMS a bien été demandée. Ce statut ne signifie pas que le SMS a été transmis ou reçu par le destinataire, mais simplement qu'il a bien été transmis à la passerelle VIDEOSMS distante et mis en queue pour l'expédition. La variable <i>msg_reference_number</i> contient la référence d'entrée du SMS dans la file d'envoi. Ce numéro, unique parmi tous les SMS gérés par le système, permet de consulter le statut du SMS.
SMSError	une erreur s'est produite.
SMSLinkLost	le lien avec la passerelle distante a été interrompu.
SMSNoVideosms	VIDÉOSMS n'est pas installé sur ce système.
SMSInvalidParameter	un des paramètres spécifiés est invalide.
SMSFileEmpty	le fichier à expédier est vide et le SMS ne sera pas transmis.
SMSFileNotFound	le fichier contenant le texte à envoyer n'a pas été trouvé. Le SMS ne sera pas transmis.
SMSTooLong	la taille du message excède la taille maximale autorisée par la passerelle et/ou le réseau GSM.
SMSNoCarrier	le message ne peut être transmis au destinataire spécifié car la gateway VIDEOSMS n'est pas raccordé à l'opérateur téléphonique gérant ce numéro.
SMSQuotaExceeded	le nombre maximum de messages ou le montant maximum autorisé pour ce compte a été atteint.
SMSUnauthorizedCarrier	le compte de l'expéditeur n'est pas autorisé à envoyer des SMS à l'opérateur spécifié.
SMSInvalidAddress	le numéro du portable du destinataire est invalide.
SMSUnrecognizedAddress	la passerelle n'a pas reconnu à quel opérateur correspond le numéro spécifié, vérifiez la syntaxe du numéro du destinataire.
SMSInvalidDate	le format d'une date spécifiée est invalide.

Routines de VIDEOSMS hors session SMS+ SMS_Send

SMSThrottleError	l'utilisateur a dépassé le nombre maximum de SMS qui lui est autorisé par unité de temps.
SMSAccountNoCredit	il ne reste pas suffisamment d'unités sur ce compte en prépaiement pour envoyer ce message.

EXAMPLES

SMS_WAPPush

Cette routine permet d'envoyer une instruction push WAP SMS au destinataire désiré.

FORMAT **SMS_WAPPush** *UInt32 *ConnectId, SMSMessage
*SMSMessage, WapData *WapData,
char *private_reference, char
private_data

RETURNS VMS Usage: **cond_values**
type: **entier long**
access: **lecture seule**
mechanism: **par valeur**

ARGUMENTS **ConnectId**
VMS Usage: **pointeur**
type: **pointeur sur entier long**
access: **lecture seule**
mechanism: **par référence**
Identifiant unique de connection, initialisé par la fonction SMS_ConnectGateway

***SMSMessage**

VMS Usage: **pointeur**
type: **pointeur sur structure de type SMSMessage**
access: **lecture/écriture**
mechanism: **par référence**

Pointeur sur une structure de type *SMSMessage*.

En entrée les champs suivants de la structure doivent être initialisés:

- **messagetype**: positionné à *SMSMessageTypeSubmitted*.
- **called_number**: numéro du mobile à qui envoyer le SMS. Ce numéro doit être de la forme suivante:
 - ◇ numéro national (ex: en France 0612345678), que la passerelle transmettra alors sur le réseau national.
 - ◇ numéro international, précédé du préfixe + et du code pays: ex: +33612345678.
 - ◇ numéro international, précédé du préfixe 00 et du code pays: ex: 0033612345678

Dans tous les cas, le numéro sera converti au format international +xxx.

Routines de VIDEOSMS hors session SMS+ SMS_WAPPush

En sortie, le champ *msg_reference_number* est rempli avec l'identifiant unique assigné par la passerelle au message SMS.

Il est possible de spécifier les informations suivantes optionnelles:

- *msg_class*: spécifie la classe du message:
 - ◇ 0: classe 0: affichage immédiat sans intervention de l'utilisateur, pas de stockage du SMS parmi les SMS reçus.
 - ◇ 1: classe 1: réception du SMS, l'utilisateur devant aller le consulter (défaut).
 - ◇ 2: classe 2: stockage dans carte SIM
 - ◇ 3: classe 3: spécifique de l'équipement.
- *srcPort*: numéro de port émetteur du message, utilisé seulement pour les messages binaires NBS.
- *destPort*: numéro de port émetteur du message, utilisé seulement pour les messages binaires NBS.
- *alternate_number*: numéro de SMS secondaire, que la passerelle utilisera pour transmettre le message si le premier numéro spécifié n'est pas accessible et que le bit 4 de *sms_options* est positionné.
- *caller_number*: numéro SMS de l'appelant. Sur la plupart des réseaux, ce numéro ne peut être positionné, et le numéro SMS de l'appelant sera toujours le numéro court affecté à la passerelle par l'opérateur.
- *sender_address*: adresse email l'expéditeur du SMS. Si les bits 20 ou 21 de *sms_options* sont positionnés, un compte-rendu d'envoi ou d'erreur pourra être expédié à cette email lors de l'envoi du SMS ou d'une erreur durant la tentative d'expédition.
- *msg_options*: spécifications d'options. Les options sont spécifiées par le positionnement de bits du longword:
 - ◇ bit 1 mis: demande d'envoi du SMS à une date et heure donnée, spécifiée dans *deferredDate*.
 - ◇ bit 2 mis: demande d'envoi du SMS en priorité.
 - ◇ bit 3: date d'expiration de la validité du message spécifiée dans *expirationDate*.
 - ◇ bit 4: numéro de secours spécifié, à utiliser si le numéro principal ne peut être atteint.
 - ◇ bit 5: demande de notification (pour tous les cas).
 - ◇ bit 17 mis: le champ *application_name* est utilisé pour stocker des données spécifiques à l'application.
 - ◇ bit 18 mis: l'application spécifie une référence interne (*private_reference*) qui sera stockée avec le message et retournée dans les notifications.
 - ◇ bit 20 mis: demande d'envoi d'un message email de non-transmission à l'expéditeur en cas de non-transmission du Wap Push.
 - ◇ bit 21 mis: demande d'envoi d'un message email de confirmation à l'expéditeur après la transmission correcte du message.

Routines de VIDEOSMS hors session SMS+ SMS_WAPPush

- ◇ bit 24 mis: le SMS Wap Push devra être recyclé s'il n'a pu être expédié correctement: il restera donc en attente de recyclage dans la file d'attente.
- ◇ bit 30 mis: le SMS Wap Push est expédié en validation obligatoire, et restera dans la file d'attente jusqu'à ce qu'un responsable le valide pour expédition.

***WapData**

VMS Usage: **pointeur**

type: **pointeur sur structure de type WapData**

access: **lecture/écriture**

mechanism: **par référence**

Pointeur sur une structure de type WapData. Les champs suivants de cette structure doivent être renseignés:

- `wap_url`: URL Wap à envoyer par Wap-Push.

Des éléments optionnels peuvent être spécifiés:

- `wap_content`: contenu multimédia à transmettre à l'expéditeur, passé soit directement soit par son nom de fichier.
- `wap_text`: texte informatif à transmettre en push au destinataire avec l'URL.
- `wap_action`: type d'action push, qui peut être:
 - ◇ `medium` (défaut): affichage non obstrusif dès que possible.
 - ◇ `high`: affichage immédiat, même obstrusif (à éviter autant que possible).
 - ◇ `low`: affichage lorsque possible.
 - ◇ `none`
 - ◇ `delete`
- `wap_options`: spécifications d'options. Les options sont spécifiées par le positionnement de bits du longword:
 - ◇ bit 0 mis: le champ `wap_content` ne contient pas le contenu à transmettre, mais le nom de fichier stockant le contenu.
 - ◇ bit 1 mis: l'URL Wap doit être générée dynamiquement et donner accès au contenu multimédia passé par `wap_content`.
 - ◇ bit 2 mis: l'URL Wap générée dynamiquement est à usage unique: après un téléchargement par l'utilisateur, l'URL est détruite.
 - ◇ bit 3 mis: la date de création de l'URL doit être spécifiée à `wap_creation_time`.
 - ◇ bit 4 mis: la date d'expiration de l'URL doit être spécifiée à `wap_expiration_time`.

***private_ref**

VMS Usage: **pointeur**

type: **pointeur sur chaîne de 0 à 15 caractères**

access: **lecture seule**

Routines de VIDEOSMS hors session SMS+ SMS_WAPPush

mechanism: **par référence**

Pointeur sur une chaîne de 15 caractères maximum contenant un identifiant alphanumérique privé du message, spécifique de l'application envoyant le SMS. N'est utilisé que lorsque le bit 18 de *msg_options* est positionné.

***private_data**

VMS Usage: **pointeur**

type: **pointeur sur chaîne de 0 à 50 caractères**

access: **lecture seule**

mechanism: **par référence**

Pointeur sur une chaîne de caractères contenant des données spécifiques de l'application, à associer au message envoyé et à retourner dans les notifications. N'est utilisé que lorsque le bit 17 de *msg_options* est positionné.

DESCRIPTION

Cette routine permet de demander l'envoi par Wap Push SMS d'une URL Wap pouvant donner accès à tout type de données:

- page Wap au format WML.
- sonnerie polyphonique.
- image couleur.
- application Java JMEE.
- etc.

