Estudio de técnicas de transmisión sin cables para control ejemplificado sobre un microcontrolador

Proyecto Fin de Máster para el
máster en Ingeniería de Sistemas y Control

Autor: José María Navarro España

Dirigido por D. José Sánchez Moreno

Convocatoria de junio

Curso: 2013-2014

Máster: Máster en Ingeniería de Sistemas y Control

Título: "Estudio de técnicas de transmisión sin cables para control ejemplificado sobre un microcontrolador"

Clase de proyecto: Tipo B (Proyecto específico propuesto por el alumno)

Alumno: José María Navarro España

Dirección: D. José Sánchez Moreno

Calificaciones

Autorización

Autorizamos a la Universidad Complutense y a la UNED a difundir y utilizar con fines académicos, no comerciales y mencionando expresamente a sus autores, tanto la memoria de este Trabajo Fin de Máster, como el código, la documentación y/o el prototipo desarrollado.

Firmado:

Firma del alumno

RESUMEN:
El presente proyecto fin de máster contiene dos partes principalmente. Una primera parte de investigación en la que se describen una variedad de comunicaciones sin cables existentes actualmente, al menos comercialmente, esbozando sus fundamentos teórico-físicos más relevantes. La segunda parte expone una implementación de uno de los sistemas de comunicaciones sin cables mencionados en la sección anterior. Esta implementación se realiza mediante un microcontrolador con un módulo de comunicaciones GSM. Se muestran los resultados conseguidos con tal implementación.

PALABRAS CLAVE:
GSM, GPS, microcontrolador, comunicaciones sin cables, sensor proximidad

Índice de imágenes

Índice de tablas

Abreviaturas

GSM: *Global System for Mobile communications*

GPS: *Global Positioning System*

SMS: *Short Message Service*

Siglas	Nombre en inglés	Nombre en castellano
BTS	*Base Transceiver Station*	Estación Emisora-receptora Base
VLR	*Visitor Location Register*	Registro de Localización de Visitantes
HLR	*Home Location Register*	Registro de Localización de Residentes
TRAU	*Transcoding Rate and Adaptation Unit*	Unidad de Frecuencia de Transcodificación y Adaptación
SIM	*Subscriber Identity Module*	Módulo de Identidad del Abonado
PLMN	*Public Land Mobile Network*	Red Móvil en Dominio Público
BSC	*Base Station Controller*	Controlador de Estación Base
EIR	*Equipment Identity Register*	Registro de Identificación de Equipos
MS	*Mobile Station*	Estación Móvil
MSC	*Mobile Services Switching Center*	Centro de Intercambio de Servicios Móviles

Elementos físicos en una red GSM

Siglas	Nombre
SM-AL	*Short Message Application Layer*
SM-TL	*Short Message Transfer Layer*
SM-RL	*Short Message Relay Layer*
SM-LL	*Short Message Lower Layer*

Niveles de la pila de protocolos de GSM

Siglas	Nombre
MTI	*Message Type Indicator*
UDHI	*User Data Header Indicator*
SRI	*Status Report Indication*
MMS	*More Messages to Send*
RP	*Reply Path*
PID	*Protocol Identifier*
TCP	*Transmission Control Protocol*
SRAM	*Static Random Access Memory*
EEPROM	*Electrically Erasable Programmable Read-Only*

Otras abreviaturas

Siglas	Nombre completo
UD	*User Data*
UDL	*User Data Length*
SCTS	*Service Centre Time Stamp*
DCS	*Data Coding Scheme*
PID	*Protocol Identifier*
OA	*Originator Address*
PDU Type	*Protocol Data Unit Type*
SCA	*Service Centre Address*

Elementos de la trama SM-TL de GPS

Contenido

Introducción

No ha sido el tiempo suficientemente cruel como para despojar al ser humano de su curiosidad; desde que el humano primitivo inventó la primera herramienta de piedra hasta los viajes espaciales de hoy día, el hombre nunca ha perdido su inventiva y curiosidad, algo que le ha hecho –y le hará- avanzar firmemente hacia nuevos descubrimientos. Dentro de este inmenso avance, el descubrimiento de la electricidad ha sido uno de los más destacados logros del ser humano. Relacionado con éste, la invención de los dispositivos de comunicación es igualmente de resaltar y es dicho avance el objeto de estudio del presente proyecto.

Las comunicaciones intrapersonales tuvieron su origen en el habla y sus códigos de transmisión oral. Miles de años después, surgió la escritura, que supuso una revolución en la historia, tanto que originó un hito en la misma, dándose en llamar la etapa previa a su conocimiento como la Prehistoria y la etapa posterior, Historia. Gracias a la escritura, la civilización avanzó increíblemente, y sus logros han quedado relatados en forma de tablillas de barro, de planchas de cera, de estelas pétreas, de pergaminos y de papeles antiguos. Con la revolución científica y tecnológica que arranca a mediados del siglo XX, las comunicaciones experimentan un indudable salto hacia delante, como así lo atestiguan las primeras comunicaciones por satélites, o más recientemente, las comunicaciones vía internet.

En el presente trabajo se relatarán las principales tecnologías de comunicaciones inalámbricas y se realizará una implementación de un envío con una de las tecnologías sin cables empleando un microcontrolador.

1. Objetivos

La invención de las tecnologías de las telecomunicaciones y más recientemente de la red de redes, ha supuesto un flujo de información antes inimaginable; este flujo se caracteriza por ser asequible, próximo e instantáneo. Asequible por cuanto el acceso a cualquier medio es relativamente barato, a diferencia de lo que ocurría hasta mediados del siglo pasado, cuando cualquier libro suponía el salario de varios días e incluso semanas de trabajo, motivo que sumado a los bajos niveles de alfabetización, hacían de los manuscritos una fuente valiosa para los que podían leerlos. Un flujo de información próximo, porque mientras que hasta mediados del siglo pasado la cultura se circunscribía a los libros y publicaciones en papel, la tecnología ha permitido una difusión amplia de los recursos culturales, de suerte que en cualquier lugar y a través de un terminal móvil o un ordenador se puede publicar información o acceder a ella. Instantáneo, puesto que merced a los avances tecnológicos comentados antes, ya no es necesario pasar el filtro de un editor y un librero como ocurría hace décadas; un simple clic permite difundir un artículo universitario, un correo u otro tipo de datos.

El presente proyecto fin de máster ha sido orientado de forma que en las páginas subsiguientes se relatarán con detalle varios de los sistemas actuales que hacen posible el flujo de información asequible, próximo e instantáneo en que se han convertido las comunicaciones hoy día. Se trata de un trabajo de investigación. Se ha llevado a cabo un proceso de investigación de varias tecnologías de comunicación, analizando sobre manera la organización lógica de tales tecnologías, como son sus tramas de datos. *Grosso modo*, se han distinguido dos medios que posibilitan el mencionado flujo de información: los

medios que usan tecnologías de satélites y aquellos que no emplean satélites. Dentro del primero de los grupos mencionados se hallan los envíos con tecnología GPS, mientras que en el otro grupo, las comunicaciones GSM y las conexiones *wireless* son los casos más representativos. Se ha decidido dirigir la investigación sobres las tecnologías mencionadas en los dos grupos anteriores.

Se describirán, pues, estos dos grandes grupos, comenzando la exposición de cada uno con un esbozo de su entorno de aplicación actual, proporcionando un marco de sus posibilidades de aplicación existentes, para después incidir sobre la estructura de sus tramas. En el caso de GSM y de las conexiones wireless, en donde conviven varios protocolos en la pila de protocolos de estos sistemas, se ha descrito la trama de uno de los protocolos, de forma que se obtenga una visión ciertamente real de su organización lógica pero sin perder de vista el sistema de comunicación en sí.

No sólo se ha considerado para el presente proyecto realizar un estudio amplio de importantes tecnologías que hacen posible actualmente las comunicaciones sin cables; la segunda parte del proyecto incluye una implementación de una comunicación simple mediante tecnología de comunicaciones GSM –*Global System for Mobile communications*- utilizando un microcontrolador sobre placa de evaluación Arduino Uno™ y un GSM Shield. La inclusión de tal implementación aporta coherencia al trabajo de investigación realizado en la primera parte de la presente memoria.

Por todo esto, se han determinado como objetivos para el presente proyecto fin de máster, cuya primera parte es de investigación:

1) Ofrecer una panorámica de las tecnologías que permiten las comunicaciones sin cables actualmente, distinguiendo en dos grandes grupos,

el que se basa en transmisión por satélites y el basado en comunicaciones sin satélites.

2) Detallar los fundamentos tecnológicos y otros aspectos más relevantes de cada uno de estos dos grupos de tipos de transmisiones en comunicaciones, prestando especial atención a tramas de datos de los sistemas de comunicaciones objeto de la investigación.

3) Realizar una implementación de una de las tecnologías sin cables descritas en uno de los dos grandes grupos anteriores; en concreto con comunicaciones GSM.

4) Usar un microcontrolador para la implementación propuesta para comunicación GSM, dada la amplia difusión de estos sistemas, sobre todo en aplicaciones empotradas.

Puesto que los microcontroladores se incluyen como un componente de las placas de evaluación y éstas son la plataforma hardware usada para prototipado generalmente, en lo que sigue se hablará indistintamente de placa de evaluación y de microcontrolador o *micro*. La implementación realizada cumple el objetivo de servir de ejemplo y de dar una mayor coherencia al trabajo de investigación. El sistema implementado se trata de un prototipo por usar una placa de evaluación en lugar de un circuito ex profeso, aunque es completamente funcional desde el punto de vista de las exigencias de los objetivos fijados.

2. Justificación

Tal y se ha expuesto anteriormente, la influencia de la tecnología en el mundo actual es innegable; ora los automóviles que permiten desplazarnos diariamente, ora los móviles para comunicarnos, etc. En concreto, el aspecto

de la relevancia de las comunicaciones telemáticas en nuestro entorno justifica en gran medida el estudio de varios sistemas de comunicación en el presente proyecto fin de máster. En cuanto a la implementación, se ha pretendido estudiar la programación de microcontroladores de forma conjunta al estudio de las tecnologías de comunicación sin cables. Se ha decidido realizar la implementación sobre un microcontrolador desde el momento en que éstos están ocupando una difusión cada vez mayor dada la disminución de sus costes. Las posibles aplicaciones de los mismos son amplias y, dado su coste reducido, viables para que sigan aplicándose a sistemas de control distribuido en instalaciones o como componentes integrantes de sistemas más complejos. Por otra parte, el continuo avance de la ingeniería del software ha logrado mejorar la optimización de recursos computacionales y la integración de sistemas complejos, entre otras cosas. Esto también tiene su reflejo en la mejora de las comunicaciones.

La aplicación diseñada en el presente proyecto se basa pues, en la interacción hardware-software para permitir un envío de datos en comunicación GSM mediante un microcontrolador –hardware- con lenguaje de programación C# y pseudo C –software-. Tal sistema creado permite dar coherencia al estudio de las tecnologías de comunicación que se desarrolla en las páginas subsiguientes, por cuanto que supone una implementación de una de ellas, y a la vez justifica la relación de las diferentes tecnologías de comunicación expuestas por servir como ejemplo del avance de las mismas.

3. Comunicaciones para largas distancias (usando satélites)

Aún queda relativamente cercano el primer satélite lanzado al espacio en el la segunda mitad siglo XX. Desde aquél, han sido numerosos los satélites

lanzados al espacio y que han permitido el desarrollo de las comunicaciones de largas distancias sin cables. Se describirá seguidamente uno de los sistemas de comunicaciones que usan satélites más destacados.

3.1 Sistemas de Posicionamiento Global (GPS)

En sus orígenes, el sistema de posicionamiento global o GPS tenía propósitos netamente militares. Fue desarrollado por el Departamento de Defensa de los Estados Unidos de América. Sin embargo, desde 1980, ha sido posible su uso civil. Consta de 24 satélites que constantemente se mueven sobre la Tierra a una altura de unos 20,200 km. En teoría permiten obtener información de posicionamiento desde cualquier punto del planeta. Su diseño garantiza que un mínimo de 4 satélites serán visibles en cada medición.

De forma general y relacionado con una de las funciones del GPS, cabe preguntarse cómo puede ser determinada la posición de un objeto. Uno de los métodos usados para determinar la posición es la triangulación con un cierto matiz que se comentará más abajo. Ciertamente, la triangulación es un método sencillo y podría ser eficaz en el propósito propuesto de determinar una posición. Obsérvese que con tres satélites tendríamos respuesta a las tres variables de posición (x, y, z) según la nomenclatura de los ejes cartesianos habitual. El problema radica en que las señales que emiten los satélites tardan un tiempo en ser recibidas, con lo que a las variables de posición (x, y, z) hay que añadir la variable tiempo. He aquí el matiz que se advirtió antes: realmente son necesarios cuatro satélites y esta es la justificación del diseño del sistema en el que se garantiza que siempre son alcanzables como mínimo cuatro satélites, puesto que de otro modo las mediciones no serían fiables –por la necesidad de calcular las cuatro variables mencionadas arriba-.

La arquitectura del GPS, desde el punto de vista de su organización interna, presenta varias secciones o segmentos [5]: segmento del usuario, segmento de control y segmento espacial. Cada uno de estos segmentos realiza una función en un entorno delimitado, y en conjunto explican la organización interna del sistema GPS.

Como intento de explicación sencillo de la actuación conjunta del entorno del GPS, supóngase el siguiente ejemplo: un usuario dispone de un temporizador para controlar el riego de su parcela, algo útil dado que el interesado vive alejado de la zona de riego. El usuario programa el temporizador para evitar regar en las horas de luz, para mantener más tiempo la humedad al evitar la evaporación máxima que se produce en las horas de luz. Siendo esta su prioridad –mantener más tiempo la humedad- y habiendo iniciado su riego en primavera, el usuario ha de ir retrasando la hora de finalización puesto que en las horas de luz van aumentando hasta llegar al solsticio de verano. Piénsese ahora en el sistema GPS: el usuario desea información de su posición, para lo que ha dispuesto una serie de satélites que, por cálculos de posición, determina sus coordenadas con cierta proximidad. Los satélites se hallan lejos del usuario y contienen relojes atómicos para conseguir averiguar el tiempo de recepción de las ondas, algo que junto con tres satélites más, hace posible determinar una posición. Cada cierto tiempo, es necesario corregir el reloj de los satélites. En el caso de los satélites, la necesidad de reajustar los relojes no se debe, como el ejemplo expuesto, por motivos horarios, si no por retrasos. De cualquier forma, es necesario un reajuste de los relojes tanto en el ejemplo con el sistema GPS. Para esto, el usuario que modificaba manualmente el dispositivo de riego en el ejemplo, se trata en el entorno del GPS de un conjunto de cinco estaciones permanentes

de control que se encuentran cercanas al ecuador [5].

3.1.1 Modelo matemático bajo GPS

Sin pretender ser excesivamente riguroso, sino más bien didáctico, se explican en este apartado grosso modo los cálculos matemáticos que se hallan bajo el sistema GPS. Servirá esta sección para delimitar el procesado de información del sistema GPS a grandes rasgos, de forma que se pueda conseguir un marco lo suficientemente concreto y con un cierto grado de exactitud como para permitir ofrecer una visión ciertamente adecuada del sistema.

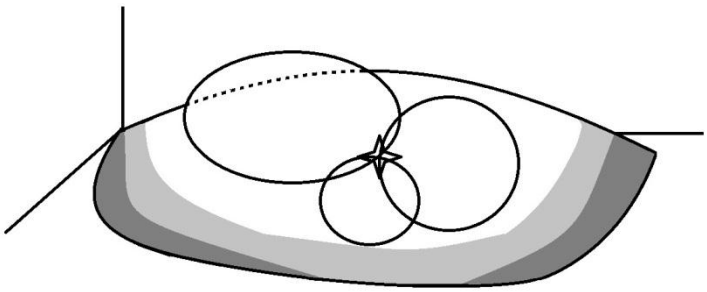

Imagen 3.1: Posición con tres satélites

En la imagen superior se ha representado un casquete de una esfera. Sobre él, aparece un punto marcada con una estrella que simboliza el punto actual del usuario. Las tres circunferencias, dos con origen próximo al punto marcado y la otra más alejada, representan las ondas de tres estaciones. Siendo así, obsérvese cómo la intersección de las tres circunferencias descritas determina la posición del usuario. Sin embargo, en el apartado precedente se mantuvo que al menos cuatro satélites eran necesarios para determinar el punto del usuario. Esto se debe a que los relojes de los satélites y del usuario tienen una leve variación en

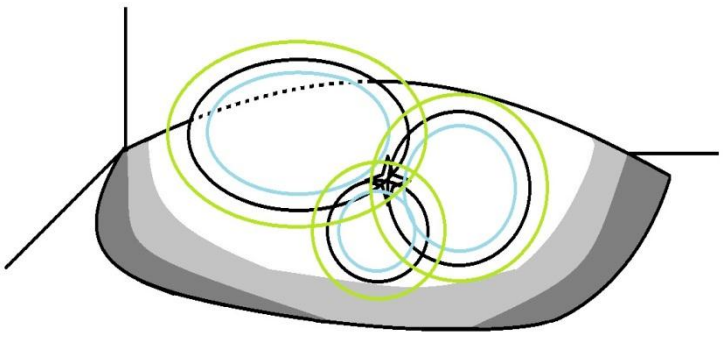

Imagen 3.2: Posición aproximada con tres satélites

sus medidas, por lo que empleando sólo tres satélites lo que se consigue es delimitar una zona bastante amplia en donde se ha de producir la intersección, pero sin exactitud. La imagen superior muestra esta zona de localización de la intersección, entre las circunferencias azul y verde, que son la representación de retrasos y adelantamientos entre el reloj del usuario y el de los satélites.

Básicamente, las ecuaciones que permiten calcular la posición del usuario se corresponden con las ecuaciones de las tres circunferencias de color negro dibujadas en la imagen 3.1. Después se introducen correcciones para evitar las zonas azules y verdes de indeterminación de la imagen 3.2.

En el caso ideal de transmisión instantánea y sin desviaciones de ningún tipo, sólo serían necesarios tres satélites, como se expuso con anterioridad. En ese caso, las ecuaciones implicadas serían:

$$Dist_{sat1} = \sqrt{(x_{sat1} - x_{usuario})^2 + (y_{sat1} - y_{usuario})^2 + (z_{sat1} - z_{usuario})^2}$$

$$Dist_{sat2} = \sqrt{(x_{sat2} - x_{usuario})^2 + (y_{sat2} - y_{usuario})^2 + (z_{sat2} - z_{usuario})^2}$$

$$Dist_{sat3} = \sqrt{(x_{sat3} - x_{usuario})^2 + (y_{sat3} - y_{usuario})^2 + (z_{sat3} - z_{usuario})^2}$$

Un cálculo aproximado, aunque con menor coste computacional, es posible para las ecuaciones. Si se considera el error en los relojes, las ecuaciones anteriores han de modificarse con un parámetro de error adicional:

$$Dist_{sat1} = \sqrt{(x_{sat1} - x_{usuario})^2 + (y_{sat1} - y_{usuario})^2 + (z_{sat1} - z_{usuario})^2} + E_{sat1}$$

$$Dist_{sat2} = \sqrt{(x_{sat2} - x_{usuario})^2 + (y_{sat2} - y_{usuario})^2 + (z_{sat2} - z_{usuario})^2} + E_{sat2}$$

$$Dist_{sat3} = \sqrt{(x_{sat3} - x_{usuario})^2 + (y_{sat3} - y_{usuario})^2 + (z_{sat3} - z_{usuario})^2} + E_{sat3}$$

En este punto se hace evidente la necesidad de incorporar una ecuación

adicional; al tener en cuenta el parámetro desconocido del error – $E_{sat\text{-}i}$ - son ahora cuatro las incógnitas y tres las ecuaciones consideradas. Es aquí cuando toma sentido la necesidad de tener en cuenta la información de un cuarto satélite, tal y como se indicó al inicio del texto. Se contempla entonces el conjunto de ecuaciones:

$$Dist_{sat1} = \sqrt{(x_{sat1} - x_{usuario})^2 + (y_{sat1} - y_{usuario})^2 + (z_{sat1} - z_{usuario})^2} + E_{sat1}$$

$$Dist_{sat2} = \sqrt{(x_{sat2} - x_{usuario})^2 + (y_{sat2} - y_{usuario})^2 + (z_{sat2} - z_{usuario})^2} + E_{sat2}$$

$$Dist_{sat3} = \sqrt{(x_{sat3} - x_{usuario})^2 + (y_{sat3} - y_{usuario})^2 + (z_{sat3} - z_{usuario})^2} + E_{sat3}$$

$$Dist_{sat4} = \sqrt{(x_{sat4} - x_{usuario})^2 + (y_{sat4} - y_{usuario})^2 + (z_{sat4} - z_{usuario})^2} + E_{sat4}$$

Sucede que la distancia de las ecuaciones precedentes es una aproximación, ya las medidas no son exactas. En las expresiones mostradas se ha supuesto las coordenadas ($x_{sat\text{-}i}$, $y_{sat\text{-}i}$, $z_{sat\text{-}i}$) como las coordenadas en tres dimensiones del satélite iésimo, y las coordenadas ($x_{usuario}$, $y_{usuario}$, $z_{usuario}$) las de la posición actual del usuario.

Se presta atención ahora al término que ha sido calificado en el presente texto como *error en los relojes*, si bien además cuantifica otros errores. Se recoge en lo que sigue lo expuesto en [3]. El mencionado término es una variable que recoge varias afecciones que se manifiestan al intentar cuantificar el tiempo de transmisión de los datos desde el satélite. Se puede desglosar como la cuantificación de la corrección del error debido a la ionosfera -ΔIi- por el conocido como fenómeno de dispersión, el error por ruido en el receptor -Ri-, la corrección del tiempo relativista -ΔRi- y un error por la posición del satélite –ΔDi-. Todos estos parámetros, en suma, han sido catalogados como el error $E_{sat\text{-}i}$ de las ecuaciones. Su forma concreta es:

$$E_{sat-i} = c(\Delta T_i + \Delta R_i + R_i + \Delta I_i) + \Delta D_i + K_i$$

Se puede comprobar la complejidad de la componente E_{sat-i}. El parámetro c es el de la velocidad de luz. El parámetro K_i alude al retraso del reloj propiamente dicho tanto en el usuario - ΔRU_i - como en el satélite - ΔRS_i - . Su valor es: $c\Delta RU_i - c\Delta RS_i + P_i$.

El término P_i se conoce como *pseudorango*. Refleja la diferencia temporal entre el momento de recepción por parte del usuario de los datos enviados por el satélite y el momento de envío de dichos datos por el satélite, multiplicada por la velocidad de la luz.

La situación esquematizada en la figura 3.2, en donde se ha querido esquematizar una cierta indefinición en la posición del usuario, es una situación más realista, ya que existen numerosos fenómenos que provocan esta poca exactitud. Algunos de los problemas son el de la forma elipsoidal del planeta Tierra y los errores debido a los relojes. El primero de los problemas se puede soslayar aplicando las ecuaciones propias de la figura geométrica en cuestión. Se reproducen en los párrafos que siguen el desarrollo expuesto en [3].

