Universidad de San Carlos de Guatemala
Facultad de Ingeniería
Escuela de Ingeniería en Ciencias y Sistemas

USO DE MENSAJES SMS, PARA CREACIÓN DE UN SISTEMA DE AVISO Y ALERTA PARA EMERGENCIAS MÉDICAS

Eric Adangumer Navarro Godínez
Asesorado por el Ing. Rubén Alfredo Barrios Toc

Guatemala, junio de 2014

UNIVERSIDAD DE SAN CARLOS DE GUATEMALA

FACULTAD DE INGENIERÍA

USO DE MENSAJES SMS, PARA CREACIÓN DE UN SISTEMA DE AVISO Y ALERTA PARA EMERGENCIAS MÉDICAS

TRABAJO DE GRADUACIÓN

PRESENTADO A LA JUNTA DIRECTIVA DE LA
FACULTAD DE INGENIERÍA
POR

ERIC ADANGUMER NAVARRO GODÍNEZ
ASESORADO POR EL ING. RUBÉN ALFREDO BARRIOS TOC

AL CONFERÍRSELE EL TÍTULO DE

INGENIERO EN CIENCIAS Y SISTEMAS

GUATEMALA, JUNIO DE 2014

UNIVERSIDAD DE SAN CARLOS DE GUATEMALA
FACULTAD DE INGENIERÍA

NÓMINA DE JUNTA DIRECTIVA

DECANO	Ing. Murphy Olympo Paiz Recinos
VOCAL I	Ing. Alfredo Enrique Beber Aceituno
VOCAL II	Ing. Pedro Antonio Aguilar Polanco
VOCAL III	Inga. Elvia Miriam Ruballos Samayoa
VOCAL IV	Br. Walter Rafael Véliz Muñoz
VOCAL V	Br. Sergio Alejandro Donis Soto
SECRETARIO	Ing. Hugo Humberto Rivera Pérez

TRIBUNAL QUE PRACTICÓ EL EXAMEN GENERAL PRIVADO

DECANO	Ing. Sydney Alexander Samuels Milson
EXAMINADOR	Ing. Marlon Antonio Pérez Türk
EXAMINADOR	Ing. Edgar René Ornelyz Hoil
EXAMINADOR	Ing. Luis Alberto Vettorazzi España
SECRETARIO	Ing. Pedro Antonio Aguilar Polanco

HONORABLE TRIBUNAL EXAMINADOR

En cumplimiento con los preceptos que establece la ley de la Universidad de San Carlos de Guatemala, presento a su consideración mi trabajo de graduación titulado:

USO DE MENSAJES SMS, PARA CREACIÓN DE UN SISTEMA DE AVISO Y ALERTA PARA EMERGENCIAS MÉDICAS

Tema que me fuera asignado por la Dirección de la Escuela de Ingeniería en Ciencias y Sistemas, con fecha octubre de 2013.

Eric Adangumer Navarro Godínez

Universidad de San Carlos de Guatemala
Facultad de Ingeniería
Carrera de Ciencias y Sistemas

Guatemala, 14 de abril del 2014

Ingeniero
Carlos Alfredo Azurdia Morales
Coordinador de Privados y Revisión de Trabajos de Graduación
Ingeniería en Ciencias y Sistemas

Respetable Ingeniero Azurdia:

Por medio de la presente, me permito informarle que he asesorado y revisado el trabajo de graduación titulado **USO DE MENSAJES SMS, PARA CREACIÓN DE UN SISTEMA DE AVISO Y ALERTA PARA EMERGENCIAS MÉDICAS**, elaborado por el estudiante **Eric Adangumer Navarro Godínez**, a mi consideración, cumple con los objetivos propuestos para su desarrollo.

Agradeciendo de antemano la atención que le preste a la presente, me suscribo de usted.

Atentamente,

Ing. Rubén Alfredo Barrios Toc
Asesor
Colegiado No. 7,504

Ruben Alfredo Barrios Toc
Ingeniero en Ciencias y Sistemas
Colegiado No. 7504

Universidad San Carlos de Guatemala
Facultad de Ingeniería
Escuela de Ingeniería en Ciencias y Sistemas

Guatemala, 30 de Abril de 2014

Ingeniero
Marlon Antonio Pérez Turk
Director de la Escuela de Ingeniería
En Ciencias y Sistemas

Respetable Ingeniero Pérez:

Por este medio hago de su conocimiento que he revisado el trabajo de graduación del estudiante **ERIC ADANGUMER NAVARRO GODINEZ** con carné **1998-12504,** titulado: **"USO DE MENSAJES SMS, PARA CREACIÓN DE UN SISTEMA DE AVISO Y ALERTA PARA EMERGENCIAS MÉDICAS",** y a mi criterio el mismo cumple con los objetivos propuestos para su desarrollo, según el protocolo.

Al agradecer su atención a la presente, aprovecho la oportunidad para suscribirme,

Atentamente,

Ing. Carlos Alfredo Azurdia
Coordinador de Privados
y Revisión de Trabajos de Graduación

ESCUELA DE CIENCIAS Y SISTEMAS

UNIVERSIDAD DE SAN CARLOS
DE GUATEMALA

FACULTAD DE INGENIERÌA
ESCUELA DE CIENCIAS Y SISTEMAS
TEL: 24767644

El Director de la Escuela de Ingeniería en Ciencias y Sistemas de la Facultad de Ingeniería de la Universidad de San Carlos de Guatemala, luego de conocer el dictamen del asesor con el visto bueno del revisor y del Licenciado en Letras, del trabajo de graduación ***"USO DE MENSAJES SMS, PARA CREACIÓN DE UN SISTEMA DE AVISO Y ALERTA PARA EMERGENCIAS MÉDICAS"****, realizado por el estudiante ERIC ADANGUMER NAVARRO GODÍNEZ, aprueba el presente trabajo y solicita la autorización del mismo.*

"ID Y ENSEÑAD A TODOS"

UNIVERSIDAD DE SAN CARLOS DE GUATEMALA
DIRECCION DE INGENIERIA EN CIENCIAS Y SISTEMAS

Ing. Marlon Antonio Pérez Türk
Director, Escuela de Ingeniería en Ciencias y Sistemas

Guatemala, 16 de junio 2014

Universidad de San Carlos
de Guatemala

Facultad de Ingeniería
Decanato

DTG. 282.2014

El Decano de la Facultad de Ingeniería de la Universidad de San Carlos de Guatemala, luego de conocer la aprobación por parte del Director de la Escuela de Ingeniería en Ciencias y Sistemas, al Trabajo de Graduación titulado: **USO DE MENSAJES SMS, PARA CREACIÓN DE UN SISTEMA DE AVISO Y ALERTA PARA EMERGENCIAS MÉDICAS,** presentado por el estudiante universitario **Eric Adangumer Navarro Godínez,** y después de haber culminado las revisiones previas bajo la responsabilidad de las instancias correspondientes, se autoriza la impresión del mismo.

IMPRÍMASE:

Ing. Murphy Olympo Paiz Recinos
Decano

UNIVERSIDAD DE SAN CARLOS DE GUATEMALA
DECANO
FACULTAD DE INGENIERÍA

Guatemala, 18 de junio de 2014

/gdech

ACTO QUE DEDICO A:

Dios	Que lo es todo, y todo lo debemos a él.
Mis padres	Juan Navarro y Yolanda Godínez. Por su guía y amor.
Hermanos	Por su apoyo y amistad que me han brindado siempre.
Familiares	Son un tesoro en la vida.
Amigos	Que me han acompañado en diversas aventuras en la vida.

AGRADECIMIENTOS A:

La Universidad de San Carlos de Guatemala	Casa de estudios que me brindó la oportunidad de convertirme en profesional.
Facultad de Ingeniería y catedráticos	Por sus enseñanzas en todos estos años y que compartieron su conocimiento.
Mis amigos de la Facultad	Rubén, José, Francisco, Paul, Kenneth, Nydia y Enner, por su apoyo, amistad y por compartir los años de la carrera.
Familiares y amigos	Que me han apoyado y alentado a culminar mi formación universitaria.
Save the Children USA Guatemala	Por haberme dado su apoyo y colaboración en la realización de este trabajo de graduación.

ÍNDICE GENERAL

ÍNDICE DE ILUSTRACIONES

FIGURAS

TABLAS

LISTA DE SÍMBOLOS

Símbolo	Significado
AI	Auto incremento
Kb/s	Kilo bit por segundo
PK	Llave primaria
ms	Milisegundos
NN	No nulo
CTS	Sello de tiempo actual o fecha y hora actual
Null	Valor nulo

GLOSARIO

Buffer	Un *buffer* de datos es un espacio de la memoria en un disco o en un instrumento digital reservada para el almacenamiento temporal de información digital, mientras que está esperando ser procesada.
Cadena	En general, una cadena de caracteres es una sucesión de caracteres (letras, números u otros signos o símbolos).
Carácter	Es una unidad de información que corresponde aproximadamente con un grafema o con una unidad o símbolo parecido, como los de un alfabeto o silabario de la forma escrita de un lenguaje natural.
CDMA	El CDMA separa las comunicaciones con códigos. La voz se descompone en bits digitalizados y los grupos de bits se etiquetan con un código. Cada código se asocia a una sola llamada en la red. Los grupos de bits de una llamada se transmiten aleatoriamente junto con los de las demás llamadas. Posteriormente se recolocan en el orden correcto para completar la conversación.

Código fuente El código fuente de un programa informático (o software) es un conjunto de líneas de texto que son las instrucciones que debe seguir la computadora para ejecutar dicho programa. Por tanto, en el código fuente de un programa está escrito por completo su funcionamiento.

Fax Aparato que permite transmitir por medio del cable telefónico documentos, fotografías y textos.

GPRS Técnica de transmisión de datos GSM que transmite y recibe datos en paquetes. GPRS ofrece una conexión permanente entre el dispositivo inalámbrico y la red (GPRS = *General Packet Radio Service*).

Hardware Se refiere a todas las partes tangibles de un sistema informático, sus componentes son: eléctricos, electrónicos, electromecánicos y mecánicos. Son cables, gabinetes o cajas, periféricos de todo tipo y cualquier otro elemento físico involucrado.

Html HTML, de las siglas de las inglés *HyperText Markup Language* en español lenguaje de marcas de hipertexto, hace referencia al lenguaje de marcado para la elaboración de páginas web.

IMEI	La IMEI (Identidad del Equipo Móvil Internacional) es un número individual, exclusivo de un teléfono concreto, que puede utilizarse para identificarlo. Los primeros seis dígitos de la IMEI identifican al fabricante y el modelo exacto, y los últimos dígitos identifican el teléfono concreto de dicha serie.
Internet	Internet es un conjunto descentralizado de redes de comunicación interconectadas que utilizan la familia de protocolos TCP/IP, lo cual garantiza que las redes físicas heterogéneas que la componen funcionen como una red lógica única, de alcance mundial.
IP	*Internet Protocol*, en español Protocolo de Internet o IP es un protocolo de comunicación de datos digitales clasificado funcionalmente en la Capa de Red según el modelo internacional OSI.
Lenguaje de programación	Es un lenguaje formal diseñado para expresar procesos que pueden ser llevados a cabo por máquinas como las computadoras.
Servidor	Una computadora en la que se ejecuta un programa que realiza alguna tarea en beneficio de otras aplicaciones llamadas clientes.

SMTP *Simple Mail Transfer Protocol* (Protocolo para la transferencia simple de correo electrónico), es un protocolo de red utilizado para el intercambio de mensajes de correo electrónico.

Software Se conoce como software al equipamiento lógico o soporte lógico de un sistema informático, que comprende el conjunto de los componentes lógicos necesarios que hacen posible la realización de tareas específicas.

SQL El lenguaje de consulta estructurado o SQL, por sus siglas en inglés *Structured Query Language*, es un lenguaje declarativo de acceso a bases de datos relacionales que permite especificar diversos tipos de operaciones en ellas.

SSL *Secure Sockets Layer*, en español capa de conexión segura, su sucesor *Transport Layer Security* son protocolos criptográficos que proporcionan comunicaciones seguras por una red, comúnmente Internet.

Telex Servicios de telegrafía automatizada con impresión, fueron establecidos sobre redes telefónicas conmutadas.

RESUMEN

La presente investigación demuestra de forma teórica y práctica, la manera de crear un Sistema de Aviso y Alerta de emergencias médicas, donde se utiliza los mensajes cortos o SMS como un medio de aviso. La logística de operación consiste en agentes monitores (personas de las comunidades) que dan la alerta de una emergencia, con la simple acción de enviar un mensaje SMS, dirigido al Sistema de Aviso y Alerta, el cual recibe el mensaje SMS a través de un módem GSM, donde es interpretado y procesado por el Sistema, y automáticamente genera un mensaje de alerta que es enviado a la organización u organizaciones responsables de atender dicha emergencia.

El primer capítulo estudia la tecnología GSM, servicios que provee, arquitectura, características, con la finalidad de comprender su funcionamiento, para aplicarlo en capítulos posteriores.

El segundo capítulo es una investigación sobre la forma de poder interactuar con la red de telecomunicaciones del país, a través de la manipulación de un dispositivo de hardware, para este objetivo un módem GSM.

En el tercer capítulo inicia la creación del Sistema de Aviso y Alerta, en este se diseñan los componentes del sistema, la funcionalidad, lógica de comportamiento y estructura y se modela a través de diagramas, para que facilite su comprensión. En el capítulo 4 se plasma el diseño realizado en el capítulo anterior y se crea el código fuente de la aplicación.

OBJETIVOS

General

Se pretende realizar un estudio sobre el Servicio de Mensajes Cortos (SMS) de teléfonos celulares, con un enfoque práctico, aplicado en el desarrollo de un sistema de aviso y alerta de emergencias médicas, para ser utilizado como una herramienta de comunicación rápida y efectiva, entre las personas afectadas y los responsables de atender estas eventualidades.

Específicos

1. Analizar las características y funcionalidad de la tecnología de mensajes SMS.

2. Establecer los recursos necesarios para crear la solución, tanto de hardware como de software.

3. Exponer soluciones o métodos alternos

4. Exponer otras posibles aplicaciones para la tecnología de SMS.

INTRODUCCIÓN

La tecnología como toda herramienta es tan buena como los usos para los cuales es destinado, el correcto uso puede tener un impacto positivo sobre la sociedad, brindando nuevos métodos o formas de hacer las cosas, hoy en día el uso que se le da a los teléfonos celulares, específicamente el envío y recepción de mensajes SMS, que es común, práctico y accesible para la población guatemalteca.

El enfoque de este trabajo de graduación propone la creación de un sistema de aviso y alerta para casos de emergencias médicas, haciendo uso de mensajes SMS. La idea se basa en satisfacer una necesidad utilizando tecnología como herramienta de comunicación, que puede ser percibida como un puente entre teoría de redes GSM y su usabilidad enfocada a satisfacer la necesidad de tener medios de comunicación en doble vía, con entidades, para informar sobre emergencias que deben ser atendidas con prontitud.

El sistema de comunicaciones móviles celular nacional, permite por medio de la red GSM el envío de mensajes cortos SMS, entre teléfonos celulares, esta característica puede ser utilizada como canal de comunicación para la creación del sistema de alerta, el proceso se inicia cuando se envía un mensaje SMS a un número específico, este mensaje es recibido e interpretado por una computadora o central, y se genera un mensaje SMS de alerta a las entidades responsables de dicha eventualidad para que tomen cartas en el asunto y que deben de darle seguimiento al caso reportado. Las entidades responsables son aquellos que deben de atender los casos y son previamente definidos.

Esta idea nace en Save the Children USA Guatemala, organización no lucrativa que vela por el bienestar de los niños, donde el director regional de esta organización Carlos Cárdenas, identifica la necesidad de crear un canal de comunicación con familias de comunidades rurales y de escasos recursos, un buen porcentaje ubicadas en el llamado Corredor Seco del país, y aplicarlo a programas de asistencia en materia de Salud y Nutrición. Así este canal de comunicación funcione como una herramienta para poder atender con prontitud las emergencias que pongan en riesgo la salud de niños de esta región, y asimismo ayudar a mitigar el impacto de la crisis de hambre y salud existente en estas comunidades.

Aunque en el título se especifica emergencias médicas, la logística propuesta se puede aplicarse a cualquier tipo de avisos de emergencias, que puede ser utilizado por las instituciones responsables que atienden estas, algunas de estas organizaciones puede ser el Ministerio de Salud, Care de Guatemala, organización no lucrativa que vela por la erradicación de la pobreza y desnutrición, entre otras organizaciones con similitudes operacionales.

Estas emergencias pueden ser de diferente índole, como lo son las emergencias médicas, avisos o denuncias sobre violaciones a nuestros derechos, avisos enfocados a informar a unas entidades específicas, sobre alguna problemática, como lo podría ser el anuncio del tráfico, accidentes de tránsito o noticias. Como resultado de los registros de estas emergencias o eventualidades, se puede obtener un conjunto de estadísticas, que con el estudio y análisis de los datos, pueden contribuir a comprender la problemática, prever y actuar de forma oportuna, para contrarrestar o minimizar los riesgos que están acompañadas a cada situación.

1. TECNOLOGÍA GSM

1.1. Sistema global de comunicaciones móviles GSM

GSM es un sistema estándar, libre utilizado en telefonía móvil digital, que a partir del siglo XXI es el estándar más usado en Europa, América del Sur, Asia y Oceanía, también se denomina segunda generación 2G, porque a diferencia de la primera generación, las comunicaciones se producen de un modo completamente digital. El estándar GSM fue desarrollado a partir de 1982. En la conferencia de telecomunicaciones CEPT de ese año fue creado el grupo de trabajo Groupe Spécial Mobile o GSM, cuya tarea era desarrollar un estándar europeo de telefonía móvil digital.

En 1990 se finalizaron las especificaciones para de la primera versión del estándar GSM-900, al que siguió DCS-1800 un año más tarde. En 1991 fueron presentados los primeros equipos de telefonía GSM como prototipos. De manera paralela, se cambió el nombre del grupo a Standard Mobile Group (SMG) y las siglas GSM a partir de este momento se usaron para el propio estándar. El estándar GSM permite un rendimiento máximo de 9,6 kbps, que permite transmisiones de voz y de datos digitales de volumen bajo, por ejemplo, mensajes de texto (SMS, Servicio de mensajes cortos) o mensajes multimedia (MMS, Servicio de mensajes multimedia).

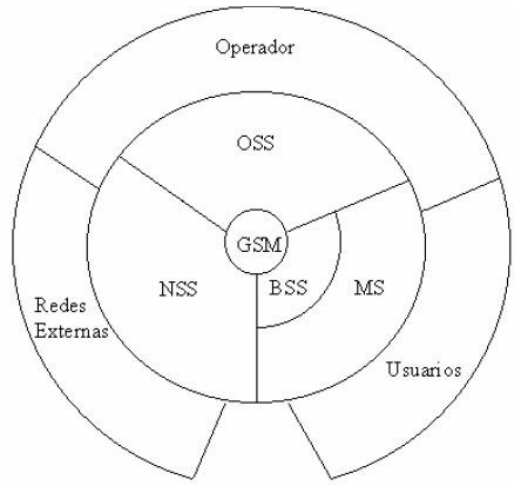
1.1.1. Servicios soportados por GSM

GSM presta los servicios de transmisión y recepción de voz de alta calidad, servicio de transmisión y recepción de datos conmutados por circuitos, en una amplia gama de bandas y por último, el servicio de transmisión y recepción de mensajes cortos SMS.

1.2. Arquitectura de red GSM

En la estructura de una red GSM está compuesta de 3 subsistemas o interfaces que interactúan entre sí para formar la vía de comunicación entre equipos móviles, también establece comunicación de equipos de distintos proveedores a través de interfaces abiertas y normalizadas. En la siguiente figura se muestra un diagrama de esta arquitectura.

Figura 1. **Arquitectura GSM**



Fuente: http://1.bp.blogspot.com/_IdGJUBz7Pik/S_wwr8UnpgI/AAAAAAAAABE/abdbPqg66HE/s320/GSM_arquitectura.jpg. Consulta: 6 de noviembre de 2013.

La arquitectura GSM está compuesta por los siguientes subsistemas:

1.2.1. Estación móvil

La estación móvil o MS es un dispositivo que transmite y recibe señales de radio dentro de una celda, una estación móvil es normalmente un teléfono celular. Las características de los aparatos móviles incluyen comunicación de voz, características de mensajes, y manejo de directorio telefónico. La estación móvil está compuesta del equipo móvil y el módulo de identidad del subscriptor conocido como SIM (*Subscriber Identity Module*) que permite identificar de manera única al usuario o abonado y a la estación móvil. La SIM es usualmente provista por el operador de la red al suscriptor en la forma de una tarjeta inteligente. Un mensaje corto de texto es típicamente almacenado en la estación móvil.

La mayoría de aparatos tienen capacidad de almacenamiento en la SIM. Algunos modelos de aparatos complementan la capacidad de almacenamiento de la SIM con almacenamiento adicional en el propio aparato.

1.2.2. Subsistema de Estación Base BSS

Este es un subsistema que agrupa infraestructura necesaria para comunicarse con un sistema central de control y para que puedan encargarse de la gestión del interfaz de radio. Está en contacto directo con las estaciones móviles a través de la interfaz de radio, también conocido por interfaz de aire, que se encuentra dentro de su área de cobertura a través de celdas radioeléctricas. Por lo tanto, incluye equipo encargado de la transmisión y recepción de radio, y de su gestión. También está en contacto con los conmutadores del subsistema de red.

La misión de este subsistema se puede resumir en conectar la Estación Móvil y el Subsistema de Red, y por lo tanto, conecta al usuario del móvil con otros usuarios. El conjunto de la antena con su electrónica y su enlace con el resto de la red, esta se compone de los siguientes elementos:

- Estación Transceptora Base o BTS (*Base Tranceiver Station*)
- Controlador de Estación Base o BSC (*Base Station Controller*)
- Unidad de Transcodificación o TRAU (*Transcoding Rate and Adaptation Unit*)

1.2.2.1. Estación Transceptora Base

En esta se implementa la comunicación por interface de radio con las Estaciones Móviles a través de celdas radioeléctricas asociadas a una Estación Base, incluye los dispositivos de transmisión y recepción por radio e incluyendo las antenas, también todo el procesado de señales específico a la interfaz de radio como la modulación, demodulación, ecualización de señal y manejo de errores. Se pueden considerar como complejos módems de radio, con funciones extras. Sus principales funciones son:

- Monitoreo de canales libres y envío de la información de los mismos hacia el Controlador de Estación Base.
- Temporización de bloqueos de radios, y edición de mensajes de aviso.
- Direcciona los accesos por parte de las Estaciones Móviles.
- Codificación y entrelazado para protección de errores.
- Medición de la intensidad de campo y calidad de señal recibida de los móviles.
- Encriptación de la información de señalización de tráfico.
- Envío de alarmas en caso de daños o mal funcionamiento del equipo.

1.2.2.2. Controlador de Estación Base

Controla las Estaciones Transceptoras, mapeo de los canales de radio terrestre, vigila que la comunicación no deba interrumpirse porque un usuario se desplace de una celda a otra (*roaming*) y salga de la zona de cobertura de una Estación Base, deliberadamente limitada para que funcione bien el sistema de celdas. Estos controladores están interconectados entre sí, teniendo como enlace a la Central de Conmutación de Móviles, la cual se encarga de iniciar, conmutar, dirigir y finalizar llamadas. Este es un componente principal de la Estación Base, que realiza las siguientes funciones:

- Establece comunicación entre la Estación Móvil y el Subsistema de Conmutación de Red (NSS).
- Gestión y configuración de canales de radio.
- Gestión de las secuencias del salto de frecuencia, las cuales son enviadas por el Controlador de Estación Base hacia la Estación Transceptora Base.
- Control de potencia tanto de la Estación Transceptora como de la Estación Móvil.
- Proporciona el soporte para la señalización de la interfaz de radio.

1.2.2.3. Unidad de Transcodificación

Este es un componente importante de la Estación Transceptora Base, es la encargada de llevar a cabo la compresión y descompresión de información para voz y datos. La Unidad de Transcodificación no es utilizada en la transmisión de datos.

1.2.3. Subsistema de red y conmutación NSS

Este sistema es el más complejo y es el responsable de las principales funciones de conmutación, gestión de la movilidad, interconexión entre redes y control del sistema, así como las bases de datos necesarios para la gestión de movilidad y los datos de los usuarios. Sus elementos son que se desarrollan a continuación.

1.2.3.1. Centro de Conmutación de Servicios Móvil

El Centro de Conmutación de Servicios Móvil o MSC (*Mobile Service Switching Center*) se encarga de iniciar, terminar y canalizar las llamadas a través del Controlador de Estación Base y la Estación Base correspondientes al abonado llamado. Es similar a una central telefónica de red fija, aunque como los usuarios pueden moverse dentro de la red realiza más actualizaciones en su base de datos interna.

1.2.3.2. Registro de ubicación de usuario

Existen dos tipos de bases de datos para el registro de ubicación, la primera de ellas es el registro de ubicación base o HLR (*Home Location Register*), que almacena la posición del usuario dentro de la red, si está conectado o no y las características de su usuario, es típico que sea capaz de almacenar información de cientos de miles de usuarios. Por razones de seguridad es una base de datos con redundancia geográfica, de manera que si un HLR deja de funcionar, se minimice o sea inapreciable los problemas para el usuario. También es independiente de la posición actual del usuario.

La segunda, corresponde al registro de ubicación de visitante o VLR (*Visitor Location Register*), que es una base de datos dinámica y volátil que almacena, información para el área cubierta por un Centro de Conmutación de Servicios Móvil, número de Identidad Internacional del Subscriptor Móvil o IMSI (*International Mobile Subscriber Identity*), número de teléfono (MSISDN), permisos, tipos de abono y localizaciones en la red de todos los usuarios activos en ese momento y en ese tramo de la red. El uso de estas dos bases de datos y el Centro de Conmutación de Servicios Móvil, permiten el enrutamiento de llamadas y el *roaming*.

1.2.3.3. Centro de Autentificación

También llamada AUC (*Authentication Center*), se encarga de la autentificación y cifrado, de los usuarios y las llamadas. Suele estar ubicado en el mismo nodo que el Registro de Ubicación Base. Tiene la función de encriptación sobre la interfaz de radio, para la verificación de la autenticidad de un usuario se utiliza la palabra secreta y el algoritmo de encriptación A3.

1.2.3.4. Registro de Identificación de Equipos

El Registro de Identificación de Equipos o EIR (*Equipment Identity Register*) es una base de datos que contiene información de las estaciones móviles que tienen autorización de operación, robados, y puede ser usado para el bloqueo del servicio. Está clasificada en 3 tipos de listas, las cuales son:

- Blanca: contiene los que tienen autorización de operación.
- Gris: contiene los equipos que es necesario localizar debido a alguna razón técnica.

- Negra: almacena la identificación de los equipos robados o utilizados de forma ilegal y también la de aquellos equipos que no pueden acceder al sistema porque podrían producir graves problemas técnicos.

1.2.3.5. Pasarela de Centro de Conmutación de Servicio Móvil

También llamada GMSC (*Gateway Mobile Service Switching Center*), es un conmutador que realiza la conexión con otras redes, establece llamadas y realizar otros servicios implementados por la red GSM. Este componente es el responsable de que las conexiones originadas o dirigidas hacia otras redes GSM, sean direccionadas al Centro de Conmutación de Servicios Móvil correcto. También generará la información necesaria para poder facturar al usuario.

1.2.3.6. Sistema de señalización SS7

El sistema de señalización número 7 es un conjunto de protocolos de señalización telefónica, su propósito es el establecimiento y finalización de llamadas, traducción de números, mecanismos de tarificación prepago y sobre este sistema se realiza el envío de mensajes cortos SMS. Está clasificado como un sistema de señalización por canal común o CCIS (*Common Channel Interoffice Signalling Systems*), debido a que separan la señal de señalización de los canales portadores. SS7 es un medio por el cual los elementos de una red de telefonía intercambian información en forma de mensajes.

Aunque es común que se le llame red SS7, en realidad es un conjunto de protocolos que funciona sobre una red física dentro del subsistema de conmutación y red NSS.

Cada nodo es identificado en la red por un número llamado un código de punto y cada punto de señalización utiliza una tabla de ruteo para seleccionar la ruta más apropiada para que un mensaje alcance su destino. SS7 divide en dos segmentos de señalización y circuitos de voz. La red está hecha de muchos tipos de enlace y tres nodos de señalización:

- Punto de Conmutación de Servicios o SSP (*Service Switching Point*)
- Punto de Transferencia de Señal o STP (*Signal Transfer Point*)
- Punto de Control de Servicio o SCP (*Service Control Point*)

1.2.4. Subsistema de Operación y Soporte

El subsistema de Operación y Soporte o OSS (*Operación and Support Subsystem*) se compone de dos entidades que no están totalmente especificadas dentro del estándar GSM. Está compuesta por:

- Centro de Operación y Mantenimiento o OMC (*Operation and Maintenance Center*)
- Centro de Gestión de Red o NMC (*Network Management Center*)

Realiza el manejo de alarmas, gestión de fallos, gestión del rendimiento, gestión de la configuración, la adquisición de tráfico de datos, activar y desactivar funciones y la planificación a largo plazo. Normalmente, centralizada en una red.

Las acciones de operación y mantenimiento se llevan a cabo con el fin de conseguir el buen funcionamiento del sistema GSM en su conjunto, ya sea solucionando los problemas y fallos que aparezcan o monitorizando y mejorando la configuración de los equipos para un mayor rendimiento.

Para realizar estas acciones se requieren interacciones entre algunas o todas máquinas de la infraestructura que se encuentra en el subsistema de Estación Base o en el subsistema de Conmutación de Red y los miembros de los equipos de servicio de las distintas compañías comerciales. La implementación de estas funciones es específica de cada operador.

1.3. Red celular

El área geográfica a la que proporciona cobertura una Estación Base se llama celda o célula del inglés *cell*, motivo por el cual a estos sistemas se les llama a veces celulares. A este modelo de reparto del ancho de banda se le denomina a SDMA o división espacial. Con este concepto, los mismos recursos de radio, caracterizados por una banda de frecuencias y espacios de tiempo, pueden ser utilizados simultáneamente por varios usuarios sin interferencias aunque se encuentren separados por distancias mínimas. La distancia mínima entre dos usuarios depende en el camino de propagación de las ondas de radio en el ambiente donde los dos suscriptores estén localizados.

El sistema GSM está basado en el concepto de celdas que permite cubrir áreas o zonas circulares que se superponen para cubrir un área geográfica. Las redes celulares se basan en el uso de un transmisor-receptor central en cada celda que es una Estación Base.