Le contenu pointé par l'URL du Wap Push peut être:

- stocké préalablement sur le serveur Wap, dans ce cas, l'URL correspondante est transmise dans le champ *Wap_URL*.
- transmis par la routine au serveur. Dans le cas, il sera stocké sous l'URL spécifiée. Il est aussi possible de ne pas spécifier d'URL mais de générer dynamiquement une URL à usage unique ou restreint lorsque le bit 1 de *wap_options* est positionné. Si le bit 2 est également positionné, l'URL sera détruite (ou rendue non accessible) après son premier téléchargement.

Il est possible de demander l'envoi d'un SMS Wap Push en priorité ou de demander l'envoi du SMS à une date et heure précise. Il est également possible de spécifier une date et heure d'expiration où le message sera détruit par l'opérateur s'il n'a pas été en mesure de le transmettre au mobile du destinataire.

STATUT DE RETOUR

SMSNormal	<p>l'expédition du SMS a bien été demandée. Ce statut ne signifie pas que le SMS a été transmis ou reçu par le destinataire, mais simplement qu'il a bien été transmis à la passerelle VIDEOSMS distante et mis en queue pour l'expédition.</p> <p>La variable msg_reference_number contient le numéro d'entrée du SMS dans la queue. Ce numéro, unique parmi tous les SMS gérés par le système, permet de consulter le statut du SMS.</p>
SMSError	<p>une erreur s'est produite.</p>
SMSLinkLost	<p>le lien avec la passerelle distante a été interrompu.</p>
SMSNoVideosms	<p>VIDÉOSMS n'est pas installé sur ce système.</p>
SMSInvalidParameter	<p>un des paramètres spécifiés est invalide.</p>
SMSFileEmpty	<p>le fichier à expédier est vide et le SMS ne sera pas transmis.</p>
SMSFileNotFound	<p>le fichier contenant le texte à envoyer n'a pas été trouvé. Le SMS ne sera pas transmis.</p>
SMSTooLong	<p>la taille du message excède la taille maximale autorisée par la passerelle et/ou le réseau GSM.</p>
SMSNoCarrier	<p>le message ne peut être transmis au destinataire spécifié car la gateway VIDEOSMS n'est pas raccordé à l'opérateur téléphonique gérant ce numéro.</p>
SMSQuotaExceeded	<p>le nombre maximum de messages ou le montant maximum autorisé pour ce compte a été atteint.</p>
SMSUnauthorizedCarrier	<p>le compte de l'expéditeur n'est pas autorisé à envoyer des SMS à l'opérateur spécifié.</p>
SMSInvalidAddress	<p>le numéro du portable du destinataire est invalide.</p>
SMSUnrecognizedAddress	<p>la passerelle n'a pas reconnu à quel opérateur correspond le numéro spécifié, vérifiez la syntaxe du numéro du destinataire.</p>
SMSInvalidDate	<p>le format d'une date spécifiée est invalide.</p>
SMSThrottleError	<p>l'utilisateur a dépassé le nombre maximum de SMS qui lui est autorisé par unité de temps.</p>
SMSAccountNoCredit	<p>il ne reste pas suffisamment d'unités sur ce compte en prépaiement pour envoyer ce message.</p>

EXAMPLES

Routines de VIDEOSMS hors session SMS+ SMS_ReadStatus

- ◇ 2: UNDELIVERABLE: SMS non transmis, supprimé définitivement de la file d'envoi. La variable *error_cause* retourne la cause de l'erreur.
- ◇ 3: TRANSMITTED: SMS assumé transmis au mobile destinataire (retourné lorsque le SMS a bien été transmis à la passerelle SMS-C de l'opérateur et acquitté, mais qu'aucune notification n'a été demandée).
- ◇ 10: en attente de validation par un superviseur.
- ◇ 11: par suite de multiples erreurs, le cycle des essais autorisés a été effectué, et le SMS reste dans la file d'attente, en attente de recyclage par l'expéditeur ou un superviseur. La variable *error_cause* retourne la cause de la dernière erreur.
- ◇ 12: EN ROUTE: en cours de transmission par l'opérateur.
- ◇ 13: ERROR: erreur dans la transmission du SMS. Le SMS reste en queue, et des essais de retransmission seront effectués régulièrement. La variable *error_cause* retourne la dernière erreur connue.
- ◇ 18: SCHEDULED: en cours d'attente, sera transmis dès que possible.
- ◇ 19: PRIORITY: SMS prioritaire, sera transmis dès que possible, et avant tout autre SMS non prioritaire.
- ◇ 21: en cours d'attente, sera transmis après *retransmit_time*.
- ◇ 99: statut inconnu ou invalide.
- *error_cause*: cause de l'erreur ayant empêché l'expédition du SMS. Cette variable peut prendre les valeurs suivantes:
 - ◇ 50: erreur non spécifique lors de l'envoi du message.
 - ◇ 51: le numéro spécifié n'existe pas.
 - ◇ 52: le mobile spécifié ne peut être atteint.
 - ◇ 53: les appels vers ce mobile sont interdits (call barred).
 - ◇ 54: le mobile est éteint ou hors couverture.
 - ◇ 55: gateway SMS-C hors service ou inaccessible.
 - ◇ 56: numéro appelé en liste rouge.
 - ◇ 57: numéro appelé en liste noire.
 - ◇ 58: numéro porté par un autre opérateur ou invalide.
 - ◇ 64: erreur ou abandon hôte.
 - ◇ 65: erreur interne.
 - ◇ 66: date d'expiration du message atteinte.
 - ◇ 67: opération interdite.
 - ◇ 68: message supprimé par un opérateur.
 - ◇ 99: autre erreur.

Routines de VIDEOSMS hors session SMS+ SMS_ReadStatus

- ◇ des erreurs supérieures à 100 spécifiques des différents réseaux peuvent également être retournées, se reporter au manuel du réseau considéré.
- price: prix de l'expédition de ce SMS, dans la devise par défaut de la passerelle.

STATUT DE RETOUR

SMSNormal	le SMS a bien été retrouvé et les éléments le concernant sont mis à jour dans la structure.
SMSError	une erreur s'est produite.
SMSLinkLost	le lien avec la passerelle distante a été interrompu.
SMSNoVideosms	VIDEOSMS n'est pas installé sur ce système.
SMSInvalidParameter	un des paramètres spécifiés est invalide.
SMSInvalidMessageId	le <i>message id</i> spécifié dans la requête ne correspond pas à un identifiant valide de message.
SMSNoPrivilege	le <i>message id</i> spécifié dans la requête correspond à un message appartenant à un autre utilisateur et/ou une autre société.

Routines de VIDEOSMS hors session SMS+

SMS_GetMessage

STATUT DE RETOUR

SMSNormal	un message SMS-MO ou une notification d'expédition a bien été récupéré et est retourné pour traitement dans la structure <i>Message_received</i> .
SMSError	une erreur s'est produite.
SMSLinkLost	le lien avec la passerelle distante a été interrompu.
SMSNoVideosms	VIDÉOSMS n'est pas installé sur ce système.
SMSNotConnected	le système n'est pas connecté à la passerelle VIDEOSMS.
SMSNotAuthenticated	le système n'est pas authentifié sur la passerelle VIDEOSMS distante.
SMSInvalidParameter	un des paramètres spécifiés est invalide.

EXAMPLES

Routines de VIDEOSMS hors session SMS+ SMS_GetLocation

STATUT DE RETOUR

SMSNormal	la géolocalisation a bien été effectuée, les données sont retournées pour traitement dans la structure <i>UserLocation</i> .
SMSNoLocation	la fonction s'est bien exécuté mais le SMS-Center distant n'avait pas de position connue pour cet abonné, ou la position était inconsistante ou non calculable: ce statut est notamment retourné pour les codes SS7/MAP suivants: Insufficient resources, Insufficient Measurement Data, Location procedure not completed, Quality of Service not attainable, position method not available in network, position method not available in location area.
SMSNoPrivilege	les droits d'accès du compte appelant ne permettent pas d'effectuer des demandes de géolocalisation.
SMSNoCarrier	l'utilisateur ne peut être géolocalisé car la gateway VIDEOSMS n'est pas raccordé à l'opérateur téléphonique gérant ce numéro.
SMSUnauthorizedNetwork	le réseau effectuant cette demande n'est pas autorisé.
SMSQuotaExceeded	le nombre maximum de messages ou le montant maximum autorisé pour ce compte a été atteint.
SMSUnauthorizedCarrier	le compte de l'expéditeur n'est pas autorisé à accéder au réseau mobile de l'utilisateur.
SMSRoamingNotAllowed	L'utilisateur est inscrit sur un réseau qui n'est pas accessible en roaming par le réseau demandeur.
SMSInvalidAddress	le numéro du portable du destinataire est invalide.
SMSUnrecognizedAddress	la passerelle n'a pas reconnu à quel opérateur correspond le numéro spécifié, vérifiez la syntaxe du numéro du destinataire.
SMSError	une erreur s'est produite.
SMSLinkLost	le lien avec la passerelle distante a été interrompu.
SMSNoVideosms	VIDÉOSMS n'est pas installé sur ce système.
SMSNotConnected	le système n'est pas connecté à la passerelle VIDEOSMS.
SMSNotAuthenticated	le système n'est pas authentifié sur la passerelle VIDEOSMS distante.
SMSInvalidParameter	un des paramètres spécifiés est invalide.
SMSAccountNoCredit	Il n'y a pas assez de crédits sur le compte en prépaiement de l'usager pour effectuer cette requête

EXAMPLES

6

Routines de gestion de session SMS+

SMS_Read—Lecture d'un message SMS-MO reçu

Cette routine demande à lire un message SMS-MO retourné par la passerelle distante.

