UNIVERSIDAD TECNOLÓGICA ISRAEL

TRABAJO DE TITULACIÓN EN OPCIÓN AL GRADO DE:

INGENIERO EN ELECTRÓNICA DIGITAL Y TELECOMUNICACIONES

TEMA: PROTOTIPO DE SISTEMA DE LOCALIZACIÓN DE UN BUS DE TRANSPORTE URBANO MEDIANTE GPS Y UNA APLICACIÓN ANDROID

AUTOR: ROMMEL UFREDO HURTADO SUÁREZ

TUTORA: Mg. SILVIA DIANA MARTÍNEZ MOSQUERA

QUITO- ECUADOR

AÑO: 2018

DECLARACION DE AUTOR

Yo, ROMMEL UFREDO HURTADO SUÁREZ, declaro que el trabajo aquí escrito es de mi autoría, que no ha sido presentando por ningún grado o alguna calificación profesional y que se han consultado todas las referencias bibliográficas que están incluidas en este documento.

La Universidad Tecnológica Israel del Ecuador, puede hacer uso de los derechos correspondientes de este trabajo, según lo establecido por la Ley de Educación Superior o por lo regido en su reglamento y normativas vigentes.

…….

ROMMEL HURTADO

CERTIFICACIÓN DEL TUTOR.

El suscrito, MSc. SILVIA MARTÍNEZ, en calidad de Docente de la Universidad Tecnológica Israel del Ecuador, certifica que el estudiante ROMMEL UFREDO HURTADO SUÁREZ, realizó el proyecto de titulación previo a la obtención del título de Ingeniería Electrónica Digital y Telecomunicaciones con el tema "PROTOTIPO DE SISTEMA DE LOCALIZACIÓN DE UN BUS DE TRANSPORTE URBANO MEDIANTE GPS Y UNA APLICACIÓN ANDROID", bajo mi dirección, habiendo cumplido satisfactoriamente con las disposiciones reglamentarias establecidas para el efecto del mismo.

MSc. SILVIA MARTÍNEZ
DIRECTORA DE PROYECTO DE TITULACIÓN

AGRADECIMIENTO

Mi agradecimiento se dirige a quien ha forjado mi camino y me ha dirigido por el camino correcto, a Dios, el que en todo momento está conmigo ayudándome a aprender de mis errores y a no cometerlos otra vez, eres quien guía el destino de mi vida.

A Dios, mi sustentador, a mi amor mi apoyo, a mi hija mi inspiración, a mis padres mi ejemplo, a mi familia mi sangre.

Te lo agradezco Padre Celestial.

ROMMEL HURTADO

DEDICATORIA

A Dios por brindarme la oportunidad y la dicha de la vida, además darme los medios necesarios para continuar mis estudios profesionales, y ser un apoyo incondicional para lograrlo ya que sin él no lo hubiese logrado.

A mis padres, porque creyeron en mí y me sacaron adelante, dándome ejemplos dignos de superación y entrega, porque en gran parte gracias a ustedes, hoy puedo ver alcanzada mi meta, ya que siempre estuvieron impulsándome en los momentos más difíciles de mi carrera, y porque el orgullo que sienten por mí fue lo que me hizo ir hasta el final. Esta tesis es dedicada a ustedes, por lo que valen, porque admiro su fortaleza y por lo que han forjado en mí.

A mi esposa, que ha estado a mi lado dándome cariño, confianza y apoyo incondicional para seguir adelante para cumplir otra etapa en mi vida. A mi Hija, que es el motivo y la razón que me ha llevado a superarme cada día, y de esta manera alcanzar mis más apreciados ideales de superación. Ellas fueron quienes en los momentos más difíciles me dieron su amor y compresión para continuar.

A mis maestros, por su gran apoyo y motivación para la culminación de mis estudios profesionales, por su tiempo compartido y por impulsar el desarrollo de nuestra formación profesional.

A mis amigos por el apoyo mutuo durante nuestra formación profesional y porque hasta el momento, seguimos siendo amigos. Gracias por haber compartido parte de sus vidas conmigo.

ROMMEL HURTADO

TABLA DE CONTENIDO

DECLARACION DE AUTOR ii
CERTIFICACIÓN DEL TUTOR. iii
AGRADECIMIENTO iv
DEDICATORIA v
TABLA DE CONTENIDO vi
LISTA DE FIGURAS x
LISTA DE TABLAS xii
LISTA DE CÓDIGOS xiii
RESUMEN xiv
ABSTRACT xv
INTRODUCCIÓN 1
Antecedentes 1
Planteamiento del problema 1
Formulación del problema 2
Justificación 2
Objetivo General 3
Objetivos Específicos 3
Alcance 4
Descripción de los capítulos 4
CAPÍTULO 1 6
1. FUNDAMENTACIÓN TEÓRICA 6
1.1 El sistema GPS 6
1.1.1 El segmento Espacial 7
1.1.2 El segmento de Control 7
1.1.3 El segmento de Usuario 8
1.1.4 Señales GPS 8

1.1.5 Determinación de la posición de un punto 9
1.1.6 Errores en la recepción 10
1.2 Tecnología GSM 11
1.2.1 Generalidades de GSM 11
1.2.2 Arquitectura de la red GSM 11
1.2.3 Short Messages Service (SMS) 13
1.3 GPRS (*General Packet Radio Service*) 16
1.3.1 Clases de terminales 16
1.3.2 Servicios ofrecidos por GPRS 17
1.4 Tecnología Arduino 17
1.4.1 Funcionamiento de Arduino 17
1.4.2 Versiones de Arduino 18
2.4.3 Descripción de la placa Arduino Uno 19
1.4.4 Arduino IDE (*Integrated Development Environment*) 20
1.4.5 Módulo GPS Modelo: GPS6MV2 24
1.4.6 GSM900 SHIELD 25
1.5 Android 26
1.5.1 Conceptos generales de Android 26
1.5.2 Arquitectura 27
1.5.3 Estructura de una aplicación Android 28
1.6 App Inventor 30
1.6.1 Diseñador 30
1.6.2 Editor de bloques 31
1.6.3 Programación en App Inventor 32
CAPÍTULO 2 34
2. PROPUESTA 34
2.1 Introducción 34

2.1.1 Diagrama de casos de uso 34
2.1.2 Diagrama de procesos del sistema 35
2.2 Diseño del Sistema 36
2.2.1 Circuito Controlador 37
2.2.2 Aplicación móvil 41
CAPÍTULO 3 46
4. IMPLEMENTACIÓN 46
3.1 Desarrollo 46
3.1.1 Requerimientos funcionales 46
3.1.2 Requerimientos no funcionales 47
3.1.3 Consideraciones importantes para el desarrollo de la aplicación. 47
3.1.4 Cronograma de actividades 47
3.2.1 Implementación del hardware del circuito controlador 47
3.2.2 Implementación del software del circuito controlador 50
3.2.3 Implementación del *software* de la aplicación móvil 57
3.2.4 Instalación de la aplicación en el *smartphone* 61
3.2.5 Opción 1 de descarga: App (provide QR code for .apk) 62
3.2.6 Opción 2 de descarga: App (save .apk to my computer) 63
3.3 Pruebas de funcionamiento 63
3.3.1 La aplicación determina mi ubicación y muestra mi dirección. 63
3.3.2 Pruebas del Sistema 65
3.4 Análisis de resultados 69
3.4.1 Funcionamiento de la autenticación 69
3.4.2 Recepción de las coordenadas de ubicación 71
3.4.3 Conversión de grados sexagesimales a decimales. 72
3.4.4 Resumen análisis de resultados. 73
CONCLUSIONES 76

RECOMENDACIONES 77
BIBLIOGRAFÍA 78
ANEXOS 81
ANEXO 1: MANUAL TÉCNICO 81
ANEXO 2: MANUAL DE USUARIO 85
ANEXO 3: CRONOGRAMA 89

LISTA DE FIGURAS

Figura 1.1 - Estaciones de control de GPS alrededor del mundo 7
Figura 1.2 - GPS topográfico 8
Figura 1.3 – Trilateración de satélites 9
Figura 1.4 – Señales GPS reflejadas en edificios o muros 10
Figura 1.5 Arquitectura GSM 12
Figura 1.6 - Placa Arduino Uno 19
Figura 1.7 - IDE de Arduino 21
Figura 1.8– Conexión Arduino UNO con módulo GPS6MV2 25
Figura 1.9 – Conexión Arduino UNO con el Shield GSM900 26
Figura 1.10- Arquitectura de Android 28
Figura 1.11 - Ciclo de vida de Activity en Android 29
Figura 1.12 - Interfaz gráfica de usuario modo diseñador 31
Figura 1.13 - Editor de Bloques 32
Figura 2.1 - Diagrama de casos de uso del Sistema. 34
Figura 2.2 - Diagrama de procesos del Sistema 36
Figura 2.3 - Diagrama de bloques del Sistema 36
Figura 2.4 - Diagrama de bloques del Hardware del Circuito Controlador 37
Figura 2.5 - Bloque de Alimentación 38
Figura 2.6 - Bloque de comunicaciones - Módulo Shield GSM 38
Figura 2.7 - Bloque de Sensado - Modulo GPS6MV2 39
Figura 2.8 - Bloque de Control – Placa Arduino Uno 39
Figura 2.9 - Diagrama de conexión del Circuito Controlador 40
Figura 2.10 - Diagrama de flujo del circuito controlador. 41
Figura 2.11 - Diagrama de flujo de la aplicación móvil. 42
Figura 2.12 - Boceto de la interfaz gráfica de la aplicación móvil. 43
Figura 3.1 - Arduino Uno Utilizado 48
Figura 3.2 - Shield GSM – SIM900 utilizado 48
Figura 3.3 - Módulo GPS6MV2 utilizado 49
Figura 3.4 - Cables utilizados 49
Figura 3.5 - Espadines 50
Figura 3.6 - Compilación correcta del código en Arduino IDE 55
Figura 3.7 - Conexión del Arduino con el PC a través de USB. 55

Figura 3.8 - Identificación del puerto de conexión del Arduino. 56
Figura 3.9 - Carga del código en el Arduino Uno 57
Figura 3.10 - Interfaz gráfica implementada. 58
Figura 3.11 - Bloque 1 y 2 58
Figura 3.12 - Bloque 3 59
Figura 3.13 - Bloque 4 60
Figura 3.14 - Bloque 5 60
Figura 3.15 – Bloque 6 61
Figura 3.16 – Opciones de descarga del archivo .apk 62
Figura 3. 17 - Código QR generado para descargar la aplicación. 62
Figura 3.18 - Aplicación recién iniciada. 64
Figura 3. 19 – Aplicación después de recibir los datos GPS 64
Figura 3.20 - Aplicación luego de presionar el botón “Mi localización” 65
Figura 3.21 - Instalación del circuito en un vehículo 65
Figura 3.22 - Envío y recepción de mensajes entre el circuito y la aplicación 66
Figura 3. 23 - Primera localización del vehículo. 67
Figura 3.24 - Determinación de la distancia, tiempo y ruta 1 67
Figura 3.25 - Determinación de posición, distancia, tiempo y ruta 68
Figura 3.26 - Tercera prueba de localización y ubicación 69
Figura 3.27 – Determinación de la ubicación del dispositivo móvil 70
Figura 3.28 – Envió de un SMS con la contraseña 70
Figura 3.29 - Recepción del SMS con las coordenadas 71
Figura 3.30 – Redirección de la información del SMS a la aplicación Android. 72
Figura 3.31 – Conversión automática de grados sexagesimales a decimales 73

LISTA DE TABLAS

Tabla 1.1 - Acrónimos de los elementos de la arquitectura de red GSM 12
Tabla 1.2 - Operadores aritméticos..... 22
Tabla 1.3 - Operadores relacionales 23
Tabla 1.4 - Operadores lógicos..... 23
Tabla 2.1 - Descripción de la interfaz gráfica de la aplicación móvil. 44
Tabla 3.1 - Prototipo desarrollado vs uno existente en el mercado. 74
Tabla 3.2 - Razones para el uso de los elementos del prototipo. 75

LISTA DE CÓDIGOS

Código 3.1 - Declaración y asignación de Variables e Importación de librerías 51
Código 3.2 - Configuración Básica de los módulos del circuito. 52
Código 3.3 - Método para el encendido del módulo GSM 52
Código 3.4 - Programación de la función loop 53
Código 3.5 - Método para la obtención de datos del GPS 54
Código 3.6 - Método para enviar ubicación mediante un SMS. 54

RESUMEN

La geolocalización de vehículos es un tema de gran interés en la comunidad actual, esto ha permitido desarrollar múltiples aplicaciones para proveer el servicio. El enfoque de este proyecto es mejorar la calidad de servicio del transporte público a través de la automatización del mismo. El problema por afrontar es la falta de cumplimiento de las rutas establecidas por los conductores, lo cual ha causado inconformidad e inseguridad en la población.

En este estudio se propone el desarrollo de un prototipo que permita localizar un bus, observar la ruta y determinar el tiempo de llegada a su destino. Para lo cual se utilizarán tecnologías actuales como GPS, GSM, entre otras.

En cuanto al *hardware* se propone el uso de un módulo Arduino UNO, un shield GSM900 y un receptor GPS. El primero forma el circuito controlador, interconecta cada una de las partes del sistema y verifica su funcionamiento. El segundo posee las funcionalidades básicas de una placa de teléfono celular, con la configuración adecuada puede transmitir y recibir mensajes o llamadas. Y por último el receptor GSM, permite obtener las coordenadas del autobús.

En la parte de *software* se dispone de una aplicación Android y el código desarrollado en el Arduino. La aplicación Android, brinda al usuario una interfaz gráfica que permite la transmisión y recepción de SMS para determinar la posición en un mapa, aquí se especifica el número al cual se debe enviar el mensaje para solicitar su ubicación. En cambio, el código desarrollado en el microcontrolador del Arduino UNO; permite la interconexión de circuito controlador con los demás componentes, en éste se especifica el número de teléfono al cuál debe responder los mensajes con las coordenadas geográficas respectivas. Las pruebas fueron realizadas en vehículos en movimiento.

Palabras clave: GPS, geolocalización, Android, GSM, Arduino.

ABSTRACT

The geolocation of vehicles is a subject of great interest in the current community, this has allowed the development of multiple applications to provide the service. The focus of this project is to improve the quality of public transport service through the automation of it. The problem to be faced is the lack of compliance with the routes established by the drivers, which has caused dissatisfaction and insecurity in the population.

In this study we propose the development of a prototype that allows locating a bus, observing the route and determining the arrival time to its destination. What are the current technologies used like GPS, GSM, among others.