En una red celular, cada celda está rodeada por 6 celdas contiguas, por esto las celdas generalmente se dibujan como un hexágono. Para evitar interferencia, las celdas adyacentes no pueden usar la misma frecuencia. En la práctica dos celdas que usan el mismo rango de frecuencia deben estar separadas por una distancia equivalente a dos o tres veces el diámetro de la celda.

El tamaño de las celdas es un parámetro de diseño que se calcula con base en el número de usuarios, por el tráfico que genera y el porcentaje de utilización de la estación base. Cuanto menor sea el radio de una celda mayor será el ancho de banda disponible, soportando una mayor cantidad de usuarios (en zonas muy pobladas), sin embargo mayor es el coste en infraestructura. Por lo tanto, en zonas densas como en las ciudades el radio de una celda es pequeño (100 m - 1 km) mientras que en las zonas menos pobladas el radio es mayor, cubriéndose con una única Estación Base una mayor superficie (hasta 30 km por celda).

1.4. Interfaz de radio

Para la comunicación entre Estación Base y una Estación Móvil se utilizan canales físicos, caracterizados por número de *slot* y una portadora. Dentro cada portadora se multiplexan en el tiempo 8 ranuras, formando una trama de TDMA. A un nivel superior los canales de dividen en:

- Canal de tráfico: llevan la voz o datos
- Canal de control: datos de control también llamado señalización
- Canales de Difusión Celular

Los canales de tráfico pueden ser de 2,4, 4,8 o 9,6 kb/s. Para el servicio SMS se utilizan canales de control. Esta interfaz también es conocida como interfaz de aire.

1.5. Servicio de mensajes cortos SMS

El servicio de mensajes cortos o SMS por sus siglas en inglés *Short Message Service*, es el que permite transferir pequeños mensajes de texto entre estaciones móviles a través de un centro de servicio SMSC.

El SMS fue diseñado como parte del estándar de telefonía móvil GSM, aunque en la actualidad es soportado por una amplia variedad de redes como el GPRS y CDMA. Estos mensajes pueden ser enviados desde dispositivos móviles GSM y CDMA, pero también un amplio rango de otros dispositivos como Internet, telex y fax. El SMS es una tecnología soportada por el 100 por ciento de aparatos GSM y por la mayoría de redes GSM alrededor del mundo, así como en la mayoría de redes CDMA. El primer mensaje corto de texto se cree que fue transferido en 1992 sobre canales de señalización de una red GSM Europea. Desde entonces con ésta éxitosa prueba, el uso de SMS ha sido un tema de gran crecimiento.

1.5.1. Características

Según las especificaciones del estándar GSM, un mensaje corto tiene una longitud de 1 120 bits, por lo que puede tener una longitud máxima de 160 caracteres alfanuméricos con una codificación de 7 bits (1 120/7=160), una longitud máxima de 140 caracteres con una codificación de 8 bits (1 120/8=140) o una longitud máxima de 70 caracteres con una codificación de 16 bits (1 120/16=70). Dependiendo del tipo de codificación varía el conjunto de caracteres a utilizar. El servicio de mensajes cortos se caracteriza por la confirmación de recepción del mensaje de salida, esto significa que el usuario que envía el mensaje, posteriormente recibe un mensaje de confirmación indicando si el mensaje fue enviado o no.

El servicio de mensajes cortos puede funcionar simultáneamente con los servicios de voz y datos, es posible ya que el servicio de mensajes viaja a través de un canal de datos dedicado a la señalización, independiente del tráfico de voz y datos.

1.5.1.1. Mensajes cortos de mayor longitud estándar

Existen varias formas o métodos de enviar mensajes cortos de mayor longitud a 160 caracteres, estos surgen como una solución para sobrepasar esta limitante y poder envía mensajes largos, los métodos son los siguientes:

- Concatenación de SMS es un proceso de encadenamiento de varios mensajes cortos como un grupo, según las especificaciones GSM, es posible concatenar hasta 255 mensajes, sin embargo SMS no fue diseñado para el alto volumen de datos y el estándar no incorpora manejo errores y recuperación de mensajes en caso de pérdida, por lo que el uso de más tres mensajes resulta poco práctico.

- La compresión de SMS, donde se logra más de 160 caracteres dentro de un solo SMS, el algoritmo especificado por las normas es *Raw untrained dynamic Huffman* que permite longitudes de aproximadamente 200 caracteres. Aunque la compresión de mensajes cortos esta especificada dentro del estándar GSM, los fabricantes de teléfonos han sido renuentes a incorporar esta característica.

1.5.1.2. Alfabeto o conjunto de caracteres

Los alfabetos definen e incorporan el conjunto de caracteres para los principales idiomas de todo el mundo, como alfabeto soportado en los mensajes cortos. Las especificaciones GSM define tres tipos de alfabetos soportados para la codificación de mensajes cortos, codificación a 8 bits, USC2 (*Universal Multiple Octet Coded Character Set 2*) que utiliza 16 bits para su codificación y alfabeto GSM por defecto de 7 bit o simplemente codificación a 7 bits. Cada alfabeto tiene asociado un conjunto de caracteres similares al de una tabla Ascii.

1.5.1.3. Seguridad y encriptación

La seguridad el contenido del mensaje corto puede ser de mucha importancia, para algunas aplicaciones basadas en SMS, como lo podría ser el servicio de banca y otros servicios comerciales sensibles. Para asegurar que los mensajes cortos no se corrompen o intercepten, la norma GSM incorpora mecanismos de seguridad, donde toda la información que se transfiere sobre el canal de señalización se divide dentro de segmentos de 23 *bytes* y cada segmento está protegido por un código CRC de 5 *bytes* llamado *fire* o fuego en español, que proporciona una comprobación de redundancia cíclica, donde toda la información dentro y fuera del mensaje corto en sí está incluido dentro del código generado.

Esta comprobación se calcula automáticamente entre la Estación Móvil y la Estación Base y posteriormente entre la Estación Base y el Centro de SMS. Los mensajes cortos son rutinariamente encriptados sobre la señal de radio entre la Estación Móvil y la Estación Base usando el algoritmo de encriptación IA5.

1.5.1.4. Tipo de mensajes según origen

Según su origen existen dos tipos de mensajes, MT-SM y MO-SM, que corresponden a mensajes de recepción y envío respectivamente. Originalmente los mensajes SMS fue creado con el objetivo que el operador de red enviara información sobre los servicios a los usuarios, por lo que los mensajes solo viajaban en un solo sentido, y a estos se les denomino MT-SM (*Mobile Terminated-Short Message*), que se consideraba un servicio de entrega de un mensaje desde centro de servicio SMSC hasta una Estación Móvil.

Posteriormente la empresa Nokia desarrollo un sistema para permitir la comunicación bidireccional para los mensajes cortos, que se denomino MO-SM (*Mobile Originated-Short Message*), que realizaba la función de entregar un mensaje originado desde la Estación Móvil hasta el centro de servicio SMSC.

1.5.1.5. Parámetros SMS

Dentro de la estructura de un SMS, se encuentra el cuerpo del mensaje corto también llamado carga útil o *Payload*, y los parámetros mínimos para el correcto procesamiento de cada mensaje corto son los siguientes:

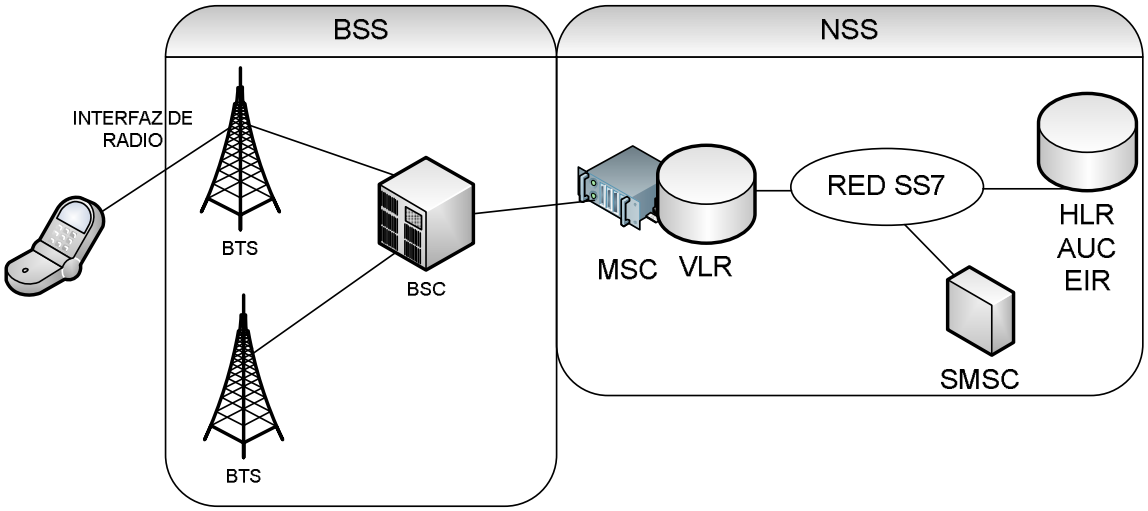
- Fecha de envío o *timestamp*.
- Tiempo de validez, que puede ser desde una hora hasta una semana, después de ese tiempo, si no ha sido entregado, el mensaje es descartado.
- Número de teléfono del remitente.
- Número de teléfono de destinatario.
- Número de SMSC que ha originado el mensaje.

1.5.2. Estructura de red para el servicio SMS

La estructura básica de la red para el servicio SMS (véase figura 2), están involucrados cuatro componentes principales que operan entre sí, y forman parte de una red GSM, estos componentes son:

- Estación Móvil, también puede ser un algún otro dispositivo o entidad que pueda enviar o recibir mensajes cortos SME (*Short Messaging Entity*)
- Estación Base BSS
- Subsistema de red y conmutación NSS
    - Centro de Conmutación de Servicios Móvil MSC
    - Bases de datos HLR y VLR
    - Sistema de señalización SS7
- SMSC

Figura 2. **Estructura de red para SMS**



Fuente: http://movilfacil.files.wordpress.com/2011/03/sms.jpg.
Consulta: 6 de noviembre de 2013.

1.5.2.1. SMSC

La central de servicio de mensajes cortos SMSC (*Short Message Service Center*) es una combinación de hardware y software que reside en los operadores de red GSM y tiene como función principal el enviar y recibir mensajes SMS, el cual centraliza y almacena los mensajes de texto hasta que son enviados con éxito a sus destinos, también es el encargado de realizar la pertinente conexión con el resto de componentes de la red GSM. Las funciones de una SMSC se pueden resumen en:

- Recibir y almacenar los mensajes cortos enviados por los usuarios (MO-SM) hasta que puedan ser enviados.
- Verifica en la base de datos HLR, los permisos del origen, para enviar mensajes.
- Verifica si el usuario destino está activo o no, mediante una consulta a la base de datos VLR, si esta activo el mensaje se envía de lo contrario se almacena temporalmente.
- Verifica periódicamente el estado de los usuarios destinos por los mensajes pendientes de entrega.

1.5.3. Proceso de envío y recepción de mensajes cortos

El medio de comunicación responsable de enlazar los componentes utilizados en el proceso de envío y recepción de SMS es la red SS7. En este proceso por cada mensaje corto, se generan dos tipos de rutas, que son necesarias para poder realizar la correcta gestión de los mensajes cortos, por dichas rutas se transportan los mensajes MT-SM y MO-SM, los cuales se explicaron anteriormente, la trayectoria de las rutas son las siguientes:

1.5.3.1. La ruta de un MO-SM

Esta transacción surge cuando un usuario de la red genera un mensaje corto, creando un mensaje de origen de tipo MO-SM, en la trayectoria de esta ruta se producen los siguientes pasos:

- El HLR donde está registrado la Estación Móvil, verifica si puede enviar o no el mensaje, también proporciona información de ruteo a la SMSC.

- El MSC al que está conectado la Estación Móvil, recibe el mensaje del SMSC y envía información sobre la tarificación al VLR y después lo remite al SMSC origen.

- La SMSC origen envía el mensaje al SMSC destino, transformando el mensaje origen en un mensaje de tipo MT-SM y se procesa como tal.

- La SMSC destino informa del estado del mensaje y devuelve un informe de recepción al MSC origen y la Estación Móvil del usuario, este mensaje solo indica que el mensaje fue enviado.

- Por último, si se el usuario ha solicitado acuse de recibido, recibirá posteriormente un mensaje de estado confirmando si el usuario destino ha recibido el mensaje o no.

Este tipo de mensajes tiene el defecto que se tarifican y confirman inicialmente cuando son enviados a la red (cuando llega a la SMSC) y no al destino final. Un mensaje puede no llegar a destino, por problemas de red, caducidad de la validez o cualquier otro motivo, sin embargo será cobrado por el operador.

1.5.3.1. La ruta de un MT-SM

Una vez que el mensaje originado por un usuario o cualquier otra entidad, se encuentra preparado para su envío en la SMSC del destino, se le denomina MT-SM y el proceso de entrega realiza los siguientes pasos:

- La SMSC destino, almacena el mensaje en su base de datos y solicita al VLR del usuario destino, la información de localización.
- Si el usuario no se encuentra disponible, el mensaje se almacena en la SMSC durante el período de vigencia, posteriormente verifica su disponibilidad hasta que este activo o período de vigencia caduca y es eliminado.
- Si el usuario está disponible la SMSC envía el mensaje al MSC, indicando a que BSS debe ser entregado.
- El MSC envía un aviso al VLR, al cual que está conectado el usuario destino, para indicar que se va a procesar un mensaje.
- El VLR avisa a la Estación Móvil del usuario y verifica si está conectado a la red.
- El VLR responde al MSC con el estado de la Estación Móvil y con la localización del mismo.
- El MSC envía el mensaje a la Estación Móvil.
- El MSC informa a la SMSC de que el mensaje se ha entregado y que puede ser eliminado de su base de datos.
- Opcionalmente, la SMSC destino responde a quien origino el mensaje, con un aviso de entrega del mensaje.

Un mensaje de tipo MT-SM se caracteriza por que siempre se entrega, a diferencia de un mensaje MO-SM.

1.5.4. Modelo de capas para SMS

En el modelo de capas para SMS, cada capa o nivel proporciona un servicio a la capa superior y este servicio se implementa mediante el protocolo correspondiente en cada capa. El modelo está divido en cuatro capas que son:

- La capa de aplicación
- La capa de transferencia
- La capa de retransmisión
- La capa de enlace

1.5.4.1. Capa de aplicación

La capa de aplicación está implementada en la Estación Móvil o alguna entidad de mensajes cortos SME con la capacidad de procesar mensajes SMS, y son aplicaciones de software que envían, reciben, e interpretan el contenido de mensajes. La capa de aplicación es también conocida como Capa de Aplicación de Mensajes Cortos o SM-AL de las siglas en inglés de *Short Message Application Layer*.

1.5.4.2. Capa de transferencia

Aquí se realiza el servicio de transferencia de un mensaje corto entre una Estación Móvil y una SMSC (en ambos sentidos) y obtención de los correspondientes informes sobre el resultado de la transmisión. En la capa de transferencia el mensaje es considerado como una secuencia de octetos conteniendo información tal como la longitud del mensaje, si el mensaje es originado o recibido, fecha de recepción.

Esta capa presta servicios en los cuales se encarga de los detalles internos de la red y permitiendo que la capa de aplicación pueda intercambiar mensajes de forma transparente. La capa de transferencia es también conocida como Capa de Transferencia de Mensajes Cortos o SM-TL de las siglas en inglés de *Short Message Tranfer Layer*.

1.5.4.3. Capa de retransmisión

Esta capa permite el transporte de un mensaje entre varios elementos de red, un elemento de red puede temporalmente almacenar un mensaje si el próximo elemento al cual el mensaje ha sido enviado no está disponible. En la capa de retransmisión el MSC de la red GSM, maneja dos funciones adicionales a las funciones usuales de capacidades de conmutación. La primera función llamada SMS *gateway* o puerta de enlace MSC, identificada como SMS-GMSC, que es capaz de recibir un mensaje de un SMSC e interrogar al HLR para obtener la información de ruta y en seguida repartir el mensaje a la red receptora.

La segunda función llamada SMS de Interfuncionamiento denotada como SMS-IWMSC que es un MSC capaz de recibir un mensaje de una red móvil y enviarlo al SMSC adecuado. La capa de retransmisión es también conocida como Capa de retransmisión de Mensajes Cortos o SM-RL de las siglas en inglés de *Short Message Relay Layer*. Los componentes SMS-GMSC y SMS-IWMSC suelen estar integrados con el SMSC.

1.5.4.3. Capa de enlace

La capa de enlace permite la transmisión del mensaje en el nivel físico. Para este propósito, el mensaje es protegido para sobrellevar con bajo nivel los errores de canal. La capa de enlace es también conocida como Capa de Enlace de Mensajes Cortos o SM-LL de las siglas en inglés de *Short Message Link Layer*.

1.5.5. Segmentación de un mensaje

Para propósitos de transporte y debido a las limitaciones en la capa de transferencia, una aplicación puede necesitar segmentar el mensaje en varias piezas llamadas mensajes segmentados, esta técnica es utilizada para la concatenación de mensajes (tema abarcado anteriormente). Un mensaje segmentado es también conocido como un mensaje corto y un mensaje segmentado es un elemento manipulado por una aplicación.

Un mensaje segmentado tiene un límite de tamaño o carga útil. En función de conducir una cantidad grande de datos, varios mensajes segmentados pueden ser combinados en un mensaje concatenado. La concatenación del mensaje es manejado por la capa de aplicación. En función de ser transportado, el mensaje segmentado necesita ser mapeado sobre un TPDU (*Transfer Protocol Data Unit*) en la capa de transferencia.

1.5.6. Transacciones entre SME y SMSC

En la capa de transferencia, puede haber seis tipos de transacciones entre un SME y un SMSC. Se clasifican según su sentido de transferencia, teniendo un formado diferente para cada uno de ellos, los tipos de transacciones son:

- *SMS-SUBMIT*: esta transacción realiza el envío de un mensaje del SME al SMSC. Esta transacción se realiza en el tipo de mensaje MO-SM.

- *SMS-SUBMIT-REPORT*: SMSC usa esta transacción en caso de error.

- *SMS-DELIVER*: esta transacción realiza el envío de un mensaje del SMSC al SME. Esta transacción se realiza en el tipo de mensaje MT-SM.

- *SMSDELIVER-REPORT*: el SME responde en la entrega con esta transacción.

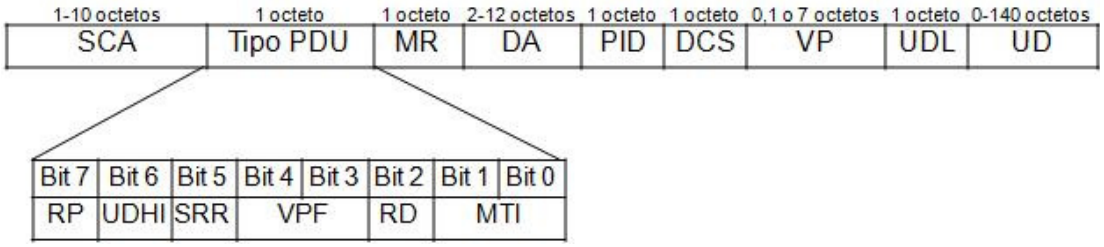
- *SMS-STATUS-REPORT*: esta transacción realiza la transferencia de un reporte de estatus de un SMSC de regreso a un SME.

- *SMS-COMMAND*: esta transacción corresponde a la solicitud de un SME, usualmente un SME externo, para la ejecución de un comando específico por el SMSC.

1.6. Estructura de SMS

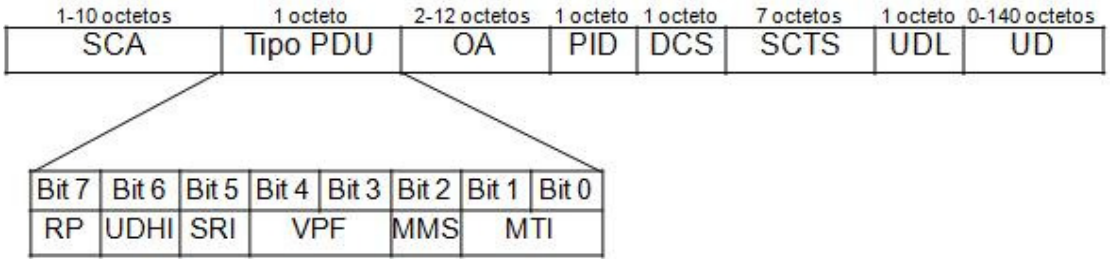
La estructura de un SMS es una trama de datos en formato PDU, siglas de *Protocol Data Unit* que significa Unidad de Datos de Protocolo, que es una cadena de datos binarios representados en hexadecimal, para su manipulación se divide en dos tipos de estructuras utilizadas para el envío y recepción de mensajes SMS, estas estructuras son muy parecidas y comparten campos en común. En las siguientes figuras se pude ver las diferencias en cada estructura.

Figura 3. **Estructura PDU de SMS envío**



Fuente: elaboración propia.

Figura 4. **Estructura PDU de SMS recepción**



Fuente: elaboración propia.

Como ya se vio, existen dos tipos de estructuras PDU, el de mensaje de envío, también conocida como *SMS-SUBMIT*, y el de recepción, también conocido como *SMS-DELIVER*, y se componen por una serie campos representados por uno o varios octetos de información en formato hexadecimal, estas estructuras se componen de los siguientes campos:

1.6.1. SCA

Dirección del Centro de Servicio SMSC o SCA de las siglas de la frase en inglés *Service Center Address* se compone de 3 campos, los cuales definen el número de teléfono que identifica al SMSC, se compone en:

- Longitud: es el número de octetos (1 octeto contiene 2 dígitos) del número de teléfono del SMSC, más un octeto para el tipo de dirección.

- Tipo de dirección: indica si se trata de un número nacional o internacional, se utiliza el número hexadecimal 81 y 91 respectivamente.

- Digitos BCD: número de teléfono del SMSC en notación BCD, que consiste en invertir el orden por pares, por ejemplo: 123456789 se utilizaría 21436587F9, donde la F se antepone al último digito solo si la longitud del número es impar.

1.6.2. Tipo PDU

Este campo contiene información sobre el tipo de PDU a procesar, su longitud es de un octeto o byte de información donde cada bit tiene su propia interpretación. La estructura de este octeto es la siguiente:

Tabla I. **Estructura de campo tipo PDU**

	Bit 7	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**
Envío	RP	UDHI	SRR	VPF		RD	MTI	
Recepción	RP	UDHI	SRI	VPF		MMS	MTI	

Fuente: elaboración propia.

- RP o *Reply Path* indica si existe respuesta, en tramas de tipo SMS-SUBMIT este campo siempre es 0.

- UDHI o *User Data Header Indicator* indica si el campo UD contiene solo el mensaje (0) o si existe un encabezado antes del mensaje (1).

- SRR o *Status Report Request* informe de estado solicitado = 1, no solicitado = 0.

- SRI o *Status Report Indication* informe de estado si fue solicitado = 1, no solicitado = 0.

- VPF o *Valid Period Format* indica el formato del período de validez. 00 No presente; 01 Reservado; 10 Formato relativo: 11 Formato absoluto.

- RD o *Reject Duplicated* indica si se debe rechazar o no los mensajes duplicados.

- MMS o *More Message to Send* indica si existe que hay mas mensajes esperando para ser entregados.

- MTI o *Message Type Indicator* indica el tipo de mensaje, siendo posible las siguientes combinaciones:
  - 00 = *SMS-DELIVER* / *SMS-DELIVER-REPORT*
  - 01 = *SMS-SUBMIT* / *SMS-SUBMIT-REPORT*
  - 10 = *SMS-STATUS-REPORT* / *SMS-COMMAND*
  - 11 = Reservado

1.6.3. MR

MR o *Message Reference* es un número entero que varía entre 0-255 (0-FFH en hexadecimal), y hace referencia al mensaje *SMS-SUBMIT*. Este número se incrementa en 1 por cada mensaje enviado y es generado extrayendo el último número incrementado en 1.

1.6.4. DA y OA

DA o *Destination Address* es el número de teléfono del recipiente o quien recibe el mensaje SMS. OA u *Originator Address* es el número de teléfono que envía o inicializa el mensaje SMS. La estructura de los campos DA y OA son similares al del campo SCA, la diferencia radica en la interpretación de la longitud, donde para SCA el valor indica el número de octetos utilizados para representar al número de teléfono más un octeto extra, para DA Y OA la longitud indica el número de dígitos utilizados para representar el número de teléfono, sin incluir el campo tipo de dirección.

1.6.5. PID

De las siglas en inglés *Protocol ID* que significa Identificación de Protocolo, con el cual se indica la naturaleza de los datos a transportar, este dato es utilizado por el SMSC para un mejor ruteo del mensaje.

1.6.6. DCS

De las siglas en inglés *Data Coding Scheme* o Esquema de Codificación de Datos, el cual indica el alfabeto con el que se codifica la trama de datos (ver sección 1.5.1.2) y la clase de mensaje SMS.

Tiene una longitud de 8 bits de los cuales los primeros 2 bits están reservados (no se utilizan) y tiene la siguiente configuración:

Tabla II. **Estructura de campo DCS**

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Reservado	Reservado	Utiliza Compresión	Especifica tipo Mensaje o no	Tipo de alfabeto		Clase de mensaje	

Fuente: elaboración propia.

1.6.6.1. Tipo de alfabeto

Este campo tiene una longitud de 2 bits, existiendo 3 posible combinaciones Codificación USC2 de 16 bits (10), Codificación de 8 bits (01) y Codificación de 7 bits (00).

1.6.6.2. Clases de mensajes

Adicionalmente a los tipos de mensajes, un mensaje también pertenece a una clase. El parámetro DCS (*Data Coding Scheme*) del TPDU indica la clase a la cual pertenece el mensaje. Cuatro clases han sido definidas e indican como un mensaje deberá ser manejado por el SME receptor, tiene una longitud de 2 bits. A continuación se proporciona una breve descripción de cada clase:

- 00 Clase 0 o flash SMS: son desplegados directamente en pantalla.
- 01 Clase 1: mensajes específicos almacenados en el equipo móvil.
- 10 Clase 2 o estándar: mensajes específicos almacenados en la SIM.
- 11 Clase 3: mensajes específicos para equipo Terminal externo al móvil.

1.6.6.3. Ejemplo de segmento de trama DSC

En este ejemplo se indica que el texto no utiliza compresión, se especifica la clase de mensaje, se utiliza codificación de 7 bits, y la clase de mensaje es de tipo 2, la DSC quedaría de la siguiente forma:

Tabla III. **Ejemplo de campo DSC**

Bit 7	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**		
Reservado		Compresión	Indica Clase	Alfabeto		Clase		=	12
0	0	0	1	0	0	1	0		

Fuente: elaboración propia.

1.6.7. SCTS

El SCTC o *Service Center Time Stamp* contiene la fecha y hora con la cual el SMSC recibió el SMS y se representa por pares de caracteres con el orden invertido, la fecha 01/02/2014 20:17:52 GMT-6 quedaría de la siguiente forma:

Tabla IV. **Ejemplo de campo SCTS**

	Año	**Mes**	**Día**	**Hora**	**Minutos**	**Segundos**	**Zona**
Datos	14	02	01	20	17	52	98
Semi-octeto	41	20	10	02	71	25	89

Fuente: elaboración propia.

El campo zona identifica la zona horaria, y representa la diferencia, en cuartos de hora, entre la hora local y el tiempo medio de Greenwich GMT, se calcula convirtiendo el número GMT multiplicado por 4 cuartos que tiene una hora, 6*4=24, su equivalente binario obteniendo es 00011000, para indicar que la zona horaria es negativa, el séptimo bit se coloca a 1, quedando así:

Tabla V. **Ejemplo de zona horaria**

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	=	98
1	0	0	1	1	0	0	0		

Fuente: elaboración propia.

1.6.8. VP

VP o *Validity Period* este parámetro indica el período o tiempo de validez del mensaje SMS, en el envío de SMS al vencer este tiempo, el mensaje caduca y es descartado. El formato de este campo depende del parámetro VPF (ver sección 1.6.2) y puede ser relativo o absoluto, el formato relativo se calcula de la siguiente forma:

Tabla VI. **Rango de valores de campo VP**

Valor	Tiempo	Rango
0-143	(VP +1) x 5 minutos	0 – 12 hrs
144-167	12 horas + ((VP-143) x 30 min.)	12.5 – 24 hrs
168-196	(VP-166) x 1 día	2 – 30 días
197-255	(VP-192) x 1 semana	5 – 63 semanas

Fuente: elaboración propia.

En el caso del formato absoluto, el campo indica la fecha y hora, y para su representación es la misma utilizada por SCTC. Es importante resaltar, que si la fecha ha vencido en al momento de enviar el mensaje, este marcara un error y el mensaje no se enviará.

1.6.9. UDL

UDL de las siglas del inglés *User Data Lenth* indica la longitud de caracteres de los datos de usuario o sea la longitud de campo UD, al igual que los otros campos su valor se representa en hexadecimal.

1.6.10. UD

UD o *User Data* son los datos de usuario, el texto que contiene el mensaje, su codificación depende del parámetro DSC. En el caso de la codificación de 8 bits los caracteres se toman directamente de su equivalente hexadecimal, a diferencia de la codificación de 7 bits donde se necesita realizar una reagrupación de septetos de caracteres (7 bits) y luego volverlos agrupa en octetos.

Para su transformación cada carácter se debe representar en su forma binaria, luego se toman los bits menos significativos del siguiente carácter y se agregan como los bits más significativos del primer carácter formando así un nuevo octeto, este proceso se repite hasta terminar de recorrer todos los caracteres del mensaje. En la siguiente tabla se muestra como se codifica el texto que conforma los datos de usuario:

Tabla VII. **Ejemplo de codificación texto a 7 bits**

Carácter	H	o	l	a
Decimal	72	111	108	97
Binario	1001000	1101111	1101100	1100001
Bits a desplazar	1001000	110111**1**	11011**00**	1100**001**
Bits desplazados	**1**1001000	**00**110111	**001**11011	1100
Hexadecimal	C8	37	3B	0C

Fuente: elaboración propia.

2. USO DE LOS SERVICIOS SMS

Con la finalidad de conocer los conceptos necesarios para el desarrollo de una aplicación, que sea capaz de manipular un dispositivo de hardware GSM, con las funciones de recibir y enviar mensajes SMS, se debe de comprender los temas abarcados en este capítulo.

En el capítulo anterior se estudio los componentes y funcionamiento de una red GSM, en este capítulo se estudian los elementos y forma de utilizar estos, para la creación de una aplicación capaz de procesar mensajes SMS. Los temas que se desarrollaran son:

- Módem GSM, que es el dispositivo de hardware que funciona como el enlace entre la aplicación que se desarrolla y la red GSM.
- Comandos AT, por medio de este conjunto comandos se manipula el módem GSM.
- Y otros conceptos relacionados al envío y recepción de mensajes SMS.