FORMAT **SMS_Read** *UInt32 *ConnectId, *Message_received*

RETURNS VMS Usage: **cond_values**
 type: **entier long**
 access: **lecture seule**
 mechanism: **par valeur**

ARGUMENTS ***ConnectId***
 VMS Usage: **pointeur**
 type: **pointeur sur entier long**
 access: **lecture seule**
 mechanism: **par référence**
 Identifiant unique de connection, initialisé par la fonction SMS_ConnectGateway

****Message_received***
VMS Usage: **pointeur**
type: **pointeur sur structure de type SMSMessage**
access: **écriture**
mechanism: **par référence**
Pointeur sur une structure de type SMSMessage.

DESCRIPTION Cette routine permet à l'application de gestion de la requête de l'utilisateur de récupérer un message SMS-MO reçu d'un mobile distant.

Elle doit être appelée préalablement à toute intervention sur la session.

Si aucun message n'a été reçu pour l'application, le statut *SMSNoMessage* est retourné. L'application devra alors, soit se déconnecter et se reconnecter ultérieurement, soit refaire une requête *SMS_Read* à intervalles réguliers.

STATUT DE RETOUR

SMSNormal	un message SMS-MO a bien été récupéré et est retourné pour traitement dans la structure <i>Message_received</i> .
SMSNoMessage	la fonction s'est bien exécuté mais la passerelle VIDEOSMS distante n'avait aucun message en attente pour cette application.

Routines de gestion de session SMS+ SMS_Read

SMSError	une erreur s'est produite.
SMSLinkLost	le lien avec la passerelle distante a été interrompu.
SMSNoVideosms	VIDÉOSMS n'est pas installé sur ce système.
SMSNotConnected	le système n'est pas connecté à la passerelle VIDEOSMS.
SMSNotAuthenticated	le système n'est pas authentifié sur la passerelle VIDEOSMS distante.
SMSInvalidParameter	un des paramètres spécifiés est invalide.

EXAMPLES

SMS_GetSessionInfo—Récupération des informations de la session courante

Cette routine récupère les informations de la session courante.

FORMAT **SMS_GetSessionData** *UInt32 *ConnectId, *Session*

RETURNS VMS Usage: **cond_values**
 type: **entier long**
 access: **lecture seule**
 mechanism: **par valeur**

ARGUMENTS ***ConnectId***
 VMS Usage: **pointeur**
 type: **pointeur sur entier long**
 access: **lecture seule**
 mechanism: **par référence**
 Identifiant unique de connection, initialisé par la fonction SMS_ConnectGateway

****Session***
 VMS Usage: **pointeur**
 type: **pointeur sur structure de type SMSSession**
 access: **écriture**
 mechanism: **par référence**
 Pointeur sur une structure de type SMSSession.

DESCRIPTION Cette routine permet à l'application de gestion de la requête de l'utilisateur de récupérer les informations de la session courante.

Cette routine permet de récupérer les informations utiles sur la session SMS+ en cours, notamment:

- le numéro de session unique identifiant la session de l'utilisateur. Une session peut comprendre plusieurs requêtes qui seront traitées séquentiellement les unes après les autres, puisqu'elles peuvent être transmises avec un certain délai par l'utilisateur, sous forme de SMS successifs.
- le numéro de téléphone de l'appelant. Pour les services SMS+ surtaxés, le numéro de l'utilisateur est masqué et remplacé par un alias dit "glissant" qui sera le même pour les différentes requêtes émises séquentiellement par l'utilisateur.
- numéro de SMS appelé par l'utilisateur.

Routines de gestion de session SMS+ SMS_GetSessionInfo

- les données de la requête, c'est à dire le contenu du SMS envoyé par l'utilisateur au centre serveur, qui sont retournées encodées par défaut au format ISO 8859-1, sauf spécification d'un autre alphabet.
- le numéro de séquence de la requête, qui débute à 1 pour la première requête de la session et s'incrémente de 1 à chaque nouvelle requête (donc à chaque SMS envoyé par l'utilisateur).

STATUT DE RETOUR

SMSNormal	les données de la session sont retournés pour traitement dans la structure <i>SMSSession</i> .
SMSError	une erreur s'est produite.
SMSNoVideosms	VIDÉOSMS n'est pas installé sur ce système.
SMSInvalidParameter	un des paramètres spécifiés est invalide.

EXAMPLES

SMS_GetSessionData—Lecture des données spécifiques sauvegardées

Cette fonction permet de récupérer les données spécifiques sauvegardées antérieurement pour ce contexte de session.

FORMAT **SMS_GetSessionData** *UInt32 *ConnectId, char *data_name, data_type, data_length, *data_value*

RETURNS VMS Usage: **cond_value**
type: **entier long**
access: **lecture seule**
mechanism: **par valeur**

ARGUMENTS **ConnectId**
VMS Usage: **pointeur**
type: **pointeur sur entier long**
access: **lecture seule**
mechanism: **par référence**
Identifiant unique de connection, initialisé par la fonction SMS_ConnectGateway

***data_name**
VMS Usage: **string**
type: **pointeur sur chaîne de caractères**
access: **lecture/écriture**
mechanism: **par référence**
Nom de la variable spécifique à récupérer. Si cette chaîne est vide, VIDEOSMS retournera les variables les unes après les autres à chaque appel de la fonction, puis SMSNoMoreData lorsque toutes les variables auront été retournées.

data_type
VMS Usage: **longword**
type: **entier long**
access: **lecture/écriture**
mechanism: **par référence**
type des données à récupérer/récupérées.
valeurs légales:

- 1: Int (longword).
- 2: Int (word).
- 3: chaîne de caractères

Routines de gestion de session SMS+

SMS_GetSessionData

- 4: structure spécifique à l'application

data_length

VMS Usage: **longword**

type: **entier long**

access: **écriture seule**

mechanism: **par référence**

longueur utile des données retournées dans *data_content*

****data_value***

VMS Usage: **variable**

type: **pointeur sur un buffer de taille adaptée**

access: **écriture seule**

mechanism: **par référence**

valeur de la variable demandée. Dans le cas d'une structure spécifique à l'application, il est de la responsabilité de l'application de passer un pointeur sur la structure adaptée.

DESCRIPTION

STATUT DE RETOUR

SMSNormal	la variable a été retrouvée et est retournée dans <i>data_value</i> .
SMSError	une erreur s'est produite.
SMSNoSession	il n'y a pas de session applicative ouverte.
SMSNoVideosms	VIDÉOSMS n'est pas installé sur ce système.
SMSInvalidParameter	un des paramètres spécifiés est invalide.

EXAMPLES

Exemple

SMS_UpdateSessionData—Sauvegarde de données spécifiques à l'application

Cette fonction permet de sauvegarder des données spécifiques à l'application, qui seront récupérées ensuite lors de la réception d'un SMS suivant pour cette même session de dialogue avec un utilisateur distant.

FORMAT **SMS_UpdateSessionData** *UInt32 *ConnectId, char *data_name, data_type, data_length, *data_value*

RETURNS VMS Usage: **cond_value**
 type: **entier long**
 access: **lecture seule**
 mechanism: **par valeur**

ARGUMENTS **ConnectId**
 VMS Usage: **pointeur**
 type: **pointeur sur entier long**
 access: **lecture seule**
 mechanism: **par référence**
 Identifiant unique de connection, initialisé par la fonction SMS_ConnectGateway

***data_name**
 VMS Usage: **string**
 type: **pointeur sur chaîne de caractères**
 access: **lecture**
 mechanism: **par référence**
 Nom de la variable spécifique à sauvegarder.

data_type
 VMS Usage: **longword**
 type: **entier long**
 access: **lecture seule**
 mechanism: **par référence**
 type des données à sauvegarder.

valeurs légales:

- 1: Int (longword).
- 2: Int (word).
- 3: chaîne de caractères.
- 4: structure spécifique à l'application.
- 5: real double precision.

Routines de gestion de session SMS+ SMS_UpdateSessionData

- 99: demande de suppression de la variable déjà sauvegardée, à supprimer de la structure. Les paramètres *data_length* et *data_value* sont dans ce cas ignorés.

data_length

VMS Usage: **longword**
type: **entier long**
access: **lecture seule**
mechanism: **par référence**
longueur utile des données à sauvegarder, stockées dans *data_content*

****data_value***

VMS Usage: **variable**
type: **pointeur sur un buffer de taille adaptée**
access: **lecture seule**
mechanism: **par référence**
valeur de la variable à sauvegarder.

DESCRIPTION

La sauvegarde effective des données ne sera effectuée que lors de l'envoi du SMS de réponse par la fonction *SMS_Reply*.