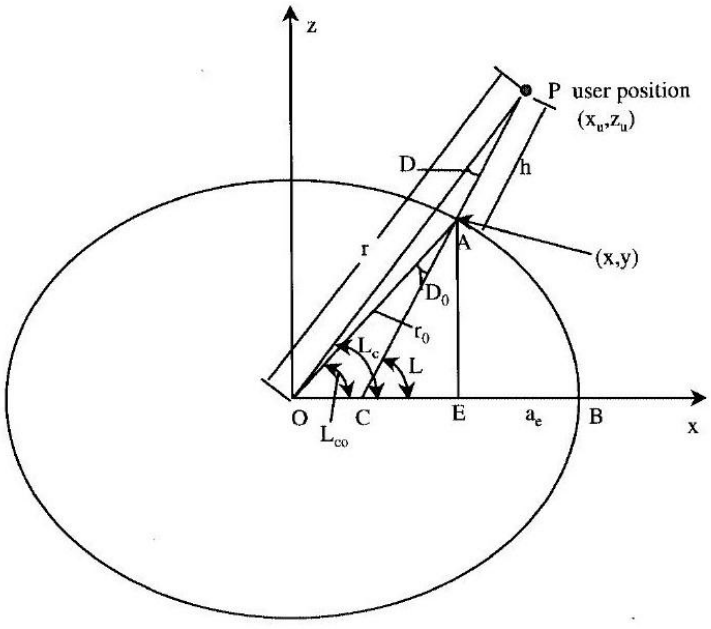

Imagen 3.3: Medidas para la latitud y la altitud

La imagen 3.3, tomada de [3], muestra los parámetros del cálculo de la altitud y la latitud. Para el cálculo de la altitud, el texto [3] parte de la expresión:

$$r = \sqrt{r_0^2 - 2r_0\, h \cos(\pi - D_0) + h^2}$$

Donde los parámetros que incluye la ecuación se corresponden con los indicados en el gráfico de arriba. Después de simplificar convenientemente y teniendo en cuenta que D0 llega a tener un valor de apenas tres milésimas para π/4 se puede obviar, llegando a:

$$r \approx h + r_0 - \frac{D_0^2\, h\, r_0}{2h + 2r_0} \approx h + r_0$$

De donde se desprende la relación de la altura h como: $h = r - r_0$.

También es necesario calcular la latitud. Para ello se parte de las relaciones geométricas contenidas en la imagen anterior. Concretamente, la referencia [3] sugiere partir de la relación: D = L - Lc, para lo que se hace necesario encontrar el valor del ángulo D.

$$OC = OE - CE = a_e \cos(\beta) - \frac{AE}{\tan(L)} = e_e^2\, OE$$

El parámetro β alude al ángulo medido desde el centro de la Tierra hasta un punto superficial en la simplificación del problema de localización empleando una esfera, tomando como sistema de referencia los ejes con origen en el centro

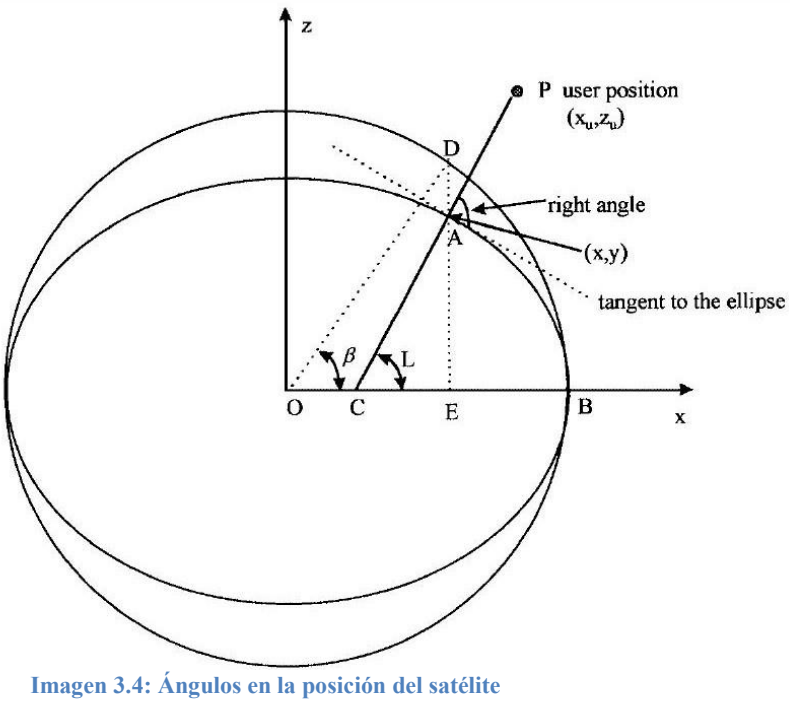

Imagen 3.4: Ángulos en la posición del satélite

terrestre, el eje vertical en el polo norte y el horizontal en el ecuador. La proyección sobre el eje x del punto superficial considerado intersecta en E. D, como se mencionó, es el ángulo formado entre el punto O y el E, a partir del punto superficial de la esfera, mostrado en la figura 3.3. Se han mostrado los cálculos de varios parámetros necesarios; no obstante no son lo únicos parámetros que son necesarios para un satélite, se remite a [3] para otros datos.

3.1.2 Segmento del usuario

Los cálculos expuestos en la subsección previa, tomados de [3], son algunos de las complejas operaciones matemáticas que sustentan el sistema GPS, junto con la parte tecnológica que permiten comunicar al usuario con el satélite para hacerle llegar al usuario su posición. Aquí se han incluido algunos de los cálculos más destacables. Se remite a [3] para mayor detalle.

Para enviar la información hasta el usuario, los satélites emiten dos ondas portadoras: L1 y L2. La imagen del margen, tomada de [5] muestra la relación entre ambas portadoras. La primera tiene una frecuencia 154 veces superior a la frecuencia fundamental generada por un reloj que existe a bordo del satélite. L2 es 120 veces superior a la frecuencia fundamental -10,23 Mhz-. Según se explica en [5], estas ondas portadoras transportan unos valores generados en el propio satélite: se tratan de los campos C/A y P. El campo P se modula a 10,23 Mhz –el mismo valor que el de la frecuencia fundamental-, mientras que C/A con la décima parte de la frecuencia fundamental. Como se

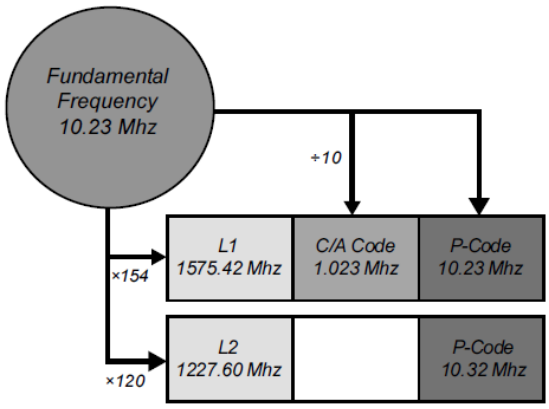

Imagen 3.5: Construcción de bandas L1 y L2

observa en la imagen 3.5, el código C/A es modulado en L1.

La imagen del margen, tomada de [3], muestra la cobertura de la principal señal del satélite: 21,3º y 23,4º para L1 y L2 respectivamente. Con el ancho de banda del gráfico, se observa cómo una zona queda oculta para el satélite en la zona que queda sombreada por la parte de la Tierra contraria a la enfrentada por el satélite.

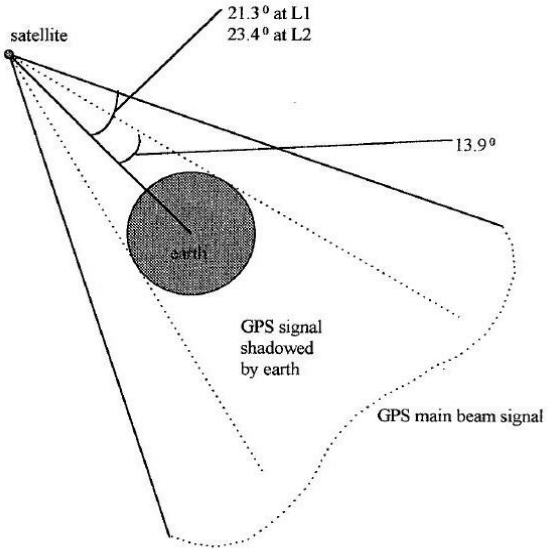

Imagen 3.6: Cobertura con L1 y L2

Como ya se explicó en el apartado 3.1.1 del presente trabajo, se determina la posición en el planeta del usuario por medio de una serie de ecuaciones propias de círculos, idealmente, y de elipses en casos reales dado que la Tierra no es exactamente esférica. Queda por concretar el modo en que se consigue determinar la distancia de cada uno de los satélites respecto del receptor. Para ello se parte del conocimiento de la velocidad de la transmisión de las ondas generadas por el satélite, que se corresponde con la velocidad de la luz, y de su definición física: velocidad como distancia recorrida por el tiempo, tal y como se indica en [5]. El tiempo que tarda la señal es también conocido, ya que es el usuario –el receptor- el que pretende conocer en un momento concreto información sobre su posición. El problema radica en saber exactamente cuándo emitió la señal el satélite desde el momento en que el usuario se interesó por la posición del satélite. Para averiguar este lapso de tiempo entre la emisión de la señal por el satélite y su recepción por parte del usuario se usa el campo C/A de las tramas que envía el satélite. Este campo se caracteriza por ser pseudoaleatorio. Quiere decir esto que su aspecto recuerda

al de un valor aleatorio, aunque sin llegar a serlo; acaba por repetirse. El campo C/A llega al receptor y lo usa para obtener una aproximación de la distancia; el receptor genera con su reloj interno un C/A y lo compara con el C/A recibido del satélite: como quiera que el valor del campo C/A es pseudoaleatorio y se acaba repitiendo cada cierto tiempo, la diferencia de tiempo entre el valor del campo C/A recibido desde el satélite y el C/A generado en el receptor se usa para calcular la distancia, teniendo en cuenta la velocidad de la luz a que están sometidas las transmisiones.

3.1.3 Segmento espacial

Este segmento es el más alejado al usuario, en lo referido a su proximidad física, ya que se halla más de 20.000 km sobre la superficie terrestre; concretamente, se sitúan a 20.200 km. Los veinticuatro satélites que conforman la red del GPS en este segmento, se disponen de la forma mostrada en la imagen anterior, tomada de [6]. Existen seis orbitales en los que se reparten equitativamente los veinticuatro satélites, constituyendo en conjunto la constelación del sistema GPS. También con una distribución equitativa sobre la esfera terrestre, los seis orbitales están separados sesenta grados para cubrir en suma a la Tierra por completo. Los satélites del segmento espacial tienen un período de 11 horas 57 minutos y 57 segundos, aproximadamente.

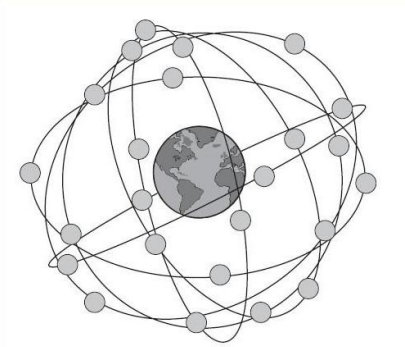

Imagen 3.7: Constelación de satélites de GPS

Cada satélite está equipado con dos antenas, una para cada una de las dos bandas, L1 y L2. Las tramas de GPS tienen el aspecto mostrado en el gráfico inferior, de [28]. Se aprecia que el mensaje de navegación contiene

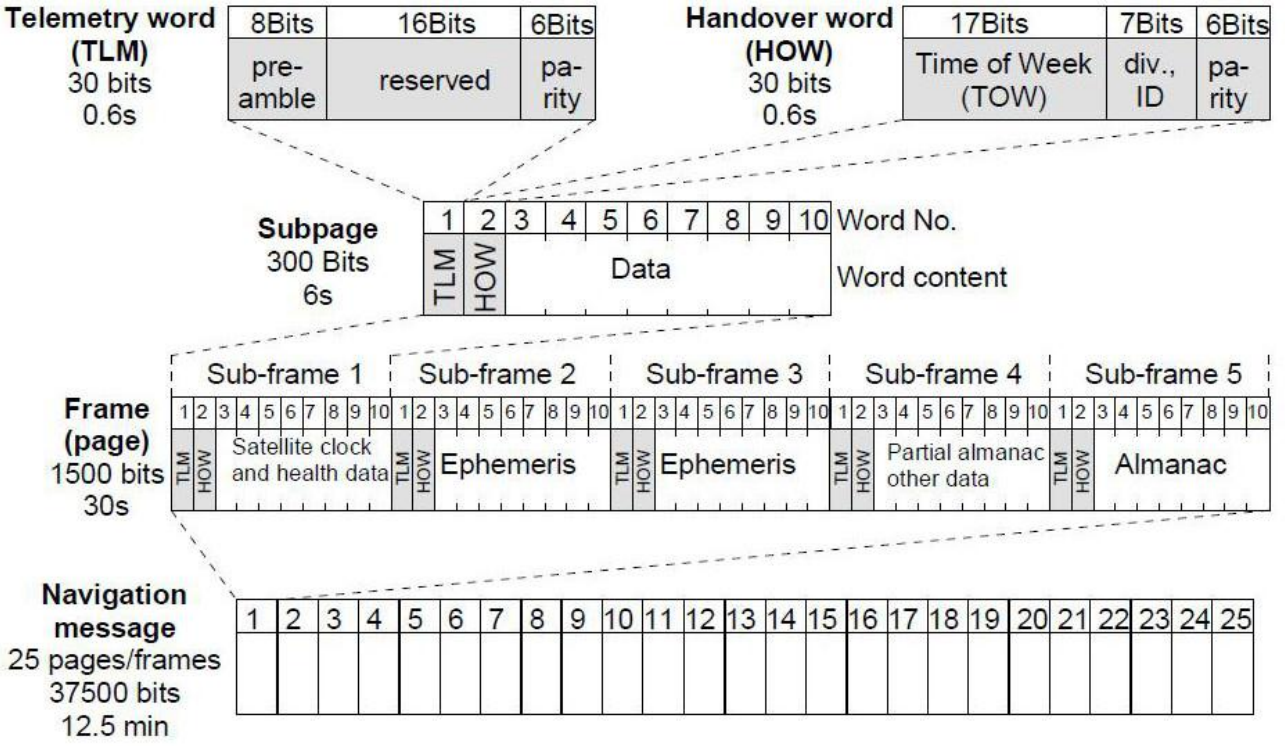

Imagen 3.8: Trama de datos de GPS

internamente una serie de campos de menor tamaño que a su vez contienen otros; el mensaje de navegación se puede contemplar como varias tramas o frames, y éstas como subtramas o subframes que a su vez contienen palabras y éstas últimas, bits. Las relaciones de las unidades de transmisión usadas en las tramas de GPS son las siguientes:

Bit < Palabra <Subframe < Página < Superframe.

Concretamente, la escala lógica anterior se cuantifica según: 30 bits = 1 palabra; 10 palabras = 1 subframe; 5 subframes = 1 página; 25 páginas = 1 superframe. En la imagen 3.8 se muestran la trama de datos. El campo TLM contiene unos bits

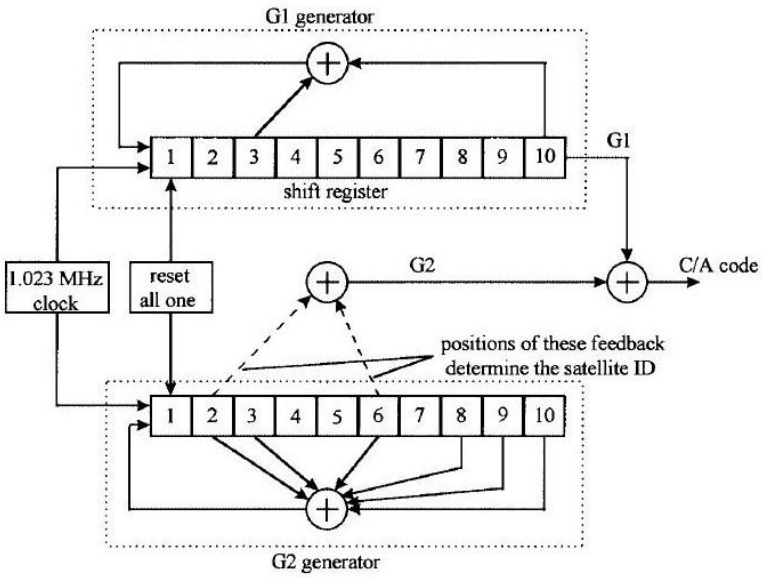

Imagen 3.9: Obtención del código C/A

reservados, antecedidos de ocho bits de preámbulo y seis de paridad al final. El campo HOW contiene el momento de la semana, además de otros bits como los de paridad. El momento, tal y como se expone en [28], se trata de un número que se va incrementando cada seis segundos que pasan en una unidad y que empieza en 0 para el instante entre 23:59:59 del sábado y las 00:00:00 del domingo. En cuanto a los *subframes* o subtramas, el primero de ellos guarda la hora y otros datos, el segundo y tercero el efemérides, el cuarto y el quinto contienen los datos conocidos en inglés como *almanac* que se refieren a datos de posición de los satélites que presentan leves desajustes pero que permiten calcular una posición aproximada de los mismos. En la imagen 3.10 se muestra los datos de un almanac de un satélite, tomada de [28]. Entre los datos del alamanc están la excentricidad y la inclinación orbital. El efeméride que aparece en las subtramas –*ephemeris* en inglés- recoge información sobre la órbita del satélite en cuestión.

```
******** Week 424 almanac for PRN-01 ********
ID:                         01
Health:                     000
Eccentricity:               0.6912231445E-002
Time of Applicability(s):   405504.0000
Orbital Inclination(rad):   0.9911766052
Rate of Right Ascen(r/s):  -0.7417838788E-008
SQRT(A)  (m 1/2):           5153.549316
Right Ascen at Week(rad):  -0.1640348434E+000
Argument of Perigee(rad):  -1.812852621
Mean Anom(rad):            -0.1197433472E+000
Af0(s):                     0.1583099365E-003
Af1(s/s):                   0.3637978807E-011
week:                       424
```

Imagen 3.10: Almanac de un satélite

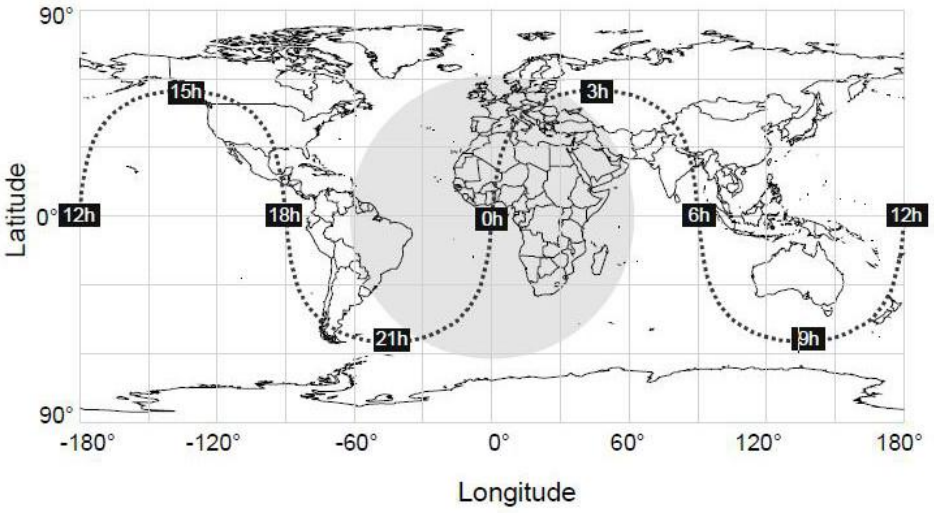

Imagen 3.11: Recorrido de un satélite en un día

Para crear el código C/A se lleva a cabo un proceso que busca generar dos secuencias de tipo ruido pseudoaleatorio [3], que son nombradas

como "G1" y "G2" en la imagen 3.8 tomada de [3]. El sistema de generación emplea dos registros, uno para cada secuencia, que son actualizados por una frecuencia 1,023 MHz. En la imagen 3.10, tomada de [28], se muestra el recorrido de un satélite durante una jornada de un día completo.

3.1.4 Segmento de control

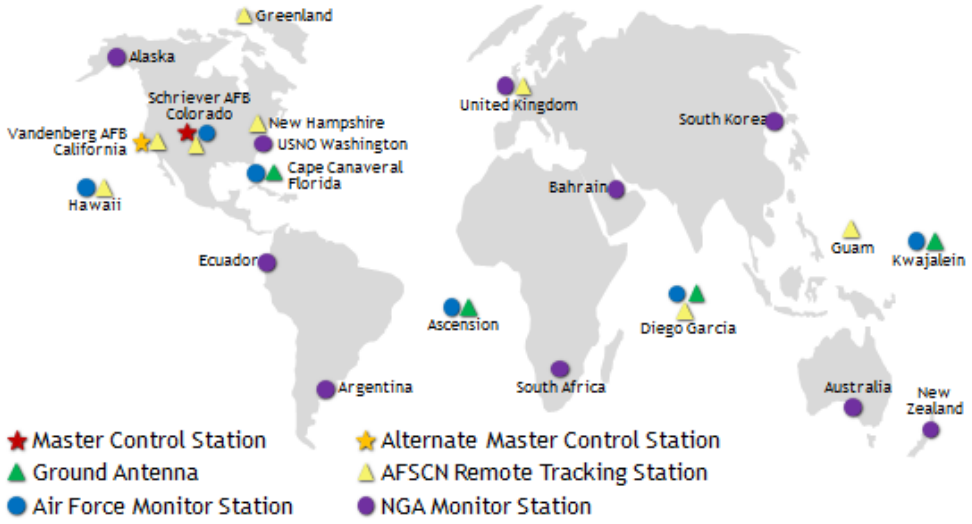

Imagen 3.12: Localización de estaciones de control y antenas para GPS

Este segmento es el encargado de controlar la programación de variables temporales en los satélites, ya que el reloj de los satélites experimenta un mínimo error cada cierto tiempo. Las estaciones de control se encargan, por ejemplo, de corregir la diferencia temporal entre el tiempo de los satélites –que se empieza a contar a las 00:00:00 del 6 de enero de 1980- con el tiempo UTC actual. Además, este segmento realiza otras tareas como la supervisión del estado del satélite, monitorización general, y otras. Se han establecido cinco estaciones fijas –*Air Force Monitor Station*- en la Tierra para hacer posible las funciones desempeñadas por este segmento en Hawaii, Colorado Springs, Cabo Cañaveral, Isla Diego García e Isla Kwajalein. Además exite una estación maestra –*Master Control Station*- en Colorado Springs, varias antenas –*Ground Antenna*– y numerosas estaciones de monitorizado –*NGA Monitorizing Station*-, como se puede observar en la imagen anterior tomada de [29].

4. Comunicaciones en distancias medias

En la sección previa de la presente memoria se ha tratado el sistema GPS, exponiendo los aspectos más relevantes del mismo. Tal tipo de comunicación se ha considerado para ilustrar el caso de un sistema comunicaciones que requiere de satélites. En los subepígrafes de 4.1 se incluyen dos sistemas de comunicación que no requieren satélites. Dado que ambos están organizados con pilas de protocolos, se hablará, antes de entrar en tales sistemas, del protocolo OSI, a modo de presentación de este tipo de organización interna.

4.1 Protocolo OSI

El protocolo OSI es el protocolo cuya definición guarda relación con otros protocolos, como por ejemplo el TCP/IP que se expone en la subsección siguiente. OSI estructura el proceso de la comunicación, distinguiendo en él hasta siete *etapas* o niveles. Cada uno de estos niveles se encarga de una función concreta. Los niveles se muestran en la figura del margen. El *nivel físico* se encarga de, entre otras cosas: representar los bits, sincronizar los bits, fijar el modo de transmisión. El *nivel de enlace* tiene la tarea de controlar los errores, controlar el flujo y el direccionamiento físico, entre otros. El *nivel de red* se encarga de hacer llegar los paquetes a los nodos mediante direccionamiento lógico y encapsulamiento. El *nivel de transporte*, básicamente, ordena los paquetes que se reciben desde niveles inferiores. El *nivel de sesión*, según [1], se encarga de "establecer, mantener y sincronizar la interacción entre sistemas de comunicación". El nivel

Imagen 4.1: Pila de protocolo OSI

de sesión, como su propio nombre indica, se encarga de las sesiones. El *nivel de presentación* tiene entre sus funciones la de cifrar la información.