In terms of hardware, the use of the Arduino UNO module, a GSM900 shield and a GPS receiver is proposed. The first forms the controller circuit, interconnects each of the parts of the system and verifies its operation. The second has the basic functions of a cell phone cell, which can send and receive messages or calls. And finally the GSM receiver, allows to obtain the bus coordinates.

In the software part you can find an Android application and the code developed in the Arduino. The Android application has a graphical interface that allows the transmission and reception of SMS to determine the position on a map, here you specify the number to which the message should be sent to request its location. In contrast, the code developed in the microcontroller of the Arduino UNO; allows the interconnection of the circuit with the other components, in this it specifies the telephone number to which the messages must respond with the respective geographic coordinates. The tests were conducted on moving vehicles.

Keywords: GPS, geolocalización, Android, GSM, Arduino

INTRODUCCIÓN

Antecedentes

La geolocalización es la ubicación de determinado objeto, a través del uso de sistemas de posicionamiento para esto se han desarrollado algunos sistemas, el más popular es el sistema GPS (*Global Position System*), el cual ha permitido la creación de múltiples y variadas aplicaciones, para distintos usos (Force, Control Segment, 2017).

En el ámbito de transporte, se han desarrollado aplicaciones que permiten enviar mensajes para conocer los horarios de los vehículos públicos. Esta aplicación consiste en que el usuario envía un SMS con el nombre de la parada en la que se encuentra y recibe las diferentes rutas que pasan por ese lugar, esta información la envía los sistemas desarrollados por las diferentes cooperativas de transporte.

Otra de las aplicaciones de geolocalización en el transporte, es el sistema que usan muchos vehículos de transporte liviano, para ser localizados en caso de robo o pérdida. Es especialmente utilizado por las aseguradoras.

Otra aplicación de la geolocalización para servicios de transporte urbano está enfocado al turismo. De esta manera las personas que llegan por primera vez a un lugar, a través de aplicaciones propias para cada lugar, pueden ubicarse en el mapa y ver las posibles rutas para llegar a un sitio turístico de interés en particular (Force, Control Segment, 2017).

Planteamiento del problema

En Ecuador, los usuarios del transporte urbano se quejan constantemente por el servicio recibido. Las causas de las principales quejas son el maltrato por parte de los conductores y ayudantes, el mal estado de los buses, el incumplimiento de horarios, la falta de número de unidades en las horas pico, entre otras.

Existen varias medidas que han tomado los municipios para intentar regular el servicio de transporte público. Entre ellas, la caja común, la incorporación de cámaras de seguridad en el interior de las unidades, entre otras. No obstante, no se han conseguido mayores resultados.

En otros países, en los cuáles el sistema de transporte ha evolucionado (Lizana, 2017). Los buses tienen rutas y horarios establecidos, los mismos que en escasas ocasiones están sujetos a cambios. Esto ha permitido a los usuarios poder planificar su rutina, tener un buen trato y mejorar su calidad de vida.

Es importante resaltar que la generalización del mal servicio que ofrecen los transportes públicos en el Ecuador ha causado pérdidas materiales y humanas. Porque el incumplimiento de rutas y horarios ha provocado que los buses excedan los límites de velocidad y existan una gran cantidad de accidentes.

Existen sistemas de timbrado, en lugares estratégicos durante la ruta establecida, para regular que los transportistas sigan el recorrido determinado y cumplan con los horarios instaurados; sin embargo, nada garantiza que durante los intervalos para el timbrado los buses cumplan con la ruta; incluso han existido quejas; puesto que, en muchas ocasiones los conductores optan por caminos no establecidos. Esto ha causado inconformidad e inseguridad en la población, pues, no pueden aseverar si una unidad va a cumplir con su horario y ruta.

Formulación del problema

El problema que se formula en esta tesis es el incumplimiento de las rutas establecidas por parte de los transportistas públicos del Ecuador. Lo cual ha causado inconformidad en la población. Para esto se plantea implementar un prototipo que permita regular el cumplimiento de rutas y horarios de los buses. Esto se puede evidenciar en la resistencia de la población al aumento de la tarifa del transporte público; debido al mal servicio que brindan los mismos.

Justificación

La comunicación es el arma más importante para el desarrollo tecnológico en la actualidad. Hace algunos años se pensaba que el desarrollo se iba a enfocar en el transporte, creando carros que vuelen; sin embargo, la complejidad del ser humano guio los avances tecnológicos hacia las telecomunicaciones. Estar conectados con otras personas, para las diferentes actividades, a cualquier momento se ha vuelto prioridad para la sociedad actual. Pero también, ha surgido la necesidad de conectar al ser humano con las máquinas u otros objetos. Una de las tecnologías que más ha revolucionado la concepción de comunicación es el GPS. Este sistema permite la ubicación de cualquier objeto que

tenga un receptor GPS, a cualquier hora, independientemente de las variaciones climatológicas y de manera gratuita. Esto ha permitido que se desarrollen un sin número de aplicaciones que, haciendo uso de esta herramienta, solucionen muchos problemas de la vida diaria.

La telefonía móvil avanza con pasos de gigante hacia nuevos horizontes, que buscan solucionar los problemas cotidianos de las personas, a través del desarrollo de aplicaciones. Las cuáles están enfocadas a la educación, al entretenimiento y a la investigación. Una aplicación posee funcionalidades específicas que ayudan al usuario.

El prototipo que se desarrolló pretende combinar la tecnología GPS, junto con red móvil celular y de esta manera lograr una automatización del servicio de transporte urbano. Esto se logra a través de censar la ubicación de los buses de transporte urbano constantemente y de esta manera lograr determinar que cumpla con la ruta establecida y los tiempos de llegada. Se decidió realizar este proyecto para que se pueda mejorar la calidad del servicio que brinda el transporte público.

Objetivo General

Implementar un prototipo de sistema de localización GPS en un bus urbano que permita mediante una aplicación Android conocer el recorrido de la ruta asignada.

Objetivos Específicos

- Definir las variables de tiempo y distancia para la estimación de coordenadas de ubicación de un bus urbano.

- Diseñar el sistema GPS que será implementado en el bus de transporte urbano para lograr obtener las coordenadas.

- Desarrollar la aplicación Android que calcule distancia y tiempo entre paradas y permita visualizar el recorrido de la ruta del bus.

- Crear el sistema completo de la aplicación Android y el monitoreo mediante el GPS para identificar el bus y la ruta.

- Analizar los resultados de las pruebas obtenidas de la implementación de la aplicación Android y GPS.

Alcance

La implementación de este prototipo en el bus ayudará a los usuarios a conocer la ubicación y la ruta de la unidad, el mismo contará con las siguientes características:

- En este sistema se desarrollará una aplicación móvil para el sistema operativo Android.
- En el bus se colocará un prototipo que incluye un módulo que tiene funciones parecidas a un teléfono celular, y a través de los mensajes de texto se obtendrá la localización de este.
- Se utilizará un módulo GPS que permitirá obtener las coordenadas del vehículo.
- La aplicación permite conocer la distancia y el tiempo estimado de llegada del bus.

Descripción de los capítulos

El presente proyecto consta de varias secciones distribuidas de la siguiente manera:

En el capítulo 1 se contempla la fundamentación teórica. En el marco teórico se agrupa conceptos e ideas que se utilizan para desarrollar el argumento. Se realiza una descripción rápida de las principales características de GPS, para de esta manera entender su funcionamiento. A continuación, se detalla el funcionamiento de las tecnologías celulares utilizadas, en este caso GPS y GSM. Después se analiza las plataformas donde se implementará el *hardware* y *software* del prototipo, entre las que están la tecnología Arduino y las aplicaciones Android. Al finalizar esta sección se tendrá una visión en conjunto de los principales conocimientos y prácticas adecuadas que permitirán llevar a cabo el trabajo del prototipo.

Por otro lado, en el capítulo 2 se encuentra la propuesta de diseño para la implementación del prototipo. Con respecto al *software*, en esta sección se encuentran los diagramas utilizados para poder crear los respectivos programas; además, se describen las respectivas clases, comandos, funciones, objetos utilizados para la programación. Para

finalizar en esta sección se realiza una descripción del *hardware*, se indican las respectivas conexiones que fueron necesarias para implementar el prototipo.

A continuación, en el capítulo 3 se explica la construcción del prototipo, indicando el desarrollo respectivo. Se indica paso a paso como se realizó la implementación. Además, se presentan las respectivas pruebas de funcionamiento para posteriormente realizarse un análisis de resultados. Para finalizar, se tienen las respectivas conclusiones y recomendaciones. Las últimas permitirán mejorar trabajos futuros enfocados al área de geolocalización del transporte público.

CAPÍTULO 1

1. FUNDAMENTACIÓN TEÓRICA

El desarrollo de nuevas tecnologías lleva a la cosmovisión humana a otro nivel. Los avances científicos permiten encontrar una solución práctica a los problemas que se presentan cotidianamente a las personas, familias, comunidades y empresas.

La geolocalización es uno de los aportes más importantes que se ha hecho en las últimas décadas porque ha permitido el desarrollo de múltiples proyectos en diversas áreas cómo el transporte, la telefonía móvil, entre otras. La geolocalización ha permitido a los sistemas de transporte evolucionar con pasos agigantados porque se ha implementado aplicaciones que han permitido garantizar un mejor servicio. Para la realización de este trabajo que vincula a la sociedad con la tecnología, es necesario la explicación de las tecnologías utilizadas tanto para el desarrollo del *software* como del *hardware*.

Para la sustentación teórica se presenta una explicación de las tecnologías utilizadas.

Primero se ejecuta la descripción del sistema GPS, el cual permite la ubicación de un objeto en cualquier lugar de la Tierra. En segundo lugar, se realiza un resumen indicando las principales características del sistema celular GSM(*Global System for Mobile*), enfatizando en el envío y recepción de SMS, lo cual es fundamental para la aplicación de localización. Después, se describen los aspectos de mayor importancia de GPRS (*General Packet Radio Service*), debido a que el módulo GSM900 puede operar tanto en GSM como en GPRS.

A continuación, se explica el funcionamiento de la tecnología Arduino, con los respectivos sensores y módulos utilizados para la ejecución del proyecto. Finalmente, se realiza un resumen de la plataforma APP Inventor en la cual se realizará la aplicación.

1.1 El sistema GPS

El Sistema de Posicionamiento Global (GPS), es responsabilidad del Departamento de Defensa de los Estados Unidos (DoD), este sistema permite conocer la ubicación de un objeto (automóvil, personas, animales, celulares, entre otros) con una precisión que varía entre algunos metros o centímetros, es un servicio gratuito y continuo a los usuarios civiles que tenga un módulo GPS, independientemente de la condición atmosférica, la hora del

día, el lugar y el número de usuarios que hagan uso del sistema simultáneamente. Está constituido por tres segmentos: (Hofmann, Collins, & Lichtenegger, 2012).

- EL segmento Espacial.
- El segmento de Control.
- El segmento de Usuario.

1.1.1 El segmento Espacial

El segmento Espacial consiste en satélites que emiten señales de tipo *broadcast*. Los mismos que se encuentran en órbita a 20200km sobre la Tierra y se demoran 12 horas en dar una vuelta completa. Los satélites forman una constelación de por lo menos 24 satélites del gobierno de los Estados Unidos distribuidos en seis planos orbitales inclinados 55 ° con respecto al plano ecuatorial. De esta manera se logra que en cualquier instante de tiempo se encuentre visibles 4 satélites desde cualquier punto de la Tierra (Hofmann et al., 2012).

1.1.2 El segmento de Control

El segmento de control está formado por estaciones de seguimiento, mantenimiento y control, las cuales se encuentran distribuidas en todo el mundo. Su objetivo principal es realizar el mantenimiento adecuado a los satélites para que éstos se mantengan en órbita y sus relojes estén calibrados correctamente, a través del envío de comandos a la constelación de satélites. En la Figura 1.1 se puede observar cómo se encuentra distribuidos las estaciones de control alrededor del mundo, de manera estratégica cercanas al plano ecuatorial (Hofmann et al., 2012).

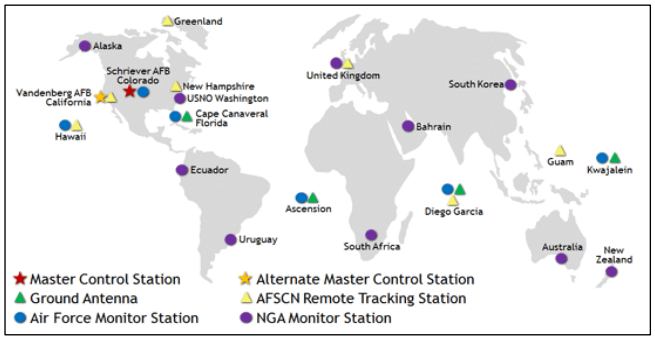


Figura 1.1 - Estaciones de control de GPS alrededor del mundo

Fuente: (Force, Control Segment, 2017)

1.1.3 El segmento de Usuario

El sistema de usuario está conformado por el equipo receptor GPS, el cual recibe las señales de los satélites GPS y las procesa para calcular la posición tridimensional y la hora exacta (Force, Control Segment, 2017).

GPS se caracteriza por ser gratuito, dar un servicio ininterrumpido y fiable. Por consiguiente, ha permitido a los usuarios de todo el mundo desarrollar múltiples aplicaciones. En la actualidad los receptores GPS se encuentran en diferentes plataformas, como celulares, *laptops*, equipos de rastreo, equipos de topografía (Hofmann et al., 2012).

Es importante que el equipo receptor GPS cuente con las especificaciones técnicas mínimas necesarias para tener una lectura de las ubicaciones con mayor exactitud, una de las principales causas de la falta de precisión en la ubicación de objetos, son los defectos que se pueden encontrar en el *hardware*. En la Figura 1.2 se observa un receptor GPS topográfico con antena, para tener una mayor precisión.

Figura 1.2 - GPS topográfico

Fuente: (Gisiberica, 2017)

1.1.4 Señales GPS

Dos señales de radio frecuencia con baja potencia son transmitidas por los satélites del GPS, denominadas L1 y L2. Cada señal GPS contiene tres elementos de información: un código pseudoaleatorio, el cual identifica al satélite; los datos de efemérides de satélite, que brindan información de la ubicación del satélite en cualquier instante de tiempo y datos de almanaque, los cuales proporcionan información adicional del satélite y la fecha y hora.

La señal L1 tiene una frecuencia de portadora igual a 1575,42 MHz. Esta señal es de uso civil, utiliza el código de adquisición aproximativa. La señal L2 es de uso militar y utiliza el código de precisión cifrado, cuya frecuencia de su portadora es de 1227,60 MHz (ETSI, 1996).