STATUT DE RETOUR

SMSNormal	les données spécifiques ont été enregistrées pour sauvegarde.
SMSNoSession	il n'y a pas de session applicative courante.
SMSBufferSizeExceeded	la taille des données excède celle du buffer utilisé pour sauvegarder les données.
SMSError	une erreur s'est produite.
SMSLinkLost	le lien avec la passerelle distante a été interrompu.
SMSNoVideosms	VIDEOSMS n'est pas installé sur ce système.
SMSNotConnected	le système n'est pas connecté à la passerelle VIDEOSMS.
SMSNotAuthenticated	le système n'est pas authentifié sur la passerelle VIDEOSMS distante.
SMSInvalidParameter	un des paramètres spécifiés est invalide.

EXAMPLES

Exemple

SMS_Reply—Renvoi d'un SMS-MT en réponse au SMS-MO préalablement reçu

Cette routine permet de renvoyer un SMS-MT répondant à la requête de l'utilisateur distant, préalablement reçue par un SMS-MO.

FORMAT **SMS_Reply** *UInt32 *ConnectId, *Message_to_send, SMSSessionClosed*

RETURNS VMS Usage: **cond_values**
 type: **entier long**
 access: **lecture seule**
 mechanism: **par valeur**

ARGUMENTS ***ConnectId***
VMS Usage: **pointeur**
type: **pointeur sur entier long**
access: **lecture seule**
mechanism: **par référence**
Identifiant unique de connection, initialisé par la fonction SMS_ConnectGateway

****Message_to_send***

VMS Usage: **pointeur**
type: **pointeur sur structure de type SMSMessage**
access: **lecture/écriture**
mechanism: **par référence**

Pointeur sur une structure de type SMSMessage.

En entrée les champs suivants de la structure doivent être initialisés:

- **messagetype**: positionné à `SMSMessageTypeReply`.
- **msg_length**: longueur du message à transmettre.
- **msg_data**: contenu du SMS à transmettre, qui peut contenir:
 - ◇ des données uniquement numériques (`msg_format=SMSFormatNumeric`).
 - ◇ des données alphanumériques (défaut).
 - ◇ des données binaires à transmettre en mode transparent, soit sous forme de données binaires, soit sous forme de données hexadécimales.
 - ◇ un nom de fichier contenant les données à expédier. Dans ce cas, le bit 0 de `msg_options` doit être positionné à 1.

Routines de gestion de session SMS+

SMS_Reply

En sortie, le champ *msg_reference_number* est rempli avec l'identifiant unique assigné par la passerelle au message SMS.

Il est possible de spécifier les informations suivantes optionnelles:

- *msg_encoding*: par défaut SMSEncodingIA5.
- *msg_class*: spécifie la classe du message:
 - ◇ 0: classe 0: affichage immédiat sans intervention de l'utilisateur, pas de stockage du SMS parmi les SMS reçus.
 - ◇ 1: classe 1: réception du SMS, l'utilisateur devant aller le consulter.
 - ◇ 2: classe 2: stockage dans carte SIM
 - ◇ 3: classe 3: spécifique de l'équipement.
- *msg_format*: format du message, si celui-ci est différent du défaut (le défaut est *SMSFormatAlphabetic*)
- *srcPort*: numéro de port émetteur du message, utilisé seulement pour les messages binaires NBS.
- *destPort*: numéro de port émetteur du message, utilisé seulement pour les messages binaires NBS.
- *sender_address*: adresse email l'expéditeur du SMS. Si les bits 20 ou 21 de *sms_options* sont positionnés, un compte-rendu d'envoi ou d'erreur pourra être expédié à cette email lors de l'envoi du SMS ou d'une erreur durant la tentative d'expédition.
- *sms_action*: action à effectuer après la transmission du SMS (utilisé seulement sur certaines passerelles UCP V.2):
 - ◇ 0: (SMS_NOOP) aucune opération, il s'agit d'un message de dialogue (défaut).
 - ◇ 1: (SMS_END) fermeture de la transaction (d'où paiement si applicable) ainsi que de la session de service; équivaut à SMS_CONFIRM + SMS_CLOSE.
 - ◇ 2 (SMS_CONFIRM) fermeture de la transaction (d'où déclenchement du paiement si applicable).
 - ◇ 3 (SMS_CLOSE) fermeture de la session de service.
- *msg_options*: spécifications d'options. Les options sont spécifiées par le positionnement de bits du longword:
 - ◇ bit 0 mis: la chaîne *msg_data* ne contient pas les données à transmettre mais le nom du fichier contenant les données.
 - ◇ bit 3: date d'expiration de la validité du message spécifiée dans *expirationDate*.
 - ◇ bit 5: demande de notification (pour tous les cas).
 - ◇ bit 20 mis: demande d'envoi d'un message email de non-transmission à l'expéditeur en cas de non-transmission du SMS.
 - ◇ bit 21 mis: demande d'envoi d'un message email de confirmation à l'expéditeur après la transmission correcte du message.

Routines de gestion de session SMS+

SMS_Reply

SMSSessionClosed

VMS Usage: **longword**

type: **entier long**

access: **lecture seule**

mechanism: **par référence**

Flag à 1 si la session applicative doit être fermée, lorsque plus aucun échange n'est attendu avec l'utilisateur dans le cadre de cette session.

DESCRIPTION

Cette routine permet de renvoyer un SMS-MT en réponse à une requête d'un utilisateur SMS+ distant, reçue préalablement par un message SMS-MO.

Le contenu du SMS à transmettre peut être passé à la routine ou stocké dans un fichier dont le nom complet est transmis à la routine. Le fichier doit être un fichier séquentiel.

Le numéro du destinataire n'est pas à spécifier puisqu'il est celui de l'expéditeur du SMS-MO précédent, connu de VIDEOSMS.

Lorsque le protocole UCP V.2 est utilisé pour la communication avec le réseau distant, la variable *sms_action* Cette routine permet de gérer la session UCP v.2:

- l'action CONFIRM ferme la session de transaction sur cet échange SMS en déclenchant le paiement si applicable.
- l'action CLOSE ferme la session de service.
- l'action END ferme les deux sessions (transaction et service).
- la session globale (série d'échange) peut être fermée par ailleurs en positionnant le flag *sms_sessionclosed* lors de l'envoi de la réponse.

STATUT DE RETOUR

SMSNormal	<p>l'expédition du SMS a bien été demandée. Ce statut ne signifie pas que le SMS a été transmis ou reçu par le destinataire, mais simplement qu'il a bien été transmis à la passerelle VIDEOSMS distante et mis en queue pour l'expédition.</p> <p>La variable <i>msg_reference_number</i> contient le numéro d'entrée du SMS dans la queue. Ce numéro, unique parmi tous les SMS gérés par le système, permet de consulter le statut du SMS.</p>
SMSError	<p>une erreur s'est produite.</p>
SMSLinkLost	<p>le lien avec la passerelle distante a été interrompu.</p>
SMSNoSession	<p>il n'y a pas de session applicative courante. On ne peut donc répondre puisqu'il n'y a pas de message reçu, il faut utiliser la fonction <i>SMS_Send</i> pour envoyer un SMS au destinataire désiré, en spécifiant alors son numéro.</p>
SMSNoVideosms	<p>VIDÉOSMS n'est pas installé sur ce système.</p>

Routines de gestion de session SMS+

SMS_Reply

SMSInvalidParameter	un des paramètres spécifiés est invalide.
SMSFileEmpty	le fichier à expédier est vide et le SMS ne sera pas transmis.
SMSFileNotFound	le fichier contenant le texte à envoyer n'a pas été trouvé. Le SMS ne sera pas transmis.
SMSTooLong	la taille du message excède la taille maximale autorisée par la passerelle et/ou le réseau GSM.
SMSInvalidDate	le format d'une date spécifiée est invalide.

EXAMPLES

7

Routines de gestion et de supervision

SMS_GetProcessedMessage—Récupération d'un message traité

Cette fonction permet d'obtenir un par un tous les SMS émis ou reçus par la passerelle pour un utilisateur donné.

FORMAT	SMS_GetProcessedMessage	<i>UInt32 *ConnectId,</i> <i>UInt32 *msgtype,</i> <i>UInt32 *msgoptions,</i> <i>SMSMessage</i> <i>*SMSMessage</i>
---------------	--------------------------------	-------------------------------------------------------------------------------------------------------------------------------

RETURNS	VMS Usage: cond_value type: entier long access: lecture seule mechanism: par valeur
----------------	--------------------------------------------------------------------------------------------------------------------------

ARGUMENTS	<i>ConnectId</i> VMS Usage: pointeur type: pointeur sur entier long access: lecture seule mechanism: par référence Identifiant unique de connection, initialisé par la fonction <i>SMS_ConnectGateway</i>
	<i>msgtype</i> VMS Usage: pointeur type: pointeur sur entier long access: lecture seule mechanism: par référence Type de message à retrouver: <ol style="list-style-type: none">1 SMS envoyé.2 SMS reçu et traité. Les SMS reçus et non encore traités ne se retrouvent pas avec cette routine mais avec la routine <i>SMS_Read</i>.
	<i>msgoptions</i> VMS Usage: pointeur type: pointeur sur entier long access: lecture seule mechanism: par référence Options: les bits suivants peuvent être positionnés:

Routines de gestion et de supervision

SMS_GetProcessedMessage

- 0: full data: dans ce cas l'intégralité du message est retrouvé et renvoyé. Si ce bit n'est pas positionné, afin de diminuer la charge réseau, seules les informations suivantes sont retournées:
 - ◇ numéro de référence du message (*msg_reference_number*)
 - ◇ numéro appelé (*called_number*)
 - ◇ numéro appelant (*caller_number*)
 - ◇ statut du message (*sms_state*)
 - ◇ erreur (*sms_error*)
 - ◇ date de soumission (*posted_date*)
 - ◇ date de transmission ou d'erreur définitive (*transmitted_date*)
- 1: get next: si ce bit est positionné alors qu'une requête identique a été effectuée précédemment, la routine retourne alors le message suivant de la liste ou une erreur s'il n'y a plus de message. La requête précédente doit impérativement être une requête *SMS_GetProcessedMessage* avec le bit 2 positionné.
- 2: expect next: si un message a été retrouvé, le programme appelant effectuera une requête suivante. Ce statut demande au programme de ne pas clôturer les fichiers ouverts pour le traitement.
- 3: return also charging data: si ce bit est positionné, le programme retourne non seulement les SMS mais également les informations de gestion de la taxation du compte: débit, crédit, et autres opérations.

