

Travail de diplôme 2002

Passerelle SMS

Dossier de programmation

Candidat
Etudes

Segalla Jacques
Technicien ET en
télécommunications

Professeurs

Michel Joss
Jean-François Joss

1 Sommaire

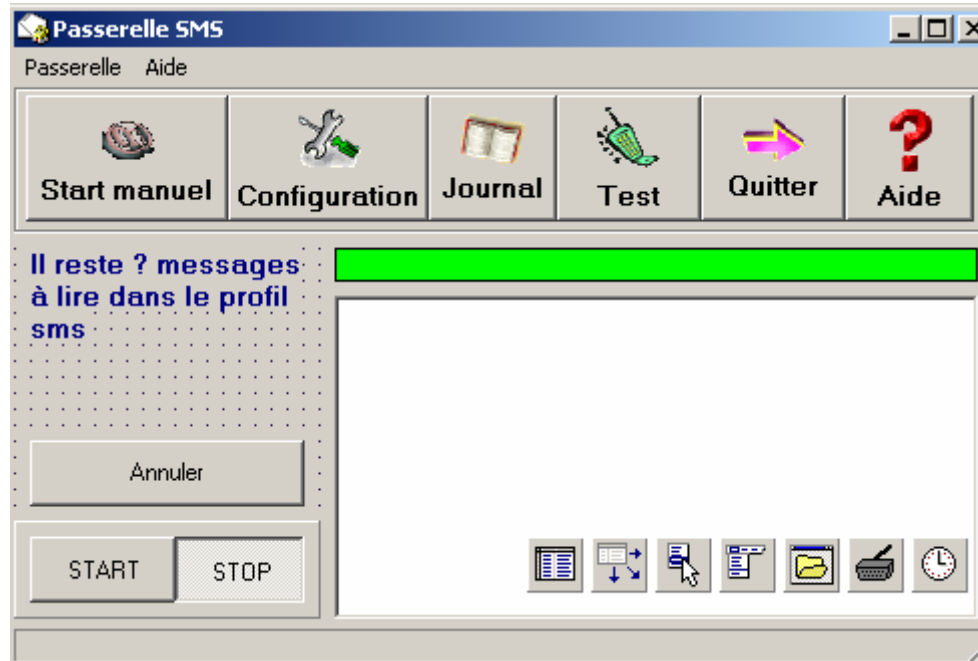
1	SOMMAIRE	1
2	FORMS	4
2.1	PRINCIPAL	4
2.2	CONFIGURATION	4
2.2.1	Général.....	4
2.2.2	Passerelle	5
2.2.3	Avancé.....	5
2.3	SELECTPORTCOM	6
2.4	IMPRIME	6
2.5	TEST.....	7
2.6	AIDE.....	7
2.7	APROPOS.....	7
3	CODES SOURCES	8
3.1	LIBRAIRIE STRUCTURE_U1	8
3.2	LIBRAIRIE NATEL_U1	10
3.2.1	<i>procedure TPrincipal.InitConfigIni;</i>	12
3.2.2	<i>procedure TPrincipal.InitConfigPort;</i>	12
3.2.3	<i>procedure TPrincipal.WriteConfigPort;</i>	13
3.2.4	<i>procedure TPrincipal.Journal.....</i>	14
3.2.5	<i>procedure TPrincipal.RenvoiCorrect.....</i>	14
3.2.6	<i>procedure TPrincipal.RenvoiErreur</i>	15
3.2.7	<i>function TPrincipal.VerifieMessage.....</i>	16
3.2.8	<i>procedure TPrincipal.EnvoiNatel</i>	18
3.2.9	<i>function TPrincipal.TrameReponse</i>	19
3.2.10	<i>procedure TPrincipal.EnvoiModem.....</i>	20
3.2.11	<i>procedure TPrincipal.ControleBD.....</i>	21
3.2.12	<i>procedure TPrincipal.DecortiqueDateMail.....</i>	22
3.2.13	<i>procedure TPrincipal.ControleBoiteMail.....</i>	22
3.2.14	<i>procedure TPrincipal.FormCreate</i>	23
3.2.15	<i>procedure TPrincipal.QuitterPopUpClick.....</i>	24
3.2.16	<i>procedure TPrincipal.ConfigurerCOMClick.....</i>	24
3.2.17	<i>procedure TPrincipal.ConfigurerPasserellePopUpClick.....</i>	24
3.2.18	<i>procedure TPrincipal.VoirJournalPopUpClick.....</i>	24
3.2.19	<i>procedure TPrincipal.ConfigurerlapasserelleMainMenuClick.....</i>	24
3.2.20	<i>procedure TPrincipal.VoirlejournalMainMenuClick</i>	24
3.2.21	<i>procedure TPrincipal.QuitterMainMenuClick.....</i>	25
3.2.22	<i>procedure TPrincipal.AproposMainMenuClick.....</i>	25
3.2.23	<i>procedure TPrincipal.PrincipalMainMenuClick.....</i>	25
3.2.24	<i>procedure TPrincipal.TestdelenvoiPopUpClick</i>	25
3.2.25	<i>procedure TPrincipal.TestdelenvoiMainMenuClick.....</i>	25
3.2.26	<i>procedure TPrincipal.SBQuitterClick.....</i>	25
3.2.27	<i>procedure TPrincipal.SBTestClick.....</i>	26
3.2.28	<i>procedure TPrincipal.SBJournalClick.....</i>	26
3.2.29	<i>procedure TPrincipal.SBConfigurationClick.....</i>	26
3.2.30	<i>procedure TPrincipal.DefiniTimer.....</i>	26
3.2.31	<i>procedure TPrincipal.SBStartClick</i>	26

3.2.32	<i>procedure</i> <i>TPrincipal.SBAideClick</i>	26
3.2.33	<i>procedure</i> <i>TPrincipal.BAnnulerClick</i>	27
3.2.34	<i>procedure</i> <i>TPrincipal.TScruteTimer</i>	27
3.2.35	<i>procedure</i> <i>TPrincipal.SBLanceClick</i>	28
3.2.36	<i>procedure</i> <i>TPrincipal.SBStopClick</i>	29
3.2.37	<i>function</i> <i>TPrincipal.Shell</i>	29
3.2.38	<i>procedure</i> <i>TPrincipal.FormDestroy</i>	29
3.2.39	<i>procedure</i> <i>TPrincipal.REInfoChange</i>	29
3.3	LIBRAIRIE MODEM_U1.....	30
3.3.1	<i>function</i> <i>WaitForString</i>	30
3.3.2	<i>procedure</i> <i>QuitteTropErreurs</i>	31
3.3.3	<i>function</i> <i>AttendOK</i>	31
3.3.4	<i>function</i> <i>DeConnectSMSC</i>	31
3.3.5	<i>function</i> <i>ConnectSMSC</i>	32
3.4	LIBRAIRIE CONFIG_U1.....	34
3.4.1	<i>procedure</i> <i>TConfiguration.BConfigComClick</i>	36
3.4.2	<i>procedure</i> <i>TConfiguration.BOkClick</i>	36
3.4.3	<i>procedure</i> <i>TConfiguration.CHBDiffereClick</i>	36
3.4.4	<i>procedure</i> <i>TConfiguration.GRenvoiClick</i>	36
3.4.5	<i>procedure</i> <i>TConfiguration.BRechercheClick</i>	37
3.4.6	<i>procedure</i> <i>TConfiguration.ChangeEnSecondes</i>	39
3.4.7	<i>procedure</i> <i>TConfiguration.TBJoursChange</i>	39
3.4.8	<i>procedure</i> <i>TConfiguration.TBHeuresChange</i>	40
3.4.9	<i>procedure</i> <i>TConfiguration.TBMinutesChange</i>	40
3.4.10	<i>procedure</i> <i>TConfiguration.TBSecondesChange</i>	40
3.4.11	<i>procedure</i> <i>TConfiguration.BAideClick</i>	40
3.4.12	<i>procedure</i> <i>TConfiguration.CBOperateurChange</i>	40
3.4.13	<i>procedure</i> <i>TConfiguration.CBNumOperateurChange</i>	41
3.4.14	<i>procedure</i> <i>TConfiguration.FormCreate</i>	41
3.4.15	<i>procedure</i> <i>TConfiguration.CBActiveEffaceClick</i>	41
3.5	LIBRAIRIE UCP_U1.....	41
3.5.1	<i>function</i> <i>AsciiToIA5</i>	42
3.5.2	<i>procedure</i> <i>UCPInitChampsHeaderOper</i>	42
3.5.3	<i>procedure</i> <i>UCPInitChampsAckNack</i>	43
3.5.4	<i>function</i> <i>UCPChecksum</i>	43
3.5.5	<i>function</i> <i>UCPConstruitTrame30</i>	43
3.5.6	<i>procedure</i> <i>UCPDecodeHeader</i>	45
3.5.7	<i>procedure</i> <i>UCPDecodeAck30</i>	45
3.5.8	<i>procedure</i> <i>UCPDecodeNack30</i>	46
3.6	LIBRAIRIE UDP_U1.....	47
3.6.1	<i>function</i> <i>DecToBin</i>	47
3.6.2	<i>function</i> <i>HexToBin</i>	48
3.6.3	<i>function</i> <i>HexCharToInt</i>	48
3.6.4	<i>function</i> <i>HexCharToBin</i>	48
3.6.5	<i>function</i> <i>Pow</i>	49
3.6.6	<i>function</i> <i>BinStrToInt</i>	49
3.6.7	<i>function</i> <i>DecodeSMS7Bit</i>	49
3.6.8	<i>function</i> <i>DecodeSMSText</i>	50
3.6.9	<i>function</i> <i>ReverseStr</i>	51
3.7	LIBRAIRIE SELECTPORTCOM_U1.....	51

3.7.1	<i>procedure TSelectPortCom.BOKClick</i>	52
3.7.2	<i>procedure TSelectPortCom.FormActivate</i>	52
3.7.3	<i>procedure TSelectPortCom.RGCOMClick</i>	52
3.7.4	<i>procedure TSelectPortCom.BAnnuleClick</i>	52
3.8	LIBRAIRIE TEST_U1.....	53
3.8.1	<i>procedure TTest.BAideClick</i>	53
3.8.2	<i>procedure TTest.BAnnuleClick</i>	54
3.8.3	<i>procedure TTest.MMessageKeyPress</i>	54
3.8.4	<i>procedure TTest.MMessageChange</i>	54
3.8.5	<i>procedure TTest.BEnvoieClick</i>	54
3.8.6	<i>procedure TTest.FormActivate</i>	54
3.9	LIBRAIRIE PRINT_U1.....	55
3.10	LIBRAIRIE AIDE_U1.....	55
3.10.1	<i>procedure TAide.ChargeAide</i>	56
3.10.2	<i>procedure TAide.OKBtnClick</i>	56
3.10.3	<i>procedure TAide.TabControl1Change</i>	56
3.10.4	<i>procedure TAide.FormActivate</i>	57
3.11	LIBRAIRIE APROPOS_U1.....	57
3.11.1	<i>procedure TAPropos.OKButtonClick</i>	57
3.12	PROGRAMME CLIQUE_OUI.....	58
3.13	PRINCIPAL: TPRINCIPAL.....	60
3.14	CONFIGURATION: TCONFIGURATION.....	66
3.15	SELECTPORTCOM: TSELECTPORTCOM.....	77
3.16	IMPRIME: TIMPRIME.....	78
3.17	TEST: TTEST.....	84
3.18	AIDE: TAIDE.....	86
3.19	APROPOS: TAPROPOS.....	87

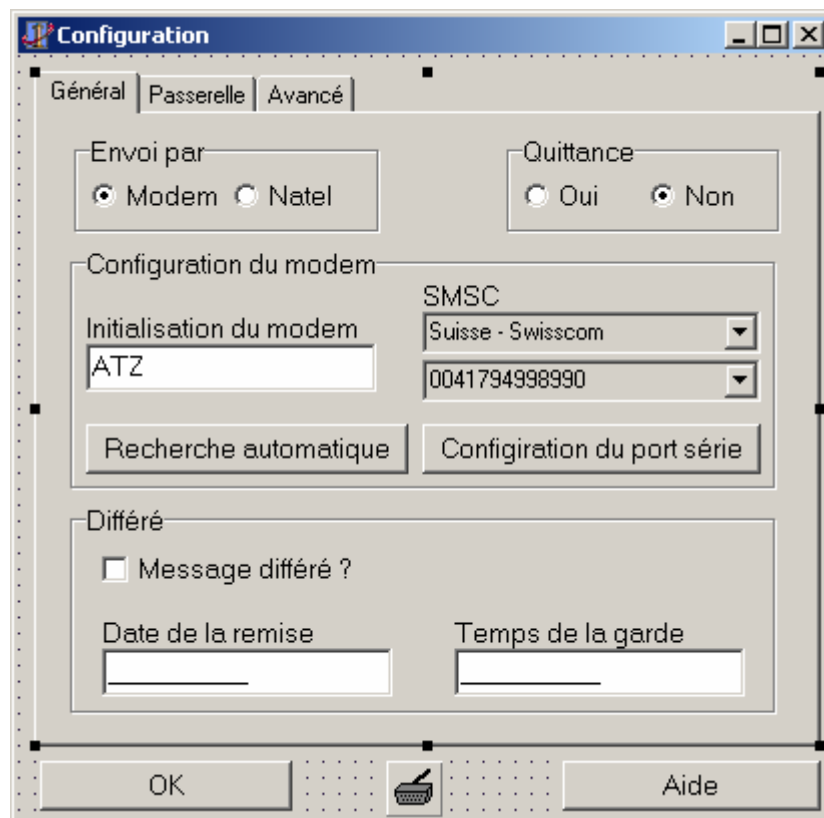
2 Forms

2.1 Principal

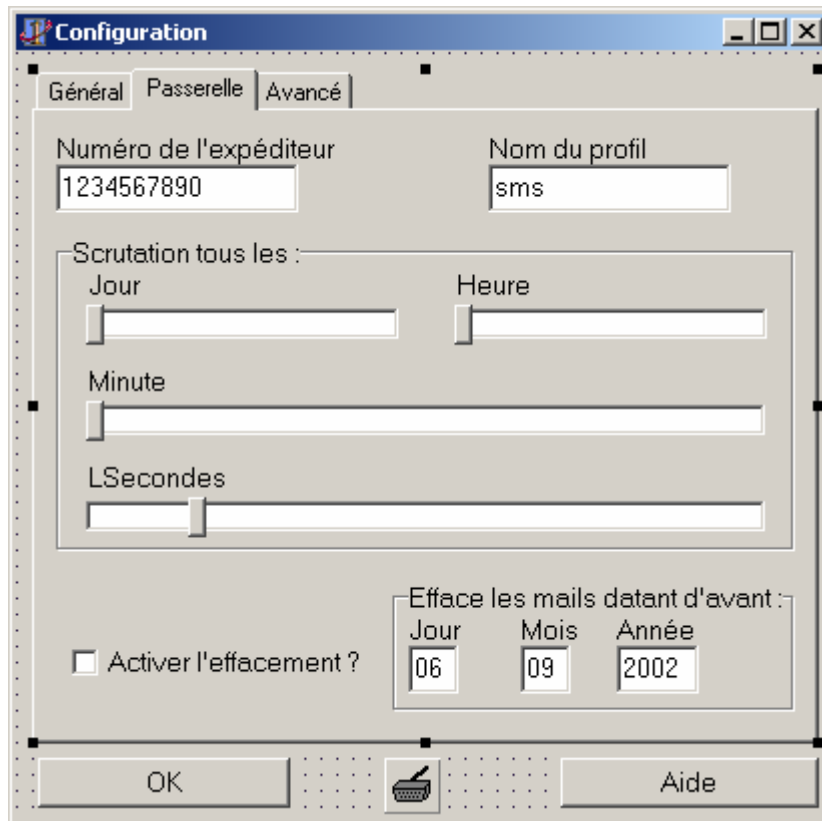


2.2 Configuration

2.2.1 Général



2.2.2 Passerelle

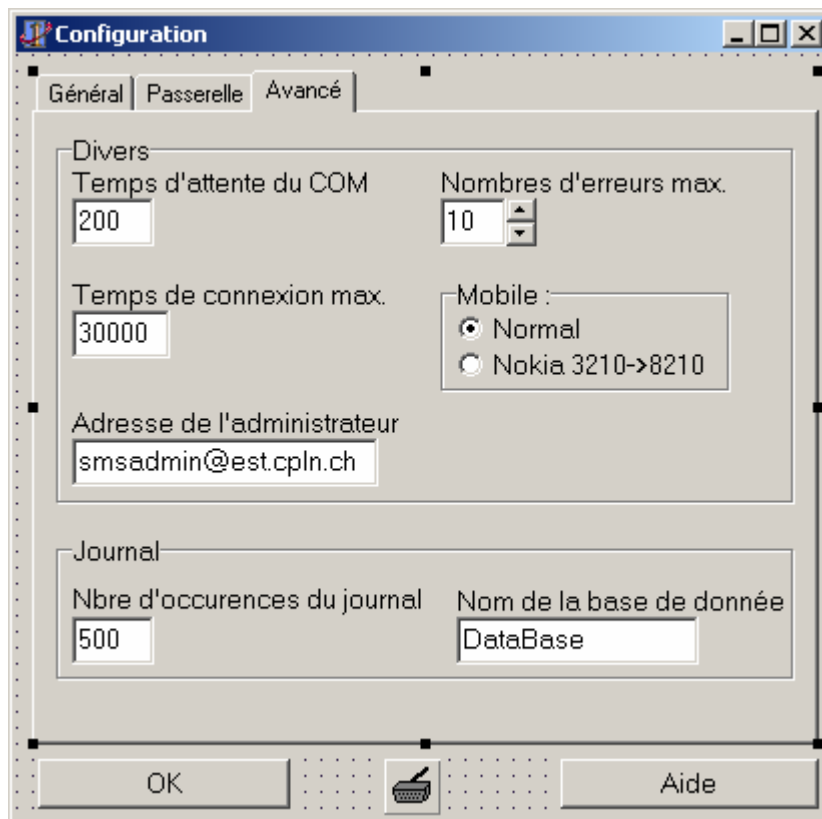


The screenshot shows the 'Configuration' dialog box with the 'Passerelle' tab selected. The window title is 'Configuration'. The tabs are 'Général', 'Passerelle', and 'Avancé'. The 'Passerelle' tab contains the following fields:

- Numéro de l'expéditeur: 1234567890
- Nom du profil: sms
- Scrutation tous les :
 - Jour: []
 - Heure: []
 - Minute: []
 - LSecondes: []
- Efface les mails datant d'avant :
 - Jour: 06
 - Mois: 09
 - Année: 2002
- Activer l'effacement ?

Buttons: OK, [Icon], Aide.

2.2.3 Avancé

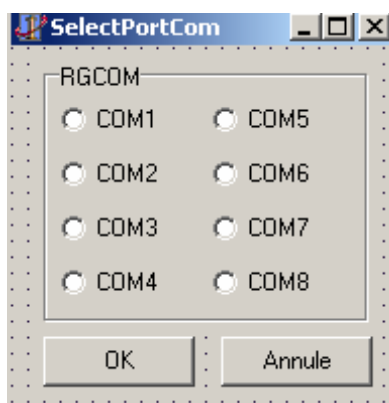


The screenshot shows the 'Configuration' dialog box with the 'Avancé' tab selected. The window title is 'Configuration'. The tabs are 'Général', 'Passerelle', and 'Avancé'. The 'Avancé' tab contains the following fields:

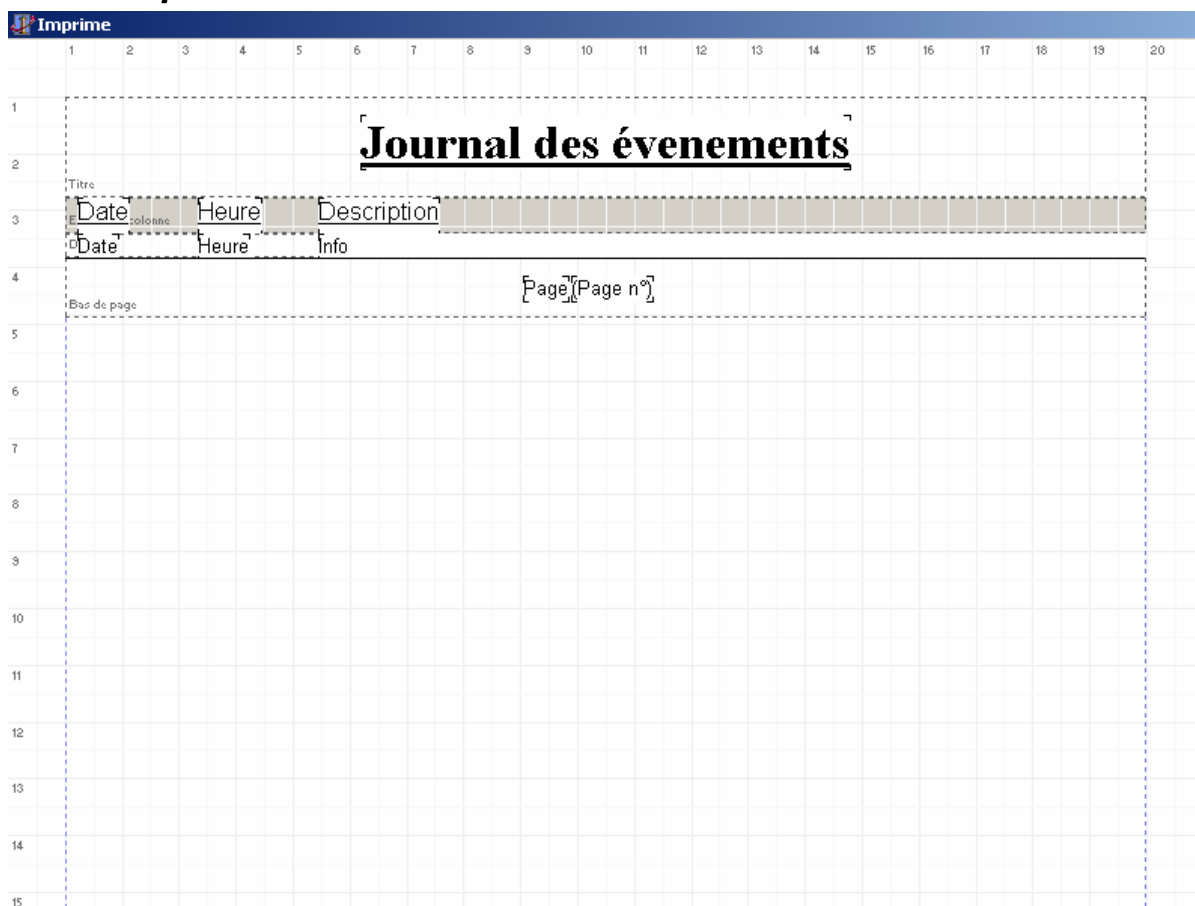
- Divers
 - Temps d'attente du COM: 200
 - Nombres d'erreurs max.: 10
 - Temps de connexion max.: 30000
 - Mobile :
 - Normal
 - Nokia 3210->8210
- Adresse de l'administrateur: smsadmin@est.cpln.ch
- Journal
 - Nbre d'occurences du journal: 500
 - Nom de la base de donnée: DataBase

Buttons: OK, [Icon], Aide.