4.2 GSM

El sistema global para las comunicaciones o GSM por sus siglas en inglés -*Global System for Mobile communications*- está estandarizado por la *European Telecommunications Standards Institute* y aunque comenzó su expansión en la década de los noventa, sus orígenes pueden rastrearse hasta mediados de los años ochenta. Su arquitectura se muestra en la imagen 4.2 tomada de [13]. En dicha imagen se ha representado la arquitectura GSM con una visión ciertamente global. Se distinguen tres partes con los recuadros más exteriores: la estación móvil, donde se originan las llamadas o sms; la red de acceso, compuesta por estaciones transmisoras-receptoras y controladoras; y finalmente la red en sí, que a su vez puede subdividirse en varios módulos. En las siguientes subsecciones se detallarán cada una de estas partes mencionadas antes.

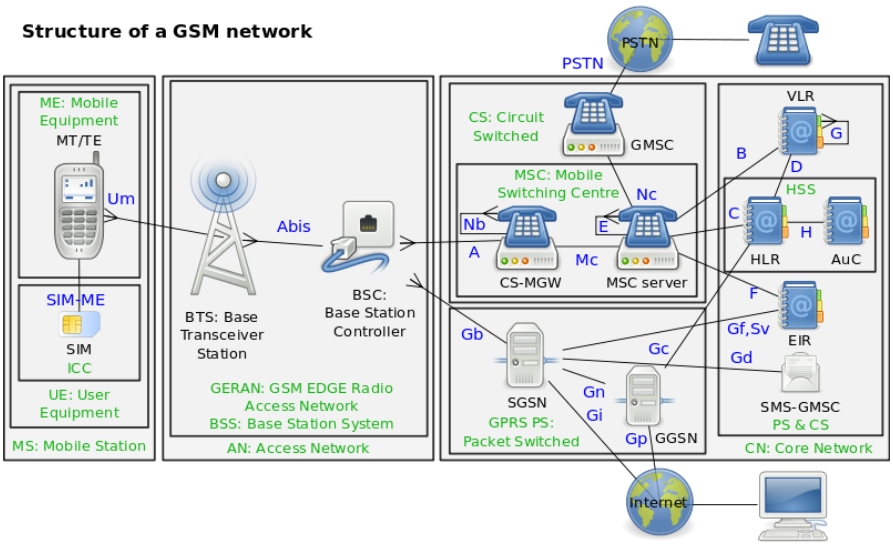

Imagen 4.2: Organización física del sistema GSM

4.2.1 Medio físico del GSM

La tecnología de GSM se basa en envío de datos por radiofrecuencias. Se definen tres bandas: GSM900, GSM1800 y GSM1900. La tabla ha sido

tomada de [22].

Sentido	GSM900
Uplink	890-915 MHz
Downlink	935-960 MHz
Sentido	GSM1800
Uplink	1710-1785 MHz
Downlink	1805-1880 MHz
Sentido	GSM1900
Uplink	1850-1910 MHz
Downlink	1930-1990 MHz

Tabla 4.1: Bandas de frecuencia para GSM en los enlaces

Donde *uplink* alude a que el sentido de la comunicación va desde la estación móvil hasta la estación base. En caso de *downlink* el sentido es el contrario que para *uplink*.

4.2.2 Arquitectura GSM

En la comunicación GSM, inicialmente la información enviada por el usuario viaja en forma de ondas de radiofrecuencias. Dado que en la presente memoria se ha dado importancia a las tramas de los distintos sistemas de comunicaciones comentados, las reflexiones que tienen por objeto el sistema de radiofrecuencias de GSM en sí quedan fuera del objeto de la presente memoria; se remite a [20] para tal información.

GSM permite enviar información por radiofrecuencias. Las frecuencias disponibles son limitadas y han de asignarse de forma temporal para una conseguir reutilizarla. Además, GSM ofrece una visión del mundo de las comunicaciones compuesto por gran cantidad de zonas de alcance de las

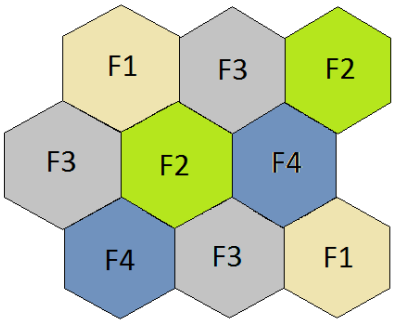

Imagen 4.3: Células en GSM

señales o células o celdas. Estas zonas han de evitar solaparse y permitir la comunicación desde cualquier punto. Teniendo en cuenta lo expuesto, la definición del protocolo acordó definir las zonas como figuras hexagonales, de suerte tal que no se asigne la misma frecuencia a dos células o celdas contiguas. Un ejemplo de la disposición de varias frecuencias usadas –F1, F2, F3, F4- sería la recogida en la ilustración del margen. Se puede comprobar que, gracias a usar hexágonos, no quedan espacios entre las celdas, de suerte que siempre sea posible la comunicación. Esto siempre, claro está, para una situación ideal. En este apartado se presentarán los dispositivos físicos que entran en juego en la comunicación GSM, para posteriormente, en la siguiente subsección, detallar una de las tramas de GSM.

Se listan a continuación los nombres de los distintos elementos físicos que componen GSM, tanto español como en inglés, dado que la mayoría de referencias usan las siglas de la terminología en inglés.

Siglas	Nombre en inglés	Nombre en castellano
BTS	*Base Transceiver Station*	Estación Emisora-receptora Base
VLR	*Visitor Location Register*	Registro de Localización de Visitantes
HLR	*Home Location Register*	Registro de Localización de Residentes
TRAU	*Transcoding Rate and Adaptation Unit*	Unidad de Frecuencia de Transcodificación y Adaptación
SIM	*Subscriber Identity Module*	Módulo de Identidad del Abonado
PLMN	*Public Land Mobile Network*	Red Móvil en Dominio Público

BSC	*Base Station Controller*	Controlador de Estación Base
EIR	*Equipment Identity Register*	Registro de Identificación de Equipos
MS	*Mobile Station*	Estación Móvil
MSC	*Mobile Services Switching Center*	Centro de Intercambio de Servicios Móviles

Tabla 4.2: Elementos físicos en una red GSM

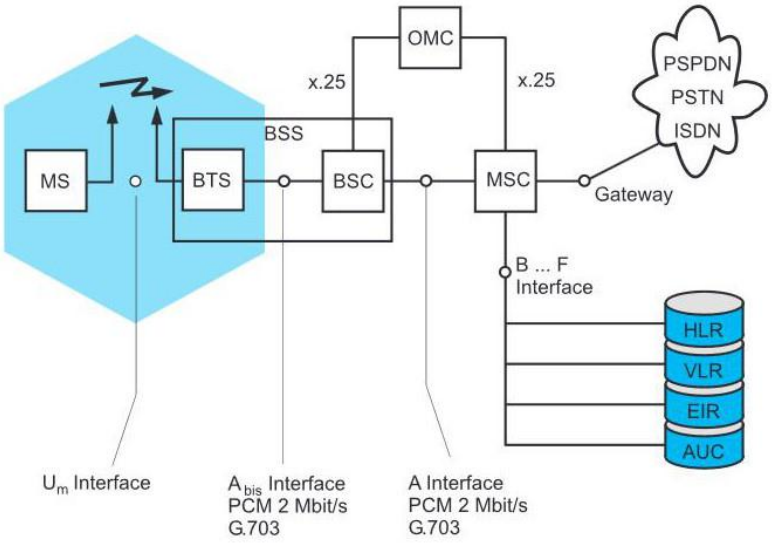

Imagen 4.4: Interfaces en una red GSM

La arquitectura GSM está compuesta, según [21] por tres bloques principales: los subsistemas de estaciones base –o *Base Station Subsystem*-, las Estaciones Móviles o -*Mobile Station*- y el Subistema de Red –o *Subsystem Network*-. Internamente, estos bloques están constituidos por otros elementos de menor entidad, cuyos nombres están recogidos en la tabla anterior. La disposición de los tres bloques principales se muestra en la imagen 4.4 de [21]. Se comentarán brevemente estos elementos para a continuación ahondar más en la arquitectura GSM.

Por MS o estación móvil se entiende a cualquier terminal, algo que puede ser identificado con un móvil de un usuario; incorpora la SIM. Un BTS o estación emisora-receptora base es básicamente, un sistema con un transmisor y un receptor. Un BSC o controlador de estación base ayuda en el control de un conjunto de BTS, informando sobre la potencia de la señal, entre otras tareas, según [21]. Los dispositivos BTS, junto con BSC constituyen

el subsistema de las estaciones base o BSS en inglés. El encaminamiento de las peticiones y la asignación de canal es tarea del MSC o centro de intercambio de servicios móviles.

Para realizar una llamada, el sistema GSM necesita tener acceso a los datos del usuario, como por ejemplo su IMSI o identificador móvil internacional de suscriptor –en inglés *International Mobile Subscriber Identification*- o identidad internacional para abonado de móvil. Por otra parte,

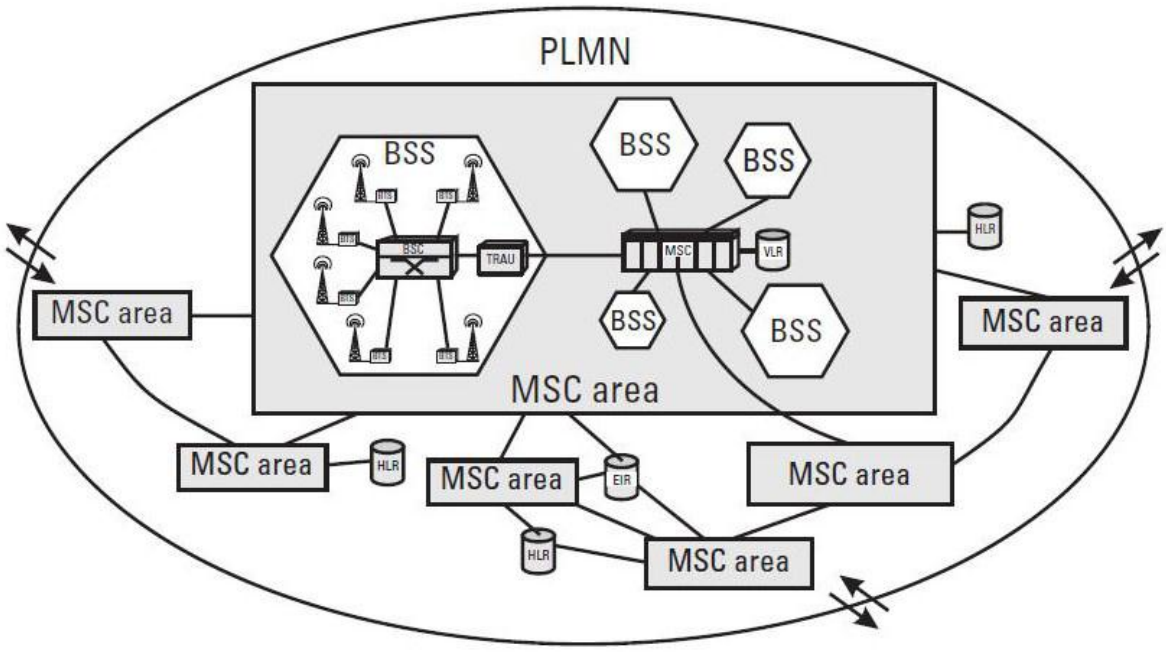

Imagen 4.5: Arquitectura de un sistema GSM

el usuario puede desplazarse a cualquier punto de otra región y necesariamente ha de poder seguir llamando con su mismo móvil, aunque como se ha comentado, el radio de alcance de una celda o célula es reducido. Para solventar este caso, se definieron dos registros: el HLR y el VLR, recogidos en la tabla última. El primero de los registros presentados responde a la necesidad de disponer de un registro permanente de los datos del usuario abonado. El segundo, solventa el problema de ofrecer el mismo servicio al usuario en caso de estar fuera de donde está abonado. Cuando un usuario se encuentra en otra zona distinta de la que se encarga el registro HLR del usuario, el registro VLR

de la zona visitada por el usuario solicita la información necesaria al registro HLR anterior, allá donde se encuentre el usuario, garantizándole el mismo servicio siempre. La información del terminal móvil en sí, como por ejemplo su IMEI o identificación internacional de dispositivo móvil es guardada en otro registro; el EIR. Para optimizar el canal de comunicación se comprimen los datos de las peticiones, tarea que realiza la unidad de frecuencia de transcodificación y adaptación o TRAU. La interacción de estos elementos se puede apreciar en la imagen 4.5 tomada de [22].

4.2.3 Generalidades sobre GSM

Los tres bloques principales expuestos, BSS, MS y subsistema de red, necesitan para la comunicación entre ellos de una interfaz para cada caso. Estas interfaces establecen unas tramas con campos perfectamente definidos y distintos, en aras de garantizar una comunicación eficaz. Estas interfaces, en las comunicaciones más próximas al usuario son: Um entre MS y BSS, Abis dentro de BSS entre BTS y BSC, y A entre BSS y MSC. Ver imagen 4.4.

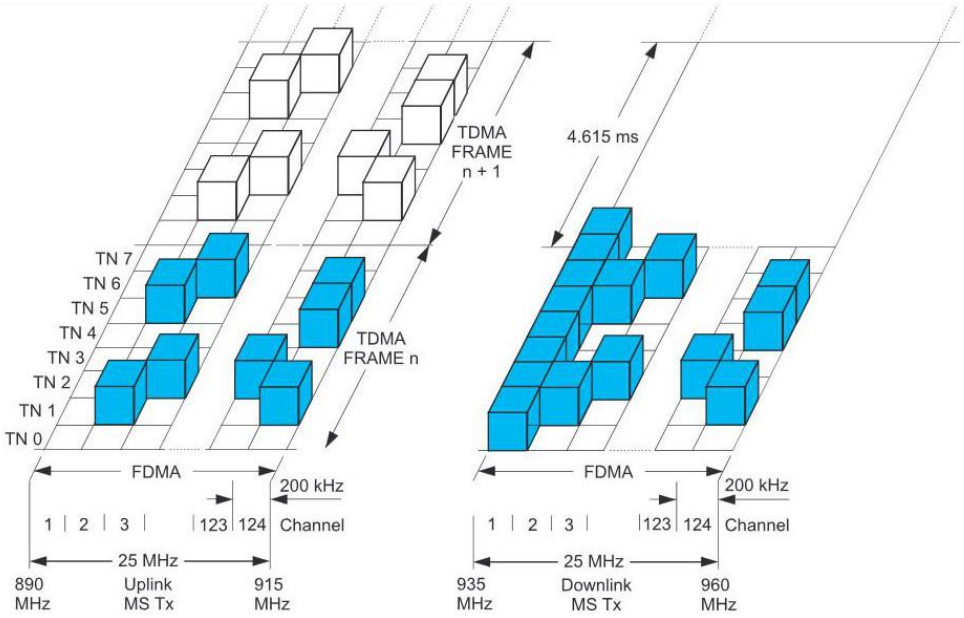

Imagen 4.6: Disposición de canales en GSM

Por otra parte, GSM divide el ancho de banda en "trozos" menores que se denominan canales. Para ello recurre al sistema TDMA/FDMA. Este

método combinado se aplica para conseguir 124 frecuencias portadoras en 25 Mhz dentro de uno de los anchos de banda definidos para GSM, como el GSM900. Como se explica en [21], después éstas son divididas temporalmente y son asignadas a las estaciones base –una o más para cada estación-. La imagen del 4.6, tomada de [21], muestra de forma gráfica la organización de los canales. Para mayor detalle se remite a [22]. En GSM los canales pueden ser de distintos tipos: canales de señalización y canales de tráfico. Estos últimos canales son los usados para llamadas de teléfono.

Hasta ahora se han expuesto los elementos más destacables de la arquitectura GSM desde el punto de vista de los módulos internos que permiten las comunicaciones con este sistema.

4.2.4 Envío de mensajes SMS

Los sms o *Short Message Service* es soportado por GSM y GPRS. Esta es la razón por la que se ha decidido implementar un sistema capaz de enviar sms; esto permite usar uno de tales sistemas de comunicaciones. La imagen inferior, tomada de [23] muestra la arquitectura para mensajes sms. Se puede observar que es similar a la arquitectura expuesta para un sistema GSM general; de hecho la imagen pretende representar la arquitectura de GSM que

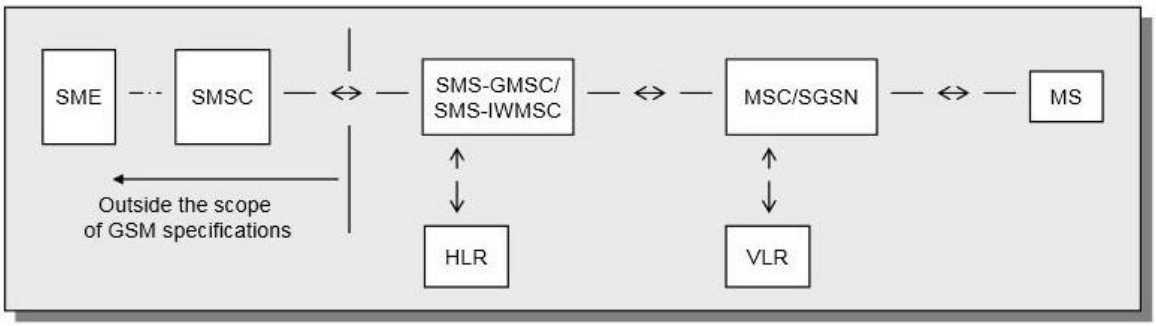

Imagen 4.7: Arquitectura del sistema GSM para mensajes

soporta el envío de sms. No obstante, aparecen varios elementos nuevos, como SME –*Short Message Entity*– que origina los mensajes; SMS-GMSC -*Gateway*

MSC for Short Message Service- que toma los mensajes de Centro de SMS –SMSC- y los encamina en el sistema.

4.2.4.1 Pila de protocolos

La pila de protocolos para el envío de sms se muestra abajo, de [25]:

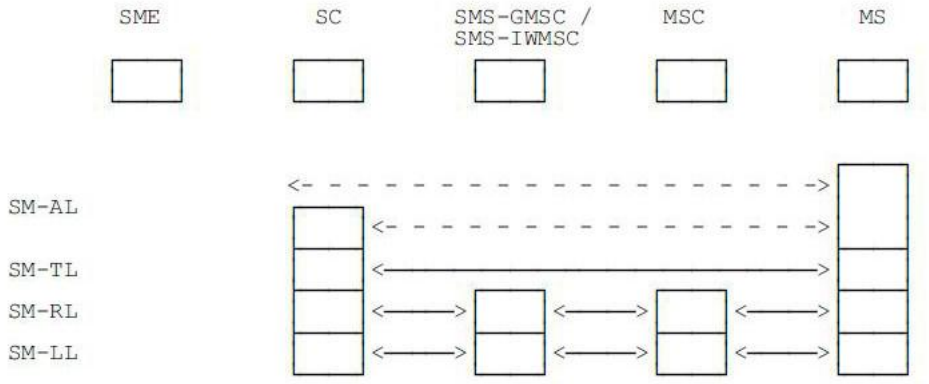

Imagen 4.8: Pila de protocolos de GSM

Dado que se busca en el presente trabajo exponer contenidos de varios sistemas de comunicación para llegar a contemplar tramas de estos sistemas, se ha considerado para esta labor que la trama de envío de sms del nivel SM-TL desde la estación móvil –MS- hasta el centro de servicio –SC- es adecuada –la PDU SMS-SUBMIT-. Los niveles, según se documenta en [25] son:

Siglas	Nombre	Funciones
SM-AL	*Short Message Application Layer*	La capa de aplicación es la superior de la pila. Usa los servicios de la capa inferior.
SM-TL	*Short Message Transfer Layer*	Sus funciones están ligadas al proceso de comunicación de los mensajes: entrega y recepción. Define hasta seis PDU distintas para sus procesos, según sean para reparto de mensajes, para recepción de éstos o para dar cuenta de posibles fallos.
SM-RL	*Short Message*	Ofrece el servicio para poder enviar el mensaje

	Relay Layer	hasta el destino –viajando por la pila-.
SM-LL	*Short Message Lower Layer*	Se trata de la capa más baja de la pila. Ofrece los servicios de cifrado, corrección de errores, comprueba la señal del canal, y otros [26].

Tabla 4.3: Niveles de la pila de protocolos de GSM

El nivel SM-TL dispone de seis PDU distintas: SMS-DELIVER-REPORT, SMS-COMMAND, SMS-SUBMIT, SMS-SUBMIT-REPORT, SMS-DELIVER y SMS-STATUS-REPORT. La trama de envío de sms, SMS-SUBMIT, en la imagen tomada de [23]:

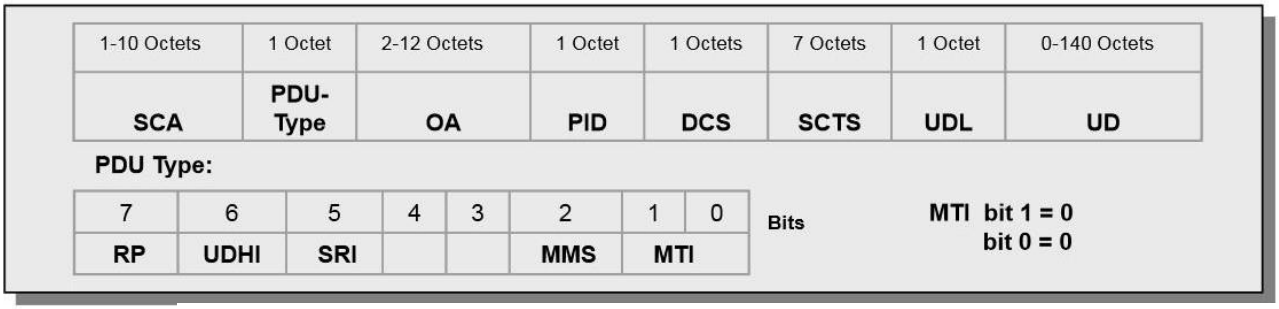

Imagen 4.9: Trama del nivel SM-TL de GMS

La identificación de los campos, tomada de [23], está recogida en la tabla de la siguiente página. El campo UD –*User Data*- contiene el mensaje que el usuario pretende remitir. UDL guarda la longitud del campo anterior. SCTS –*Service Center Time Stamp*- guarda un registro del momento en que en el Centro SMS fue captado el mensaje. DCS –*Data Coding Scheme*- alude al tipo de cifrado del mensaje remitido. El campo PID –*Protocol Identifier*- marca al Centro de SMS el protocolo a usar. OA –*Originator Address*- guarda la dirección desde la que se envió el mensaje –la entidad de mensaje corto o SME desde la que fue remitido-. El tipo de PDU a su vez contiene otros subcampos: MTI –*Message Type Indicator*-, UDHI –*User Data Header Indicator*-, SRI –*Status Report Indication*-, RP -*Reply Path*- y MMS –*More Messages to Send*-.

Siglas	Nombre completo
UD	*User Data*
UDL	*User Data Length*
SCTS	*Service Centre Time Stamp*
DCS	*Data Coding Scheme*
PID	*Protocol Identifier*
OA	*Originator Address*
PDU Type	*Protocol Data Unit Type*
SCA	*Service Centre Address*

Tabla 4.4: Elementos de la trama SM-TL de GPS

SCA guarda el número de teléfono del SC o centro de servicio en castellano.