***SMSMessage**

VMS Usage: **pointeur**

type: **pointeur sur structure**

access: **écriture seule**

mechanism: **par référence**

Pointeur sur une structure de type *SMSMessage* contenant les données du message retrouvé.

DESCRIPTION

Cette routine est utilisée pour retrouver tous les messages émis ou reçus par un compte utilisateur donné. Il n'est pas nécessaire de connaître le numéro de référence des messages pour les retrouver. Cette routine doit être appelée après identification préalable sous ce compte utilisateur, par le superviseur de sa société ou par un autre compte disposant du privilège management.

Le status retourné dans le message de réponse peut être:

- x00: normal, message retrouvé et retourné
- x02: command length is invalid
- x04: incorrect bind status for given command: la session n'a pas été préalablement ouverte par une commande *SMSLoginGateway* effectuée avec succès
- x08: system error: la passerelle est actuellement indisponible.

Routines de gestion et de supervision

SMS_GetProcessedMessage

- x0101: prohibited from using specified operation: le compte de l'utilisateur n'est pas autorisé à effectuer cette opération. On renvoie l'erreur *SMSNoPrivilege*.
- x0403: aucun message retrouvé (ou plus de message après ceux renvoyé précédemment): on renvoie l'erreur *SMSNoMessage*

RETURN VALUES

SMSNormal	un message a été retrouvé et est retourné.
SMSNoMessage	aucun ou plus de message à renvoyer.
SMSNoPrivilege	les droits d'accès du compte appelant ne permettent pas d'accéder aux données de ce compte.
SMSError	une erreur s'est produite.
SMSLinkLost	le lien avec la passerelle distante a été interrompu.
SMSNoVideosms	VIDÉOSMS n'est pas installé sur ce système.
SMSNotConnected	le système n'est pas connecté à la passerelle VIDEOSMS.
SMSInvalidParameter	un des paramètres spécifiés est invalide.

EXAMPLES

Exemple

Routines de gestion et de supervision

SMS_GetAccountRecord

- 1: get next: si ce bit est positionné alors qu'une requête identique a été effectuée précédemment, la routine retourne alors l'enregistrement suivant de la liste ou une erreur s'il n'y a plus d'enregistrement. La requête précédente doit impérativement être une requête *SMS_GetAccountRecord* avec le bit 2 positionné.
- 2: expect next: si un message a été retrouvé, le programme appelant effectuera une requête suivante. Ce statut demande au programme de ne pas clôturer les fichiers ouverts pour le traitement.

***SMSChargingData**

VMS Usage: **pointeur**

type: **pointeur sur structure**

access: **écriture seule**

mechanism: **par référence**

Pointeur sur une structure de type *SMSChargingData* contenant les données de taxation retrouvées.

DESCRIPTION

Cette routine est utilisée pour retrouver toutes les opérations de taxation par un compte utilisateur donné. Cette routine doit être appelée après identification préalable sous ce compte utilisateur, par le superviseur de sa société ou par un autre compte disposant du privilège management.

Le status retourné dans le message de réponse peut être:

- x00: normal, données de taxation retrouvées et retournées
- x02: command length is invalid
- x04: incorrect bind status for given command: la session n'a pas été préalablement ouverte par une commande *SMSLoginGateway* effectuée avec succès
- x08: system error: la passerelle est actuellement indisponible.
- x0101: prohibited from using specified operation: le compte de l'utilisateur n'est pas autorisé à effectuer cette opération. On renvoie l'erreur *SMSNoPrivilege*.
- x0403: aucune données retrouvées (ou plus de données après celles renvoyées précédemment): on renvoie l'erreur *SMSNoMessage*

RETURN VALUES

SMSNormal	un ticket de taxation a été retrouvé et est retourné.
SMSNoMessage	aucun ou plus de ticket de taxation à renvoyer.
SMSNoPrivilege	les droits d'accès du compte appelant ne permettent pas d'accéder aux données de ce compte.
SMSError	une erreur s'est produite.
SMSLinkLost	le lien avec la passerelle distante a été interrompu.
SMSNoVideosms	VIDÉOSMS n'est pas installé sur ce système.

Routines de gestion et de supervision

SMS_GetAccountRecord

SMSNotConnected	le système n'est pas connecté à la passerelle VIDEOSMS.
SMSInvalidParameter	un des paramètres spécifiés est invalide.

EXAMPLES

Exemple

8 Fichier d'inclure smsApi1.h

Example 8-1 Fichier d'inclure smsApi1.h

```
**
** Header file smsApi/smsApi.h
** Interface publique pour smsApi.
** Ce fichier centralise toute la partie publique necessaire a l'utilisation
** de smsApi version 2.
**
** Authors LOKHATE Eric [EL], ZVODAR Philippe [ZP]
** Copyright (c) 2004-2005 Monaco Telematique / MCTEL-SAM
*/
#ifdef __MCTEL_SMSAPI_H__
#define __MCTEL_SMSAPI_H__

#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <time.h>

/*-----*/

#ifdef WIN32
/* Windows */
#include <windows.h>
#define _STDCALL_ __stdcall
#else
/* Linux ou autre plate-forme que windows */
#define _STDCALL_

typedef char* LPSTR;
#endif /* WIN32 */

typedef char CHAR;

/*-----*/

#ifdef __cplusplus
extern "C" {
#endif

#define SMS_ASSERT_CT( expr ) do { \
switch( 1 ) { \
case 0: \
break; \
case (expr): \
break; \
} \
} while ( 0 )

#define SMS_CHECK_FIXEDSIZE( ) do { \
SMS_ASSERT_CT( sizeof( smsInt8 ) == 1 ); \
SMS_ASSERT_CT( sizeof( smsInt16 ) == 2 ); \
SMS_ASSERT_CT( sizeof( smsInt32 ) == 4 ); \
SMS_ASSERT_CT( sizeof( smsUInt8 ) == 1 ); \
SMS_ASSERT_CT( sizeof( smsUInt16 ) == 2 ); \
SMS_ASSERT_CT( sizeof( smsUInt32 ) == 4 ); \
} while ( 0 )
```

Example 8-1 Cont'd on next page

Example 8-1 (Cont.) Fichier d'inclure smsApi1.h

```

/** @name Types "Fixed Size"
 * @{
 */
typedef unsigned char smsInt8; /* Entier signe sur 8 bits */
typedef short smsInt16; /* Entier signe sur 16 bits */
typedef int smsInt32; /* Entier signe sur 32 bits */
typedef unsigned char smsUInt8; /* Entier non signe sur 8 bits */
typedef unsigned short int smsUInt16; /* Entier non signe sur 16 bits */
typedef unsigned long int smsUInt32; /* Entier non signe sur 32 bits */
#define smsByte smsInt8
#define smsWord smsInt16
#define smsDWord smsInt32
#define smsUByte smsUInt8
#define smsUWord smsUInt16
#define smsUDWord smsUInt32
#define smsReal float
#define smsDReal double
/** @} */

/** @name Type "Return Code"
 * @{ */
typedef enum {
SMSNoError = 0, /* Aucune erreur signalée */
SMSNormal = 1, /* Execution correcte de la fonction */
SMSError = 2, /* Erreur non documentee */
SMSUpdateSuggested = 3, /* Une mise a jour est disponible */
SMSNoVideosms = 4, /* Videosms n'est pas installe sur ce systeme */
SMSInvalidParameter = 6, /* Un ou plusieurs parametres specifiées sont invalides */
SMSNoNetwork = 8, /* Pas de liaison reseau utilisable pour acceder a la passerelle dist
SMSCannotConnect = 10, /* Connection avec la passerelle distante impossible */
SMSNoAnswer = 12, /* Pas de reponse de la passerelle distante */
SMSInvalidIPAddress = 14, /* Adresse ip appelante non autorisee a utiliser la passerelle
SMSInvalidLoginInfo = 16, /* Username et/ou mot de passe de la societe invalides */
SMSInvalidMessageID = 18, /* Format ou numero de message invalide */
SMSNoMessage = 20, /* Pas de message en attente */
SMSLinkLost = 22, /* Liaison avec la passerelle SMS distante interrompue */
SMSNoCarrier = 24, /* Pas d'operateur reseau SMS pour transmettre le message a ce numer
SMSMissingParameter = 26, /* Un parametre ou un champ obligatoire est manquant */
SMSNoPrivilege = 28, /* Les droits d'accès de l'utilisateur ne permettent pas d'effectu
SMSInvalidUserID = 30, /* UserID invalide ou non retrouve */
SMSCompanyLocked = 32, /* Le compte de la societe est bloque */
SMSUserIDLocked = 34, /* Le compte utilisateur est bloque */
SMSAccountNoCredit = 36, /* Le compte n'a plus de credits */
SMSObsoleteVersion = 38, /* La version du client est obsolete, upgrade necessaire */
SMSAccountAlreadyExist = 40, /* Le compte utilisateur existe deja */
SMSNotIdentified = 42, /* Appel à une fonction sans Login préalable */
SMSAccountNotExist = 44, /* Compte inexistant */
SMSInvalidDate = 46, /* Date incorrecte */
SMSFileNotFound = 48, /* Fichier introuvable */
SMSUpdateCfgFiles = 50, /* Mise à jour des fichiers de configuration nécessaire */
SMSUnknownOrInvalidNumber = 51, /* Numéro inconnu/invalidé */
SMSMobileAlreadyExists = 52, /* Numéro déjà utilisé pour un autre compte */
SMSRecipientUnreachable = 54, /* Mobile éteint/hors couverture, ou fax occupé/ne répond p
SMSSGatewayUnreachable = 55, /* Gateway SMS-C hors service ou inaccessible (pour un fax :
SMSNumberInRedList = 56, /* Numéro appelé en liste rouge */
SMSNumberInBlackList = 57, /* Numéro appelé en liste noire */
SMSOperatorChanged = 58, /* Numéro porté à un autre opérateur (pour fax : pas un fax gro
SMSSFaxInterfaceNotReady = 59, /* Interface fax non initialisée */
SMSRedListIsFull = 60, /* Liste rouge pleine */
SMSBlackListIsFull = 61, /* Liste noire pleine */
SMSNumberNotInUse = 62, /* Numéro non utilisé */
SMSNotAFax = 63, /* Ce n'est pas un numéro de fax */
SMSHostFailed = 64, /* Erreur ou abandon hôte */
SMSInternalError = 65, /* Erreur interne */