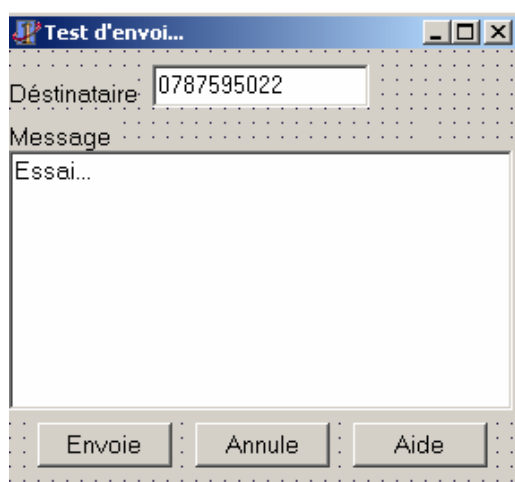
2.3 SelectPortCom



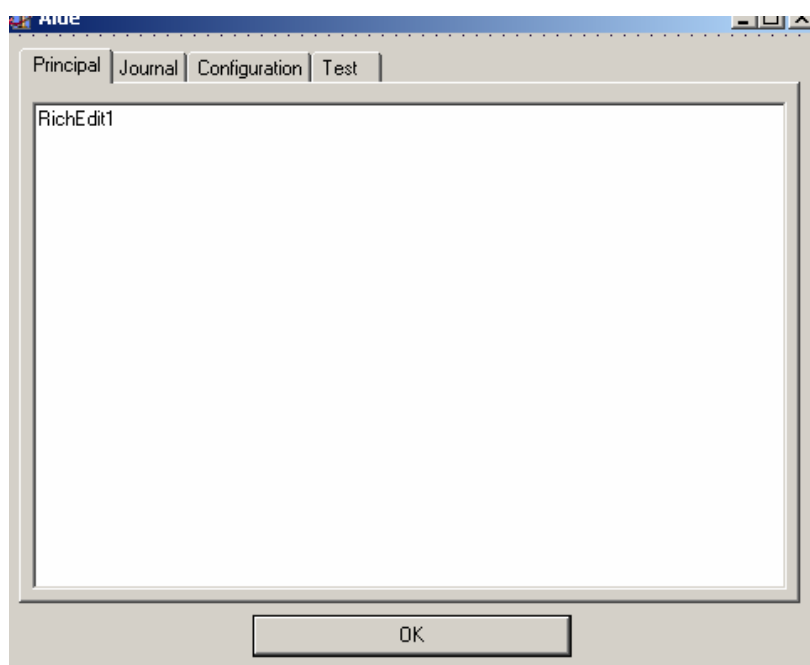
2.4 Imprime



2.5 Test



2.6 Aide



2.7 APropos



3 Codes sources

3.1 Librairie Structure_u1

```

=====
VERSION      : 1.0
COMPILEUR   : Borland Delphi 5.0 sous Windows 2000 PRO
CREE PAR    : Segalla Jacques
DATE        : 11 septembre 2002
EFFECTUE    : Structures pour le programme "Passerelle SMS".
              Si par la suite d'autre type de trame devraient être utilisées,
              il suffit de d'ajouter les structures ici.
=====
}
unit Structure_u1;

interface
uses email;
type
//Trame PDU pour l'envoi depuis un natel
t_PDUEnvoi = Record
  LengthInfo      : String[2]; //Taille des informations du SMSC
  Octet1          : String[2]; //1er octet du SMS envoyé
  TPReference     : String[2]; //Indique qui définis la référence
  AdresseLength   : String[2]; //Taille du numéro de natel
  TypeAdresse    : String[2]; //Format international du numéro du natel
  PhoneNumber     : String[16]; //Numéro du natel encodé
  TPPID          : String[2]; //Identificateur du protocole TP-ID
  TPDCS          : String[2]; //Type de codage des données
  TPPeriode      : String[2]; //Période de validité du SMS
  TPUserDataLength : String[2]; //Taille des données en octets
  TPUserData     : String //Message du SMS codé sur 7 bits
end;

//En-tête du protocole UCP
t_Header = Record
  TRN : String[2]; //Numéro de la transaction
  LEN : String[5]; //Longueur du message entre STX et ETX
  _OR : String[1]; //O = Opération, R = Résultat
  OT  : String[2]; //Type d'opération (voir structures ci-dessous)
  Check : String[2] //Somme de contrôle (Checksum)
end;

//Trame EMI avec Type Operation 30
t_Oper30 = Record
  Adc : String[16]; //Adresse du destinataire
  OAdC : String[16]; //Adresse de l'expéditeur
  AC : String; //Authentification du code d'origine
  NRq : String[1]; //Quittance (1 ou 0)
  NAd : String; //Adresse de la quittance
  NPID : String[4]; //Nombre pour la quittance (0100 pour les mobiles)
  DD : String[1]; //Message différé (1 ou 0)
  DDT : String[10]; //Date et heure du message à différé
  VP : String[10]; //Période de garde du message différé
  AMsg : String //Texte du SMS codé selon la table IA5
end;

//Message système (pour trame positive de type 30)
t_SM = Record
  AdC : String[16]; //Adresse du destinataire
  SCTS : String[12] //Date et heure du SMSC
end;

//Trame EMI positive de type 30
t_Ack30 = Record
  Ack : String[1]; //Acquittement positif
  MVP : String[10]; //Validité de période modifié

```

```
SM : t_SM // "System message"
end;

//Trame EMI négative de type 30
t_Nack30 = Record
  Nack : String[1]; //Acquittement négatif
  EC : String[2]; //Code d'erreur
  SM : String //Message d'erreur selon le code d'erreur
end;

//Réponses possibles du modem (Compatible Hayes)
t_ReponseModem = Record
  Ok : String[2]; //Commande bien reçue et exécutée
  Connect : String[7]; //Modem connecté
  Ring : String[4]; //Signal de sonnerie détecté
  No_carrier : String[10]; //Pas de porteuse
  Error : String[5]; //Erreur dans les commandes
  No_dialtone : String[11]; //Pas de tonalité
  Busy : String[4]; //Ligne occupée
  No_answer : String[9] //Pas de réponse
end;

//Enumère les diverses erreurs possibles
t_Erreur = (ErrCarFaux, //Il y a d'autres caractères que des nombres
  ErrTropCar, //Plus de 160 caractères
  ErrPieceJointe, //Il y a une ou plusieurs pièces jointes
  ErrAutorisation, //Pas l'autorisation d'envoyer des SMS
  ErrSMSC, //Erreur lors de l'envoi du SMS via le SMSC
  ErrSMSCNack, //Lorsque le SMSC renvoi un Nack
  ErrSMSCCheck, //Si le Checksum reçus n'est pas correct
  ErrCritique); //Lorsque l'application a fait trop d'erreurs

var
  Structure_ul_v_mail : TEmail; //Occurence de TEmail
  Structure_ul_v_reponse : t_ReponseModem; //Réponses du modem définis
  //dans modem.ini
  Structure_ul_v_adresse : String; //Chemin de répertoire ou se trouve l'exe
  Structure_ul_v_Transaction : Word; //Numéro de la transaction en nombre

  Structure_ul_v_Tete : t_Header; //En-tête de la trame UCP
  Structure_ul_v_Oper : t_Oper30; //Données de la trame de type 30
  Structure_ul_v_Ack : t_Ack30; //Données de la trame positive de type 30
  Structure_ul_v_Nack : t_Nack30; //Données de la trame négatives de type 30

  Structure_ul_v_ListeCom : array[1..8] of Boolean; //Liste des ports série
  Structure_ul_v_CptErreur : Word; //Compteur pour le nombre maximum d'erreurs
  //choisi

implementation

end.
```

3.2 Librairie Natel_u1

```

=====
VERSION      : 1.0
COMPILEUR   : Borland Delphi 5.0 sous Windows 2000 PRO
CREE PAR    : Segalla Jacques
DATE        : 11 septembre 2002
EFFECTUE    : Passerelle SMS. Permet à des utilisateurs ou à des logiciels
              d'application, d'envoyer des messages SMS par le biais de la
              messagerie. Les diverses unités utilisées pour ce programme sont
              déclarées dans le 'uses' de la partie implementation.
=====
}
unit Natel_u1;

interface

uses
  Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
  PiconeBarreTache, Menus, StdCtrls, CPort, ExtCtrls, ToolWin, Structure_u1,
  ComCtrls, email, Buttons, Db, DBTables, Grids, DBGrids, ShellAPI, Gauges;

type
  TPrincipal = class(TForm)
    PopupMenu: TPopupMenu;
    ConfigurerCOM: TMenuItem;
    ComPort1: TComPort;
    ConfigurerPasserellePopUp: TMenuItem;
    VoirJournalPopUp: TMenuItem;
    QuitterPopUp: TMenuItem;
    TScrute: TTimer;
    OpenDialog: TOpenDialog;
    StatusBar1: TStatusBar;
    MainMenu: TMainMenu;
    Fichier1: TMenuItem;
    Aidel: TMenuItem;
    ConfigurerlapasserelleMainMenu: TMenuItem;
    VoirlejournalMainMenu: TMenuItem;
    QuitterMainMenu: TMenuItem;
    PrincipalMainMenu: TMenuItem;
    AproposMainMenu: TMenuItem;
    TestdelenvoiPopUp: TMenuItem;
    TestdelenvoiMainMenu: TMenuItem;
    ToolBar1: TToolBar;
    SBQuitter: TSpeedButton;
    SBTest: TSpeedButton;
    SBJournal: TSpeedButton;
    SBConfiguration: TSpeedButton;
    SBStart: TSpeedButton;
    SBAide: TSpeedButton;
    BAnnuler: TButton;
    DataSource: TDataSource;
    Table: TTable;
    Gauge: TGauge;
    REInfo: TRichEdit;
    LInfo1: TLabel;
    Panel: TPanel;
    SBLance: TSpeedButton;
    SBStop: TSpeedButton;
    LInfo2: TLabel;
    LInfo3: TLabel;

    //Mes procedures et fonctions
    procedure InitConfigIni;
    procedure InitConfigPort;
    procedure WriteConfigPort;
    procedure Journal({IN} Texte : String);           //Texte du journal
    procedure RenvoiCorrect;                         //Numéro de l'action
  end;

```

```

procedure RenvoiErreur({IN} Erreur : t_Erreur); //Type d'erreur
procedure EnvoiNatel;
procedure EnvoiModem;
procedure ControleBD;
procedure ControleBoiteMail;
procedure DefiniTimer;
procedure DecortiqueDateMail(var Annee,           //Année de la date du mail
                             Mois,             //Mois de la date du mail
                             Jour : String); //Jour de la date du mail

function Shell({IN} CommandLine : String; //
                {IN} Sw : Cardinal) //
                : Longword; //Handle de la fenêtre lancée

function VerifieMessage : Boolean;
function TrameReponse({IN} Trame : String) //Trame reçue complète
                : String; //Trame de retour

procedure FormCreate(Sender: TObject);
procedure QuitterPopUpClick(Sender: TObject);
procedure ConfigurerCOMClick(Sender: TObject);
procedure ConfigurerPasserellePopUpClick(Sender: TObject);
procedure VoirJournalPopUpClick(Sender: TObject);
procedure ConfigurerlapasserelleMainMenuClick(Sender: TObject);
procedure VoirlejournalMainMenuClick(Sender: TObject);
procedure QuitterMainMenuClick(Sender: TObject);
procedure AproposMainMenuClick(Sender: TObject);
procedure PrincipalMainMenuClick(Sender: TObject);
procedure TestdelenvoiPopUpClick(Sender: TObject);
procedure TestdelenvoiMainMenuClick(Sender: TObject);
procedure SBQuitteClick(Sender: TObject);
procedure SBTestClick(Sender: TObject);
procedure SBJournalClick(Sender: TObject);
procedure SBConfigurationClick(Sender: TObject);
procedure SBStartClick(Sender: TObject);
procedure SBAideClick(Sender: TObject);
procedure BAnnulerClick(Sender: TObject);
procedure TScruteTimer(Sender: TObject);
procedure SBLanceClick(Sender: TObject);
procedure SBStopClick(Sender: TObject);
procedure FormDestroy(Sender: TObject);
procedure REInfoChange(Sender: TObject);
private
  { Déclarations privées }
public
  { Déclarations publiques }
end;

var
  Principal      : TPrincipal;

implementation
uses Config_ul, UCP_ul, udp_ul, modem_ul, APropos_ul, IniFiles, Aide_ul,
      Test_ul, Print_ul;

//Variables globales à la librairie
var
  v_tampon      : String; //Variable utilisée pour "décortiquer" les 2
                          //trames de retour
  v_debut_trame : Boolean; //Détection le début de la trame de retour
  v_Handle      : HWND; //Handle utilisé pour le programme Clique_Oui
{$R *.DFM}

```

3.2.1 procedure TPrincipal.InitConfigIni;

```

/*****
//      Initialise les réponses du modem selon le fichier modem.ini
/*****
procedure TPrincipal.InitConfigIni;
var
  ConfigIni  : TIniFile; //Contiendra le fichier Modem.ini

begin
  ConfigIni := TIniFile.Create(Structure_ul_v_adresse + '\fichiers\Modem.ini');
  Structure_ul_v_Reponse.OK := ConfigIni.ReadString('Reponses', 'ok', 'OK');
  Structure_ul_v_Reponse.CONNECT := ConfigIni.ReadString('Reponses', 'connect',
    'CONNECT');
  Structure_ul_v_Reponse.RING := ConfigIni.ReadString('Reponses', 'ring',
    'RING');
  Structure_ul_v_Reponse.NO_CARRIER := ConfigIni.ReadString('Reponses',
    'no_carrier', 'NO CARRIER');
  Structure_ul_v_Reponse.ERROR := ConfigIni.ReadString('Reponses', 'error',
    'ERROR');
  Structure_ul_v_Reponse.NO_DIALTONE := ConfigIni.ReadString('Reponses',
    'no_dialtone', 'NO DIALTONE');
  Structure_ul_v_Reponse.BUSY := ConfigIni.ReadString('Reponses', 'busy',
    'BUSY');
  Structure_ul_v_Reponse.NO_ANSWER := ConfigIni.ReadString('Reponses',
    'no_answer', 'NO ANSWER');
  ConfigIni.Free;
end;

```

3.2.2 procedure TPrincipal.InitConfigPort;

```

/*****
//      Lis la configuration du port COM selon le fichier configcom.ini
/*****
procedure TPrincipal.InitConfigPort;
var
  ConfigPort : TIniFile; //Contiendra le fichier ConfigCom.ini
  Port, Vitesse, Stop, Donnees, Parite, RTSin, DTRIn : Word;

begin
  ConfigPort := TIniFile.Create(Structure_ul_v_adresse +
    '\fichiers\configcom.ini');
  Port := ConfigPort.ReadInteger('Principal', 'Port Com', 1);
  Vitesse := ConfigPort.ReadInteger('Principal', 'Vitesse', 7);
  Stop := ConfigPort.ReadInteger('Principal', 'Bit Stop', 2);
  Donnees := ConfigPort.ReadInteger('Principal', 'Bit donnee', 4);
  parite := ConfigPort.ReadInteger('Principal', 'Parite', 1);
  RTSin := ConfigPort.ReadInteger('Flux', 'RTSin', 1);
  DTRIn := ConfigPort.ReadInteger('Flux', 'DTRIn', 1);
  ComPort1.FlowControl.OutCTSFlow := ConfigPort.ReadBool('Flux', 'CTSOut',
    False);
  ComPort1.FlowControl.OutDSRFlow := ConfigPort.ReadBool('Flux', 'DSROut',
    False);
  ComPort1.FlowControl.XonXoffOut := ConfigPort.ReadBool('Flux', 'Xon/Xoff Out',
    False);
  ComPort1.FlowControl.XonXoffIn := ConfigPort.ReadBool('Flux', 'Xon/Xoff In',
    False);
  ConfigPort.Free;

  //Conversion des types pour le port COM
  case Port of
    1 : ComPort1.Port := com1;  2 : ComPort1.Port := com2;
    3 : ComPort1.Port := com3;  4 : ComPort1.Port := com4;
    5 : ComPort1.Port := com5;  6 : ComPort1.Port := com6;
    7 : ComPort1.Port := com7;  8 : ComPort1.Port := com8;
  end;
  case Vitesse of

```

```

1 : ComPort1.BaudRate := br110;      2 : ComPort1.BaudRate := br300;
3 : ComPort1.BaudRate := br600;      4 : ComPort1.BaudRate := br1200;
5 : ComPort1.BaudRate := br2400;     6 : ComPort1.BaudRate := br4800;
7 : ComPort1.BaudRate := br9600;     8 : ComPort1.BaudRate := br14400;
9 : ComPort1.BaudRate := br19200;    10 : ComPort1.BaudRate := br38400;
11 : ComPort1.BaudRate := br56000;   12 : ComPort1.BaudRate := br57600;
13 : ComPort1.BaudRate := br115200;

end;
case Stop of
  1 : ComPort1.StopBits := sbOneStopBit;
  2 : ComPort1.StopBits := sbOne5StopBits;
  3 : ComPort1.StopBits := sbTwoStopBits;
end;
case Donnees of
  1 : ComPort1.DataBits := dbFive;    2 : ComPort1.DataBits := dbSix;
  3 : ComPort1.DataBits := dbSeven;   4 : ComPort1.DataBits := dbEight;
end;
case parite of
  1 : ComPort1.Parity.Bits := prNone;  2 : ComPort1.Parity.Bits := prOdd;
  3 : ComPort1.Parity.Bits := prEven;  4 : ComPort1.Parity.Bits := prMark;
end;
case RTSin of
  1 : ComPort1.FlowControl.ControlRTS := rtsDisable;
  2 : ComPort1.FlowControl.ControlRTS := rtsEnable;
  3 : ComPort1.FlowControl.ControlRTS := rtsHandshake;
  4 : ComPort1.FlowControl.ControlRTS := rtsToggle;
end;
case DTRin of
  1 : ComPort1.FlowControl.ControlDTR := dtrDisable;
  2 : ComPort1.FlowControl.ControlDTR := dtrEnable;
  3 : ComPort1.FlowControl.ControlDTR := dtrHandshake;
end;
end;

```

3.2.3 procedure TPrincipal.WriteConfigPort;

```

//*****
//          Ecrit la configuration du port COM dans Configcom.ini
//*****
procedure TPrincipal.WriteConfigPort;
var
  ConfigPort : TIniFile; //Contiendra le fichier ConfigCom.ini
  Port, Vitesse, Stop, Donnees, Parite, RTSin, DTRin : Word;

begin
  //Initialisation au cas ou il y aurait un problème.
  Port := 1; Vitesse := 7; Stop := 1; Donnees := 4; Parite := 1;
  RTSin := 3; DTRin := 2;

  //Conversion des types
  case ComPort1.Port of
    com1 : Port := 1; com2 : Port := 2; com3 : Port := 3; com4 : Port := 4;
    com5 : Port := 5; com6 : Port := 6; com7 : Port := 7; com8 : Port := 8;
  end;
  case ComPort1.BaudRate of
    br110   : Vitesse := 1; br300   : Vitesse := 2;
    br600   : Vitesse := 3; br1200  : Vitesse := 4;
    br2400  : Vitesse := 5; br4800  : Vitesse := 6;
    br9600  : Vitesse := 7; br14400 : Vitesse := 8;
    br19200 : Vitesse := 9; br38400 : Vitesse := 10;
    br56000 : Vitesse := 11; br57600 : Vitesse := 12;
    br115200 : Vitesse := 13;
  end;
  case ComPort1.StopBits of
    sbOneStopBit : Stop := 1; sbOne5StopBits : Stop := 2;
    sbTwoStopBits : Stop := 3;
  end;
  case ComPort1.DataBits of

```

```

dbFive : Donnees := 1; dbSix : Donnees := 2;
dbSeven : Donnees := 3; dbEight : Donnees := 4;
end;
case ComPort1.Parity.Bits of
prNone : Parite := 1; prOdd : Parite := 2;
prEven : Parite := 3; prMark : Parite := 4;
prSpace : Parite := 5;
end;
case ComPort1.FlowControl.ControlRTS of
rtsDisable : RTSin := 1; rtsEnable : RTSin := 2;
rtsHandshake : RTSin := 3; rtsToggle : RTSin := 4;
end;
case ComPort1.FlowControl.ControlDTR of
dtrDisable : DTRin := 1; dtrEnable : DTRin := 2;
dtrHandshake : DTRin := 3;
end;

//Fichier configcom.ini
ConfigPort := TIniFile.Create(Structure_ul_v_adresse +
'\fichiers\Configcom.ini');
ConfigPort.WriteInteger('Principal', 'Port Com', Port);
ConfigPort.WriteInteger('Principal', 'Vitesse', Vitesse);
ConfigPort.WriteInteger('Principal', 'Bit Stop', Stop);
ConfigPort.WriteInteger('Principal', 'Bit donnee', Donnees);
ConfigPort.WriteInteger('Principal', 'Parite', Parite);
ConfigPort.WriteInteger('Flux', 'RTSIn', RTSin);
ConfigPort.WriteInteger('Flux', 'DTRIn', DTRin);
ConfigPort.WriteBool('Flux', 'CTSOut', ComPort1.FlowControl.OutCTSFlow);
ConfigPort.WriteBool('Flux', 'DSROut', ComPort1.FlowControl.OutDSRFlow);
ConfigPort.WriteBool('Flux', 'Xon/Xoff Out', ComPort1.FlowControl.XonXoffOut);
ConfigPort.WriteBool('Flux', 'Xon/Xoff In', ComPort1.FlowControl.XonXoffIn);
ConfigPort.Free;
end;

```

3.2.4 procedure TPrincipal.Journal

```

//*****
//          Ecris le fichier journal avec le texte reçus
//*****
procedure TPrincipal.Journal({IN} Texte : String); {Texte à insérer dans
le journal}
begin
{Se place au début de la table et insert la date, l'heure et le texte dans
les bons champs de la base de donnée}
Table.First;
Table.Insert;
Table.FieldName('Date').AsString := DateToStr(Date);
Table.FieldName('Heure').AsString := TimeToStr(Time);
//Met le texte reçus dans le champ Info
Table.FieldName('Info').AsString := Texte;
//Ecrit un enregistrement modifié dans la base de données
Table.Post;
end;

```

3.2.5 procedure TPrincipal.RenvoiCorrect

```

//*****
//  Envoi un mail et inscrit dans le journal que tout c'est passé correctement
//*****
procedure TPrincipal.RenvoiCorrect; //Numéro de l'action
begin
//Adresse mail de l'expéditeur sans 'SMTP:'
Structure_ul_v_mail.Recipient.Text :=
StringReplace(Structure_ul_v_mail.OrigAddress, 'SMTP:', '', [rfIgnoreCase]);
//Sujet du mail
Structure_ul_v_mail.Subject := 'Envoi de SMS';
//Texte du mail
Structure_ul_v_mail.Text := 'Votre mail du ' + Structure_ul_v_mail.DateRecvd +

```

```

    ' à été correctement envoyé';
//Ecris dans le journal
Principal.Journal('SMS de ' + Structure_ul_v_mail.Originator +
    ', correctement envoyé.');
```

//Envoi le mail d'accusé à l'expéditeur
 Structure_ul_v_mail.SendMail;

end;