1.1.5 Determinación de la posición de un punto

La base para determinación de la posición de un punto es la triangulación o trilateración de satélites. A continuación, se describe este procedimiento, el mismo que se observa en la Figura 1.3.

Un receptor GPS en la tierra determina la distancia de cada una de las señales con respecto al objeto de medición, usando el producto del tiempo de transmisión de las señales de radio con el valor de la velocidad de la luz y la posición exacta del primer satélite, de esta manera se ubica el receptor GPS en un punto cualquiera de la superficie de una esfera.

Se mide la distancia de un segundo satélite al mismo receptor, se genera otra esfera de diferente radio, se realiza la intercepción de las dos esferas, encontrando un círculo de intercepción. Se agrega una tercera esfera, la intersección de esta con las dos anteriores esferas corta en dos puntos al círculo encontrado, de esta manera encontramos las coordenadas x, y. Para determinar la coordenada z (altura) es necesario una cuarta esfera (Lizana, 2008).

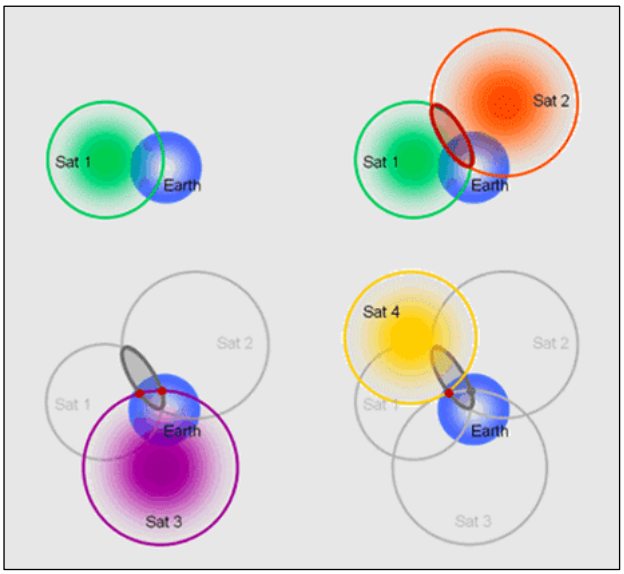


Figura 1.3 – Trilateración de satélites

Fuente: (YouBioit.com, 2017)

1.1.6 Errores en la recepción

Existen muchas causas que degradan la precisión del posicionamiento GPS. A continuación, se enlistan las más comunes.

- Bloqueo de señales de satélite debido a edificios, puentes, árboles, etc. Porque la señal satelital no atraviesa ciertas superficies.
- Uso interior o subterráneo, debido a que para la recepción y transmisión de señales GPS es necesario tener línea de vista con los satélites.
- Señales reflejadas en edificios o muros, lo cual provoca multitrayectoria (propagación de una onda por caminos diferentes), como se observa en la Figura 1.4.

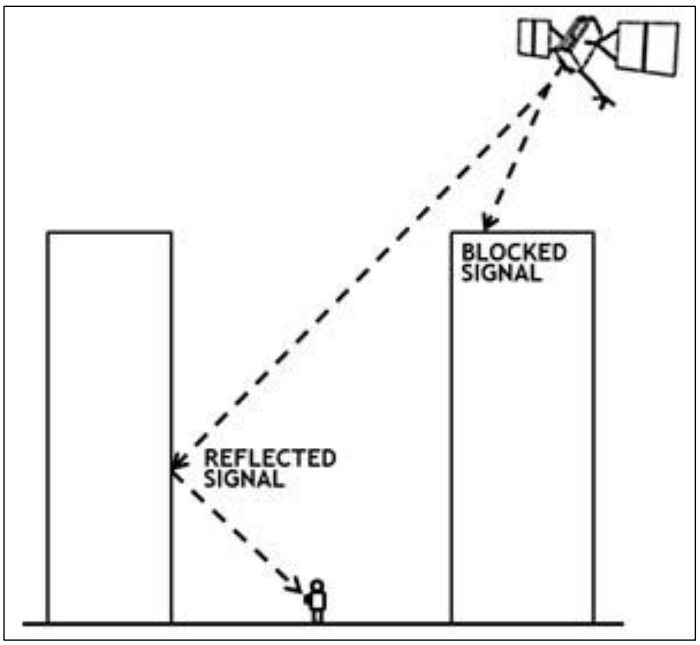


Figura 1.4 – Señales GPS reflejadas en edificios o muros

Fuente: (Force, GPS Accuracy, 2017)

Entre las causas menos comunes se encuentran la interferencia de radio, las tormentas solares, brechas temporales en la cobertura debido a mantenimiento de los satélites, dispositivos de diseño inadecuado que no cumplen con las especificaciones mínimas de la interfaz GPS.

En muchas ocasiones el hardware GPS se encuentra funcionando correctamente, sin embargo, su *software* de asignación presenta fallas. Los usuarios suelen ser engañados por:

- Mapas dibujados incorrectamente.
- Negocios identificados, como cafeterías, bares y otras localidades de interés.
- Direcciones de calles que se estimaron erróneamente.

1.2 Tecnología GSM

1.2.1 Generalidades de GSM

En la década de los 80s, a causa de, la saturación de los sistemas de primera generación de telefonía móvil se desarrolló el estándar GSM. Éste fue desarrollado por la ETSI (*European Telecommunications Standards Institute*), el mismo que cuenta con las siguientes características:

- Es abierto, no propietario, evolutivo y el más utilizado a nivel mundial.
- Utiliza tecnología digital, lo que permite la transmisión de voz y datos a diferentes velocidades.
- Uso de la encriptación de la información para asegurar confidencialidad en las comunicaciones.
- Utiliza modulación GSMK (*Gaussian minimum shift keying*) y técnica de acceso FDMA/TDMA (*Frequency Division Multiple Access / Frequency Division Multiple Access*) con 8/16 canales por portadora.
- Permitió la simplificación de los terminales de usuario, en comparación con la primera generación.
- Permite el envío y recepción de SMS (*Short Message Service*)

2.2.2 Arquitectura de la red GSM

La arquitectura de red GSM presenta dos componentes importantes: la infraestructura fija y los suscriptores móviles. La infraestructura física se puede dividir en tres subsistemas:

- Subsistema de estación base (BSS).
- Subsistema de red (NSS).
- Subsistema de mantenimiento y operación (OMSS).

En la Figura 1.5 se pueden observar las principales partes de la arquitectura de la red GSM, las cuales se describirán en los puntos posteriores. En la Tabla 1.1 se encuentran las definiciones de los acrónimos de los componentes que forman parte de la arquitectura GSM.

Tabla 1. 1 - Acrónimos de los elementos de la arquitectura de red GSM

Acrónimo	Descripción
BTS	Base Transceiver Station
BSC	Base Station Controller
MSC	Mobile Switching Center
GMSC	Gateway MSC
MS	Mobile Station
HLR	Home Location Register
AUC	Authentication Center
OMC	Operation and Maintenance Center
VLR	Visitors Location Register
EIR	Equipment Identity Registe

Fuente: Elaborado por el autor

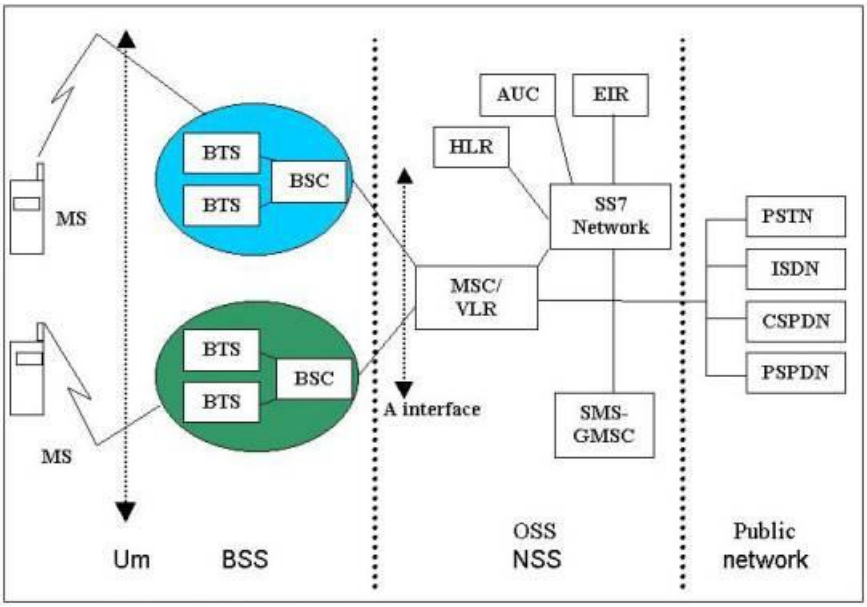


Figura 1.5 Arquitectura GSM

Fuente: (Gete-Alonso Roldán, 2008)

1.2.2.1 Subsistema de estación base (BSS)

Está compuesto por las BTS y BSC. Sus funciones involucran el control de los recursos de radio. La BTS tiene un equipo que se encarga de la transmisión y recepción de las señales de radio. La BSC se encarga de administrar los recursos de radio de un grupo de BTS, especialmente para mantener las llamadas (Gete-Alonso Roldán, 2008).

1.2.2.2 Subsistema de Red (NSS)

El subsistema de red es el encargado de la conmutación y control durante una llamada. Además, administra la movilidad, mediante la asociación de la MSC al VLR. El MSC, es la interfaz entre la red de telefonía celular y otras redes fijas. Enruta las llamadas hacia y desde la MS. El HLR se encarga de almacenar la identidad y otros datos de los clientes. El registro VLR proporciona la información necesaria cuando la llamada se origina en el móvil (Gete-Alonso Roldán, 2008).

1.2.2.3 Subsistema de mantenimiento y operación (OMSS)

El OMSS se encarga de la gestión, explotación y mantenimiento del sistema GSM. El OMC se encarga de la tarifación de llamadas y mensajería realizada por los usuarios, además, administra los equipos de la red. El NOC (*Network Operations Center*) es un espacio físico en el cual se llevan a cabo todas las funciones de mantenimiento y operación de la red. Así, por ejemplo, monitoreo de alarmas, localización de errores y respuesta a incidentes.

El AUC trabaja en conjunto con el HLR para proporcionar los parámetros necesarios para la autenticación del MS, por lo que necesita conocer el cifrado que cada abonado utiliza. El registro EIR almacena toda la información de cada uno de los terminales, como si fueron robados o tienen algún defecto (Gete-Alonso Roldán, 2008).

1.2.3 Short Messages Service (SMS)

1.2.3.1 Definición de SMS

Para la transmisión de SMS se utiliza la modalidad *Store and Forward* (almacenamiento y reenvío). La entrega de mensajes no es en tiempo real, porque estos deben ser procesados por la base de datos. El envío de mensajes se realiza mediante los

canales de control generados por los centros remotos. Si el móvil de destino está desconectado de la red, se guardarán los *SMS* hasta su entrega; pero el tiempo de almacenamiento es limitado, después del tiempo límite el mensaje es eliminado de la base de datos (Hillebrand, Trosby, & Harris, 2010).

En la Figura 1.5 se observa el bloque SMSC (*Short Message Service Center*), el cual representa a la base datos sin la cual no se podría llevar a cabo el envío y transmisión de los SMS, ya que esta lo registra y los envía a través de la red. Sus principales funciones son:

- Autenticación de los usuarios.
- Recolecta y entrega los SMS a su destino.
- Almacenamiento de los SMS por un tiempo límite, hasta su posible entrega.
- Interconexión con los SMSC de otras operadoras.

2.2.3.2 Mensajes MT-SM (*Mobile Terminated-Short Message*) y MO-SM (*Mobile Originated-Short Message*)

Al comienzo, los SMS fueron especificados dentro del estándar GSM, con la única finalidad de enviar información a los abonados, desde los operadores. Los equipos terminales no podían responder, tampoco podían enviar SMS a otros abonados. No obstante Nokia desarrolló un sistema que permite la comunicación bidireccional de SMS. De esta manera en la actualidad se tiene dos tipos de mensajes SMS:

- SMS-MO (*Service Short Message- Mobile Originated)*: son los mensajes enviados entre el equipo terminal y la SMSC encargada de la gestión de los mismos, es decir los emitidos por los usuarios.
- SMS-MT (*Service Short Message- Mobile Terminated*): son los mensajes que envía la SMSC hacia el móvil de destino.

También existe una tercera clase de mensajes denominados SMS/CB (SMS Cell Broadcast), que son aquellos que se envían a todos los MS dentro de una celda.

2.2.3.3 Parámetros de un SMS

Los SMS que envía o recibe el usuario debe tener por lo menos la siguiente información, para aseverar el correcto procesamiento de los mensajes en el SMSC y a lo largo de toda su trayectoria.

- Fecha de transmisión del mensaje.
- Número de teléfono celular del remitente y del destinatario
- Número de identificación del SMSC que ha producido el SMS

1.2.3.4 Proceso de entrega de un SMS-MO

Cuando un abonado de la red celular genera un SMS-MO ocurren los siguientes sucesos:

- El HLR correspondiente al registro del usuario, toma la decisión de si está permitido el envío de mensajes, verificando que no haya ninguna novedad.
- El MSC al que se encuentra conectado el usuario recibe el SMS, remite todos los datos necesarios al VLR para su tarifación, a continuación, lo exporta al SMSC de origen.
- El SMSC de origen envía el mensaje al SMSC de destino. Una vez allí, se convierte en MT-SM y se procesa como se explicará más adelante.
- El SMSC de destino comunica el estado del mensaje y responde con un informe de recepción al MSC y al usuario. En la pantalla del usuario se visualiza el estado del mensaje (ETSI, 1996).

1.2.3.5 Proceso de entrega de un SMS-MT

Los SMS-MT son aquellos que se encuentran en el SMSC para su envío, ya sean estos originados por otro usuario, la operadora o cualquier otra circunstancia, su proceso de entrega es el siguiente:

- El SMSC que ha recibido el SMS lo almacena en su base de datos y realiza una petición al VLR del usuario acerca de la información de localización.

- Si el destinatario está disponible, el SMSC transmite el mensaje al MSC, enseñando en que parte del BSS debe ser entregado; caso contrario, se almacena en el SMSC durante un tiempo límite.
- Si el destinatario está disponible, el MSC remite un aviso al VLR al que está conectado el abonado de destino, cual puede pertenecer a cualquier operadora, para indicarle se le hará entrega de un mensaje.
- El VLR avisa al terminal del remitente y verifica si se encuentra en la zona de cobertura de la red.
- El VLR responde al MSC indicando el estado del usuario y, si está disponible, adjunta la información de localización.
- El MSC envía el mensaje al destinatario.
- El MSC informa al SMSC de que el mensaje se ha entregado correctamente, para que este sea borrado de su base de datos (ETSI, 1996).