2.1. Módem GSM

Un módem GSM es un módem inalámbrico que funciona sobre una red inalámbrica GSM. Un módem inalámbrico se comporta como un módem de acceso telefónico, la principal diferencia entre ellos es que un módem de acceso telefónico envía y recibe datos a través de una línea telefónica fija, mientras que un módem inalámbrico que envía y recibe datos a través de las ondas de radio.

Un módem GSM puede ser un dispositivo externo o una tarjeta PC Card / PCMCIA Card. Por lo general, un módem GSM se conecta a un ordenador a través de un cable serie o un cable USB. Al igual que un teléfono móvil GSM, un módem GSM requiere una tarjeta SIM de un operador de telefonía móvil para operar. En la figura 3 se muestra el aspecto físico de un módem GSM.

Figura 5. **Módem GSM**

Fuente: http://www.blauden.com/files/MC52it_0.jpg. Consulta: 9 de enero de 2014.

Los equipos GSM utilizan los comandos AT para controlar módems. Ambos módems GSM y módems dial-up admiten un conjunto común de los comandos AT estándar, es posible utilizar un módem GSM al igual que un módem de acceso telefónico.

Además de los comandos AT estándar, módems GSM dan soporte a un amplio conjunto de comandos AT. Este conjunto de comandos extendida AT se definen en las normas GSM. Con los extendidos comandos AT, puede realizar funciones como:

- Lectura, escritura y eliminación de mensajes SMS
- Envío de mensajes SMS
- Monitoreo de la intensidad de la señal
- El monitoreo del nivel de carga y estado de carga de la batería
- Lectura, escritura y búsqueda en la agenda del dispositivo

El número de mensajes SMS que pueden ser procesados por un módem GSM por minuto es muy bajo, y puede variar dependiendo del dispositivo, sólo alrededor de diez a quince mensajes SMS por minuto.

Haciendo referencia a la arquitectura GSM desarrollado en el capítulo anterior, el módem GSM en conjunto con el ordenador al cual se conecta, mas la aplicación a desarrollar, equivale a lo que se denominaba SME.

Un gran conjunto de teléfonos móviles tiene la capacidad de ser utilizados como módems, aunque para hacer uso de ellos se deban utilizar controladores y software suministrados por el fabricante.

Un teléfono móvil puede no admitir algunos comandos AT, comando de parametrización o valores de parámetros, por ejemplo algunos teléfonos son restringidos para no poder utilizar el comando AT+CGMG=1, encargado de parametrizar el envío de mensajes SMS en modo texto, por lo que solo se puede hacer uso en la modalidad PDU, por lo general un módem GSM soporta un conjunto más complejo de comandos AT que los teléfonos móviles. En términos de velocidad de transmisión de mensajes SMS, no existe una diferencia notable, ya que el factor determinante para la transmisión de SMS es la red GSM a la que está conectada.

Utilizar un teléfono móvil como módem puede tener una serie de deficiencias en comparación a un módem GSM, en la tabla VIII se realiza una comparación entre utilizar un módem GSM y un teléfono móvil.

Tabla VIII. **Comparación entre un módem GSM y un teléfono móvil**

Característica	**Módem GSM**	**Teléfono móvil**
Continuo uso	Opera continuamente, su fuente de energía es constante.	Está limitado a la duración de la batería interna.
Uso de Comandos AT	Conjunto estándar y extendidos	Se encuentra limitado ya que son de uso específico.
Soporte para envío SMS en modo texto	Si	Algunos no soportan esta modalidad.

Fuente: elaboración propia.

2.2. Comandos AT

La compañía Hayes Communications creó el conjunto de instrucciones Hayes, que posteriormente se convirtió en un estándar abierto de comandos, orientados a interactuar con módems, realizando actividades como configurar y parametrizar. Los caracteres AT que preceden en la sintaxis, a todos los comandos Hayes, son la abreviatura de la palabra del idioma inglés *attention*, que significa atención, e hicieron que estas instrucciones se conocieran por comandos AT.

El medio físico de comunicación de los módem con un ordenador es a través de un puerto serial, los comandos AT hacen posible que el software pueda comunicarse con el módem.

2.2.1. Clasificación de los comandos AT

Los comandos AT se pueden clasificar de dos formas, por su naturaleza operacional y por su función. Por su naturaleza se dividen en dos grandes grupos que son:

- Ejecución de acciones inmediatas, tales como comandos ATD de marcación, ATA de contestación y ATH de desconexión.
- Configuración, como ATV que define como responde el módem tras la ejecución de un parámetro, ATE de selección del eco.

Los comandos AT se pueden subdividir en cuatro grupos delimitados como:

- Básicos: fueron los que inicialmente se definieron y cumplen funciones elementales.
- De registro: estos modifican los valores de los registros internos de los módems o solicitan sus valores, como los comandos se selección y configuración de SMSC.
- Extendidos: en un inicio el total de comandos creado por Hayes era básico, posteriormente se requirieron mas funciones para manipular los módem y se fueron agregando nuevos comandos, a estos comandos se le denomino extendidos y tienen la forma AT+X.
- Propietarios: cada fabricante introdujo sus propios comandos, los cuales no fueron parte del conjunto estándar y tienen una funcionalidad específica para sus propios dispositivos.

2.2.2. Sintaxis

Como ya se indicó anteriormente, la mayoría de comandos AT inician con la secuencia AT, a excepción del comando A/ que repite el último comando introducido. La sintaxis general de los comandos AT es sencilla, siendo de la siguiente forma: AT<Comando><CR>.

Donde:

- AT indica Atención
- <Comando>: es el comando a ejecutar
- <CR>: es el carácter que representa Enter o retorno de carro

Existen algunas reglas que se deben aplicar a la sintaxis:

- El comando puede estar escrito en mayúsculas o minúsculas, aunque en las especificaciones de SMS indica que todos los comandos son en mayúsculas.

- La cadena de texto que representa un comando debe contener menos de 40 caracteres.

- Si se está usando una consola, un error de escritura en el comando se puede corregir usando la tecla retroceder.

- Todos los comandos terminan con la tecla Enter, a excepción de los comandos +++ y A/.

- El conjunto de caracteres que se pueden utilizar para representar un número telefónico son 1 2 3 4 5 6 7 8 9 * = , ; # + >, los demás caracteres son ignorados.

- Varios comandos AT se pueden enviar en una sola cadena de comando, y solo el primer comando AT debe llevar el prefijo A.

- Los comandos AT en la misma cadena de comandos deben ser separados por punto y comas.

- Los comandos que utilizan un parámetro numérico, pueden ser utilizados sin valor numérico, en este caso el comando emite el valor cero.

- Si la cadena de comando contiene dos comandos sin parámetros, el módem responderá con un error.

- El intérprete de comandos reconoce AT o at, pero no puede reconocer At o aT, ambos caracteres tiene que estar en minúsculas o mayúsculas.

- Un parámetro tipo texto debe ir encerrado entre comillas dobles.

2.2.3. Información de respuesta y código de resultado

Los códigos de resultado son los mensajes enviados desde el módem para ofrecer información sobre la ejecución de un comando y la ocurrencia de un evento. Existen dos tipos de códigos de resultado que se utilizan:

- Códigos de resultado final
- Información de respuesta

2.2.3.1. Códigos de resultado final

El código de resultado final marca el fin de una respuesta del comando ejecutado. Indica que el módem ha completa la ejecución y dos códigos son de uso frecuente: OK, en caso de éxito y ERROR, si ocurrió algún problema.

2.2.3.2. Información de respuesta

Es la información de respuesta que se ha solicitado, y el contenido de la misma varía dependiendo del comando ejecutado. Por ejemplo, al ejecutar el comando AT+CGMI, que solicita el nombre del fabricante del módem, el módem puede devolver Nokia como información de respuesta.

2.2.4. Resumen de comandos AT

El conjunto de comandos AT es muy amplio y se pueden agrupar por categorías, a continuación se presenta una lista de los comandos de más uso para la manipulación de los módems GSM:

- Básicos
  - AT: comando base, comprueba la disponibilidad y funcionamiento del módem.
  - ATA: contesta una llamada, también se configura en respuesta automática.
  - ATB: elige el estándar de comunicación, cuando se inicia una comunicación.
  - ATD o ATE: activación/desactivación del eco del módem 1 activa, 0 desactiva. Ej. Para desactivar el eco se usa ATE0.
  - ATH: permite finalizar la llamada actual.

- Uso general
  - AT+CGMI: identificación del fabricante.
  - AT+CGSN: obtener número de serie.
  - AT+CGMM: petición de identificación del modelo.
  - AT+CIMI: obtener el IMSI.
  - AT+CPAS: leer estado del módem.
  - AT+CBC: da a conocer el estado de la batería.

- Seguridad
  - AT+CPIN: introducir el PIN.
  - AT+CPINC: obtener el número de reintentos que quedan.
  - AT+CPWD: cambiar clave.

- Servicio de red
  - AT+CSQ: obtener calidad de la señal.
  - AT+COPS: selección de un operador.
  - AT+CREG: registrarse en una red.
  - AT+WOPN: leer nombre del operador.

- Manipulación de agenda telefónica
  - AT+CPBR: leer todas las entradas.
  - AT+CPBF: encontrar una entrada.
  - AT+CPBW: almacenar una entrada.
  - AT+CPBS: buscar una entrada.

- Para manipulación de SMS
  - AT+CPMS: seleccionar lugar de almacenamiento de los SMS.
  - AT+CMGF: seleccionar formato de los mensajes SMS texto o PDU.

- AT+CMGR: leer un mensaje SMS almacenado.
- AT+CSMP: establece los parámetros de modo texto.
- AT+CSDH: muestra los parámetros de modo texto.
- AT+CSAS: graba la configuración.
- AT+CMGL: listar los mensajes almacenados.
- AT+CMGS: enviar mensaje SMS.
- AT+CMGW: almacenar mensaje en memoria.
- AT+CMSS: enviar mensaje almacenado.
- AT+CSCA: establecer el centro de mensajes SMSC a usar.
- AT+WMSC: modificar el estado de un mensaje.
- AT+CSMS: da lista de servicios soportados, también indica el servicio a usar.
- AT+CNMA: acuse nuevo mensaje para ME/TA.
- AT+CMGC: enviar comando.
- AT+CMT: se utiliza para desviar los mensajes SMS recibidos para el ordenador.
- AT+CBM: se utiliza para desviar mensajes de difusión del celular al ordenador.
- AT+CDS: utilizado para los informes de estado de avance recibidos al ordenador.
- AT+CSMP: coloca los parámetros del modo de texto.
- AT+CDSS: mostrar parámetros del modo de texto.

2.3. Comunicación PC a módem

Como ya se mencionó anteriormente, el ordenador o PC se comunica con el módem GSM a través de un puerto serial (rs232 o USB), para hacer uso del módem hay que tener lo siguiente:

- Módem GSM compatible con comandos AT
- Cable que conecta el PC con el módem
- Controladores del módem para el sistema operativo a utilizar

Sobre el cable, generalmente viene junto con el módem GSM, y el conector que da hacia el ordenador son del tipo RS232 o USB. Si el conector es el tipo RS232 es probable que el ordenador no tenga este tipo de conectores, ya que en la actualidad el uso de estos conectores ha disminuido y ha sido sustituido por los puertos USB, si ese es el caso, existe la posibilidad de adquirir un convertidor de RS232 a USB.

Figura 6. **Convertidor RS232 a USB**

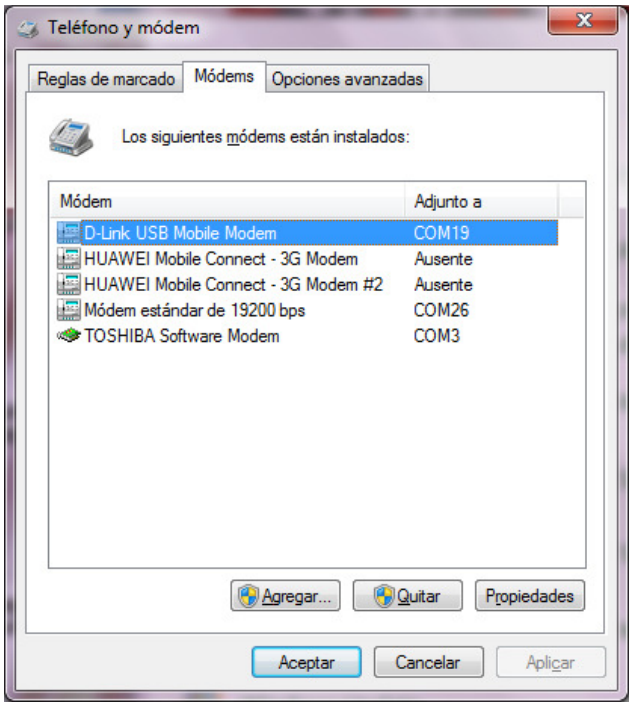
Fuente: http://www.seektron.eu/images/MT609-2 Convertor USB la RS232.jpg. Consulta: 9 de enero de 2014.

El controlador o *driver* del módem, es un pequeño programa que permite al Sistema Operativo controlar las funciones del módem, esto requiere que el controlador del módem de soporte a la versión del Sistema Operativo en uso.

La instalación del controlador por lo general es simple, consiste en ejecutar el instalador, seguir las instrucciones que indique (con el módem conectado), y al finalizar el instalador, el controlador ya estaría preparado para su utilización. El Sistema Operativo le asigna un identificador al dispositivo, conocido como número de puerto COM, por medio de este identificador es posible manipular el módem desde una aplicación.

Para conocer el número de puerto que el Sistema Operativo ha asignado al módem GSM, se debe de ir a Opciones de teléfono y módem del Control Panel del Sistema Operativo, en la siguiente imagen se puede visualizar:

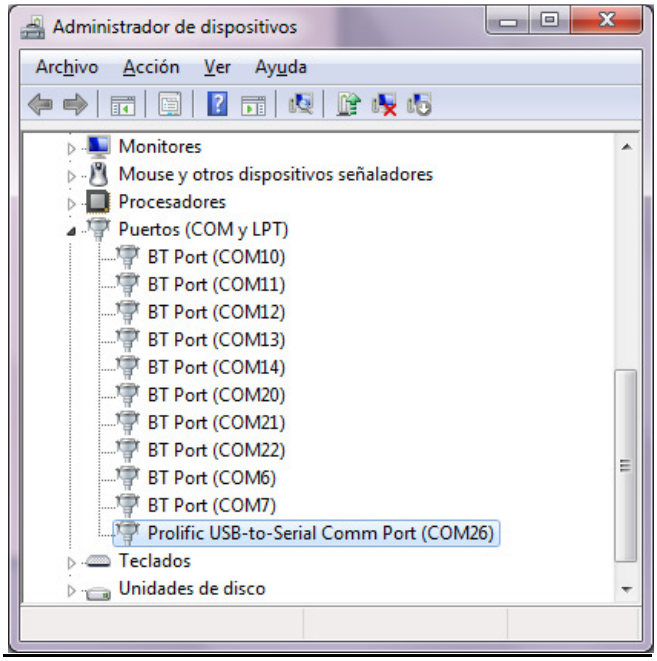
Figura 7. **Configuración de teléfono y módem del Sistema Operativo**



Fuente: elaboración propia.

En la imagen anterior se puede ver que hay varios módem instalados, dos de ellos son GSM que son los que identifican con COM19 y COM26. Existe la posibilidad que el módem GSM no se identificado como un dispositivo tipo módem, si esto sucede no aparecerá, ya que el controlador del módem solo crearía un acceso por medio de un puerto COM, si ese el caso, para ver el puerto asignado, solo se debe de ir al Administrador de dispositivos, en el nodo con el título Puertos (COM y LPT), tal como se ve en la siguiente imagen:

Figura 8. **Administrador de dispositivos del Sistema Operativo**



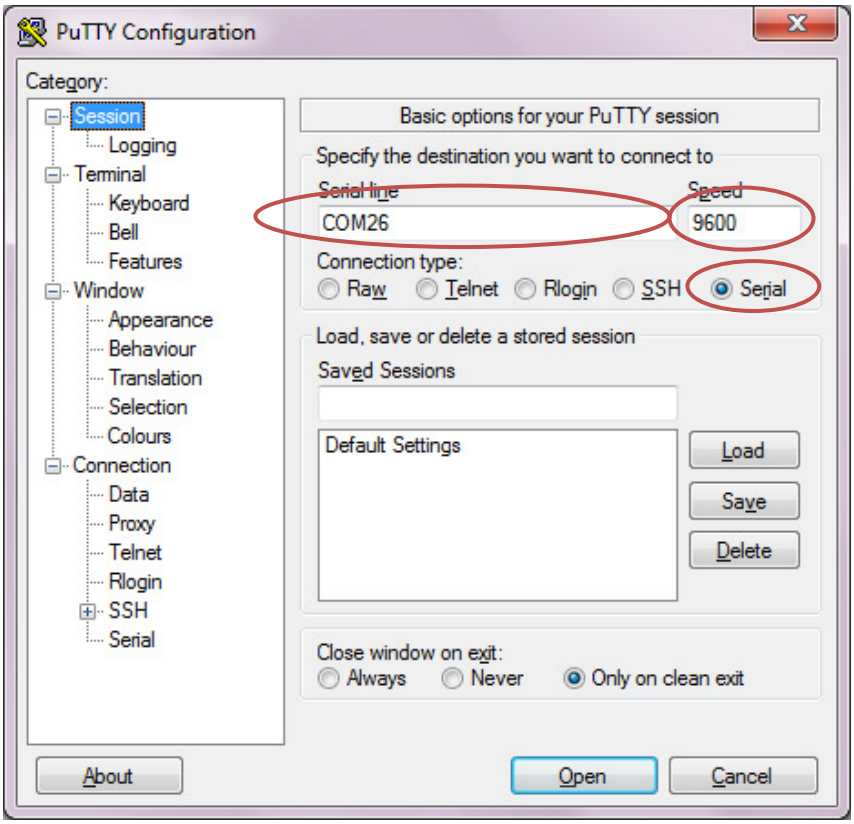
Fuente: elaboración propia.

En este caso se puede visualizar que ha sido clasificado no como un módem sino como un puerto de comunicación tipo serial y se ha asignado el puerto COM26, y el nombre con el que se identifica es Prolific USB-to-Serial Comm Port (COM26).

2.3.1. Prueba de comunicación entre PC y el módem GSM

Al tener instalado y conectado el módem al PC el siguiente paso es comprobar el correcto funcionamiento, tanto del módem, el controlador y cable. Para realizar las pruebas se necesita de una aplicación tipo terminal, para establecer una sesión de conexión con el módem GSM, esta aplicación puede ser HyperTerminal de Microsoft u otros como la famosa terminal Putty, para este ejemplo se realizara con Putty. Se procede a ejecutar Putty que se visualiza en la siguiente pantalla:

Figura 9. **Terminal Putty**



Fuente: elaboración propia.

Donde se muestra las opciones que se tienen disponibles para configurar la sesión. Primero se selecciona la opción Serial, en la casilla titulada como Serial Line, se digita el número de COM correspondiente al módem GSM, para este caso es el COM26, por último en la casilla Speed se establece la velocidad de operación del módem, respeto a la comunicación en bits por segundo, la velocidad indicada debe ser de acuerdo a las que el módem soporte de lo contrario no funcionara. Ahora se puede abrir la sesión presionando el botón Open, y se verá lo siguiente:

Figura 10. **Comunicación con el módem GSM desde una terminal Putty**

Fuente: elaboración propia.

Para comprobar el correcto funcionamiento del módem, basta con digitar AT en la sesión de la aplicación Putty y si todo es correcto el módem responderá con el código de resultado final OK, indicando que ha recibido el comando y lo procesado correctamente. Para la prueba se digita en la sesión el siguiente texto:

AT<CR>
OK

En este ejemplo y en otros encontrara textos encerrados entre < y >, esto se utiliza para indica los caracteres no imprimibles, como lo son:

- <CR>: que representa el retorno de carro o tecla Enter.
- <Ctrl+Z>: que representa el envío de las teclas Control y la tecla Z.

2.3.2. Comprobar si el módem es compatible con SMS

Hasta este punto se ha probado el funcionamiento del módem, lo siguiente es comprobar que el módem es compatible con uso de comandos AT, para el envío y recepción de mensajes SMS. Aunque la mayoría de módem GSM y teléfonos móviles soportan estas funciones, existen algunos que no son totalmente compatibles.

Para comprobar si el módem da soporte para la manipulación de mensajes SMS a través de comandos AT, se utiliza el comando AT+CSMS?, este comando da como respuesta una cadena de 1 o 0, la cual indica los tipos de mensajes que el módem da soporte, a excepción del primer valor, para los siguientes tres, 1 indica que lo soporta y 0 que no lo soporta. El formato de la respuesta está estructurado de la siguiente forma:

+CSMS: <Servicio>,<Valor MT>,< Valor MO>,< Valor BM>

Donde:

- <Servicio>: indica el tipo de servicio seleccionado, 0 indica que los comandos AT para SMS son compatibles con el estándar GSM 7.05 Fase 2, 1 indica que los AT para SMS son compatibles con el estándar GSM 7.05 Fase 2+.
- <Valor MT>: indica si da soporto a los mensajes de tipo MT-SM.

- <Valor MO>: indica si da soporte a los mensajes de tipo MO-SM.
- <Valor BM>: indica si da soporte a los mensajes de tipo difusión o *Broadcast*, este tipo de mensajes, permite enviar un SMS a un conjunto de receptores que se encuentren dentro una determinada área.

En la figura 9 se puede ver la ejecución del comando que da como respuesta +CSMS: 1,1,1,1 indicando que soporta comandos AT para SMS compatibles con el estándar GSM 7.05 Fase 2+, y que soporta los mensajes de tipo MT-SM, MO-SM y BM. Si el código de resultado final devuelve ERROR, significa que el módem no soporta el conjunto de comandos AT para SMS.

Figura 11. **Prueba de soporte SMS**

Fuente: elaboración propia.

2.3.3. Modos de funcionamiento

Las especificaciones SMS han definido dos modos en el que un módem GSM puede operar, que son el modo texto y PDU siglas de *Protocol Data Unit* que significa Unidad de Datos de Protocolo. Dependiendo el modo de operación del módem, determina la sintaxis y formato de respuesta de algunos de los comandos AT para SMS. Los comandos afectos al modo de operación son: +CMGS, +CMSS, +CMGR, +CMGL, +CMGW, +CNMA, +CMGC, +CMT, +CBM y +CDS. Los comandos +CSMP y +CSDH solo pueden usarse en modo texto.

2.3.4. Comparación de modo texto y PDU

A continuación se compara el modo texto y PDU desde diversos aspectos, esta comparación ayudara a comprender las diferencias entre estos dos modos de operación y elegir qué modo de operación debe ser utilizado en el desarrollo de la aplicación de mensajería SMS.

2.3.4.1. Formato y sintaxis de comandos AT

Cuando un módem GSM opera en diferentes modalidades, la sintaxis de ciertos comandos AT y el formato de sus respuestas son diferentes. En modo texto el encabezado y cuerpo del mensaje SMS son ingresados en parámetros separados, mientras que en modo PDU se ingresan en un solo parámetro en formato hexadecimal codificado. Como ejemplo, para el envío de mensajes SMS que contenga el texto Hola, al número 59383474, en modo texto se realiza de la siguiente forma:

AT+CMGS=”59383474”<CR>Hola<Ctrl+Z>

Si ese mismo comando se ejecuta en modo PDU, retornaría un error, y esto se debe a que la sintaxis del comando +CMGS es diferente en modo texto y en modo PDU. El cuerpo del mensaje, el encabezado del número telefónico destino y otros encabezados son codificados en una secuencia hexadecimal. Para realizar la misma operación en modo PDU se realizaría de la siguiente forma:

AT+CMGS=25<CR>
0691055203200011000891958343470000AA0EC8373B0C3AD6C3F4723BCC0E03<Ctrl+Z>

2.3.4.2. Facilidad de uso

En el ejemplo anterior, se puede ver que la utilización del modo texto es mucho mas simplificada y clara que en el modo PDU. Ya que para crear un mensaje SMS en modo PDU es necesario comprender la estructura de los diferentes parámetros TPDU a nivel de bit de información, y codificación y decodificación de la secuencia hexadecimal.

2.3.4.3. Valores definidos en parámetros

Los valores que se utilizan en ciertos parámetros de configuración del módem GSM también son diferentes en modo texto y en modo PDU, por lo general en modo texto los parámetros se consulta y establecen por medio de cadenas de texto, mientras que en modo PDU estos mismos parámetros se representan en valores numéricos. Por ejemplo el comando +CMGL es utilizado para listar los mensajes SMS almacenados en el módem, para su uso es necesario enviar por medio de un parámetro el estado del mensajes que se desea recuperar, en la siguiente tabla se muestran los valores definidos en modo texto y PDU.

Tabla IX. **Valores definidos para estado SMS en modo texto y PDU**

Estado del mensaje	Valor en modo texto	Valor en modo PDU
Recibido sin leer	“REC UNREAD”	0
Recibido y leído	“REC READ”	1
Almacenado sin enviar	“STO UNSENT”	2
Almacenado enviado	“STO SENT”	3
Todos	“ALL”	4

Fuente: http://www.developershome.com/sms/operatingMode.asp.

Consulta: 31 de enero de 2014.

Para ver los mensajes recibidos sin leer en modo texto se realizaría así:

AT+CMGL=”REC UNREAD”<CR>

Y en modo PDU sería de la siguiente manera:

AT+CMGL=0<CR>

2.3.4.4. Soporte de características SMS

Desde el punto de vista de la compatibilidad sobre características SMS, el modo texto soporta menos funciones de mensajería SMS que el modo PDU y esto se debe a que no se tiene el control completo sobre los valores del encabezado y el cuerpo del mensaje SMS. Aunque algunas tareas se pueden realizar en modo texto, para realizar otras más específicas se requiere el conocimiento sobre el modo PDU y TPDU. Por ejemplo, para solicitar un reporte de estado la SMSC en modo texto, se debe de colocar el quinto bit del primer octeto de la TPDU *SMS-SUBMIT* a 1 utilizando el comando +CSMP.

Tareas similares incluyen el establecimiento del período de validez del mensaje y el envío de mensajes SMS distinto al estándar de clase 2, como lo son los mensajes *flash* que son inmediatamente mostrados en la pantalla cuando llegan al destino.

2.3.4.5. Soporte de los modos de operación

Desde el punto de vista de soporte, normalmente el modo PDU es más soportado en los módems GSM y en dispositivos móviles GSM, como los teléfonos celulares, que el modo texto.

2.4. Envío y lectura de SMS a través de un módem GSM

El envío y lectura de mensajes SMS son las dos operaciones más comunes que un módem GSM puede realizar en la manipulación de SMS, y estas se pueden realizar en dos modalidades que son:

2.4.1. Modo texto

Las operaciones realizadas en modo texto suelen ser prácticas para su uso, ya que se utilizan comandos de fácil entendimiento y con una sintaxis simple, a continuación se muestra la forma en que se realiza el envío y lectura de mensajes SMS en modo texto:

2.4.1.1. Envío de mensajes SMS

Para el envío de un de un mensaje SMS en modo texto, desde una consola con comunicación al módem GSM (ver sección 2.3.1), se realiza la siguiente secuencia de comandos AT:

Tabla X. **Secuencia de comandos para envío de SMS**

Comando a ejecutar	**Explicación del comando**
AT	Se inicializa el módem.
AT+CMGF=1	Aquí se le está indicando al módem que los próximos comandos a ejecutar serán en modo texto.
AT+CMGS=”59383474”<CR>Prueba<Ctrl+Z>	Donde “59383474” corresponde al número de teléfono al cual se envía el mensaje SMS. Prueba: es el mensaje a enviar.

Fuente: elaboración propia.

2.4.1.2. Lectura de mensajes SMS

Para la lectura de mensajes SMS en modo texto, se realiza a través de una secuencia de comandos AT, donde los primeros comandos preparan al módem para que posteriormente pueda leer los mensajes, se ejecuta la siguiente secuencia de comandos:

Tabla XI. **Secuencia de comandos para lectura de SMS**

Comando	Explicación del comando ejecutado
AT	Se inicializa el módem.
AT+CMGF=1	Se indica al módem que trabaje en modo texto.
AT+CPMS=”SM”	Define el espacio de memoria o área de almacenamiento donde se leerán o escribirán los mensajes SMS, solo existen 3 posibilidades: SM para indica que es en memoria de la tarjeta SIM; ME para indica que es en memoria del módem; y MT se utiliza para indicar que leerá en ambos.
AT+CMGL=”ALL”	Con este comando se le está indicando al módem que lea y muestre todos los mensajes, la indicación “ALL”, define el estado del mensaje a leer, en la tabla II se detallan los diferentes tipos.

Fuente: elaboración propia.

El resultado final será el despliegue de todos los mensajes SMS almacenados en memoria de la tarjeta SIM, se visualizara lo siguiente:

+CMGL: 1,"REC UNREAD","+50259383474",,"14/02/18,00:05:10+32"
Prueba de lectura SMS.
+CMGL: 2,"REC UNREAD","+50259383474",,"14/02/18,00:07:22+32"
Esta es otra prueba.

La sintaxis utilizada en el es: +CMGL: <MR>,<EstadoMensaje><Nombre del OA si existe en agenda contactos><OA>, <Fecha><CR><Datos>.

2.4.2. Modo PDU

El modo PDU permite la manipulación de todos los campos de la trama del mensaje SMS, por lo que puede ser útil, cuando se desee realizar tareas que no serian posibles utilizando el modo texto.

2.4.2.1. Envío de mensajes SMS

En esta modalidad, antes de poder envía el mensaje primero se debe codificar el mensaje, la trama PDU que se crea, no solo contiene el texto del mensaje sino que también incluye otros datos necesarios como: centro de servicio SMS, fecha y hora, tipo de mensaje, emisor, alfabeto, que permite la correcta gestión del mensaje. En la sección 1.6 del capítulo 1, se explicó detalladamente el formato PDU de un mensaje SMS, con lo cual se puede determinar el valor de cada uno de los campos que componen la trama PDU. La trama PDU a generar será en base a los siguientes datos:

Tabla XII. **Datos a codificar en trama PDU**

Parámetro	**Valor o configuración a utilizar**
Número SMSC	5025300200, con formato internacional, notación ISDN
Tipo PDU	SMS-SUBMIT, configuración: VPF con Formato Absoluto, RD para que eliminar duplicados
MR	La asignación de número de referencia, lo realiza el equipo.
Número destino	59383474
Información destino	Formato local, notación ISDN
PID	Protocolo SME-to-SME
DCS	Grupos de codificación de datos generales, sin comprimir, alfabeto por defecto, Clase de mensaje 2.
Fecha vencimiento	Hasta 2014-02-01 20:17:52 GMT -6
Longitud de datos	4 caracteres, 4 *bytes*
Datos a enviar	Hola

Fuente: elaboración propia.