3.2.6 procédure TPrincipal.RenvoiErreur

```

//*****
//      Envoi les mail d'erreur et increis les erreurs dans le journal
//*****
procedure TPrincipal.RenvoiErreur({IN} Erreur : t_Erreur); //Type d'erreur
begin
  Structure_ul_v_mail.Recipient.Text :=
  StringReplace(Structure_ul_v_mail.OrigAddress, 'SMTP:', '', [rfIgnoreCase]);

  //Décide du message selon le type d'erreur (Voir type Erreur)
  case Erreur of
    ErrCarFaux      : begin
      Structure_ul_v_mail.Subject := 'Erreur de syntaxe';
      Structure_ul_v_mail.Text := 'Erreur lors de l''envoi' +
        ' du mail du ' + Structure_ul_v_mail.DateRecvd + #13 +
        'Le numéro de mobile contient des caractères ' +
        'non autorisés';
      Journal('Erreur de syntaxe. Le message de ' +
        Structure_ul_v_mail.Originator + ' contient des ' +
        'caractères non autorisés dans le numéro de mobile.');
```

end;

```

    ErrTropCar      : begin
      {Utilise la méthode SetLongText, car si cette erreur se
      produit, alors le renvoie du mail se feras avec plus
      de 255 car. alors que dans les autres cas d'erreurs,
      moins de 255 car. sont utilisés.}
      Structure_ul_v_mail.Subject := 'Erreur du message';
      Structure_ul_v_mail.SetLongText(PChar('Erreur lors ' +
        'de l''envoi du mail du ' + Structure_ul_v_mail.DateRecvd
        + '.' + #13 + 'Le message contient plus de 160 ' +
        'caractères ou il est vide.));
      Journal('Erreur du message de ' +
        Structure_ul_v_mail.Originator + '. Le message ' +
        'contient plus de 160 caractères ou il est vide.');
```

end;

```

    ErrPieceJointe  : begin
      Structure_ul_v_mail.Subject := 'Erreur du message';
      Structure_ul_v_mail.Text := 'Erreur lors de l''envoi' +
        ' du mail du ' + Structure_ul_v_mail.DateRecvd + #13 +
        'Il y a une pièce jointe au message.';
      Journal('Erreur du message de ' +
        Structure_ul_v_mail.Originator + '. Il y a une pièce' +
        ' jointe au message.');
```

end;

```

    ErrAutorisation : begin
      Structure_ul_v_mail.Subject := 'Erreur du message';
      Structure_ul_v_mail.Text := 'Erreur lors de l''envoi' +
        ' du mail du ' + Structure_ul_v_mail.DateRecvd + #13 +
        'Vous n''êtes pas autorisé à envoyer des messages';
      Journal('Erreur du message. ' +
        Structure_ul_v_mail.Originator + ' n''est pas ' +
        'autorisé à envoyer des messages.');
```

end;

```

    ErrSMSC         : begin
      Structure_ul_v_mail.Subject := 'Erreur du SMSC';
      Structure_ul_v_mail.Text := 'Erreur lors de l''envoi' +
        ' du mail du ' + Structure_ul_v_mail.DateRecvd + #13 +
        'La trame reçue n''est pas correcte.';
      Journal('Erreur du SMSC. La trame reçue n''est ' +
```

```

        'pas correcte.');"
    end;
ErrSMSCNack      : begin
    Structure_ul_v_mail.Subject := 'Erreur du SMSC';
    Structure_ul_v_mail.Text := 'Erreur lors de l''envoi' +
    ' du mail du ' + Structure_ul_v_mail.DateRecvd + #13 +
    'Erreur : ' + Structure_ul_v_Nack.SM;
    Journal('Erreur du SMSC : ' + Structure_ul_v_Nack.SM);
    end;
ErrSMSCCheck    : begin
    Structure_ul_v_mail.Subject := 'Erreur du SMSC';
    Structure_ul_v_mail.Text := 'Erreur lors de l''envoi' +
    ' du mail du ' + Structure_ul_v_mail.DateRecvd + '.' +
    #13 + 'Le Checksum reçus n''est pas correct.';
    Journal('Erreur du SMSC. Le Checksum reçus n''est ' +
    'pas correct');
    end;
ErrCritique     : begin
    Structure_ul_v_mail.Recipient.Text :=
    Configuration.EAdresseAdmin.Text;
    Structure_ul_v_mail.Subject := 'Erreur critique';
    Structure_ul_v_mail.Text := 'L''application à dû ' +
    'redémarrée à cause de plusieurs erreurs';
    Principal.Journal('Erreur critique ! L''application ' +
    'à dû redémarrée à cause de plusieurs erreurs');
    end;
end; //case
//Envoi le mail d'erreur à l'expéditeur
Structure_ul_v_mail.SendMail;
end;

```

3.2.7 fonction TPrincipal.VerifieMessage

```

//*****
//          Effectue diverses vérification des messages
//*****
function TPrincipal.VerifieMessage : Boolean;
var
    Autorisations      : TextFile; //Fichier texte d'autorisations
    AdresseMailFichier : String;  //Adresse mail lu dans le fichier
    AdresseMailProgramme : String; //Adresse mail de l'expéditeur sans 'SMTP:'
    NumeroNatel       : String;  //Numéro de téléphone
    CarPlus           : Boolean;   //Identifie le caractère '+'
    Cpt               : Word;     //Compteur pour le numéro de natel

begin
    CarPlus := False;
    NumeroNatel := structure_ul_v_mail.Subject;

    //Il ne faut pas que le numéro soit vide
    if NumeroNatel = '' then
    begin
        RenvoiErreur(ErrCarFaux);
        Result := False;
        Structure_ul_v_mail.Subject := '';
        exit;
    end;

    //Test que ce soit bien des numéros
    //Remplace le '+' par '00' si il existe en début de numéro
    if NumeroNatel[1] = '+' then
    begin
        Delete(NumeroNatel, 1, 1);
        NumeroNatel := '00' + NumeroNatel;
        CarPlus := True;
    end;

    //Enlève tous les espaces qu'il peut y avoir dans le numéro

```

```

Cpt := 1;
while Cpt <= Length(NumeroNatel) do
begin
  if NumeroNatel[cpt] = ' ' then
  begin
    Delete(NumeroNatel, Cpt, 1);
    //Décrémente au cas ou il y aurais plusieurs espaces dans le numéro
    Dec(Cpt);
  end;
  Inc(Cpt);
end;

//Prends en compte le caractère '/'
if CarPlus then
begin
  if NumeroNatel[8] = '/' then
    Delete(NumeroNatel, 8, 1)
  else if NumeroNatel[7] = '/' then
    Delete(NumeroNatel, 7, 1);
end
else if NumeroNatel[4] = '/' then
  Delete(NumeroNatel, 4, 1);

{A partir de ce moment, il ne peut plus y avoir que des chiffres ou des points
dans le numéro de téléphone.}
Cpt := 1;
While Cpt <= Length(NumeroNatel) do
begin
  {Test si le numéro contient des points. Si oui, ils doivent être précédés
et suivis par un numéro.}
  if (NumeroNatel[Cpt] = '.') and (NumeroNatel[cpt-1] in ['0'..'9']) and
(NumeroNatel[cpt+1] in ['0'..'9']) then
  begin
    Delete(NumeroNatel, Cpt, 1);
    Dec(Cpt);
  end;

  //Tous les caractères doivent maintenant se trouver entre 0 et 9
  if not(NumeroNatel[Cpt] in ['0'..'9']) then
  begin
    RenvoiErreur(ErrCarFaux);
    Result := False;
    Structure_ul_v_mail.Subject := '';
    exit;
  end;
  Inc(Cpt);
end;
Structure_ul_v_mail.Subject := NumeroNatel;

{Test que la longueur du message ne soit pas plus grande que 160 caractères
ou qu'elle ne sois vide}
if (Length(Structure_ul_v_mail.Text) > 160) or
(Length(Structure_ul_v_mail.Text) = 0) then
begin
  RenvoiErreur(ErrTropCar);
  Result := False;
  Exit;
end;

//Vérifie qu'il n'y ai pas de pièces jointes
if Structure_ul_v_mail.Attachment.Text <> '' then
begin
  RenvoiErreur(ErrPieceJointe);
  Result := False;
  Exit;
end;

//Identification de l'utilisateur

```

```
//Essaie d'ouvrir le fichier autorisations
try
  AssignFile(Autorisations, Structure_u1_v_adresse +
    '\fichiers\autorisations.txt');
  Reset(Autorisations);
except
  on EInOutError do
  begin
    REInfo.Lines.Add('Fichier "Autorisations.txt" non trouvé.' + #13 +
      'Veuillez sélectionner le bon fichier...');
    OpenDialog.Execute;
    AssignFile(Autorisations, Structure_u1_v_adresse +
      '\fichiers\autorisations.txt');
    Reset(Autorisations);
  end;
end; //try..except

{Lis toutes les lignes du fichier jusqu'à la fin. Si il trouve une
correspondance, alors retourne Vrai. Sinon renvoie un mail d'erreur à
l'expéditeur.}
while not eof(Autorisations) do
begin
  Readln(Autorisations, AdresseMailFichier);
  //Ne prends pas en compte 'SMTP:' de l'adresse
  AdresseMailProgramme :=
  StringReplace(Structure_u1_v_mail.OrigAddress, 'SMTP:', '', [rfIgnoreCase]);

  if AnsiCompareText(AdresseMailProgramme, AdresseMailFichier) = 0 then
  begin
    Result := True;
    Exit;
  end;
end; //while

if eof(Autorisations) then
begin
  RenvoiErreur(ErrAutorisation);
  Result := False;
  Exit;
end;

//Renvoi vrai s'il n'y a pas eu d'erreurs
Result := True;
end;
```

3.2.8 procédure TPrincipal.EnvoiNatel

```
//*****
//          Crée une trame PDU pour l'envoi depuis un natel
//*****
procedure TPrincipal.EnvoiNatel;
var
  TramePDU : String;          //Trame PDU de STX à ETX
  Longueur : Real;           //Taille de la trame PDU
  PDU      : t_PDUEnvoi;      //Variable de type TPDUEnvoi

begin
  //Remplis tous les champs de la trame PDU
  PDU.LengthInfo := '00'; PDU.Octet1 := '11'; PDU.TPReference := '00';
  PDU.AdresseLength := IntToHex(Length(Structure_u1_v_mail.Subject), 2);
  PDU.TypeAdresse := '81'; PDU.TPDCS := '00'; PDU.TPPeriode := 'AA';
  PDU.TPPID := '00'; PDU.PhoneNumber := ReverseStr(Structure_u1_v_mail.Subject);
  PDU.TPUserData := DecodeSMSText(Trim(Structure_u1_v_mail.Text));
  PDU.TPUserDataLength := IntToHex(Length(pdu.TPUserData) DIV 2, 2);

  //Construis la trame PDU
  TramePDU := PDU.LengthInfo + PDU.Octet1 + PDU.TPReference + PDU.AdresseLength
    + PDU.TypeAdresse + PDU.PhoneNumber + PDU.TPPID + PDU.TPDCS +
```

PDU.TPPeriode + PDU.TPUserDataLength + PDU.TPUserData;

```

Longueur := (Length(TramePDU)/2)-1;
ComPort1.WriteStr('AT+CMGS='+ FloatToStr(Longueur) + #13);
//Attends 50ms pour être sûr que le modem puisse recevoir d'autre données
sleep(50);
//Envoie la trame PDU avec CTRL-Z
ComPort1.WriteStr(TramePDU + #26);
end;
```

3.2.9 fonction TPrincipal.TrameReponse

```

//*****
//      Trouve la trame de réponse du SMSC dans un flot de données
//*****
function TPrincipal.TrameReponse({IN} Trame : String) : String;
var
    PosStx : Word; //Position du caractère STX
    NbCara : Word; //Nombre de caractères de la trame reçus
    PosEtx : Word; //Position du caractère ETX
    Cpt    : Word; //Compteur

begin
    Cpt := 0;

    //Détecte si la 1ère trame est celle de retour
    if Trame[1] = 'R' then v_debut_trame := True;

    if not v_debut_trame then
    begin
        {Si il y a un STX dans la trame, prends la trame sans le 1er STX ni le
        dernier ETX.}
        if pos(#2, Trame) <> 0 then
        begin
            v_debut_trame := TRUE;
            Inc(Cpt);
            PosStx := Pos(#2, Trame);
            NbCara := Length(Trame) - PosStx - 1;
            v_tampon := Copy(Trame, PosStx + 1, NbCara)
        end;
    end;

    //Détecte si la trame se termine correctement
    if Pos(#3, Trame) <> 0 then
    begin
        Inc(Cpt);
    end;

    //Cpt=1 identifie la 1ère trame et Cpt=2 identifie la seconde
    if Cpt = 1 then
    begin
        Result := Copy(Trame, 2, Length(Trame) - 2);
        v_debut_trame := False;
    end
    //Ne prends que le 2ème message de la trame après le STX.
    else if Cpt = 2 then
    begin
        Trame := v_tampon;
        PosEtx := Pos(#3, Trame);
        Result := Copy(Trame, PosEtx + 2, Length(Trame) - PosEtx - 1);
        v_debut_trame := False;
    end
    else
        Result := '';
    end;
end;
```

3.2.10 procédure TPrincipal.EnvoiModem

```

/*****
//
// Se connecte avec le modem et envoi la trame
/*****
procedure TPrincipal.EnvoiModem;
var
  StrRetour   : String; //Chaîne reçus du SMSC
  TrameAck    : String; //Trame de retour refaite pour calculer le Checksum
  CheckCalc   : String; //Calcul du checksum de la trame TrameAck

begin
  //Attends que le modem soit connecté
  repeat until ConnectSMSC;

  //Envoi la trame de type 30 au SMSC via le port série
  ComPort1.WriteString(#02 + UCPConstruitTrame30 + Structure_ul_v_Tete.Check
    + #03);
  //Attends les trames de réponses du SMSC
  //Reçois la trame d'envoi en echo
  StrRetour := WaitForString(3000);
  //Reçois la trame de retour
  StrRetour := WaitForString(3000);
  //Identifie la trame de réponse pour être sûr
  StrRetour := TrameReponse(StrRetour);

  {La longueur de l'en-tête fais 15 caractères. Il faut donc au moins minimum
  16 caractères pour que la trame soit bonne.}
  if Length(StrRetour) < 16 then
  begin
    RenvoiErreur(ErrSMSC);
    REInfo.SelAttributes.Color := clRed;
    REInfo.Lines.Add('Trame reçue fausse...');
    DeConnectSMSC;
    Exit;
  end;

  //Met les bonnes valeurs dans les champs correspondant de la structure UCP
  UCPDecodeHeader(StrRetour);

  {Test si c'est une réponse. Si oui, regarde si c'est Ack ou Nack et agis
  en conséquence.}
  if (Structure_ul_v_Tete._OR = 'R') then
  begin
    //Si c'est un Ack...
    if StrRetour[15] = 'A' then
    begin
      UCPDecodeAck30(StrRetour);
      {Construit la trame de retour pour calculer le checksum. Ensuite le
      checksum calculé viens comparer au checksum reçus. Si ce ne sont pas
      les mêmes, alors execute l'erreur ErrSMSCCheck.}
      TrameAck := Structure_ul_v_Tete.TRN + '/' + Structure_ul_v_Tete.LEN +
        '/' + Structure_ul_v_Tete._OR + '/' + Structure_ul_v_Tete.OT
        + '/' + Structure_ul_v_Ack.Ack + '/' +
        Structure_ul_v_Ack.MVP + '/' + Structure_ul_v_Ack.SM.AdC +
        ':' + Structure_ul_v_Ack.SM.SCTS + '/';
      CheckCalc := UCPCheckSum(TrameAck);
      if AnsiCompareText(Structure_ul_v_Tete.Check, CheckCalc) <> 0 then
      begin
        RenvoiErreur(ErrSMSCCheck);
        DeConnectSMSC;
        Exit;
      end;
      //Si il n'y à pas d'erreur dans le Checksum alors...
      RenvoiCorrect;
      REInfo.SelAttributes.Color := clBlue;
      REInfo.Lines.Add('Le SMS a été correctement envoyé...');
    end
  end

```

```
//Si c'est un Nack...
else if StrRetour[15] = 'N' then
begin
  {Pas besoin de calculer le checksum, car si on a reçu un Nack, cela
  veut dire que le SMS n'a de toute façon pas été envoyé.}
  UCPDecodeNack30(StrRetour);
  RenvoiErreur(ErrSMSCNack);
  REInfo.SelAttributes.Color := clRed;
  REInfo.Lines.Add('Erreur lors de l''envoi du message.' + #13 + 'Nack ' +
  'reçus : ' + Structure_ul_v_Nack.SM);
end
else
  RenvoiErreur(ErrSMSC);
end
//Si c'est une trame d'envoi...
else if (Structure_ul_v_Tete._OR = 'O') then
begin
  RenvoiErreur(ErrSMSC);
  REInfo.SelAttributes.Color := clRed;
  REInfo.Lines.Add('ERREUR. Ne peut recevoir d''opération du SMSC...');
end
//Ou alors une autre trame... (Normalement impossible)
else
begin
  RenvoiErreur(ErrSMSC);
  REInfo.SelAttributes.Color := clRed;
  REInfo.Lines.Add('ERREUR. N''est ni une opération, ni une réponse ?!?!?');
end;

DeConnectSMSC;
end;
```

3.2.11 procédure TPrincipal.ControleBD

```
//*****
//          Contrôle le nombre d'occurrence dans la base de donnée
//*****
procedure TPrincipal.ControleBD;
var
  OccurencesPossibles : Integer; //Paramètre de la fenêtre 'Configuration'
  Cpt                  : Integer; //Compteur pour la boucle for

begin
  //Commence par la dernière occurrence
  Table.Last;
  OccurencesPossibles := StrToInt(Configuration.ENbreOccurrence.Text);

  {Si le nombre d'occurrences est plus grand que dans le paramètre "Nbre
  d'occurrence max.", alors on va effacer les occurrences en trop, en partant de
  la fin.}
  if Table.RecordCount > OccurencesPossibles then
    for Cpt := 1 to Table.RecordCount - OccurencesPossibles do
      Table.Delete;
end;
```

3.2.12 procedure TPrincipal.DecortiqueDateMail

```

/*****
//
//      Sort l'année, le mois et le jours d'une date d'un mail
/*****
procedure TPrincipal.DecortiqueDateMail(var Annee, Mois, Jour : String);
var
    Cpt    : Word; //Compteur

begin
    Annee := ''; Mois := ''; Jour := '';

    //Format de la date : AAAA/MM/JJ
    //Prends l'année
    Cpt := 1;
    While Cpt <= 4 do
    begin
        Annee := Annee + Structure_u1_v_mail.DateRecvd[Cpt];
        Inc(Cpt);
    end;

    //Prends le mois
    Inc(Cpt);
    While Cpt <= 7 do
    begin
        Mois := Mois + Structure_u1_v_mail.DateRecvd[Cpt];
        Inc(Cpt);
    end;

    //Prends le jour
    Inc(Cpt);
    While Cpt <= 10 do
    begin
        Jour := Jour + Structure_u1_v_mail.DateRecvd[Cpt];
        Inc(Cpt);
    end;
end;

```

3.2.13 procedure TPrincipal.ControleBoiteMail

```

/*****
//
//      Efface les mails qui sont trop anciens
/*****
procedure TPrincipal.ControleBoiteMail;
var
    Annee    : String; //Année de la date du mail
    Mois     : String; //Mois de la date du mail
    Jour     : String; //Jour de la date du mail
    Cpt      : Integer; //Compteur

begin
    Cpt := 0;

    //Ne lire que les mail non lu
    Structure_u1_v_mail.UnreadOnly := False;

    Structure_u1_v_mail.GetNextMessageId;
    Structure_u1_v_mail.ReadMail;

    //Test que la date du mail soit plus ancienne que ce qui a été paramétré
    while Structure_u1_v_mail.MessageId <> '' do
    begin
        DecortiqueDateMail(Annee, Mois, Jour);
        {Commence les tests. Si l'année est plus petite alors on peut l'effacer.
        Sinon, si l'année est la même on test le mois, et ainsi de suite...}
        if Annee < Configuration.EAnnee.Text then
        begin

```

```

Structure_ul_v_mail.DeleteMail;
Inc(Cpt);
end
else if Annee = Configuration.EAnnee.Text then
begin
if Mois < Configuration.EMois.Text then
begin
Structure_ul_v_mail.DeleteMail;
Inc(Cpt);
end
else if Mois = Configuration.EMois.Text then
if (Jour < Configuration.EJour.Text) or
(Jour = Configuration.EJour.Text) then
begin
Structure_ul_v_mail.DeleteMail;
Inc(Cpt);
end;
end;
end;
Structure_ul_v_mail.GetNextMessageId;
Structure_ul_v_mail.ReadMail;
end;

//Affiche les informations sur l'effacement des mails
REInfo.SelAttributes.Color := clBlack;
if Cpt = 0 then
REInfo.Lines.Add('Pas d'anciens messages.')
else
REInfo.Lines.Add(IntToStr(Cpt) + ' anciens messages effacés.');
```

end;

//*****

3.2.14 procedure TPrincipal.FormCreate

```

//*****
//                               A la création de la fenêtre principale
//*****
procedure TPrincipal.FormCreate(Sender: TObject);
begin
//Lance le programme qui permet de cliquer sur Oui sur les fenêtres Outlook
v_Handle := Shell('Clique_Oui.exe', 1);

Structure_ul_v_Transaction := 0;

{Chemin ou le fichier Natel.exe s'exécute. Est utilisé pour contrer une
erreur de MAPI qui fais changer le répertoire de l'exécutable.}
Structure_ul_v_adresse := ExtractFileDir(Application.ExeName);

//Crée une instance de TEmail
Structure_ul_v_mail := TEmail.Create(Self);

//Initialise la base de donnée pour le journal
Table.DatabaseName := Structure_ul_v_adresse + '\DataBase';
Table.TableName := 'database';
Table.Active := True;

//Lis les fichiers configcom.ini et modem.ini
InitConfigIni;
InitConfigPort;
end;
```

3.2.15 procedure TPrincipal.QuitterPopUpClick

```
/**
//
// Quitte le programme
//
procedure TPrincipal.QuitterPopUpClick(Sender: TObject);
begin
    SBQuitter.Click;
end;
```

3.2.16 procedure TPrincipal.ConfigurerCOMClick

```
/**
// Affiche la fenêtre de configuration du port série
//
procedure TPrincipal.ConfigurerCOMClick(Sender: TObject);
begin
    ComPort1.ShowSetupDialog;
end;
```

3.2.17 procedure TPrincipal.ConfigurerPasserellePopUpClick

```
/**
// Affiche la fenêtre de configuration
//
procedure TPrincipal.ConfigurerPasserellePopUpClick(Sender: TObject);
begin
    SBConfiguration.Click;
end;
```

3.2.18 procedure TPrincipal.VoirJournalPopUpClick

```
/**
// Affiche le journal
//
procedure TPrincipal.VoirJournalPopUpClick(Sender: TObject);
begin
    SBJournal.Click;
end;
```

3.2.19 procedure TPrincipal.ConfigurerlapasserelleMainMenuClick

```
/**
// Affichage de la fenêtre de configuration de la passerelle
//
procedure TPrincipal.ConfigurerlapasserelleMainMenuClick(Sender: TObject);
begin
    SBConfiguration.Click;
end;
```

3.2.20 procedure TPrincipal.VoirlejournalMainMenuClick

```
/**
// Affichage de la fenêtre du journal
//
procedure TPrincipal.VoirlejournalMainMenuClick(Sender: TObject);
begin
    SBJournal.Click;
end;
```

3.2.21 procedure TPrincipal.QuitterMainMenuClick

```
//*****  
//                               Quitte l'application passerelle  
//*****  
procedure TPrincipal.QuitterMainMenuClick(Sender: TObject);  
begin  
    SBQuitter.Click;  
end;
```

3.2.22 procedure TPrincipal.AproposMainMenuClick

```
//*****  
//                               Affiche la boîte 'A Propos'  
//*****  
procedure TPrincipal.AproposMainMenuClick(Sender: TObject);  
begin  
    APropos.Show;  
end;
```

3.2.23 procedure TPrincipal.PrincipalMainMenuClick

```
//*****  
//                               Affiche l'aide principale  
//*****  
procedure TPrincipal.PrincipalMainMenuClick(Sender: TObject);  
begin  
    SBAide.Click;  
end;
```

3.2.24 procedure TPrincipal.TestdelenvoiPopUpClick

```
//*****  
//                               Affichage de la fenêtre de test  
//*****  
procedure TPrincipal.TestdelenvoiPopUpClick(Sender: TObject);  
begin  
    SBTest.Click;  
end;
```

3.2.25 procedure TPrincipal.TestdelenvoiMainMenuClick

```
//*****  
//                               Affichage de la fenêtre de test  
//*****  
procedure TPrincipal.TestdelenvoiMainMenuClick(Sender: TObject);  
begin  
    SBTest.Click;  
end;
```

3.2.26 procedure TPrincipal.SBQuitterClick

```
//*****  
//                               Quitte l'application passerelle  
//*****  
procedure TPrincipal.SBQuitterClick(Sender: TObject);  
begin  
    DeConnectSMSC;  
    Application.Terminate;  
end;
```

3.2.27 procedure TPrincipal.SBTestClick

```
/**
// Affichage de la fenêtre de test
/**
procedure TPrincipal.SBTestClick(Sender: TObject);
begin
    Test.Show;
end;
```