1.3 GPRS (*General Packet Radio Service*)

GPRS nació con el objetivo de hacer uso eficiente de las redes móviles para brindar el servicio de datos. Esta tecnología introduce servicios de conmutación de paquetes orientados a usuarios GSM, de esta manera entrega mayores velocidades y prestaciones. El subsistema de estación base (BSS) de GSM se mantiene. Esta nueva tecnología de telefonía móvil asigna calidad de servicio a los paquetes. Sus principales características son:

- Funcionamiento basado en la conmutación de paquetes.
- La facturación tiene como base el volumen de datos.
- GRPS comparte las bandas de frecuencia con GSM.
- Existe asignación dinámica de canales.
- Se hace uso de la infraestructura existente de la red GSM.

1.3.1 Clases de terminales

Como GSM y GPRS son dos tecnologías que cohabitan. Los terminales se pueden clasificar en las siguientes clases, en función de la capacidad de la red del MS.

- Clase A: Existe conexión simultánea, es decir un *slot* de tiempo para GSM y otro para GPRS; por lo tanto, no hay degradación de ninguno de los dos servicios.
- Clase B: Conexión GSM y GPRS; pero no de manera simultánea. Existe prioridad para GSM, por consiguiente, se degrada la calidad de GPRS.
- Clase C: La selección de la red GSM o GPRS es de manera manual.

1.3.2 Servicios ofrecidos por GPRS

GPRS ofrece los mismos servicios que GSM, pero modificados para ofrecer un mejor servicio, además cuenta con nuevas actividades para el usuario. Los principales servicios ofrecidos por GPRS son:

- Evolución de los servicios SMS, la longitud de los mensajes es ilimitada.
- Aplicaciones WAP (*Wireless Aplication Protocol*), que permite implementar transacciones bancarias, noticias, correo electrónico, entre otros.
- Telemetría que permite la medición remota de algún objetivo en particular de manera continúa.

1.4 Tecnología Arduino

Arduino es una plataforma electrónica de código abierto que incluye *hardware* y *software*. El *hardware* consta de una placa de circuito impreso con entradas y salidas analógicas y digitales controladas un microcontrolador ATmega328 y el *software* es implementado en el entorno de desarrollo Arduino IDE.

El entorno de programación Arduino (IDE) es un software de código abierto para el lenguaje de programación AVR-C basado en el lenguaje C++ con posibilidad de ser ampliado por programadores con experiencia a través de librerías de C++. Los programas de Arduino pueden ser divididos en tres partes principales: estructura, valores y funciones (Ruiz, 2007).

1.4.1 Funcionamiento de Arduino

Su funcionamiento se basa en la lectura de entradas como por ejemplo lecturas de sensores, el presionar un botón, leer un mensaje de texto, entre otras; siendo estas señales analógicas o digitales y convirtiendo estas lecturas en salidas como instrucciones de

prender un led, activar o desactivar dispositivos, enviar o publicar un mensaje de texto dependiendo de la serie de instrucciones utilizadas en el microcontrolador (Ruiz, 2007)

1.4.2 Versiones de Arduino

Arduino posee una variedad de tarjetas con distintas características las cuales son utilizadas de acuerdo con sus funcionalidades, tamaños, precios adaptándose a las distintas aplicaciones desarrolladas para sus proyectos. Entre los principales modelos se pueden describir las siguientes versiones que se detallarán a continuación.

1.4.2.1 Arduino Uno

Fue la primera placa lanzada al mercado por lo cual ha sido la base para desarrollar los distintos modelos. Consta de un microcontrolador ATmega328 que posee una memoria Flash de 32 KB de los cuales 5 KB son usados por el cargador de arranque, 2KB de SRAM (*Static Random Access Memory*) y 1 KB de EEPROM (*Electrically Erasable Programmable Read-Only Memory*), 14 pines de entradas y salidas digitales de las cuales 6 proveen salidas PWM (*Pulse Width Modulation*), 6 pines de entrada analógicos. Entre las características tiene un voltaje de operación de 5V, La corriente por PIN de entrada o salida es de 40 mA y una velocidad de Reloj de 16 MHz (Enríquez, 2009).

1.4.2.2 Arduino Leonardo

El Arduino Leonardo posee un microcontrolador ATmega32u4 con una memoria Flash de 32 KB, 1 KB de memoria EEPROM, 2.5 KB de SRAM, dispone de 20 pines de entrada y salida digitales, de las cuales 7 proveen de salidas PWM, 12 pines de entradas analógicas. Trabaja con una velocidad de reloj de 16 MHz (JAMECO Electrónica, 2017).

1.4.2.3 Arduino Mega 2560

El Arduino Mega posee un microcontrolador ATmega2560 de 8 bits con una memoria Flash de 256 KB, 4 KB de memoria EEPROM, 8 KB de SRAM, dispone de 54 pines de entrada y salida digitales de las cuales 15 proveen de salidas PWM, 16 pines de entradas analógicas (DSpace, 2017).

1.4.2.4 Arduino Nano

El Arduino Nano es una tarjeta de pequeño tamaño que posee un microcontrolador ATmega328 el cual fue remplazado por un ATmega168 de 16 MHz con memoria Flash de

16 KB, 512 Bytes de memoria EEPROM, 1 KB de SRAM, dispone de 14 pines de entrada y salida digitales de las cuales 6 proveen de salidas PWM, 8 pines de entradas analógicas (Arduino Nano- User Manual, 2017).

1.4.2.5 LilyPad Arduino

El LilyPad Arduino es una placa diseñada para poder adherirse a las prendas, esta placa posee un microcontrolador ATmega168 o ATmega328 dependiendo de su versión, cuenta con una memoria Flash de 16 KB, 512 Bytes de memoria EEPROM, 1 KB de SRAM, dispone de 14 pines de entrada y salida digitales de las cuales 6 proveen de salidas PWM, 6 pines de entradas analógicas (LilyPad Arduino, 2017).

1.4.2.6 Arduino Pro

El Arduino Pro posee un microcontrolador ATmega168 de 8 MHz o un ATmega328 de 16 MHz, dependiendo del microcontrolador utilizado dispondrá con una memoria Flash de 16 o 32 KB, 512 Bytes o 1 KB de memoria EEPROM, 1 o 2 KB de SRAM, dispone de 14 pines de entrada y salida digitales de las cuales 6 proveen de salidas PWM, 6 pines de entradas analógicas. (Romero, 2017)

2.4.3 Descripción de la placa Arduino Uno

En la Figura 1.6 se encuentra la placa de Arduino Uno, señalando sus respectivos componentes.

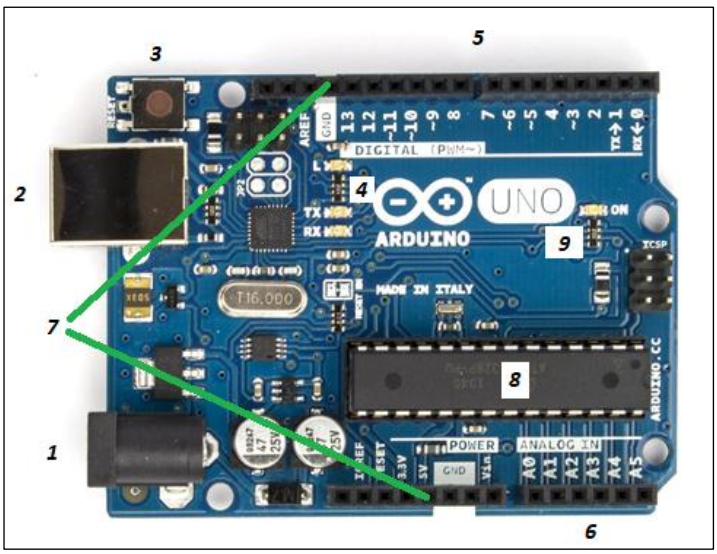


Figura 1.6 - Placa Arduino Uno

Fuente: (Ruiz, 2007). Modificado por el autor

1. **Conector de Alimentación:** Utilizado para alimentar al Arduino con un voltaje entre 7 a 12V cuando no se encuentre conectado a un puerto USB**.**
2. **Puerto USB:** Entre sus funciones está el de alimentar al circuito, cargar los programas y comunicarse con el programa de Arduino.
3. **Botón Reset:** Al Presionar este botón se conecta a tierra el PIN de Reset lo que reiniciará el código cargado en el Arduino.
4. **LEDs de transmisión y recepción:** Estos LEDs indican cuando se esté realizando una comunicación serie entre el Arduino y el computador.
5. **Pines de entrada/salida digital:** Estos pines pueden funcionar como entrada digital (Presionar un botón, etc)
6. **Entradas Analógicas:** Permiten sensar una señal leida de un sensor y transformarla a un equivalente digital.
7. **Pin GND (tierra) y 5V:** Utilizadas para la alimentación de la placa Arduino.
8. **Microcontrolador:** Microcontrolador ATmega328 del fabricante AVR usado para cargar las instrucciones del programa.
9. **Led de Encendido:** Indicador de que la placa está correctamente polarizada y todos sus elementos funcionando.

1.4.4 Arduino IDE (*Integrated Development Environment*)

Arduino ofrece un entorno de desarrollo integrado que nos permite escribir código con instrucciones y parámetros, compilarlos y cargarlos en la placa Arduino. Este entorno de desarrollo contiene un editor de texto para escribir el código, además una consola de texto, una barra de herramientas y botones para funciones comunes. Las cuales se pueden observar en la Figura 1.7.

Barra de herramientas: Permite verificar, cargar programas, crear, abrir, guardar bocetos y abrir el monitor en serie.

Editor de texto: El editor tiene funciones para copiar, cortar, pegar y para buscar y/o reemplazar texto.

El área de mensajes: Da retroalimentación al momento de guardar y exportar además muestra errores.

La consola: Muestra el texto generado por el *software* Arduino (IDE), incluyendo mensajes de error completos y otra información.

En la esquina inferior derecha de la ventana muestra la tarjeta configurada y el puerto serie.

Figura 1.7 - IDE de Arduino

Fuente: (Aprendiendo Arduino, 2017)

El IDE de Arduino utiliza el lenguaje de programación C/C++, y puede soportar otros lenguajes derivados de C como: C#, PHP, Java, Python, entre otros, lo que permite a los desarrolladores usar varios lenguajes de programación (Aprendiendo Arduino, 2017).

1.4.4.1 Estructura del programa

Los programas escritos con el Arduino IDE se denominan bocetos los cuales son escritos en el editor de texto y se guardan en un archivo con extensión *.ino* los cuales pueden utilizar los siguientes componentes: estructuras, variables y operadores.

1.4.4.1.2 Estructuras

Cualquier programa de Arduino está compuesto por dos funciones principales:

- setup(): En esta función se ubica el código utilizado para configurar e inicializar el programa, esta función va al inicio del código y solo se ejecuta una sola vez.
- loop(): En esta función se ubica el código que va a ser ejecutado una y otra vez hasta que la placa Arduino sea desconectada, esta función se ubica luego del setup() (Aprendiendo Arduino, 2017).

1.4.4.1.3 Variables

Son una forma de nombrar y almacenar un dato o conjunto de datos los cuales pueden ser modificados durante la ejecución del programa, estas variables pueden ser del tipo char, byte, int, long, float, doublé entre otros (Ruiz, 2007).

1.4.4.1.4 Operadores

Es un símbolo que representa a una función predefinida la cual recibe argumentos realizando operaciones entre ellos. Los principales operadores son:

- **Operadores aritméticos:** Permiten realizar cualquier operación aritmética que necesitemos. En la Tabla 1.2, se muestran algunos operadores aritméticos con su función asociada.

Tabla 1.2 - Operadores aritméticos

Operador	Acción
-	Resta
+	Suma
*	Multiplicación
/	División
·/.	Módulo
--	Decremento
++	Incremento

Fuente: Elaborado por el autor

- **Operadores relacionales:** Permiten realizar comparaciones entre un par de operador, el resultado de esta operación es un tipo de valor "verdadero" o "falso" representados por valores enteros. En la Tabla 1.3, se mostrarán algunos operadores relacionales que disponemos.

Tabla 1.3 – Operadores relacionales

Operador	Acción
>	Mayor que
>=	Mayor o igual que
<	Menor que
<=	Menor o igual que
==	Igual
!=	Distinto

Fuente: Elaborado por el autor

- **Operadores lógicos:** Permiten conectar un par de propiedades de manera lógica. Los operadores lógicos disponibles se encuentran en la Tabla 1.4.

Tabla 1.4– Operadores lógicos

Operador	Acción
&&	Conjunción (Y)
||	Disyunción (O)
!	Negación

Fuente: Elaborado por el autor

1.4.4.1.5 Estructuras de control

Son instrucciones que nos permiten tomar decisiones que sirven para controlar la ejecución del programa.

Las estructuras de control más importante son:

- **Condicionales:** Sirven para la toma de decisiones, por ejemplo: If-else y Switch/Case.

- **Ciclos:** Sirven para realizar acciones repetitivas dependiendo de una condición, por ejemplo: while, for.

1.4.4.1.6 Funciones

Son líneas de código que realizan acciones determinadas las cuales son ejecutadas al momento de ser llamadas. Estas funciones pueden ser llamadas una o varias veces por otras funciones o sí mismas.

En Arduino existen funciones básicas las cuales nos permiten configurar, leer, entre otras acciones los pines tanto analógicos como digitales.

1.4.5 Módulo GPS Modelo: GPS6MV2

Este módulo está basado en el chip receptor NEO 6M de la marca UBLOX, es un receptor GPS, puede ser controlado por Arduino o cualquier otro microcontrolador, incluye una antena cerámica la cual incluye los conectores para la alimentación y transmisión de datos. En la Figura 1.8 se observa el módulo GPS6MV2 y sus respectivas conexiones con un Arduino UNO. Las principales características técnicas de este módulo GPS son: (leantec-robotics&electronics, 2017)

- Receptor: Ublox NEO 6M.
- Voltaje de alimentación: 3V a 5V.
- Interfaz: UART, comunicación asíncrona.