```

Example 8-1 Cont'd on next page

Example 8-1 (Cont.) Fichier d'inclure smsApi1.h

```

SMSMessageExpired = 66, /* Date d'expiration atteinte */
SMSForbiddenOperation = 67, /* Opération interdite */
SMSMsgDeletedByOperator = 68, /* Message supprimé par un opérateur */
SMSOpenPayingAccount = 70, /* Compte payant uniquement : pas d'unités gratuites */
SMSOtherError = 99, /* Autre erreur */
SMSWebHtmlErr = 2000,
SMSWebCfgErr = 2002,
SMSWebCfgParErr = 2004,
SMSWebUnreachable = 2006,
SMSWebInvalidPage = 2008,
SMSWebStringNotFound = 2010,
SMSWebNearDataNotFound = 2012,

/* Valeurs temporaires */
SMSInvalidAddress = 1000, /* Adresse incorrecte */
SMSThrottleError = 1002,
SMSUnauthorizedCarrier = 1004, /* Opérateur non supporté */
SMSQuotaExceeded = 1006,
SMSTooManyResendPwd = 1007, /* Trop de demandes de renvoi du mot de passe */
SMSExpiredPromoCode = 1008, /* Code promo hors délai */
SMSMemoryError = 6000 /* Erreur d'allocation mémoire */
} smsRet;
/** @} */

/** @name Type "ConnID"
 * @{ */
typedef struct sms_connid_s *smsConnID; /* Type "opaque" permettant d'identifier une conn

/* Flags à utiliser avec SMS_LoginGateway */
typedef enum {
    SMSLoginOpt_ReceiveOnly = 0x00000001 // bit 0
} SMSLoginOpt;

/* Message type */
typedef enum {
    SMSMessageTypeReceived = 0, /* Message SMS-MO reçu */
    SMSMessageTypeReport = 1, /* Message de notification SMS */
    SMSMessageTypeSubmitted = 2, /* Message SMS-MT non sollicité (push SMS-MT) à transmettre
    SMSMessageTypeDelivered = 3, /* Message SMS-MT transmis au destinataire */
    SMSMessageTypeReply = 4, /* Message SMS-MT envoyé en réponse à une requête préalable
    FAXMessageTypeReceived = 10, /* Message fax reçu */
    FAXMessageTypeReport = 11, /* Message de notification fax */
    FAXMessageTypeSubmitted = 12 /* Message fax à transmettre */
} SMSMessageType;

/* Message format */
typedef enum {
    SMSFormatAlphabetic = 3, /* Données alphanumériques encodées selon l'encodage spécifié
    SMSFormatNumeric = 4, /* Données uniquement numériques */
    SMSFormatBinary = 2, /* Données binaires */
    SMSFormatHexa = 1 /* Données binaires encodées en hexadécimal */
} SMSFormat;

/* Message Encoding */
typedef enum {
    SMSEncodingIA5 = 1, /* Encodage CCITT T.50/ISO 8859-1 (défaut) */
    SMSEncodingGSM = 2, /* Encodage GSM */
    SMSEncodingUnicode = 3 /* Encodage Unicode */
} SMSEncoding;

```

Example 8-1 Cont'd on next page

Example 8–1 (Cont.) Fichier d'inclure smsApi1.h

```
/* Message State */
typedef enum {
    SMSStateNothing = 0, /* Aucun statut reçu */
    SMSStateDelivered = 1, /* Reçu par le destinataire (voir date et heure de réception) */
    SMSStateUndeliverable = 2, /* Erreur fatale : impossible d'envoyer ce message */
    SMSStateTransmitted = 3, /* Reçu (sans autre information) */
    SMSStateEnroute = 12, /* Transmis à la passerelle MCTEL */
    SMSStateError = 13, /* Erreur temporaire (voir code d'erreur) : essaiera */
    SMSStateScheduled = 18, /* En cours d'acheminement chez l'opérateur */
    SMSStatePriority = 19, /* Msg prioritaire: sera transmis dès que possible avant tout autre */
    SMSStateDeferred = 21, /* Message différé : attente de l'heure prévue */
    SMSStateUnknown = 99 /* Etat inconnu ou incorrect */
} SMSState;

/* Types à utiliser avec SMSGetProcessedMessage */
typedef enum {
    SMSGetType_All = 0,
    SMSGetType_Sent = 1,
    SMSGetType_Received = 2
} SMSGetType;

/* Options à utiliser avec SMSGetProcessedMessage */
typedef enum {
    SMSGetOpt_FullData = 1, /* L'intégralité du message est renvoyé, sauf ChargingData */
    SMSGetOpt_GetNext = 2, /* Demande le message suivant */
    SMSGetOpt_ExpectNext = 4, /* Ceci n'est pas la dernière demande */
    SMSGetOpt_ChargingData = 8 /* Demande aussi les infos de taxation */
} SMSGetOpt;

/* Longueur maximale des champs de la structure SMSMessage */
#define SMS_MESSAGE_REFERENCE_MAX_LEN 60
#define SMS_MESSAGE_NUMBER_MAX_LEN 20
#define SMS_MESSAGE_SENDER_ADDRESS_MAX_LEN 50
#define SMS_MESSAGE_MSG_DATA_MAX_LEN 500
#define SMS_MESSAGE_DATE_MAX_LEN 18
#define SMS_MESSAGE_COMMENTS_MAX_LEN 10

typedef smsUInt32 smsDateTime;

typedef struct {
    SMSMessageType message_type;
    CHAR msg_reference_number[SMS_MESSAGE_REFERENCE_MAX_LEN];
    CHAR called_number[SMS_MESSAGE_NUMBER_MAX_LEN];
    CHAR caller_number[SMS_MESSAGE_NUMBER_MAX_LEN];
    CHAR alternate_number[SMS_MESSAGE_NUMBER_MAX_LEN];
    smsUInt16 dest_port;
    smsUInt16 src_port;
    smsUInt32 msg_length;
    SMSFormat msg_format;
    smsUInt32 msg_class;
    SMSEncoding msg_encoding;
    smsUInt32 msg_action;
    smsUInt32 msg_gsm_flags;
    smsUInt32 msg_options;
    smsUInt32 country;
    smsUInt32 carrier;
    SMSState sms_state;
    smsUInt32 sms_error;
    smsUInt32 credit_cost;
    smsDateTime posted_date;
    CHAR transmitted_date[SMS_MESSAGE_DATE_MAX_LEN];
    smsDateTime deferred_date;
    smsDateTime expiration_date;
    CHAR sender_address[SMS_MESSAGE_SENDER_ADDRESS_MAX_LEN];
    CHAR msg_data[SMS_MESSAGE_MSG_DATA_MAX_LEN];
};
```