3.2.28 procedure TPrincipal.SBJournalClick

```
/**
// Affichage de la fenêtre du journal
/**
procedure TPrincipal.SBJournalClick(Sender: TObject);
begin
    Imprime.Rapport.Preview;
end;
```

3.2.29 procedure TPrincipal.SBConfigurationClick

```
/**
// Affichage de la fenêtre de configuration de la passerelle
/**
procedure TPrincipal.SBConfigurationClick(Sender: TObject);
begin
    Configuration.Show;
end;
```

3.2.30 procedure TPrincipal.DefiniTimer

```
/**
// Remet le Timer en place selon que le programme tourne ou non
/**
procedure TPrincipal.DefiniTimer;
begin
    if SBLance.Down then
        TScrute.Enabled := True
    else
        TScrute.Enabled := False;
end;
```

3.2.31 procedure TPrincipal.SBStartClick

```
/**
// Lance manuellement le programme
/**
procedure TPrincipal.SBStartClick(Sender: TObject);
begin
    TScrute.OnTimer(Sender);
    DefiniTimer;
end;
```

3.2.32 procedure TPrincipal.SBAideClick

```
/**
// Affiche l'aide principale
/**
procedure TPrincipal.SBAideClick(Sender: TObject);
begin
    Aide.TabControll.TabIndex := 0;
    Aide.Show;
end;
```

3.2.33 procedure TPrincipal.BAnnulerClick

```

/*****
//
// Annule les actions du port série
//
*****/
procedure TPrincipal.BAnnulerClick(Sender: TObject);
begin
  Screen.Cursor := crDefault;
  try
    ComPort1.Close;
    StatusBar1.Panels[0].Text := 'Action annulée par l''utilisateur';
    REInfo.SelAttributes.Color := clRed;
    REInfo.Lines.Add('Action annulée par l''utilisateur');
  except
    REInfo.Lines.Add('Communication interrompu par l''utilisateur.');
```

3.2.34 procedure TPrincipal.TScruteTimer

```

/*****
// Procédure principale. C'est elle qui coordine la bonne marche du programme
//
*****/
procedure TPrincipal.TScruteTimer(Sender: TObject);
begin
  REInfo.Clear;
  //Contrôle du nombre d'occurences dans la base de donnée
  ControleBD;
  //Réinitialise le nombre d'erreurs que l'application peut faire
  Structure_ul_v_CptErreur := 0;
  //Vide tous les champs de v_mail
  Structure_ul_v_mail.Clear;
  //Arrête Timer1 en attendant la fin de la procédure
  TScrute.Enabled := False;
  Structure_ul_v_mail.DownloadFirst := True;
  //Quitte si il y a un ancien profil lancé
  if Structure_ul_v_mail.Logoff <> EMAIL_OK then
    begin
      REInfo.SelAttributes.Color := clRed;
      REInfo.Lines.Add('Fin de MAPI incorrect.');
```

```

LInfol.Caption := 'Il reste ' + IntToStr(Structure_ul_v_mail.CountUnread) +
    ' messages';
//Lis le 1er message
Structure_ul_v_mail.GetNextMessageId;
StatusBar1.Panels[0].Text := 'Lis le prochain message...';
REInfo.SelAttributes.Color := clBlack;
REInfo.Lines.Add('Lis le prochain message...');

if Structure_ul_v_mail.ReadMail <> EMAIL_OK then
begin
    REInfo.SelAttributes.Color := clRed;
    REInfo.Lines.Add('Lecture du message incorrect. ');
    DefiniTimer;
    Exit;
end;

//Vérification du format du mail
while VerifieMessage = False do
begin
    Gauge.Progress := Structure_ul_v_mail.CountUnread;
    LInfol.Caption := 'Il reste ' + IntToStr(Structure_ul_v_mail.CountUnread) +
        ' messages';
    Structure_ul_v_mail.GetNextMessageId;
    //Test de nouveau qu'il y ai encore des mails non-lus
    if Structure_ul_v_mail.CountUnread = 0 then
    begin
        ControleBoiteMail;
        StatusBar1.Panels[0].Text := 'Plus de SMS à envoyer...';
        REInfo.SelAttributes.Color := clBlack;
        REInfo.Lines.Add('Plus de SMS à envoyer...');
        DefiniTimer;
        Exit;
    end;
    //Lecture du message
    if Structure_ul_v_mail.ReadMail <> EMAIL_OK then
    begin
        REInfo.SelAttributes.Color := clRed;
        REInfo.Lines.Add('Lecture du message incorrect. ');
        DefiniTimer;
        Exit;
    end;
end;

//Envoi soit par modem, soit par natel
if Configuration.GREnvoi.ItemIndex = 0 then
    EnvoiModem
else if Configuration.GREnvoi.ItemIndex = 1 then
    EnvoiNatel;

end;
if Configuration.CBActiveEfface.Checked then
    //Efface les anciens mails
    ControleBoiteMail;
StatusBar1.Panels[0].Text := 'Plus de SMS à envoyer...';
REInfo.SelAttributes.Color := clBlack;
REInfo.Lines.Add('Plus de SMS à envoyer...');
DefiniTimer;
end;

```

3.2.35 procedure TPrincipal.SBLanceClick

```

//*****
//
// Lance automatiquement le programme
//*****
procedure TPrincipal.SBLanceClick(Sender: TObject);
begin
    TScrute.Enabled := True;
end;

```

3.2.36 procedure TPrincipal.SBStopClick

```

//*****
//
//          Stop le scrutage automatique
//*****
procedure TPrincipal.SBStopClick(Sender: TObject);
begin
    TScrute.Enabled := False;
end;

```

3.2.37 fonction TPrincipal.Shell

```

//*****
// Crée un processus et retourne son hProcess (Trouvé sur le forum ngscan.com)
//*****
function TPrincipal.Shell(CommandLine : String; Sw : Cardinal) : Longword;
var
    SInfo : STARTUPINFO;
    PInfo : PROCESS_INFORMATION;

begin
    ZeroMemory(@SInfo, SizeOf(SInfo));
    SInfo.CB := SizeOf(sinfo);
    SInfo.dwFlags := STARTF_USESHOWWINDOW;
    SInfo.wShowWindow := Sw;
    CreateProcess(nil, PChar(CommandLine), nil, nil, True, NORMAL_PRIORITY_CLASS,
        GetEnvironmentStrings, nil, SInfo, PInfo);
    Result := PInfo.hProcess;
    CloseHandle(pinfo.hThread);
end;

```

3.2.38 procedure TPrincipal.FormDestroy

```

//*****
//          A la fermeture du programme..
//*****
procedure TPrincipal.FormDestroy(Sender: TObject);
begin
    WriteConfigPort;
    //Ferme la base de donnée
    Table.Close;
    //Détruit l'instance et ferme le port série
    Structure_ul_v_mail.Free;
    //Ferme le port série
    ComPort1.Close;
    //Quitte le programme Clique_Oui.exe
    TerminateProcess(v_Handle, 0)
end;

```

3.2.39 procedure TPrincipal.REInfoChange

```

//*****
//          Permet de toujours voir les dernières lignes entrées dans le RichEdit
//*****
procedure TPrincipal.REInfoChange(Sender: TObject);
begin
    REInfo.Perform( EM_SCROLL, SB_LINEDOWN, 0);
end;

end.

```

3.3 Librairie Modem_u1

```

=====
VERSION      : 1.0
COMPILEUR   : Borland Delphi 5.0 sous Windows 2000 PRO
CREE PAR    : Segalla Jacques
DATE        : 11 septembre 2002
EFFECTUE    : Librairie modem
=====
}
unit Modem_u1;

interface
  function WaitForString(Temps : LongWord) : String;
  function DeConnectSMSC : Boolean;
  function ConnectSMSC : Boolean;

implementation
uses
  Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
  PiconeBarreTache, Menus, email, StdCtrls, CPort, ExtCtrls, Natel_u1,
  Config_u1, Structure_u1;

```

3.3.1 fonction WaitForString

```

//*****
// Fonction qui attends une réponse du port série pendant un temps donné
//*****
function WaitForString({IN} Temps : LongWord //Temps que la fonction attends
                       ) : String;          //Chîne reçue sur le port série

var
  t1      : LongWord; //Temps de départ
  DeltaT  : LongWord; //Diférence de temps (temps actuel - t1)
  Quittance : Boolean; //Indique le passage de caractères
  Cpt     : Word;     //Compteur

begin
  DeltaT := 0;
  t1 := GetTickCount;
  Quittance := False;

  //Tant que le temps reçus n'est pas écoulé, alors essayé de lire le port COM
  while (DeltaT < temps) and (not Quittance) do
  begin
    {Test si il y a un caratère dans le buffer. Si oui, alors lis le flot et
    renvois le résultat}
    if Principal.ComPort1.InputCount <> 0 then
    begin
      //Attends de recevoir toute la trame
      Principal.ComPort1.ReadStr(Result, 1024);
      Quittance := True;
    end;
    DeltaT := GetTickCount - t1;
    Sleep(StrToInt(Configuration.ETempsAttenteCOM.Text));
    //Donne la main au autres processus de Windows.
    Application.ProcessMessages;
  end;
  //Enlève les retour à la ligne pour pouvoir afficher dans le Memo
  for Cpt := 1 to Length(Result) do
    if (Result[Cpt] = #13) and (Result[Cpt+1] = #10) then
      Delete(Result, Cpt, 2);

  //Ecrit les trames reçus en vert
  Principal.REInfo.SelAttributes.Color := clGreen;
  Principal.REInfo.Lines.Add(result);
  Exit;
end;

```

3.3.2 procedure QuitteTropErreurs

```

/*****
//Si il y a trop d'erreur, alors l'indique à l'utilisateur et ferme le programme
/*****
procedure QuitteTropErreurs;
begin
  if Structure_ul_v_CptErreur >=
    StrToInt(Configuration.ENbreErreurPossible.Text) then
    begin
      Principal.REInfo.SelAttributes.Color := clRed;
      Principal.REInfo.Lines.Add('Trop d'erreurs.' + #13 + 'Vérifier si le ' +
        'modem est connecté et si il est bien ' +
        'configuré.' + #13 + 'Sinon, essayé d''éteindre'+
        ' puis de rallumer le modem.' + #13 + 'Le ' +
        'programme va quitter, veuillez le relancer...');
      Principal.RenvoiErreur(ErrCritique);
      Screen.Cursor := crDefault;
      DeConnectSMSC;
      //Attends ~7 secondes avant de quitter
      Sleep(7000);
      //Quitte le programme quoi qu'il se passe
      Halt;
    end;
  end;

```

3.3.3 function AttendOK

```

/*****
//
//          Test que la trame reçue contienne un "OK"
//
/*****
function AttendOK : Boolean; //True si la fonction à reçues 'OK'
var
  Ok : String; //Trame reçue

begin
  Ok := WaitForString(1000);
  if Pos(Structure_ul_v_Reponse.Ok, Ok) = 0 then
    begin
      Inc(Structure_ul_v_CptErreur);
      QuitteTropErreurs;
      Principal.REInfo.SelAttributes.Color := clRed;
      Principal.REInfo.Lines.Add('Ne renvoi pas OK!');
      Result := False;
    end
  else
    Result := True;
  end;

```

3.3.4 function DeConnectSMSC

```

/*****
//
//          Déconnexion du modem si il est connecté (Baisse le DTR)
//
/*****
function DeConnectSMSC : Boolean; //Renvoie TRUE si OK sinon FALSE
begin
  DeConnectSMSC := False;
  if (Principal.ComPort1.Connected) and (msRLSD in Principal.ComPort1.
    ModemSignals) then
    begin
      //Baisse le DTR
      Principal.ComPort1.SetDTR(False);
      Principal.REInfo.SelAttributes.Color := clBlack;
      Principal.REInfo.Lines.Add('Déconnecté du SMSC.');
```

//Attends un "NO CARRIER" dans la trame
if Pos(Structure_ul_v_Reponse.No_carrier, WaitForString(1000)) <> 0 **then**
 DeConnectSMSC := True
 end;

```

else
  DeConnectSMSC := False;
end
end;

```

3.3.5 fonction ConnectSMSC

```

//*****
//      Connexion au serveur SMSC distant avec contrôle des erreurs
//*****
function ConnectSMSC : Boolean; //Renvoi TRUE si OK, sinon FALSE
var
  Erreur      : String; //Message reçus du SMSC

begin
  ConnectSMSC := False;
  QuitteTropErreurs;

  //Essaie d'ouvrir le port COM
  Principal.ComPort1.Close;
  try
    Principal.StatusBar1.Panels[0].Text := 'Ouverture du port...';
    Principal.REInfo.SelAttributes.Color := clBlack;
    Principal.REInfo.Lines.Add('Ouverture du port...');
    Principal.ComPort1.Open;
  except
    Principal.StatusBar1.Panels[0].Text := 'Port COM déjà ouvert !';
    Principal.REInfo.SelAttributes.Color := clRed;
    Principal.REInfo.Lines.Add('Port COM déjà ouvert !');
    Exit;
  end;

  Principal.StatusBar1.Panels[0].Text := 'Le port de communication est ouvert';
  Principal.REInfo.SelAttributes.Color := clBlack;
  Principal.REInfo.Lines.Add('Le port de communication est ouvert');
  Principal.ComPort1.ClearBuffer(True, True);

  //Initialisation du modem
  repeat
    Principal.ComPort1.WriteStr(Configuration.EInit.Text + #13);
  until AttendOK;
  repeat
    Principal.ComPort1.WriteStr('ATE'#13);
  until AttendOK;
  repeat
    Principal.ComPort1.WriteStr('ATS7=60'#13);
  until AttendOK;

  //Appelle du SMSC
  Principal.ComPort1.WriteStr('ATDT' + Configuration.CBNumOperateur.Text + #13);
  Principal.StatusBar1.Panels[0].Text := 'Appelle du SMSC...';
  Principal.REInfo.SelAttributes.Color := clBlack;
  Principal.REInfo.Lines.Add('Appelle du SMSC...');
  Screen.Cursor := crHourGlass;

  //Attends x temps que la connexion se fasse avec le SMSC
  Erreur := WaitForString(StrToInt(Configuration.ETempsConnexion.Text));

  {Détecte les diverses erreurs possible. Si il y a une erreur, alors
  l'application va essayer de se reconnecter}
  //Signal de sonnerie
  if Pos(Structure_u1_v_Reponse.Ring, Erreur) <> 0 then
  begin
    Inc(Structure_u1_v_CptErreur);
    DeConnectSMSC;
    Principal.ComPort1.Close;
    Screen.Cursor := crDefault;
    Principal.StatusBar1.Panels[0].Text := 'Signal de sonnerie détecté.';
  end

```

```
Principal.REInfo.SelAttributes.Color := clRed;
Principal.REInfo.Lines.Add('Signal de sonnerie détecté.');
```

```
end
//Pas de tonalité sur la ligne
else if Pos(Structure_ul_v_Reponse.No_carrier, Erreur) <> 0 then
begin
  Inc(Structure_ul_v_CptErreur);
  DeConnectSMSC;
  Principal.ComPort1.Close;
  Screen.Cursor := crDefault;
  Principal.StatusBar1.Panels[0].Text := 'Pas de tonalité.';
  Principal.REInfo.SelAttributes.Color := clRed;
  Principal.REInfo.Lines.Add('Pas de tonalité.');
```

```
end
//Erreurs diverses du modem
else if Pos(Structure_ul_v_Reponse.Error, Erreur) <> 0 then
begin
  Inc(Structure_ul_v_CptErreur);
  DeConnectSMSC;
  Principal.ComPort1.Close;
  Screen.Cursor := crDefault;
  Principal.StatusBar1.Panels[0].Text := 'Erreur du modem.';
  Principal.REInfo.SelAttributes.Color := clRed;
  Principal.REInfo.Lines.Add('Erreur du modem.');
```

```
end
//Pas de tonalité
else if Pos(Structure_ul_v_Reponse.No_dialtone, Erreur) <> 0 then
begin
  Inc(Structure_ul_v_CptErreur);
  DeConnectSMSC;
  Principal.ComPort1.Close;
  Screen.Cursor := crDefault;
  Principal.StatusBar1.Panels[0].Text := 'Pas de tonalité.';
  Principal.REInfo.SelAttributes.Color := clRed;
  Principal.REInfo.Lines.Add('Pas de tonalité.');
```

```
end
//Ligne occupée
else if Pos(Structure_ul_v_Reponse.Busy, Erreur) <> 0 then
begin
  Inc(Structure_ul_v_CptErreur);
  DeConnectSMSC;
  Principal.ComPort1.Close;
  Screen.Cursor := crDefault;
  Principal.StatusBar1.Panels[0].Text := 'La ligne est occupée.';
  Principal.REInfo.SelAttributes.Color := clRed;
  Principal.REInfo.Lines.Add('La ligne est occupée.');
```

```
end
//Pas de réponse
else if Pos(Structure_ul_v_Reponse.No_answer, Erreur) <> 0 then
begin
  Inc(Structure_ul_v_CptErreur);
  DeConnectSMSC;
  Principal.ComPort1.Close;
  Screen.Cursor := crDefault;
  Principal.StatusBar1.Panels[0].Text := 'Pas de réponse du modem.';
  Principal.REInfo.SelAttributes.Color := clRed;
  Principal.REInfo.Lines.Add('Pas de réponse du modem.');
```

```
end
//Connexion au SMSC
else if Pos(Structure_ul_v_Reponse.Connect, Erreur) <> 0 then
begin
  Structure_ul_v_CptErreur := 0;
  ConnectSMSC := True;
  Principal.REInfo.SelAttributes.Color := clBlack;
  Principal.StatusBar1.Panels[0].Text := 'Le modem est connecté au SMSC.';
  Principal.REInfo.Lines.Add('Le modem est connecté au SMSC.');
```

```
Screen.Cursor := crDefault;
Exit; //Assure la sortie de la fonction
```

```

end
//Pas de réponse dans le temps donné
else if Erreur = '' then
begin
  Inc(Structure_ul_v_CptErreur);
  DeConnectSMSC;
  Principal.ComPort1.Close;
  Screen.Cursor := crDefault;
  Principal.StatusBar1.Panels[0].Text := 'Pas de réponse dans les ' +
    Configuration.ETempsConnexion.Text +
    'ms';

  Principal.REInfo.SelAttributes.Color := clRed;
  Principal.REInfo.Lines.Add('Pas de réponse dans les ' +
    Configuration.ETempsConnexion.Text + 'ms');
end
else
//Erreur inconnue de l'application
begin
  Inc(Structure_ul_v_CptErreur);
  Principal.ComPort1.Close;
  Screen.Cursor := crDefault;
  Principal.StatusBar1.Panels[0].Text := 'Erreur du modem inconnue.';
  Principal.REInfo.SelAttributes.Color := clRed;
  Principal.REInfo.Lines.Add('Erreur du modem inconnue.');
```

end;

```

ConnectSMSC := False;
Screen.Cursor := crDefault
end;

end.
```

3.4 Librairie Config_u1

```

{=====
VERSION      : 1.0
COMPILEUR    : Borland Delphi 5.0 sous Windows 2000 PRO
CREE PAR     : Segalla Jacques
DATE        : 11 septembre 2002
EFFECTUE    : Fenêtre de configuration de l'application Passerelle SMS
=====}
```

unit Config_u1;

interface

uses

```

Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
StdCtrls, ExtCtrls, Mask, CPort, ComCtrls, Buttons;
```

type

```

TConfiguration = class (TForm)
  BOk: TButton;
  CPAuto: TComPort;
  PCConfig: TPageControl;
  TSGeneral: TTabSheet;
  GRTemps: TGroupBox;
  LDateRemise: TLabel;
  LTempsGarde: TLabel;
  CHBDiffere: TCheckBox;
  MEdat: TMaskEdit;
  MEGarde: TMaskEdit;
  GRModem: TGroupBox;
  LInitModem: TLabel;
  LSMSC: TLabel;
  BConfigCom: TButton;
  EInit: TEdit;
  BRecherche: TButton;
  GRQuittance: TRadioGroup;
  GREnvoi: TRadioGroup;
```

```

TSPasserelle: TTabSheet;
LNumExpediteur: TLabel;
LProfil: TLabel;
EOriginator: TEdit;
EProfil: TEdit;
GBTemps: TGroupBox;
LJours: TLabel;
LHeures: TLabel;
LMinutes: TLabel;
TBJours: TTrackBar;
TBHeures: TTrackBar;
TBMinutes: TTrackBar;
TSAvance: TTabSheet;
GRAvance: TGroupBox;
LAttenteCOM: TLabel;
LErreurMax: TLabel;
ETempsAttenteCOM: TEdit;
ENbreErreurPossible: TEdit;
UpDown1: TUpDown;
BAide: TButton;
CBOperateur: TComboBox;
CBNumOperateur: TComboBox;
LTempsConnexion: TLabel;
ETempsConnexion: TEdit;
TBSecondes: TTrackBar;
LSecondes: TLabel;
LAdresseAdmin: TLabel;
EAdresseAdmin: TEdit;
GBJournal: TGroupBox;
LNbreOccurrences: TLabel;
LNomBD: TLabel;
ENbreOccurrence: TEdit;
ENomBD: TEdit;
RGMobile: TRadioGroup;
GBEffaceMail: TGroupBox;
EJour: TEdit;
EMois: TEdit;
EAnnee: TEdit;
LJour: TLabel;
LMois: TLabel;
LAnnee: TLabel;
CBActiveEfface: TCheckBox;
procedure BConfigComClick(Sender: TObject);
procedure BOKClick(Sender: TObject);
procedure CHBDiffereClick(Sender: TObject);
procedure GREnvoiClick(Sender: TObject);
procedure BRechercheClick(Sender: TObject);
procedure TBJoursChange(Sender: TObject);
procedure TBHeuresChange(Sender: TObject);
procedure TBMinutesChange(Sender: TObject);
procedure BAideClick(Sender: TObject);
procedure CBOperateurChange(Sender: TObject);
procedure CBNumOperateurChange(Sender: TObject);
procedure ChangeEnSecondes;
procedure FormCreate(Sender: TObject);
procedure TBSecondesChange(Sender: TObject);
procedure CBActiveEffaceClick(Sender: TObject);
private
    { Déclarations privées }
public
    { Déclarations publiques }
end;

var
    Configuration : TConfiguration;

implementation

```

uses Natel_ul, SelectPortCom_ul, modem_ul, Aide_ul, Structure_ul;

{\$R *.DFM}

3.4.1 procedure TConfiguration.BConfigComClick

```
//*****
//                               Ouvre la fenêtre de configuration du port série
//*****
procedure TConfiguration.BConfigComClick(Sender: TObject);
begin
    principal.ComPort1.ShowSetupDialog
end;
```

3.4.2 procedure TConfiguration.BOkClick

```
//*****
//                               Ferme la fenêtre de configuration
//*****
procedure TConfiguration.BOkClick(Sender: TObject);
begin
    Principal.LInfo3.Caption := EProfil.Text;
    Principal.Table.Active := False;
    Principal.Table.DatabaseName := Structure_ul_v_adresse + '\' + ENomBD.Text;
    Principal.Table.TableName := ENomBD.Text;
    Principal.Table.Active := True;
    Configuration.Close
end;
```

3.4.3 procedure TConfiguration.CHBDiffereClick

```
//*****
//                               Remplis les champs pour la différenciation des messages
//*****
procedure TConfiguration.CHBDiffereClick(Sender: TObject);
begin
    if CHBDiffere.Checked = True then
        begin
            MEdat.Enabled := True;
            MEGarde.Enabled := True;
            MEdat.Text := FormatDateTime('ddmmyy', Date) +
                FormatDateTime('hhmm', Time);
            MEGarde.Text := FormatDateTime('ddmmyy', Date + 1) +
                FormatDateTime('hhmm', Time);
        end
    else
        begin
            MEdat.Enabled := False;
            MEGarde.Enabled := False;
            MEdat.Text := '';
            MEGarde.Text := '';
        end
end;
```