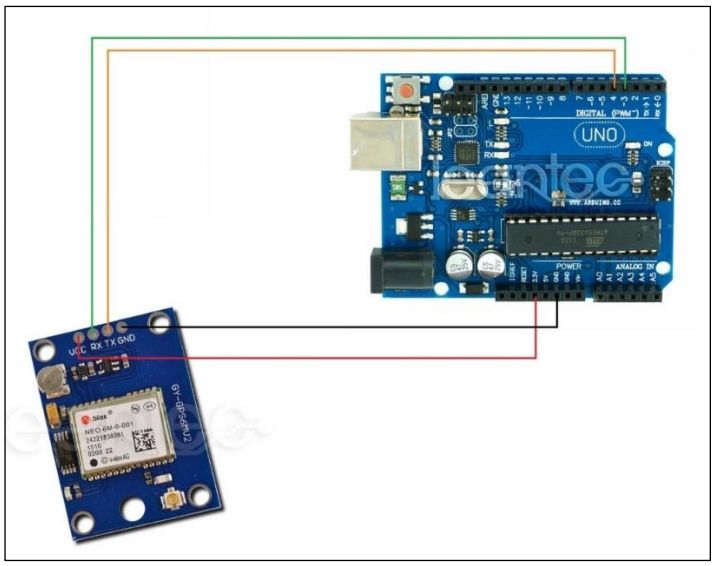


Figura 1.8– Conexión Arduino UNO con módulo GPS6MV2

Fuente: (leantec-robotics&electronics, 2017)

1.4.6 GSM900 SHIELD

Es una tarjeta GPRS, compacta que permite la comunicación inalámbrica. Es compatible con todas las versiones de Arduino UNO. En la Figura 1.9 se puede ver el shield GSM900 y su respectiva conexión con Arduino UNO. Sus especificaciones técnicas más importantes son: (HETPRO, 2017)

- Compatibilidad total con Arduino.
- Puerto serial para conexión.
- Quad-Band 850/ 900/ 1800/ 1900 MHz.
- GPRS multi-slot clase 10/8.
- GPRS mobile station clase B.
- Compatible GSM fase 2/2+.
- Consumo de 1.5 mA (suspendido).

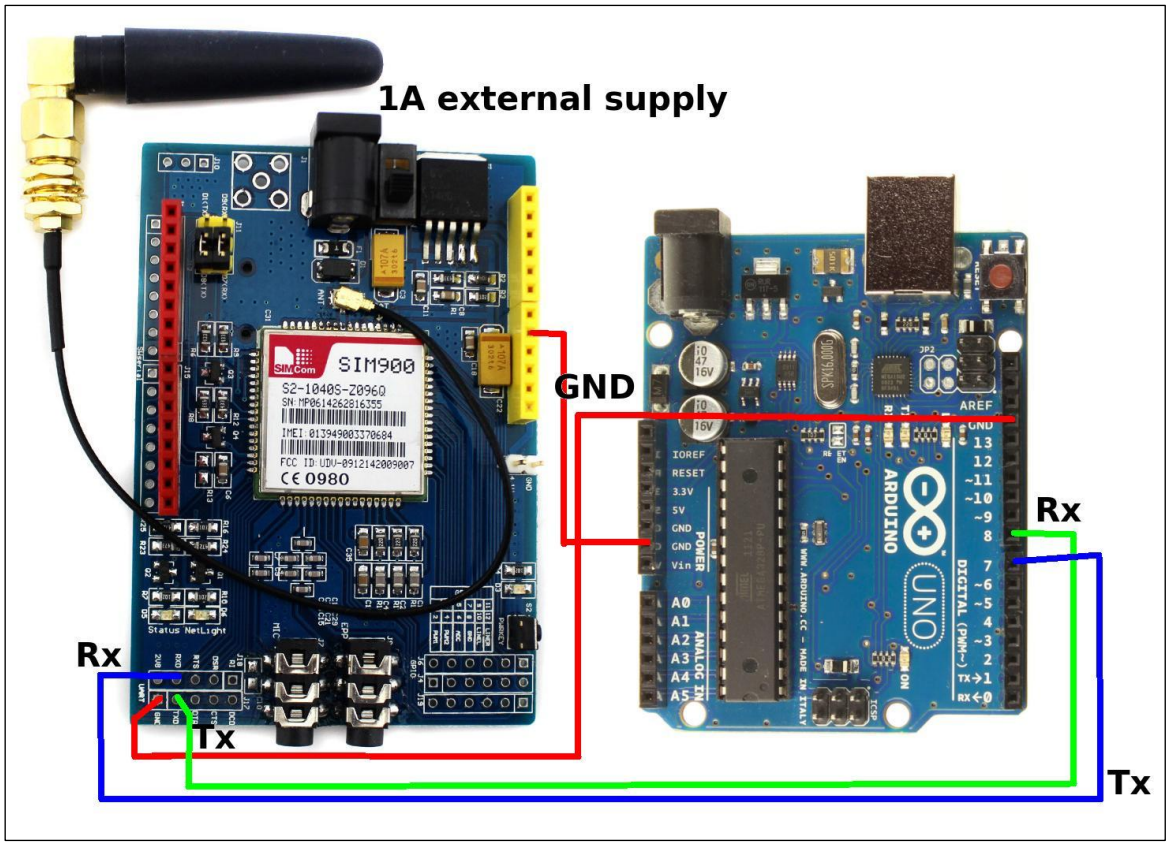


Figura 1.9 – Conexión Arduino UNO con el Shield GSM900

Fuente: (Arduino.stack, 2017)

1.5 Android

1.5.1 Conceptos generales de Android

Android es una plataforma de *software* para dispositivos móviles que incluye sistema operativo, *middleware* y aplicaciones.

El Sistema Operativo Android desarrollado por *Google Inc.* basado en el núcleo de Linux, siendo así un sistema con licencia de *software* libre diseñado para ser utilizado en dispositivos móviles como teléfonos inteligentes, *tablets*, entre otros (Baz, Ferreira, Álvarez, & García).

Android incluye un conjunto de aplicaciones básicas incluyendo un cliente de correo electrónico, un programa SMS, un calendario, mapas, navegador, contactos y otras funciones.

Las aplicaciones Android se desarrollan mediante una variación del lenguaje de programación Java, utilizando su propia máquina virtual llamada *Dalvik*, una variación de

la máquina virtual de Java encargada de interpretar el código generado por el programa y ejecutarlo. Para compilar el lenguaje Java se utiliza el Android SDK no compatibles con Java ni viceversa (Blanco, Camarero, Fumero, Werterski, & Rodríguez, 2016).

Cada aplicación de Android se ejecuta en su propio proceso, con su propia instancia de la máquina virtual *Dalvik. Dalvik* se ha escrito para que un dispositivo pueda ejecutar múltiples máquinas virtuales de manera eficiente. La máquina virtual *Dalvik* ejecuta archivos en el formato *Dalvik Executable* (.dex), que está optimizado para una huella de memoria mínima. (Blanco, Camarero, Fumero, Werterski, & Rodríguez, 2016)

1.5.2 Arquitectura

Android posee una arquitectura por cuatro capas o niveles quienes se relacionan entre sí facilitando a un desarrollador la creación de aplicaciones.

- Núcleo Linux: Sirve como base de la pila de *software* encargado de las funciones básicas del sistema.
- Bibliotecas: Contiene bibliotecas de bajo nivel en C y C++ para diferentes funciones como: SQLite para permanencia de datos, OpenGL ES para administración de gráficos y Webkit como Navegador Web embebido.
- Entorno de Aplicación: Dividido en subsistemas utilizados para que los desarrolladores tengan acceso completo a los distintos APIs del *Framework* simplificando la reutilización de componentes. Incluyendo sistemas de vistas para manipular la interfaz gráfica de usuario de diferentes aplicaciones.
- Aplicaciones: Son aplicaciones base que contiene nuestro sistema las cuales a su vez puedes ser utilizadas por otras aplicaciones (Blanco, Camarero, Fumero, Werterski, & Rodríguez, 2016).

En la Figura 1.10 se observa la arquitectura de Android, sus respectivas capas con sus respectivos parámetros.

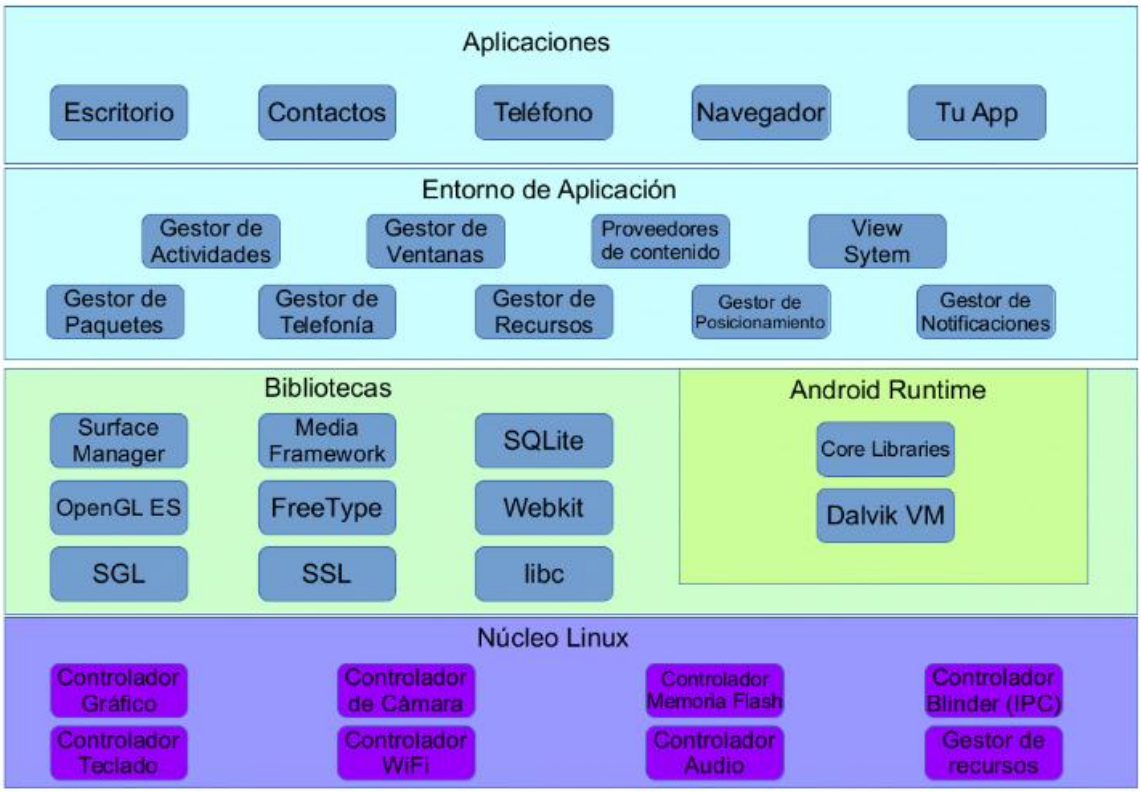


Figura 1.10- Arquitectura de Android

Fuente: (Romero, 2017)

1.5.3 Estructura de una aplicación Android

Una aplicación Android está conformada por cuatro componentes para facilitar la reutilización de código y agilitar el proceso de desarrollo. A continuación, se va a describir cada uno de estos componentes.

1.5.3.1 Actividades

Son las interfaces graficas de usuario con el propósito de que el usuario pueda realizar alguna acción como: escribir un texto, visualizar el mapa, hacer una llamada entre otras. Cada aplicación consta de una o varias actividades las cuales pueden ser invocadas una hacia otra.

Cada actividad tiene un ciclo de vida y en cada uno de sus procesos tienen asociados una determinada acción como se muestra en la Figura 1.11.

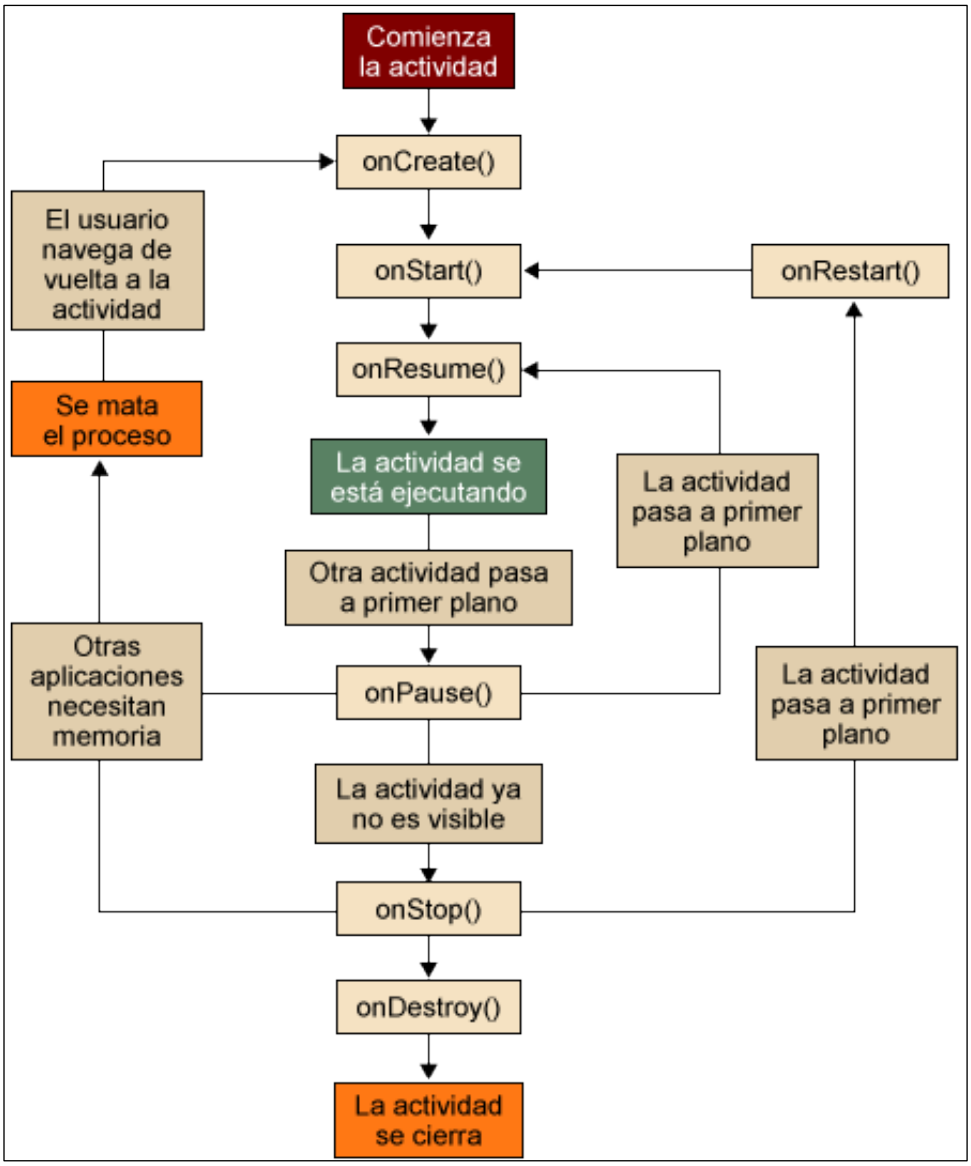


Figura 1.11 - Ciclo de vida de Activity en Android

Fuente: (UOC, 2017)

1.5.3.2 Servicios

Son los encargados de realizar tareas en segundo plano, no poseen de una interfaz gráfica. Entre los servicios se pueden mencionar ejemplos como: recibir notificaciones, actualizar datos sin necesidad de que el usuario interactúe directamente en ningún momento.