Example 8–1 Cont'd on next page

Example 8-1 (Cont.) Fichier d'include smsApi1.h

```

} SMSMessage;

/* Masques à combiner pour les options d'envoi */
typedef enum {
    SMSSendOpt_Deferred      = 0x00000002, // bit 1
    SMSSendOpt_Priority      = 0x00000004, // bit 2
    SMSSendOpt_Valididid    = 0x00000008, // bit 3
    SMSSendOpt_ReplyEnabled  = 0x00010000, // bit 16
    //SMSSendOpt_PrivateRef  = 0x00040000, // bit 18
    SMSSendOpt_NotifOnError  = 0x00100000, // bit 20
    SMSSendOpt_NotifOnSuccess = 0x00200000, // bit 21
    SMSSendOpt_NotifByHTTP   = 0x20000000, // bit 29
    SMSSendOpt_Confidential  = 0
} SMSSendOpt;

/* Longueur maximale des paramètres privés */
#define SMS_SEND_PRIVATE_REF_MAX_LEN 15
#define SMS_SEND_PRIVATE_DATA_MAX_LEN 50

#define SMS_WAP_URL_MAX_LEN 100
#define SMS_WAP_TEXT_MAX_LEN 50
#define SMS_MSISDN_MAX_LEN 20
#define SMS_WAP_CONTENT_MAX_LEN 500

typedef struct {
    smsUInt16  dest_Port;
    smsUInt16  src_Port;
    smsUInt32  wap_length;
    smsUInt32  wap_action;
    smsUInt32  wap_encoding;
    smsUInt32  wap_options;
    smsDateTime create_date;
    smsDateTime expiration_date;
    CHAR      wap_url[SMS_WAP_URL_MAX_LEN];
    CHAR      wap_text[SMS_WAP_TEXT_MAX_LEN];
    CHAR      MSISDN[SMS_MSISDN_MAX_LEN];
    CHAR      wap_content[SMS_WAP_CONTENT_MAX_LEN];
} WAPData;

typedef enum {
    WAPPushOpt_FileContent  = 0x0001, // bit 0
    WAPPushOpt_GenerateURL  = 0x0002, // bit 1
    WAPPushOpt_UniqueURL    = 0x0004, // bit 2
    WAPPushOpt_CreateDate   = 0x0008, // bit 3
    WAPPushOpt_Validity     = 0x0010 // bit 4
} WAPPushOpt;

/*-----*/

```

Example 8-1 Cont'd on next page

Example 8-1 (Cont.) Fichier d'inclure smsApi1.h

```

/**
 * Etablissement d'une connexion avec une passerelle distante
 * @param[out] ConnID      Pointeur sur une variable qui recevra un pointeur sur le ConnID
 * @param[in]  GatewayAddress Adresse de la passerelle distante
 * @param[in]  GatewayAddressLen Nombre de caractere dans GatewayAddress (0 est autorise)
 * @param[in]  GatewayPort   Numero de port
 * @return
 *   SMSNormal
 *   SMSError
 *   SMSNoVideosms
 *   SMSNoNetwork
 *   SMSCannotConnect
 *   SMSAlreadyConnected
 *   SMSNoAnswer
 *   SMSInvalidIPAddress
 */
smsRet _STDCALL_ SMS_ConnectGateway( smsConnID* ConnID,
                                     LPSTR  GatewayAddress,
                                     smsUInt32 GatewayAddressLen,
                                     smsUInt32 Port );

/*-----*/
/**
 * Deconnexion de la passerelle distante
 * @param[in] ConnID Identifiant de connection
 * @return
 *   SMSNormal
 */
smsRet _STDCALL_ SMS_DisconnectGateway( smsConnID ConnID );

/*-----*/
/**
 * Identification sur la passerelle distante
 * @param[in] ConnID      Identifiant de connection
 * @param[in] CompanyID  IdentifiantSociete
 * @param[in] CompanyPassword Mot de passe Societe
 * @param[in] UserID     Identifiant Utilisateur
 * @param[in] UserPassword Mot de passe Utilisateur
 * @param[in] SystemID   Identifiant system
 * @param[in] Flags      Flags d'options SMS_
 * @param[out] NbrCredits Pointeur sur une variable qui recevra le nombre de credits d
 * @return
 *   SMSNormal
 *   SMSError
 *   SMSLinkLost
 *   SMSNoVideosms
 *   SMSInvalidParameter
 *   SMSInvalidLoginInfo
 *   SMSInvalidUserID
 *   SMSCompanyLocked
 *   SMSUserIDLocked
 *   SMSAccountNoCredit
 *   SMSInvalidIPAddress
 *   SMSObsoleteVersion
 */
smsRet _STDCALL_ SMS_LoginGateway( smsConnID ConnID,
                                   LPSTR  CompanyID,
                                   LPSTR  CompanyPassword,
                                   LPSTR  UserID,
                                   LPSTR  UserPassword,
                                   LPSTR  SystemID,
                                   SMSLoginOpt Flags,
                                   smsUInt32* NbrCredits);

```

Example 8-1 Cont'd on next page

Example 8-1 (Cont.) Fichier d'include smsApi1.h

```

/*-----*/
/**
 * Fermeture d'une session sur la passerelle distante
 * @param[in] ConnID identifiant de connection
 * @return
 *   SMSNormal
 *   SMSError
 *   SMSLinkLost
 *   SMSNoVideosms
 */
smsRet _STD_CALL_ SMS_LogoutGateway( smsConnID ConnID );
/*-----*/

/**
 * Envoi d'un sms
 * @param[in] ConnID Identifiant de connection
 * @param[in,out] Message Pointeur sur un SMSMessage valide
 * @return
 *   SMSNormal
 *   SMSError
 *   SMSLinkLost
 *   SMSNoVideosms
 *   SMSInvalidParameter
 *   SMSFileEmpty
 *   SMSFileNotFound
 *   SMSTooLong
 *   SMSNoCarrier
 *   SMSQuotaExceeded
 *   SMSUnauthorizedCarrier
 *   SMSInvalidIPAddress
 *   SMSUnrecognizedAddress
 *   SMSInvalidDate
 *   SMSThrottleError
 */
smsRet _STD_CALL_ SMS_Send( smsConnID ConnID,
    SMSMessage* Message,
    char PrivateReference[SMS_SEND_PRIVATE_REF_MAX_LEN],
    char PrivateData[SMS_SEND_PRIVATE_DATA_MAX_LEN] );
/*-----*/

```

Example 8-1 Cont'd on next page

Example 8-1 (Cont.) Fichier d'inclure smsApi1.h

```
/**
 * Envoi d'un message wap
 * @param[in] ConnID Identifiant de connection
 * @param[in,out] Message Pointeur sur un SMSMessage valide
 * @return
 *   SMSNormal
 *   SMSError
 *   SMSLinkLost
 *   SMSNoVideosms
 *   SMSInvalidParameter
 *   SMSFileEmpty
 *   SMSFileNotFound
 *   SMSTooLong
 *   SMSNoCarrier
 *   SMSQuotaExceeded
 *   SMSUnauthorizedCarrier
 *   SMSInvalidIPAddress
 *   SMSUnrecognizedAddress
 *   SMSInvalidDate
 *   SMSThrottleError
 */
smsRet _STDCALL SMS_WapPush( smsConnID ConnID,
    SMSMessage* Message,
    WAPData* WapData,
    char PrivateReference[SMS_SEND_PRIVATE_REF_MAX_LEN],
    char PrivateData[SMS_SEND_PRIVATE_DATA_MAX_LEN] );

/*-----*/

/**
 * Renvoi le status d'un sms
 * @param[in] ConnID Identifiant de connection
 * @param[in,out] Message Pointeur sur une structure SMSMessage
 * @return
 *   SMSNormal
 *   SMSError
 */
smsRet _STDCALL SMS_ReadStatus( smsConnID ConnID,
    SMSMessage* Message,
    char PrivateReference[SMS_SEND_PRIVATE_REF_MAX_LEN],
    char PrivateData[SMS_SEND_PRIVATE_DATA_MAX_LEN]);

/*-----*/

/**
 * Renvoi un par un tous les SMS émis ou reçus par la passerelle pour l'utilisateur coura
 * @param[in] ConnID Identifiant de connection
 * @param[in] MsgType Type de message à récupérer
 * @param[in] MsgOptions Données demandées pour le message
 * @param[in,out] Message Pointeur sur une structure SMSMessage
 * @param[in,out] PrivateRef Chaine qui recevra la référence privée du message
 * @param[in,out] PrivateData Chaine qui recevra les données privées du message
 * @return
 *   SMSNormal
 *   SMSError
 */
smsRet _STDCALL SMS_GetProcessedMessage( smsConnID ConnID,
    smsUInt32 MsgType,
    smsUInt32 MsgOptions,
    smsDateTime FromDate,
    SMSMessage* Message,
    char PrivateReference[SMS_SEND_PRIVATE_REF_MAX_LEN],
    char PrivateData[SMS_SEND_PRIVATE_DATA_MAX_LEN] );

/*-----*/
```

Example 8-1 Cont'd on next page

Exemple 8-1 (Cont.) Fichier d'include smsApi1.h

```
/**
 * Récupère le prochain message disponible pour l'utilisateur actuellement connecté.
 * @param[in] ConnID Identifiant de connexion
 * @param[in,out] Message Pointeur sur une structure SMSMessage
 * @param[in,out] PrivateRef Chaine qui recevra la référence privée du message
 * @param[in,out] PrivateData Chaine qui recevra les données privées du message
 * @return
 *   SMSNormal
 *   SMSError
 */
smsRet _STDCALL_ SMS_GetMessage( smsConnID ConnID,
    SMSMessage* Message );

/*-----*/

#ifdef __cplusplus
}
#endif

#endif /* __MCTEL_SMSAPI_H__ */
```

9

Exemples

9.1 Programme d'envoi Visual Basic

Ce programme VB déclenche l'envoi d'un SMS et permet de vérifier en temps réel le statut de sa transmission.