3.4.4 procedure TConfiguration.GREnvoiClick

```
//*****
// Cache ou active certaines zones lors du changement d'envoi (modem ou natel)
//*****
procedure TConfiguration.GREnvoiClick(Sender: TObject);
begin
    if GREnvoi.ItemIndex = 1 then
        begin
            CBOperateur.Enabled := False;
            CBNumOperateur.Enabled := False;
            CHBDiffere.Enabled := False;
            EOriginator.Enabled := False;
            MEGarde.Enabled := True;
        end
    end;
```

```

    RGMobile.Enabled := False;
end
else
begin
    CBOperateur.Enabled := True;
    CBNumOperateur.Enabled := True;
    CHBDiffere.Enabled := True;
    EOriginator.Enabled := True;
    MEGarde.Enabled := False;
    RGMobile.Enabled := True;
end
end;

```

3.4.5 procedure TConfiguration.BRechercheClick

```

//*****
// Recherche automatique des modems ou natels connectés sur le port série
//*****
procedure TConfiguration.BRechercheClick(Sender: TObject);
const
    COM_1 = True; COM_2 = True; COM_3 = True; COM_4 = True;
    COM_5 = True; COM_6 = True; COM_7 = True; COM_8 = True;

var
    cpt : Byte;

begin
    Screen.Cursor := crHourGlass;
    Principal.ComPort1.Close;

    //Initialise la liste des port à Faux (désactivé)
    for cpt := 1 to 8 do
        Structure_u1_v_ListeCom[cpt] := False;

    {Test automatiquement chaque port COM pour savoir si un modem réponds à la
    commande AT envoyée par le port série. Tous les modems testés répondent à
    une vitesse de 115200 bits/s.}
    //Test du COM1
    CPAuto.Close;
    CPAuto.Port := com1;
    CPAuto.BaudRate := br115200;
    try
        CPAuto.Open;
        CPAuto.WriteString('AT' + #13);

        if Pos('OK', WaitForString(200)) <> 0 then
            begin
                Principal.REInfo.SelAttributes.Color := clBlack;
                Principal.REInfo.Lines.Add('Il y a un modem sur le port Com1');
                Structure_u1_v_ListeCom[1] := COM_1;
            end;
        except
        end;

    //Test du COM2
    CPAuto.Close;
    CPAuto.Port := com2;
    try
        CPAuto.Open;
        CPAuto.WriteString('AT' + #13);

        if Pos('OK', WaitForString(200)) <> 0 then
            begin
                Principal.REInfo.Lines.Add('Il y a un modem sur le port Com2');
                Structure_u1_v_ListeCom[2] := COM_2;
            end;
        except
        end;

```

```
//Test du COM3
CPAuto.Close;
CPAuto.Port := com3;
try
  CPAuto.Open;
  CPAuto.WriteStr('AT' + #13);

  if Pos('OK', WaitForString(200)) <> 0 then
    begin
      Principal.REInfo.Lines.Add('Il y a un modem sur le port Com3');
      Structure_ul_v_ListeCom[3] := COM_3;
    end;
except
end;

//Test du COM4
CPAuto.Close;
CPAuto.Port := com4;
try
  CPAuto.Open;
  CPAuto.WriteStr('AT' + #13);

  if Pos('OK', WaitForString(200)) <> 0 then
    begin
      Principal.REInfo.Lines.Add('Il y a un modem sur le port Com4');
      Structure_ul_v_ListeCom[4] := COM_4;
    end;
except
end;

//Test du COM5
CPAuto.Close;
CPAuto.Port := com5;
try
  CPAuto.Open;
  CPAuto.WriteStr('AT' + #13);

  if Pos('OK', WaitForString(200)) <> 0 then
    begin
      Principal.REInfo.Lines.Add('Il y a un modem sur le port Com5');
      Structure_ul_v_ListeCom[5] := COM_5;
    end;
except
end;

//Test du COM6
CPAuto.Close;
CPAuto.Port := com6;
try
  CPAuto.Open;
  CPAuto.WriteStr('AT' + #13);

  if Pos('OK', WaitForString(200)) <> 0 then
    begin
      Principal.REInfo.Lines.Add('Il y a un modem sur le port Com6');
      Structure_ul_v_ListeCom[6] := COM_6;
    end;
except
end;

//Test du COM7
CPAuto.Close;
CPAuto.Port := com7;
try
  CPAuto.Open;
  CPAuto.WriteStr('AT' + #13);
```

```

if Pos('OK', WaitForString(200)) <> 0 then
  begin
    Principal.REInfo.Lines.Add('Il y a un modem sur le port Com7');
    Structure_ul_v_ListeCom[7] := COM_7;
  end;
except
end;

//Test du COM8
CPAuto.Close;
CPAuto.Port := com8;
try
  CPAuto.Open;
  CPAuto.WriteStr('AT' + #13);

  if Pos('OK', WaitForString(200)) <> 0 then
    begin
      Principal.REInfo.Lines.Add('Il y a un modem sur le port Com8');
      Structure_ul_v_ListeCom[8] := COM_8;
    end;
  except
end;

{Ferme le port COM et ouvre la fenêtre de sélection des ports COM plausibles,
 puis ouvre le port sélectionné}
CPAuto.Close;
SelectPortCom.Show;
Principal.ComPort1.Open;
Screen.Cursor := crDefault;
end;

```

3.4.6 procédure TConfiguration.ChangeEnSecondes

```

//*****
//          Convertis la position du curseur en secondes
//*****
procedure TConfiguration.ChangeEnSecondes;
begin
  {Le nombre dans l'interval de Timer va de 0 à 4294967295 (Type Cardinal). Donc
  comme 1 seconde = 1000ms dans le timer, alors le temps maximum sera de plus de
  49 jours. Les curseurs permettent donc de sélectionner le temps de 1 seconde
  minimum à 48 jours, 23 heures, 59 minutes et 59 secondes au maximum ce qui
  équivaut à 4233540000 ms}
  Principal.TScrute.Interval := (Configuration.TBJours.Position * 86400000) +
    (Configuration.TBHeures.Position * 3600000) +
    (Configuration.TBMinutes.Position * 60000) +
    (Configuration.TBSecondes.Position * 1000);
end;

```

3.4.7 procédure TConfiguration.TBJoursChange

```

//*****
//          Redéfinis le Timer lors du changement du curseur du jour
//*****
procedure TConfiguration.TBJoursChange(Sender: TObject);
begin
  if TBJours.Position in [0..1] then
    LJours.Caption := IntToStr(TBJours.Position) + ' jour'
  else
    LJours.Caption := IntToStr(TBJours.Position) + ' jours';
  ChangeEnSecondes
end;

```

3.4.8 procédure TConfiguration.TBHeuresChange

```

//*****
//          Redéfinis le Timer lors du changement du curseur de l'heure
//*****
procedure TConfiguration.TBHeuresChange(Sender: TObject);
begin
  if TBHeures.Position in [0..1] then
    LHeures.Caption := IntToStr(TBHeures.Position) + ' heure'
  else
    LHeures.Caption := IntToStr(TBHeures.Position) + ' heures';
  ChangeEnSecondes
end;

```

3.4.9 procédure TConfiguration.TBMinutesChange

```

//*****
//          Redéfinis le Timer lors du changement du curseur des minutes
//*****
procedure TConfiguration.TBMinutesChange(Sender: TObject);
begin
  if TBMinutes.Position in [0..1] then
    LMinutes.Caption := IntToStr(TBMinutes.Position) + ' minute'
  else
    LMinutes.Caption := IntToStr(TBMinutes.Position) + ' minutes';
  ChangeEnSecondes
end;

```

3.4.10 procédure TConfiguration.TBSecondesChange

```

//*****
//          Redéfinis le Timer lors du changement du curseur des secondes
//*****
procedure TConfiguration.TBSecondesChange(Sender: TObject);
begin
  if TBSecondes.Position = 1 then
    LSecondes.Caption := IntToStr(TBSecondes.Position) + ' seconde'
  else
    LSecondes.Caption := IntToStr(TBSecondes.Position) + ' secondes';
  ChangeEnSecondes
end;

```

3.4.11 procédure TConfiguration.BAideClick

```

//*****
//          Affiche l'aide sur la configuration
//*****
procedure TConfiguration.BAideClick(Sender: TObject);
begin
  Aide.TabControl1.TabIndex := 2;
  Aide.Show;
end;

```

3.4.12 procédure TConfiguration.CBOperateurChange

```

//*****
//          Lorsqu'on change d'opérateur le numéro change aussi
//*****
procedure TConfiguration.CBOperateurChange(Sender: TObject);
begin
  CBNumOperateur.ItemIndex := CBOperateur.ItemIndex;
end;

```

3.4.13 **procedure TConfiguration.CBNumOperateurChange**

```

//*****
//          Lorsqu'on change le numéro, l'opérateur change aussi
//*****
procedure TConfiguration.CBNumOperateurChange(Sender: TObject);
begin
    CBOperateur.ItemIndex := CBNumOperateur.ItemIndex;
end;

```

3.4.14 **procedure TConfiguration.FormCreate**

```

//*****
//          A la création de la fenêtre de configuration
//*****
procedure TConfiguration.FormCreate(Sender: TObject);
begin
    //Donne le bon temps au Timer
    LJours.Caption := IntToStr(TBJours.Position) + ' jour';
    LHeures.Caption := IntToStr(TBHeures.Position) + ' heure';
    LMinutes.Caption := IntToStr(TBMinutes.Position) + ' minute';
    LSecondes.Caption := IntToStr(TBSecondes.Position) + ' secondes';
    ChangeEnSecondes;

    //Charge la liste des opérateurs et de leurs numéros
    CBOperateur.Items.LoadFromFile(Structure_ul_v_adresse +
                                   '\fichiers\Operateurs.txt');
    CBNumOperateur.Items.LoadFromFile(Structure_ul_v_adresse +
                                       '\fichiers\NumOperateurs.txt');

    //Au cas ou la base de donnée aurais un autre nom
    if Principal.Table.TableName = '' then
        Principal.Table.TableName := ENomBD.Text;
end;

```

3.4.15 **procedure TConfiguration.CBActiveEffaceClick**

```

//*****
//          Affiche ou masque certains objets
//*****
procedure TConfiguration.CBActiveEffaceClick(Sender: TObject);
begin
    if CBActiveEfface.Checked then
        GBEffaceMail.Visible := True
    else
        GBEffaceMail.Visible := False;
end;

end.

```

3.5 **Librairie UCP_u1**

```

{=====
VERSION    : 1.0
COMPILEUR  : Borland Delphi 5.0 sous Windows 2000 PRO
CREE PAR   : Segalla Jacques
DATE      : 11 septembre 2002
EFFECTUE   : Librairie de manipulation des trames UCP
=====}
unit UCP_u1;

interface
uses
    Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
    PiconeBarreTache, Menus, email, StdCtrls, CPort, ExtCtrls, structure_ul;

procedure UCPInitChampsHeaderOper;

```

```
function UCPCheckSum(buf : String) : String;
function UCPConstruitTrame30 : String;
procedure UCPDecodeHeader(Trame : String);
procedure UCPDecodeAck30(Trame : String);
procedure UCPDecodeNack30(Trame : String);
```

implementation

uses

```
Config_ul, natel_ul;
```

3.5.1 function AsciiToIA5

```

//*****
//          Convertis quelques caractères Ascii en son équivalent IA5
//*****
function AsciiToIA5({IN} car : Word) : String;
begin
  if (car = 10) or (car = 13) or (car in [32..90, 97..122]) then
    Result := IntToHex(car, 2);

  case car of
    64 : Result := '00'; 163 : Result := '01'; 36 : Result := '02';
    165 : Result := '03'; 232 : Result := '04'; 233 : Result := '05';
    249 : Result := '06'; 236 : Result := '07'; 242 : Result := '08';
    231 : Result := '09'; 216 : Result := '0B'; 248 : Result := '0C';
    197 : Result := '0E'; 229 : Result := '0F'; 95 : Result := '11';
    198 : Result := '1C'; 230 : Result := '1D'; 223 : Result := '1E';
    201 : Result := '1F'; 196 : Result := '5B'; 92 : Result := '5C';
    209 : Result := '5D'; 220 : Result := '5E'; 167 : Result := '5F';
    191 : Result := '60'; 228 : Result := '7B'; 246 : Result := '7C';
    241 : Result := '7D'; 252 : Result := '7E'; 224 : Result := '7F';
  end;
end;
```

3.5.2 procedure UCPLnitChampsHeaderOper

```

//*****
//          Initialise les champs de Oper et Header
//*****
procedure UCPLnitChampsHeaderOper;
begin
  //Initialisation des champs
  Structure_ul_v_Tete.TRN := '';
  Structure_ul_v_Tete.LEN := '';
  Structure_ul_v_Tete._OR := '';
  Structure_ul_v_Tete.OT := '';
  Structure_ul_v_Tete.Check := '';
  Structure_ul_v_Oper.Adc := '';
  Structure_ul_v_Oper.OAdC := '';
  Structure_ul_v_Oper.AC := '';

  {Test si il faut renvoyer une quittance. Définis dans la boîte de
  configuration. N'est pas utilisé pour l'instant}
  if Configuration.GRQuittance.ItemIndex = 0 then
  begin
    Structure_ul_v_Oper.NRq := '1';
    Structure_ul_v_Oper.NAd := '0327211443';
    Structure_ul_v_Oper.NPID := '0138';
  end
  else
  begin
    Structure_ul_v_Oper.NRq := '';
    Structure_ul_v_Oper.NAd := '';
    Structure_ul_v_Oper.NPID := '';
  end;
end;

//Test si il faut différé le message
```

```

if Configuration.CHBDiffere.Checked = True then
begin
    Structure_ul_v_Oper.DD := '1';
    Structure_ul_v_Oper.DDT := Configuration.MEDate.Text;
    Structure_ul_v_Oper.VP := Configuration.MEGarde.Text;
end
else
begin
    Structure_ul_v_Oper.DD := '';
    Structure_ul_v_Oper.DDT := '';
    Structure_ul_v_Oper.VP := '';
end;
    Structure_ul_v_Oper.AMsg := ''
end;

```

3.5.3 procédure UCPIInitChampsAckNack

```

//*****
//                               Initialise les champs de Ack et Nack
//*****
procedure UCPIInitChampsAckNack;
begin
    Structure_ul_v_Ack.Ack := '';
    Structure_ul_v_Ack.MVP := '';
    Structure_ul_v_Ack.SM.AdC := '';
    Structure_ul_v_Ack.SM.SCTS := '';
    Structure_ul_v_Nack.Nack := '';
    Structure_ul_v_Nack.EC := '';
    Structure_ul_v_Nack.SM := '';
end;

```

3.5.4 fonction UCPCheckSum

```

//*****
//                               Calcul le Checksum de la trame UCP
//*****
function UCPCheckSum({IN} Buf : String //Trame reçue de STX à ETX
                    ) : String;      //Checksum au format hexa.

var
    Cpt      : Word; //Compteur pour la boucle 'for'
    CRCInt   : Word; //CRC calculé (sur 8 bits)

begin
    CRCInt := 0;

    for Cpt := 1 to Length(buf) do
    begin
        CRCInt := CRCInt + Ord(buf[Cpt]);
        if CRCInt >= 256 then CRCInt := CRCInt - 256;
    end;

    //Convertis le nombre en 2 caractères
    Result:= IntToHex(CRCInt, 2)
end;

```

3.5.5 fonction UCPConstruitTrame30

```

//*****
//                               Construit la trame de type 30 pour l'envoi par modem
//*****
function UCPConstruitTrame30 : String; //Trame de type 30

var
    Mes      : String; //Texte de l'email
    Cpt      : Integer; //Compteur
    Longueur : Integer; //Longueur de la trame entre STX et ETX

begin
    Screen.Cursor := crDefault;

```

```

//Initialisation des champs
UCPInitChampsHeaderOper;
Structure_ul_v_Tete._OR := '0';
Structure_ul_v_Tete.OT := '30';
Structure_ul_v_Oper.Adc := Structure_ul_v_mail.Subject;
Structure_ul_v_Oper.OAdC := Configuration.EOriginator.Text;

Mes := Structure_ul_v_mail.Text;
Cpt := 1;

//Mobile normal ou Nokia 3210->8210
if Configuration.RGMobile.ItemIndex = 0 then
begin
  //Enlève les #10 (Line Feed)
  while Cpt <= Length(Mes) do
    begin
      if (Mes[Cpt] = #13) and (Mes[Cpt+1] = #10) then
        Delete(Mes, Cpt+1, 1);
        Inc(Cpt);
      end;
    end;

  //Convertis le caractères en Héxa
  for Cpt := 1 to Length(Mes) do
    begin
      Structure_ul_v_Oper.AMsg :=
        Structure_ul_v_Oper.AMsg + AsciiToIA5(Ord(Mes[Cpt]));
    end;

  //Longueur de la trame entre STX et ETX
  Longueur := 14 + 10 + 2 + Length(Structure_ul_v_Oper.Adc) +
    Length(Structure_ul_v_Oper.OAdC) + Length(Structure_ul_v_Oper.AC)
    + Length(Structure_ul_v_Oper.NRq) + Length(Structure_ul_v_Oper.NAd)
    + Length(Structure_ul_v_Oper.NPID) + Length(Structure_ul_v_Oper.DD)
    + Length(Structure_ul_v_Oper.DDT) + Length(Structure_ul_v_Oper.VP)
    + Length(Structure_ul_v_Oper.AMsg);

  //Construit le numéro de référence au format 2 caractères
  case Length(IntToStr(Structure_ul_v_transaction)) of
    1 : Structure_ul_v_Tete.TRN := '0' + IntToStr(Structure_ul_v_transaction);
    2 : Structure_ul_v_Tete.TRN := IntToStr(Structure_ul_v_transaction);
  end;

  //Construit la longueur au format 5 caractères
  case Length(IntToStr(Longueur)) of
    1 : Structure_ul_v_Tete.LEN := '0000' + IntToStr(Longueur);
    2 : Structure_ul_v_Tete.LEN := '000' + IntToStr(Longueur);
    3 : Structure_ul_v_Tete.LEN := '00' + IntToStr(Longueur);
    4 : Structure_ul_v_Tete.LEN := '0' + IntToStr(Longueur);
    5 : Structure_ul_v_Tete.LEN := IntToStr(Longueur);
  end;

  //Trame complète sans le CheckSum
  Result := Structure_ul_v_Tete.TRN + '/' + Structure_ul_v_Tete.LEN + '/' +
    Structure_ul_v_Tete._OR + '/' + Structure_ul_v_Tete.OT + '/' +
    Structure_ul_v_Oper.Adc + '/' + Structure_ul_v_Oper.OAdC + '/' +
    Structure_ul_v_Oper.AC + '/' + Structure_ul_v_Oper.NRq + '/' +
    Structure_ul_v_Oper.NAd + '/' + Structure_ul_v_Oper.NPID + '/' +
    Structure_ul_v_Oper.DD + '/' + Structure_ul_v_Oper.DDT + '/' +
    Structure_ul_v_Oper.VP + '/' + Structure_ul_v_Oper.AMsg + '/';

  //Calcul du CheckSum
  Structure_ul_v_Tete.Check := UCPCheckSum(Result)
end;

```

3.5.6 procédure UCPDecodeHeader

```

/*****
//
//          Décode l'en-tête de type 30
//
*****/
procedure UCPDecodeHeader({IN} Trame : String); //Trame de type 30
var
  Cpt : Integer; //Compteur

begin
  UCPInitChampsHeaderOper;
  //Lis les champs de l'en-tête. Les champs sont séparés par le caractère '/'
  //Champ TRN
  Cpt := 1;
  while not(Trame[Cpt] = '/') do
  begin
    Structure_ul_v_Tete.TRN := Structure_ul_v_Tete.TRN + Trame[Cpt];
    Inc(Cpt);
  end;

  //Champ LEN
  Inc(Cpt);
  while not(Trame[Cpt] = '/') do
  begin
    Structure_ul_v_Tete.LEN := Structure_ul_v_Tete.LEN + Trame[Cpt];
    Inc(Cpt);
  end;

  //Champ OR
  Inc(Cpt);
  while not(Trame[Cpt] = '/') do
  begin
    Structure_ul_v_Tete._OR := Structure_ul_v_Tete._OR + Trame[Cpt];
    Inc(Cpt);
  end;

  //Champ OT
  Inc(Cpt);
  while not(Trame[Cpt] = '/') do
  begin
    Structure_ul_v_Tete.OT := Structure_ul_v_Tete.OT + Trame[Cpt];
    Inc(Cpt);
  end
end;

```

3.5.7 procédure UCPDecodeAck30

```

/*****
//
//          Décode une trame positive de type 30
//
*****/
procedure UCPDecodeAck30({IN} Trame : String); //Trame Ack de type 30
var
  Cpt : Integer; //Compteur

begin
  UCPInitChampsAckNack;
  {Le 1er caractère se trouve à la position 15, car il y a d'abord les 14
  caractères de l'en-tête}
  //Champ Ack
  Cpt := 15;
  while not(Trame[Cpt] = '/') do
  begin
    Structure_ul_v_Ack.Ack := Structure_ul_v_Ack.Ack + Trame[Cpt];
    Inc(Cpt);
  end;

  //Champ MVP
  Inc(Cpt);

```

```

while not(Trame[Cpt] = '/') do
begin
  Structure_ul_v_Ack.MVP := Structure_ul_v_Ack.MVP + Trame[Cpt];
  Inc(Cpt);
end;

//Champ AdC
Inc(Cpt);
while not(Trame[Cpt] = ':') do
begin
  Structure_ul_v_Ack.SM.AdC := Structure_ul_v_Ack.SM.AdC + Trame[Cpt];
  Inc(Cpt);
end;

//Champ SCTS
Inc(Cpt);
while not(Trame[Cpt] = '/') do
begin
  Structure_ul_v_Ack.SM.SCTS := Structure_ul_v_Ack.SM.SCTS + Trame[Cpt];
  Inc(Cpt);
end;

Structure_ul_v_Tete.Check := Copy(Trame, Length(Trame) - 1, 2);
end;

```

3.5.8 procédure UCPDecodeNack30

```

//*****
//                               Décode une trame négative de type 30
//*****
procedure UCPDecodeNack30({IN} Trame : String); //Trame Nack de type 30
var
  Cpt : Integer; //Compteur

begin
  UCPInitChampsHeaderOper;
  {Le 1er caractère se trouve à la position 15, car il y a d'abord les 14
  caractères de l'en-tête}
  //Champ Nack
  Cpt := 15;
  while not(Trame[Cpt] = '/') do
  begin
    Structure_ul_v_NAck.Nack := Structure_ul_v_NAck.Nack + Trame[Cpt];
    Inc(Cpt);
  end;

  //Champ EC
  Inc(Cpt);
  while not(Trame[Cpt] = '/') do
  begin
    Structure_ul_v_NAck.EC := Structure_ul_v_NAck.EC + Trame[Cpt];
    Inc(Cpt);
  end;

  //Champ SM
  Inc(Cpt);
  while not(Trame[Cpt] = '/') do
  begin
    Structure_ul_v_NAck.SM := Structure_ul_v_NAck.SM + Trame[Cpt];
    Inc(Cpt);
  end;

  Structure_ul_v_Tete.Check := Copy(Trame, Length(Trame) - 1, 2);
end;

end.