1.5.3.4 Proveedores de Contenidos

A través de este componente una aplicación puede poner a disposición sus datos para que puedan ser utilizados por otras aplicaciones las cuales pueden consultar,

modificar o eliminar dichos datos, estos datos pueden ser almacenados en una base de datos SQLite, en la web o en otros lugares donde las otras aplicaciones puedan ser accedidas.

1.5.3.5 Receptores de Eventos

Este componente escucha en segundo plano a que se produzcan determinados eventos y reaccionar ante ellos como, por ejemplo: la llegada de un correo, alerta de batería baja etc.

1.6 App Inventor

App Inventor es una plataforma de programación visual de "bloques" para crear aplicaciones móviles para dispositivos inteligentes basados en Android.

La interfaz de usuario se basa en una idea de desarrollo por pisos y techos y consta de las siguientes partes (Wolber).

- Diseñador.
- Editor de bloques.
- Compilador.
- Aplicación para la depuración.

1.6.1 Diseñador

Es utilizado para bosquejar la interfaz gráfica de usuario seleccionando los componentes que se utilizará tanto en pantalla como fuera de ella en la aplicación. Se lo puede observar en la Figura 1.12.

Figura 1.12 - Interfaz gráfica de usuario modo diseñador

Fuente: (Designers and Blocks Editor, 2015)

- Paleta: En esta sección se encuentran los componentes que pueden ser utilizados por la aplicación. Para incluirlos solo es necesario seleccionar un componente y arrastrarlo hacia el visor.
- Visor: En esta sección se muestra el diseño con el componente visible y no visible que tendrá la aplicación.
- Propiedades: En esta sección se muestra las propiedades del componente seleccionado para poder modificarlas, por ejemplo, color, tamaño, comportamiento entre otras.

1.6.2 Editor de bloques

Es utilizado para establecer el comportamiento de la aplicación mediante los bloques de construcción los cuales son elementos comunes con la interfaz de usuario junto con las características de nuestro dispositivo. En la Figura 1.13 se pueden observar las diferentes partes de esta interfaz.

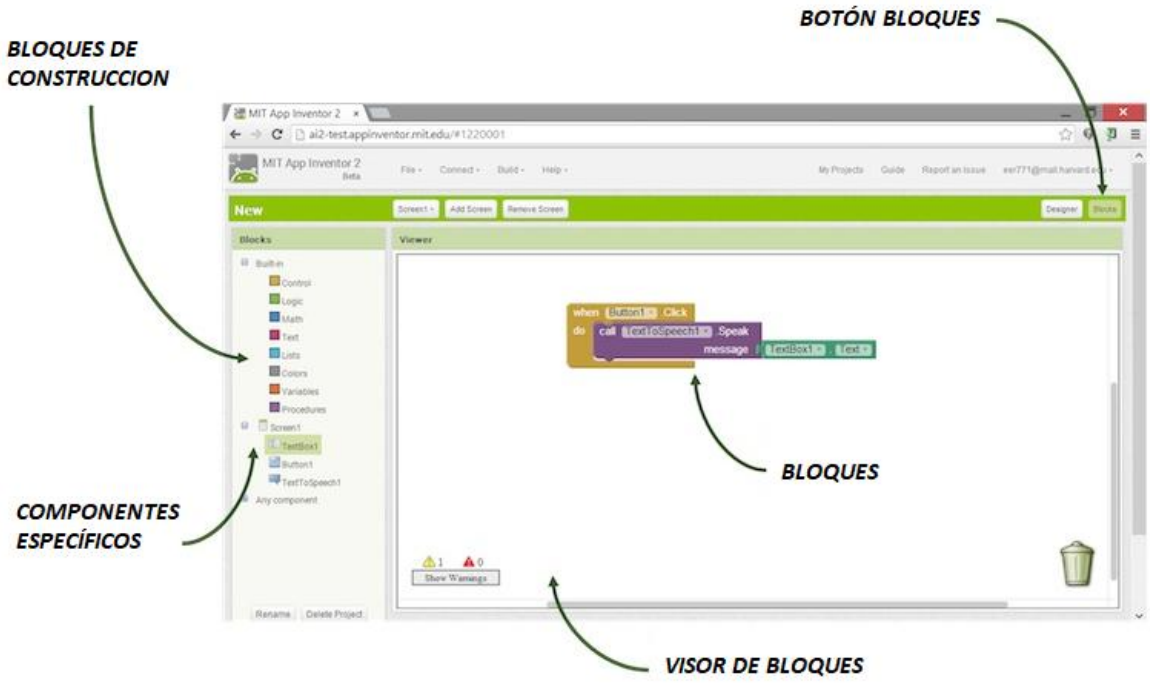


Figura 1.13 - Editor de Bloques

Fuente: (Designers and Blocks Editor, 2015)

- Bloques de construcción: En esta sección se encuentran los bloques de comportamiento que son utilizados generalmente en una aplicación.
- Componentes específicos: En esta sección se encuentran bloques de comportamiento menos utilizados en aplicaciones Android.
- Visor de Bloques: En esta área es donde se arrastra los bloques para crear relaciones y comportamiento que tendrá nuestra aplicación.
- Bloque: Establece el comportamiento de la aplicación y sus componentes mediante la unión de diferentes bloques.

1.6.3 Programación en App Inventor

App Inventor utiliza el lenguaje de programación por bloques y orientado a eventos lo que proporciona una facilidad de aprendizaje debido a que elimina la mayoría de código escrito reduciendo dramáticamente las frustraciones que tienen con la sintaxis de las personas que están iniciándose en el desarrollo de Apps proporcionando señales visuales reduciendo así la posibilidad de errores (Wolber).

El lenguaje funciona definiendo el comportamiento de una aplicación mediante un conjunto de manejadores de eventos, es decir cuando se produzca el evento X realice la acción Y.

Una vez completada la aplicación es posible implementarla empaquetándola como .apk o generando un código QR el cual nos permitirá descargar nuestra aplicación e instalarla en un dispositivo.

CAPÍTULO 2

2. PROPUESTA

2.1 Introducción

En este capítulo se muestra el diseño del sistema prototipo propuesto para el cual se ha considerado el uso de herramientas *hardware* y *software* libre porque cumplen con las necesidades para resolver el problema planteado anteriormente.

2.1.1 Diagrama de casos de uso

Para el diseño del diagrama de casos de uso se consideró los requerimientos del usuario identificado para la solución del problema. En la Figura 2.1 se presenta el diagrama de casos de uso correspondiente al sistema.

El diagrama de la Figura 2.1 muestra las funciones que poseen cada uno de los actores que intervienen en el sistema.

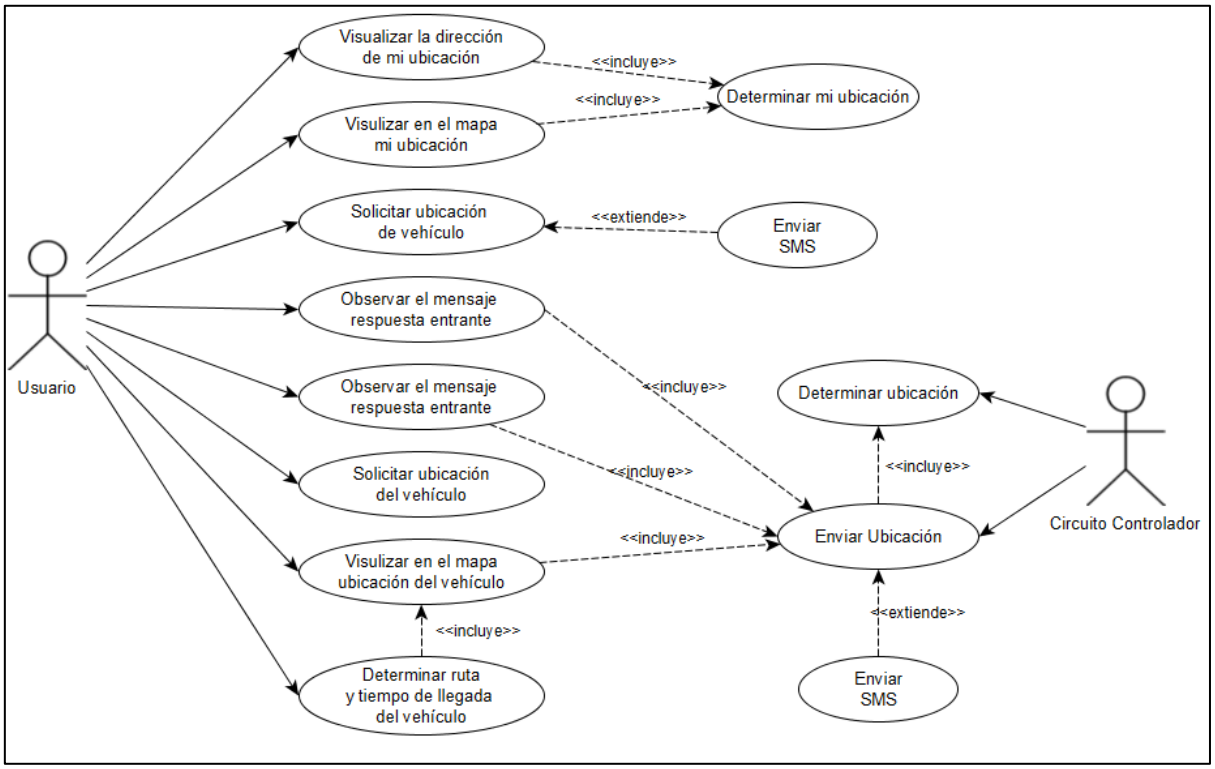


Figura 2.1 - Diagrama de casos de uso del Sistema.

Fuente: Elaborado por el autor

El sistema consta con dos componentes: un circuito controlador y una aplicación Android que se describirá detalladamente en la siguiente sección.

2.3.1.1 Circuito controlador

El circuito controlador consta de una placa Arduino Uno porque tiene las funciones y componentes necesarios para cumplir con los objetivos planteados, un módulo SIM900 que nos permite comunicarnos mediante la red GSM entre la placa y el *smartphone* y un módulo GPS6MV2 el cual nos permite realizar lecturas GPS. El circuito será instalado dentro de un vehículo el cual realizará lecturas de su posición mientras espera una solicitud de ubicación mediante mensajes de texto. Una vez recibida la solicitud este componente responde con un mensaje de texto indicando los datos de sus coordenadas actuales.

2.3.1.2 Aplicación Android

Este segundo componente podrá determinar la ubicación del *smartphone*, hacer una solicitud de ubicación hacia el circuito controlador, leer la respuesta del circuito. Estas funciones nos permiten determinar la ubicación del vehículo, mostrar la ruta y el tiempo de llegada desde su ubicación hacia la ubicación del *smartphone*.

2.1.2 Diagrama de procesos del sistema

En la Figura 2.2 se detalla los procesos generales que realizará cada uno de los componentes del sistema.

Se han determinado dos actores el *smartphone* y el circuito que se ubica en el vehículo. Las funciones del circuito controlador son de determinar la ubicación mediante la lectura del GPS, la espera a una solicitud y el envío de la ubicación en respuesta, y las funciones del *smartphone* son de determinar la ubicación, solicitar la ubicación del circuito, mostrar la ubicación determinar la ruta y el tiempo de llegada del circuito dichas funciones se encuentran detallas en la Figura 2.2.

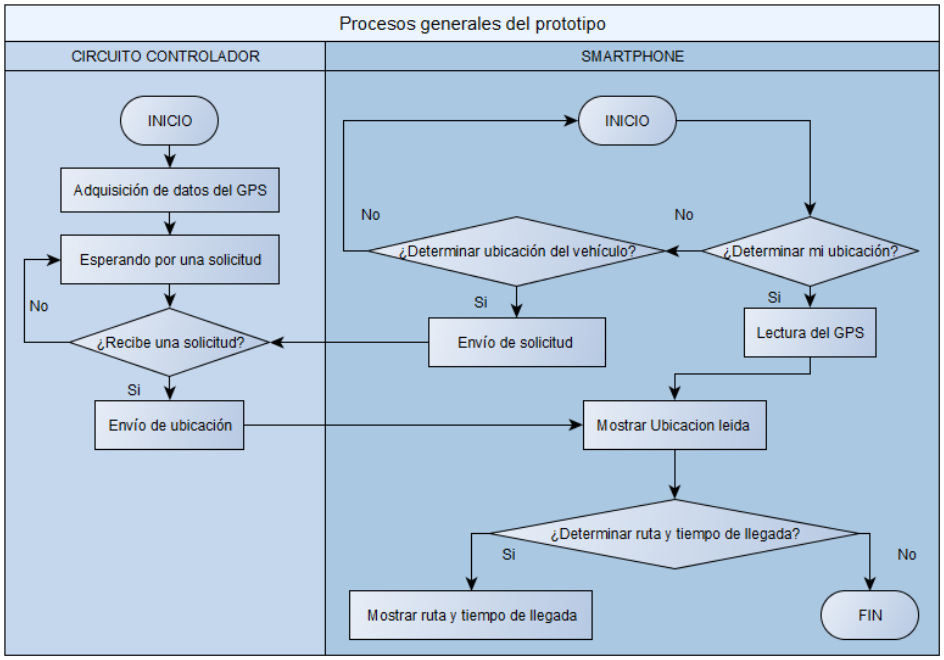


Figura 2.2 - Diagrama de procesos del Sistema

Fuente: Elaborado por el autor

2.2 Diseño del Sistema

Una vez identificados los componentes y procesos necesarios para el diseño del sistema prototipo usando el diagrama de casos de uso y el diagrama de procesos, los cuales tienen como objetivo determinar la ubicación y el tiempo de llegada de un vehículo mediante el uso de un circuito controlador ubicado dentro del vehículo y una aplicación para el sistema operativo Android dentro de un *smartphone*, en la Figura 2.3 se presenta el diagrama del sistema prototipo que consta de un circuito controlador conectado a un sensor GPS y una tarjeta SIM900 juntamente con una aplicación Android para un *smartphone*.