Tabla XIII. **Datos codificados de trama PDU**

<table>
<tr><th>Campo</th><th>Valor</th><th>Calculo</th></tr>
<tr><td>SCA</td><td>069105520320
00</td><td>El número del SMSC a utilizar es 5025300200, para calcular la longitud se hace con la formula: (Digitos/2 + 1)= 10/2+1=6, se debe recordar que todos los valores son octetos por lo que se representara por 06 (00000110).
El valor tipo del SCA es 91 ya el número esta en formato internacional.
El paso final es transformar el número en su notación BCD que seria 0552032000.</td></tr>
<tr><td>Tipo PDU</td><td>19</td><td>Para calcular el tipo PDU, que consiste en indicar con 0 o 1 si se usa cada uno sus los parámetros definidos en la sección 1.6.2.

<table>
<tr><td>Bit 7</td><td>Bit 6</td><td>Bit 5</td><td>Bit 4</td><td>Bit 3</td><td>Bit 2</td><td>Bit 1</td><td>Bit 0</td></tr>
<tr><td>RP</td><td>UD HI</td><td>SR R</td><td colspan="2">VPF</td><td>RD</td><td colspan="2">MTI</td></tr>
<tr><td>0</td><td>0</td><td>0</td><td>1</td><td>1</td><td>0</td><td>0</td><td>1</td></tr>
</table>

De donde 00011001 su equivalente hexadecimal es 19. En este ejemplo se indica que el formato del período de valides VPF es el Absoluto = 11, y el tipo de mensaje es SMS-SUBMIT=01, un mensaje de envío.</td></tr>
<tr><td>MR</td><td>00</td><td>00, indicando que es el primer mensaje.</td></tr>
<tr><td>DA</td><td>088195834347</td><td>Número teléfono destino 59383474, primero se calcula longitud que es 08, luego el formato, en este caso es nacional por lo que se representa con 81, por último el número en notación BCD 95834347.</td></tr>
<tr><td>PID</td><td>00</td><td>Aquí se le asigna el valor 00.</td></tr>
<tr><td>DSC</td><td>12</td><td>Se utiliza la misma del ejemplo de la sección 1.6.6.3.</td></tr>
<tr><td>VP</td><td>412010027125
89</td><td>Este campo está relacionado al Tipo PDU, ahí se indica que el formato es Absoluto, la fecha a representar será 01/02/2014 20:17:52 GMT-6, mismo ej. de la sección 1.6.7.</td></tr>
<tr><td>UDL</td><td>04</td><td>Longitud del texto a enviar, en este ejemplo es de 4 caracteres y se representa 04.</td></tr>
<tr><td>UD</td><td>C8373B0C</td><td>Datos codificados utilizando el tipo de alfabeto indicado en DSC, codificación de 7 bits. El texto a enviar es “Hola”, el mismo del ej. de la sección 1.6.10. Hola=C8373B0C.</td></tr>
</table>

Fuente: elaboración propia.

El resultado es la trama PDU, que es una cadena compuesta de números hexadecimales 691055203200019000881958343470012412010027125890 4C8373B0C esta cadena de datos es la que se utilizará para el envío del SMS.

La secuencia de comandos a ejecutar para enviar el SMS se realiza en 4 líneas, la primera línea inicializa el módem, la segunda el indica el módem que los próximos comandos a ejecutar serán en modo PDU, en la tercera y cuarta línea se realiza el envío del SMS, usando el comando AT+CMGS, donde el número 22 corresponde a la longitud del mensaje, o cantidad octetos que utilizan sin contar la longitud de campo SCA.

AT<CR>
AT+CMGF=0<CR>
AT+CMGS=22<CR>
069105520320001900088195834347001241201002712 54A04C8373B0C
<CTRLZ>

2.4.2.2. Lectura de mensajes SMS

La lectura de mensajes en modo PDU, en cuanto a la extracción de los mensajes es similar al modo texto y en cuanto a la interpretación del mensaje, es muy similar al envío, con la diferencia que el proceso es inverso. Para la lectura de mensajes, se ejecuta la siguiente secuencia de comandos AT:

Tabla XIV. **Secuencia de comandos para el envío SMS**

Comando	**Explicación del mando ejecutado**
AT	Se inicializa el módem.
AT+CMGF=0	Aquí se le está indicando al módem que los próximos comandos a ejecutar serán en modo PDU.
AT+CPMS=4	Este comando define el espacio de memoria o área de almacenamiento donde se leerán o escribirán los mensajes SMS, solo existen 3 posibilidades: SM para indica que es en memoria de la tarjeta SIM; ME para indica que es en memoria del módem; y MT se utiliza para indicar que leerá en ambos.
AT+CMGL=4	Con este comando se le está indicando al módem que lea y muestre todos los mensajes, el número 4 indica el estado del mensaje a leer, en la tabla IX se detallan los diferentes tipos.

Fuente: elaboración propia.

El resultado final será el despliegue de todos los mensajes SMS almacenados en memoria de la tarjeta SIM en formato PDU, se visualizara lo siguiente:

+CMGL: 1,1,,31
06910552032000C40891214365870000414020610321 4A100500036F020
2C8E9B03CFD064567
+CMGL: 2,2,,16
07910552350022F2110008919583434700 00AA04C8373B0C
+CMGL: 3,2,,18
07910552350022F211000B910552393874F40000AA04C8373B0C
+CMGL: 4,2,,18
0691055203200011000B910552393874F40000AA04C8373B0C

La sintaxis utilizada es: +CMGL:<MR>,<Estado Mensaje>,,<Longitud del mensaje excluyendo SCA><CR><Datos en formato PDU>.

2.5. El centro de SMS y comandos AT

Una central de servicios de mensajes cortos o SMSC como ya se explicó en la sección 1.5.2.1, en la red de telecomunicaciones es la que se encarga de recibir y enviar los mensajes SMS a los usuarios, esta central en la red se identifica con un número de identificación único, el cual sirve para indicar al SME (el teléfono celular o módem) a quién tiene que enviar los mensajes SMS. Por lo tanto la importancia radia en la configuración del módem GSM, aunque esta es una configuración que el módem extrae de la tarjeta SIM, esta se puede establecer por medio de comandos AT+CSCA, de la siguiente forma:

AT+CSCA="+5025205280004",145

Donde el primer parámetro +5025205280004 es el ID o dirección de la SMSC que es similar a un número telefónico, como se puede observar a este número se le antepone el código de área, que para Guatemala es 502. El segundo parámetro indica el formato del ID, este puede ser 129 indicando que el formato usado es ISDN y puede ser un número nacional o internacional; 145 indica que el formato usado es ISDN y el número es internacional. Los números que identifican a la SMSC de Claro de Guatemala es +5025300200, para Tigo Guatemala es +5025205280004 y para Movistar Guatemala es +5026099902.

2.5. Pasarelas SMS

Es un servicio que permite a un ordenador envía o recibir mensajes SMS a través de internet para luego transferirlo a una red de telecomunicaciones. Por lo general este tipo de servicios son de paga, por medio de paquetes de mensajes transferidos.

Una pasarela SMS es una alternativa al uso de un módem GSM, o dicho de otra forma, parte del desarrollo del software del cual trata este documento es la creación de una pasarela SMS, aunque no se hable como tal, y que su capacidad de atención, en relación a carga de trabajo, es reducida ya que este solamente está integrado por un solo componente de hardware que es el módem.

La forma en que proveen la comunicación entre el ordenador y el servicio es a través de una variedad de protocolos de comunicación como páginas Web, Correo electrónico, Servicios Web o por medio de un componente API, cada uno con sus procedimientos para en envío y recepción de los mensajes SMS, cuando se habla de desarrollo las mas indicadas serian por medio de un Servicio Web o una API ya que facilitan la integración a cualquier proyecto de desarrollo. También los lenguajes de programación a los que dan soporte pueden variar, pero los más utilizados por lo general son: PHP, Java, .Net.

3. CREACIÓN DEL SISTEMA DE AVISOS

En este capítulo se expondrá los detalles del proyecto, necesarios para la creación del sistema de avisos. Primero se realiza una descripción del proyecto, descripción de las herramientas a utilizar y posteriormente se procede con el diseño del proyecto.

El objetivo de este capítulo es realizar una descripción de proyecto, los roles o entidades involucradas, sus funciones y procesos que realizan en torno a la operatividad del sistema de avisos.

3.1. Descripción del proyecto

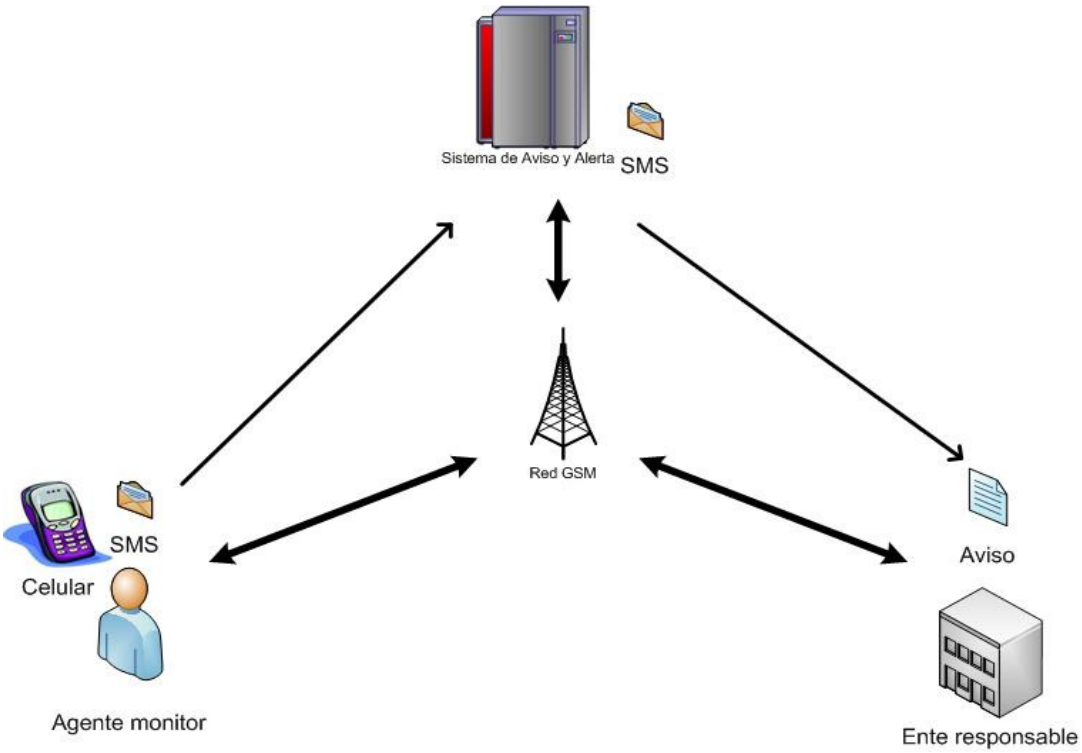
Se busca dar solución a la necesidad de tener un medio de comunicación ágil, accesible y práctico, que facilite la alerta para casos de emergencias. Para esto se propone la creación de un sistemas de aviso y alerta, orientada a eventualidades de índole medica que afecten a una comunidad, en la cual el acceso inmediato a atención medica no es posible, por factores de ubicación, falta de centros asistenciales, falta de recursos, o no pueden ser tratadas adecuadamente el en sitio.

La idea se basa en el uso, que hoy en día se da a los teléfonos celulares, específicamente el envío y recepción de mensajes SMS, que es común, simple y accesible para la población guatemalteca.

Esta característica puede ser utilizada como canal de comunicación para la creación del sistema de alerta, donde un agente monitor envía un mensaje SMS a una central, la cual recibe e interpreta el mensaje para posteriormente tomar una acción, esta acción consisten en generar una alerta o notificación dirigida a los entes encargados de atender dichas emergencias, en este caso una entidad que vela por el bienestar físico, la salud de las comunidades y tomen cartas en el asunto reportado.

La función del sistema de avisos y alerta es encargarse de trasladar el aviso reportado por el agente monitor y trasladarlo al ente responsable de atenderlo. El sistema de avisos y alerta, tiene el siguiente flujo de información:

Figura 12. **Flujo de información del sistema**



Fuente: elaboración propia, con Microsoft Office Visio 2007.

Donde el agente envía un SMS al sistema de aviso, este a su vez, dependiendo del tipo de eventualidad, genera y envía un mensaje de alerta (por correo y por SMS) al ente responsable, el contenido del aviso, indica el tipo de emergencia que se ha generado y el lugar donde ha ocurrido.

3.1.1. Antecedentes

Existen investigaciones previas relacionadas con este tema, en las tesis de Carmen Itzep[1] y Mónica Aragón[2], en ambas tesis proponen el establecimiento de procedimientos donde organizan e involucran a personas de las comunidades, para establecer un sistema de vigilancia y alerta nutricional. Aquí las personas de las comunidades se involucran para vigilar el estado alimenticio de sus comunidades, tomando acciones como las de dar aviso a organizaciones gubernamentales y no gubernamentales, para que les den su apoyo.

La relación que tiene con este documento, es que el sistema de aviso y alerta de emergencias utilizando mensajes SMS, el cual se propone desarrollar, es perfectamente aplicable a estas situaciones, claro que también depende de una organización por parte de las comunidades y de las organizaciones que atienden estas emergencias.

---

[1] ITZEP MANUEL, Carmen Beatriz. *Propuesta de un Sistema Municipal de Alerta Temprana de Inseguridad Alimentaria y Nutricional en Rabinal, Baja Verapaz*. Trabajo de graduación de Nutricionista. Universidad de San Carlos de Guatemala, Facultad de Ciencias Químicas y Farmacia. 2005. 74 p.

[2] ARAGÓN AGUILAR, Mónica. *Sistema de vigilancia alimentaria nutricional y de alerta temprana (Sisvan-at) para el programa de mejoramiento económico y seguridad alimentaria (promesa) de Care Guatemala*. Informe final de Maestría en Nutrición y Alimentación MANA. Universidad de San Carlos de Guatemala, Facultad de Ciencias Químicas y Farmacia. 2009. 71 p.

Este trabajo puede funcionar de forma complementaria, como herramienta a utilizar en las normas y procedimientos desarrollados en las tesis mencionadas anteriormente.

3.1.2. Premisas

Existe una serie de premisas y supuestos que son factores que se consideran de importancia, y que se deben cumplir para que el sistema a desarrollar funcione adecuadamente, estas son:

3.1.2.1. Comunidad organizada

Una comunidad organizada, la comunidad donde se implemente el sistema de alerta debe estar organizada, esto es esencial ya que ellos conocen la situación de su comunidad y deben interactuar con las organizaciones que atienden las emergencias.

Como parte de la organización de la comunidad debe de existir agentes monitores que tengan acceso a un teléfono celular, estos agentes son lo que monitorean y vigilan, atentos de las eventualidades que emerjan, estas son personas que residen en la misma comunidad, forman parte de comunidad, son líderes de las comunidades.

3.1.2.2. Organizaciones o entes responsables

Debe existe una entidad responsable y encargada de atender las emergencias, también podría ser un grupo de personas organizadas con el interés y la capacidad de atender estas emergencias.

En nuestro país una organización que debiese estar atento a emergencia de índole médica debería ser el Ministerio de Salud Pública y Asistencia Social, aunque claro que no son los únicos, ya que existen otras organizaciones con la finalidad de dar ayuda y soporte en este tema, algunas gubernamentales y otras de tipo ONG.

3.1.2.3. Grupo de emergencias a atender

Un grupo de emergencias especificas a atender, estas emergencias a tratar pueden ser de diferente índole, en materia de salud y medicina las emergencias pueden varias de una institución a otra, ya que su especialización o enfoque puede ser diferente, por mencionar algunos ejemplos, si el enfoque es la prevención de muerte en niños a causa de la desnutrición, las emergencias a atender podría ser:

- Desnutrición aguda moderada
- Desnutrición aguda moderada con complicaciones
- Desnutrición aguda severa con complicaciones

Por otro lado si el enfoque fuera la prevención, tratamiento y rehabilitación de niños, desde su gestación hasta la adolescencia, las emergencias podrían ser:

- Emergencia obstétrica preparto
- Dificulta para respirar, ataques de asma
- Desvanecimiento, desmayo del niño
- Reacción alérgica
- Fiebre alta
- Dificultar para despertarse

- Tos o vomito
- Fracturas
- Frecuencia cardiaca rápida

Para el caso de emergencias médicas en general, podría ser:

- Enfermedades respiratorias (neumonía, paros respiratorios)
- Problemas cardiacos
- Pedida de sangre, por heridas
- Pedida de conciencia
- Perforaciones torácicas o heridas penetrantes abdominales
- Fracturas graves o leves
- Quemaduras

Como se hizo mención en la introducción, también es posible aplicarlo a otro tipo de emergencias, como lo son las emergencias ocasionados por los desastres naturales, para poder identificarlo anticipadamente y prevenirlo, si ese fuera el caso, el listado de emergencias podría ser el siguiente:

- Deslaves
- Incendios forestales
- Actividad volcánica
- Inundaciones
- Tsunamis
- Tormentas

3.1.3. Actores involucrados

Un actor es una entidad externa (persona, organización, dispositivo o componente de software) que interactúa con el sistema. En el sistema que se describió anteriormente, existen dos entidades externas que interactúan con el sistema, y son los agentes monitores, que vigilan atentos de las eventualidades que emerjan, y los entes encargados de atender dichas emergencias, en este caso una organización que vela por el bienestar físico, la salud. De aquí ya se puede catalogar dos actores, los cuales son:

- Los agentes monitores: encargados de dar aviso, este es quien inicia el proceso de aviso.
- El ente u organización: que vela y se encarga de atender la emergencia reportada.

3.2. Recursos necesarios

Los recursos que se necesitan son de dos tipos de software y de hardware, el primero es necesario para la creación del sistema y el segundo para el funcionamiento del sistema.

3.2.1. Software

Las herramientas de desarrollo que se utilizará la plataforma de desarrollo Microsoft Visual Studio 2010, como base de datos se utilizará MySQL y para la administración y manipulación de la base de datos se usara la herramienta MySQL Workbench CE.

3.2.2. Hardware

Los recursos de hardware que son necesarios son una computadora o servidor, que tendrá las funciones de central y albergará el sistema de aviso y alerta, el módem GSM, dispositivo que permite la comunicación a través de la red GSM del sistema de telecomunicaciones guatemalteca.

El módem GSM en conjunto a la computadora y la aplicación a desarrollar, equivale a lo que se denomina SME, que se desarrollo en capítulo 1.

3.2.3. Otros

Existen otros recursos que son necesarios, además de los de hardware y software, estos son complementarios y están ligados al funcionamiento del sistema, estos recursos son:

- Una SIM de teléfono celular, de cualquiera de los operadores locales, que debe de tener saldo, necesario para el envío de mensajes SMS. La frecuencia de banda de operación del servicio asociado debe ser compatible con los soportados por el módem GSM, de lo contrario el módem no será operacional.
- Servicio de internet para el envío mensajes por medio de correo electrónico, este servicio se necesita en el servidor.
- Cada agente monitor debe de contar con un teléfono celular, con saldo, de cualquiera de las operadoras locales.

3.3. Diseño del proyecto

En esta sección se definen los componentes que integran el Sistema de Aviso y Alerta, con suficiente detalle como para permitir su interpretación y realización. Para una mejor compresión primero se explica la arquitectura de software del sistema, posteriormente se utiliza un diagrama de secuencia para mostrar la interacción del sistema, luego se detalla el comportamiento del sistema por medio de los diagramas de casos de uso y de actividades, y por último se define la estructura del sistema por medio del modelo entidad relación y del diagrama de clases.

3.3.1. Arquitectura de software del sistema

La arquitectura de software de sistema se plasma en la figura 13, en el diagrama se puede ver como fluye la información entre las diferentes capas de los cuales está compuesta la arquitectura del Sistema de Aviso y Alerta, este diagrama delimita la arquitectura del sistema por medio de capas que colaboran entre sí y en conjunto proveen la totalidad de funcionalidad del sistema. El sistema está compuesto por 5 capas:

- Capa de presentación: conformada por la interfaz que el sistema presenta, en la cual se monitorean los mensajes que entran al sistema y por la interfaz de usuario propio de cada teléfono celular, desde el cual los agentes monitores envía el mensaje de aviso, en esta capa el usuario interactúa directamente con él.

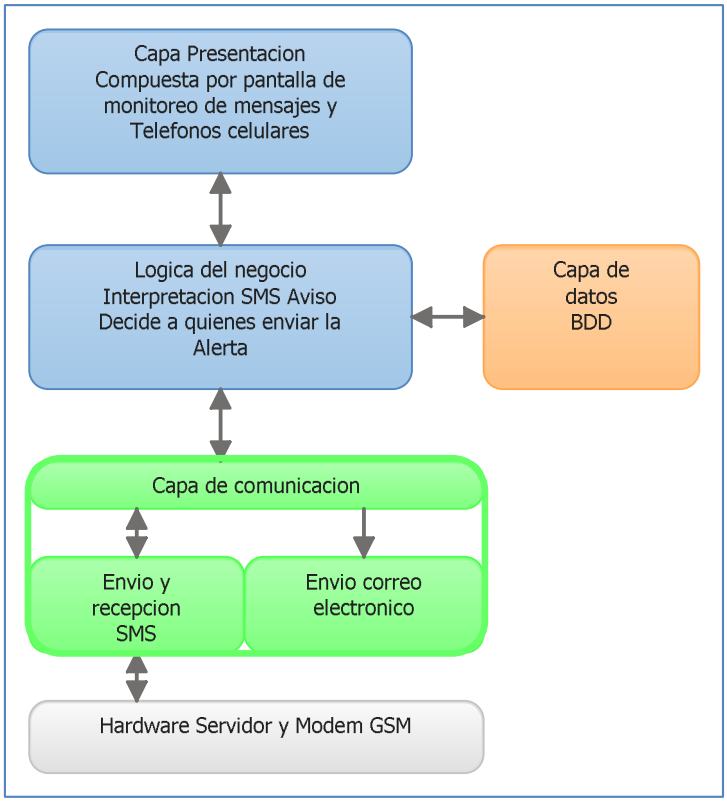
- Capa lógica del negocio: en esta capa se plasma todas la reglas del negocio, o sea la funcionalidad ofrecida por el sistema, y se encarga de las tareas relacionadas con la forma de interpretar el mensaje de aviso, que acciones tomar, a quienes enviar el mensaje de alerta. Como se ve en el diagrama esta capa interactúa con el usuario a través de la capa de presentación y no directamente, esta capa se considera una capa intermedia y es la de más importancia.

- Capa de datos o capa de acceso a los datos: su principal propósito consiste en separar el proveedor de los datos, para esta aplicación es MySQL, del resto de la aplicación, no contiene mucha complejidad y sus funciones son las de almacenar, consultar, actualiza y eliminar datos. Como se ve en el diagrama únicamente interactúa con la capa de lógica del negocio.

- Capa de comunicación: esta capa provee el enlace de comunicación y transferencia de datos, tiene dos funciones primarias que son el envío de mensajes por medio de correo electrónico y del envío y recepción de mensajes SMS, para lleva a cabo este último, interactúa con la capa de hardware.

- Capa de hardware: está compuesta por el equipo que funciona como servidor y el módem GSM, necesario para poder interactuar con la red GSM del sistema de telecomunicaciones.

Figura 13. **Arquitectura del sotware del sistema**



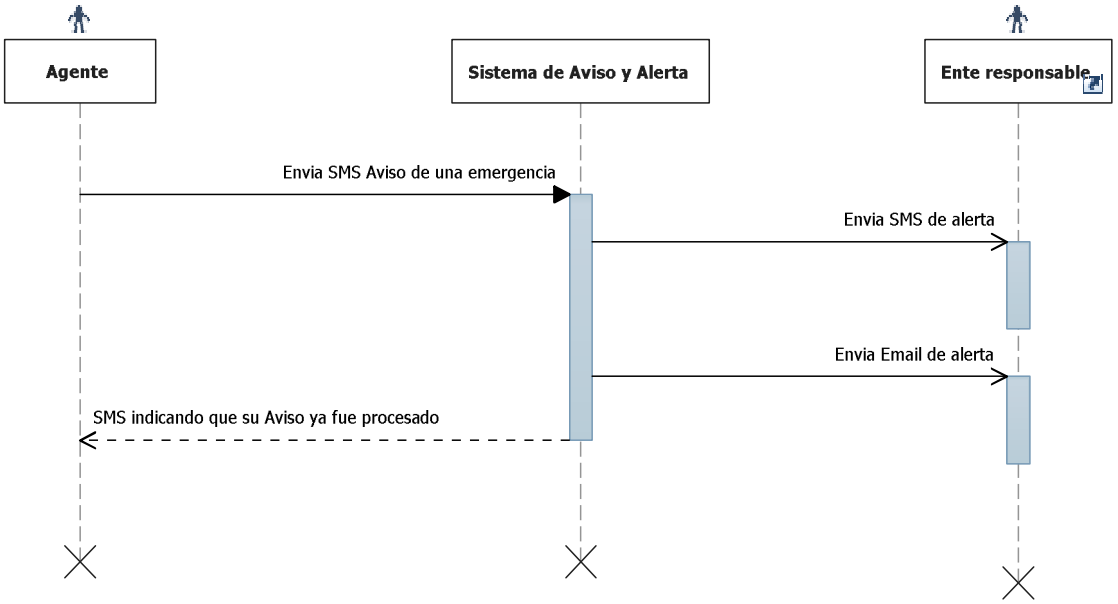
Fuente: elaboración propia, con Microsoft Visual Studio 2010.

3.3.2. Diagrama de secuencia

Anteriormente se identificaron dos actores o roles principales, estos son los agentes monitores y las entidades u organizaciones responsables de atender las emergencias reportadas. En el diagrama de secuencia de la figura 14 se muestra la interacción que el Sistema de Aviso y Alerta con los dos actores mencionados, primero se ve la interacción del agente con el sistema, donde el agente inicia la interacción enviando un mensaje SMS, en el cual especifica el tipo de emergencia y lugar donde ha ocurrido.

El mensaje llega al Sistema de Aviso y Alerta, el sistema interpreta y procesa el mensaje generando un mensaje de alerta, este nuevo mensaje es enviado por dos vías de forma simultánea, por SMS y por correo electrónico, que posteriormente envía al segundo actor, al ente responsable.

Figura 14. **Diagrama de secuencia**



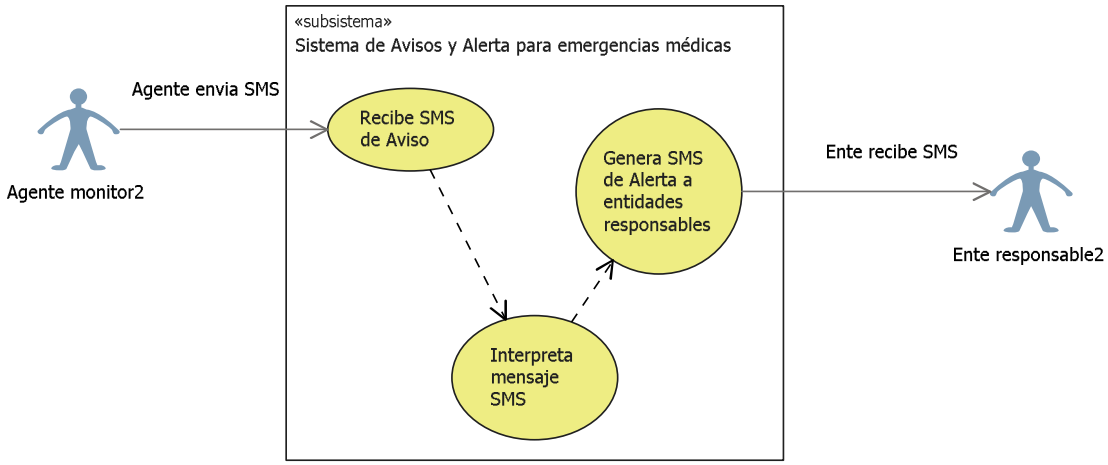
Fuente: elaboración propia, con Microsoft Visual Studio 2010.

Por último, luego que el mensaje de alerta ha sido enviado, el Sistema de Aviso y Alerta procede a envía un mensaje de acuse de recibido al agente responsable, donde indica que el mensaje de aviso ya fue procesado.

3.3.3. Casos de uso

En la figura 15 se da una vista general y gráfica del comportamiento del Sistema de Aviso y Alerta, ahí se describe la interacción entre los actores y los componentes primarios del sistema, se observa que el agente monitor envía un mensaje SMS de aviso que el sistema recibe, interpreta para posteriormente genera y enviar el mensaje de aviso al ente responsable de atender la emergencia reportada.

Figura 15. **Casos de uso del comportamiento del sistema**



Fuente: elaboración propia, con Microsoft Visual Studio 2010.

Para comprender el funcionamiento del sistema de forma amplia es necesario profundizar más, por lo que se muestra en la figura 16 un diagrama de casos de uso más detallado. La explicación del diagrama es la siguiente:

- Recibe SMS de aviso: este caso de uso representa el proceso que recibe el mensaje SMS de aviso, el cual envío el agente monitor, más adelante se podrá ver que la recepción del mensaje por parte sistema es realizado por medio de una bandeja de entrada de mensajes SMS, implementado en la clase GSM.

- Interpreta SMS: como su nombre indica este caso de uso es el que se encarga de procesar el contiendo del mensaje SMS de aviso, y se compone de los casos de uso Extrae datos SMS y Valida estructura.