```

3.6 Librairie UDP_u1

```

=====
VERSION      : 1.0
COMPILEUR   : Borland Delphi 5.0 sous Windows 2000 PRO
CREE PAR    : Segalla Jacques
DATE        : 11 septembre 2002
EFFECTUE    : Librairie de manipulation des trames UDP
=====
}
{ FUNCTION   : RESULTSTRING = DecToBin(DECIMAL)
  Convertis un nombre décimal en binaire (7bits)
=====
}
{ FUNCTION   : RESULTSTRING = HexToBin(HEXSTRING)
  Convertis un nombre Hexa (string) en Binaire (string)
=====
}
{ FUNCTION   : RESULTINTEGER = HexCharToInt(HEXCHAR)
  Convertis un caractère Hexa (0..9 & A..F or a..f) en entier
=====
}
{ FUNCTION   : RESULTSTRING = HexCharToBin(HEXCHAR)
  Convertis un caractère Hexa (0..9 & A..F or a..f) en binaire
=====
}
{ FUNCTION   : RESULTINTEGER = Pow(BASE,POWER)
  Simple power routine resulting in an integer (16bit)
=====
}
{ FUNCTION   : RESULTINTEGER = BinStrToInt(BINSTRING)
  Convertis un 16 bit binaire string en un entier
=====
}
{ FUNCTION   : RESULTSTRING = DecodeSMS7Bit(PDUSTRING)
  Decode un 7 bits SMS to ASCII
=====
}
{ FUNCTION   : RESULTSTRING = DecodeSMSText(SOURCESTRING)
  Décode un ASCII en 7 bits
=====
}
{ FUNCTION   : RESULTSTRING = ReverseStr SOURCESTRING)
  Met une chaîne à l'envers
=====
}
unit UDP_u1;

interface
function DecToBin(var valeur : Word) : String;
function HexToBin(HexNr : string) : string;
function HexCharToInt(HexToken : Char) : Integer;
function HexCharToBin(HexToken : Char) : string;
function Pow(base, puissance : Integer) : Integer;
function BinStrToInt(BinStr : string) : Integer;
function DecodeSMS7Bit(PDU : string) :string;
function DecodeSMSText(texte : String) : String;
function ReverseStr(SourceStr : string) : String;

```

```

implementation
uses sysutils, dialogs;

```

3.6.1 fonction DecToBin

```

//*****
//          Convertis un nombre décimal en binaire (7bits)
//*****
function DecToBin(      var Valeur : Word           //Nombre en décimal
                    ) : String;                 //Chaîne en binaire

var
  Cpt   : Integer; //Compteur pour la boucle for
  Resul : Integer; //Résultat en entier

const
  Tab : array[1..7] of Integer = (64, 32, 16, 8, 4, 2, 1);

begin
  //Initialise la chaîne

```

```

Result := '    ';
for Cpt:=1 to 7 do
begin
  Resul := Valeur div Tab[Cpt];
  if Resul = 0 then
    Result[Cpt]:='0'
  else
    Result[Cpt]:='1';

  Valeur := Valeur mod Tab[Cpt];
end //for
end;
```

3.6.2 function HexToBin

```

//*****
//          Convertis un nombre Hexa (string) en Binaire (string)
//*****
function HexToBin(HexNr : string //Chaîne en Hexa.
                  ): string;    //Chaîne en binaire

var
  Cpt : Integer; //Compteur

begin
  Result := '';

  for Cpt := 1 to length(HexNr) do
    Result := Result + HexCharToBin(HexNr[Cpt])
  end;
```

3.6.3 function HexCharToInt

```

//*****
//          Convertis un caractère Hexa (0..9 & A..F or a..f) en entier
//*****
function HexCharToInt(HexToken : char //Caractère en Hexa.
                     ):Integer;     //Entier du caractère Hexa.

begin
  Result:=0;

  if (HexToken>#47) and (HexToken<#58) then { car 0....9 }
    Result:=Ord(HexToken)-48
  else if (HexToken>#64) and (HexToken<#71) then { car A....F }
    Result:=Ord(HexToken)-65 + 10
  end;
```

3.6.4 function HexCharToBin

```

//*****
//          Convertis un caractère Hexa (0..9 & A..F or a..f) en binaire
//*****
function HexCharToBin(HexToken : char //Caractère Hexa.
                     ): string;     //Chaîne binaire du caractère hexa.

var
  DivGauche : Integer;

begin
  DivGauche := HexCharToInt(HexToken);
  Result := '';

  //Division par 2. Si le resultat est impair alors bit à 1 sinon bit à 0
  repeat
    if odd(DivGauche) then
      Result := '1' + Result
    else
      Result := '0' + Result;

    //Continue à divisé jusqu'à tout à gauche et longueur = 4
```

```

DivGauche := DivGauche div 2;
until (DivGauche = 0) and (Length(Result) = 4)
end;

```

3.6.5 function Pow

```

//*****
//                               Base à la puissance Puissance
//*****
function Pow(Base, Puissance: Integer): Integer; //Base^puissance = Result
var
  Cpt : Integer; //Compteur

begin
  Result := 1;

  for Cpt := 1 to Puissance do
    Result := Result * Base
  end;

```

3.6.6 function BinStrToInt

```

//*****
//                               Convertis un 16 bit binaire string en un entier
//*****
function BinStrToInt(BinStr : string //Chaîne binaire
                    ) : Integer;    //Chaîne binaire converti en Entier
var
  Cpt : Integer; //Compteur

begin
  if Length(BinStr) > 16 then
    raise ERangeError.Create(#13 + BinStr + #13 + ' n''est pas un nombre 16 ' +
                              'bits valide.'+#13);

  Result := 0;

  for Cpt := 1 to Length(BinStr) do
    if BinStr[Cpt] = '1' then
      Result := Result + Pow(2, Length(BinStr) - Cpt)
  end;

```

3.6.7 function DecodeSMS7Bit

```

//*****
// Décode du format PDU au format texte (Algorithme trouvé sur ngscan.com)
//*****
function DecodeSMS7Bit(PDU : string //Trame PDU
                      ):string;    //Trame PDU au format 7 bits
var
  OctetStr   : string; //
  OctetBin   : string; //
  Charbin    : string; //
  PrevOctet  : string; //
  Cpt1, Cpt2 : Integer; //Compteurs

begin
  PrevOctet:='';
  Result:='';

  for Cpt1 := 1 to Length(PDU) do
    begin
      if Length(PrevOctet) >= 7 then //Si il y a un débordement de bit
        begin
          if BinStrToInt(PrevOctet) <> 0 then
            Result := Result + Chr(BinStrToInt(PrevOctet))
          else
            Result := Result + ' ';
        end
    end;
  end;

```

```

    PrevOctet := '';
end;

//Test que Cpt1 soit impair
if Odd(Cpt1) then
begin
    OctetStr := Copy(PDU, Cpt1, 2);
    OctetBin := HexToBin(OctetStr);

    Charbin := '';
    for Cpt2 := 1 to Length(PrevOctet) do
        Charbin := Charbin + PrevOctet[Cpt2];

    for Cpt2 := 1 to 7 - Length(PrevOctet) do
        Charbin := OctetBin[8- Cpt2 + 1] + Charbin;

    if BinStrToInt(Charbin) <> 0 then
        Result := Result + Chr(BinStrToInt(Charbin))
    else
        Result := Result + ' ';

    PrevOctet := Copy(OctetBin, 1, Length(PrevOctet) + 1);
end;
end
end;

```

3.6.8 function DecodeSMSText

```

/*****
//          Met le texte ASCII au format PDU pour l'envoi depuis un natel
/*****
function DecodeSMSText(Texte : String //Trame au format ASCII
                        ) : String; //Trame au format PDU

var
    Cpt1, Cpt2 : Integer; //Compteurs
    Temp       : Word;    //
    Chaine2    : String; //
    Temp1      : String; //

begin
    Chaine2 := '';
    Cpt1 := 1;

    //Enlève les #10 (Line Feed)
    while Cpt1 <= Length(Texte) do
    begin
        if (Texte[Cpt1] = #13) and (Texte[Cpt1+1] = #10) then
            Delete(Texte, Cpt1+1, 1);
            Inc(Cpt1);
        end;

    //Forme toutes les lettres au format binaire 7 bits
    for Cpt1 := Length(Texte) downto 1 do
    begin
        Temp := Ord(Texte[Cpt1]);

        Chaine2 := Chaine2 + DecToBin(Temp);
    end;

    //Ajoute le nombre de '0' nécessaire pour faire des mots de 8 bits
    for Cpt1 := 1 to Length(Texte) Mod 8 do
    begin
        Chaine2 := '0' + Chaine2;
    end;

    //Prends tous les caractères par 8 et les transforment en hexa.
    for Cpt1 := 1 to (Length(Chaine2) + 1) DIV 8 do

```

```

begin
  Temp1 := '';
  for Cpt2 := 1 to 8 do
    begin
      Temp1 := Temp1 + Chaîne2[(8*(Cpt1-1))+Cpt2];
    end;
  Result := IntToHex(BinStrToInt(Temp1), 2) + Result;
end
end;
```

3.6.9 fonction ReverseStr

```

//*****
//                               Inverse une chaîne
//*****
function ReverseStr(SourceStr : string //Chaîne à inversée
                   ) : String;       //Chaîne inversée

var
  Cpt : Integer; //Compteur

begin
  Result:='';
  Cpt := 1;

  while Cpt < length(SourceStr) do
    begin
      Result := Result + SourceStr[Cpt + 1] + SourceStr[Cpt];
      Inc(Cpt, 2);
    end
  end;
end.

end.
```

3.7 Librairie SelectPortCom_u1

```

{=====
VERSION      : 1.0
COMPILEUR   : Borland Delphi 5.0 sous Windows 2000 PRO
CREE PAR    : Segalla Jacques
DATE        : 11 septembre 2002
EFFECTUE    : Sélection des ports séries autorisés par l'unité de configuration
=====}
unit SelectPortCom_u1;

interface

uses
  Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
  StdCtrls, ExtCtrls;

type
  TSelectPortCom = class(TForm)
    RGC0M: TRadioGroup;
    BOK: TButton;
    BAnnule: TButton;
    procedure BOKClick(Sender: TObject);
    procedure FormActivate(Sender: TObject);
    procedure RGC0MClick(Sender: TObject);
    procedure BAnnuleClick(Sender: TObject);
  private
    { Déclarations privées }
  public
    { Déclarations publiques }
  end;

var
  SelectPortCom : TSelectPortCom;
```

implementation

uses Natel_ul, CPort, Config_ul, Structure_ul;

{\$R *.DFM}

3.7.1 procedure TSelectPortCom.BOKClick

```
//*****
//                               Sélectionne le bon port série
//*****
procedure TSelectPortCom.BOKClick(Sender: TObject);
begin
    Case RGC.COM.ItemIndex of
        0 : Principal.ComPort1.Port := Com1;
        1 : Principal.ComPort1.Port := Com2;
        2 : Principal.ComPort1.Port := Com3;
        3 : Principal.ComPort1.Port := Com4;
        4 : Principal.ComPort1.Port := Com5;
        5 : Principal.ComPort1.Port := Com6;
        6 : Principal.ComPort1.Port := Com7;
        7 : Principal.ComPort1.Port := Com8;
    end;

    //Ferme la fenêtre de sélection du port série
    SelectPortCom.Close
end;
```

3.7.2 procedure TSelectPortCom.FormActivate

```
//*****
//                               Désactive les Radio boutons qu'on ne peut pas sélectionner
//*****
procedure TSelectPortCom.FormActivate(Sender: TObject);
var
    cpt : Byte; //Compteur (Nombre de port série)

begin
    for cpt := 1 to 8 do
        TRadioButton(RGC.COM.Controls[cpt-1]).Enabled := Structure_ul_v_ListeCom[cpt]
    end;
```

3.7.3 procedure TSelectPortCom.RGC.COMClick

```
//*****
//                               Si port série sélectionné, alors active le bouton OK
//*****
procedure TSelectPortCom.RGC.COMClick(Sender: TObject);
begin
    if RGC.COM.ItemIndex <> -1 then
        BOK.Enabled := True
    end;
```

3.7.4 procedure TSelectPortCom.BAnnuleClick

```
//*****
//                               Ferme la fenêtre de sélection des ports série
//*****
procedure TSelectPortCom.BAnnuleClick(Sender: TObject);
begin
    SelectPortCom.Close
end;

end.
```

3.8 Librairie Test_u1

```

=====
VERSION      : 1.0
COMPILEUR   : Borland Delphi 5.0 sous Windows 2000 PRO
CREE PAR    : Segalla Jacques
DATE        : 11 septembre 2002
EFFECTUE    : Fenêtre de test d'envoi des SMS
=====
}
unit Test_u1;

interface

uses
  Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
  StdCtrls;

type
  TTest = class (TForm)
    LDestinataire: TLabel;
    EDestinataire: TEdit;
    LMessage: TLabel;
    LCompte: TLabel;
    MMessage: TMemo;
    BEnvoie: TButton;
    BAide: TButton;
    BAnnule: TButton;
    procedure BAideClick(Sender: TObject);
    procedure BAnnuleClick(Sender: TObject);
    procedure MMessageKeyPress(Sender: TObject; var Key: Char);
    procedure MMessageChange(Sender: TObject);
    procedure BEnvoieClick(Sender: TObject);
    procedure FormActivate(Sender: TObject);
  private
    { Déclarations privées }
  public
    { Déclarations publiques }
  end;

var
  Test: TTest;

implementation

uses Aide_u1, Natel_u1, Structure_u1;

{$R *.DFM}

```

3.8.1 **procedure TTest.BAideClick**

```

//*****
//                               Affiche la fenêtre d'aide (onglet Test)
//*****
procedure TTest.BAideClick(Sender: TObject);
begin
  Aide.TabControl1.TabIndex := 3;
  Aide.Show;
end;

```

3.8.2 procedure TTest.BAnnuleClick

```

//*****
//                               Ferme la fenêtre de Test
//*****
procedure TTest.BAnnuleClick(Sender: TObject);
begin
  Test.Close;
end;

```

3.8.3 procedure TTest.MMessageKeyPress

```

//*****
//                               Bloque à 160 caractères max. dans le mémo
//*****
procedure TTest.MMessageKeyPress(Sender: TObject; var Key: Char);
begin
  if Length(MMessage.Text) >= 160 then
    MMessage.ReadOnly := True;

  if Key = Chr(VK_BACK) then
    MMessage.ReadOnly := False;
end;

```

3.8.4 procedure TTest.MMessageChange

```

//*****
//                               Affiche le nombre de caractères dans le Memo
//*****
procedure TTest.MMessageChange(Sender: TObject);
begin
  LCompte.Caption := IntToStr(160 - Length(MMessage.Text)) + ' car.';
end;

```

3.8.5 procedure TTest.BEnvoieClick

```

//*****
//                               Envoi le SMS
//*****
procedure TTest.BEnvoieClick(Sender: TObject);
begin
  Test.Hide;
  Structure_ul_v_mail.Subject := EDestinataire.Text;
  Structure_ul_v_mail.Text := MMessage.Text;
  Principal.EnvoiModem;
end;

```

3.8.6 procedure TTest.FormActivate

```

//*****
//                               Affiche la le nombre de caractères dans le Memo
//*****
procedure TTest.FormActivate(Sender: TObject);
begin
  LCompte.Caption := IntToStr(160 - Length(MMessage.Text)) + ' car.';
end;

end.

```

3.9 Librairie Print_u1

```

=====
VERSION      : 1.0
COMPILEUR   : Borland Delphi 5.0 sous Windows 2000 PRO
CREE PAR    : Segalla Jacques
DATE        : 11 septembre 2002
EFFECTUE    : Librairie pour l'impression de la base de donnée
=====
}
unit Print_u1;

interface

uses
  Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
  QuickRpt, QrCtrls, ExtCtrls, StdCtrls;

type
  TImprime = class(TForm)
    Rapport: TQuickRep;
    ColumnHeaderBand1: TQRBand;
    QRLabel2: TQRLabel;
    QRLabel3: TQRLabel;
    QRLabel4: TQRLabel;
    DetailBand1: TQRBand;
    QRDBText1: TQRDBText;
    QRDBText2: TQRDBText;
    QRDBText3: TQRDBText;
    PageFooterBand1: TQRBand;
    QRLabel5: TQRLabel;
    QRSysData1: TQRSysData;
    TitleBand1: TQRBand;
    QRLabel1: TQRLabel;
  private
    { Déclarations privées }
  public
    { Déclarations publiques }
  end;

var
  Imprime: TImprime;

implementation

{$R *.DFM}

end.

```

3.10 Librairie Aide_u1

```

=====
VERSION      : 1.0
COMPILEUR   : Borland Delphi 5.0 sous Windows 2000 PRO
CREE PAR    : Segalla Jacques
DATE        : 11 septembre 2002
EFFECTUE    : Aide du programme 'Passerelle SMS'
=====
}
unit Aide_u1;

interface

uses
  Windows, SysUtils, Classes, Graphics, Forms, Controls, StdCtrls,
  Buttons, ComCtrls, ExtCtrls;

type
  TAide = class(TForm)
    Panel2: TPanel;

```

```

OKBtn: TButton;
TabControl1: TTabControl;
RichEdit1: TRichEdit;
procedure OKBtnClick(Sender: TObject);
procedure ChargeAide;
procedure TabControl1Change(Sender: TObject);
procedure FormActivate(Sender: TObject);
private
  { Private declarations }
public
  { Public declarations }
end;

```

```

var
  Aide: TAide;

```

```

implementation
uses Structure_ul;

```

```
{$R *.DFM}
```

3.10.1 **procedure TAide.ChargeAide**

```

//*****
//                               Charge les bon fichiers d'aide dans le memo
//*****
procedure TAide.ChargeAide;
begin
  if TabControl1.TabIndex = 0 then
    RichEdit1.Lines.LoadFromFile(Structure_ul_v_adresse + '\aide\principal.rtf')
  else if TabControl1.TabIndex = 1 then
    RichEdit1.Lines.LoadFromFile(Structure_ul_v_adresse + '\aide\journal.rtf')
  else if TabControl1.TabIndex = 2 then
    RichEdit1.Lines.LoadFromFile(Structure_ul_v_adresse +
'\aide\configuration.rtf')
  else if TabControl1.TabIndex = 3 then
    RichEdit1.Lines.LoadFromFile(Structure_ul_v_adresse + '\aide\test.rtf')
end;

```

3.10.2 **procedure TAide.OKBtnClick**

```

//*****
//                               Ferme la fenêtre de l'aide
//*****
procedure TAide.OKBtnClick(Sender: TObject);
begin
  Aide.Close;
end;

```

3.10.3 **procedure TAide.TabControl1Change**

```

//*****
//                               Charge l'aide lors du changement de contrôle
//*****
procedure TAide.TabControl1Change(Sender: TObject);
begin
  ChargeAide;
end;

```

3.10.4 procedure TAide.FormActivate

```

//*****
//                               Charge l'aide lors du chargement de la fenêtre d'aide
//*****
procedure TAide.FormActivate(Sender: TObject);
begin
  ChargeAide;
end;

end.

```

3.11 Librairie Apropos_u1

```

{=====
  VERSION      : 1.0
  COMPILEUR   : Borland Delphi 5.0 sous Windows 2000 PRO
  CREE PAR    : Segalla Jacques
  DATE       : 11 septembre 2002
  EFFECTUE   : Boîte de dialogue 'A propos'
=====}
unit APropos_u1;

interface

uses Windows, SysUtils, Classes, Graphics, Forms, Controls, StdCtrls,
  Buttons, ExtCtrls;

type
  TAPropos = class (TForm)
    Panell: TPanel;
    ProgramIcon: TImage;
    LProductName: TLabel;
    LVersion: TLabel;
    Comments: TLabel;
    OKButton: TButton;
    LCopyright: TLabel;
    procedure OKButtonClick(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  APropos: TAPropos;

implementation

```

```
{ $R *.DFM }
```

3.11.1 procedure TAPropos.OKButtonClick

```

//*****
//                               Ferme la fenêtre A propos
//*****
procedure TAPropos.OKButtonClick(Sender: TObject);
begin
  APropos.Close;
end;

end.

```

3.12 Programme Clique_Oui

```

=====
VERSION      : 1.0
COMPILEUR   : Borland Delphi 5.0 sous Windows 2000 PRO
CREE PAR    : Segalla Jacques
DATE        : 11 septembre 2002
EFFECTUE    : Clique sur le bouton 'Oui' de la fenêtre de sécurité d'Outlook
REMARQUE    : C'est suite à une question posé sur le forum de www.developpez.com
              que j'ai eu diverses réponse au sujet de comment cliquer sur le
              bouton d'une application externe. J'ai simplement adapté à mon
              utilisation les parties de codes qui m'ont été fournis
=====
unit Clique_Oui_ul;

interface
uses
  Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
  shellapi;

implementation
//*****
//                               Enumère les Handles fils de l'application
//*****
function EnumChildProc(Wnd: hWnd; SL: TStringList): Bool;stdcall;
var
  szFull: array[0..1000] of Char; // Buffer for window caption

begin
  Result := Wnd <> 0;
  if Result then
    begin
      //Met le texte dans le buffer
      GetWindowText(Wnd, szFull, SizeOf(szFull));
      //Test pour text
      if (Pos(SL[0], StrPas(szFull)) > 0)
      //Test si il y a plusieurs Handles
      and (SL.IndexOfObject(TObject(Wnd)) < 0) then
        //Ajoute l'élément à la liste
        SL.AddObject(StrPas(szFull), TObject(Wnd));
      EnumChildWindows(Wnd, @EnumChildProc, Longint(SL));
    end;
end;
//*****
//                               Cherche le Handle du nom donné en paramètre et simule un clique
//*****
function ClickButton(ParentWindow: hWnd; ButtonCaption: string): Boolean;
var
  SL : TStringList;
  H  : hWnd;

begin
  SL := TStringList.Create;
  try
    //Le premier élément dans la liste est le texte à recherché
    SL.AddObject(ButtonCaption, nil);
    EnumChildWindows(ParentWindow, @EnumChildProc, Longint(SL));
    H := 0;
    case SL.Count of
      1: //ShowMessage('Titre de fenêtre non trouvé');
      2: H := hWnd(SL.Objects[1]);
    else
      ShowMessage('Titre de fenêtre ambigü');
    end;
  finally
    //Vide la StringList
    SL.Free;
  end;
end;

```

```
end;
Result := H <> 0;
//Clique sur le bouton de type ButtonCaption du Handle
if Result then
begin
    PostMessage(H, BM_CLICK, 0, 0);
end;
end;
//*****
//                               Cliquez sur le bouton 'Oui'
//*****
var
    LeHandle : HWND;
    Nom       : PChar;

begin
    while True do
    begin
        // #32770 représente le nom de la classe de la fenêtre de sécurité Outlook
        while FindWindow('#32770', nil) = 0 do
        begin
            Sleep(50);
            Application.ProcessMessages;
        end;
        // Ferme les message d'Outlook
        Nom := 'Microsoft Outlook'; // caption de la fenêtre
        LeHandle := FindWindow('#32770', nil);
        if LeHandle <> 0 then
        begin
            Application.ProcessMessages;
            // Outlook 2000
            ClickButton(LeHandle, '&Oui'); // Attention ne pas oublier le & pour
            // les touches d'accès rapide !

            // Outlook 2002
            ClickButton(LeHandle, 'Oui');
            // Fait un clique pour accepter
            keybd_event(VK_LBUTTON, 0, 0, 0);
            // Donne le focus à la fenêtre Outlook sinon l'application ne peut plus
            continuer
            SetFocus(LeHandle);
        end;
    end;
end;
end.
```