El subcampo MTI permite determinar el sentido de la comunicación y los entes que se ponen en contacto por medio de ella, según [24]: 2 bits para indicar el tipo de mensaje; con valor '01' se usa para la trama anterior, SMS-SUBMIT. El subcampo MMS indica si hay más mensajes, caso en que toma el valor 0, mientras que siendo 1 no hay más mensajes. Se usa para las PDU SMS-STATUS-REPORT y SMS-DELIVER.

4.2.4.2 Ruta física en el envío de sms

Cuando se realiza un envío de un sms, entre la estación móvil y el centro de servicio, el mensaje ha de pasar por los módulos recogidos en la imagen, de [25].

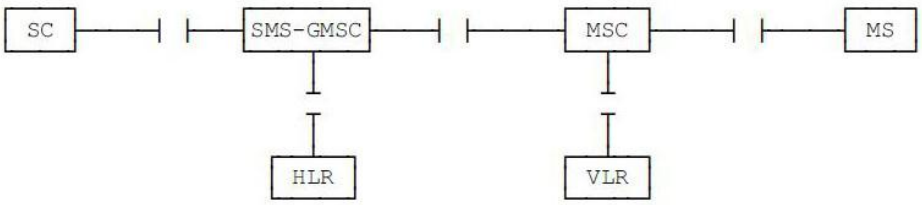

Imagen 4.10: Estructura GPS para mensajes

4.3 Redes wireless para Internet

Con un uso igual o mayor que el de las comunicaciones GSM, se encuentran las comunicaciones con redes wireless para internet. Aunque el concepto de red inalámbrica no se identifica inequívocamente con el de las redes wireless para internet, este tipo se trata ciertamente, de uno de los más

extendidos mundialmente. Por la importancia de internet en el mundo actual se ha centrado campo de la redes inalámbricas en este apartado del trabajo para redes wireless para internet.

Los inicios de internet están ligados al desarrollo de comunicaciones con intereses estratégicos. Bajo el cobijo del Departamento de Defensa de los Estados Unidos de América, los esfuerzos para el desarrollo de un sistema de envío de paquetes de datos fructificaron en la primera red de envío, conocida como ARPANET. Corrían los años 1960. Después, le siguieron otras redes como la Tymnet o la Telenet, entre otras.

El sistema de internet se puede describir grosso modo como un entorno físico que permite intercambio de información entre nodos. La información se codifica y empaqueta. Con el empaquetado se consiguen varios beneficios:

- En caso de pérdida de la información, se traduciría en pérdida de paquetes en lugar del total de la información enviada.
- Un nodo puede atender más de una comunicación de envíos de paquetes de forma que al usuario parezca que estas comunicaciones son respondidas en el mismo instante.

La interconexión entre los nodos terminales o computadores se realizó en los inicios de internet por cables como la única forma de conectar físicamente ambos ordenadores de la comunicación. Actualmente uno de los modos de uso de internet está en las redes sin cables, o wireless. La simplicidad de su arquitectura física es uno de sus principales alicientes, que no se necesita cableado alguno, al contrario de lo expuesto anteriormente. Únicamente basta con un nodo difusor, que puede ser cualquier rúter que incorpore una antena

wifi. Este nodo difusor es conocido como punto de acceso. Cada punto de acceso tiene un cierto radio de cobertura, que por lo general tiene el aspecto de una gran esfera centrada en dicho punto de acceso, debido a la forma en que se propagan las ondas.

4.3 Protocolo TCP/IP

El protocolo TCP/IP guarda ciertas similitudes con el protocolo OSI, aunque no dispone de exactamente el mismo número de niveles. Sin embargo, con otra agrupación distinta, los cuatro niveles de TCP/IP pueden compararse con los niveles de OSI desde el punto de vista de las funciones de cada uno, como se muestra en la figura del margen. Se observa que el primer nivel TCP/IP llamado *nivel de acceso a red* o *host*, es similar a los niveles uno y dos de OSI. El nivel segundo de TCP/IP denominado *de internet* se corresponde con el nivel de red de OSI. En ambos protocolos el nivel de transporte está identificado de igual forma. El cuarto nivel de TCP/IP o *de aplicación*, es comparable a los niveles de sesión, de presentación y de aplicación del protocolo OSI. De igual manera que en el caso de comunicaciones GPS se prestó cierta atención a las tramas, para TCP/IP se expondrán sus tramas en lo que sigue, las cuales se generan en el nivel de internet. Se centrará pues, la explicación de tramas en lo referente a tal nivel.

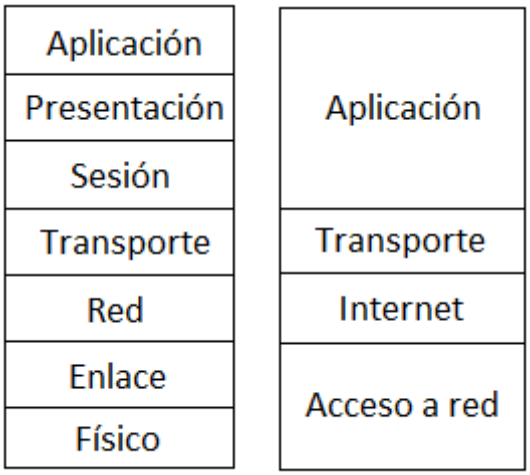

Imagen 4.11: Niveles del protocolo TCP y comparación con los niveles OSI

4.3.1 Organización del protocolo

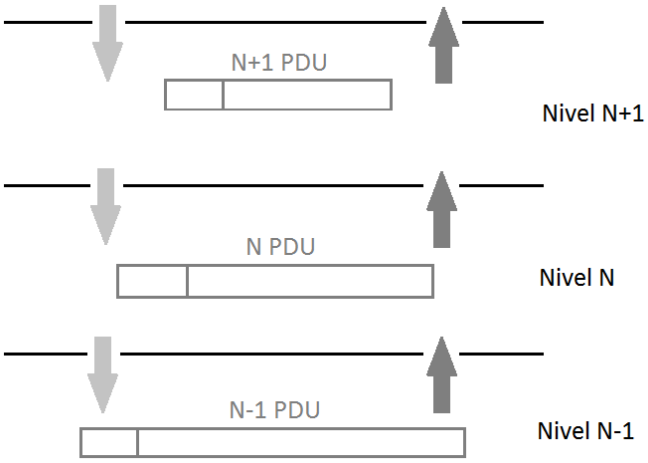

Imagen 4.12: Encapsulamiento y desencapsulamiento

Desde que la información es enviada por un usuario hasta que es visualizada por otro usuario, esta información es tratada, dividida, codificada, reunificada... Son muchos los procesos que se llevan a cabo. La información, para ser sometida a estos procesos diversos pasa por los distintos niveles de la pila de protocolos que se mostró antes. Como se ha comentado, cada uno de los niveles realiza alguna función concreta que es propia de dicho nivel. En el modelo OSI, comparable como se ha visto con TCP/IP en la imagen 4.11, se distribuye la información por la pila en forma de un objeto denominado PDU –*Protocol Transfer Unit* o en castellano Unidad de Datos del Protocolo-. El mencionado objeto se caracteriza por una parte inicial denominada *cabecera*, que es distinta y propia para cada nivel de la pila por el pasa la información. Cuando se desciende en la pila, la cabecera de un nivel concreto junto a los datos que llegan a ese nivel definen la PDU de dicho nivel, que después se pasará al nivel inferior donde se formará otra PDU con la adición de la cabecera del nivel inferior a esos nuevos datos, etc. Este proceso por el que los datos de un nivel superior, más próximo al usuario, son enviados hacia niveles inferiores, se conoce por *encapsulamiento*. Ascendiendo por la pila ocurre algo similar, aunque en sentido inverso; si bien en tal caso, en lugar de anteponer cabeceras a los datos, éstas son *retiradas*, de forma ordenada y fijada según el nivel en que se encuentre la información. Este otro proceso es el *desencapsulamiento*.

4.3.2 Nivel de internet

En el nivel de internet, el encargado es el protocolo IP. Aunque hace relativamente poco surgió IPv6, se hablará aquí de otro previo muy extendido: IPv4. La estructura interna de los datagramas es la siguiente:

Versión	HLEN	Servicio	Longitud total	
Identificación			Indicador	Offset
Tiempo de vida		Protocolo	Checksum	
Dirección IP de origen				
Dirección IP de destino				
Opciones				

Imagen 4.13: Campos del datagrama de IPv4

- **Versión:** dado que existen varias versiones de IP –IPv4 e IPv6- este campo de cuatro bits permite identificar la versión usada para el datagrama.

- **HLEN:** con 4 bits, este campo indica la longitud de la cabecera. La longitud oscilará entre 20 bytes como mínimo y 60 como máximo.

- **Servicio:** con longitud de 18 bits. Según [1], este campo permite definir dos significados: uno de los significados corresponde con los bits usados para determinar la prioridad de los datagramas –algo que se indica con los bits primero, segundo y tercero tomando valores entre cero y siete en binario-; el otro significado usa sólo los seis primeros bits –llamados subcampo *codepoint*- y determina una serie de servicios con diferente prioridad: para ser usados durante un breve tiempo –números de la forma $x_{i+1}=x_i+4$ desde x=1 hasta llegar a 61-, para administradores locales –números de la forma $x_{i+1}=x_i+4$ desde x=3 hasta llegar a 63- y para internet –números pares hasta 62-.

- **Longitud total:** con este campo se alude a la dimensión de la trama, desde el inicio de la cabecera hasta el final del datagrama.

- **Identificación:** la necesidad de este campo se hace patente al fragmentar un datagrama; un mismo valor en este campo indica que todos esos fragmentos con igual valor *Identificación* pertenecen al mismo datagrama.

Se consigue reunir así a los fragmentos. El tamaño es de 16 bits.

- Indicador: es una terna de tres bits que avisa de la situación de la trama respecto de la fragmentación: si el datagrama se ha de fragmentar o no y si hay más fragmentos o no después del datagrama actual. Estas dos posibilidades se indican con los dos últimos bits de la terna, ya que el primero está reservado.

* Combinación Reservado|0|1: el datagrama no se puede fragmentar y no se trata del último datagrama.

* Combinación Reservado|1|0: el datagrama se puede fragmentar y se trata del último datagrama.

- Offset: como se ha explicado arriba, el campo *Identificación* permite saber si un fragmento pertenece o no a un datagrama, pero una vez que se conoce el grupo de los fragmentos en que se ha dividido un datagrama, para volver a construir el datagrama original falta por saber la posición de cada fragmento dentro del datagrama; esta es la misión del campo *Offset*. Tiene una longitud de trece bits. El valor de este campo, según [1], se codifica como el cociente de la división por ocho de la posición del fragmento.

- Tiempo de vida: sirve para fijar el número máximo de rúteres que puede pasar el datagrama antes de ser considerado como descartable. Se evita así que el datagrama siga por un tiempo innecesario viajando por la red.

- Protocolo: existen numerosos protocolos, algunos de los más comunes se incluyen en la tabla inferior de [1] junto con el valor asociado para el campo.

Nombre de protocolo	Campo *Protocolo* de IPv4
TCP	6
ICMP	1
UDP	17

Tabla 4.5: Valores del campo Protocolo de datagrama de IPv4

- Checksum: comprobación de integridad de la trama mediante la

aplicación de operaciones sobre grupos de bits. Su uso es extensible a tramas de otros protocolos.

- **Dirección IP de origen:** remitente de la trama en formato IP de 32 bits.

- **Dirección IP de destino:** el destinatario de la trama IP de 32 bits.

- **Opciones:** es un campo de 60 bytes como máximo y 20 bytes como mínimo. Permite realizar operaciones como: guardar una relación de los rúteres por los que pase el datagrama, guardar el tiempo en que fue procesado el datagrama en un rúter, y otros más.

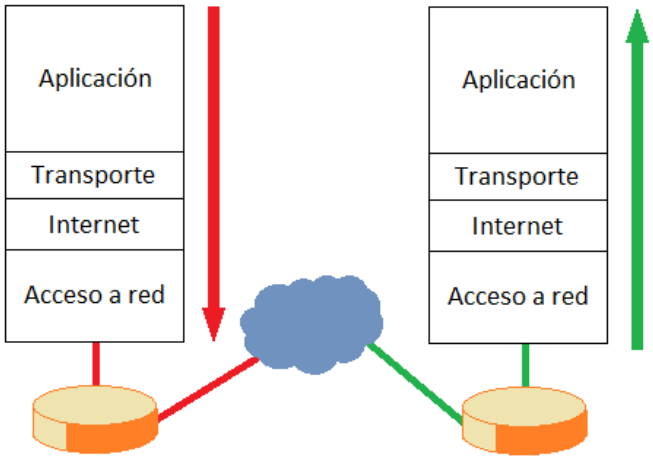

Imagen 4.14: Viaje de los datos entre nodos

La estructura del datagrama corresponde al del nivel 2 TCP/IP o nivel de internet o de red. En caso de ser enviado hasta otro usuario, el datagrama aún debería pasar por el nivel 1 o nivel de acceso a red, después del cual sería dejado en manos de un rúter o más, como se ha querido representar en la imagen superior con una zona nubosa, hasta llegar al rúter último antes de pasar ascendentemente por la pila de protocolo del destinatario. Al final, el destinatario llegará a ver, prescindiendo de las cabeceras, la información remitida por el usuario origen desde el nivel 4 o de aplicación de su pila de protocolo.

5. Sistema implementado

Hasta ahora, se han descrito algunos de los principales sistemas que son usados actualmente para comunicaciones. La importancia de los entornos expuestos –GPS, GSM, redes wireless- es innegable: la determinación de la

posición por zonas con escasas referencias y bajo cualquier condición atmosférica es vital para la seguridad en la navegación, tanto en usos militares como civiles; el envío de datos para comunicaciones móviles es la garantía de una mayor información; las comunicaciones con redes wireless, sobremanera internet, refuerzan la interconexión que en estos días parece ya algo irrenunciable.

Como ejemplificación de las tecnologías inalámbricas en general y de las comunciones GSM en particular, se ha diseñado un sistema que emplea este tipo de tecnología. El sistema consta de dos partes: el componente software y el componente hardware. La idea general, consiste en implementar una comunicación desde el microcontrolador con un teléfono móvil. Ambos elementos se podrían catalogar como de tipo hardware, -si bien incluyen un software avanzado-. Queda por concretar el componente software del sistema: consiste en dos programas, uno para el microcontrolador y otro será un programa en C# que se ejecutará en el ordenador y desde el que se podrá enviar la información que se desea hacer llegar al microcontrolador para que éste después, empleando comunicaciones GSM con el programa con tal propósito diseñado, haga llegar dicha información en formato sms al móvil.

Se tratará, pues, de una implementación que constata la realidad que suponen las comunicaciones inalámbricas, dando coherencia a la parte primera de la presente memoria que se trataba principalmente de aspectos técnico-teóricos sobre este tipo de comunicaciones. La imagen 4.15 recoge de forma esquemática el sistema aquí implementado. El ordenador representa el componente software, ya que el programa creado para especificar por parte del usuario los datos a enviar estará funcionando en el ordenador.

5.1 Organización del sistema

Como se ha comentado antes, el sistema consta de dos componentes, de forma general: uno software que es un programa en lenguaje C# y que se ejecutará en un ordenador, indicando mediante su interfaz los datos que se enviarán al móvil, y un programa en lenguaje para Arduino™ que se cargará en el microcontrolador para procesar los datos; en cuanto al componente hardware, incluye cuatro elementos que son el móvil que recibirá los datos del programa del ordenador, el ordenador, el microcontrolador de la placa Arduino™ Uno junto con la ampliación GSM Shield que permitirá hacer llegar los datos introducidos vía GSM al móvil y un sensor que se comentará después.

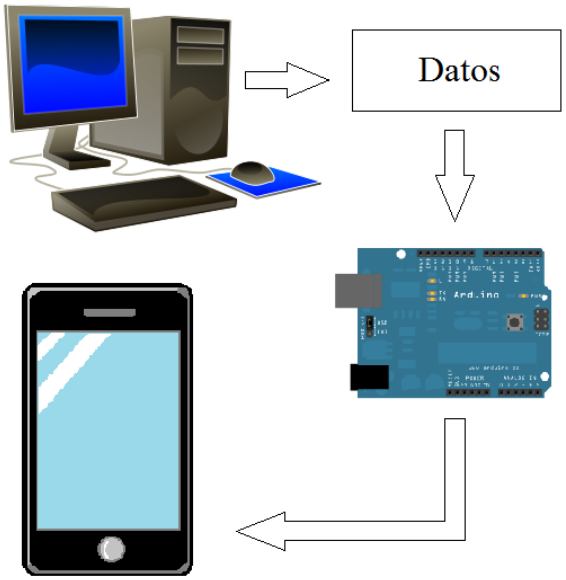

Imagen 5.1: Organización general del sistema sin sensor

Los datos enviados desde el ordenador podrían ser un texto legible o información cifrada. En la implementación aquí propuesta, para conferir una funcionalidad adicional, se permitirá enviar un texto con un cifrado afín. La configuración de los dos parámetros del cifrado afín que constituyen la clave de cifrado podrá ser efectuada desde el programa escrito en C# desde el que se puede introducir el texto. Como se comentará en la subsección 5.3.1, el cifrado afín aquí implementado requiere de una clave compuesta por dos valores, o de dos claves, expresado de otra forma. Estos dos valores han de cumplir ciertos requerimientos, como se detallará en la mencionada subsección. Teniendo en cuenta esto, la interfaz ha sido diseñada para permitir una fácil selección de las

claves –o el par de valores de la clave-.

Además de permitir enviar mensajes desde el ordenador a un dispositivo móvil directamente o con cifrado, se ha incluido en el sistema un sensor para detección de proximidad, reflejado en la imagen superior. De esta forma el sistema aquí propuesto tendrá, además del carácter cohesionador por tratar algunos de los conceptos de comunicaciones expuestos en la parte primera de la presente memoria, un carácter funcional, ya que junto con el sensor, son muchas las aplicaciones del sistema que pueden usar el sensor como disparador de otras acciones.

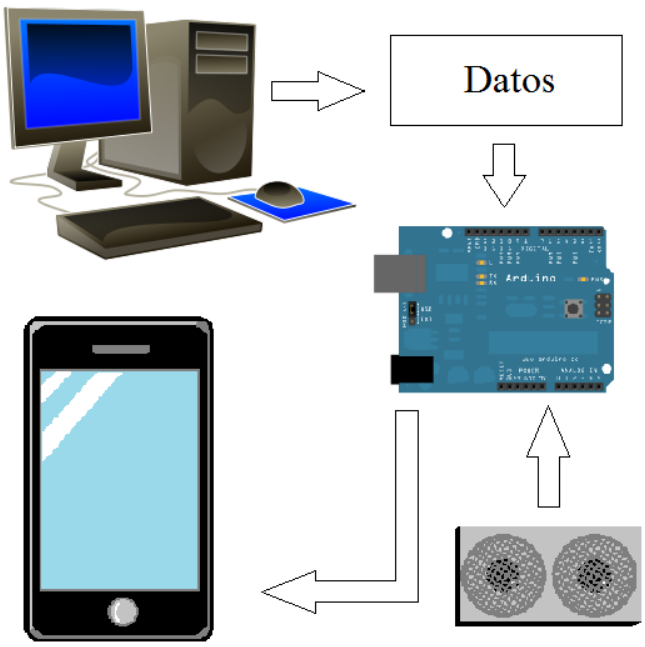

Imagen 5.2: Organización general del sistema incluyendo sensor

Se podría considerar el sistema aquí propuesto como una simplificación de cualquier sistema convencional de alarmas: se detecta movimiento mediante varios sensores y esta información se hace llegar al centro de procesado y de ahí, por un aviso como un mensaje sms, al usuario. En caso de usar el sensor, el esquema de funcionamiento es similar al expuesto para el caso de las alarmas, con la excepción de que en el sistema aquí propuesto no se incluye un centro de procesado sino que simplemente es la comunicación del sensor vía ordenador y microcontrolador al usuario, y que el sistema sólo cuenta con un sensor en lugar de los varios disponibles en sistemas de vigilancia avanzados.

En el diseño de la interfaz se ha tenido en cuenta la posibilidad del sistema de interactuar con un sensor y sin él, como un modo de enviar mensajes sin otra información adicional. El diseño de la interfaz puede consultarse en el apartado 5.4.

5.2 Dispositivos hardware para la implementación del sistema

El sistema diseñado consta de una parte de software y otra de hardware. La primera alude tanto al programa para el ordenador como el programa para el microcontrolador. La otra parte representa los elementos hardware necesarios para que el sistema funcione, junto al software. Básicamente son cuatro: el ordenador, el móvil, la placa de evaluación y el sensor. El ordenador y el móvil pueden ser cualquier modelo o tipo, siempre que el computador posea conexión USB. Se obvian por tanto de la explicación detallada de que son objetos los otros dos elementos hardware en la sección actual de la presente memoria.

5.2.1 Arduino™ Uno

La placa de evaluación usada es la Arduino™ Uno. Se ha elegido esta placa por la gran disponibilidad de complementos hardware existentes en el mercado actual para dicha placa. Esto tiene sentido desde el momento en que el tipo de comunicaciones aquí implementado, GSM, necesita de un módulo específico para tal sistema y se puede comprar un módulo que implemente dicha funcionalidad listo para acoplar a la placa de evaluación de Arduino™ conocida como Arduino™ Uno, sin necesidad de soldar o de conectores.

La mencionada placa presenta como características principales:

Imagen 5.3: Placa de evaluación Arduino™ Uno

- Reloj de 16 MHz
- SRAM de 2 KBytes
- EEPROM de 1 KByte
- 14 pines de I/O digitales
- Memoria Flash de 32 KBytes
- Alimentación de 5 voltios
- Microcontrolador ATmega328™

Además incluye una conexión USB que será la que se use en el sistema implementado para hacerle llegar los datos introducidos desde el ordenador al dispositivo –desde este último, la información llegará en formato sms al destinatario del mensaje indicado por el usuario en la interfaz diseñada-.

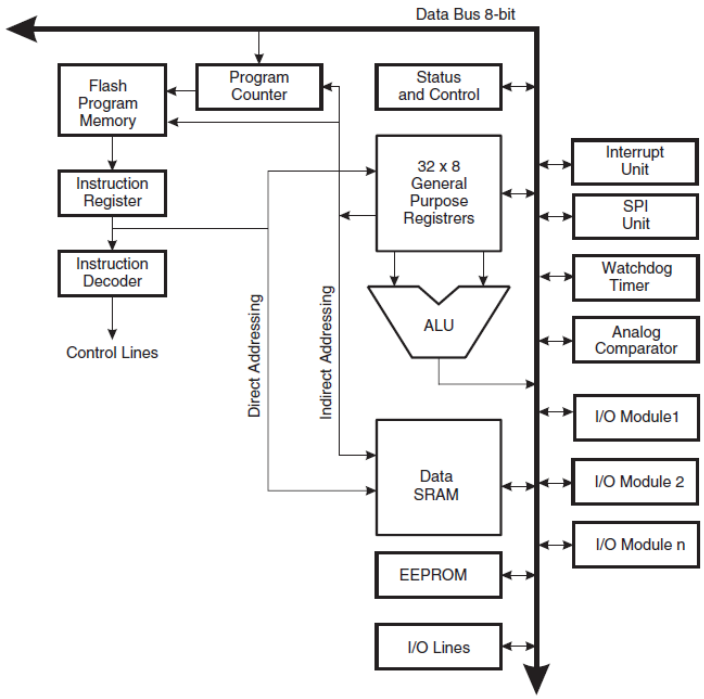

Imagen 5.4: Módulos conectados al bus de datos del microcontrolador

Arduino™ Uno está basado en un micro ATmega328™. Este dispositivo se caracteriza por basarse en una arquitectura RISC y ser un micro de 8 bits. Estas magnitudes son ciertamente reducidas y existen otros micros de mayores prestaciones por similar precio. No obstante, la placa de evaluación que lo contiene –Arduino™ Uno- dispone de gran variedad de dispositivos directamente conectables a ella en el mercado. Uno de estos añadidos comercializados es un sistema de comunicación GSM. Esto ha hecho que, dado que la aplicación diseñada era viable con un micro de sólo 8 bits, se

haya optado por el empleo de esta placa con tal micro. El micro de la placa presenta una arquitectura Harvard, en la que se mantienen memorias separadas para los programas y para datos. La imagen del margen muestra el bus de datos, así como registros, memorias y módulos de entrada/salida. Para la conexión al sensor, uno de los módulos de entrada salida será requerido.