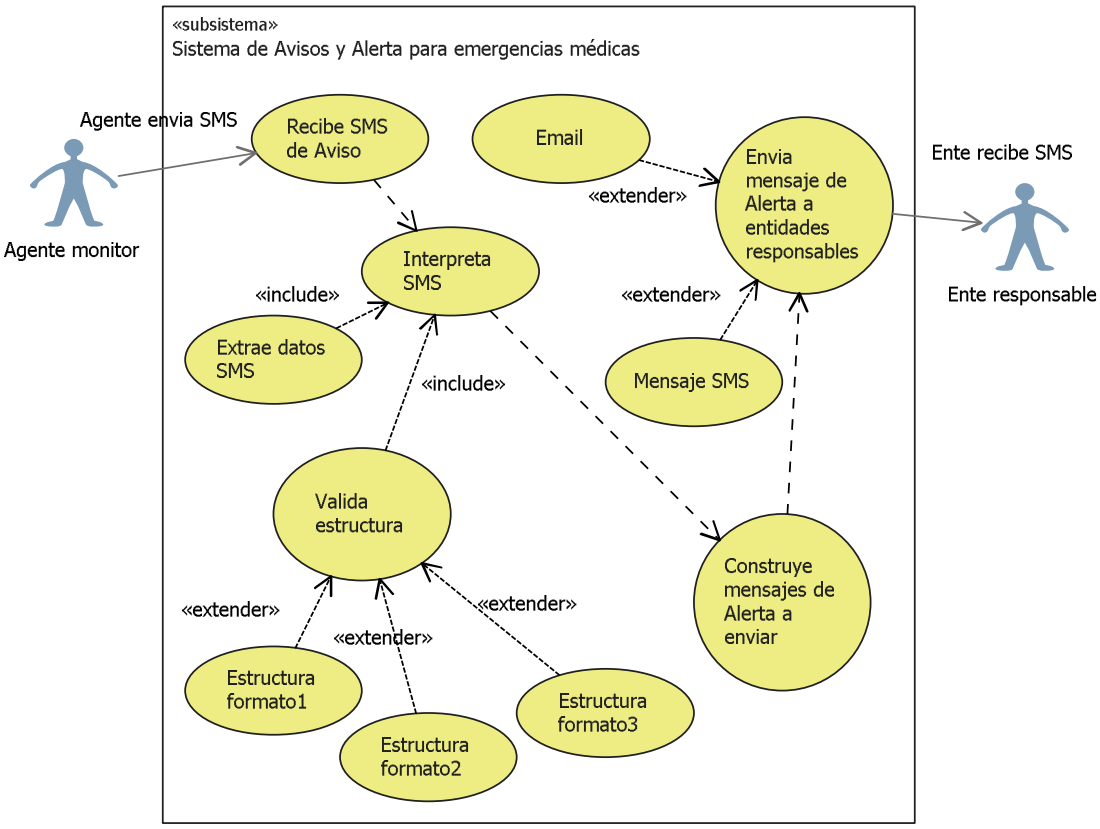
- Extrae datos: en este caso de uso se desglosa el mensaje SMS de aviso y se extraen los diferentes parámetros necesarios para su interpretación.

- Valida estructura: su función radica en la capacidad de discriminar el formato en el cual enviaron el mensaje SMS de aviso, ya que existe tres posibles formatos validos, de los cuales se puede extraer exactamente la misma información, aunque de diferente forma. Los casos de uso Estructura formato representan los posibles formatos a interpretar.

- Construye el mensaje de alerta a envía: este caso de uso representa la construcción del mensaje de alerta, del cual se arma a partir de un machote de texto almacenado en la base de datos.

- Envía mensaje de alerta a entidades responsables: representa el envío del mensaje de alerta al Ente responsable por medio de dos canales de comunicación, por correo electrónico y por medio de un mensaje SMS, ya que el medio de comunicación es diferente se realiza por medio de dos procesos especializados para cada medio, los cuales están representados por los casos de uso Email y Mensajes SMS.

Figura 16. **Casos de uso del comportamiento detallado del sistema**



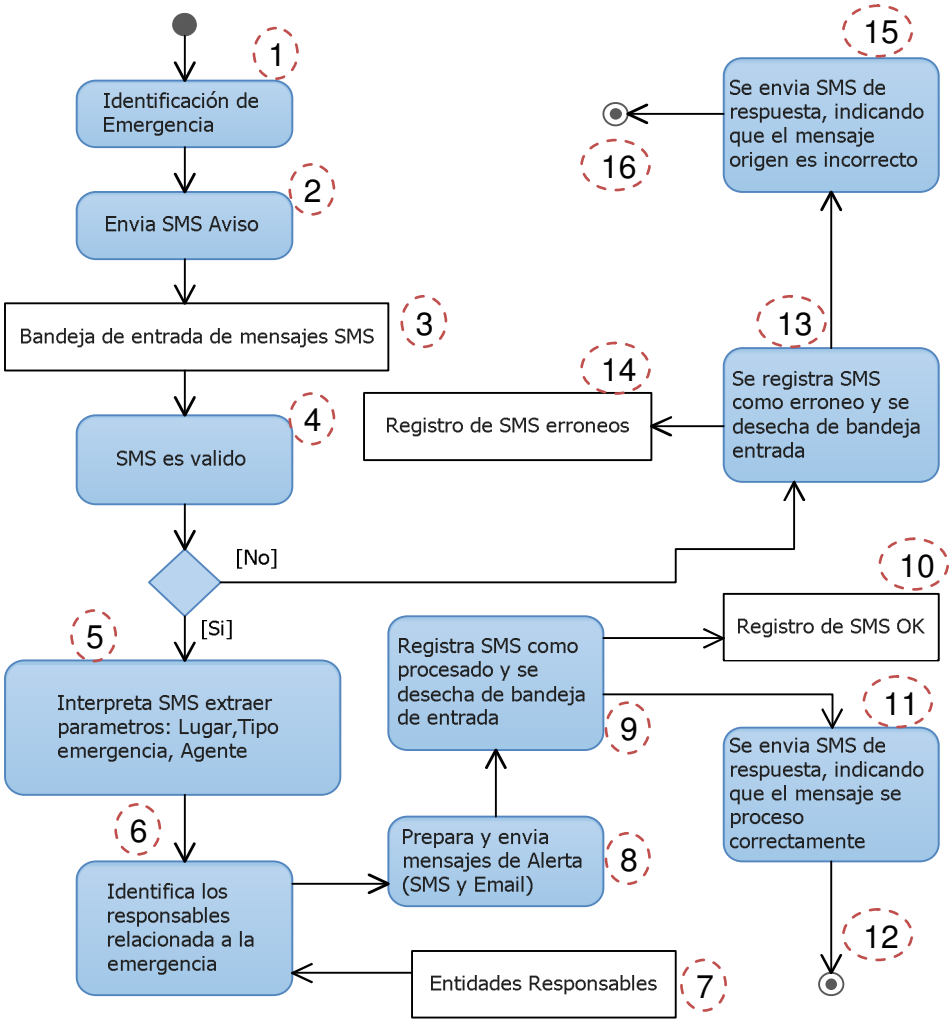
Fuente: elaboración propia, con Microsoft Visual Studio 2010.

3.3.4. Diagrama de actividades

Por medio de un diagrama de actividades se puede describir el comportamiento del sistema como un flujo de trabajo a través de una serie de pasos o acciones. A continuación se utilizara un diagrama de actividades para representar la misma información que se mostró en el diagrama de casos de uso pero a través de una vista diferente. En el diagrama de la figura 17 se ve cada uno de los pasos que se llevan a cabo en el sistema, desde la identificación de la emergencia hasta la entrega del mensaje de alerta.

Las primeras dos actividades son realizadas por el agente monitor, (1) el cual primero identifica el tipo de emergencia, (2) para luego enviar el mensajes SMS de aviso, (3) este mensaje posteriormente es recibido por el sistema y entra a una bandeja de entrada de mensajes SMS, (4) luego pasa por un proceso de validación, (5) si es válido entonces el mensaje se interpreta y se extraen los parámetros del mensaje.

Figura 17. **Diagrama de actividades**



Fuente: elaboración propia, con Microsoft Visual Studio 2010.

Luego de interpretar el mensaje, (6) se identifica a los responsables relacionado con la emergencia reportada, (7) para esto se consulta en la base de datos, (8) posteriormente se procede a prepara y enviar los mensajes de alerta que son un SMS y un correo electrónico, (9) el mensaje de alerta se desecha de la bandeja de entrada y se registra en la base de datos (10) indicando que el mensaje de aviso se proceso correctamente, (11) se envía un SMS de respuesta al agente monitor donde se indica que su aviso fue proceso correctamente, (12) ahí finaliza el proceso si el mensaje de alerta es válido.

En caso que el mensaje SMS de aviso no es válido (13) entonces el mensaje de alerta se desecha de la bandeja de entrada y se registra en la base de datos (14) como mensaje de aviso con errores, (15) se envía un SMS de respuesta al agente monitor donde se indica que su aviso no fue posible procesar por contener datos incorrectos, (16) ahí finaliza el proceso si el mensaje de alerta no es válido.

3.3.5. Modelo entidad relación

En este punto ya se ha definido la lógica funcional del sistema en relación a interacción y comportamiento, ahora toca definirlo de forma estructural, para esto primero se definiera el modelo de datos o diseño de la base de datos, por medio de un modelo entidad relación el cual representa la forma en que los datos se organizan y almacenan. En este diagrama cuando se habla de una entidad es la abstracción de objetos como por ejemplo conceptos, personas, objetos, del cual el sistema necesita guardar información, los atributos constituye cada una de las características y propiedades de una entidad, y las relaciones muestran el tipo de asociación que existe entre dos entidades.

Cada relación tiene una cardinalidad o número de ocurrencias que una entidad puede tener en otra y se representa de la siguiente forma:

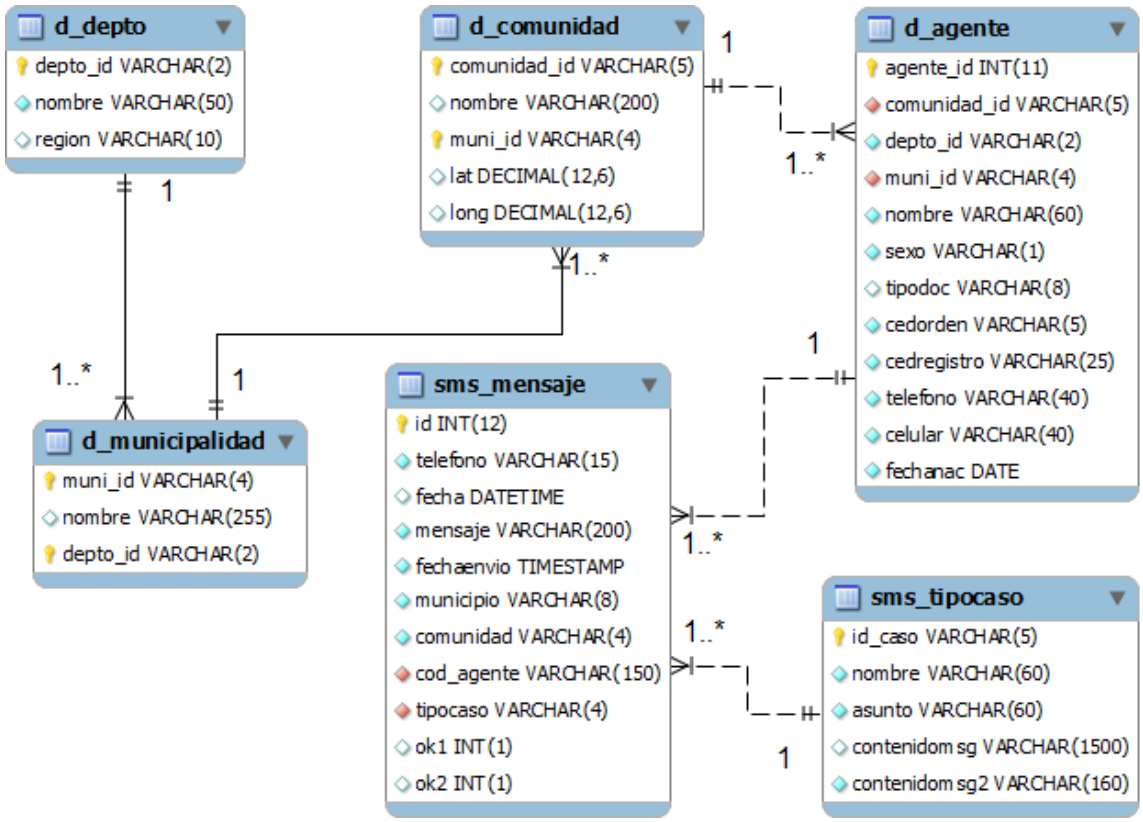
- 1 si cada instancia o registro de la entidad está obligada a participar solamente una vez en la relación.
- 1..* si cada instancia o registro de la entidad está obligada a participar una o más veces.

En el diagrama de la figura 18 se muestra las entidades con sus atributos que lo componen y sus relaciones entre sí. La relación entre la entidad d_depto y d_municipalidad es de 1 a muchos, lo que indica que por cada instancia de d_depto pueden existir 1 o más de d_municipalidad, en palabras simples indica que por cada departamento, pueden existir uno o más municipalidades asociadas a un departamento.

De la misma forma se relaciona la entidad d_municipalidad con d_comunidad, donde existen uno o más comunidades asociadas a una municipalidad. La relación entre d_comunidad y d_agente es del mismo tipo que las anteriores, por cada comunidad existen uno o más agentes.

La relación existente entre d_agente y sms_tipocaso se plasma una tercera entidad llamada sms_mensaje, donde se interpreta por cada d_agente existe uno a mas sms_mensajes y por cada sms_tipocaso existe uno o más sms_mensajes.

Figura 18. **Modelo entidad relación**



Fuente: elaboración propia, con MySQLWorkbench 5.2.31 CE.

Se hará uso de un diccionario de datos para describir cada uno de los atributos de las entidades que conforman el modelo entidad relación. La entidad d_depto representa los datos de todos los departamentos de Guatemala, su estructura es la siguiente:

Tabla XV. **Diccionario de datos entidad d_depto**

Atributo	Tipo dato	PK	NN	AI	Valor	Descripción
depto_id	varchar(2)	✔	✔			Código del departamento
Nombre	varchar(50)		✔		"	Nombre del departamento
Región	varchar(10)				"	Región donde se ubica

Fuente: elaboración propia.

La entidad d_municipalidad representa los datos de todas las municipalidades de cada uno de los departamentos, el atributo muni_id de esta entidad, tiene una particularidad y es que los primeros dos dígitos del código corresponde al código del departamento, su definición es la siguiente:

Tabla XVI. **Diccionario de datos entidad d_municipalidad**

Atributo	Tipo dato	PK	NN	AI	Valor	Descripción
muni_id	varchar(4)	✔	✔			Código del municipio, campo compuesto, los primeros 2 dígitos identifica al depto. y los restantes 2 al municipio
nombre	varchar(255)				"	Nombre del municipio
depto_id	varchar(2)	✔	✔		"	Código del departamento al cual pertenece

Fuente: elaboración propia.

La entidad d_comunidad representa los datos de las comunidades de cada uno de los municipios, en esta entidad se encuentran los atributos lat y long que representan las coordenadas geográficas de la comunidad y su función será de ubicar la posición exacta donde ocurrió la emergencia reportada, esto ayudaran a encontrar una comunidad que no sea muy conocida o muy alejada. Su definición es la que se muestra a continuación:

Tabla XVII. **Diccionario de datos entidad d_comunidad**

Atributo	Tipo dato	PK	NN	AI	Valor	Descripción
comunidad_id	varchar(5)	✔	✔			Código de la comunidad
nombre	varchar(200)				NULL	Nombre de la comunidad
muni_id	varchar(4)	✔	✔		''	Código del municipio
Lat	decimal(12,6)				NULL	Coord. latitud geográfica
Long	decimal(12,6)				NULL	Coord. longitud geográfica

Fuente: elaboración propia.

Las tres entidades que se explicaron anteriormente conforman los datos geográficos, la ubicación geográfica donde una emergencia puede ocurrir. Ahora se explicaran las otras entidades, d_agente entidad que representa los agentes monitores, sus atributos representan los datos generales del agente, su definición es la siguiente:

Tabla XVIII. **Diccionario de datos entidad d_agente**

Atributo	Tipo dato	PK	NN	AI	Valor	Descripción
agente_id	int(11)	✔	✔	✔		Código del agente monitor
comunidad_id	varchar(5)		✔			Código de la comunidad
depto_id	varchar(2)		✔		''	Código del departamento
muni_id	varchar(4)		✔		''	Código del municipio
Nombre	varchar(60)		✔			Nombre del agente monitor
Sexo	varchar(1)		✔		''	Sexo del agente
Tipodoc	varchar(8)				NULL	Tipo doc. de identificación
Cedorden	varchar(5)		✔		''	No orden del doc. Identificación
Cedregistro	varchar(25)		✔		''	Código de registro o DPI
Teléfono	varchar(40)		✔		''	Número de teléfono
Celular	varchar(40)		✔		''	Número de Celular
Fechanac	date		✔		NULL	Fecha de nacimiento

Fuente: elaboración propia.

La entidad sms_tipocaso representa las diferentes emergencias o casos que pueden existir, el atributo contenidomsg y contenidomsg2 son plantillas o machotes de texto que se utilizaran para crear el mensaje de alerta, el cual es enviara a los responsables de atender la emergencia.

Tabla XIX. **Diccionario de datos entidad sms_tipocaso**

Atributo	Tipo dato	PK	NN	AI	Valor	Descripción
id_caso	varchar(5)	✔	✔			Código del caso o Código de emergencia
nombre	varchar(60)		✔			Nombre de la emergencia
asunto	varchar(60)		✔		''	Texto asunto a mostrar en email
contenidomsg	varchar(1500)				NULL	Plantilla de texto para crear email alerta
contenidomsg2	varchar(160)		✔		''	Plantilla de texto para crear SMS alerta

Fuente: elaboración propia.

La entidad que representa los mensajes de aviso que se reciben y que son procesan por el sistema es sms_mensaje, los atributos ok1 y ok2 tendrán el valor de 0 si el mensaje no se envío y 1 si se envío con éxito.

El valor CTS que se observa en el atributo fechaenvio significa *Current Time Stamp* y se traduce como sello de tiempo actual o fecha y hora actual, y su función es la de registrar la fecha y hora en que el dato es registrado.

Tabla XX. **Diccionario de datos entidad sms_mensaje**

Atributo	Tipo dato	PK	NN	AI	Valor	Descripción
id	int(12)	✔	✔	✔		Código que identifica el mensaje procesado
telefono	varchar(15)		✔			No. celular quien envío el SMS aviso
fecha	datetime				NULL	Fecha del mensaje SMS
mensaje	varchar(200)		✔		''	Texto original que contenía el mensaje
fechaenvio	timestamp		✔		CTS	Fecha de envío del mensaje de alerta
municipio	varchar(8)		✔		''	Código del municipio donde se reporta alerta
comunidad	varchar(4)		✔		''	Código de la comuna donde se reporta alerta
cod_agente	varchar(150)		✔		''	Código del agente que reporto emergencia
tipocaso	varchar(4)		✔		''	Código del caso o Código de emergencia
ok1	int(1)				'0'	Indica que se envío mensaje email alerta
ok2	int(1)				'0'	Indica que se envío mensaje SMS alerta

Fuente: elaboración propia.

Por último está la entidad sms_contactos, y representa los datos de los responsables de atender las emergencias reportadas. Los atributos de importancia son zonageografica que define que departamento o municipio tiene a cargo, celular y correo, donde enviaran el mensaje de alerta por mensaje SMS y por correo electrónico respectivamente.

Tabla XXI. **Diccionario de datos entidad sms_contactos**

Atributo	Tipo dato	PK	NN	AI	Valor	Descripción
contacto_id	int(11)	✔	✔	✔		Código del ente responsable
nombre	varchar(60)		✔			Nombre del ente responsable
Titulo	varchar(15)		✔		''	Titulo asociado al ente responsable
Cargo	varchar(50)		✔		''	Cargo del responsable
zonageografica	varchar(4)		✔		''	Zona geográfica del que es responsable, aquí se indica el Código del departamento o municipalidad
telefono	varchar(40)		✔		''	Teléfono del entre responsable
celular	varchar(40)		✔		''	Celular del entre responsable
correo	varchar(150)		✔		''	Dirección email del entre responsable

Fuente: elaboración propia.

Eso es todo lo relacionado al modelo y estructura de la base de datos, ahora se definirá la estructura del sistema por medio de un diagrama de clases.

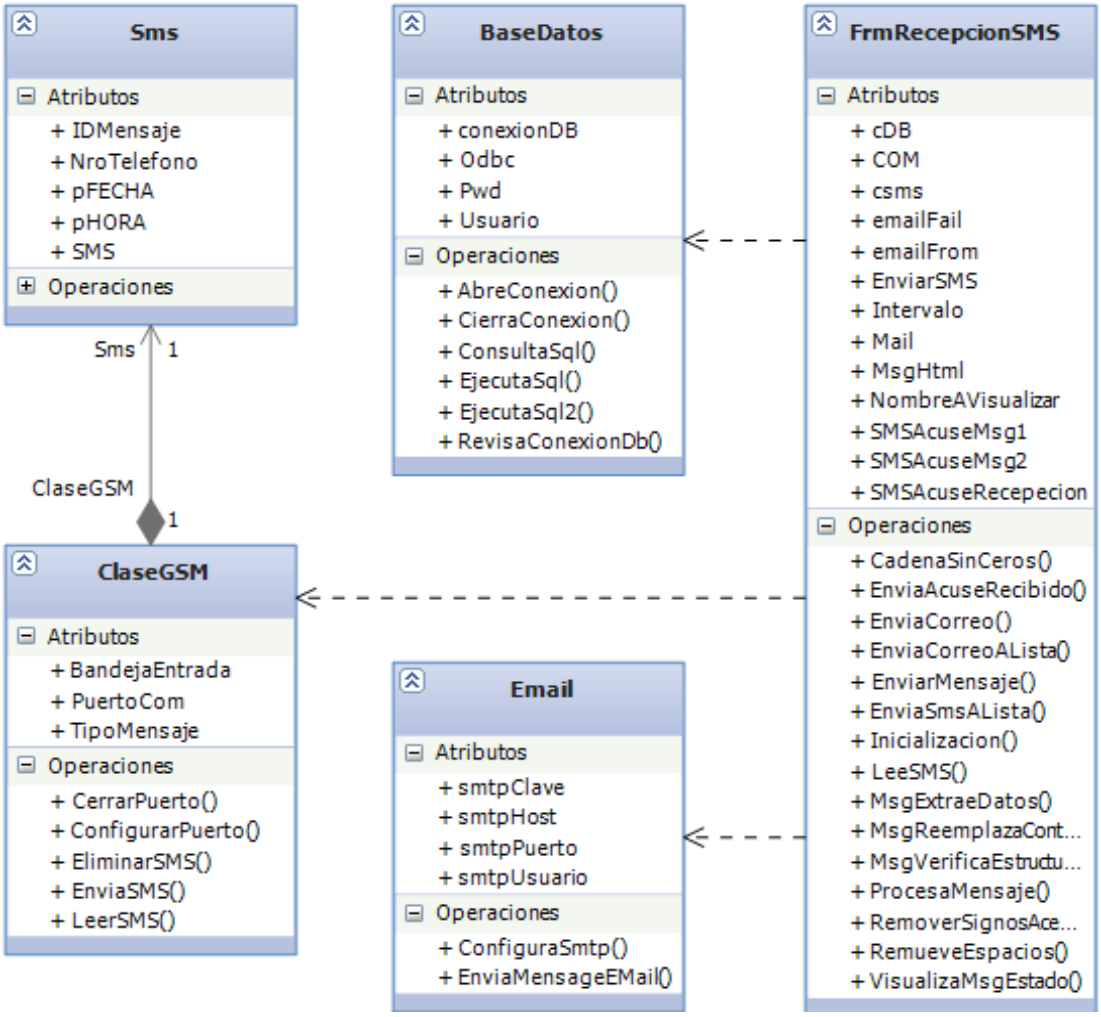
3.3.6. Diagrama de clases

En este diagrama se ven las clases involucradas, sus propiedades, operaciones y sus relaciones estructurales. Se describe los componentes de objeto de software y las estructuras de información que se utilizan en el sistema de forma interna. El diagrama que se muestran en la figura 19, es un diagrama de clases, que muestra la interacción que tienen las clases del sistema, también se aprecian sus atributos y operaciones, se puede ver las siguientes relaciones entre las clases:

- Sms y ClaseGSM: relación de agregación y composición, la clase Sms es parte de ClaseGSM.

- FrmRecepcionSMS y ClaseGSM: relación de dependencia (uso), la clase FrmRecepcionSMS utiliza la clase ClaseGSM.
- FrmRecepcionSMS y Email: relación de dependencia (uso), la clase FrmRecepcionSMS hace uso de la clase Email.
- FrmRecepcionSMS y BaseDatos: relación de dependencia, la clase FrmRecepcionSMS utiliza la clase BaseDatos.

Figura 19. **Diagrama de clases**



Fuente: elaboración propia, con Microsoft Visual Studio 2010.

3.3.6.1. Clase Sms

Esta clase representa la estructura del mensaje SMS y es utilizada por la clase ClaseGSM, la función de esta es únicamente la de contener y almacenar los datos que contiene un SMS, su estructura es la siguiente:

Tabla XXII. **Diccionario de datos de clase Sms**

Nombre	Sms	
Atributos		
Atributo	**Tipo**	**Descripción**
IDMensaje	Texto	Id o código del mensajes
NroTelefono	Texto	No de teléfono origen del SMS
SMS	Texto	Texto del mensaje
pFECHA	Texto	Fecha de recepción del mensaje
pHORA	Texto	Hora de recepción del mensaje
Métodos/Operaciones		
Método	**Descripción**	
IDMensaje	Retorna o asigna el código del mensaje	
NroTelefono	Retorna o asigna el número de teléfono	
SMS	Retorna o asigna el texto del mensaje	
pFECHA	Retorna o asigna la fecha del mensaje	
pHORA	Retorna o asigna la hora del mensaje	

Fuente: elaboración propia.

3.3.6.2. Clase ClaseGSM

La clase ClaseGMS, es la encargada de la manipulación del módem GSM, realiza la tareas de lectura, escritura y eliminación de mensajes SMS, es una de las clases más importantes del sistema, y es aquí donde se aplica la teoría vista en los primeros capítulos tal como el uso de los comandos AT, necesario para el envío y recepción de mensajes SMS.

Los mensajes que ingresan al módem son almacenados en una bandeja de entrada de mensajes SMS, la razón es porque es mucho más práctico manipular la bandeja de mensajes ya que esta tiene una estructura bien definida, mientras el mensaje SMS original contiene todos los datos (teléfono, fecha, hora) mezclados en un texto plano. La definición de la clase es:

Tabla XXIII. **Diccionario de datos de clase ClaseGSM**

Nombre	ClaseGSM	
Atributos		
Atributo	**Tipo**	**Descripción**
BandejaEntrada	Sms	Es una colección o arreglo de datos de tipo Sms
PuertoCom	SerialPort	Objeto para controlar el puerto serial al cual está conectado el módem GSM
TipoMensaje	Constante	Constante que indica el tipo de mensaje a procesar
Métodos/Operaciones		
Método	**Descripción**	
CerrarPuerto	Termina la comunicación con el puerto COM	
ConfigurarPuerto	Asigna los valores básicos al puerto COM, tales como número de puerto, frecuencia, paridad, tiempo de espera de conexión.	
EliminarSMS	Elimina un SMS del módem, indicando el índice del mensaje a eliminar, devuelve verdadero si elimina el mensaje y falso si falla.	
EnviaSMS	Envía un mensajes de texto SMS, utiliza dos parámetros el teléfono al que se envía el mensaje y el texto del mensaje, devuelve verdadero si envía el mensaje y falso si falla el envío.	
LeerSMS	Lee los mensajes de texto SMS del módem GSM y los envía a la bandeja de entrada, utiliza un parámetro el cual indica el tipo de mensaje a leer.	

Fuente: elaboración propia.

3.3.6.3. Clase Email

Esta clase tiene el propósito de enviar correo electrónico a un destino, es una clase relativamente simple, sus atributos son los datos de conexión y sus métodos son el de configuración y envío. El detalle de sus elementos se muestra a continuación:

Tabla XXIV. **Diccionario de datos de clase Email**

Nombre	Email	
Atributos		
Atributo	**Tipo**	**Descripción**
smtpHost	Texto	Dirección del servidor SMTP
smtpPuerto	Texto	Puerto utilizado para realizar la conexión SMPT
smtpUsuario	Texto	Cuenta de correo a utilizar para el envío de correo electrónico.
smtpClave	Texto	Clave de la cuenta de correo electrónico
Métodos/Operaciones		
Método	**Descripción**	
ConfiguraSmtp	Asigna la dirección del host, número de puerto, usuario y clave para realizar la conexión SMTP	
EnviaMensageEMail	Envía un mensaje por correo electrónico, devuelve verdadero si envía el mensaje y falso si falla el envío	

Fuente: elaboración propia.

3.3.6.4. Clase BaseDatos

La interacción con la base de datos MySQL es provista por esta clase, también es una clase de poca complejidad. Su función es la de proveer un enlace de comunicación entre la aplicación y la base de datos, esta clase corresponde a la capa de datos que muestran en la arquitectura del sistema expuesta en la sección 3.3.1. Se compone de los siguientes elementos:

Tabla XXV. **Diccionario de datos de clase BaseDatos**

Nombre	BaseDatos	
Atributos		
Atributo	**Tipo**	**Descripción**
conexionDB	OdbcConnection	Conexión a BD por medio de OBDC
Odbc	Texto	Nombre del ODBC a utilizar
Usuario	Texto	Usuario de la base de datos
Pwd	Texto	Clave del usuario de la base de datos
Métodos/Operaciones		
Método	**Descripción**	
AbreConexion	Establece conexión con la BDD, tomando los parámetros configurados en el archivo de configuración	
CierraConexion	Finaliza la conexión a la BDD	
ConsultaSql	Elimina un mensaje SMS del módem GSM, indicando el índice del mensaje a eliminar	
EjecutaSql	Ejecuta una instrucción SQL (select) en la BD, retornando el valor indicado por Índice	
EjecutaSql2	Ejecuta una instrucción SQL (insert, delete, exec) en la BD, retornando si se ejecuto con éxito(falso o verdadero)	
RevisaConexionDb	Ejecuta un consulta SQL(select) y devuelve los datos en un OdbcDataReader	

Fuente: elaboración propia.

Sus atributos son los necesarios para realizar la conexión a la base de datos por medio de un ODBC y sus métodos proporcionan la funcionalidad necesaria para consultar, agregar, editar o eliminar datos.

3.3.6.5. Clase frmRecepcionSMS

Esta es un clase muy importante, es la que integra toda la funcionad, y por lo tanto hace uso de todos los otros componentes. Si se realiza una comparación entre el diagrama de casos de usos y los métodos de esta clase se observara que proporciona la misma funcionalidad y comportamiento.

La funcionalidad de esta clase se resume en la recepción del mensaje SMS de aviso, que luego interpreta el SMS, extrae datos del SMS y valida estructura, construye mensaje de alerta y por último envía mensaje de alerta a entidades responsables.