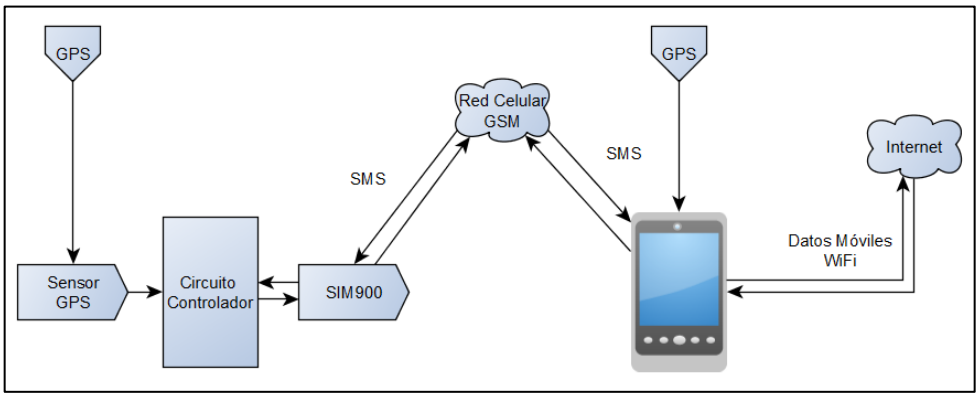


Figura 2.3 - Diagrama de bloques del Sistema

Fuente: Elaborado por el autor

2.2.1 Circuito Controlador

Es la parte del sistema a la cual se le considera el cerebro del prototipo, debido a que desde éste realiza el control de casi todas las actividades desarrolladas. Además, permite la interconexión de los diferentes componentes. Sus dos componentes principales son el software y el hardware del prototipo

2.2.1.1 Hardware

Para la descripción del *hardware* se utilizará el diagrama de bloques de la Figura 2.4, en el cual se observa la interconexión de las diferentes etapas del prototipo.

2.2.1.1.1 Diagrama de bloques

Este componente del sistema posee cuatro bloques principales los cuales se muestran en la Figura 2.4. A continuación se detalla cada uno de los componentes.

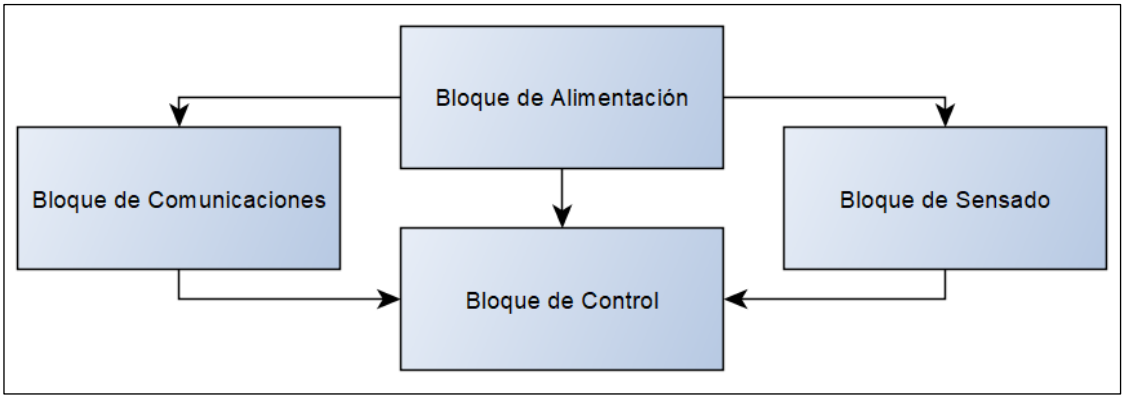


Figura 2.4 - Diagrama de bloques del Hardware del Circuito Controlador

Fuente: Elaborado por el autor

2.2.1.1.1.1 Bloque de Alimentación

Es el bloque encargado de proveer de energía eléctrica para el funcionamiento del circuito que se alimenta con 5 voltios. El circuito consta con dos entradas para la alimentación, la primera entrada es USB Tipo B hembra y la segunda entrada es de un adaptador de 5 milímetros para corriente continua integradas en la placa Arduino Uno.

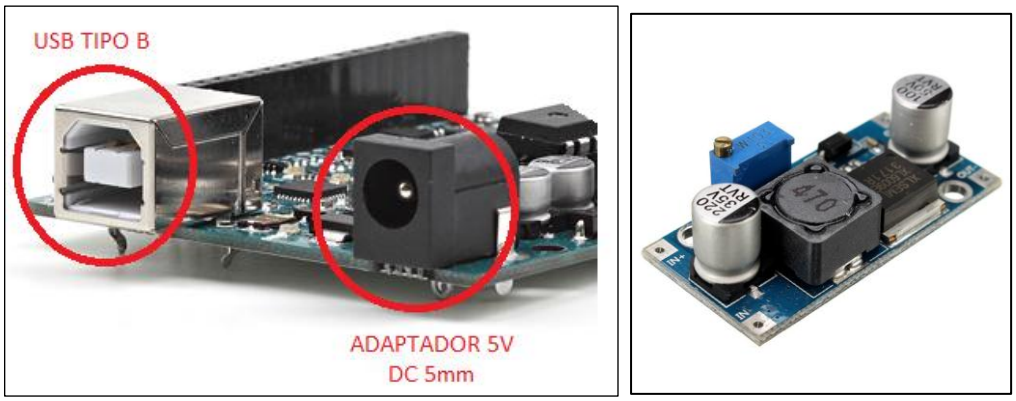


Figura 2.5 - Bloque de Alimentación

Fuente: Elaborado por el autor

Para la conexión de la energía eléctrica del bus se utilizará un módulo de regulación de voltaje DC-DC ajustable con salida de 5V mediante el circuito integrado modelo CN6009 mostrado en la Figura 2.5. El integrado mantiene un nivel de voltaje de salida regulado a 5V, valor que corresponde al de operación óptima de la placa Arduino razón por la que se eligió dicho integrado.

2.2.1.1.1.2 Bloque de Comunicaciones

Es la parte encargada del intercambio de mensajes solicitud/respuesta el cual se lo realizara mediante mensajes de texto GSM utilizando el módulo SIM900 para responder a las peticiones del otro componente del sistema. Se eligió este módulo porque es compatible con Arduino Uno y es ideal para mandar mensajes de texto a celulares. Como se observa en la Figura 2.6.

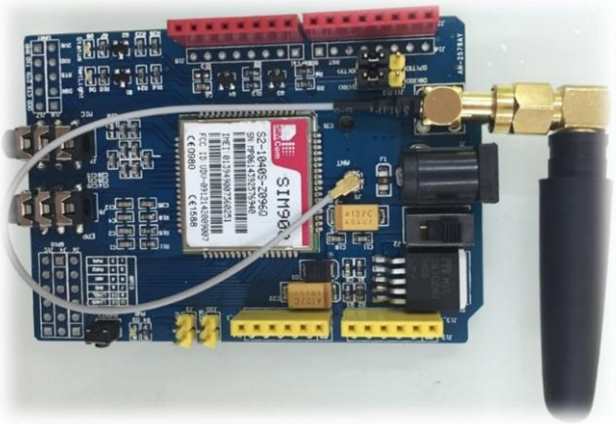


Figura 2.6 - Bloque de comunicaciones - Módulo Shield GSM

Fuente: (HETPRO, 2017)

2.2.1.1.1.3 Bloque de Sensado

Es el encargado de la determinación de la ubicación para lo cual se utilizará el sistema GPS mediante el módulo GPS6MV, se lo observa en la Figura 2.7, el cual realizará las lecturas de latitud y longitud de su ubicación. Este módulo fue elegido debido a la facilidad de adquisición y que consta con las funciones necesarias para el presente proyecto.

Figura 2.7 - Bloque de Sensado - Modulo GPS6MV2

Fuente: (Inteligencia artificial, 2017)

2.2.1.1.1.4 Bloque de Control

Es el encargado de controlar los diferentes módulos para tener el correcto funcionamiento del circuito. Para este bloque se utilizará el microcontrolador ATmega328P porque se encuentra incorporado dentro de la placa Arduino Uno, el mismo se observa en la Figura 2.8.

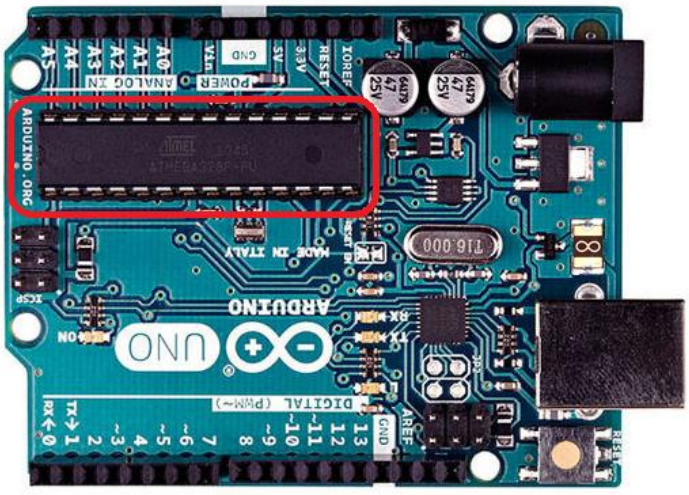


Figura 2.8 - Bloque de Control – Placa Arduino Uno

Fuente: (LinuxGizmos.com, s.f.)

2.2.1.1.5 Diagrama de Circuito

A continuación, en la Figura 2.9 se muestra las conexiones implementadas entre los distintos módulos para su correcto funcionamiento. Las conexiones se realizaron de acuerdo con el manual de Arduino y de los módulos correspondientes.

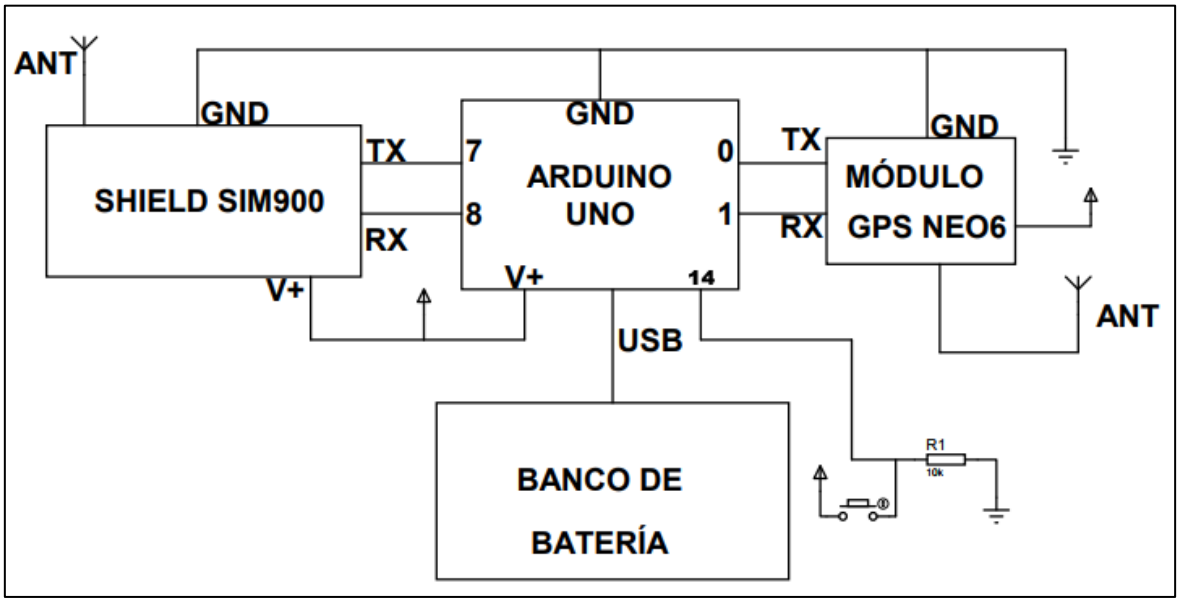


Figura 2.9 - Diagrama de conexión del Circuito Controlador

Fuente: Elaborado por el autor

2.2.1.2 Diseño de Software

2.2.1.2.1 Diagrama de flujo del *software* del circuito controlador

En la Figura 2.10 se presenta el diagrama de flujo del programa del circuito controlador dentro del Arduino Uno, el cual obtiene la posición y espera una solicitud para posteriormente responder con la ubicación.

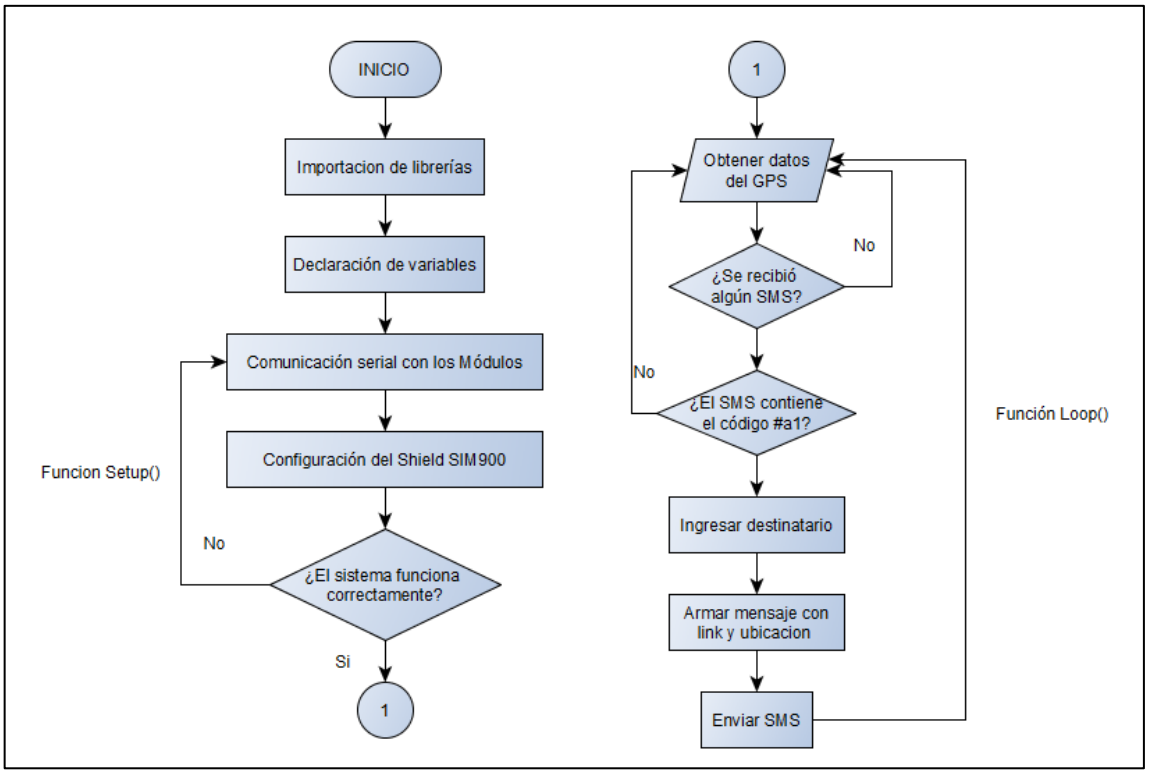


Figura 2.10 - Diagrama de flujo del circuito controlador.