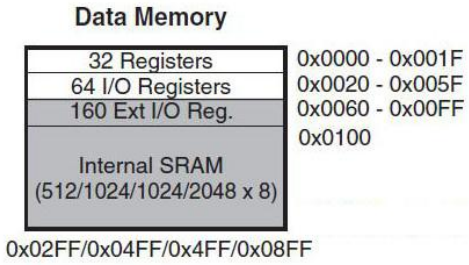

Imagen 5.5: Organización de la memoria de datos

El microcontrolador dispone de una memoria de datos organizada como se indica en la imagen del margen. Esta memoria de datos dispone de varios registros y de memoria SRAM. En la zona inicial de la memoria se encuentran los registros de propósito general, 32 de ellos. En la dirección inferior están los 64 registros de entrada/salida. Después existen registros especiales y finalmente la memoria SRAM, desde la dirección 0x100 en adelante. La imagen 4.20 muestra las direcciones de los 32 registros de propósito general de la CPU. Se extienden desde la dirección 0x0000 hasta la dirección 0x001F.

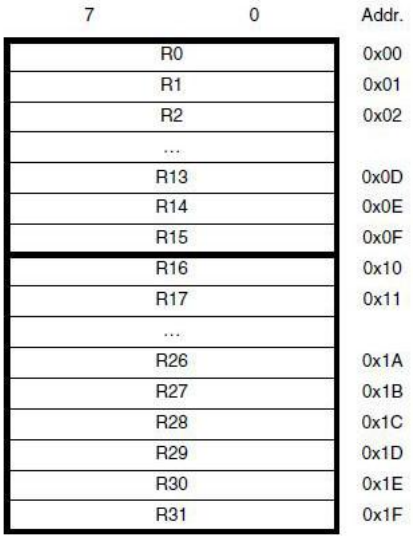

Imagen 5.6: Direcciones de registros de propósito general de CPU

La imagen siguiente recoge el esquemático de la placa de evaluación Arduino™ Uno. Se aprecian en ella en los extremos los distintos conectores que incluye dicha placa: para pines, un USB, un conector para alimentación. Los pines permiten la adición de diversos módulos hardware para ser controlados por el microcontrolador de la placa sin necesidad de realizar soldaduras, como es el caso del módulo GSM Shield.

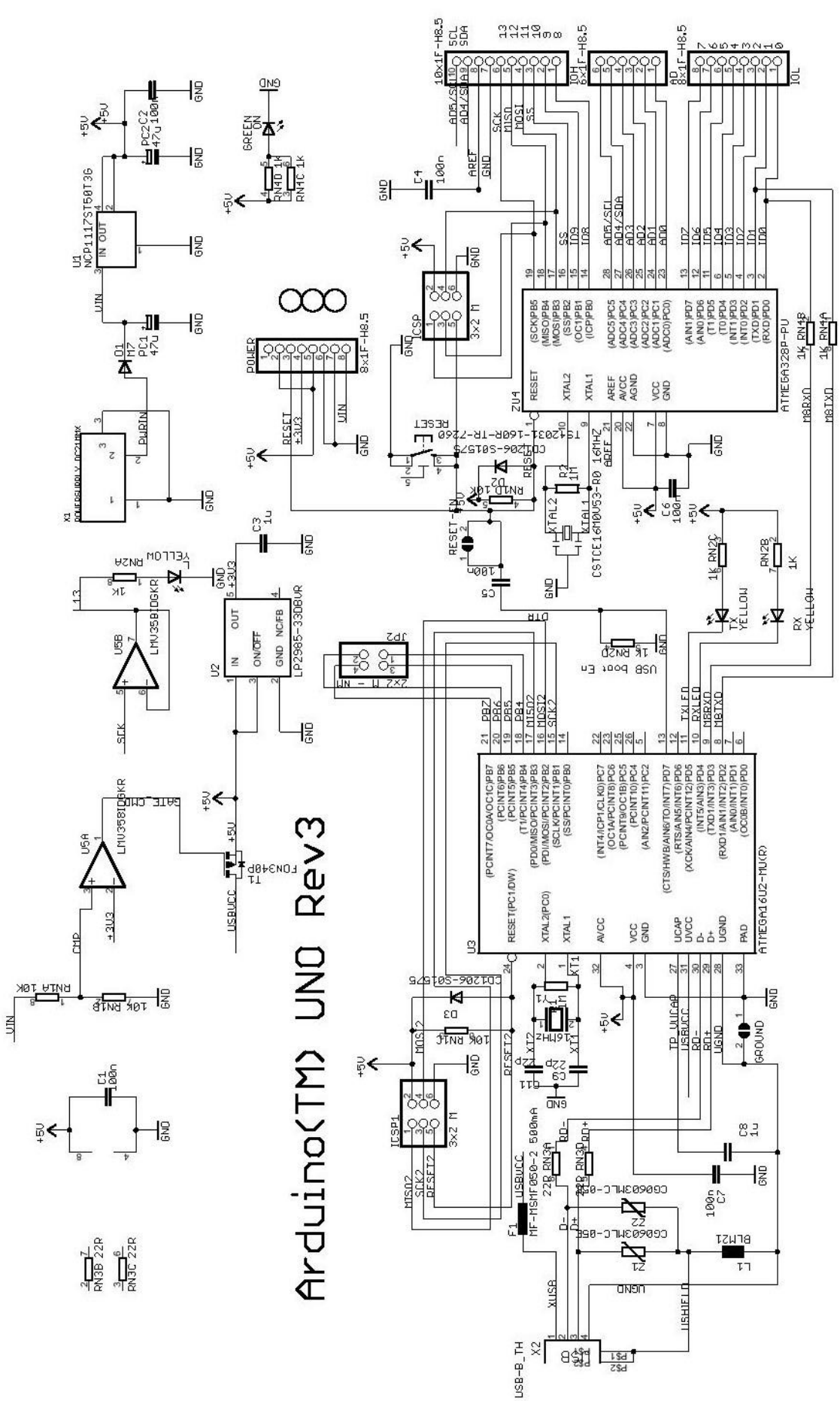

Imagen 5.7: Esquemático de placa Arduino™ Uno

Se aprecian en la imagen del esquemático de la placa Arduino™ Uno varios elementos importantes:

- La alimentación de la placa a 5V.
- Una conexión para USB.
- Canales ADC –convertidores analógico/digital-.
- Canales para interrupciones, concretamente 6.
- Un pin para transmitir en serie –TX-.
- Un pin para recibir en serie –RX-.

Para las comunicaciones serie, el USB es necesario. Los datos son enviados desde el ordenador hasta el microcontrolador, desde donde viajan hasta el terminal móvil elegido por el usuario como destinatario. Se ha elegido la placa Arduino™ Uno porque permite incorporar un módulo para GSM/GPRS mediante el que ejecutar un programa que haga uso de tales tecnologías de comunicación.

5.2.1.1 Lenguaje de programación para Arduino™ Uno

Según se ha expuesto, el sistema diseñado necesita de dos programaciones distintas pero dependientes entre sí: el programa del ordenador y el del microcontrolador. La parte correspondiente al computador se detallará en la subsección 5.4.1.1 y siguientes. En este apartado se describe el lenguaje usado para software de la placa usada para el presente trabajo.

La programación de la placa Arduino™ Uno se basa en un lenguaje que comparte muchas palabras reservadas con otros lenguajes. Su sintaxis es similar a la del lenguaje C. La tabla recoge las principales de estas palabras [4]:

if	while	switch case	char	float	!=

if/else	for	void	int	double	&&
return	break	boolean	long	==	unsigned

Tabla 5.1: Principales palabras reservadas del lenguaje para Arduino™

Comparando con otros lenguajes, las palabras reservadas de la tabla anterior son comunes en la mayoría de casos, como por ejemplo en Java o en C#, entre otros lenguajes más.

Un aspecto importante y que distingue al lenguaje para Arduino™ del lenguaje C es la existencia de un tipo para las cadenas, denominado *String*. Este tipo facilita la labor de trabajar con conjuntos de caracteres, algo necesario en el sistema aquí propuesto al tener que tratar los datos que llegan hasta la placa vía USB desde el ordenador. De otra forma, habrían de utilizarse variables tipo *array* de caracteres para suplantar al tipo String.

Otro aspecto destacable y relacionado con la inclusión del tipo *String* es la disponibilidad de varios métodos para convertir de números a cadenas y viceversa de forma intuitiva. Así por ejemplo, para convertir de entero –int- a cadena –string- basta con la llamada a la función String(...). Ésta toma como argumento el entero y devuelve su conversión en cadena. Con puntos suspensivos en prototipos de funciones se indica que la función en cuestión toma uno o más parámetros como argumentos.

Aunque la sintaxis del lenguaje para la placa Arduino™ guarda cierta similitud con el lenguaje C, la estructura de cualquier programa es distinta. Un programa para la placa usada para el sistema diseñado consta de cuatro partes diferenciadas, dos de las cuales no están relacionadas con un programa usual en lenguaje C. Una primera zona del programa se usa para definiciones de

variables; esto es común en C. Después hay dos zonas propias de Arduino™: un parte para inicialización de variables y llamadas a funciones que se ejecutará al iniciar el programa, y una zona de repetición que es un método de nombre *loop*. La cuarta zona es la última, final del fichero, donde pueden definirse funciones auxiliares por parte del usuario y que es reconocible en otros lenguajes de programación.

```
//Zona de definición de variables
//Zona de inicialización
void setup(){
//… }
//Zona de repetición
void loop(){
 //…}
//Zona para funciones auxiliares
```

Imagen 5.8: Zonas de un programa para Arduino™

En las subsecciones que siguen se detallan las funciones del lenguaje que han sido usadas en la construcción del programa de la placa del trabajo.

5.2.1.2 Conversión de tipos

La conversión de tipos es similar a la de otros lenguajes como C, exceptuando el tratamiento de cadenas y sus conversiones. En el primer caso, y de forma general, la conversión se realiza con un *cásting* o "conversión de tipos". Esto es; incluyendo entre paréntesis el tipo a conseguir con el proceso.

```
double variableDecimal = 5.2;
int variableEntera = (int) variableDecimal;
```

Imagen 5.9: Código de conversión de tipos

El ejemplo de la imagen 5.9 es una muestra de un cásting de un tipo decimal doble –double- a entero –int-. Ejecutadas las líneas de la tabla, la variable variableEntera tendría valor de 5, en lugar de 5,2.

La conversión de enteros a cadenas se indicó en la sección previa. Basta con llamar a la función String(…). Como ejemplo se recoge una conversión en la imagen inferior.

```
Int variableEntera = 44;
String variableCadena = String(variableEntera);
```

Imagen 5.10: Código de conversión de cadenas

Ejecutadas las líneas de la tabla, la variable variableCadena tiene el valor de cadena "44".

5.2.1.3 Control de los pines de entrada/salida

La placa de evaluación dispone de hasta 14 pines de entrada /salida. La programación de estos pines se basa en dos comandos principalmente. Uno de ellos permite programar si se trata de una salida o de una entrada el pin en cuestión, y el otro comando es utilizado para enviar un uno lógico o un cero lógico por el pin.

En el primer caso, se trata del comando pinMode(…). Para la enviar uno o cero es la orden digitalWrite(…). Un ejemplo de programación:

```
pinMode(8, INPUT); //Se programa el pin 8 como entrada
pinMode(4, OUTPUT); //Se programa el pin 4 como salida
digitalWrite(4, HIGH); //Se envía un uno lógico por el pin 4
digitalWrite(4, LOW); //Se envía un cero lógico por el pin 4
```

Imagen 5.11: Código de control de pines en Arduino™

En el ejemplo se ha enviado un uno y un cero lógico por el pin 4 porque este pin ha sido programado como una salida –output-. Al haber especificado su función como salida, pueden enviarse unos y ceros por ese mismo pin. Las dos barras inclinadas a la derecha (//) indican que el resto de la línea es un comentario; se trata dos caracteres que juntos, en Arduino™ y otros lenguajes

de programación como C o Java, indican que lo que sigue a dichas barras en la misma línea del marcador será ajeno al procesado del programa.

5.2.1.4 Comunicaciones serie

En el sistema diseñado se usa la comunicación serie para enviar la información desde el computador hasta la placa de evaluación. La conexión se realiza por medio de un USB. En este sistema se ha usado únicamente la función de recepción de datos. Esto es así porque la funcionalidad del sistema lo requiere: el microcontrolador recibe los datos desde el ordenador mediante ese modo de comunicación y los reenvía por GSM, sin cables, sin responder al ordenador de su envío o con otra información de confirmación.

Las funciones para recibir usadas en el trabajo son dos principalmente: una para leer los caracteres y otra para comprobar si está usándose el canal serie. Antes de estar en condiciones de recibir información es necesario especificar la velocidad del canal serie. Esto se realiza con la función begin(...). Aunque se ha diseñado al interfaz para la elección de velocidad, dado que es usual la comunicación a 9600 baudios, internamente en el programa diseñado para el microcontrolador se ha fijado a esa cantidad la velocidad. Además se usa otra función para imprimir información por el monitor serie incluido en el entorno de programación de Arduino™ al que accede con la secuencia de teclas Crtl + Máyusculas + M, desde la ventana principal del mencionado entorno de programación, disponible en [4].

Para comprobar la disponibilidad del canal serie se usa la función available(). Ésta devuelve el número de caracteres que hay en el canal serie. Si devuelve 0 indica que no ha sido recibido ningún carácter por el canal serie.

Por tanto, cualquier valor mayor que cero es un aviso de que la información está llegando por el canal para ser leída.

```
char caracter;
Serial.begin(9600); //Se programa velocidad de 9600 baudios para el canal serie
if(Serial.available()>0){ //Se comprueba que el canal serie esté recibiendo datos
   caracter = (char) Serial.read(); //Se guarda el carácter recibido en la variable "caracter"
   Serial.print("Se ha recibido el caracter "); //Se imprime frase por el monitor serie
   Serial.println(caracter); //Se Imprime carácter recibido en el monitor serie en nueva línea
}
```

Imagen 5.12: Código de recepción de caracteres

Ocurre que para distinguir que estas funciones son del módulo de comunicaciones serie, en el lenguaje de Arduino™ ha de anteponerse la palabra Serial.

5.2.1.5 Comunicaciones GSM

Se han usado sólo los comandos para enviar datos mediante GSM. El módulo de comunicaciones GSM incorporado a la placa ha de ser programado con funciones de la biblioteca GSM.h que se ofrece con el dispositivo. La biblioteca mencionada incluye numerosas funciones que se encuentran clasificadas en diez clases por su funcionalidad, según [10]:

- Clase GSM: incluye un constructor y dos funciones básicas para conectar y desconectar.

- Clase GSMVoiceCall: permite realizar llamadas de voz. Incluye funciones como retrieveCallingNumber() para guardar el número de teléfono, y voiceCall(...) para realizar una llamada al número de teléfono indicado como argumento, entre otras funciones.

- Clase GSM_SMS: permite enviar mensajes sms. Es la clase de biblioteca GSM.h que se ha usado para el sistema diseñado. Incluye funciones relativas al envío de sms. Se verán con más detalle posteriormente.

- Clase GSMScanner: sus funciones se usan para comprobar el estado de las conexiones. Una de ellas, getSignalStrength() devuelve un valor 99 si no hay señal y entre 0 y 31 cuando se detecta señal del módem en una escala que para el valor 31 implica que la señal supera los 51 dBm.

- Clase GRPS: incluye la función attachGPRS(…) que es requerida para establecer conexión con el punto APN indicado por parámetros.

- Clase GSMServer: esta clase facilita la creación de servidores para actuar como tales; pudiendo enviar información y recibir información. Algunas de sus funciones son beginWrite() para indicar el comienzo de una comunicación con un cliente, write(…) para enviar datos al mismo y endWrite() para finalizar la conexión establecida con beginWrite().

- Clase GSMBand: esta clase proporciona determinar la banda de frecuencia del módem. Entre las funciones incluidas están begin(), que devuelve 1 cuando el módem está activo y getBand(), que informa sobre la banda de frecuencia del módem. Las principales frecuencias para 3G se recogen en la tabla inferior, tomadas de [11].

España	3G 2100	Japón	3G 1700/2100
México	3G 1700/2100, 3G 850/1900	Rusia	3G 2100
Brasil	3G 1900/2100, 3G 850	Reino Unido	3G 2100
Perú	3G 850/1900	Alemania	3G 2100
Chile	3G 850, 3G 1900, 3G 1700/2100	Italia	3G 2100
India	3G 2100 BSNL / MTNL	Estados Unidos	3G 850/1900, 3G 1700/2100
Sudáfrica	3G 900/2100	China	3G 2100

Tabla 5.2: Frecuencias para 3G por países

- Clase GSMModem: incluye un constructor para el módem. La función begin() de esta clase devuelve el estado del módem. La función getIMEI() permite obtener el número IMEI del módem. Este número identifica al dispositivo de

entre todos los conectados a la red.

- **Clase GSMPIN:** las funciones de esta clase incorporan dos para comprobar la tarjeta SIM; checkPIN(...) comprueba el número *pin* y checkPUK(...) verifica el número *puk*. En ambos casos se devuelve 0 si la comprobación es positiva y -1 en caso contrario.

- **Clase GSMClient:** esta clase es requerida para peticiones GPRS. Algunas de sus funciones son connect(...) para conectar con el puerto y la dirección IP pasadas por parámetros; write(...) para enviar información al servidor; stop() para cerrar la conexión establecida.

5.2.2 Sensor de ultrasonidos

Para el correcto funcionamiento del sistema en dos de los cuatro subcasos programados es necesario tener en cuenta un sensor de proximidad. Estos subcasos, que en siguientes subsecciones se detallarán, son: el envío de un mensaje sin cifrar, el envío de un mensaje cifrado, el envío de un mensaje sin cifrar cuando se detecte un objeto desde el sensor, y envío de un mensaje cifrado cuando se detecte algo desde el sensor. Entre los varios disponibles en el mercado, se ha elegido el sensor HC-SR04 [17]. Este sensor se basa en la emisión y recepción de ultrasonidos. Siendo conocida la velocidad de propagación del sonido en el aire, se llega a calcular la distancia a la que se detecta el obstáculo teniendo en cuenta el lapso de tiempo entre el envío y recepción de la señal y la velocidad del sonido. Como terminales incorpora cuatro patillas que se aprecian en la imagen del margen tomada de [18]. La primera para

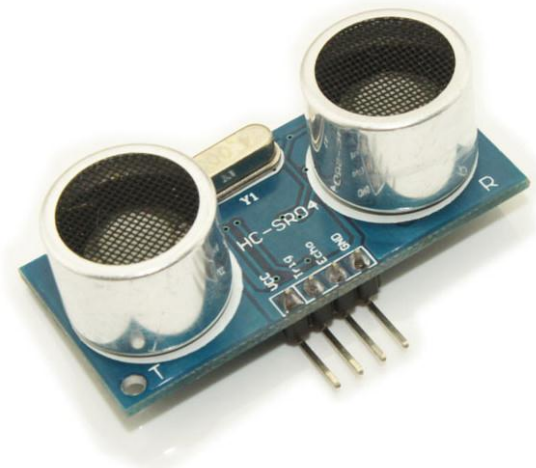

Imagen 5.13: Sensor de proximidad HC-SR04

alimentación a 5V, la segunda para envío de la señal, la tercera para retransmisión de la señal rebotada y la cuarta para la tierra. En la sección 5.8 de la presente memoria se detalla el conexionado con la placa Arduino™ Uno. Se relatan algunas de las características del sensor según [17].

Frecuencia	40 Hz
Ángulo de trabajo	15 grados
Mayor distancia detectable	400 cm
Menor distancia detectable	2 cm
Alimentación	5 V

Tabla 5.3: Principales características de sensor HC-SR04

5.3 Cifrados

Desde la antigüedad se ha recurrido a cifrados como una forma de asegurar la confidencialidad de la información. Métodos como el cifrado César, la sustitución homofónica o la sustitución polialfabética han sido usados con similares propósitos: servir al emisor para hacer llegar un mensaje a un destinatario o destinatarios concretos, negando a cualquier otro la información enviada. Históricamente, la criptografía ha seguido una evolución paralela a la de los avances en las matemáticas, con métodos de cifrado con una complejidad cada vez mayor, a la par que la madurez de la Matemática.

Según [8], las características de la criptografía son pueden resumirse en los puntos siguientes:

- Autentificación: Comprobación de la identidad del emisor.
- Integridad: El mensaje transmitido ha de llegar al o a los destinarios en su completitud; sin errores ni mermas.
- No repudio: Evitar el rechazo de un envío de información.

- Confidencialidad: Se pretende que la información sólo sea inteligible al o a los destinatarios.

5.3.1 Cifrado afín

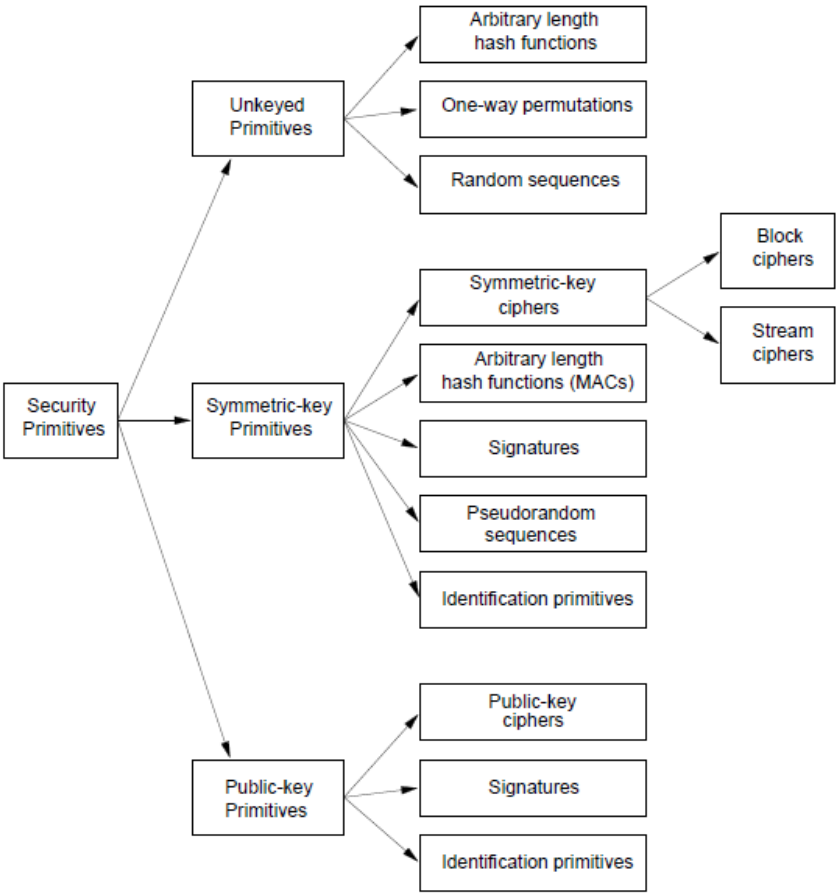

Imagen 5.14: Clasificación de métodos criptográficos

De entre los posibles cifrados, se ha elegido el cifrado afín por permitir una gran flexibilidad en la configuración del cifrado por medio de los posibles valores para la clave de cifrado, así como por necesitar una potencia de cálculo reducida.