3.13 Principal: Tprincipal

```
object Principal: TPrincipal
  Left = 363
  Top = 330
  Width = 497
  Height = 333
  Caption = 'Passerelle SMS'
  Color = clBtnFace
  Font.Charset = DEFAULT_CHARSET
  Font.Color = clWindowText
  Font.Height = -11
  Font.Name = 'MS Sans Serif'
  Font.Style = []
  Icon.Data = {...}
  Menu = MainMenu
  OldCreateOrder = False
  PopupMenu = PopupMenu
  Position = poDefaultPosOnly
  Visible = True
  OnCreate = FormCreate
  OnDestroy = FormDestroy
  PixelsPerInch = 96
  TextHeight = 13
  object Gauge: TGauge
    Left = 160
    Top = 80
    Width = 321
    Height = 17
    Hint = 'Indique sous forme graphique le nombre de messages à lire.'
    Anchors = [akLeft, akTop, akRight]
    BackColor = clLime
    ForeColor = clRed
    ParentShowHint = False
    Progress = 0
    ShowHint = True
    ShowText = False
  end
  object LInfo1: TLabel
    Left = 8
    Top = 80
    Width = 136
    Height = 16
    Caption = 'Il reste ? messages'
    Font.Charset = DEFAULT_CHARSET
    Font.Color = clNavy
    Font.Height = -13
    Font.Name = 'MS Sans Serif'
    Font.Style = [fsBold]
    ParentFont = False
  end
  object LInfo2: TLabel
    Left = 8
    Top = 96
    Width = 130
    Height = 16
    Caption = 'à lire dans le profil'
    Font.Charset = DEFAULT_CHARSET
    Font.Color = clNavy
    Font.Height = -13
    Font.Name = 'MS Sans Serif'
    Font.Style = [fsBold]
    ParentFont = False
  end
  object LInfo3: TLabel
```

```
Left = 8
Top = 112
Width = 29
Height = 16
Caption = 'sms'
Font.Charset = DEFAULT_CHARSET
Font.Color = clNavy
Font.Height = -13
Font.Name = 'MS Sans Serif'
Font.Style = [fsBold]
ParentFont = False
end
object StatusBar1: TStatusBar
Left = 0
Top = 268
Width = 489
Height = 19
Panels = <
    item
        Width = 50
    end>
SimplePanel = False
end
object ToolBar1: TToolBar
Left = 0
Top = 0
Width = 489
Height = 73
ButtonHeight = 63
Caption = 'ToolBar1'
EdgeBorders = [ebLeft, ebTop, ebRight, ebBottom]
Indent = 4
TabOrder = 1
object SBStart: TSpeedButton
Left = 4
Top = 2
Width = 101
Height = 63
Hint = 'Lance l'#39'application manuellement...'
BiDiMode = bdLeftToRight
Caption = 'Start manuel'
Font.Charset = DEFAULT_CHARSET
Font.Color = clWindowText
Font.Height = -13
Font.Name = 'MS Sans Serif'
Font.Style = [fsBold]
Glyph.Data = {...}
Layout = blGlyphTop
ParentFont = False
ParentShowHint = False
ParentBiDiMode = False
ShowHint = True
OnClick = SBStartClick
end
object SBConfiguration: TSpeedButton
Left = 105
Top = 2
Width = 100
Height = 63
Hint = 'Fenêtre de configuration...'
Caption = 'Configuration'
Font.Charset = DEFAULT_CHARSET
Font.Color = clWindowText
Font.Height = -13
Font.Name = 'MS Sans Serif'
Font.Style = [fsBold]
Glyph.Data = {...}
Layout = blGlyphTop
```

```
ParentFont = False
ParentShowHint = False
ShowHint = True
OnClick = SBConfigurationClick
end
object SBJournal: TSpeedButton
  Left = 205
  Top = 2
  Width = 64
  Height = 63
  Hint = 'Journal des événements...'
  Caption = 'Journal'
  Font.Charset = DEFAULT_CHARSET
  Font.Color = clWindowText
  Font.Height = -13
  Font.Name = 'MS Sans Serif'
  Font.Style = [fsBold]
  Glyph.Data = {...}
  Layout = blGlyphTop
  ParentFont = False
  ParentShowHint = False
  ShowHint = True
  OnClick = SBJournalClick
end
object SBTest: TSpeedButton
  Left = 269
  Top = 2
  Width = 72
  Height = 63
  Hint = 'Test d'#39'envoi d'#39'un SMS...'
  Caption = 'Test'
  Font.Charset = DEFAULT_CHARSET
  Font.Color = clWindowText
  Font.Height = -13
  Font.Name = 'MS Sans Serif'
  Font.Style = [fsBold]
  Glyph.Data = {...}
  Layout = blGlyphTop
  ParentFont = False
  ParentShowHint = False
  ShowHint = True
  OnClick = SBTestClick
end
object SBQuitter: TSpeedButton
  Left = 341
  Top = 2
  Width = 72
  Height = 63
  Hint = 'Quitte l'#39'application passerelle'
  Caption = 'Quitter'
  Font.Charset = DEFAULT_CHARSET
  Font.Color = clWindowText
  Font.Height = -13
  Font.Name = 'MS Sans Serif'
  Font.Style = [fsBold]
  Glyph.Data = {...}
  Layout = blGlyphTop
  ParentFont = False
  ParentShowHint = False
  ShowHint = True
  OnClick = SBQuitterClick
end
object SBAide: TSpeedButton
  Left = 413
  Top = 2
  Width = 64
  Height = 63
  Hint = 'Aide de l'#39'application...'
```

```
Caption = 'Aide'
Font.Charset = DEFAULT_CHARSET
Font.Color = clWindowText
Font.Height = -13
Font.Name = 'MS Sans Serif'
Font.Style = [fsBold]
Glyph.Data = {...}
Layout = blGlyphTop
ParentFont = False
ParentShowHint = False
ShowHint = True
OnClick = SB AideClick
end
end
object BAnnuler: TButton
  Left = 8
  Top = 176
  Width = 137
  Height = 33
  Anchors = [akLeft, akBottom]
  Caption = 'Annuler'
  TabOrder = 2
  OnClick = BAnnulerClick
end
object REInfo: TRichEdit
  Left = 160
  Top = 104
  Width = 321
  Height = 161
  Hint = 'Tous les messages sont indiqués ici'
  Anchors = [akLeft, akTop, akRight, akBottom]
  Font.Charset = DEFAULT_CHARSET
  Font.Color = clWindowText
  Font.Height = -13
  Font.Name = 'MS Sans Serif'
  Font.Style = [fsBold]
  ParentFont = False
  ParentShowHint = False
  ReadOnly = True
  ScrollBars = ssVertical
  ShowHint = True
  TabOrder = 3
  OnChange = REInfoChange
end
object Panel: TPanel
  Left = 0
  Top = 216
  Width = 153
  Height = 49
  Anchors = [akLeft, akBottom]
  TabOrder = 4
  object SBLance: TSpeedButton
    Left = 8
    Top = 8
    Width = 73
    Height = 33
    Hint = 'Lance l'#39'application automatiquement'
    Anchors = [akLeft, akBottom]
    GroupIndex = 1
    Caption = 'START'
    ParentShowHint = False
    ShowHint = True
    OnClick = SBLanceClick
  end
  object SBStop: TSpeedButton
    Left = 80
    Top = 8
    Width = 65
```

```
    Height = 33
    Hint = 'Stop l'#39'application'
    Anchors = [akLeft, akBottom]
    GroupIndex = 1
    Down = True
    Caption = 'STOP'
    ParentShowHint = False
    ShowHint = True
    OnClick = SBStopClick
end
end
object PopupMenu: TPopupMenu
  Left = 320
  Top = 224
  object ConfigurerCOM: TMenuItem
    Caption = 'Configurer COM...'
    OnClick = ConfigurerCOMClick
  end
  object ConfigurerPasserellePopUp: TMenuItem
    Caption = 'Configurer Passerelle...'
    OnClick = ConfigurerPasserellePopUpClick
  end
  object VoirJournalPopUp: TMenuItem
    Caption = 'Voir Journal...'
    OnClick = VoirJournalPopUpClick
  end
  object TestdelenvoiPopUp: TMenuItem
    Caption = 'Test de l'#39'envoi...'
    OnClick = TestdelenvoiPopUpClick
  end
  object QuitterPopUp: TMenuItem
    Caption = 'Quitter'
    OnClick = QuitterPopUpClick
  end
end
end
object ComPort1: TComPort
  BaudRate = br9600
  Port = COM1
  Parity.Bits = prNone
  Parity.Check = False
  Parity.Replace = False
  Parity.ReplaceChar = 0
  StopBits = sbOneStopBit
  DataBits = dbEight
  DiscardNull = False
  EventChar = 0
  Events = [evRxChar, evTxEmpty, evRxFlag, evRing, evBreak, evCTS, evDSR,
evError, evRLSD, evRx80Full]
  Buffer.InputSize = 1024
  Buffer.OutputSize = 1024
  FlowControl.OutCTSFlow = True
  FlowControl.OutDSRFlow = True
  FlowControl.ControlDTR = dtrEnable
  FlowControl.ControlRTS = rtsHandshake
  FlowControl.XonXoffOut = True
  FlowControl.XonXoffIn = True
  FlowControl.DSRsensitivity = False
  FlowControl.TxContinueOnXoff = False
  FlowControl.XonChar = 17
  FlowControl.XoffChar = 19
  Timeouts.ReadInterval = 20
  Timeouts.ReadTotalMultiplier = 20
  Timeouts.ReadTotalConstant = 20
  Timeouts.WriteTotalMultiplier = 100
  Timeouts.WriteTotalConstant = 1000
  SyncMethod = smThreadSync
  Left = 416
  Top = 224
```

```
end
object TScrute: TTimer
  Enabled = False
  Interval = 5000
  OnTimer = TScruteTimer
  Left = 448
  Top = 224
end
object OpenFileDialog: TOpenDialog
  Filter = 'Texte|*.txt'
  Title = 'Fichier Journal...'
  Left = 384
  Top = 224
end
object MainMenu: TMainMenu
  Left = 352
  Top = 224
  object Fichier1: TMenuItem
    Caption = 'Passerelle'
    object ConfigurerlapasserelleMainMenu: TMenuItem
      Bitmap.Data = {...}
      Caption = 'Configurer la passerelle...'
      OnClick = ConfigurerlapasserelleMainMenuClick
    end
    object VoirlejournalMainMenu: TMenuItem
      Caption = 'Voir le journal...'
      OnClick = VoirlejournalMainMenuClick
    end
    object TestdelenvoiMainMenu: TMenuItem
      Caption = 'Test de l'#39'envoi...'
      OnClick = TestdelenvoiMainMenuClick
    end
    object QuitterMainMenu: TMenuItem
      Caption = 'Quitter'
      OnClick = QuitterMainMenuClick
    end
  end
  object Aidel: TMenuItem
    Caption = 'Aide'
    object PrincipalMainMenu: TMenuItem
      Caption = 'Principal...'
      OnClick = PrincipalMainMenuClick
    end
    object AproposMainMenu: TMenuItem
      Caption = 'A propos...'
      OnClick = AproposMainMenuClick
    end
  end
end
object DataSource: TDataSource
  DataSet = Table
  Left = 288
  Top = 224
end
object Table: TTable
  TableName = 'Database'
  TableType = ttParadox
  Left = 256
  Top = 224
end
end
```

3.14 Configuration: Tconfiguration

```
object Configuration: TConfiguration
  Left = 337
  Top = 292
  Width = 413
  Height = 410
  Caption = 'Configuration'
  Color = clBtnFace
  Font.Charset = DEFAULT_CHARSET
  Font.Color = clWindowText
  Font.Height = -11
  Font.Name = 'MS Sans Serif'
  Font.Style = []
  OldCreateOrder = False
  Position = poDesktopCenter
  OnCreate = FormCreate
  PixelsPerInch = 96
  TextHeight = 13
object BOk: TButton
  Left = 11
  Top = 352
  Width = 126
  Height = 25
  Anchors = [akLeft]
  Caption = 'OK'
  Font.Charset = DEFAULT_CHARSET
  Font.Color = clWindowText
  Font.Height = -13
  Font.Name = 'MS Sans Serif'
  Font.Style = []
  ParentFont = False
  TabOrder = 0
  OnClick = BOkClick
end
object PCConfig: TPageControl
  Left = 8
  Top = 8
  Width = 393
  Height = 337
  ActivePage = TSGeneral
  Anchors = [akLeft, akTop, akRight, akBottom]
  TabOrder = 1
  object TSGeneral: TTabSheet
    Caption = 'Général'
    object GRTemps: TGroupBox
      Left = 14
      Top = 192
      Width = 353
      Height = 105
      Caption = 'Différé'
      Font.Charset = DEFAULT_CHARSET
      Font.Color = clWindowText
      Font.Height = -13
      Font.Name = 'MS Sans Serif'
      Font.Style = []
      ParentFont = False
      TabOrder = 3
      object LDateRemise: TLabel
        Left = 16
        Top = 56
        Width = 106
        Height = 16
        Caption = 'Date de la remise'
      end
      object LTempsGarde: TLabel
        Left = 192
```

```
    Top = 56
    Width = 115
    Height = 16
    Caption = 'Temps de la garde'
end
object CHBDiffere: TCheckBox
    Left = 16
    Top = 24
    Width = 129
    Height = 17
    Hint = 'Message différé dans le temps'
    Caption = 'Message différé ?'
    ParentShowHint = False
    ShowHint = True
    TabOrder = 0
    OnClick = CHBDiffereClick
end
object MEDate: TMaskEdit
    Left = 16
    Top = 72
    Width = 145
    Height = 24
    Hint = 'Date de la remise du message. Format : DDMMYYHHmm'
    Enabled = False
    EditMask = '9999999999;1;_'
    MaxLength = 10
    ParentShowHint = False
    ShowHint = True
    TabOrder = 1
    Text = '          '
end
object MEGarde: TMaskEdit
    Left = 192
    Top = 72
    Width = 145
    Height = 24
    Hint = 'Temps de garde du message différé'
    Enabled = False
    EditMask = '9999999999;1;_'
    MaxLength = 10
    ParentShowHint = False
    ShowHint = True
    TabOrder = 2
    Text = '          '
end
end
object GRModem: TGroupBox
    Left = 14
    Top = 64
    Width = 353
    Height = 121
    Caption = 'Configuration du modem'
    Font.Charset = DEFAULT_CHARSET
    Font.Color = clWindowText
    Font.Height = -13
    Font.Name = 'MS Sans Serif'
    Font.Style = []
    ParentFont = False
    TabOrder = 2
    object LInitModem: TLabel
        Left = 8
        Top = 32
        Width = 136
        Height = 16
        Caption = 'Initialisation du modem'
    end
end
object LSMSC: TLabel
    Left = 176
```

```
    Top = 16
    Width = 38
    Height = 16
    Caption = 'SMSC'
    Font.Charset = DEFAULT_CHARSET
    Font.Color = clWindowText
    Font.Height = -13
    Font.Name = 'MS Sans Serif'
    Font.Style = []
    ParentFont = False
end
object BConfigCom: TButton
    Left = 176
    Top = 88
    Width = 169
    Height = 25
    Caption = 'Configuration du port série'
    Font.Charset = DEFAULT_CHARSET
    Font.Color = clWindowText
    Font.Height = -13
    Font.Name = 'MS Sans Serif'
    Font.Style = []
    ParentFont = False
    TabOrder = 2
    OnClick = BConfigComClick
end
object EInit: TEdit
    Left = 8
    Top = 48
    Width = 145
    Height = 24
    Hint = 'Commande AT utilisée pour initialiser le modem'
    ParentShowHint = False
    ShowHint = True
    TabOrder = 0
    Text = 'ATZ'
end
object BRecherche: TButton
    Left = 8
    Top = 88
    Width = 161
    Height = 25
    Caption = 'Recherche automatique'
    TabOrder = 1
    OnClick = BRechercheClick
end
object CBOperateur: TComboBox
    Left = 176
    Top = 32
    Width = 169
    Height = 21
    Color = clMenu
    Font.Charset = DEFAULT_CHARSET
    Font.Color = clWindowText
    Font.Height = -11
    Font.Name = 'MS Sans Serif'
    Font.Style = []
    ItemHeight = 13
    ParentFont = False
    TabOrder = 3
    Text = 'Suisse - Swisscom'
    OnChange = CBOperateurChange
end
object CBNumOperateur: TComboBox
    Left = 176
    Top = 56
    Width = 169
    Height = 21
```

```
Color = clMenu
Font.Charset = DEFAULT_CHARSET
Font.Color = clWindowText
Font.Height = -11
Font.Name = 'MS Sans Serif'
Font.Style = []
ItemHeight = 13
ParentFont = False
TabOrder = 4
Text = '0041794998990'
OnChange = CBNumOperateurChange
end
end
object GRQuittance: TRadioGroup
Left = 232
Top = 8
Width = 137
Height = 49
Hint = 'Renvoi un mail à l'#39'expéditeur'
Caption = 'Quittance'
Columns = 2
Enabled = False
Font.Charset = DEFAULT_CHARSET
Font.Color = clWindowText
Font.Height = -13
Font.Name = 'MS Sans Serif'
Font.Style = []
ItemIndex = 1
Items.Strings = (
'Oui'
'Non')
ParentFont = False
ParentShowHint = False
ShowHint = True
TabOrder = 1
end
object GREnvoi: TRadioGroup
Left = 16
Top = 8
Width = 153
Height = 49
Caption = 'Envoi par'
Columns = 2
Font.Charset = DEFAULT_CHARSET
Font.Color = clWindowText
Font.Height = -13
Font.Name = 'MS Sans Serif'
Font.Style = []
ItemIndex = 0
Items.Strings = (
'Modem'
'Natel')
ParentFont = False
TabOrder = 0
OnClick = GREnvoiClick
end
end
object TSPasserelle: TTabSheet
Caption = 'Passerelle'
ImageIndex = 1
object LNumExpediteur: TLabel
Left = 8
Top = 8
Width = 139
Height = 16
Caption = 'Numéro de l'#39'expéditeur'
Font.Charset = DEFAULT_CHARSET
Font.Color = clWindowText
```

```
Font.Height = -13
Font.Name = 'MS Sans Serif'
Font.Style = []
ParentFont = False
end
object LProfil: TLabel
Left = 224
Top = 8
Width = 79
Height = 16
Caption = 'Nom du profil'
Font.Charset = DEFAULT_CHARSET
Font.Color = clWindowText
Font.Height = -13
Font.Name = 'MS Sans Serif'
Font.Style = []
ParentFont = False
end
object EOriginator: TEdit
Left = 8
Top = 24
Width = 121
Height = 24
Hint = 'Numéro que le destinataire recevra avec le SMS'
Font.Charset = DEFAULT_CHARSET
Font.Color = clWindowText
Font.Height = -13
Font.Name = 'MS Sans Serif'
Font.Style = []
ParentFont = False
ParentShowHint = False
ShowHint = True
TabOrder = 0
Text = '1234567890'
end
object EProfil: TEdit
Left = 224
Top = 24
Width = 121
Height = 24
Hint = 'Nom du profil de la messagerie'
Font.Charset = DEFAULT_CHARSET
Font.Color = clWindowText
Font.Height = -13
Font.Name = 'MS Sans Serif'
Font.Style = []
ParentFont = False
ParentShowHint = False
ShowHint = True
TabOrder = 1
Text = 'sms'
end
object GBTemps: TGroupBox
Left = 8
Top = 60
Width = 369
Height = 157
Hint = 'Relève périodique des messages'
Caption = 'Scrutation tous les :'
Font.Charset = DEFAULT_CHARSET
Font.Color = clWindowText
Font.Height = -13
Font.Name = 'MS Sans Serif'
Font.Style = []
ParentFont = False
ParentShowHint = False
ShowHint = True
TabOrder = 2
```

```
object LJour: TLabel
  Left = 16
  Top = 16
  Width = 26
  Height = 16
  Caption = 'Jour'
end
object LHeures: TLabel
  Left = 200
  Top = 16
  Width = 37
  Height = 16
  Caption = 'Heure'
end
object LMinutes: TLabel
  Left = 16
  Top = 64
  Width = 39
  Height = 16
  Caption = 'Minute'
end
object LSecondes: TLabel
  Left = 16
  Top = 112
  Width = 69
  Height = 16
  Caption = 'LSecondes'
end
object TBJours: TTrackBar
  Left = 8
  Top = 32
  Width = 170
  Height = 25
  LineSize = 5
  Max = 48
  Orientation = trHorizontal
  PageSize = 1
  Frequency = 1
  Position = 0
  SelEnd = 0
  SelStart = 0
  TabOrder = 0
  TickMarks = tmBoth
  TickStyle = tsNone
  OnChange = TBJoursChange
end
object TBHeures: TTrackBar
  Left = 192
  Top = 32
  Width = 170
  Height = 25
  Max = 23
  Orientation = trHorizontal
  PageSize = 1
  Frequency = 1
  Position = 0
  SelEnd = 0
  SelStart = 0
  TabOrder = 1
  TickMarks = tmBoth
  TickStyle = tsNone
  OnChange = TBHeuresChange
end
object TBMinutes: TTrackBar
  Left = 8
  Top = 80
  Width = 353
  Height = 25
```

```
Max = 59
Orientation = trHorizontal
PageSize = 1
Frequency = 1
Position = 0
SelEnd = 0
SelStart = 0
TabOrder = 2
TickMarks = tmBoth
TickStyle = tsNone
OnChange = TBMinutesChange
end
object TBSecondes: TTrackBar
Left = 8
Top = 128
Width = 353
Height = 25
Max = 59
Min = 1
Orientation = trHorizontal
Frequency = 1
Position = 10
SelEnd = 0
SelStart = 0
TabOrder = 3
TickMarks = tmBoth
TickStyle = tsNone
OnChange = TBSecondesChange
end
end
object GBEffaceMail: TGroupBox
Left = 176
Top = 232
Width = 201
Height = 65
Caption = 'Efface les mails datant d'#39'avant :'
Font.Charset = DEFAULT_CHARSET
Font.Color = clWindowText
Font.Height = -13
Font.Name = 'MS Sans Serif'
Font.Style = []
ParentFont = False
TabOrder = 3
Visible = False
object LJour: TLabel
Left = 8
Top = 16
Width = 26
Height = 16
Caption = 'Jour'
end
object LMois: TLabel
Left = 64
Top = 16
Width = 29
Height = 16
Caption = 'Mois'
end
object LAnnee: TLabel
Left = 112
Top = 16
Width = 39
Height = 16
Caption = 'Année'
end
object EJour: TEdit
Left = 8
Top = 32
```

```
Width = 25
Height = 24
Font.Charset = DEFAULT_CHARSET
Font.Color = clWindowText
Font.Height = -13
Font.Name = 'MS Sans Serif'
Font.Style = []
ParentFont = False
TabOrder = 0
Text = '06'
end
object EMois: TEdit
Left = 64
Top = 32
Width = 25
Height = 24
Font.Charset = DEFAULT_CHARSET
Font.Color = clWindowText
Font.Height = -13
Font.Name = 'MS Sans Serif'
Font.Style = []
ParentFont = False
TabOrder = 1
Text = '09'
end
object EAnnee: TEdit
Left = 112
Top = 32
Width = 41
Height = 24
Font.Charset = DEFAULT_CHARSET
Font.Color = clWindowText
Font.Height = -13
Font.Name = 'MS Sans Serif'
Font.Style = []
ParentFont = False
TabOrder = 2
Text = '2002'
end
end
object CBAActiveEfface: TCheckBox
Left = 16
Top = 264
Width = 145
Height = 17
Caption = 'Activer l'#39'effacement ?'
Font.Charset = DEFAULT_CHARSET
Font.Color = clWindowText
Font.Height = -13
Font.Name = 'MS Sans Serif'
Font.Style = []
ParentFont = False
TabOrder = 4
OnClick = CBAActiveEffaceClick
end
end
object TSAvance: TTabSheet
Caption = 'Avancé'
ImageIndex = 2
object GRAvance: TGroupBox
Left = 8
Top = 8
Width = 369
Height = 185
Caption = 'Divers'
Font.Charset = DEFAULT_CHARSET
Font.Color = clWindowText
Font.Height = -13
```

```
Font.Name = 'MS Sans Serif'
Font.Style = []
ParentFont = False
TabOrder = 0
object LAttenteCOM: TLabel
  Left = 8
  Top = 16
  Width = 148
  Height = 16
  Caption = 'Temps d'#39'attente du COM'
  Font.Charset = DEFAULT_CHARSET
  Font.Color = clWindowText
  Font.Height = -13
  Font.Name = 'MS Sans Serif'
  Font.Style = []
  ParentFont = False
end
object LErreurMax: TLabel
  Left = 192
  Top = 16
  Width = 143
  Height = 16
  Caption = 'Nombres d'#39'erreurs max.'
end
object LTempsConnexion: TLabel
  Left = 8
  Top = 72
  Width = 157
  Height = 16
  Caption = 'Temps de connexion max.'
end
object LAdresseAdmin: TLabel
  Left = 8
  Top = 136
  Width = 163
  Height = 16
  Caption = 'Adresse de l'#39'administrateur'
end
object ETempsAttenteCOM: TEdit
  Left = 8
  Top = 32
  Width = 41
  Height = 24
  Hint =
    'Petit pour un ordinateur puissant et grand pour un ordinateur le' +
    'nt. Temps en ms'
  Font.Charset = DEFAULT_CHARSET
  Font.Color = clWindowText
  Font.Height = -13
  Font.Name = 'MS Sans Serif'
  Font.Style = []
  ParentFont = False
  ParentShowHint = False
  ShowHint = True
  TabOrder = 0
  Text = '200'
end
object ENbreErreurPossible: TEdit
  Left = 192
  Top = 32
  Width = 33
  Height = 24
  Hint =
    'Nombre d'#39'erreurs maximum que l'#39'application accepte pour se
conne' +
    'cter'
  Font.Charset = DEFAULT_CHARSET
  Font.Color = clWindowText
```

```
Font.Height = -13
Font.Name = 'MS Sans Serif'
Font.Style = []
ParentFont = False
ParentShowHint = False
ShowHint = True
TabOrder = 1
Text = '10'
end
object UpDown1: TUpDown
Left = 225
Top = 32
Width = 15
Height = 24
Associate = ENbreErreurPossible
Min = 1
Max = 50
Position = 10
TabOrder = 2
Wrap = False
end
object ETempsConnexion: TEdit
Left = 8
Top = 88
Width = 49
Height = 24
TabOrder = 3
Text = '30000'
end
object EAdresseAdmin: TEdit
Left = 8
Top = 152
Width = 153
Height = 24
TabOrder = 4
Text = 'smsadmin@est.cpln.ch'
end
object RGMobile: TRadioGroup
Left = 192
Top = 72
Width = 145
Height = 57
Caption = 'Mobile :'
ItemIndex = 0
Items.Strings = (
'Normal'
'Nokia 3210->8210')
TabOrder = 5
end
end
end
object GBJournal: TGroupBox
Left = 8
Top = 208
Width = 369
Height = 73
Caption = 'Journal'
Font.Charset = DEFAULT_CHARSET
Font.Color = clWindowText
Font.Height = -13
Font.Name = 'MS Sans Serif'
Font.Style = []
ParentFont = False
TabOrder = 1
object LNbreOccurences: TLabel
Left = 8
Top = 24
Width = 175
Height = 16
```

```
Caption = 'Nbre d'#39'occurences du journal'
Font.Charset = DEFAULT_CHARSET
Font.Color = clWindowText
Font.Height = -13
Font.Name = 'MS Sans Serif'
Font.Style = []
ParentFont = False
end
object LNomBD: TLabel
  Left = 200
  Top = 25
  Width = 164
  Height = 16
  Caption = 'Nom de la base de donnée'
  Font.Charset = DEFAULT_CHARSET
  Font.Color = clWindowText
  Font.Height = -13
  Font.Name = 'MS Sans Serif'
  Font.Style = []
  ParentFont = False
end
object ENbreOccurence: TEdit
  Left = 8
  Top = 41
  Width = 41
  Height = 24
  Font.Charset = DEFAULT_CHARSET
  Font.Color = clWindowText
  Font.Height = -13
  Font.Name = 'MS Sans Serif'
  Font.Style = []
  ParentFont = False
  TabOrder = 0
  Text = '500'
end
object ENomBD: TEdit
  Left = 200
  Top = 41
  Width = 121
  Height = 24
  Font.Charset = DEFAULT_CHARSET
  Font.Color = clWindowText
  Font.Height = -13
  Font.Name = 'MS Sans Serif'
  Font.Style = []
  ParentFont = False
  TabOrder = 1
  Text = 'DataBase'
end
end
end
end
object BAide: TButton
  Left = 272
  Top = 352
  Width = 129
  Height = 25
  Anchors = [akLeft]
  Caption = 'Aide'
  Font.Charset = DEFAULT_CHARSET
  Font.Color = clWindowText
  Font.Height = -13
  Font.Name = 'MS Sans Serif'
  Font.Style = []
  ParentFont = False
  TabOrder = 2
  OnClick = BAideClick
end
```

```

object CPAuto: TComPort
  BaudRate = br9600
  Port = COM1
  Parity.Bits = prNone
  Parity.Check = False
  Parity.Replace = False
  Parity.ReplaceChar = 0
  StopBits = sbOneStopBit
  DataBits = dbEight
  DiscardNull = False
  EventChar = 0
  Events = [evRxChar, evTxEmpty, evRxFlag, evRing, evBreak, evCTS, evDSR,
evError, evRLSD, evRx80Full]
  Buffer.InputSize = 1024
  Buffer.OutputSize = 1024
  FlowControl.OutCTSFlow = True
  FlowControl.OutDSRFlow = True
  FlowControl.ControlDTR = dtrEnable
  FlowControl.ControlRTS = rtsHandshake
  FlowControl.XonXoffOut = True
  FlowControl.XonXoffIn = True
  FlowControl.DSRsensitivity = False
  FlowControl.TxContinueOnXoff = False
  FlowControl.XonChar = 17
  FlowControl.XoffChar = 19
  Timeouts.ReadInterval = -1
  Timeouts.ReadTotalMultiplier = 0
  Timeouts.ReadTotalConstant = 0
  Timeouts.WriteTotalMultiplier = 100
  Timeouts.WriteTotalConstant = 1000
  SyncMethod = smThreadSync
  Left = 184
  Top = 352
end
end
end