Tabla XXVI. **Diccionario de datos de clase frmRecepcionSMS**

Nombre	frmRecepcionSMS	
Atributos		
Atributo	**Tipo**	**Descripción**
cDB	BaseDatos	Instancia de Clase BaseDatos, conexión a la base de datos
COM	Texto	Puerto COM a utilizar
csms	ClaseGSM	Instancia de Clase GMS para el envío y recepción de SMS
emailFail	Texto	Dirección de Correo electrónico, a utilizar cuando falla el proceso de un SMS
emailFrom	Texto	Dirección de Correo electrónico, a utilizar como origen (quien envía los mensajes)
EnviarSMS	Falso, verdadero	Indica si se envía SMS después de enviar correo electrónico, en la notificaciones
Intervalo	Numérico	Intervalo en segundos, que se revisar por nuevos mensajes SMS
Mail	eMail	Instancia de clase eMail a utilizar para envío de correo electrónico
MsgHtml	Falso, verdadero	Indica si el Mensaje a enviar por Email será Html o no
NombreAVisualizar	Texto	Nombre a visualizar de la cuenta email From (nombre del usuario emailFrom)
SMSAcuseMsg1	Texto	Texto de mensaje a utilizar en los SMS de acuse de recibido, cuando es exitoso
SMSAcuseMsg2	Texto	Texto de mensaje a utilizar en los SMS de acuse de recibido, cuando falla el proceso

Continuación de la tabla XXVI.

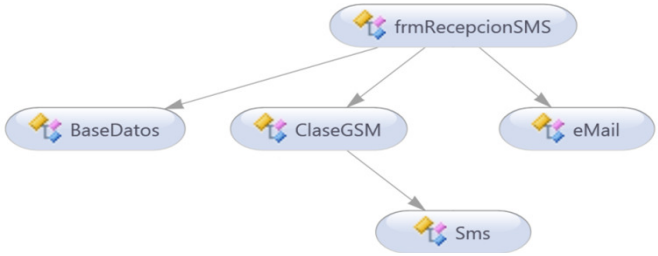
SMSAcuseRecepecion	Falso, verdadero	Indica si se utiliza SMS para acuse de recibido (envía SMS al teléfono origen)
SinSignos	Constante	Caracteres a reemplazar en SMS, son caracteres no validos, ConSignos es reemplazado por SinSignos
ConSignos	Constante	
Métodos/Operaciones		
Método	**Descripción**	
CadenaSinCeros	Remueve el caracteres 0 de una cadena	
DarValor	Devuelve el valor de un campo de la tabla indicada de la base de datos.	
EnviaAcuseRecibido	Envía SMS de acuse de recibo, a quien origino el mensaje.	
EnviaCorreo	Envía un correo electrónico	
EnviaCorreoAListа	Envía email a la lista de contactos, según zona geográfica	
EnviarMensaje	Envía mensajes de texto SMS a el número especificado	
EnviaSmsAListа	Envía SMS a la lista de contactos, según zona geográfica	
Inicializacion	Configuración inicial para la base de datos, módem, email	
LeeSMS	Lee mensajes de texto SMS del módem	
MsgExtraeDatos	Extrae solo los datos útiles del mensaje, remueve espacios en blanco, y divide el mensaje en los parámetro enviados	
MsgReemplazaContenido	Reemplaza los <tags> del mensaje original por los valores correspondientes.	
MsgVerificaEstructura	Verifica el formato del mensaje, que tenga todos los campos necesarios	
ProcesaMensaje	Procesa el Mensaje recibido: Interpreta, inserta en BDD, genera/envía SMS y email a destinatario	
RemoverSignosAcentos	Quita acentos y otros signos a una cadena	
RemueveEspacios	Quita los espacios en blanco que están sobrantes	
VisualizaMsgEstado	Muestra los mensaje que se procesan en un textbox	
frmRecepcionSMS_Load	Llama al método de Inicializacion	
Tmr1_Tick	Temporizador que revisar constantemente la bandeja de entradas de SMS	

Fuente: elaboración propia.

3.3.7. Dependencia de métodos entre clases

Este apartado tiene como objetivo mostrar de forma visual, las dependencias que existen entre los métodos de todas las clases que se han descrito, la siguiente figura muestra la relación que existe entre las clases:

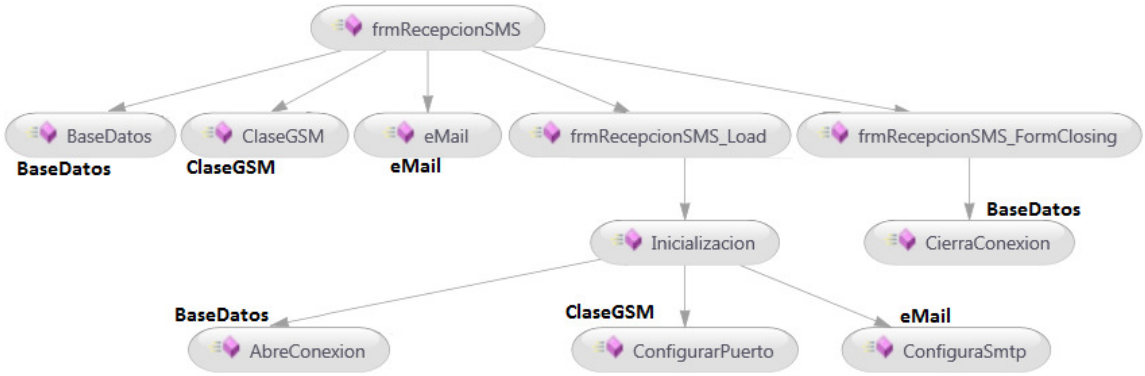
Figura 20. **Relación entre clases**



Fuente: elaboración propia, con Microsoft Visual Studio 2010.

La figura 21 muestra como se relaciona frmRecepcionSMS con los métodos de las otras clases (en negrilla se resalta la clase a la que pertenece).

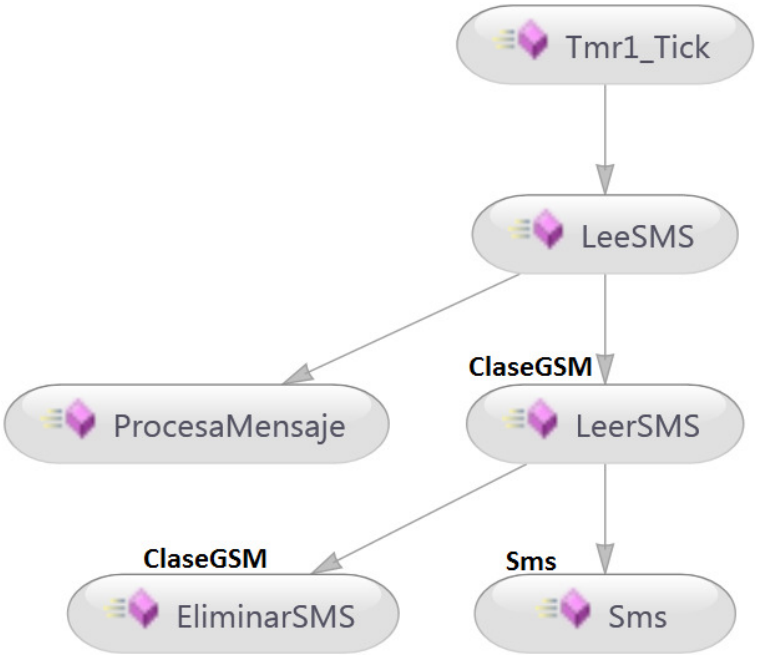
Figura 21. **Relación entre frmRecepcionSMS y el resto de clases**



Fuente: elaboración propia, con Microsoft Visual Studio 2010.

El método Tmr1_tick de frmRecepcionSMS se puede considerar independiente, en cuanto a la invocación del método, ya que este se ejecuta periódicamente de forma automática, ningún otro método lo invoca.

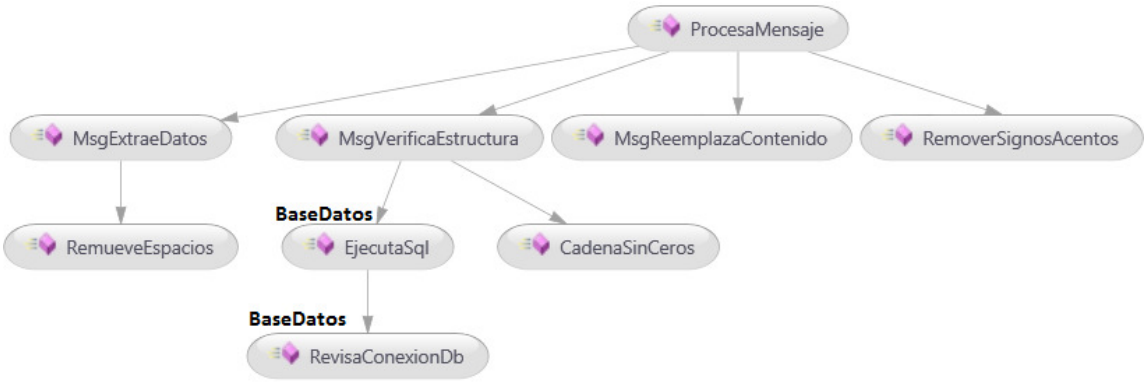
Figura 22. **Relación de método Tmr1_tic con otros métodos**



Fuente: elaboración propia, con Microsoft Visual Studio 2010.

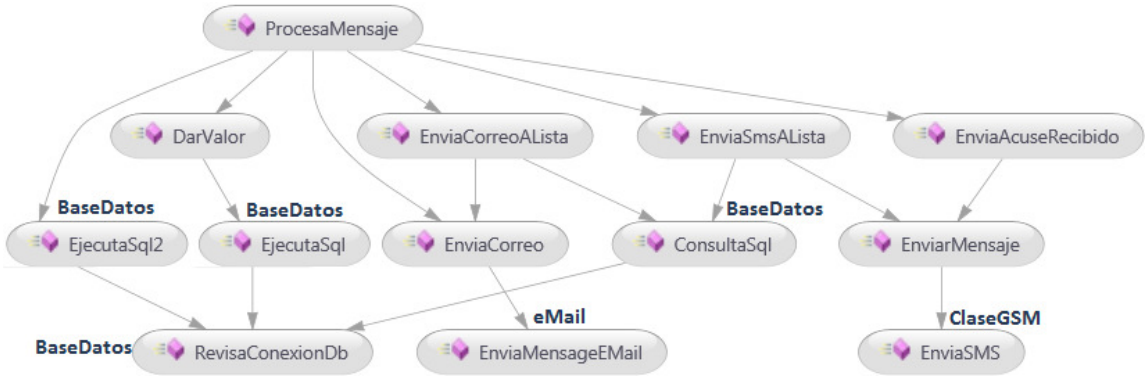
Hablando únicamente de la clase frmRecepcionSMS, el método ProcesaMensaje es el de mayor importancia y complejidad, en las siguientes dos figuras se muestra la relación que tiene el método ProcesaMensaje y los otros métodos.

Figura 23. **Relación de ProcesaMensaje con otros métodos parte 1**



Fuente: elaboración propia, con Microsoft Visual Studio 2010.

Figura 24. **Relación de ProcesaMensaje con otros métodos parte 2**



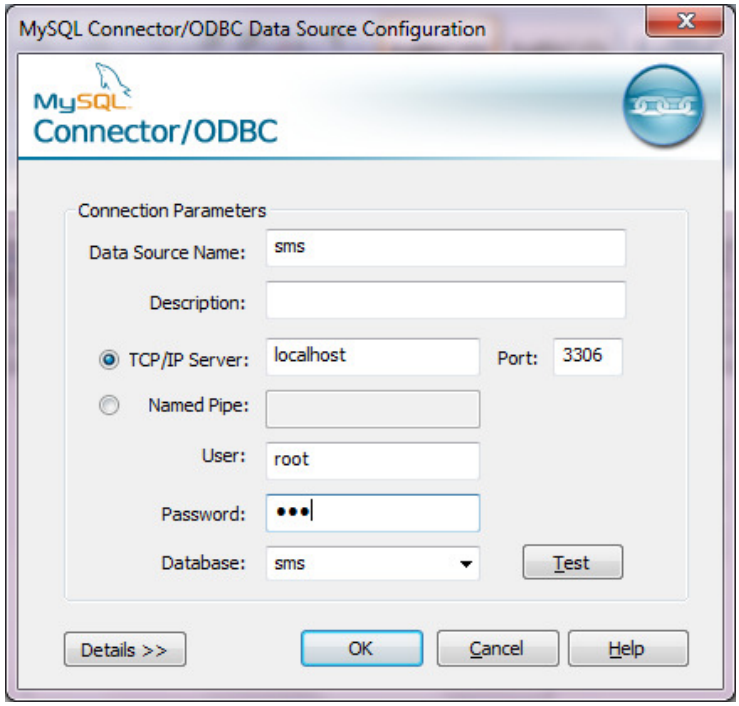
Fuente: elaboración propia, con Microsoft Visual Studio 2010.

Todas las relaciones que se observan en la figura 23 y 24 son de dependencia o uso, por ejemplo, la clase ProcesaMensaje depende de las clases EnviaAcuseRecibido, EnviaCorreoALista, EnviaCorreo, EnviaSmsALista, DarValor y BaseDatos.

3.4. Conexión a la base de datos

La base de datos a utilizar es MySql que es un DBMS y el sistema se conecta a ella por medio de un ODBC. Antes de finaliza el capítulo y pasar a la codificación, se explicar cómo crear el ODBC de conexión a la base de datos MySQL. Para crear un ODBC, abra Orígenes de datos (ODBC) que se ubica en el Panel de control, Herramientas administrativas. Seleccione DSN de sistema, luego presione en el botón Agregar, se mostrara la siguiente pantalla:

Figura 25. **Creación de ODBC para MySQL**



Fuente: elaboración propia.

Ingrese el nombre del ODBC, la descripción es opcional, en el campo TPC/IP Server debe ingresar el nombre del servidor alberga la base de datos, el usuario de la base de datos y su respectiva clave, al presionar en Test, dará un mensaje indicando que se realizó la conexión con éxito.

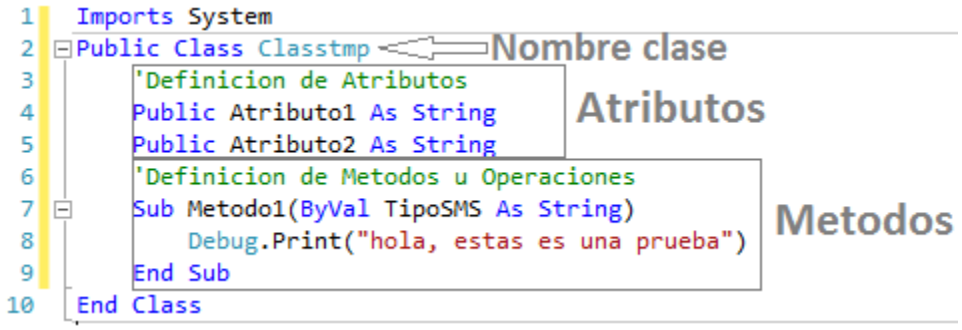
4. CODIFICACIÓN Y PRUEBAS

4.1. Generalidades

Al inicio del capítulo anterior, se dijo que se utilizaría Visual Studio 2010, como herramienta de desarrollo y se utilizará como lenguaje de programación Visual Basic y la plataforma .NET. Para que cada formulario, módulo o clase del proyecto funcione adecuadamente necesita hacer uso de espacio de nombre o *namespace* (en inglés), los cuales proveen acceso a un conjunto de librerías de la plataforma .Net y permiten utilizar las funciones, clases, estructuras, tipos de datos propias de cada espacio de nombres. Para hacer referencia a un espacio de nombres, al inicio del archivo de código fuente se utiliza la palabra reservada *imports* seguido por el espacio de nombres, ejemplo: *Imports System.IO.Ports*.

La estructura o definición de una clase es la siguiente:

Figura 26. **Estructura de una clase**



Fuente: elaboración propia.

En los siguientes apartados, no se detallara en la explicación de los atributos, ya que estos son los mismos definidos en el diccionario de datos. En este capítulo cuando se hable de código se hace referencia al código fuente y cuando se haga mención de métodos se refiere a funciones o procedimientos y viceversa.

4.2. Codificación de la clase ClaseGSM

En esta sección se muestra y explica el código generado de cada uno de los elementos de la clase ClaseGSM como lo son sus atributos y métodos definidos en la etapa de diseño expuestos en el capítulo anterior. Los espacios de nombres que necesita esta clase son:

- *System.IO.Ports*
- *System.Threading*

Los atributos que utiliza son: BandejaEntrada, TipoMensaje y PuertoCom.

Figura 27. **Declaración de atributos de ClaseGSM**

```
'Es una colección o arreglo de datos de tipo Sms
Public BandejaEntrada As New List(Of Sms)
'Tipo de mensajes a leer del modem
Public Class TipoMensaje
    Public Const Recibidos_Leidos As String = "REC READ"
    Public Const Recibidos_No_Leidos As String = "REC UNREAD"
    Public Const Enviados As String = "STO UNSEND"
    Public Const Escritos_Sin_Enviar As String = "STO SENT"
    Public Const Todos As String = "ALL"
End Class
'Puerto COM
Friend WithEvents PuertoCom As System.IO.Ports.SerialPort
```

Fuente: elaboración propia.

El método ConfigurarPuerto, asigna los parámetros de comunicación al puerto serial, como el nombre del puerto COM a utilizar al que esté conectado el módem GSM, si tiene alguna duda consulte la sección 2.3. Otro parámetro que asigna es la velocidad de comunicación, aquí se debe de asignar una velocidad que sea compatible por el módem, si se utiliza uno incorrecto el módem no funcionario o lo hará de forma inestable, por lo que se recomienda utilizar el valor de 9 600 ya que es soportado por la mayoría de módem actuales.

El incrementar el valor de velocidad de comunicación puede hacer que la comunicación entre la computadora y el módem sea más rápida, pero a mayor velocidad existe una mayor probabilidad de pérdida de datos.

El valor del parámetro *DataBits* indica el número de bits a utilizar en codificación de cada carácter de los datos transferidos, y puede variar entre 5 y 8, aquí lo recomendable es utilizar 8 ya que es el estándar más utilizado hoy en día. *Parity* o paridad es un método de detección de errores en la transmisión, no se está utilizando. *StopBits* indica el número de bits de parada, o sea la cantidad de bits que existen entre byte y byte de información enviada. ReadTimeOut y WriteTimeOut indican el tiempo de espera en ms, que deben transcurrir antes de cancelar la operación de lectura o escritura respectivamente. *Newline* indica el código de carácter utilizado para interpretar el retorno de carro o fin de línea.

Figura 28. **Código de método ConfigurarPuerto**

```
'settea los valores basicos para el puerto COM
Sub ConfigurarPuerto(ByVal COM As String, Optional ByVal Baudrate As Integer = 9600)
    PuertoCom = New SerialPort
    With (PuertoCom)
        .PortName = COM
        .BaudRate = Baudrate
        .Parity = Parity.None
        .DataBits = 8
        .StopBits = StopBits.One
        .Handshake = Handshake.RequestToSend
        .ReadTimeout = 300
        .WriteTimeout = 300
        .DtrEnable = True
        .RtsEnable = True
        .NewLine = vbCrLf
    End With
End Sub
```

Fuente: elaboración propia.

El método CerrarPuerto es muy simple, solo verifica el estado del puerto y si está abierto lo cierra.

Figura 29. **Código de método CerrarPuerto**

```
Sub CerrarPuerto()
    If PuertoCom.IsOpen Then PuertoCom.Close()
End Sub
```

Fuente: elaboración propia.

El método LeerSMS se encarga de leer los mensajes SMS que entran al módem GSM y los inserta en BandejaEntrada que es un atributo de la clase ClaseGSM, a la cual también pertenece. Utiliza un solo parámetro que es TipoSMS el cual indica que tipo de mensajes se desean leer del módem, los valores que puede tomar este parámetro son los definidos por TipoMensaje y descrito en la tabla IX del capítulo 2.

En la sección 2.4.1.2 se trató sobre los comandos AT necesarios para leer mensajes SMS de un módem GSM, el método LeerSMS hace uso de esta secuencia de comandos. En el inciso (1) de la figura de abajo se muestra abre el puerto de comunicación PuertoCom, posteriormente se procede a vaciar la bandeja de entrada.

La función *Thread.Sleep*(tiempo en ms) provoca un tiempo de espera necesario para que el módem termine de procesar el comando solicitado, se verá que el tiempo varía dependiendo del comando ya que hay algunos que se tardan más en responder.

En (2) inicia la secuencia de comandos AT que se envía al módem, el primer comando activa el módem, el segundo AT+CMGF=1 le indica al módem que trabaje en modo texto, el tercero AT+CPMS="SM" indica que leerá los SMS de la memoria SIM del módem, el último comando AT+CMGL=TipoSMS indica al módem que lea los mensajes del tipo indica por TipoSMS, estos mensajes son devueltos en *buffer* de datos, que se almacena en la variable rxBuff y se extrae por medio del comando PuertoCom.ReadExisting.

En (3) inicia la extracción de mensajes, el ciclo *while* verifica que existe algún mensaje que extraer del *buffer*, y posteriormente comienza la extracción del mensaje buscando en el *buffer* el texto +CMGL: ya que a partir de ahí inician los mensajes SMS.

Figura 30. **Código de método LeerSMS parte 1/3**

```
Function LeerSMS(ByVal TipoSMS As String) As String
    Dim rxBuff As String 'aqui almacena todos SMS como una sola cadena
    Dim Indice As Integer = 0 'indica en que posicion de rxBuff esta buscando
    Dim Comienzo As Integer 'indica donde inicia un SMS
    Dim Final As Integer 'indica donde finaliza un SMS
    Dim Mensaje As String 'alberga un SMS sin procesar
    Dim componentes() As String 'los diferetentes campos del SMS
    Dim Tel As String 'el tel del SMS origen
    Dim fecha As String 'fecha del SMS
    Dim tmpfecha() As String 'se utiliza para tranformar formato fecha
    Dim hora As String 'hora del SMS
    Dim Msg As String 'texto o contenido del SMS
    Dim IndiceMensaje As String 'el indice del mensaje leido
    Dim IIndice As Integer = 0 'para extraer IndiceMensaje
    Dim FIndice As Integer = 0 'para extraer IndiceMensaje
    Dim FinalIndice As Integer 'para extraer Telefono
    Dim ComienzoMsgIdx As Integer 'donde inicia texto SMS
    Dim FinalMsgIdx As Integer 'donde finaliza texto SMS
    Try
1       If Not PuertoCom.IsOpen Then PuertoCom.Open()
        BandejaEntrada.Clear()
        Thread.Sleep(50)
2       PuertoCom.WriteLine("AT" & vbCr)
        Thread.Sleep(300)
        'indica al modem que trabaje en modo texto
        PuertoCom.WriteLine("AT+CMGF=1" & vbCr)
        Thread.Sleep(500)
        'Selecciona lugar de almacenamiento SMS
        PuertoCom.WriteLine("AT+CPMS=""SM""" & vbCr) 'ME=mem modem, MT=ambos
        Thread.Sleep(300)
        PuertoCom.WriteLine("AT+CMGL=""" & TipoSMS & """" & vbCr)
        Thread.Sleep(1000)
        rxBuff = (PuertoCom.ReadExisting)
3       If rxBuff.IndexOf("+CMGL:", Indice) > 0 Then
            While rxBuff.IndexOf("+CMGL:", Indice) > 0 'CMGR:
                Comienzo = rxBuff.IndexOf("+CMGL:", Indice)
                Final = rxBuff.IndexOf("+CMGL:", Comienzo + 1) 'rxBuff.Length
```

Fuente: elaboración propia.

En (4) se extrae los datos del SMS y el texto que se almacena en la variable Mensaje, es similar a esto:

+CMGL: 1,"REC UNREAD","+50259383474",,"01/04/14,06:38:33-24"

1

La primera línea en Mensaje tiene la siguiente secuencia: +CMGL: <MR>, <TipoMensaje>,<Teléfono o OA>,,<Fecha o SCTS>. La segunda línea contiene el texto del mensaje SMS o campo UD, la simbología de MR, OA, SCTS y UD fue explicado en la sección 1.6. De ahí se divide el mensaje en sus componentes y se extrae de ahí el número de teléfono, fecha y hora del mensaje, los cuales se cambia el formato ya el orden es inverso, y el texto del mensaje. En (5) extrae el MR o índice del mensaje asignado por módem.

Figura 31. **Código de método LeerSMS parte 2/3**

```
'Si no hay +CMGL el Final es igual al Tamaño
If Final <= 0 Then
    Final = rxBuff.Length
End If
'Buscamos el Mensaje Completo
Mensaje = rxBuff.Substring(Comienzo, Final - Comienzo)
componentes = Mensaje.Split(","c)
'Buscamos no cel.
Tel = componentes(2).Replace("""", "")
fecha = componentes(4).Replace("""", "")
tmpfecha = fecha.Split("/")
fecha = tmpfecha(2) & "/" & tmpfecha(1) & "/" & tmpfecha(0)
IndiceMensaje = componentes(0)
IIndice = 0
FIndice = 0
IIndice = IndiceMensaje.IndexOf("+CMGL:") & "+CMGL:".Length
FIndice = IndiceMensaje.Length - IIndice
IndiceMensaje = IndiceMensaje.Substring(IIndice, FIndice)
IndiceMensaje = IndiceMensaje.Trim()
FinalIndice = Tel.IndexOf("" & vbCrLf & "")
If FinalIndice <= 0 Then
    FinalIndice = Tel.Length
End If
Tel = Tel.Substring(0, FinalIndice)
Tel = Tel.Replace("+502", "") 'quita 502 del telefono
' Buscamos MS
'Comienza despues de una rxBuff Nueva
ComienzoMsgIdx = Mensaje.IndexOf("" & vbCrLf & "")
If ComienzoMsgIdx < 0 Then
    ComienzoMsgIdx = 0
End If
' Finaliza antes de una Nueva rxBuff
hora = componentes(5).Substring(0, componentes(5).IndexOf("" & vbCrLf & "", 0))
hora = hora.Substring(0, 8)
FinalMsgIdx = Mensaje.IndexOf("" & vbCrLf & "", ComienzoMsgIdx + 1)
```

Fuente: elaboración propia.

Ya completo el mensaje (6) se crea una instancia de la clase Sms, en cual se asigna todos los valores del mensaje SMS, para posteriormente (7) insertarlo en la bandeja de entrada o sea el atributo BandejaEntrada. Ahora que ya se ha leído el mensaje SMS del módem, se procede a eliminarlo utilizando el método EliminarSMS, indicando el índice del mensaje que se desea eliminar.

Esta serie de pasos se repite una y otra vez, por medio del ciclo *while* que inicia en el paso (3) y finaliza en el (8), hasta que no existan mensajes que leer del módem. Por último se cierra el puerto COM.

Figura 32. **Código de método LeerSMS parte 3/3**

```
            If FinalMsgIdx <= 0 Then
                FinalMsgIdx = Mensaje.Length
            End If
            'Almacenamos el Mensaje
            Msg = Mensaje.Substring(ComienzoMsgIdx, FinalMsgIdx - ComienzoMsgIdx)
            Msg = Msg.Trim
6           Dim _Mensaje As New Sms()
            _Mensaje.IDSms = IndiceMensaje.Trim()
            _Mensaje.Telefono = Tel.Trim()
            _Mensaje.Mensaje = Msg.Trim()
            _Mensaje.Fecha = fecha.Trim()
            _Mensaje.Hora = hora.Trim
            'Almacenamos en la lista generica el mensaje
7           BandejaEntrada.Add(_Mensaje)
            EliminarSMS(IndiceMensaje)
            Indice = Comienzo + 1
8       End While
        End If
        If PuertoCom.IsOpen Then PuertoCom.Close()
        Return ""
    Catch err As Exception
        Return err.Message
    End Try
End Function
```

Fuente: elaboración propia.

El método que realiza la función de enviar mensajes SMS través del módem GSM se llama EnviaSMS y recibe dos parámetros para realizar esta tarea, el primer parámetro es el número de teléfono destino y el segundo el texto del mensaje a enviar.

Figura 33. **Código de método EnviaSMS**

```
Public Function EnviaSMS(ByVal NroTelefono As String, ByVal Mensaje As String) As Boolean
    Dim tels() As String 'telefonos a enviar SMS
    Dim i As Integer 'contador
    Dim rxBuff As String 'cadena buffer
    Dim res As Boolean 'respuesta
    Try
1       If Not PuertoCom.IsOpen Then PuertoCom.Open()
        res = True
        'Inicializamos el Modem
        PuertoCom.WriteLine("AT" & vbCr & "")
        Thread.Sleep(50)
        rxBuff = (PuertoCom.ReadExisting)
        'Indicam al modem que trabaje en modo texto
        PuertoCom.WriteLine("AT+CMGF=1" & vbCr & "")
        Thread.Sleep(50)
        Mensaje = Mensaje.Trim()
2       If Mensaje.Length > 160 Then 'Valida tamaño de mensaje <= 160 caracteres
            Mensaje = Mid(Mensaje, 1, 160)
        End If
        tels = NroTelefono.Split(",")
3       For i = 0 To tels.Length - 1
            'Indicamos el Nro de Telefono al que se enviara el mensaje
            PuertoCom.WriteLine("AT+CMGS=""" & tels(i).Trim() & """" & vbCr & "")
            Thread.Sleep(50)
            'enviamos el mensaje de texto
            PuertoCom.WriteLine(Mensaje & vbCrLf & Chr(26))
            Thread.Sleep(4800)
            rxBuff = (PuertoCom.ReadExisting)
            If rxBuff.EndsWith(vbCrLf & "OK" & vbCrLf) Then
            ElseIf rxBuff.Contains("ERROR") Then
                res = False
            End If
4       Next
        If PuertoCom.IsOpen Then PuertoCom.Close()
        Return res
    Catch err As Exception
        Return False
    End Try
End Function
```

Fuente: elaboración propia.

Devuelve como resultado un valor de verdadero o falso indicando si se envío el mensaje o no. En (1) inicia abriendo el puerto de comunicación con el módem, después establece la secuencia de comandos AT, la misma vista en la sección 2.4.1.1, donde el primer comando activa el módem, el segundo AT+CMGF=1 le indica al módem que trabaje en modo texto.

En (2) valida que la longitud del texto del mensaje no sobrepase el tamaño de 160 caracteres, ya que este es el máximo que se puede utilizar usando la codificación de 7 bits que es la se usa por defecto, la longitud del texto puede variar ya que depende del tipo de codificación que se use ver sección 1.5.1 y 1.6.6.1.

La siguiente instrucción extrae los números de teléfonos que se encuentran en la variable NroTelefono, luego en (3) se recorre cada uno de los números contenidos ahora en la variable tels, y envía el mensaje utilizando el comando AT+CMGS=”NoTel” seguido por el texto del mensaje donde la instrucción finaliza con la combinación de teclas <Ctrl+z>, la instrucción *Thread.Sleep*(4800), es un tiempo de espera como se indico anteriormente, y el tiempo puede variar dependiendo de las capacidades del módem, se ha especificado un número elevado para asegurarse de dar el tiempo suficiente de terminar la operación, pero este valor puede ser mucho más pequeño, de aproximadamente 1 segundo.