La clave de cifrado consta de un par de números enteros, j y k, aunque también es necesario conocer la cardinalidad del alfabeto usado -C-. En el caso del español, se tomará C = 27. La clave puede ser cualquier par (j,k) donde {j,k} varía en los enteros desde 0 hasta C, y ha de satisfacer para j: mcd(j,C)=1. El cifrado afín se realiza según:

$$\text{Afín}(j,k,n) = (jn + k) \text{ módulo}(C)$$

El número n se trata de la posición ordinal en el alfabeto de un carácter. El parámetro j se conoce como *factor de decimación* y k como *factor de*

desplazamiento. Este cifrado se realiza carácter a carácter. Es importante remarcar las limitaciones en la elección de los valores de las claves j y k: ambos han de ser un número entero comprendido entre 1 y C, y j además, ha de cumplir que mcd(j,C)=1. Esto limita bastante la cardinalidad de los enteros que satisfacen las restricciones impuestas para j y para k. Concretamente, las restricciones para j son cumplidas con los enteros {1, 2, 4, 5, 7, 8, 10, 11, 13, 14, 16, 17, 19, 20, 22, 23, 25, 26}. Los valores de k son menos restrictivos, pudiendo ser cualquier entero entre 1 y 27, en el caso de C=27; caso que ocurre para el alfabeto español.

5.4 Interfaz del sistema

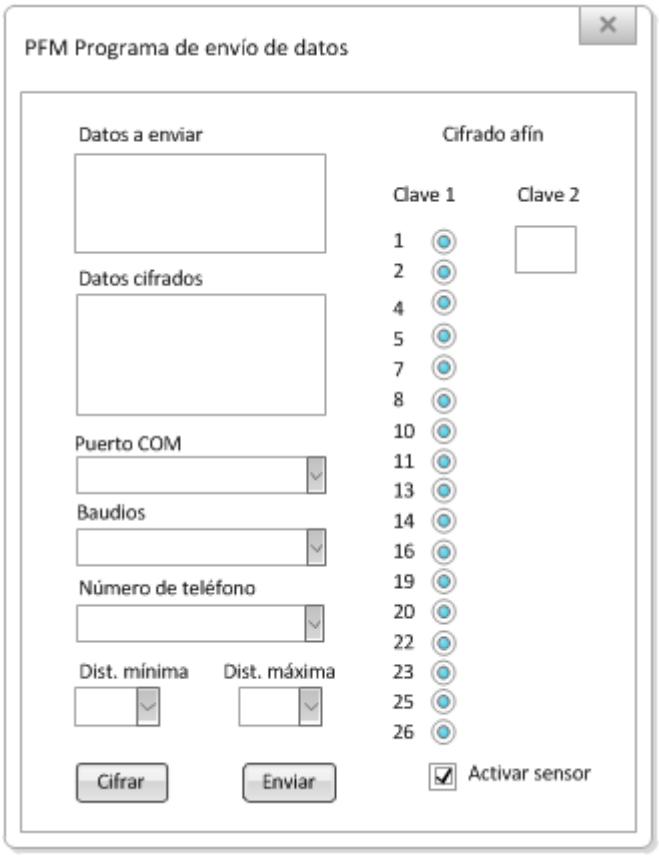

Imagen 5.15: Prototipo de interfaz del sistema diseñado

Se muestra en el margen el prototipo de la interfaz creada empleado C#. La interfaz es el medio a través del cual se hacen llegar los argumentos a la función Afín(DatosAEnviar, Clave1, Clave2). También se ha de pasar el argumento del número de teléfono porque de otra forma, el mensaje no tendría destinatario. Así, las dos cajas de texto de parte izquierda superior son de los datos; la segunda será rellenada por el propio programa después de pulsar el botón "Cifrar". La primera caja es donde ha de teclearse el texto a

enviar cifrado. En la caja inferior se ha de introducir el número de teléfono. Las claves son especificadas en la sección derecha de la interfaz. Para asegurar que se cumplen las restricciones expuestas en el apartado previo sobre las claves, y dado que la primera clave era que estaba sujeta a mayores restricciones, se ha decidido mostrar los posibles valores de la clave primera como botones tipo radio, de los que únicamente se podrá marcar uno a la vez. Finalmente, la caja de texto junto a los botones de radio en la derecha es para indicar la clave segunda. En este caso se ha optado por no especificar todos los posibles valores, ya que para esta clave serían 27; casi el doble de los posibles para la clave primera.

Imagen 5.16: Interfaz del sistema diseñado

El prototipo mostrado se ha diseñado para posibilitar una manipulación sencilla y lo más clara posible; se ha incluido un breve texto indicativo de cada elemento de la interfaz, como el puerto, el cifrado, etc. La interfaz permite distinguir claramente entre los dos procesos que se llevan a cabo para la exitosa consecución del programa: el cifrado de los datos y el envío de los datos. Para lo primero se han de usar los elementos de la mitad derecha –claves- y los elementos de la mitad superior de la interfaz –textos-. El

envío necesita de los elementos de la mitad inferior de la interfaz creada. En este punto hay que distinguir entre los dos envíos que se suceden al ejecutar el código: el envío de los datos cifrados hasta el microcontrolador y el envío de dichos datos desde el micro hasta el teléfono. Para ello hay que indicar el puerto y velocidad de envío así como el número de teléfono que será el destino del texto cifrado.

Con los datos indicados en la interfaz se genera una trama. Con el término "trama" se alude al tipo y la estructura de los datos proporcionados a través de la interfaz y que serán enviados con un orden y una estructura concreta, de forma similar al de una trama propia de cualquier protocolo. La información enviada carece de un campo "comando" o "destino/origen". Sin embargo, cumple con el propósito propuesto, que no es otro que el de mostrar una implementación con una de las tecnologías expuestas en las secciones dedicadas a sistema de comunicaciones inalámbricos.

5.4.1 Módulos de la interfaz

La construcción de la trama de una forma amigable para el usuario ha sido tenida en cuenta para la disposición de los componentes de la interfaz. Se alude con módulo a cada una de los elementos gráficos o agrupaciones de elementos gráficos que han sido incluidos en la interfaz para facilitar una tarea al usuario. La imagen del margen muestra los cinco módulos que han sido necesarios. Para acentuar cada uno de estos módulos, han sido marcados con diferentes colores: naranja para el módulo del campo *mensaje*; rojo para cifrado del mensaje si se desea cifrar; amarillo para el campo *destino*, verde para la programación los parámetros de la comunicación serie; azul para definir el intervalo de distancias analizas por el sensor y que permitirá construir

el campo *información del sensor*. Entre el módulo marcado en naranja y el marcado en amarillo se encuentra un elemento gráfico que simplemente mostrará el texto cifrado –si es que se ha elegido la opción de cifrar el mensaje-. Sobre la elección de la velocidad en baudios, aunque en el programa del ordenador se ha creado un elemento gráfico con varias velocidades, en el código del programa del microcontrolador se ha fijado la velocidad a 9600.

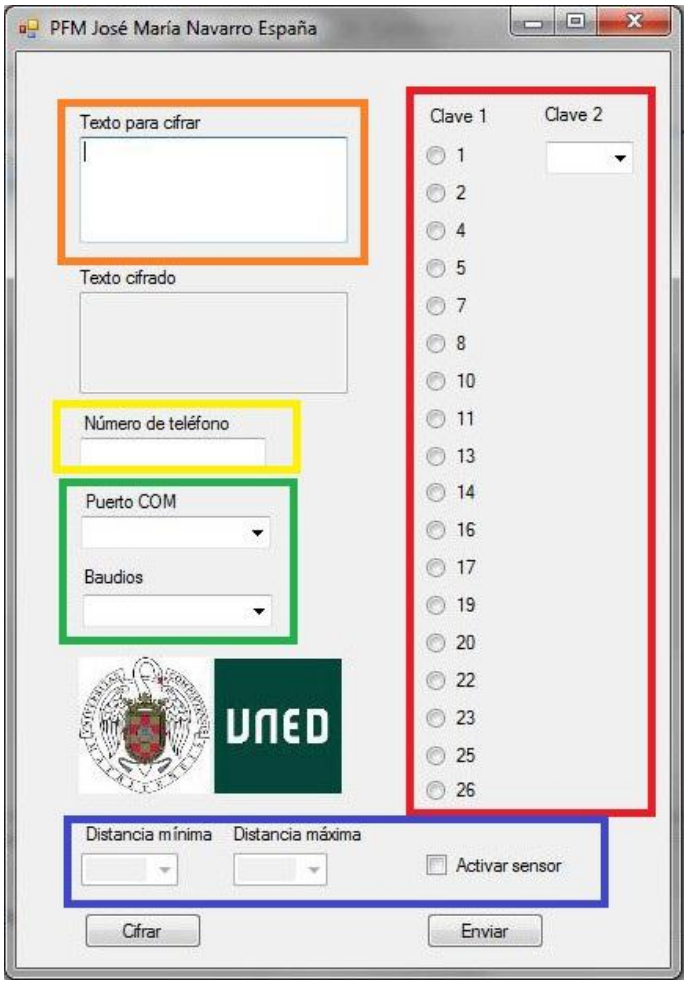

Imagen 5.17: Módulos de la interfaz del sistema diseñado

5.4.1.1 Cajas de texto

Para introducir el texto se han usado objetos tipo textBox. Estos elementos permiten leer los caracteres en ellos tecleados con el comando Text que se muestra, para un ejemplo de una variable llamada "textBox1":

```
private System.Windows.Forms.TextBox textBox1; //Código generado por Visual Studio™
String texto = textBox1.Text; //Se guardan los caracteres en forma de cadena (String)
```

Imagen 5.18: Código de caja de texto

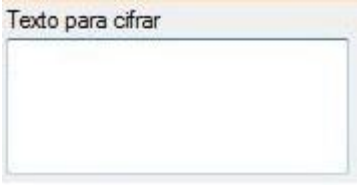

Imagen 5.19: Caja de texto de interfaz

Han sido empleados para el texto del mensaje, como se muestra en la imagen del margen, así como

para indicar el número de teléfono destino.

5.4.1.2 Selectores de opciones fijas

Se alude con este nombre a los objetos denominados en C# como *comboBox*. Se han empleado en la interfaz diseñada para la elección del puerto COM, de la velocidad –aunque como ya se comentó, internamente este campo está fijado a 9600 baudios-, la clave 2 del cifrado así como para los límites inferior y superior del intervalo de distancias que se considerará por el sensor. Se muestra como ejemplo el tratamiento de la clave 2:

```
private System.Windows.Forms.ComboBox comboBox1;
String clave2 = comboBox1.Text;
```

Imagen 5.20: Código de *comboBox*

Imagen 5.21: *ComboBox* para distancias

Este elemento se ha usado porque asegura que los valores que el usuario pueda introducir en los respectivos campos en los que son empleados se restringen a un conjunto de valores que han sido proporcionados por los comboBox implicados en cada caso. Se evita sí, por ejemplo, que el usuario pueda teclear un valor negativo para el límite inferior del sensor, lo cual no tendría sentido y de no comprobarse internamente no resultaría en una medición correcta. La imagen del margen recoge una de las aplicaciones de los comboBox: la elección del límite inferior para el intervalo de distancias que será observado para el sensor. Otra aplicación reseñable es la de la elección de la clave 2 para el cifrado afín.

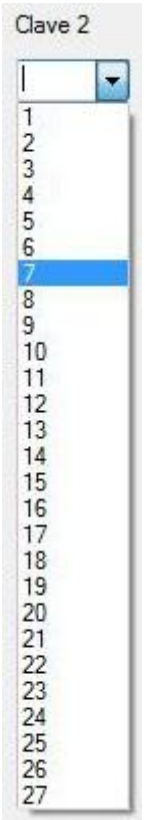

Imagen 5.22: *ComboBox* para clave de cifrado

5.4.1.3 Botón de única elección

Imagen 5.23: *RadioButon* para la clave

También conocido como botón de radio o por su nomenclatura en C# como *radioButton*, este objeto gráfico tiene la particularidad de que sólo permite la elección de uno de estos botones de radio entre un grupo de ellos; es un selector exclusivo. Esta característica ha sido la que ha decidido su uso para programar la clave 1 del algoritmo de cifrado. En la imagen del margen se muestra la estructura de *radioButtons* de la interfaz, de forma parcial –sigue hasta el número 26-, cumpliendo que módulo(número, 27)=1. Siendo 27 el cardinal del alfabeto castellano. Esta restricción sólo es satisfecha por los enteros: son cumplidas con los enteros {1, 2, 4, 5, 7, 8, 10, 11, 13, 14, 16, 17, 19, 20,22, 23, 25, 26}.

5.4.1.4 Casilla marcable

Imagen 5.24: *CheckBox* para sensor

Este tipo de objetos gráficos se conoce como *checkBox* en C#. Al contrario que los elementos tipo radioButton, los checkBox permiten una elección múltiple. Aunque tal caracterísitica puede ser de utilidad en ciertos procesos, para la interfaz aquí diseñada no ha sido necesaria ya que sólo se ha necesitado uno de estos objetos, concretamente para programar el campo *modo* de la trama: si se marca el checkBox se programa el campo modo con valor 1, lo cual permite saber que se está en uno de los dos subcasos en los que la información del sensor es tenida en cuenta; de no marcarse, el campo *modo* valdría 0, con lo que se sabría que se está en uno de los dos subcasos en los que el sensor no es tenido en cuenta.

Aunque podría haberse recurrido a otro elemento gráfico para programar el campo modo, como por ejemplo un radioButton o una caja de

texto que se valoraría con '0' o con '1', para ampliar el abanico de objetos gráficos incluidos en la interfaz creada se ha decidido usar el recurso de un checkBox.

5.5 Tramas del sistema diseñado

La información que se envía al microcontrolador se hace llegar hasta aquél en forma de tramas; su estructura es más simple que las de cualquier trama de cualquier protocolo y su alcance se circunscribe a un dispositivo receptor único, omitiendo campo origen, campo comprobación de integridad tipo CRC y campos que permitan una mayor funcionalidad. No obstante estas tramas permiten cumplir los objetivos propuestos para el presente trabajo, sin excepción alguna.

Se han definido las tramas con una longitud fija para los cuatro primeros campos, dado que esta exigencia era suficiente para el sistema aquí diseñado. El número de campos está fijado en un mínimo de cinco y un máximo de seis. La estructura de la trama es tal que:

Número de teléfono (9 carácteres)	Modo (1 carácter)	Distancia mínima (3 carácteres)	Distancia máxima (3 carácteres)	Mensaje (49 caracteres)	Información del sensor * (variable)

Imagen 5.25: Estructura de la trama creada para el sistema diseñado

Hay que distinguir entre los cuatro casos posibles que permite el sistema diseñado: enviar un mensaje sin cifrado, enviar un mensaje cifrado, enviar un mensaje sin cifrar si se detecta un objeto con el sensor y enviar un mensaje cifrado si se detecta un objeto con el sensor. La estructura de la trama responde a los cuatro subcasos posibles mencionados antes.

Sobre la trama creada hay que señalar que el campo *información del*

sensor tiene una longitud variable ya que contiene la distancia a la se ha detectado un objeto medida desde el sensor, y dicha cantidad oscila entre 20 cm –dos caracteres- y 150 cm –tres caracteres. Además de la distancia incluye la cadena "Detectado algo a " y " cms" de 21 caracteres.

En cualquiera de los subcasos del caso sin sensor, la longitud de la trama es la misma y su estructura es la mostrada arriba exceptuando el campo marcado con un asterisco; ese campo incluye información del sensor que sólo será enviada en formato sms al usuario en caso de usar el sensor y de haber detectado algún objeto dentro del intervalo de distancias definido en la interfaz.

El primer campo guarda el número de teléfono del destinatario del mensaje sms final. El campo modo valdrá '0' en caso de no usar el sensor y '1' en caso contrario. La elección del modo se hace marcando o dejando sin marcar una casilla tipo `checkBox` de la interfaz, como se explicará posteriormente. Los dos campos siguientes almacenan los límites del intervalo de distancia que será tenido en cuenta en el modo '1' para enviar el mensaje o no; es decir, en el modo '1' –se tiene en cuenta el sensor- si la distancia detectada en el instante actual está dentro del intervalo [Distancia mínima, Distancia máxima] el microcontrolador remite el mensaje al destinatario. En caso contrario no lo envía. El quinto campo es el mensaje propiamente dicho. Tiene un límite de 49 caracteres e irá cifrado de haber seleccionado las dos claves del cifrado y haber pulsado el botón de cifrar. El sexto campo, será añadido al mensaje original introducido por el usuario en la caja de texto correspondiente de la interfaz en caso de estar funcionando en modo '1'. Este campo consiste en un cadena en la que se informa de la distancia a la que se ha detectado un objeto.

5.6 Pseudocódigo

El sistema implementado se puede caracterizar, entre otras formas, por sus entradas y salidas así como el formato de las mismas. Dependiendo del subcaso en que se esté ejecutando el sistema diseñado, los pasos a realizar variarán según se muestra en los apartados que se pasan a detallar a continuación. Se mostrará por separado el pseudocódigo de cada uno de los casos, compuestos por subcasos –cuatro en total-.

5.6.1 Pseudocódigo del caso para mensaje sin cifrar y sin sensor

En el caso de enviar un mensaje sin cifrar, el pseudocódigo es:

1) Capturar el texto introducido como mensaje.
2) Capturar el puerto COM por donde se enviará el mensaje.
3) Guardar la velocidad del puerto.
4) Capturar el número de teléfono del destinatario.
5) Cargar con ceros el campo distancia mínima y distancia máxima.
6) Fijar el campo *modo* con valor 0 (valor del checkBox para el sensor).
7) Enviar la trama construida con los datos de los pasos 1) a 6), en el orden: número de teléfono, modo, distancia mínima, distancia máxima, mensaje sin cifrar.

5.6.2 Pseudocódigo del caso para mensaje cifrado y sin sensor

En el caso de enviar el mensaje cifrado, adicionalmente hay que proporcionar los valores de las claves 1 y 2 del cifrado afín. Como entrada, recibirá varios elementos: el número de teléfono al que enviar la información, el modo, la distancia mínima con valor 0, la distancia máxima con valor 0, el texto a enviar cifrado y los dos valores de la clave. Las claves no son enviadas; con estos últimos dos números, se irá cifrando el texto introducido, carácter a

carácter; será este texto cifrado el que se envíe a la placa y desde esta será recibido en el móvil con el número indicado.

1) Capturar el texto introducido como mensaje.
2) Capturar el puerto COM por donde se enviará el mensaje.
3) Guardar la velocidad del puerto.
4) Capturar el número de teléfono del destinatario.
5) Cargar con ceros el campo distancia mínima y distancia máxima.
6) Fijar el campo *modo* con valor 0 (valor del checkBox para el sensor).
7) Guardar los valores de las claves 1 y 2 para el cifrado afín.
8) Realizar el cifrado afín del mensaje con las claves anteriores, según:

   $Afín(Datos, Clave1, Clave2) \Rightarrow DatosCifrados$, donde la variable Datos es la información introducida como texto en la interfaz, y Clave1 y Clave2 son las claves del algoritmo de cifrado afín que se expuso en la sección 5.3.1.
9) Enviar la trama construida con los datos de los pasos 1) a 8), en el orden: número de teléfono, modo, distancia mínima, distancia máxima, mensaje cifrado.

5.6.3 Pseudocódigo del caso para mensaje sin cifrado y con sensor

En el caso de enviar un mensaje sin cifrar en función de la lectura del sensor de proximidad,

1) Capturar el texto introducido como mensaje.
2) Capturar el puerto COM por donde se enviará el mensaje.
3) Guardar la velocidad del puerto.
4) Capturar el número de teléfono del destinatario.
5) Fijar el campo *modo* con valor 1 (valor del checkBox para el sensor).
6) Guardar el valor seleccionado para la distancia mínima de detección.

7) Guardar el valor seleccionado para la distancia máxima de detección.
8) Enviar la trama construida con los datos de los pasos 1) a 7), en el orden: número de teléfono, modo, distancia mínima, distancia máxima, mensaje sin cifrar. Es el microcontrolador de la placa el que, en caso de ser detectado un objeto entre los límites del intervalo de distancias, añada el campo *información del sensor* y reenvíe el mensaje como un texto en modo sms.

5.6.4 Pseudocódigo del caso para mensaje cifrado y con sensor

1) Capturar el texto introducido como mensaje.
2) Capturar el puerto COM por donde se enviará el mensaje.
3) Guardar la velocidad del puerto.
4) Capturar el número de teléfono del destinatario.
5) Fijar el campo *modo* con valor 1 (valor del checkBox para el sensor).
6) Guardar el valor seleccionado para la distancia mínima de detección.
7) Guardar el valor seleccionado para la distancia máxima de detección.
8) Guardar los valores de las claves 1 y 2 para el cifrado afín.
9) Realizar el cifrado afín del mensaje con las claves anteriores, según:

   $Afín(Datos, Clave1, Clave2) \Rightarrow DatosCifrados$, donde la variable Datos es la información introducida como texto en la interfaz, y Clave1 y Clave2 son las claves del algoritmo de cifrado afín que se expuso en la sección precedente.
10) Enviar la trama construida con los datos de los pasos 1) a 9), en el orden: número de teléfono, modo, distancia mínima, distancia máxima, mensaje cifrado. Es el microcontrolador de la placa el que, en caso de ser detectado un objeto entre los límites del intervalo de distancias, añada el campo *información del sensor* y reenvíe el

mensaje como un texto en modo sms.

En los cuatro casos, el orden de rellenado o marcado de los distintos parámetros es irrelevante, siempre y cuando para cada caso concreto de los enumerados se hayan completado todos los valores antes de pulsar el botón "Enviar". Obviamente, el cifrado afín sólo puede realizarse cuando las dos claves han sido elegidas, por lo que antes de pulsar el botón "Cifrado" las dos claves han de haberse seleccionado. El botón "Enviar" debe ser el último botón en pulsar.

5.7 Código del sistema

La parte software del ordenador es tan importante como la contraparte del microcontrolador; sin él, los datos no llegarían al microcontrolador. Los cuatro posibles casos de envíos al microcontrolador se dividen en dos en función de si se tiene en cuenta o no la información captada por el sensor de proximidad. Estos dos casos a su vez, se dividen en dos subcasos según se envíe el mensaje cifrado o sin cifrar. Estas distinciones han sido reflejadas en el programa del computador; se ha definido un campo denominado "modo" que permite distinguir entre los dos subcasos en que hay que tener en cuenta al sensor y los otros dos subcasos en los que el sensor no se tiene en cuenta.

5.7.1 Código del ordenador

Bajo la interfaz del software del ordenador existen varias funciones que son las encargadas de recoger los valores depositados en la interfaz por el usuario para construir la trama. Se destacan dos funciones como principales: la del botón "Cifrar" y la del botón "Enviar". Se describen ambas.