```

3.15 SelectPortCom: TselectPortCom

```

object SelectPortCom: TSelectPortCom
  Left = 602
  Top = 388
  Width = 198
  Height = 206
  BorderIcons = [biMinimize, biMaximize]
  Caption = 'SelectPortCom'
  Color = clBtnFace
  Font.Charset = DEFAULT_CHARSET
  Font.Color = clWindowText
  Font.Height = -11
  Font.Name = 'MS Sans Serif'
  Font.Style = []
  OldCreateOrder = False
  OnActivate = FormActivate
  PixelsPerInch = 96
  TextHeight = 13
  object RGCOM: TRadioGroup
    Left = 16
    Top = 8
    Width = 161
    Height = 129
    Caption = 'Port série'
    Columns = 2
    Items.Strings = (
      'COM1'
      'COM2'
      'COM3'
      'COM4'
      'COM5'
    )
  end
end

```

```

    'COM6'
    'COM7'
    'COM8')
    TabOrder = 0
    OnClick = RGCOMClick
end
object BOK: TButton
    Left = 16
    Top = 144
    Width = 75
    Height = 25
    Anchors = [akLeft]
    Caption = 'OK'
    Enabled = False
    TabOrder = 1
    OnClick = BOKClick
end
object BAnnule: TButton
    Left = 104
    Top = 144
    Width = 75
    Height = 25
    Caption = 'Annule'
    TabOrder = 2
    OnClick = BAnnuleClick
end
end
end

```

3.16 Imprime: TImprime

```

object Imprime: TImprime
    Left = 290
    Top = 173
    Width = 696
    Height = 481
    Caption = 'Imprime'
    Color = clBtnFace
    Font.Charset = DEFAULT_CHARSET
    Font.Color = clWindowText
    Font.Height = -11
    Font.Name = 'MS Sans Serif'
    Font.Style = []
    OldCreateOrder = False
    Scaled = False
    PixelsPerInch = 96
    TextHeight = 13
    object Rapport: TQuickRep
        Left = 0
        Top = 0
        Width = 794
        Height = 1123
        Frame.Color = clBlack
        Frame.DrawTop = False
        Frame.DrawBottom = False
        Frame.DrawLeft = False
        Frame.DrawRight = False
        DataSet = Principal.Table
        Font.Charset = DEFAULT_CHARSET
        Font.Color = clWindowText
        Font.Height = -13
        Font.Name = 'Arial'
        Font.Style = []
        Functions.Strings = (
            'PAGENUMBER'
            'COLUMNNUMBER'
            'REPORTTITLE'
            'QRSTRINGSBAND1')
        Functions.DATA = (

```

```
'0'  
'0'  
#39#39  
#39#39)  
Options = [FirstPageHeader, LastPageFooter]  
Page.Columns = 1  
Page.Orientation = poPortrait  
Page.PaperSize = A4  
Page.Values = (  
  100  
  2970  
  100  
  2100  
  100  
  100  
  0)  
PrinterSettings.Copies = 1  
PrinterSettings.Duplex = False  
PrinterSettings.FirstPage = 0  
PrinterSettings.LastPage = 0  
PrinterSettings.OutputBin = Auto  
PrintIfEmpty = True  
SnapToGrid = True  
Units = MM  
Zoom = 100  
object ColumnHeaderBand1: TQRBand  
  Left = 38  
  Top = 105  
  Width = 718  
  Height = 24  
  Frame.Color = clBlack  
  Frame.DrawTop = False  
  Frame.DrawBottom = False  
  Frame.DrawLeft = False  
  Frame.DrawRight = False  
  AlignToBottom = False  
  Color = clScrollBar  
  ForceNewColumn = False  
  ForceNewPage = False  
  Size.Values = (  
    63.5  
    1899.708333333333)  
BandType = rbColumnHeader  
object QRLabel2: TQRLabel  
  Left = 8  
  Top = 0  
  Width = 35  
  Height = 26  
  Frame.Color = clBlack  
  Frame.DrawTop = False  
  Frame.DrawBottom = False  
  Frame.DrawLeft = False  
  Frame.DrawRight = False  
  Size.Values = (  
    68.79166666666667  
    21.16666666666667  
    0  
    92.60416666666667)  
Alignment = taLeftJustify  
AlignToBand = False  
AutoSize = True  
AutoStretch = False  
Caption = 'Date'  
Color = clWhite  
Font.Charset = DEFAULT_CHARSET  
Font.Color = clWindowText  
Font.Height = -16  
Font.Name = 'Arial'
```

```
Font.Style = [fsUnderline]
ParentFont = False
Transparent = True
WordWrap = True
FontSize = 12
end
object QRLabel3: TQRLabel
  Left = 88
  Top = 0
  Width = 43
  Height = 26
  Frame.Color = clBlack
  Frame.DrawTop = False
  Frame.DrawBottom = False
  Frame.DrawLeft = False
  Frame.DrawRight = False
  Size.Values = (
    68.7916666666667
    232.833333333333
    0
    113.770833333333)
  Alignment = taLeftJustify
  AlignToBand = False
  AutoSize = True
  AutoStretch = False
  Caption = 'Heure'
  Color = clWhite
  Font.Charset = DEFAULT_CHARSET
  Font.Color = clWindowText
  Font.Height = -16
  Font.Name = 'Arial'
  Font.Style = [fsUnderline]
  ParentFont = False
  Transparent = True
  WordWrap = True
  FontSize = 12
end
object QRLabel4: TQRLabel
  Left = 168
  Top = 0
  Width = 81
  Height = 26
  Frame.Color = clBlack
  Frame.DrawTop = False
  Frame.DrawBottom = False
  Frame.DrawLeft = False
  Frame.DrawRight = False
  Size.Values = (
    68.7916666666667
    444.5
    0
    214.3125)
  Alignment = taLeftJustify
  AlignToBand = False
  AutoSize = True
  AutoStretch = False
  Caption = 'Description'
  Color = clWhite
  Font.Charset = DEFAULT_CHARSET
  Font.Color = clWindowText
  Font.Height = -16
  Font.Name = 'Arial'
  Font.Style = [fsUnderline]
  ParentFont = False
  Transparent = True
  WordWrap = True
  FontSize = 12
end
```

```
end
object DetailBand1: TQRBand
  Left = 38
  Top = 129
  Width = 718
  Height = 16
  Frame.Color = clBlack
  Frame.DrawTop = False
  Frame.DrawBottom = False
  Frame.DrawLeft = False
  Frame.DrawRight = False
  AlignToBottom = False
  Color = clWhite
  ForceNewColumn = False
  ForceNewPage = False
  Size.Values = (
    42.33333333333333
    1899.708333333333)
  BandType = rbDetail
object QRDBText1: TQRDBText
  Left = 8
  Top = 0
  Width = 28
  Height = 20
  Frame.Color = clBlack
  Frame.DrawTop = False
  Frame.DrawBottom = False
  Frame.DrawLeft = False
  Frame.DrawRight = False
  Size.Values = (
    52.91666666666667
    21.16666666666667
    0
    74.08333333333333)
  Alignment = taLeftJustify
  AlignToBand = False
  AutoSize = True
  AutoStretch = False
  Color = clWhite
  DataSet = Principal.Table
  DataField = 'Date'
  Font.Charset = DEFAULT_CHARSET
  Font.Color = clWindowText
  Font.Height = -13
  Font.Name = 'Arial'
  Font.Style = []
  ParentFont = False
  Transparent = False
  WordWrap = True
  FontSize = 10
end
object QRDBText2: TQRDBText
  Left = 88
  Top = 0
  Width = 35
  Height = 20
  Frame.Color = clBlack
  Frame.DrawTop = False
  Frame.DrawBottom = False
  Frame.DrawLeft = False
  Frame.DrawRight = False
  Size.Values = (
    52.91666666666667
    232.8333333333333
    0
    92.60416666666667)
  Alignment = taLeftJustify
  AlignToBand = False
```

```
AutoSize = True
AutoStretch = False
Color = clWhite
DataSet = Principal.Table
DataField = 'Heure'
Font.Charset = DEFAULT_CHARSET
Font.Color = clWindowText
Font.Height = -13
Font.Name = 'Arial'
Font.Style = []
ParentFont = False
Transparent = False
WordWrap = True
FontSize = 10
end
object QRDBText3: TQRDBText
  Left = 168
  Top = 0
  Width = 577
  Height = 20
  Frame.Color = clBlack
  Frame.DrawTop = False
  Frame.DrawBottom = False
  Frame.DrawLeft = False
  Frame.DrawRight = False
  Size.Values = (
    52.9166666666667
    444.5
    0
    1526.64583333333)
  Alignment = taLeftJustify
  AlignToBand = False
  AutoSize = False
  AutoStretch = True
  Color = clWhite
  DataSet = Principal.Table
  DataField = 'Info'
  Font.Charset = DEFAULT_CHARSET
  Font.Color = clWindowText
  Font.Height = -13
  Font.Name = 'Arial'
  Font.Style = []
  ParentFont = False
  Transparent = False
  WordWrap = True
  FontSize = 10
end
end
object PageFooterBand1: TQRBand
  Left = 38
  Top = 145
  Width = 718
  Height = 40
  Frame.Color = clBlack
  Frame.DrawTop = True
  Frame.DrawBottom = False
  Frame.DrawLeft = False
  Frame.DrawRight = False
  AlignToBottom = False
  Color = clWhite
  ForceNewColumn = False
  ForceNewPage = False
  Size.Values = (
    105.833333333333
    1899.70833333333)
  BandType = rbPageFooter
  object QRLabel15: TQRLabel
    Left = 304
```

```
Top = 12
Width = 31
Height = 18
Frame.Color = clBlack
Frame.DrawTop = False
Frame.DrawBottom = False
Frame.DrawLeft = False
Frame.DrawRight = False
Size.Values = (
  47.625
  804.333333333333
  31.75
  82.0208333333333)
Alignment = taLeftJustify
AlignToBand = False
AutoSize = True
AutoStretch = False
Caption = 'Page'
Color = clWhite
Transparent = False
WordWrap = True
FontSize = 10
end
object QRSysData1: TQRSysData
Left = 336
Top = 12
Width = 55
Height = 18
Frame.Color = clBlack
Frame.DrawTop = False
Frame.DrawBottom = False
Frame.DrawLeft = False
Frame.DrawRight = False
Size.Values = (
  47.625
  889
  31.75
  145.520833333333)
Alignment = taLeftJustify
AlignToBand = False
AutoSize = True
Color = clWhite
Data = qrsPageNumber
Transparent = False
FontSize = 10
end
end
object TitleBand1: TQRBand
Left = 38
Top = 38
Width = 718
Height = 67
Frame.Color = clBlack
Frame.DrawTop = False
Frame.DrawBottom = False
Frame.DrawLeft = False
Frame.DrawRight = False
AlignToBottom = False
Color = clWhite
ForceNewColumn = False
ForceNewPage = False
Size.Values = (
  177.270833333333
  1899.70833333333)
BandType = rbTitle
object QRLabel1: TQRLabel
Left = 196
Top = 12
```

```

Width = 326
Height = 37
Frame.Color = clBlack
Frame.DrawTop = False
Frame.DrawBottom = False
Frame.DrawLeft = False
Frame.DrawRight = False
Size.Values = (
  97.89583333333333
  518.5833333333333
  31.75
  862.5416666666667)
Alignment = taLeftJustify
AlignToBand = False
AutoSize = True
AutoStretch = False
Caption = 'Journal des événements'
Color = clWhite
Font.Charset = ANSI_CHARSET
Font.Color = clWindowText
Font.Height = -32
Font.Name = 'Times New Roman'
Font.Style = [fsBold, fsUnderline]
ParentFont = False
Transparent = False
WordWrap = True
FontSize = 24
end
end
end
end
end

```

3.17 Test: Ttest

```

object Test: TTest
  Left = 684
  Top = 478
  Width = 291
  Height = 269
  Caption = 'Test d'#39'envoi d'#39'un SMS'
  Color = clBtnFace
  Font.Charset = DEFAULT_CHARSET
  Font.Color = clWindowText
  Font.Height = -11
  Font.Name = 'MS Sans Serif'
  Font.Style = []
  OldCreateOrder = False
  OnActivate = FormActivate
  PixelsPerInch = 96
  TextHeight = 13
  object LDestinataire: TLabel
    Left = 0
    Top = 16
    Width = 72
    Height = 16
    Caption = 'Destinataire'
    Font.Charset = DEFAULT_CHARSET
    Font.Color = clWindowText
    Font.Height = -13
    Font.Name = 'MS Sans Serif'
    Font.Style = []
    ParentFont = False
  end
  object LMessage: TLabel
    Left = 0
    Top = 40
    Width = 57
    Height = 16
  end
end

```

```
Caption = 'Message'
Font.Charset = DEFAULT_CHARSET
Font.Color = clWindowText
Font.Height = -13
Font.Name = 'MS Sans Serif'
Font.Style = []
ParentFont = False
end
object LCompte: TLabel
  Left = 232
  Top = 40
  Width = 3
  Height = 16
  Font.Charset = DEFAULT_CHARSET
  Font.Color = clWindowText
  Font.Height = -13
  Font.Name = 'MS Sans Serif'
  Font.Style = []
  ParentFont = False
end
object EDestinataire: TEdit
  Left = 80
  Top = 8
  Width = 121
  Height = 24
  Font.Charset = DEFAULT_CHARSET
  Font.Color = clWindowText
  Font.Height = -13
  Font.Name = 'MS Sans Serif'
  Font.Style = []
  ParentFont = False
  TabOrder = 0
  Text = '0787595022'
end
object MMessage: TMemo
  Left = 0
  Top = 56
  Width = 281
  Height = 145
  Anchors = [akLeft, akTop, akRight, akBottom]
  Font.Charset = DEFAULT_CHARSET
  Font.Color = clWindowText
  Font.Height = -13
  Font.Name = 'MS Sans Serif'
  Font.Style = []
  Lines.Strings = (
    'Essai...')
  ParentFont = False
  TabOrder = 1
  OnChange = MMessageChange
  OnKeyPress = MMessageKeyPress
end
object BEnvoie: TButton
  Left = 16
  Top = 208
  Width = 75
  Height = 25
  Anchors = [akLeft, akBottom]
  Caption = 'Envoie'
  Font.Charset = DEFAULT_CHARSET
  Font.Color = clWindowText
  Font.Height = -13
  Font.Name = 'MS Sans Serif'
  Font.Style = []
  ParentFont = False
  TabOrder = 2
  OnClick = BEnvoieClick
end
```

```
object BAide: TButton
  Left = 192
  Top = 208
  Width = 75
  Height = 25
  Anchors = [akLeft, akBottom]
  Caption = 'Aide'
  Font.Charset = DEFAULT_CHARSET
  Font.Color = clWindowText
  Font.Height = -13
  Font.Name = 'MS Sans Serif'
  Font.Style = []
  ParentFont = False
  TabOrder = 3
  OnClick = BAideClick
end
object BAnnule: TButton
  Left = 104
  Top = 208
  Width = 75
  Height = 25
  Anchors = [akLeft, akBottom]
  Caption = 'Annule'
  Font.Charset = DEFAULT_CHARSET
  Font.Color = clWindowText
  Font.Height = -13
  Font.Name = 'MS Sans Serif'
  Font.Style = []
  ParentFont = False
  TabOrder = 4
  OnClick = BAnnuleClick
end
end
```

3.18 Aide: Taide

```
object Aide: TAide
  Left = 465
  Top = 183
  Width = 492
  Height = 403
  Caption = 'Aide'
  Color = clBtnFace
  ParentFont = True
  OldCreateOrder = True
  Position = poScreenCenter
  OnActivate = FormActivate
  PixelsPerInch = 96
  TextHeight = 13
object Panel2: TPanel
  Left = 0
  Top = 342
  Width = 484
  Height = 34
  Align = alBottom
  BevelOuter = bvNone
  ParentColor = True
  TabOrder = 0
object OKBtn: TButton
  Left = 147
  Top = 2
  Width = 190
  Height = 25
  Anchors = [akLeft, akTop, akRight, akBottom]
  Caption = 'OK'
  Default = True
  ModalResult = 1
  TabOrder = 0
```

```
OnClick = OKBtnClick
end
end
object TabControl1: TTabControl
Left = 8
Top = 8
Width = 465
Height = 329
Anchors = [akLeft, akTop, akRight, akBottom]
TabOrder = 1
Tabs.Strings = (
'Principal'
'Journal'
'Configuration'
'Test')
TabIndex = 0
OnChange = TabControl1Change
object RichEdit1: TRichEdit
Left = 8
Top = 32
Width = 449
Height = 289
Anchors = [akLeft, akTop, akRight, akBottom]
Lines.Strings = (
'RichEdit1')
ReadOnly = True
ScrollBars = ssVertical
TabOrder = 0
end
end
end
```

3.19 APropos: TAPropos

```
object APropos: TAPropos
Left = 402
Top = 332
BorderStyle = bsDialog
Caption = 'A propos'
ClientHeight = 184
ClientWidth = 287
Color = clBtnFace
Font.Charset = DEFAULT_CHARSET
Font.Color = clWindowText
Font.Height = -11
Font.Name = 'MS Sans Serif'
Font.Style = []
OldCreateOrder = True
Position = poScreenCenter
PixelsPerInch = 96
TextHeight = 13
object Panell1: TPanel
Left = 8
Top = 8
Width = 273
Height = 137
BevelInner = bvRaised
BevelOuter = bvLowered
ParentColor = True
TabOrder = 0
object ProgramIcon: TImage
Left = 8
Top = 8
Width = 97
Height = 73
IncrementalDisplay = True
Picture.Data = {...}
Stretch = True
```

```
        IsControl = True
    end
    object LProductName: TLabel
        Left = 120
        Top = 16
        Width = 147
        Height = 24
        Caption = 'Passerelle SMS'
        Font.Charset = DEFAULT_CHARSET
        Font.Color = clWindowText
        Font.Height = -19
        Font.Name = 'MS Sans Serif'
        Font.Style = [fsBold]
        ParentFont = False
        IsControl = True
    end
    object LVersion: TLabel
        Left = 120
        Top = 40
        Width = 66
        Height = 16
        Caption = 'Version 1.0'
        Font.Charset = DEFAULT_CHARSET
        Font.Color = clWindowText
        Font.Height = -13
        Font.Name = 'MS Sans Serif'
        Font.Style = []
        ParentFont = False
        IsControl = True
    end
    object Comments: TLabel
        Left = 8
        Top = 96
        Width = 121
        Height = 13
        Caption = 'Crée par Segalla Jacques'
        WordWrap = True
        IsControl = True
    end
    object LCopyright: TLabel
        Left = 8
        Top = 112
        Width = 55
        Height = 13
        Caption = 'CPLN 2002'
    end
end
object OKButton: TButton
    Left = 111
    Top = 156
    Width = 75
    Height = 25
    Caption = 'OK'
    Default = True
    ModalResult = 1
    TabOrder = 1
    OnClick = OKButtonClick
end
end
```