La siguiente instrucción busca al final del *buffer* del módem la combinación vbCrLf & "OK" & vbCrLf, la cual indica que el mensaje se envío exitosamente, si por el contrario en el *buffer* de datos se encuentra el texto ERROR, indica que ocurrió un error y el mensaje no se pudo enviar y asigna la variable de resultado res con el valor de falso. En (4) se tiene el fin del ciclo que inicio en (3). Por último se cierra el puerto de COM y se devuelve el valor de la variable res.

El último método de la clase ClaseGSM es el de EliminarSMS, el cual como su nombre indica elimina un mensaje de texto de la memoria, enviando como parámetro el número de índice del mensaje que se desea eliminar, devuelve como resultado el valor de verdadero si lo elimina o falso si no lo hace.

Figura 34. **Código de método EliminaSMS**

```
Public Function EliminarSMS(ByVal IndiceMensaje As Integer) As Boolean
    Dim rxBuff 'cadena buffer
    Try
        If Not PuertoCom.IsOpen Then PuertoCom.Open()
        'Inicializamos el Modem
        PuertoCom.WriteLine("AT+CMGD=" & IndiceMensaje & vbCr) '*
        Thread.Sleep(100)
        rxBuff = (PuertoCom.ReadExisting)
        Return True
    Catch err As Exception
        Return False
    End Try
End Function
```

Fuente: elaboración propia.

Este método es simple, se resume en abrir el puerto de comunicación, ejecutar el comando AT+CMGD=IndiceMensaje que es el comando que le indica al módem de debe eliminar el mensaje que coincide con el índice.

4.3. Codificación de la clase eMail

Aquí se detalla y explica el código generado de cada uno de los elementos de la clase eMail, sus atributos y métodos definidos en la etapa de diseño. El espacio de nombre que necesita esta clase es:

- *System.Net.Mail*

Los atributos que utiliza son: smtpHost, smtpPuerto, smtpUsuario y smtpClave.

Figura 35. **Declaración de atributos de clase eMail**

```
Private smtpHost As String
Private smtpPuerto As String
Private smtpUsuario As String
Private smtpClave As String
```

Fuente: elaboración propia.

El primer método de la clase eMail se llama ConfiguraSmtp, su función consiste en asignar los valores necesarios para la conexión SMTP. Recibe como parámetros las variables host, puerto, usuario y clave, y los asigna a los atributo de la clase definidos previamente.

Figura 36. **Código de método ConfiguraSmtp**

```
Public Sub ConfiguraSmtp(ByVal host As String, ByVal puerto As String, _
                         ByVal usuario As String, ByVal clave As String)
    smtpHost = host
    smtpPuerto = puerto
    smtpUsuario = usuario
    smtpClave = clave
End Sub
```

Fuente: elaboración propia.

El segundo y último método de la clase eMail se llama EnviaMensageEMail, su función es la de generar y enviar un mensaje por medio de un correo electrónico.

Como parámetros recibe las variables: eMailOrigen, cuenta de correo de quien envía el mensaje; Recipiente, a quien envía o destinatario; bcc, destinatario con copia oculta; cc, destinatario con copia; Titulo, es el asunto del mensaje; CuerpoMensaje, es el contenido del mensaje; EsHtml, indica si el formato del correo a enviar es formato texto o html; y NombreAVisualizar, es el nombre de la cuenta asociada a eMailOrigen o nombre de quien envía.

En la figura 37 se muestra el código fuente, en (1) se declara sMensajeEMail una instancia de *MailMessage*, y que se utiliza para armar el correo electrónico.

En (2) se declara mClienteSmtp que es una instancia de SmptClient, donde se especifica con stmpHost y stmpPuerto, el nombre o dirección IP del servidor y puerto de comunicación con el servidor de correo utilizando el protocolo SMTP, su utilidad es la de enviar el mensaje contenido por sMensajeEMail. La siguiente instrucción asigna las credenciales necesarias para el envío, asignando el usuario y clave con los parámetros smtpUsuario y stmpClave.

En (3) remueve los espacios en blanco al recipiente, luego asigna el correo origen, los destinatarios, el asunto del mensaje, el contenido del mensaje. En (4) se indica el formato del mensaje, la prioridad del mensaje, se habilita el uso SSL y por último (5) se envía el mensaje.

Figura 37. **Código de método EnviaMensajeEMail**

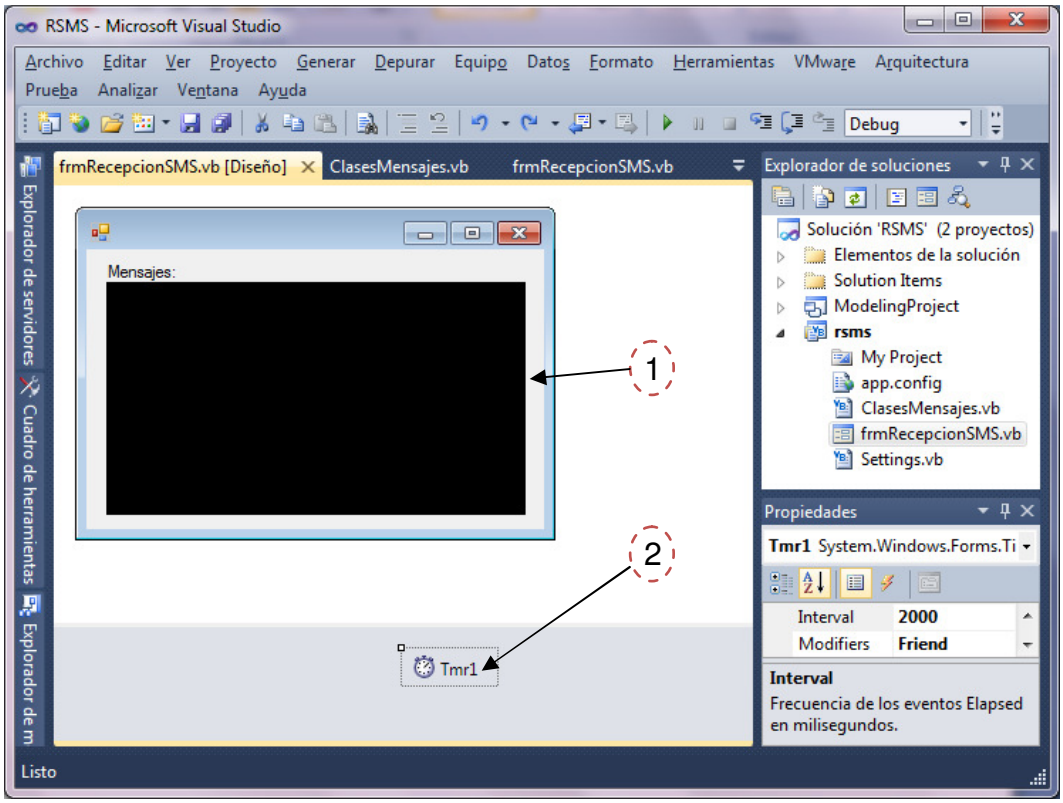
```
Public Function EnviaMensageEMail(ByVal eMailOrigen As String, ByVal Recipiente As String, _
ByVal bcc As String, ByVal cc As String, ByVal Titulo As String, ByVal CuerpoMensaje As String, _
Optional ByVal EsHmtl As Boolean = True, Optional ByVal NombreAVisualizar As String = "") As Boolean
    ' Crea una nueva instancia de MailMessage
    Dim mMensajeEMail As New MailMessage()
    'Aqui almacenara el listado de correos destino
    Dim mRecipientes() As String
    Dim tmp As String
    Dim i As Integer
    ' Crea instancia de SmtpClient
    Dim mClienteSmtp As New SmtpClient(smtpHost, smtpPuerto)
    mClienteSmtp.Credentials = New System.Net.NetworkCredential(smtpUsuario, smtpClave)
    Try
        tmp = Recipiente.Replace(" ", "") 'Remueve espacios en blanco
        mRecipientes = tmp.Split(",")
        ' Asigna la direccion de quien envia el mensaje
        mMensajeEMail.From = New MailAddress(eMailOrigen, NombreAVisualizar)
        ' Asigna la direccion de quien(es) recive el mensaje
        For i = 0 To mRecipientes.Length - 1
            mMensajeEMail.To.Add(New MailAddress(mRecipientes(i)))
        Next
        ' Verifica si BCC esta vacio/notienenada
        If Not bcc Is Nothing And bcc <> String.Empty Then
            'Asigna la direccion de BCC
            mMensajeEMail.Bcc.Add(New MailAddress(bcc))
        End If
        ' Verifica si CC  esta vacio/notienenada
        If Not cc Is Nothing And cc <> String.Empty Then
            'Asigna la direccion de CC
            mMensajeEMail.CC.Add(New MailAddress(cc))
        End If
        ' Asigna el asunto del mensaje
        mMensajeEMail.Subject = Titulo
        ' Asigna el cuerpo del mensaje
        mMensajeEMail.Body = CuerpoMensaje
        ' Indica el formato del mensjae, que es HTML, de lo contrario seria tipo Texto.
        mMensajeEMail.IsBodyHtml = EsHmtl 'False
        ' Asigna la prioridad del mensaje, que es Normal, puede ser High y low tambien
        mMensajeEMail.Priority = MailPriority.Normal
        'Activa el uso Secury Socket Layer, en la comunicacion con el servidor de correo.
        mClienteSmtp.EnableSsl = True
        ' Envia el mensaje
        mClienteSmtp.Send(mMensajeEMail)
        Return True
    Catch err As Exception
        Return False
    End Try
End Function
```

Fuente: elaboración propia.

4.4. Codificación de la clase frmRecepcionSMS

Esta clase es de tipo formulario y por medio de este el administrador del sistema podrá visualizar los mensajes SMS de aviso que el sistema procesa. Tiene dos elementos definidos desde el diseñador de formularios de Visual Studio, que son de interés, estos son: un elemento tipo *Label* de nombre LblMsg, el cual es usado por el método VisualizaMsgEstado, en que se muestran los mensajes que se van procesando y el resultado del mismo; el otro elemento es un *Timer* de nombre Tmr1, y ejecuta de forma automática el método LeeSMS, la figura de abajo muestra los elementos.

Figura 38. **Entorno de desarrollo en Visual Studio 2010**



Fuente: elaboración propia.

Ahora se definen los espacios de nombres de esta clase que son:

- *System*
- *System.Configuration*
- *System.Data.Odbc*

Los atributos que utiliza son: cDB, COM, csms, emailFail, mailFrom, EnviarSMS, Intervalo, Mail, MsgHtml, NombreAVisualizar, SMSAcuseMsg1, SMSAcuseMsg2, SMSAcuseRecepecion, SinSignos y ConSignos.

Figura 39. **Declaración de atributos de clase frmRecepcionSMS**

```
'Caracteres a reemplazar en SMS, son caracteres no validos.
Private Const ConSignos As String = "áàäéèëíìïóòöúùuñÁÀÄÉÈËÍÌÏÓÒÖÚÙÜçÇ"
Private Const SinSignos As String = "aaaeeeiiiooouuunAAAEEEIIIOOOUUUcC"
'Instancia de Clase BaseDatos, conexion a la base de datos
Dim cDB As New BaseDatos
'Instancia de Clase GMS para el envio y recepcion de SMS
Dim csms As New ClaseGSM
'Instancia de calse eMail a utilizar para envio de correo electronico
Dim Mail As New eMail
'Puerto Com a utilizar
Public Shared COM As String
'Direccion de Correo electronico, a utilizar como origen (quien envia los mensajes)
Dim emailFrom As String
'Direccion de Correo electronico, a utilizar cuando falla el proceso de un SMS
Dim emailFail As String
'Nombre a visualizar de la cuenta email From (nombre del usuario emailFrom)
Dim NombreAVisualizar As String
'Indica si el Mensaje a enviar por Email sera Html o no
Dim MsgHtml As Boolean
'Indica si se envia SMS despues de envia correo electronico, en la notificaciones
Dim EnviarSMS As Boolean
'Intervalo en segundos, que se revisar por nuevos mensajes SMS
Dim Intervalo As Integer
'Indica si se utliliza mensajes SMS para acuse de recibido (envia SMS al Cel. origen)
Dim SMSAcuseRecepecion As Boolean
'Texto de mensaje a utlizar en los SMS de acuse de recibido
Dim SMSAcuseMsg1, SMSAcuseMsg2 As String
'nlineas indica el numero de linea que puede contener LblMsg, clineas indica el numero de linea
Dim nlineas, clineas As Integer
```

Fuente: elaboración propia.

El método llamado Inicializacion es el responsable de cargar los parámetros de configuración del sistema, los cuales se almacenan en el archivo app.config. También inicializa las instancias de las clases BaseDatos, ClaseGSM y eMail representadas por cDB, csms y Mail respectivamente.

Figura 40. **Código de método Inicializacion**

```
Sub Inicializacion()
   With ConfigurationManager.AppSettings 'Lectura de arch. app.config
      COM = .Get("COM").ToString
      emailFrom = .Get("EMAILFROM").ToString
      NombreAVisualizar = .Get("DISPLAYNAME").ToString
      EnviarSMS = .Get("ENVIARSMS").ToString
      emailFail = .Get("EMAILFAIL").ToString
      MsgHtml = .Get("MSGHTML").ToString
      Intervalo = .Get("INTERVALO").ToString
      SMSAcuseRecepecion = .Get("SMS_ACUSE").ToString
      If SMSAcuseRecepecion Then
         SMSAcuseMsg1 = .Get("SMS_ACUSE_MSG1").ToString
         SMSAcuseMsg2 = .Get("SMS_ACUSE_MSG2").ToString
      End If
      cDB.AbreConexion(.Get("ODBC").ToString, .Get("USUARIO").ToString, .Get("PASSWORD").ToString)
      csms.ConfigurarPuerto(COM)
      Mail.ConfiguraSmtp(.Get("stmphost"), .Get("stmpport"), .Get("stmpuser"), .Get("stmppw"))
   End With
   nlineas = CInt(LblMsg.Height / 13)
   clineas = 0
   If Intervalo > 1 Then
      Tmr1.Interval = Intervalo * 1000
   End If
   Tmr1.Enabled = True
   VisualizaMsgEstado("Inicializando... Ok")
End Sub
```

Fuente: elaboración propia.

El método EnviarMensaje, es simple, solo realiza una llamada al método EnviaSMS de la clase ClaseGMS a través de la instancia csms, recibe como parámetros el número de teléfono y el texto del mensaje SMS a enviar.

Figura 41. **Código de método EnviarMensaje**

```
Public Function EnviarMensaje(ByVal NroTelefono As String, ByVal Mensaje As String) As Boolean
   Try
      Return csms.EnviaSMS(NroTelefono, Mensaje)
   Catch err As Exception
      Return False
   End Try
End Function
```

Fuente: elaboración propia.

El método EnviaSmsALista se encarga de enviar un mensaje SMS de aleta a los responsables, lo hace extrayendo los datos de la base de datos, de la tabla sms_contactos, el mensaje es enviado a todos los responsables de esa región geográfica, utilizando el método EnviaMensaje.

Figura 42. **Código de método EnviaSmsALista**

```
Function EnviaSmsALista(ByVal zonageografica As String, ByVal Mensaje As String) As Boolean
   Dim Sql As String
   Dim resultadoSQL As OdbcDataReader
   Dim res As String
   Dim repuesta As Boolean
   Try
      repuesta = False
      If EnviarSMS Then
         Sql = "select * from sms_contactos where celular!='' and (zonageografica = '" & zonageografica _
             & "' or zonageografica='" & zonageografica.Substring(0, 2) & "' or zonageografica='');"
         resultadoSQL = cDB.ConsultaSql(Sql)
         While resultadoSQL.Read()
            res = resultadoSQL("celular")
            If EnviarMensaje(res, Mensaje) Then
               repuesta = True
            End If
         End While
         Return repuesta
      Else
         Return False
      End If
   Catch err As Exception
      Return False
   End Try
End Function
```

Fuente: elaboración propia.

El método EnviaAcuseRecibido genera y envía un mensaje SMS de respuesta, dirigido al agente monitor que reporto alguna emergencia, el contenido del mensaje varía dependiendo si es de éxito, tipo=1, o de fracaso tipo=2.

Figura 43. **Código de método EnviarAcuseRecibido**

```
Sub EnviaAcuseRecibido(ByVal Tel As String, ByVal Tipo As Integer)
   Dim Mensaje As String
   If SMSAcuseRecepecion Then
      If Tipo = 1 Then 'SMS recibido y procesado correctamente
         Mensaje = IIf(SMSAcuseMsg1 = "", "Su mensaje sera atendido proximamente.", SMSAcuseMsg1)
      Else 'tipo=2 SMS recibido pero el contenido es incorrecto (no fue posible procesarlo)
         Mensaje = IIf(SMSAcuseMsg2 = "", "No se pudo procesar SMS, contenido incorrecto.", SMSAcuseMsg2)
      End If
      VisualizaMsgEstado("Generando acuse Tipo " & Tipo.ToString & ".", True)
      EnviarMensaje(Tel, Mensaje)
   End If
End Sub
```

Fuente: elaboración propia.

El método EnviarCorreo envía un correo electrónico al destino y con los datos indicados por los parámetros, y lo realiza a través de la instancia Mail de la clase eMail.

Figura 44. **Código de método EnviaCorreo**

```
Sub EnviaCorreo(ByVal emailFrom As String, ByVal recipiente As String, ByVal bcc As String, _
                ByVal cc As String, ByVal subject As String, ByVal cuerpo As String, _
                ByVal html As Boolean, ByVal nombrever As String)
   If Mail.EnviaMensageEMail(emailFrom, recipiente, bcc, cc, subject, cuerpo, html, nombrever) = False Then
      Mail.EnviaMensageEMail(emailFrom, recipiente, bcc, cc, subject, cuerpo, html, nombrever)
   End If
End Sub
```

Fuente: elaboración propia.

El método EnviaCorreoALista, en cuanto a lógica de funcionamiento es similar a EnviaSmsALista, ya que envía un correo electrónico como mensaje de alerta, a todos los responsables de una región geográfica.

Figura 45. **Código de método EnviaCorreoALista**

```
Function EnviaCorreoALista(ByVal zonageografica As String, ByVal Asunto As String, _
                           ByVal Mensaje As String) As Boolean
   Dim Sql As String
   Dim resultadoSQL As OdbcDataReader
   Dim res As String
   Try
      Sql = "select * from sms_contactos where correo!='' and (zonageografica = '" & zonageografica _
          & "' or zonageografica='" & zonageografica.Substring(0, 2) & "' or zonageografica='');"
      resultadoSQL = cDB.ConsultaSql(Sql)
      While (resultadoSQL.Read())
         res = resultadoSQL("correo")
         EnviaCorreo(emailFrom, res, "", "", Asunto, Mensaje, MsgHtml, NombreAVisualizar)
      End While
      Return True
   Catch err As Exception
      VisualizaMsgEstado("Error, no se pudo envia el correo.", True)
      Return False
   End Try
End Function
```

Fuente: elaboración propia.

El método DarValor, construye una sentencia SQL de consulta, con el objetivo de retraer de la base de datos, el valor de la columna especifica. Como parámetros recibe el nombre de la tabla, el campo a extraer Dar, y los parámetros restantes sirven para restringir de la consulta.

Figura 46. **Código de método DarValor**

```
Function DarValor(ByVal Tabla As String, ByVal Dar As String, ByVal Campo1 As String, _
                  Optional ByVal Campo2 As String = "") As String
   Dim sql, where As String
   sql = "SELECT " & Dar & " FROM " & Tabla
   Select Case Tabla
      Case "d_comunidad"
         where = "comunidad_id='" & Campo1 & "' and muni_id='" & Campo2 & "'"
      Case "d_municipalidad"
         where = "muni_id='" & Campo1 & "'"
      Case "d_depto"
         where = "depto_id='" & Campo1 & "'"
      Case "sms_tipocaso"
         where = "cast(id_caso as SIGNED)=cast('" & Campo1 & "' as SIGNED)"
      Case "d_agente"
         where = "agente_id='" & Campo1 & "'"
      Case Else
         where = "0=0"
   End Select
   sql = sql & " WHERE " & where
   Return cDB.EjecutaSql(sql, 0)
End Function
```

Fuente: elaboración propia.

El método RemueveEspacios, elimina los espacios dobles en blanco que estén consecutivos, por ejemplo para el texto: H o l a; devolvería: H o l a.

Figura 47. **Código de método RemueveEspacios**

```
Private Function RemueveEspacios(ByVal texto As String) As String
   While texto.Contains("  ")
      texto = texto.Replace("  ", " ")
   End While
   Return texto
End Function
```

Fuente: elaboración propia.

El método RemoverSignosAcentos, utiliza las cadenas de caracteres ConSignos (caracteres no válidos) y SinSignos (válidos), para remover de un texto los caracteres no válidos en un SMS.

Figura 48. **Código de método RemoverSignosAcentos**

```
Public Shared Function RemoverSignosAcentos(ByVal texto As String) As String
   Dim textoSinAcentos = String.Empty
   Dim var As Char

   For Each var In texto
      Dim indexConAcento = ConSignos.IndexOf(var)
      If indexConAcento > -1 Then
         textoSinAcentos = textoSinAcentos & (SinSignos.Substring(indexConAcento, 1))
      Else
         textoSinAcentos = textoSinAcentos & (var)
      End If
   Next
   Return textoSinAcentos
End Function
```

Fuente: elaboración propia.

El método CadenaSinCeros, remueve todos los ceros (0) a la izquierda contenidos en un texto, se utiliza cuando el formato de los parámetros contenidos en el SMS tiene ceros como carácter de relleno.

Figura 49. **Código de método CadenaSinCeros**

```
Public Function CadenaSinCeros(ByVal sEntrada As String) As String
   Dim j As Integer, sIntermedio As String
   sIntermedio = ""
   For j = 1 To Len(sEntrada)
      If Mid(sEntrada, j, 1) <> "0" Then
         sIntermedio = Mid(sEntrada, j)
         Exit For
      End If
   Next
   CadenaSinCeros = sIntermedio
End Function
```

Fuente: elaboración propia.

El método ExtraeDatos, tiene como parámetro el texto del mensaje entrante y devuelve los mismos datos, pero sin espacio en blanco y segmentados en un arreglo de datos.

Figura 50. **Código de método ExtraeDatos**

```
Function MsgExtraeDatos(ByVal Msg As String) As String()
   Dim MsgLimpio As String
   Dim datosmsg() As String
   Dim i As Integer
   MsgLimpio = Msg
   If Msg.IndexOf(">") > 0 Then
      MsgLimpio = Msg.Substring(Msg.IndexOf(">") + 1).Trim
   End If
   MsgLimpio = RemueveEspacios(MsgLimpio)
   datosmsg = MsgLimpio.Split(" ")
   For i = 0 To datosmsg.Length - 1
      datosmsg(i) = datosmsg(i).Trim()
   Next
   Return datosmsg
End Function
```

Fuente: elaboración propia.

El método VerificaEstructura verifica el formato de mensaje, con el objetivo de constatar que están los campos necesarios para procesar el aviso, recibe como parámetros un vector, donde contiene los campos a validar, y una serie de variables por referencia, donde devuelve los código extraídos del arreglo.

Devuelve verdadero o falso indicando con falso que al estructura esta completa y con verdadero que está incompleta.

Figura 51. **Código de método MsgVerificaEstructura**

```
Function MsgVerificaEstructura(ByVal datosmsg() As String, ByVal tel As String, ByRef muni_id As String, _
         ByRef comunidad_id As String, ByRef caso_id As String, ByRef agente_id As String, _
         ByRef nombreagente As String) As Boolean
   If datosmsg.Length = 1 Then
      agente_id = cDB.EjecutaSql("SELECT agente_id FROM d_agente WHERE celular='" & tel & "'", 0)
      'solo contiene el codigo de la emergencia, los demas datos se retraen del codigo agente
      If agente_id <> "" Then
         nombreagente = cDB.EjecutaSql("SELECT nombre FROM d_agente WHERE agente_id='" & agente_id & "'", 0)
         comunidad_id=cDB.EjecutaSql("SELECT comunidad_id FROM d_agente WHERE agente_id='"& agente_id &"'",0)
         muni_id = cDB.EjecutaSql("SELECT muni_id FROM d_agente WHERE agente_id='" & agente_id & "'", 0)
         caso_id = datosmsg(0)
         If nombreagente = "" Then 'valida que codigo de agente exista
            Return True
         Else
            Return False
         End If
      Else 'Codigo formato: 4DigMuni+3DigComunidad+3DigCodAgente+3DigCodEmergencia ej:2006005002003
         muni_id = IIf(datosmsg(0).Length >= 4, datosmsg(0).Substring(0, 4), "")
         comunidad_id = IIf(datosmsg(0).Length >= 7, CadenaSinCeros(datosmsg(0).Substring(4, 3)), "")
         agente_id = IIf(datosmsg(0).Length >= 10, CadenaSinCeros(datosmsg(0).Substring(7, 3)), "")
         caso_id = IIf(datosmsg(0).Length > 10, CadenaSinCeros(datosmsg(0).Substring(10)), "")
         Return False
      End If
   ElseIf datosmsg.Length >= 4 Then 'formato con 4 campos
      muni_id = datosmsg(0)
      comunidad_id = datosmsg(1)
      agente_id = datosmsg(2)
      caso_id = datosmsg(3)
      Return False
   Else 'error, no tiene datos
      Return True
   End If
End Function
```

Fuente: elaboración propia.

El método MsgReemplazaContenido, realiza una sustitución de etiquitas denotadas por <ValorVariable1> dentro del parámetro contenido y lo reemplaza por un valor, así en la primera sentencia que se mira en la figura de abajo, se está reemplazando el texto <caso> por el valor contenido en la variable tipocaso.

Figura 52. **Código de método MsgReemplazaContenido**

```
Function MsgReemplazaContenido(ByVal contenido As String,ByVal tipocaso As String,ByVal comunidad As String, _
ByVal municipio As String, ByVal depto As String, ByVal nombreagente As String, ByVal Tel As String, _
ByVal CodigoRegAgente As String, ByVal latitud As Double, ByVal longitud As Double) As String
   contenido = contenido.Replace("<caso>", tipocaso.Trim)
   contenido = contenido.Replace("<comunidad>", comunidad.Trim)
   contenido = contenido.Replace("<municipio>", municipio.Trim)
   contenido = contenido.Replace("<depto>", depto.Trim)
   contenido = contenido.Replace("<firma>", nombreagente.Trim)
   contenido = contenido.Replace("<celular>", Tel.Trim)
   contenido = contenido.Replace("<codigo>", CodigoRegAgente)
   contenido = contenido.Replace("<lat>", latitud)
   contenido = contenido.Replace("<long>", longitud)
   Return contenido
End Function
```

Fuente: elaboración propia.

El método de mayor importancia de esta clase es ProcesaMensaje, realiza las funciones de procesar el mensaje de alerta que ser recibe, lo interpreta y luego genera mensajes (por correo y SMS), que son enviados a las entidades responsables de atender las emergencias, para realizar todo este proceso se vale de otros métodos, tal como se puede apreciar en la sección 3.3.4. Como parámetros recibe los datos del mensaje, el número de teléfono, fecha, hora y el texto del mensaje.

En la siguiente imagen se puede ver, que primero se encuentra la declaración de variables a utilizar, en la que resalta datosmsg() que es un vector, en el cual se almacenará los datos del mensaje (código del departamento, municipio, comunidad, agente y tipo emergencia) de manera desglosada. En (1) se utiliza el método MsgExtraeDatos, el cual se encarga de dividir los datos del mensaje, luego MsgVerificaEstructura valida que la estructura del mensaje, tenga todos los datos en el formato correcto.

Figura 53. **Código de método ProcesaMensaje parte 1**

```
Function ProcesaMensaje(ByVal Tel As String, ByVal fechamsg As String, ByVal horamsg As String, _
                        ByVal Msg As String) As Boolean
   Dim fecha As Date
   Dim muni_id, comunidad_id, agente_id, caso_id, coddepto, codmuni As String
   Dim municipio, comunidad, depto, tipocaso, agente As String
   Dim tm_asunto, tm_contenido1, tm_contenido2, mensajefalla, Sql As String
   Dim msgincompleto, Resultado As Boolean
   Dim ok1, ok2 As Integer
   Dim datosmsg() As String
   Dim latitud, longitud As Double
   Resultado = False
   muni_id = ""
   comunidad_id = ""
   caso_id = ""
   agente_id = ""
   agente = ""
   ok1 = 0
   ok2 = 0
   fecha = fechamsg & " " & horamsg
   Try
1     datosmsg = MsgExtraeDatos(Msg)
      msgincompleto = MsgVerificaEstructura(datosmsg,Tel,muni_id,comunidad_id,caso_id,agente_id,agente)
```

Fuente: elaboración propia.

Si la validación anterior indica que el mensaje esta completo (msgincompleto=false) (2) se procede a retraer los nombres del departamento, municipio, agente, tipo emergencia (tipocaso), (3) ahora se valida que estos existan en la base de datos, si existen sigue al paso (4) donde inicia la construcción de los mensajes de alerta, de lo contrario se va al paso (7) donde registra el mensaje de aviso como fallido y envía un SMS a quien generó el aviso, indicando que el mensaje no se proceso.

En (4), retrae la plantilla o machote de texto, para generar los mensajes de alerta, también retrae otros datos como las coordenadas geográficas de la comunidad, que posteriormente utiliza para armar el mensaje usando el método MsgReemplazoContenido. (5) Ahora se envían los mensajes de alerta por correo y SMS, a los responsables, usando los métodos EnviaCorreoAlista y EnviaSmsALista.

Luego en (6) se registra el mensaje de aviso en la base de datos como mensaje procesado con éxito y envía un SMS a quien generó el aviso, indicando que la emergencia ya fue reportada.