5.7.1.1 Cifrado del mensaje

Para el cifrado del mensaje se usa el cifrado afín. Este cifrado, expuesto en la sección 5.3.1, necesita de dos claves para realizarse. Las líneas de código del botón "Cifrar" se encargan de recoger el texto introducido y los valores elegidos para las dos claves y con estos elementos aplicar el cifrado afín, resultado un texto cifrado.

```
/*Botón "Cifrar"*/
private void button1_Click(object sender, EventArgs e)
{
    texto = textBox1.Text;
    clave1 = getValorClave1();
    clave2 = comboBox1.Text;
  //Se obtiene la cadena cifrada con el algoritmo afín con clave1 y clave2
    cifrado = afin(texto, clave1, clave2);
    textBox2.Enabled = true;
    textBox2.Text = cifrado; //(1)
}
```

Imagen 5.26: Código del método *Enviar*

Se observa en el fragmento del método que el texto cifrado es mostrado en el textBox2 con las líneas (1) y (2), variable que se corresponde con la caja de texto segunda de la interfaz. El verdadero proceso de cifrado se realiza según:

```
cifrado = afin(texto, clave1, clave2);
```

Imagen 5.27: Llamada al método de cifrado afín

Esta línea es una llamada a la función "afin". Su código es:

```
String afin(String tex, String k1, String k2)
{
    int c1 = Convert.ToInt32(k1);
    int c2 = Convert.ToInt32(k2);
    String res = "";
    int tam = tex.Length;
    Char[] aux = new Char[tam];
    Char[] t = tex.ToArray();
    for (int i = 0; i < tam; i++)
    {
        int pos = posicionAfin(t[i]);
        if (pos == 28)
        {
            aux[i] = ' ';
        }
        else if (pos == 29)
        {
            aux[i] = '.';
```

```
                }
                else
                {
                    aux[i] = caracterAfin((pos * c1 + c2) % 27);
                }
            }
            res = new string(aux);
            return res;
        }
```

Imagen 5.28: Código del método de cifrado afín

Obsérvese que los puntos y los espacios son los únicos delimitadores contemplados en el mensaje. Además, estos delimitadores no son cifrados. Se destaca también la línea del código:

```
            aux[i] = caracterAfin((pos * c1 + c2) % 27);
```

Imagen 5.29: Llamada del método *caracterAfin(…)*

Es en esta línea donde se implementa la operación módulo que se describía en el apartado 5.3.1 del cifrado afín. El número de caracteres es de 27 posibles valores. Estos valores corresponden a la posición en el alfabeto de las letras, tanto minúsculas como mayúsculas. Para obtener la posición de cada letra en el alfabeto castellano se ha creado una función auxiliar, con la forma:

```
int posicionAfin(Char caracter)
{
    int res = 1; /*Valor por defecto*/
    /*El punto y el espacio en blanco no se cifran*/
    if (caracter.Equals(' '))
    {
        res = 28;
    }
    else if (caracter.Equals('.'))
    {
        res = 29;
    }
    else
    {
        switch (caracter)
        {
            case 'A':
                res = 1;
                break;
            case 'a':
                res = 1;
                break;
            case 'B':
            // . . .
            //Sigue la función para el resto del alfabeto
```

```
            default:
                res = 27;
                break;
        }
    }
    return res;
}
```

Imagen 5.30: Código del método *posicionAfin(…)*

Hay que destacar la etiqueta default del switch de la imagen anterior: cualquier carácter que pertenezca al alfabeto castellano de letras y que no sea un signo punto o un espacio, será identificado con el valor 27. Esto tendrá su reflejo en el resultado del cifrado afín. Se recomienda por tanto, limitar los caracteres introducidos como texto a las letras del alfabeto sin acentuar ni otros símbolos y al punto y el espacio. Como se aprecia en el fragmento del código mostrado en la imagen 5.3, el lenguaje de los mensajes viene dado por las letras mayúsculas y minúsculas del alfabeto castellano sin acentuar además del punto y el espacio. Los caracteres entre los delimitadores "/*" y "*/" son obviados por el programa. Obsérvese que se permite cifrar tanto letras mayúsculas como minúsculas, pero que el resultado del cifrado son letras mayúsculas en ambos casos.

5.7.1.2 Construcción de la trama

Con los datos introducidos, la función del botón "Enviar" construye la trama justo antes de enviarla. Dado que la implementación realizada no lleva a cabo comprobación de los datos introducidos a través de la interfaz creada, es responsabilidad del usuario completar correctamente la información requerida en el subcaso en que se decida ejecutar el programa de entre los cuatro subcasos posibles: de no completarlos correctamente el programa podría no funcionar correctamente y no avisará de ello. El código del botón "Enviar" es:

```
/*Botón "Enviar"*/
private void button2_Click(object sender, EventArgs e)
{
    SerialPort p;
```

```
    String numPuerto = comboBox2.Text; //(1)
    String enviando = cifrado;
    String sensor = "0";
    String distanciaMaxima = "000";
    String distanciaMinima = "000";
    if (checkBox1.Checked) //(2)
    {
        sensor = "1"; //(3)
        distanciaMaxima = normalizaCadena(comboBox5.Text, 3, '0', 1); //(4)
        distanciaMinima = normalizaCadena(comboBox4.Text, 3, '0', 1); //(5)
    }
    telefono = textBox3.Text; //(6)
    baudios = Convert.ToInt32(comboBox3.Text); //(7)
    if(cifrado==""){ //(8)
        enviando = textBox1.Text; //(9)
    }
    String datos = normalizaCadena(telefono, 9, '0', 1) + sensor + distanciaMinima +
distanciaMaxima + normalizaCadena(enviando, 17, ' ', 2) + '\n'; //(10)
    p = abrirPuerto(numPuerto, baudios); //(11)
    enviarPorPuerto(p, datos); //(12)
    cerrarPuerto(p);
}
```

Imagen 5.31: Código del botón "Enviar"

Se han destacado varias líneas del fragmento de código superior como comentarios. En la línea (1) se guarda el puerto COM en la variable numPuerto. En la línea (9) se recoge el texto introducido por el usuario en la primera caja de texto de la interfaz; será el contenido del sms que se enviará al móvil. El contenido se guarda en otra variable tipo cadena llamada "enviando". De no haber pulsado el botón "Cifrar", el contenido de la variable global cifrado que se guarda de la caja de texto segunda de la interfaz, será la cadena vacía. Esto indica que el usuario no ha querido cifrar el mensaje sms: esta comprobación está en la línea (8). De no haberse ordenado cifrar el mensaje, el valor de la variable local enviando se reemplaza con el contenido de la primera caja de texto; es decir, con el mensaje original sin cifrado alguno.

La línea (2) comprueba si se está en alguno de los dos subcasos en que hay que tener en cuenta la información del sensor. Esto está basado en si se ha marcado o no el checkBox1, en cuyo caso el campo *modo* valdría '1' o '0', respectivamente. La línea (3) fija la variable "sensor" a '1', que es la variable

que representa al campo *modo* en el código del ordenador. De tener el cuenta el sensor, habría que conocer los límites del intervalo de distancias que se pretende analizar con dicho sensor; esto se realiza en las líneas (4) y (5) para el límite superior y el inferior del mencionado intervalo, respectivamente. Obsérvese que se recurre a la una función auxiliar normalizaCadena(…) para conseguir fijar la longitud de ambos campos a tres, como se indicaba en el gráfico de la trama en la sección 5.5 del presente trabajo.

La línea (6) recoge el valor del número del teléfono introducido y la línea (7) el de la velocidad del puerto elegida. Su valor será generalmente de 9600 baudios. La línea (10) construye la trama con los cinco primeros campos; el sexto, si el campo modo –en el código es la variable sensor- vale '1' –indica que se tiene en cuenta al sensor- será añadido por el microcontrolador justo antes de enviar el sms al destinatario. La línea (11) abre el puerto indicado por el usuario. La línea (12) procede al envío de la trama al puerto abierto en (11).

Se comenta brevemente la función auxiliar para fijar el tamaño de las cadenas, denominada normalizaCadena(…). Esta función inserta el carácter pasado por parámetros a la cadena también pasada por parámetros. Tiene dos formas de aplicación: añadiendo tantos caracteres como el pasado al final de la cadena pasada o añadiendo la misma cantidad del carácter indicado al principio de la cadena pasada. Esta distinción en la posición del relleno se consigue con el cuarto parámetro de la función: de valer 1, el relleno se antepone a la cadena original; de valer 2, el relleno se pospone a la cadena.

```
String normalizaCadena(String cadena, int longitud, Char caracter, int posicion)
{
    String res = cadena;
    String relleno = "";
    int tam = cadena.Length;
    if (cadena.Length != longitud)
    {
```

```
        int dif = longitud - tam;
        while (dif > 0){
            relleno = relleno + caracter;
            dif--;
        }
    }
    if (posicion == 1)/*El relleno se añade delante de la cadena*/
    {
        res = relleno + res;
    }
    else if (posicion == 2)/*El relleno se añade al final de la cadena*/
    {
        res = res + relleno;
    }
    return res;
}
```

Imagen 5.32: Código de función auxiliar *normalizaCadena(…)*

La función normalizaCadena(…) también se usa para fijar el tamaño del número de teléfono a nueve caracteres. En caso de ser menos los introducidos en la caja de texto de la interfaz para tal efecto dispuesta en ella, se anteponen tantos ceros como sean necesarios haciendo uso de la mencionada función hasta llegar a nueve caracteres numéricos. Esto se hace para mantener fija la longitud de la trama –exceptuando el campo sexto-.

5.7.2 Código de la placa

El software que se carga en el microcontrolador de la placa guarda ciertas similitudes con el del ordenador: ha de distinguir entre los cuatro subcasos posibles haciendo uso del valor del campo *modo*. Para ello, en el código se procura en primer lugar procesar la trama recibida y atender al valor de cada uno de sus campos, prestando especial atención al valor del campo *modo*.

5.7.2.1 Captura de la trama

La trama llega por el cable USB hasta el microcontrolador, funcionando como una comunicación serie. Esto hace imperativo usar las funciones para

comunciones serie de que dispone el lenguaje de programación de la placa de Arduino™. Mediante un evento se realiza la captura de la trama con las funciones mencionadas. Su código se incluye abajo:

```
void serialEvent() {
    while (Serial.available()) {
        char entrada = (char)Serial.read();
        datosRecibidos = datosRecibidos + entrada;
        if (entrada == '\n') {
            datosAlCompleto = true;
        }
    }
}
```

Imagen 5.33: Código de captura de caracteres

Se hace uso de funciones para comunicaciones serie para detectar cuándo llegan los caracteres –llamada a Serial.available()- y para leer el carácter que acaba de ser detectado –llamada a Serial.read()-. Está basado en el ejemplo para serialEvent propuesto en [4].

5.7.2.2 Envío de sms al destinatario

Una vez los distintos campos de la trama han sido leídos, se está en condiciones de enviar el mensaje sms al destinatario. Para ello, se ha de discernir cuál de los subcasos ha sido empleado para el envío de los datos desde el ordenador. Aunque para el software del computador se distingue claramente entre cuatro subcasos –con el sensor y sin cifrar, con el sensor y cifrado, sin sensor y sin cifrar y sin sensor y cifrado- en el software del microcontrolador esta distinción se reduce a la información aportada por el campo *modo*, que sólo permite concretar si se está en alguno de los dos subcasos que tienen en cuenta la información captada por el sensor o si se está en alguno de los dos subcasos que no tienen en cuenta al sensor. Esto se justifica en el hecho de que los subcasos en que el texto del mensaje está

cifrado o sin cifrar no necesitan ser tratados de diferente forma; simplemente son enviados al destinatario indicado, mientras que los subcasos en que se escucha al sensor o no se escucha difieren en qué mensaje hay que remitir al destinatario, puesto que de atender a las mediciones del sensor se adjunta un aviso de detección de un objeto junto al texto introducido por el usuario en la interfaz creada. Se muestra el fragmento del código encargado de la trama.

```
void enviarDatos(String var, int funcion){
    Serial.println("********** ENVIANDO... **********");
    Serial.print("FUNCION: ");
    Serial.println(funcion);
    char numeroTelf[10]; //(1)
    char mensaje[50];  //(2)
    var.substring(0,9).toCharArray(numeroTelf, 10); //(3)
    var.substring(16, 65).toCharArray(mensaje, 50); //(4)
    if(funcion == 0){  //(5)
        sms.beginSMS(numeroTelf); //(6)
        sms.print(mensaje); //(7)
        sms.endSMS();  //(8)
    }
    else if(funcion == 1){  //(9)
        String infoSensor = "Detectado algo a "+String(distanciaCm)+" cms"; //(10)
        sms.beginSMS(numeroTelf); //(11)
        sms.print(mensaje+infoSensor); //(12)
        sms.endSMS(); //(13)
    }
    Serial.println("******** ...FIN DE ENVIO ********");
}
```

Imagen 5.34: Código de función *enviarDatos(…)*

Se han numerado las líneas más destacadas del código para comentarlas. Tanto en la línea (1) como en la (2) se definen dos variables locales en las que guardar el valor del campo del *número de teléfono* y el valor del campo *mensaje*, respectivamente. La función enviarDatos(…) toma como argumentos dos valores: la trama sin procesar que ha sido capturada vía USB y una variable "funcion". Dicha variable se corresponde con el campo modo de la trama y es gracias a la cual se distingue entre los subcasos, de forma similar a como se

procedía en el software del ordenador, como ya se ha expuesto. Quiere esto decir que, aunque se pase por argumentos a enviarDatos(…) la trama sin procesar, previamente a la invocación de la mencionada función ha de haberse analizado el valor del campo *modo*, ya que de otra forma no se podría identificar el caso concreto en el que se está operando. Las líneas (3) y (4) extraen los valores de los campos *número de teléfono* y *mensaje* mediante la función substring(…). Obsérvese que en el primer campo se tratan de los caracteres en posiciones 0 a 8. Ocurre así puesto que substring(…) ha sido definida para el entorno Arduino™ para leer la subcadena [0,9) de manera que la subcadena que se consigue con su invocación es la que va desde la posición 0 hasta la 8. En el otro campo ocurre algo similar por usar la misma función para subcadenas, con lo que se leen las posiciones entre la 16 y la 64.

Las líneas (5) y (9) se encargan de discernir entre los casos sin y con sensor; se comprueba el valor del campo *modo* que se ha pasado como argumento en la variable de entrada "función": un cero índica que se está ante alguno de los subcasos en los que no se atiende al sensor, mientras que un uno es propio de los subcasos en los que sí se escucha las mediciones del sensor.

En las líneas (6) a (8) se procede a remitir la trama al destinatario que ha sido especificado por el usuario con un número de teléfono en la interfaz creada para el software del ordenador. Se trata de los comandos incluidos en el anexo A.1 de la presente memoria. Externa a enviarDatos(…), como variable global del programa para Arduino™, se ha definido la variable sms para hacer uso de los métodos de envío de sms. En la línea (6) con beginSMS(…) se indica el número de teléfono al que se enviará el mensaje. En (7) se remite el mensaje indicado por el usuario y que ha sido recogido en la variable mensaje gracias a la función print(…) que dispone la biblioteca GSM. La línea (8) finaliza el envío.

Algo similar a lo expuesto ocurre con las líneas (10) a (13). Sin embargo hay una diferencia que es la que permite ver que se está en el otro caso, además de por el valor 1 del campo *modo* –línea (9)-. La discrepancia radica en (10): allí se construye una cadena con el texto "Detectado algo a distancia...". Es una información que es proporcionada por el sensor, concretamente la distancia a la que se ha detectado una presencia o movimiento puntual. Se recuerda aquí que para campo *modo* = 1 se está en el caso en el que se tiene en cuenta el sensor –que a su vez puede ser uno de los dos subcasos: con mensaje cifrado o sin cifrar-. En (9) se comprueba que se está en este caso. El hecho de prestar atención a las mediciones del sensor se refleja en la adición al mensaje del usuario de la cadena con información del sensor mencionada, en la que se mostrará la distancia a la que fue detectado el objeto en el instante en que fue percibido por el sensor. Sobre el código de GSM del software del micro, está basado en el ejemplo proporcionado en [10].

5.7.2.3 Rutina del sensor de proximidad

El sistema diseñado incorpora un sensor de proximidad con alcance de hasta 4 metros. No obstante, para el sistema diseñado se ha limitado rango de mediciones a (20,150) centímetros. Dentro de este intervalo, las medidas tomadas por el sensor son consideradas como disparador del envío especificado por el usuario.

```
long rutinaSensorProximidad(){
    long distancia;
    long intervalo;

    digitalWrite(sensorDisp, LOW); //(1)
    delayMicroseconds(10); //(2)
    digitalWrite(sensorDisp, HIGH); //(3)
    delayMicroseconds(10); //(4)
```

```
    digitalWrite(sensorDisp, LOW); //(5)

    intervalo = pulseIn(eco, HIGH); //(6)
    distancia =(long)(intervalo / 58); //(7)

    Serial.print("La distancia detectada es de ");
    Serial.print(distancia);
    Serial.print(" centimetros\n");
    delay(500); //(8)

    return distancia;
}
```

Imagen 5.35: Código del sensor de proximidad

El sensor usado en el sistema diseñado es un sensor de ultrasonidos. El dispositivo elegido presenta cuatro patillas, dos de las cuales son para alimentación y tierra. Las otras dos patillas son las empleadas para la detección propiamente dicha. Por una de estas dos se envía una señal –variable sensorDisp en el programa- y se recoge por la otra patilla en su vuelta hasta el sensor –variable eco-, después de haber rebotado. En la línea (1) se activa a nivel bajo y en (3) a nivel alto el pin que generará la señal que ha de rebotar; se sigue la especificación del fabricante y se mantiene a nivel alto durante 10 microsegundos. La activación a nivel bajo y alto se realiza como se explica en el epígrafe 5.2.1.3 de la presente memoria: con la función digitalWrite(LOW) y digitalWrite(HIGH), respectivamente. Las líneas (2) y (4) con delayMicroseconds(…) construyen el pulso que se emite por el sensor para recibir su rebote; esto es, definen el tiempo por el que se mantiene el nivel alto y bajo, como se recoge en el gráfico del margen, en el que se ha indicado el pulso. La línea (5) completa el pulso con la transición desde el uno lógico hasta el cero lógico, como se aprecia en la

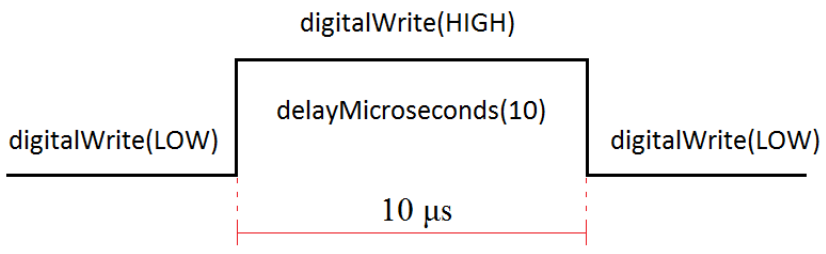

Imagen 5.36: Señal del sensor de proximidad

imagen 5.35 que muestra el código del sensor de proximidad.

Las líneas (6) y (7) son las que permiten concretar el punto detectado –de haberse detectado algo en el intervalo de distancias-. En (6) se hace uso de la función pulseIn(…). Esta función tiene dos prototipos: uno con dos parámetros y otro con tres. Permite calcular el tiempo de un pulso, ya sea alto o bajo inicialmente: tiempo entre nivel alto y bajo o tiempo entre nivel bajo y alto. Aquí se usa el prototipo de dos parámetros; como parámetros toma el número de un pin y el valor de inicio, concretamente, en el sistema diseñado se ha fijado el pin 8 de la placa para *capturar* la señal de rebote, y el nivel alto como nivel inicial, de forma que esta función devuelve el tiempo que ha estado el pin 8 desde el nivel alto hasta el nivel bajo. Más detalles del sensor como tal se exponen en la sección 5.2.2 de la presente memoria. En la línea (7) se convierte la información devuelta por la función anterior a centímetros, que siguiendo las indicaciones del fabricante del sensor [17] aconsejan realizar la conversión dividiendo el valor devuelto por cincuenta y ocho.

La línea (8) es una función que simplemente detiene el flujo del programa por el lapso de tiempo indicado como argumento –en unidades de milisegundos- y es un punto clave en el análisis de la distancia para detección de los obstáculos: de no esperar tiempo alguno, la función realizaría numerosas medidas por la presencia puntual de un objeto delante del sensor, ya que los pulsos que se envían por el sensor necesitan unas decenas de microsegundos para ser generados; con una espera entre cada generación de pulso se evitan las numerosas lecturas del sensor. Se han realizado varias pruebas y se ha fijado el tiempo de espera en el orden de centenas de milisegundos. No obstante, este valor puede modificarse con sólo variar el valor del argumento de delay(…) para ajustarlo a la velocidad esperada para el objeto. De no esperar, en caso de

detectarse algo se enviarían numerosos sms al destinatario; con la inclusión del delay(...) se consigue enviar uno o dos mensajes cuando se pasa frente al sensor a un paso normal. Una posible mejora para futuras versiones del sistema diseñado sería la inclusión de un sensor adicional, de forma que se determine que algo ha sido detectado cuando haya sido percibido por los dos sensores.

5.8 Montaje del prototipo

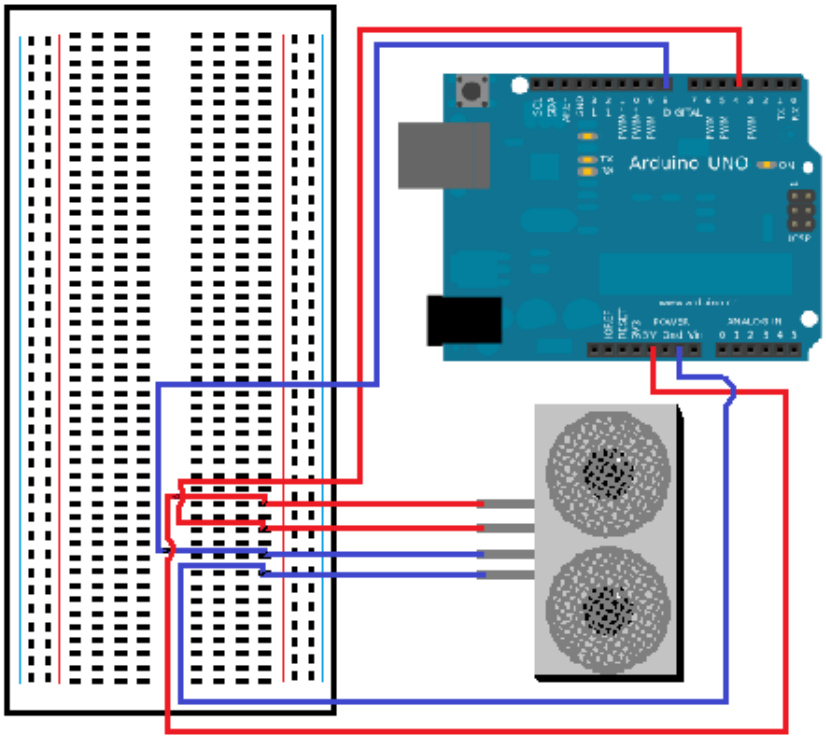

Imagen 5.37: Esquema de conexionado del sistema diseñado

El sistema diseñado puede verse como un prototipo, puesto que aunque incorpora las funcionalidades de un sistema completo, se ha implementado haciendo uso de un *protoboard* o regleta de montaje y cables sin soldar. El esquema del conexionado se muestra en la imagen inferior. Dado que sólo se ha utilizado un sensor, el conexionado sólo se preocupa del funcionamiento de dicho sensor: como se menciona en la sección 5.2.2, el sensor usado, como interfaz física, tiene soldadas cuatro patillas, cada una de las cuales con una función. La de la alimentación a 5V, se conecta con el pin que proporciona igual tensión de Arduino™ Uno. La patilla de tierra se liga a pin de igual propósito de la placa. Para enviar la señal de ultrasonidos por el sensor se conecta la patilla "trig" del

sensor con el pin 4 de la placa. La señal recibida por el sensor, tras rebotar la señal enviada desde la patilla "trig" se hace llegar al pin 8, el cual, como se ha comentado en la subsección previa, recurriendo a la función pulseIn(…) y una conversión posterior permite determinar la distancia del rebote, la cual coincidirá con la distancia del obstáculo que originó tal rebote hasta el sensor.