Exemple 9-1 Programme d'envoi de SMS VB

```
' -----  
'  
' Programme de démonstration de l'utilisation de l'API SMS  
' -----  
'  
' Copie la chaine 'Src' (format VB) dans le tableau de caractères 'Dest' (format C)  
' en ajoutant le '0' de fin de chaine  
Private Sub CopyVBStringToCString(ByVal Src As String, ByRef Dest() As Byte)  
  
    Dim i  
    Lg = Len(Src)  
    For i = 1 To Lg Step 1  
        cc = Mid(Src, i, 1)  
        Dest(i) = Asc(cc)  
    Next  
    Dest(Lg + 1) = 0  
End Sub  
  
' -----  
' Exemple d'envoi d'un SMS  
' -----  
Private Sub BtEnvoyer_Click()  
  
    Dim Ret As smsRet  
    Dim ConnID As Long 'smsConnID  
  
    Trace.Caption = "Connexion en cours..."  
    Ret = 0  
  
    ' -----  
    ' Connexion au serveur de SMS  
    ' -----  
    GatewayAddress = "smssmpp.mctel.fr"  
    GatewayAddressLen = Len(GatewayAddress)  
    Port = 80  
    Ret = SMS_ConnectGateway(ConnID, _  
                             GatewayAddress, _  
                             GatewayAddressLen, _  
                             Port)  
  
    If (Ret <> SMSNormal) Then  
        Trace.Caption = "Erreur SMS_ConnectGateway : " + Str(Ret)  
        MsgBox (Trace.Caption)  
        Exit Sub  
    End If  
    Trace.Caption = "SMS_ConnectGateway OK"
```

Example 9-1 Cont'd on next page

Example 9–1 (Cont.) Programme d'envoi de SMS VB

```
'-----  
' Identification  
'-----  
CompanyID = "MCTELWEBSMS"  
CompanyPwd = "1yZa87w"  
UserID = UCase(Compte.Text)  
UserPwd = MotDePasse.Text  
  
SoftVersion = "DEMO"  
Dim NbrCredits As Long  
Dim LoginOptFlags As Long  
Ret = SMS_LoginGateway(ConnID, _  
                        CompanyID, CompanyPwd, _  
                        UserID, UserPwd, _  
                        SoftVersion, _  
                        LoginOptFlags, _  
                        NbrCredits)  
If (Ret <> SMSNormal) Then  
    Trace.Caption = "Erreur SMS_LoginGateway : " + Str(Ret)  
    MsgBox (Trace.Caption)  
    Exit Sub  
End If  
Trace.Caption = "SMS_LoginGateway OK"  
  
'-----  
' Envoi du message  
'-----  
Dim Msg As String  
Dim MsgStruct As SMSMessage  
  
NumDest = Numero.Text  
Msg = "Ceci est mon message de test à partir de VB."  
  
Call CopyVBStringToCString(NumDest, MsgStruct.called_number)  
MsgStruct.msg_length = Len(Msg)  
Call CopyVBStringToCString(Msg, MsgStruct.msg_data)  
MsgStruct.msg_options = 0  
MsgStruct.deferred_date = 0  
  
'MsgBox ("AVANT Send - Numéro : " + NumDest)  
Ret = SMS_Send(ConnID, MsgStruct)  
If (Ret <> SMSNormal) Then  
    Trace.Caption = "Erreur SMS_Send : " + Str(Ret)  
    MsgBox (Trace.Caption)  
Else  
    RefDernierMessage.Caption = MsgStruct.msg_reference_number  
    Trace.Caption = "SMS_Send OK" + Str(RefDernierMessage.Caption)  
End If  
  
'-----  
' Fermeture de la session  
'-----  
Ret = SMS_LogoutGateway(ConnID)  
If (Ret <> SMSNormal) Then  
    Trace.Caption = "Erreur SMS_LogoutGateway"  
    MsgBox (Trace.Caption)  
    Exit Sub  
End If  
Trace.Caption = "SMS_LogoutGateway OK"
```

Example 9–1 Cont'd on next page

Example 9–1 (Cont.) Programme d'envoi de SMS VB

```

' -----
' Déconnexion
' -----
Ret = SMS_DisconnectGateway(ConnID)
If (Ret <> SMSNormal) Then
    Trace.Caption = "Erreur SMS_DisconnectGateway : " + Str(Ret)
    MsgBox (Trace.Caption)
    Exit Sub
End If
Trace.Caption = "SMS_DisconnectGateway OK"

Trace.Caption = "Envoi terminé."
End Sub

' -----
' Exemple de récupération de statut
' -----
Private Sub BtStatut_Click()
    If RefDernierMessage.Caption = "?" Then
        Exit Sub
    End If

    Dim Ret As smsRet
    Dim ConnID As Long

    Trace.Caption = "Connexion en cours..."
    Ret = 0

    ' -----
    ' Connexion au serveur de SMS
    ' -----
    GatewayAddress = "smsmmp.mctel.fr"
    GatewayAddressLen = Len(GatewayAddress)
    Port = 80
    Ret = SMS_ConnectGateway(ConnID, _
        GatewayAddress, _
        GatewayAddressLen, _
        Port)

    If (Ret <> SMSNormal) Then
        Trace.Caption = "Erreur SMS_ConnectGateway : " + Str(Ret)
        MsgBox (Trace.Caption)
        Exit Sub
    End If
    Trace.Caption = "SMS_ConnectGateway OK"

    ' -----
    ' Identification
    ' -----
    CompanyID = "MCTELWEBSMS"
    CompanyPwd = "0yZt78w"
    UserID = UCase(Compte.Text)
    UserPwd = MotDePasse.Text
    SoftVersion = "DEMO"
    Dim NbrCredits As Long
    Ret = SMS_LoginGateway(ConnID, _
        CompanyID, CompanyPwd, _
        UserID, UserPwd, _
        SoftVersion, _
        NbrCredits)

    If (Ret <> SMSNormal) Then
        Trace.Caption = "Erreur SMS_LoginGateway : " + Str(Ret)
        MsgBox (Trace.Caption)
        Exit Sub
    End If
    Trace.Caption = "SMS_LoginGateway OK"

```

Example 9–1 Cont'd on next page

Exemple 9-1 (Cont.) Programme d'envoi de SMS VB

```
'-----  
' Vérification du statut  
'-----  
Dim MsgStruct As SMSMessage  
  
MsgStruct.msg_reference_number = RefDernierMessage.Caption  
  
Ret = SMS_ReadStatus(ConnID, MsgStruct)  
If (Ret <> SMSNormal) Then  
    Trace.Caption = "Erreur SMS_ReadStatus : " + Str(Ret)  
    MsgStatus.Caption = Trace.Caption  
    MsgBox (Trace.Caption)  
Else  
    Trace.Caption = "SMS_ReadStatus OK"  
    MsgStatus.Caption = Str(MsgStruct.sms_state)  
End If  
  
'-----  
' Fermeture de la session  
'-----  
Ret = SMS_LogoutGateway(ConnID)  
If (Ret <> SMSNormal) Then  
    Trace.Caption = "Erreur SMS_LogoutGateway"  
    MsgBox (Trace.Caption)  
    Exit Sub  
End If  
Trace.Caption = "SMS_LogoutGateway OK"  
  
'-----  
' Déconnexion  
'-----  
Ret = SMS_DisconnectGateway(ConnID)  
If (Ret <> SMSNormal) Then  
    Trace.Caption = "Erreur SMS_DisconnectGateway : " + Str(Ret)  
    MsgBox (Trace.Caption)  
    Exit Sub  
End If  
Trace.Caption = "SMS_DisconnectGateway OK"  
  
Trace.Caption = "Demande terminée."  
End Sub  
  
Private Sub Form_Load()  
    'Compte.Text = "mon compte"  
    'MotDePasse.Text = "mon mot de passe"  
End Sub
```

Index

A

Alphabet • 3–5

C

C

link • 2–1

CCITT

Recommandations • v

D

DLL • 2–1

E

Erreurs • 3–6

G

Géolocalisation • 3–2, 5–17

O

OpenVMS

link • 2–2

P

Programmation • 2–1

S

SMPP

temps • 3–8

SMS

encodage alphabet • 3–5

format • 3–5

type • 3–4

SMSMessage • 3–1

SMSReport • 3–2

SMSSession • 3–1

SMSUserLocation • 3–2

SMS_ConnectGateway • 4–2

SMS_DisconnectGateway • 4–9

SMS_EnquireLink • 4–8

SMS_GetAccountRecord • 7–5

SMS_GetLocation • 5–17

SMS_GetMessage • 5–15

SMS_GetProcessedMessage • 7–2

SMS_GetSessionData • 6–6

SMS_GetSessionInfo • 6–4

SMS_LoginGateway • 4–4

SMS_LogoutGateway • 4–7

SMS_Read • 6–2

SMS_ReadStatus • 5–12

SMS_Reply • 6–10

SMS_Send • 5–2

SMS_UpdateSessionData • 6–8

SMS_WapPush • 5–7

Structure

SMSMessage • 3–1

SMSReport • 3–2

SMSSession • 3–1

SMSUserLocation • 3–2

WapData • 3–3

T

Temps SMPP • 3–8

U

Unix

link • 2-2

versions • 2-1

V

Visual Basic

link • 2-1

Visual C++

link • 2-1

W

WapData • 3-3

Windows

environnement • 2-1