Figura 54. **Código de método ProcesaMensaje parte 2**

```
If msgincompleto = False Then 'tiene los 4 campos necesarios y con el formato correcto
   If muni_id.Length >= 2 Then coddepto = muni_id.Substring(0, 2) Else coddepto = ""
   If muni_id.Length >= 4 Then codmuni = muni_id.Substring(0, 4) Else codmuni = ""
2  municipio = DarValor("d_municipalidad", "nombre", muni_id)
   comunidad = DarValor("d_comunidad", "nombre", comunidad_id, muni_id)
   depto = DarValor("d_depto", "nombre", coddepto)
   tipocaso = DarValor("sms_tipocaso", "nombre", caso_id)
   agente = DarValor("d_agente", "nombre", agente_id)
3  If municipio <> "" And comunidad <> "" And depto <> "" And agente <> "" And tipocaso <> "" Then
      'Construye/arma los mensajes a enviar
4     tm_asunto = DarValor("sms_tipocaso", "asunto", caso_id)
      tm_contenido1 = DarValor("sms_tipocaso", "contenidomsg", caso_id)
      tm_contenido2 = DarValor("sms_tipocaso", "contenidomsg2", caso_id)
      latitud = DarValor("d_comunidad", "lat", comunidad_id, muni_id)
      longitud = DarValor("d_comunidad", "`long`", comunidad_id, muni_id)
      'reemplaza contenido de tag de mensaje machote con los valeres correspondientes
      tm_contenido1 = MsgReemplazaContenido(tm_contenido1, tipocaso, comunidad, municipio, depto, _
         agente, Tel, codmuni & comunidad_id & "-" & agente_id, latitud, longitud)
      tm_contenido2 = MsgReemplazaContenido(tm_contenido2, tipocaso, comunidad, municipio, depto, _
         agente, Tel, codmuni & comunidad_id & "-" & agente_id, latitud, longitud)
      tm_contenido2 = RemoverSignosAcentos(tm_contenido2)
      VisualizaMsgEstado("Generando y enviando correo electrónico.", True)
      'envia un email a toda la lista de contactos responsables x zonaGeografica
5     ok1 = IIf(EnviaCorreoALista(codmuni, tm_asunto, tm_contenido1), 1, 0)
      If EnviarSMS Then VisualizaMsgEstado("Generando y enviando SMS.", True)
      'envia el mensaje2 por mensaje de texto SMS
      ok2 = IIf(EnviaSmsALista(codmuni, tm_contenido2), 1, 0)
      'Almacena el SMS en la BDD
6     Sql="insert into sms_mensajes(telefono,fecha,mensaje,municipio,comunidad,cod_agente,tipocaso,ok1,ok2)"
         & "values('" & Tel & "','" & Format(fecha, "yyyy-MM-dd HH:mm:ss") & "','" & Msg & "','" & muni_id _
         & "','" & comunidad_id & "','" & agente_id & "','" & caso_id & "'," & ok1 & "," & ok2 & ");"
      cDB.EjecutaSql2(Sql)
      EnviaAcuseRecibido(Tel, 1)
      If ok1 = 1 Then Resultado = True Else Resultado = False
   Else
      msgincompleto = True
      'Almacena el SMS en la BDD
7     Sql="insert into sms_mensajes(telefono,fecha,mensaje,municipio,comunidad,cod_agente,tipocaso,ok1,ok2)"
         & "values('" & Tel & "','" & Format(fecha, "yyyy-MM-dd HH:mm:ss") & "','" & Msg & "','" & muni_id
         & "','" & comunidad_id & "','" & agente_id & "','" & caso_id & "'," & ok1 & "," & ok2 & ");"
      cDB.EjecutaSql2(Sql)
   End If
End If
```

Fuente: elaboración propia.

Si la validación realizada en (1) indica que el mensaje está incompleto, entonces se va al paso (8) donde genera un correo al administrador del sistema de Aviso y Alerta, indicando los datos del mensaje erróneo. También se genera un SMS a quien generó el aviso, indicando que su mensaje es incorrecto.

Figura 55. **Código de método ProcesaMensaje parte 3**

```
    If msgincompleto Then 'genera email, avisando del fallo del mensaje
8       Resultado = True
        VisualizaMsgEstado("No se pudo procesar SMS, contenido incorrecto.", True)
        mensajefalla = "SMS, no se pudo procesar." & vbCr & vbCr & "Fecha: " & _
            fechamsg & " " & horamsg & vbCr & "No Celular: " & Tel & vbCr & "Contenido: " & Msg & vbCr
        EnviaCorreo(emailFrom, emailFail, "", "", "SMS incorrecto", mensajefalla, False, NombreAVisualizar)
        'indica que el contenido SMS Aviso es incorrecto
        EnviaAcuseRecibido(Tel, 2)
    End If
    Return Resultado
  Catch err As Exception
    MsgBox("Ocurrio un error(ProcesaMensaje): " & err.Message)
    Return False
  End Try
End Function
```

Fuente: elaboración propia.

El método frmRecepcionSMS_Load realiza la función de llamar al método Inicializacion al instante de cargar el formulario.

Figura 56. **Código de método frmRecepcionSMS_Load**

```
Private Sub frmRecepcionSMS_Load(ByVal sender As System.Object, ByVal e As System.EventArgs)
  Inicializacion()
End Sub
```

Fuente: elaboración propia.

El método frmRecepcionSMS_FormClosing es simple, cierra la conexión a la base de datos.

Figura 57. **Código de métodofrmRecepcionSMS_FormClosing**

```
Private Sub frmRecepcionSMS_FormClosing(ByVal sender As Object, ByVal e As System.Win
   cDB.CierraConexion()
End Sub
```

Fuente: elaboración propia.

LeeSMS es un método de importancia, utiliza la variable csms, instancia de la clase ClaseGMS, con la cual llama al método LeerSMS, y extrae los mensajes SMS contenidos en el módem GSM. Si existen mensajes que procesar, ejecuta el método ProcesaMensaje.

Figura 58. **Código de método LeeSMS**

```
Sub LeeSMS()
   Dim iMsg As Integer
   Dim Resp As String
   Try
      VisualizaMsgEstado("Buscando mensajes SMS...", True)
      Resp = csms.LeerSMS(ClaseGSM.TipoMensaje.Todos)
      If Resp = "" Then
         If csms.BandejaEntrada.Count > 0 Then
            VisualizaMsgEstado("Ok", True, False)
            For iMsg = 0 To csms.BandejaEntrada.Count - 1
               VisualizaMsgEstado("Procesando mensaje: " & csms.BandejaEntrada(iMsg).Mensaje, True)
               ProcesaMensaje(csms.BandejaEntrada(iMsg).Telefono, csms.BandejaEntrada(iMsg).Fecha, _
                              csms.BandejaEntrada(iMsg).Hora, csms.BandejaEntrada(iMsg).Mensaje)
            Next
         Else
            VisualizaMsgEstado("No hay mensajes.", True, False)
         End If
      Else
         VisualizaMsgEstado("Ocurrio un error al leer modem: " & Resp, True)
      End If
   Catch err As Exception
      MsgBox("Ocurrio un error(LeeSMS): " & err.Message)
   End Try
End Sub
```

Fuente: elaboración propia.

El método Tmr1_Tick, está vinculado al temporizador Tmr1, y hace que se ejecute periódicamente. Su única función es la llamar al método LeeSMS.

Figura 59. **Código de método Tmr1_Tick**

```
Private Sub Tmr1_Tick(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Tmr1.Tick
   Tmr1.Stop()
   LeeSMS()
   Tmr1.Start()
End Sub
```

Fuente: elaboración propia.

El método VisualizaMsgEstado, es muy utilizado, su función es la ir plasmando la actividad del sistema, los mensajes que se están procesando, y mostrarlos en la etiqueta de nombre LblMsg. Sirve para monitorear el funcionamiento del sistema.

Figura 60. **Código de método VisualizaMsgEstado**

```
Sub VisualizaMsgEstado(ByVal info As String, Optional ByVal concatenar As Boolean = False, _
                       Optional ByVal Retorno As Boolean = True)
   Dim posretorno As Integer
   If clineas >= nlineas Then
      If Retorno Then
         posretorno = LblMsg.Text.LastIndexOf(vbCr)
         LblMsg.Text = LblMsg.Text.Substring(0, IIf(posretorno = -1, 1, posretorno) - 1)
      End If
   Else
      If Retorno Then clineas = clineas + 1
   End If
   If concatenar Then
      If Retorno Then
         LblMsg.Text = info & vbCr & LblMsg.Text
      Else
         posretorno = LblMsg.Text.IndexOf(vbCr)
         LblMsg.Text = LblMsg.Text.Insert(IIf(posretorno = -1, 0, posretorno), " " & info)
      End If
   Else
      LblMsg.Text = info
      clineas = 1
   End If
   If LblMsg.Text.Length > 500 Then LblMsg.Text.Substring(1, 500)
   LblMsg.Update()
End Sub
```

Fuente: elaboración propia.

4.5. Codificación de la clase Sms

Esta clase no necesita la declaración de algún espacio de nombres, su única función es la almacenar los datos un mensaje SMS, sus atributos son los mismos los cuales compone a un mensaje SMS, que son: Id del mensaje, No teléfono, texto del mensaje (SMS), fecha y hora del mensaje. Es utilizado por la clase ClaseGSM, en la cual la utiliza para manipular mensajes SMS que ingresan módem, para posteriormente cargar los datos en una bandeja de entrada, que también es una colección de la clase Sms.

Figura 61. **Código de la clase Sms**

```
Public Class Sms
   Private IDMensaje As String
   Private NroTelefono As String
   Private SMS As String
   Private pFECHA As String
   Private pHORA As String
   Public Property Mensaje() As String
      Get
         Return SMS
      End Get
      Set(ByVal value As String)
         SMS = value
      End Set
   End Property
   Public Property IDSms() As String
      Get
         Return IDMensaje
      End Get
      Set(ByVal value As String)
         IDMensaje = value
      End Set
   End Property
   Public Property Telefono() As String
      Get
         Return NroTelefono
      End Get
      Set(ByVal value As String)
         NroTelefono = value
      End Set
   End Property
End Class
```

```
Public Property Fecha() As String
   Get
      Return pFECHA
   End Get
   Set(ByVal value As String)
      pFECHA = value
   End Set
End Property
Public Property Hora() As String
   Get
      Return pHORA
   End Get
   Set(ByVal value As String)
      pHORA = value
   End Set
End Property
```

Fuente: elaboración propia.

4.6. Codificación de la clase BaseDatos

En este apartado se detalla el código de cada uno de los elementos de la clase BaseDatos como lo son sus atributos y métodos definidos en la etapa de diseño. El espacio de nombres que necesita es *System.Data.Odbc*, sus atributos son:

Figura 62. **Declaración de atributos de clase BaseDatos**

```
'Conexion a BD por medio de OBDC
Public conexionDB As OdbcConnection
'Nombre del ODBC a utilizar
Dim Odbc As String
'Usuario
Dim Usuario As String
'password
Dim Pwd As String
```

Fuente: elaboración propia.

Los métodos de esta clase permiten al sistema interactuar con la base de datos de forma transparente, todos sus métodos son simples, y como no forma parte del objetivo de este trabajo no se profundizara más en la explicación.

Figura 63. **Código de la clase BaseDatos parte 1**

```
'Establece conexion con la BDD, tomando los paremetros configurados en el archivo .config
Sub AbreConexion(ByVal vOdbc As String, ByVal Usr As String, ByVal pw As String)
    Try
        Odbc = vOdbc
        Usuario = Usr
        Pwd = pw
        conexionDB = New OdbcConnection("dsn=" & Odbc & ";uid=" & Usuario & ";pwd=" & Pwd & ";")
        conexionDB.ConnectionTimeout = 60
        conexionDB.Open()
    Catch ex As OdbcException
        MsgBox("Error en la conexión " & ex.Message)
    End Try
End Sub
'Cierra la conexion de la BDD
Sub CierraConexion()
    Try
        conexionDB.Close()
    Catch ex As OdbcException
    End Try
End Sub
'Verifica que la conexion a la BDD este abierta
Sub RevisaConexionDb()
    If conexionDB.State = ConnectionState.Broken Or conexionDB.State = ConnectionState.Closed Then
        conexionDB.Open()
        Debug.Print("Abrir RevisaConexionDb " & Now & "")
    End If
End Sub
'ejecuta una instruccion SQL (select) en la BD, retornando el valor indicado por Indice.
Function EjecutaSql(ByVal Sql As String, ByVal Indice As Integer) As String
    Dim comandoSQL As OdbcCommand
    Dim resultadoSQL As OdbcDataReader
    Dim res As String
    Try
        RevisaConexionDb()
        comandoSQL = New OdbcCommand(Sql, conexionDB)
        resultadoSQL = comandoSQL.ExecuteReader
        resultadoSQL.Read()
        If resultadoSQL.RecordsAffected <> 0 Then
            res = resultadoSQL.GetValue(Indice)
        Else
            res = ""
        End If
        Return res
    Catch err As Exception
        MsgBox("Error en ejecucion SQL: " & err.Message)
        Return ""
    End Try
End Function
```

Fuente: elaboración propia.

Figura 64. **Código de la clase BaseDatos parte 2**

```
'ejecuta una instruccion SQL (insert,exec) en la BD, retornando si se ejecuto con exito(falso o ver
Function EjecutaSql2(ByVal Sql As String) As Boolean
    Dim comandoSQL As OdbcCommand
    Dim resultadoSQL As Integer
    Try
        RevisaConexionDb()
        comandoSQL = New OdbcCommand(Sql, conexionDB)
        resultadoSQL = comandoSQL.ExecuteNonQuery
        If resultadoSQL <> 0 Then
        Else
        End If
        Return True
    Catch err As Exception
        MsgBox("Error en ejecucion SQL2: " & err.Message)
        Return False
    End Try
End Function
'Ejecuta un consulta sql(select) y devuelve los datos en un OdbcDataReader
Function ConsultaSql(ByVal Sql As String) As OdbcDataReader
    Dim comandoSQL As OdbcCommand
    Try
        RevisaConexionDb()
        comandoSQL = New OdbcCommand(Sql, conexionDB)
        Return comandoSQL.ExecuteReader
    Catch err As Exception
        MsgBox("Error en ConsultaSql: " & err.Message)
        Return Nothing
    End Try
End Function
```

Fuente: elaboración propia.

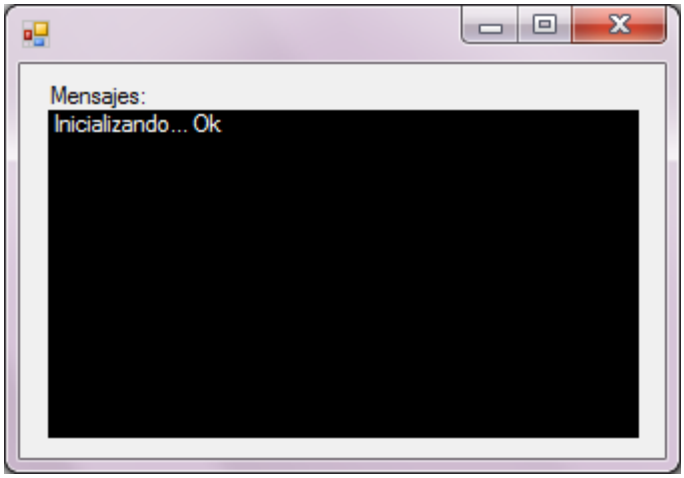
4.7. Pruebas

Hasta aquí ya se tiene el código fuente de todas las clases, lo único pendiente es la creación de la base de datos, para esto en el apéndice se incluye un *script* o secuencia de comandos, con el cual se puede crear la base de datos y las tablas, tienen exactamente las mismas características definidas en el capítulo 3.

Se de de asegurar de ingresar datos en la base de datos, tales como los departamentos, municipios, comunidades, agentes, contactos o responsables y los tipos de emergencias, si la base de datos está vacía las pruebas no funcionaran. Para estas pruebas en la base de datos previamente se han registrado estos datos.

Al ejecutar la aplicación se visualiza la pantalla de monitoreo, la cual será como se ve en la siguiente imagen:

Figura 65. **Ejecución de la aplicación**

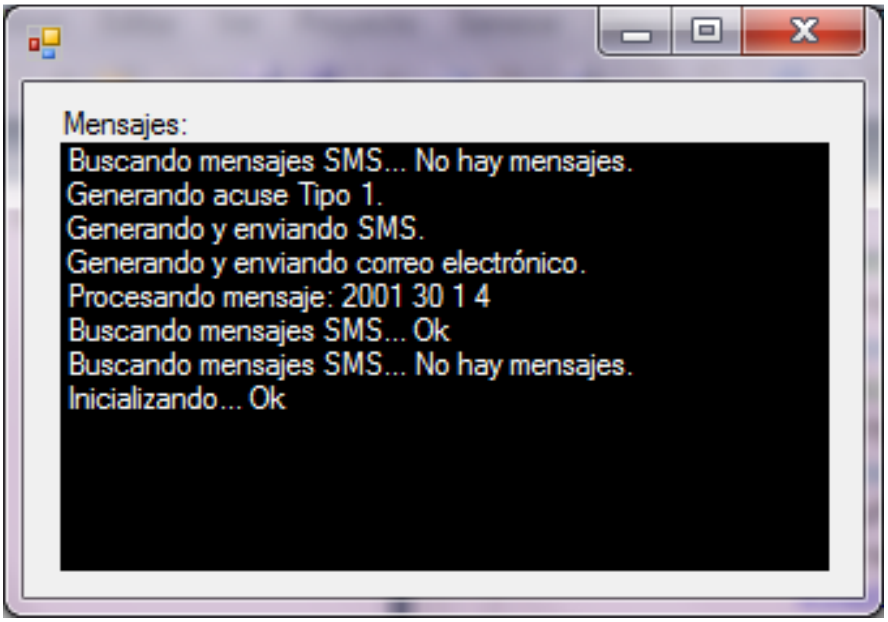


Fuente: elaboración propia.

Ahora que el sistema se está ejecutando, se procede a enviar un mensaje de texto con el código de alerta, para esta demostración se envía desde un teléfono celular el texto: 2001 30 1 4, donde 2001 es el código del municipio, y los primero 2 dígitos representan el código del departamento, 30 es el código que identifica a la comunidad, 1 es el código del agente monitor y por último 4 indica el tipo de emergencia. El mensaje se envía al número de teléfono asociado a la SIM que tiene el módem GSM, ver sección 2.3.

El mensaje es recibido y procesado por el sistema el cual muestra la siguiente información:

Figura 66. **SMS recibido y procesado**



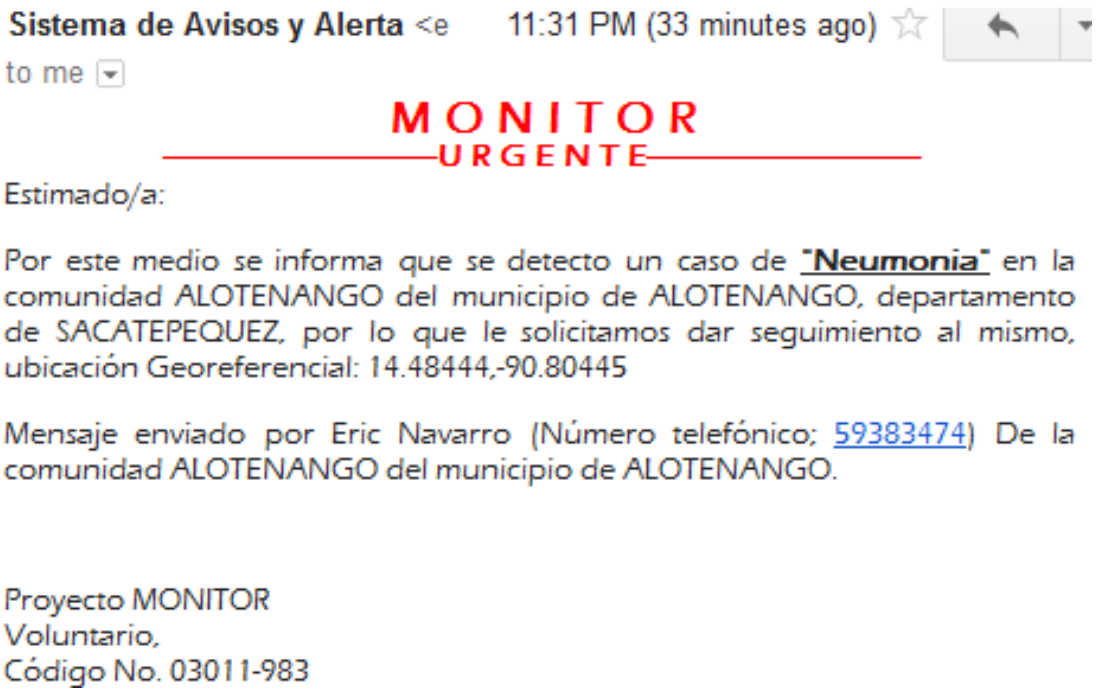
Fuente: elaboración propia.

Al teléfono del agente ha retornado un mensaje SMS de acuse de recibido con el texto: MONITOR: Su mensaje será atendido próximamente.

Al teléfono del responsable de la tabla sms_contactos, ha enviado el mensaje de alerta con el texto: Aviso de 59383474: Neumonia, comunidad: Alotengango muni: Alotenango, depto: Sacatepequez.

Y al correo electrónico, del mismo responsable ha generado un correo electrónico con un mensaje un poco más detallado pero que ofrece la misma información acerca de la emergencia:

Figura 67. **Email generado por el sistema**



Sistema de Avisos y Alerta <e 11:31 PM (33 minutes ago)

to me

MONITOR
URGENTE

Estimado/a:

Por este medio se informa que se detecto un caso de **"Neumonia"** en la comunidad ALOTENANGO del municipio de ALOTENANGO, departamento de SACATEPEQUEZ, por lo que le solicitamos dar seguimiento al mismo, ubicación Georeferencial: 14.48444,-90.80445

Mensaje enviado por Eric Navarro (Número telefónico; 59383474) De la comunidad ALOTENANGO del municipio de ALOTENANGO.

Proyecto MONITOR
Voluntario,
Código No. 03011-983

Fuente: elaboración propia.

CONCLUSIONES

1. Crear un Sistema de Aviso y Alerta usando un módem GSM es factible, y su funcionamiento es estable si dispone de componentes de software y hardware robustos.

2. Es posible tomar tecnologías existentes y encontrar nuevas formas de utilizarlas en beneficio de la sociedad.

3. He expandido los conocimientos en el área de telecomunicaciones, en la compresión de funcionamiento de las redes GSM, fruto del estudio realizado.

4. Los recursos necesarios para crear el sistema de Aviso y Alerta, no son difíciles de adquirir.

5. La solución aquí expuesta, es aplicable a otros sistemas de notificación y puede tener un enfoque social o empresarial.

RECOMENDACIONES

1. Como dispositivo de comunicación es posible utilizar los módem de internet móvil, que proveen las empresas locales de telecomunicaciones, y que también soportan los estándares GSM, pero para la solución sea estable y funcional, este tipo de módem no proveen la estabilidad necesaria, y dejan de funcionar correctamente al estar constantemente operando.

2. Se realizaron pruebas de estabilidad con un módem marca Wavecom, que únicamente da soporte al estándar GSM, los resultados fueron satisfactorios, por lo que se recomienda el uso de este tipo de módem especializados únicamente en tecnología GSM.

3. Para que una persona se emprenda en la creación de una solución basada en la tecnología GSM, es necesario que comprenda los conceptos desarrollados en el capítulo 2 de este documento, antes de aventurarse en el diseño del mismo.

4. Existen otros métodos alternos al de utilizar un módem GSM, como lo es el uso de Pasarelas de SMS, que son servicios de paga, proveídos a través de interfaces de software proporcionadas por quienes comercializan dichas soluciones.

BIBLIOGRAFÍA

1. ARAGÓN AGUILAR, Mónica. *Sistema de vigilancia alimentaria nutricional y de alerta temprana (Sisvan-at) para el programa de mejoramiento económico y seguridad alimentaria (promesa) de Care Guatemala*. Informe final de Maestría en Nutrición y Alimentación MANA. Universidad de San Carlos de Guatemala, Facultad de Ciencias Químicas y Farmacia. 2009. 71 p.

2. Comunidad Tecnológica Umts. *Arquitectura GSM* [en línea]. <http://intercambiostecnologicosdejago.blogspot.com/2010_05_01_archive.html>. [Consulta: 28 de marzo de 2014].

3. DEVELOPER'S HOME. *Short Message Service / SMS Tutorial* [en línea]. <http://www.developershome.com/sms/>. [Consulta: 28 de marzo de 2014].

4. HARRINGTON, Michael. *Undertanding SMS: Practitioner's basics* [en línea]. <http://mobileforensics.files.wordpress.com/2007/06/understanding_sms.pdf>. [Consulta: 10 de febrero de 2014].

5. IRIDIUM SATELLITE LLC. *Short Message Service Developer's Guide* [en línea]. August 1st 2004 <http://michrosat.com/downloads/SMS DevelopersGuideV1.pdf>. [Consulta: 10 de febrero de 2014].

6. ITZEP MANUEL, Carmen Beatriz. *Propuesta de un Sistema Municipal de Alerta Temprana de Inseguridad Alimentaria y Nutricional en Rabinal, Baja Verapaz*. Trabajo de graduación de Nutricionista. Universidad de San Carlos de Guatemala, Facultad de Ciencias Químicas y Farmacia. 2005. 74 p.

7. MICROSOFT DEVELOPER NETWORK. *SerialPort* [en línea]. <http://msdn.microsoft.com/es-es/library/system.io.ports.serialport%28v=vs.110%29.aspx>. [Consulta: 28 de marzo de 2014].

8. WIKIPEDIA. *Sistema global para telecomunicaciones móviles* [en línea]. <http://es.wikipedia.org/wiki/Sistema_global_para_las_comunicaciones_móviles>. [Consulta: 10 de febrero de 2014].

APÉNDICE

A continuación se presenta una secuencia de comandos, que crea una base de datos en MySQL, incluyendo todas las tablas que se usaron en la creación del sistema.

```
SET SQL_MODE="NO_AUTO_VALUE_ON_ZERO";
CREATE DATABASE `sms` DEFAULT CHARACTER SET latin1 COLLATE latin1_swedish_ci;
USE `sms`;
-- --------------------------------------------------------
CREATE TABLE IF NOT EXISTS `d_depto` (
  `depto_id` varchar(2) CHARACTER SET utf8 NOT NULL,
  `nombre` varchar(50) CHARACTER SET utf8 NOT NULL DEFAULT '',
  `region` varchar(10) CHARACTER SET utf8 DEFAULT '',
  PRIMARY KEY (`depto_id`)
) ENGINE=InnoDB DEFAULT CHARSET=latin1;
-- --------------------------------------------------------
CREATE TABLE IF NOT EXISTS `d_municipalidad` (
  `muni_id` varchar(4) CHARACTER SET utf8 NOT NULL,
  `nombre` varchar(255) CHARACTER SET utf8 DEFAULT '',
  `depto_id` varchar(2) CHARACTER SET utf8 NOT NULL DEFAULT '',
  PRIMARY KEY (`muni_id`,`depto_id`)
) ENGINE=InnoDB DEFAULT CHARSET=latin1;
-- --------------------------------------------------------
CREATE TABLE IF NOT EXISTS `d_comunidad` (
  `comunidad_id` varchar(5) CHARACTER SET utf8 NOT NULL,
  `nombre` varchar(200) CHARACTER SET utf8 DEFAULT NULL,
  `depto_id` varchar(2) CHARACTER SET utf8 NOT NULL DEFAULT '',
  `muni_id` varchar(4) CHARACTER SET utf8 NOT NULL DEFAULT '',
  `lat` decimal(12,6) DEFAULT NULL,
  `long` decimal(12,6) DEFAULT NULL,
  PRIMARY KEY (`comunidad_id`,`muni_id`)
) ENGINE=InnoDB  DEFAULT CHARSET=latin1 AUTO_INCREMENT=54713 ;
-- --------------------------------------------------------
CREATE TABLE IF NOT EXISTS `d_agente` (
  `agente_id` int(11) NOT NULL AUTO_INCREMENT,
  `comunidad_id` varchar(5) CHARACTER SET utf8 NOT NULL,
  `depto_id` varchar(2) CHARACTER SET utf8 NOT NULL DEFAULT '',
  `muni_id` varchar(4) CHARACTER SET utf8 NOT NULL DEFAULT '',
  `nombre` varchar(60) CHARACTER SET utf8 NOT NULL,
  `sexo` varchar(1) CHARACTER SET utf8 NOT NULL DEFAULT '',
  `tipodoc` varchar(8) CHARACTER SET utf8 DEFAULT NULL,
  `cedorden` varchar(5) CHARACTER SET utf8 NOT NULL DEFAULT '',
  `cedregistro` varchar(25) CHARACTER SET utf8 NOT NULL DEFAULT '',
  `telefono` varchar(40) CHARACTER SET utf8 NOT NULL DEFAULT '',
  `celular` varchar(40) CHARACTER SET utf8 NOT NULL DEFAULT '',
  `fechanac` date NOT NULL DEFAULT '0000-00-00',
  PRIMARY KEY (`agente_id`)
) ENGINE=InnoDB  DEFAULT CHARSET=latin1 AUTO_INCREMENT=984 ;
-- --------------------------------------------------------
CREATE TABLE IF NOT EXISTS `sms_contactos` (
  `contacto_id` int(11) NOT NULL AUTO_INCREMENT,
  `nombre` varchar(60) NOT NULL,
  `titulo` varchar(15) NOT NULL DEFAULT '',
  `cargo` varchar(50) NOT NULL DEFAULT '',
  `zonageografica` varchar(4) NOT NULL DEFAULT '',
  `telefono` varchar(40) NOT NULL DEFAULT '',
```

```
  `celular` varchar(40) NOT NULL DEFAULT '',
  `correo` varchar(150) NOT NULL DEFAULT '',
  PRIMARY KEY (`contacto_id`)
) ENGINE=MyISAM  DEFAULT CHARSET=latin1 AUTO_INCREMENT=9 ;
-- --------------------------------------------------------
CREATE TABLE IF NOT EXISTS `sms_mensajes` (
  `id` int(12) NOT NULL AUTO_INCREMENT,
  `telefono` varchar(15) NOT NULL,
  `fecha` datetime DEFAULT NULL,
  `mensaje` varchar(200) NOT NULL DEFAULT '',
  `fechaenvio` timestamp NOT NULL DEFAULT CURRENT_TIMESTAMP,
  `municipio` varchar(8) NOT NULL DEFAULT '',
  `comunidad` varchar(4) NOT NULL DEFAULT '',
  `cod_agente` varchar(150) NOT NULL DEFAULT '',
  `tipocaso` varchar(4) NOT NULL DEFAULT '',
  `ok1` int(1) DEFAULT '0',
  `ok2` int(1) DEFAULT '0',
  PRIMARY KEY (`id`)
) ENGINE=MyISAM  DEFAULT CHARSET=latin1 AUTO_INCREMENT=401 ;
-- --------------------------------------------------------
CREATE TABLE IF NOT EXISTS `sms_tipocaso` (
  `id_caso` varchar(5) NOT NULL,
  `nombre` varchar(60) NOT NULL,
  `asunto` varchar(60) NOT NULL DEFAULT '',
  `contenidomsg` varchar(1500) DEFAULT NULL,
  `contenidomsg2` varchar(160) NOT NULL DEFAULT '',
  PRIMARY KEY (`id_caso`)
) ENGINE=MyISAM DEFAULT CHARSET=latin1;
```