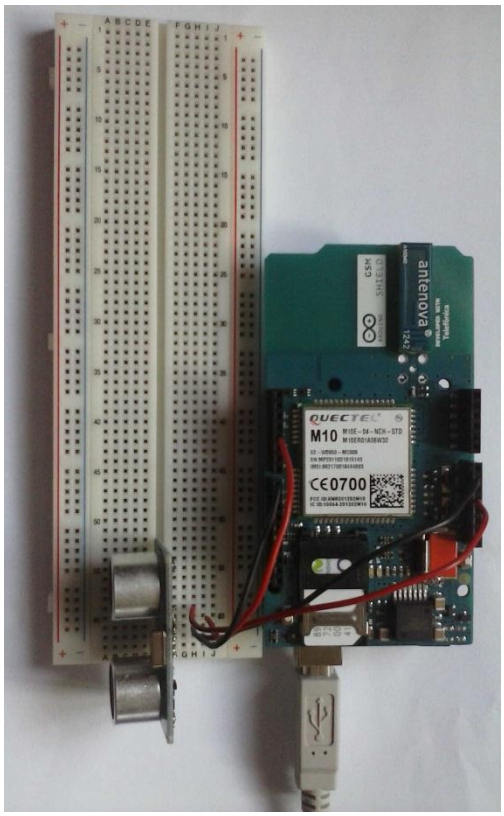

Imagen 5.38: Montaje del sistema sobre *protoboard*

5.9 Manual de usuario

Para el usuario, únicamente se encuentra accesible el programa del ordenador. El software de la placa se ha de cargar previamente. Para ello se puede emplear el entorno de Arduino™ descargable en [4]; una vez abierto en dicho entorno el programa para la placa se guardaría en ella con el botón "Cargar" del entorno. En cuanto al software del ordenador, el usuario puede a través de su interfaz introducir los datos necesarios para el correcto y completo del sistema, como son el número de teléfono destino, el mensaje, etc. Tanto el software del ordenador como el de la placa se encuentran en la carpeta de códigos adjuntada con la presente memoria. En el caso del ordenador, el usuario iniciará el ejecutable para acceder al programa. En este epígrafe se incluyen todos los pasos a seguir para poder realizar la configuración del sistema diseñado.

1º) Abrir el ejecutable del programa.

2º) Introducir el mensaje en la caja de texto correspondiente.

3º) Introducir el número de teléfono del destinatario.

4º) Seleccionar puerto en el que está conectada la placa Arduino™ Uno.

5º) Elegir la velocidad de transmisión.

Si se quisiera cifrar el mensaje a enviar, adicionalmente a los pasos anteriores habrían de realizar los pasos:

6º a) Marcar la clave 1 del cifrado afín.

7º a) Indicar la clave 2 del cifrado afín.

8 aº) Pulsar el botón "Cifrar".

Si se quisiera activar el sensor de proximidad para tener en cuenta sus mediciones antes de enviar el mensaje, adicionalmente habría que:

6 bº) Marcar la casilla de "Activar sensor".

7 bº) Elegir la distancia mínima de detección.

8 bº) Seleccionar la distancia máxima de detección.

Y finalmente, pulsar el botón "Enviar" para transmitir los datos a la placa Arduino™ Uno vía USB.

Para el manual se mostrará el caso más completo: el usuario desea enviar un mensaje cifrado teniendo en cuenta la información del sensor de proximidad antes de remitir dicha información en forma de mensaje sms al destinatario final.

Con el programa abierto, el usuario teclea el mensaje en la caja de texto primera de la interfaz, como se muestra en la imagen 5.39. Este mensaje, además del resto de datos relevantes será enviado a la placa; en ella será procesado por el programa instalado allí y con la información conseguida se enviará un mensaje en formato sms al destinatario del número de teléfono

indicado en el paso siguiente.

Imagen 5.40: Número de teléfono en la interfaz

Imagen 5.39: Mensaje del usuario en la interfaz

Imagen 5.41: Puerto COM en la interfaz

En el segundo paso se ha especificar el número de teléfono que será el destino del mensaje sms final. El usuario lo ha introducir manualmente, como se muestra en la imagen 5.40. Internamente, como se mostró en la sección previa, se comprueba que el valor de este campo sea de longitud nueve, como los números de teléfono habituales. No se verifica si se trata de caracteres numéricos o no. Tal comprobación queda en manos del usuario y es responsabilidad suya introducir el tipo

de datos correcto.

Después hay que seleccionar el número del puerto COM en que se halle conectado la placa de evaluación Arduino™ Uno, como en imagen 5.41, en donde se ha elegido el puerto número 6. Para asegurar que al menos un puerto está disponible, en el checkBox incluido en la interfaz para este valor se han habilitado hasta siete posibles puertos, de entre los que se ha de seleccionar uno. Esta elección, su valor, será usada para hacer llegar la trama hasta el microcontrolador pero no será enviada al destinatario.

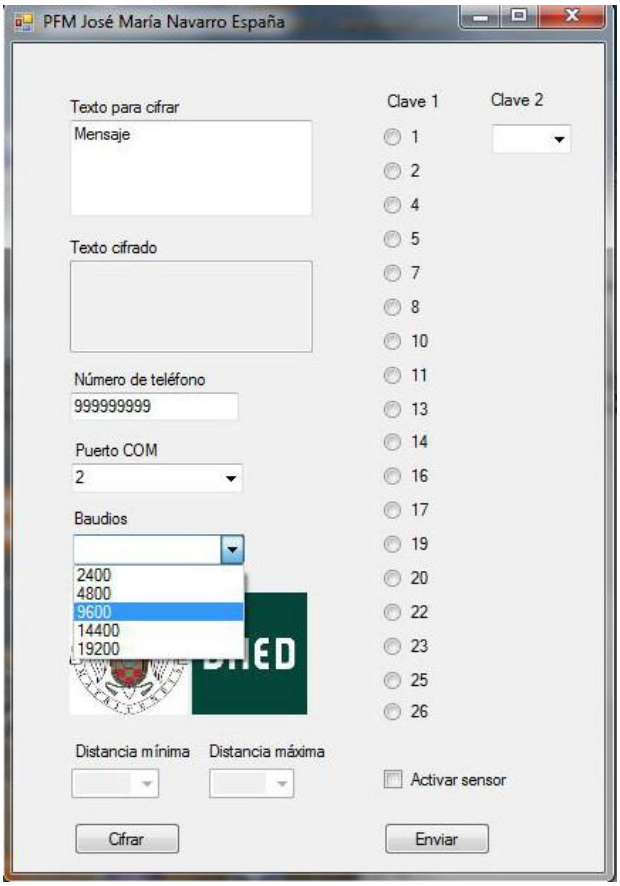

Imagen 5.42: Elección de velocidad en interfaz

La velocidad del puerto es otro factor necesario para una correcta transmisión. Siendo habitual en las transmisiones serie la velocidad de 9600 baudios, aunque se ha implementado un comboBox en la interfaz para permitir una elección, internamente en el micro se ha fijado su valor a 9600 baudios. Si es distinto el valor elegido en la interfaz los caracteres recibidos pueden no ser reconocidos correctamente. No obstante, para mayor flexibilidad en posibles mejoras del sistema aquí diseñado, se ha añadido un comboBox con más valores.

Si se pretende enviar la información cifrada al destinatario es necesario seleccionar las dos claves que son requeridas para llevar a cabo el cifrado afín aquí implementado. Sólo serán reconocidos como caracteres las letras del alfabeto castellano sin acentuar, tanto en minúsculas en como en mayúsculas, el punto y el espacio. Para mostrar el resultado del cifrado afín se ha de pulsar el botón "Cifrar".

Imagen 5.43: Cifrado del mensaje la interfaz

Imagen 5.44: Selección de intervalo de distancia en la interfaz

Para indicar que se desea enviar el mensaje en función de la información captada por el sensor de proximidad, el usuario debe marcar el checkBox identificado como "Activar sensor". Este texto se incluye para dejar claro que será tenido en cuenta el sensor para enviar el mensaje –esté cifrado o no-, dependiendo de que sus mediciones sean válidas para el intervalo definido cuando se fije unos límites con los dos comboBox inferiores de la interfaz para

distancias, ya que por defecto, se ha programado el sensor con los límites 9999999 para el inferior y 0 para el superior, algo que es imposible. Se pretende así evitar que se envíe el mensaje si no se ha especificado un límite inferior y uno superior, como en la imagen inferior, para las distancias detectables. Entre ambos límites se tendrá en cuenta la información del sensor, y si se detecta algo a una distancia dentro del intervalo marcado, se enviará el mensaje sms al destinatario. Para hacer efectivo el envío de la trama al microcontrolador ha de pulsarse el botón "Enviar".

Imagen 5.45: Enviar trama desde la interfaz

6. Coste

La realización del presente trabajo lleva aparejada un coste temporal, por las horas de búsqueda de información, diseño, redacción, implementación y depuración; y también un coste económico por el hardware usado.

En cuanto al coste temporal, éste se ha basado en un esquema de organización propio, en el que se han considerado varios módulos, cada uno de los cuales se compone de ciertas tareas relacionadas entre sí y que por la existencia de dicha relación, se ha convenido en agrupar por bloques. Se ha estructurado el coste temporal en siete bloques. Se detallan cada uno.

Módulo A: identificación de objetivos. Aquí se incluye el trabajo previo a la búsqueda de referencias bibliográficas relacionadas directamente con el trabajo final; este bloque refleja el proceso de análisis necesario para llegar a determinar los objetivos del proyecto. Tiene sentido puesto que el presente proyecto ha sido propuesto por parte del alumno al tutor.

Módulo B: búsqueda de información. Una vez determinados los objetivos que se han de perseguir en el proyecto, se hace necesario realizar la búsqueda bibliográfica para conseguir las referencias que permitan estudiar y reunir los conocimientos para cada uno de los objetivos antes propuestos.

Módulo C: análisis de contenido *teórico*. Dado que el presente proyecto tiene una parte eminentemente de investigación, de exposición de varios sistemas de comunicaciones inalámbricos, además del estudio convencional de cualquier fuente bibliográfica, se ha de prestar atención a los elementos reseñables de los sistemas de comunicación sin cables, haciendo necesario un estudio y análisis, más pausado, si cabe.

Módulo D: diseño del sistema. Con los objetivos fijados y los conocimientos estudiados en las fuentes de información, llega el momento de diseñar el sistema cuya realización es uno de los objetivos propuestos. En el caso del presente proyecto, varios de los objetivos estaban ligados al sistema propiamente dicho, mientras que otros se referían a la profundización de recursos teóricos para investigación sobre las comunicaciones sin cables.

Módulo E: redacción de la memoria. Implementado el sistema, es necesario registrar su diseño y sus fundamentos teóricos en la memoria del proyecto. Además, como varios de los objetivos aludían a profundizar en la

teoría de las comunicaciones sin cables, tiene aún mayor importancia el considerar la redacción de la memoria como un bloque del coste temporal por sí mismo. Con forme se ha avanzado en la implementación se ha hecho necesario ampliar la memoria con los detalles de las nuevas adiciones.

Módulo F: implementación del sistema. Habiéndose realizado el diseño del sistema y su explicación pormenorizada en la memoria, se puede proceder a su implementación. En este caso se ha recurrido a la programación de dos programas software; uno escrito en C# para el ordenador y otro en lenguaje para Arduino™ para la placa de evaluación. Después de la implementación se ha documentar el resultado final conseguido, con lo cual se ha redactar en la memoria la implementación llevada a cabo del sistema previamente diseñado. El tiempo que la redacción del proceso de implementación necesite se incluye en el cómputo del módulo E.

Módulo G: revisión. Aquí se incluye el tiempo necesario para garantizar la corrección de errores, tanto ortográficos como de diseño. Es el módulo que menos tiempo ha requerido.

El coste económico se eleva a 155,11 euros en total, que desglosado es:

Placa de evaluación Arduino™ Uno	25,64 euros	+
Ampliación GSM Shield	94,56 euros	+
Sensor de proximidad HC-SR04.....	7,60 euros	+
Protoboard de montaje	12,00 euros	+
Cable para montaje de circuitos	1,00 euro	+
Adaptador de corriente continua	14,31 euros	+
Total	155,11 euros	

En cuanto al coste temporal, se ha computado el tiempo empleado como unidades de coste temporal o U.C.T. que se han medido en horas (1 U.T.C. = 1 hora).

Identificación de objetivos	30 U.C.T.	+
Búsqueda de información	38 U.C.T.	+
Análisis de contenido *teórico*	51 U.C.T.	+
Diseño del sistema	22 U.C.T.	+
Redacción de memoria	123 U.C.T.	+
Implementación del sistema	21 U.C.T.	+
Revisión	19 U.C.T.	+
Total	304 U.C.T.	

Se observa que, en lo referente al coste temporal, el mayor es el empleado para la redacción, seguido del módulo de análisis referente a la parte de investigación y *teórica* de la memoria. En la siguiente figura se han representado los costes, temporales mediante gráficas de sectores circulares.

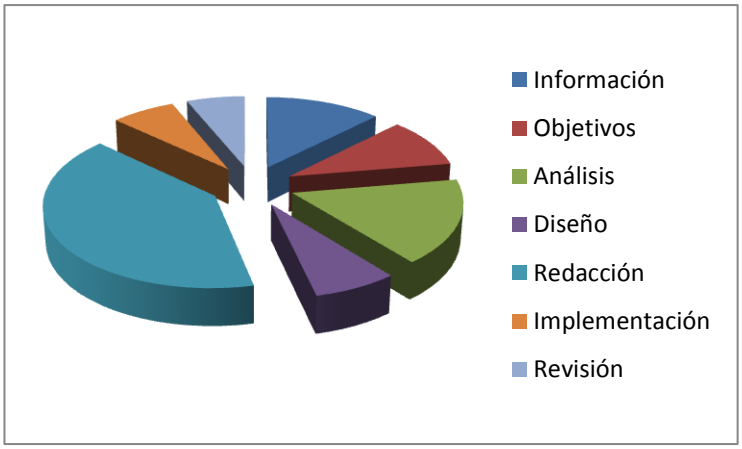

Imagen 6.1: Gráfico de sectores del coste temporal

En cuanto al coste económico de los componentes hardware, con diferencia, el mayor ha sido el de la ampliación para GSM de Arduino™, con un 61% del coste total. La placa Arduino™ Uno ha sido el segundo elemento más caro –un 16,4%-.

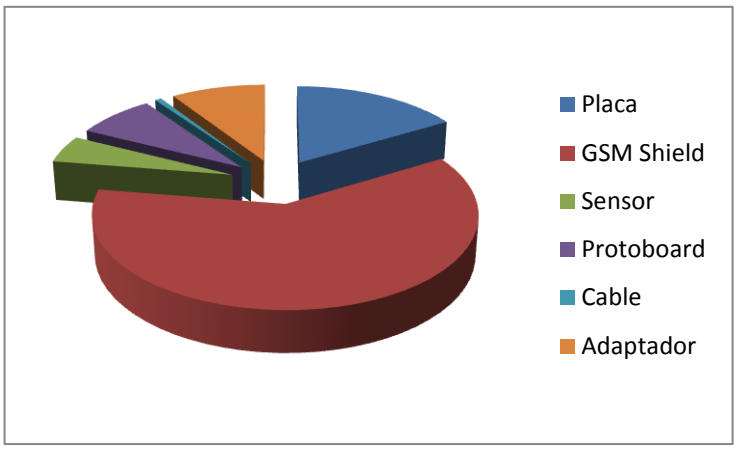

Imagen 6.2: Gráfico de sectores del coste económico

7. Conclusiones

A lo largo de las páginas del presente trabajo, con un gran contenido de investigación, se han detallado aspectos importantes de varias tecnologías de comunicaciones inalámbricas, entre ellas, las comunicaciones GPS, GSM y de redes *wireless*. En el contenido de las páginas de esta memoria se han completado todos y cada uno de los objetivos propuestos al inicio de la misma:

1) Efectivamente, como resultado de la investigación se ha ofrecido una panorámica de las tecnologías que permiten las comunicaciones sin cables actualmente, mencionando dos grupos, uno encabezado por el sistema de comunicación GPS cuya transmisión de datos involucra satélites y un grupo que no usa satélites en el que se han destacado las redes *wireless* como redes wifi y las comunicaciones GSM.

2) Se han detallado los fundamentos tecnológicos y otros más relevantes de cada uno de estos dos grupos de tipos de transmisiones en comunicaciones. En el caso del GPS, se han relatado algunos los conceptos principales de soporte matemático que se hallan bajo dicho sistema, comentando el tipo de

trama enviada en el posicionamiento y sus campos principales. En las comunicaciones GSM se han concretado elementos importantes en ésta, como las bandas de frecuencias empleadas, su arquitectura física y una de sus tramas. Las redes *wireless* han sido enfocadas hacia las redes wifi tan habituales actualmente; se ha partido del protocolo OSI para posteriormente detallar una de las tramas de los varios protocolos involucrados en tales redes, describiendo sus campos.

3) Se ha realizado una implementación de una de las tecnologías sin cables descritas en el segundo de los dos grandes grupos anteriores –sin emplear satélites-; en concreto con comunicaciones GSM. La implementación se ha definido como la interacción de dos bloques: uno puramente software y otro hardware básicamente. Su funcionalidad consiste en permitir la transmisión de unos datos desde un programa en un ordenador hasta un móvil, recurriendo a comunicaciones GSM. En el primer caso se ha creado una interfaz para un programa desde la que especificar los datos que se desean enviar hasta el móvil, así como un programa para el microcontrolador que envía el sms al destinatario indicado. La parte hardware consta de un sensor de proximidad y una placa de evaluación con un microcontrolador específico junto a una ampliación hardware que implemente GSM. Se podría incluir como hardware el ordenador y el móvil como terminal del envío. La aplicación creada es relativamente sencilla pero ha permitido justificar una de las tecnologías sin cables a la par que ha permitido reforzar el trabajo de investigación, especialmente sobre la combinación software-hardware para transmisiones GSM, algo que podría emplearse para futuros trabajos que usen en una comunicación sin cables para envío de datos de tamaño reducido.

4) Se ha usado un microcontrolador para la implementación propuesta

para comunicación GSM. Se decidió emplear esta plataforma hardware dado que los precios de microcontroladores, y por ende de las placas de evaluación, son ciertamente competitivos a día de hoy. Desde el momento en que ambos términos –microcontrolador y placa de evaluación- están relacionados al incluir la placa de evaluación al menos un microcontrolador, ambos términos han sido referidos de forma indiferente en este trabajo.

Por todo esto, puede afirmarse que han sido satisfechos los cuatro objetivos que han sido el motor del presente trabajo. La mencionada utilidad futura de lo expuesto en el presente proyecto podría ser realmente amplia: cualquier aplicación que necesite enviar información sin tener que disponer de un sistema cableado para la transmisión de la misma.

8. Agradecimientos

En primer lugar, debo agradecer al tutor del presente proyecto, D. José Sánchez, el que aceptara mi propuesta de proyecto y el que me permitiera llevar a cabo mis ideas iniciales con total libertad. Su total disposición para dirigir mi propuesta ha sido de un valor enorme. Gracias. Durante el desarrollo del presente proyecto he contado con los comentarios siempre constructivos del tutor D. José Sánchez, algo que también le agradezco; sin el apoyo de D. José el proyecto hubiera sido distinto.

Siempre presentes, mis padres y hermano, me han encomiado siempre a continuar mis estudios, dándome ánimos y apoyándome en todo momento, algo que me ha permitido dedicarme a los estudios de máster decididamente. Mil gracias.

No todos los míos han podido estar acompañándome en estos últimos tiempos. Ellos me han hecho mejor persona. A mis abuelos, por siempre, gracias, de corazón.

9. Bibliografía

[1] *Transmisión de datos y redes de comunicaciones*, 4ª edición, Forouzan

[2] *Global Positioning Systems Directorate Systems Engineering & Integration Interface Specification IS-GPS-200*, US Government, Versión de Septiembre 2012

[3] *Fundamentals of Global Positioning System Receivers: A Software Approach*, J. Bao-Yen

[4] http://www.arduino.cc [2/12/2013]

[5] *GPS Basics: Introduction to GPS(Global Posiotining System)*, Version 1.0, Leica™

[6] *GSM Networks: Protocols, Terminology and Implementation*, G. Heine

[7] http://en.wikipedia.org/wiki/Global_Positioning_System [2/12/2013]

[8] *Handbook of Applied Cryptography*, A. Menezes, P. van Oorschot, S. Vanstone.

[9] *Atmel 8-bit Microcontroller with 4/8/16/32Kbytes In-System Programmable Flash: ATmega48A; ATmega48PA; ATmega88A; ATmega88PA; ATmega168A; ATmega168PA; ATmega328; ATmega328P*, Atmel™

[10] http://arduino.cc/en/Reference/GSM [19/12/2013]

[11] http://www.worldtimezone.com/gsm.html [10/1/2014]

[12] http://www.element14.com [14/2/2014]

[13] http://en.wikipedia.org/wiki/GSM [14/2/2014]

[14] https://micro-manager.org/wiki/ [14/2/2014]

[15] *Muestreo, control y comunicación basados en eventos*, S. Dormido, J. Sánchez, E. Koffman, UNED, FCEIA-UNR-CONICET

[16] *Application of event-based sampling strategies for fusión research*, J. Sánchez, S. Dormido-Canto, J. Vega, N. Duro, R. Dormido, S. Dormido, UNED, EURATOM/CIEMAT FUSION Association

[17] *Ultrasonic Ranging Module HC - SR04*, ElecFreaks

[18] http://www.elecfreaks.com [23/2/2014]

[19] http://www.networksorcery.com/enp/protocol/ip.htm [1/3/2014]

[20] *Overview Of The GSM System and Protocol Architecture*, M. Rahnema

[21] *GSM Pocket Guide Vol. 2*, Wandel & Goltermann

[22] *GSM Mobile Networks: Protocol, Terminology and Implementation*, G. Heine

[23] *SMS the Telecom Source 10 Slide Technology Series*, Telecom Source Consulting Inc

[24] http://www.gsmfavorites.com/documents/sms/packetformat/ [9/3/2014]

[25] *Digital cellular telecommunications system (Phase 2+); Technical realization of the Short Message Service (SMS) Point-to-Point (PP) (GSM 03.40)*, ETSI

[26] *GSM - Architecture, Protocols and Services*, J. Eberspächer, H. Vögel,C. Bettstetter,C. Hartmann

[27] *Global Positioning System (GPS) Time Dissemination for Real-Time Applications*, P. H. Dana

[28] *GPS Essentials of Satellite Navigation (Compendium)*, Empresa ublox

[29] http://www.gps.gov/systems/gps/control/ [28/3/2014]

Anexo A.1

Se incluye en este anexo una breve relación de los principales comandos y variables de Arduino™ empleados para la consecución del presente trabajo. Para mayor información sobre los mismos y sobre otros comandos, ver [4].

Serial	Es la variable del sistema para abrir, cerrar, y en general, trabajar con una conexión serie.
begin(vel)	Método que permite establecer como velocidad de datos medida en baudios el valor "vel". Se invoca con Serial.
beginSMS(núm)	Se incluye como método de la librería GSM. Inicia un envío de sms al número "núm" indicado como destino.
endSMS(núm)	Se incluye como método de la librería GSM. Finaliza un envío de sms al número "núm" indicado como destino.
print(cadena)	En el contexto de la librería GSM, este método permite enviar un sms con los datos indicados en "cadena".
println(cadena)	En el contexto de la librería Serial, este método muestra por pantalla el valor de "cadena" seguido de una línea nueva usando un retorno de carro.
GSM	Librería que se usa para envíos con el GSM Shield acoplable a la placa Arduino™ Uno.

Tabla A1.1: Principales comandos para Arduino™ utilizados