Fuente: Elaborado por el autor

2.2.2 Aplicación móvil

Este componente del sistema requiere de componentes tanto de *hardware* como de *software* los cuales se irán detallando a continuación.

2.2.2.1 Hardware

Para la parte del *hardware* se utilizará un teléfono celular inteligente con sistema operativo Android y que cuente con un sensor GPS incluido.

2.2.2.2 Software

El *software* implementado es una aplicación móvil desarrollada en el entorno de desarrollo App Inventor utilizando un lenguaje de programación por bloques ya que es fácil de aprender e implementar.

2.2.2.2.1 Diagrama de flujo general de la aplicación

En esta sección se presenta en la Figura 2.11 el diagrama de flujo que describen todos los procesos importantes que realizará la aplicación.

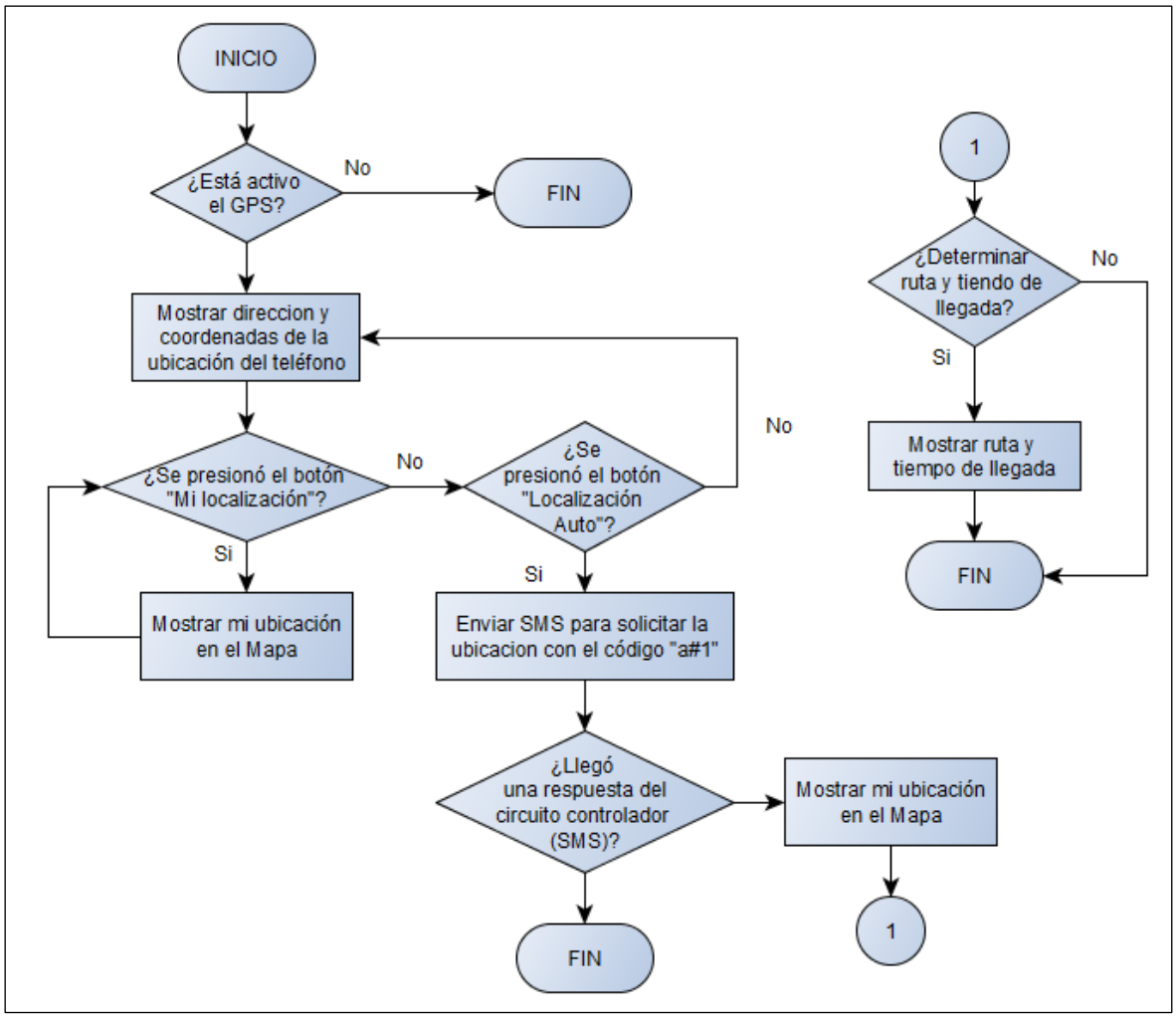


Figura 2.11 - Diagrama de flujo de la aplicación móvil.

Fuente: Elaborado por el autor

Una vez identificados los procesos necesarios con la ayuda de la Figura 2.11 a continuación se muestra el diseño de la interfaz gráfica.

2.2.2.2.2 Diseño de la interfaz gráfica para la aplicación móvil

Una interfaz gráfica contempla las distintas acciones que el usuario puede realizar, para esto el diseño visual tiene que ser simple y amigable para el usuario para que este pueda realizar las acciones de manera intuitiva y familiarizarse rápidamente con la aplicación.

El diseño de la interfaz gráfica se muestra en la Figura 2.12 y sus componentes se detallan a continuación en la Tabla 2.1.

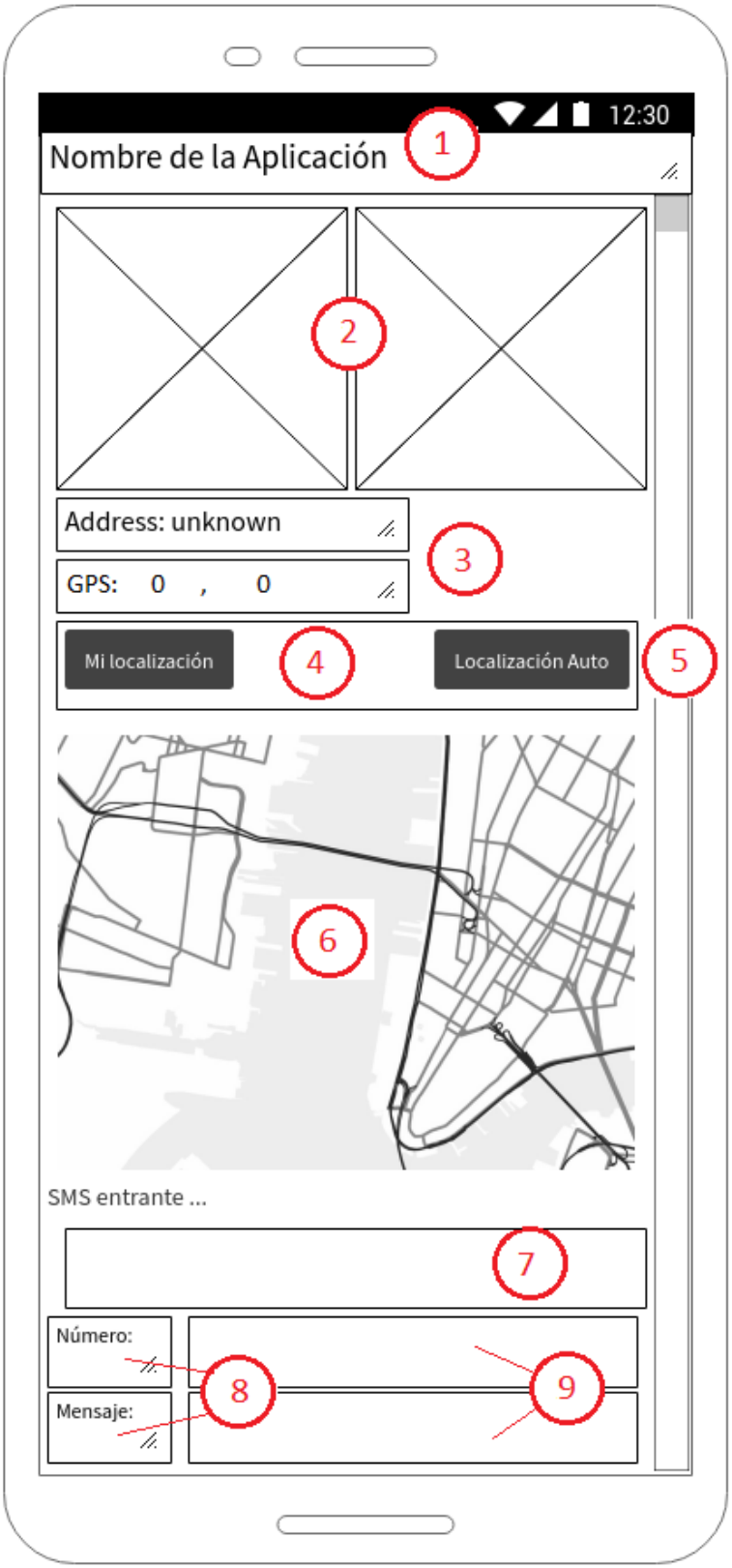


Figura 2.12 - Boceto de la interfaz gráfica de la aplicación móvil.

Fuente: Elaborado por el autor

Tabla 2. 1 - Descripción de la interfaz gráfica de la aplicación móvil.

Identificador	Descripción	Tipo de Objeto	Id de Objeto
1	Muestra el nombre de la aplicación	Toolbar	
2	Muestra imágenes de presentación de la aplicación	ImageView	HorizontalArrangement12 HorizontalArrangement13
3	Etiqueta que muestra la dirección y coordenadas GPS del teléfono	Label	AddressLabel GPSLabel
4	Botón para mostrar mi ubicación en el mapa	Button	Button1
5	Botón para mostrar la ubicación del vehículo en el mapa	Button	Button2
6	Visor Web utilizado para la visualización del mapa.	WebViewer	WebViewer1
7	Texto donde se muestra el mensaje de respuesta del circuito controlador	TextBox	TextBox2
8	Etiqueta para identificar los TextBox.	Label	Label4 Label5
9	Textos que pueden ser modificados por el usuario donde va el número de teléfono asociado al circuito controlador y el mensaje a enviar	TextBox	TextBox3 TextBox4

Fuente: Elaborado por el autor

La Tabla 2.1 está compuesta de la columna identificador la cual es un número que nos permite asociar los elementos de la Figura 2.12 con sus respectivos atributos mostrados, la columna descripción que describe brevemente las funciones de cada componente, la tercera columna Tipo de objeto indica que control proporcionado por App Inventor será utilizado, y la última columna Id de Objeto indica un nombre único para cada control que sirve para asociar el diseño con la parte lógica. Para el cálculo de la distancia esta la realiza directamente la API de Google incorporada dentro del servicio gratuito de mapas.

CAPÍTULO 3

4. IMPLEMENTACIÓN

3.1 Desarrollo

El desarrollo del prototipo consta de varias etapas que se describirán a continuación en las siguientes secciones del capítulo. Para comenzar se extraerá los requisitos de un producto de *software*, ya que esta es la primera etapa para crearlo.

3.1.1 Requerimientos funcionales

En los requerimientos funcionales se encuentran los requerimientos que debe cumplir el prototipo desarrollado, cuando ya se encuentre en funcionamiento. A continuación, se realiza una lista de estos.

- La Aplicación Android debe permitir conocer la ubicación del bus a un tiempo determinado.
- La información con las coordenadas de la ubicación deberá ser enviada a través de mensajes de texto.
- La aplicación debe permitir la visualización de la ubicación del dispositivo móvil para poder determinar la distancia y tiempo de llegada del bus.
- La aplicación debe ser capaz de determinar en el mapa la ubicación del bus.
- Se debe determinar el tiempo aproximado de llegada del bus.
- La aplicación debe mostrar la ruta por la cual el transporte llegará a su destino.
- El prototipo utilizará los mapas de Google porque son los más completos actualmente.
- La aplicación debe mostrar la ruta del bus.
- No se podrá localizar el bus si no se dispone de saldo o mensajes disponibles en las dos tarjetas SIM que se encuentran operando tanto del circuito como el del *smartphone*.
- El prototipo no podrá localizar el autobús si éste se encuentra fuera de la cobertura de red celular de la operada correspondiente.
- El prototipo no podrá transmitir imágenes, ni tampoco videos con la aplicación
- La aplicación no dispondrá de la opción de intercambiar audio con el circuito controlador.

3.1.2 Requerimientos no funcionales

Los requerimientos no funcionales son aquellos que no interfieren con la funcionalidad del sistema.

- La aplicación móvil debe ser desarrollada para el sistema operativo Android.
- El sistema debe ser realizada con componentes de *software* y *hardware* libre.

3.1.3 Consideraciones importantes para el desarrollo de la aplicación.

Para el desarrollo de la aplicación es importante tener en cuenta los siguientes aspectos:

- El GPS para su correcto funcionamiento necesita tener línea de vista con los satélites, caso contrario no se efectuará correctamente la lectura satelital, lo cual ocasionará falta de precisión en la ubicación, además de que la toma de datos tomará mucho más tiempo.
- Se debe revisar en la interfaz gráfica de la aplicación que el número telefónico asignado, sea el correspondiente a la tarjeta SIM que se encuentra en el shield SIM900.
- En caso de que no se reciba inmediatamente la ubicación, se deberá movilizar el dispositivo móvil a otro lugar para que la recepción de señal celular y GPS sean óptimas.

3.1.4 Cronograma de actividades

En el Anexo 3 detallan las actividades a realizar y se estima el tiempo aproximado que cada una debe cumplir. Las actividades han sido establecidas de acuerdo con la planificación para el cumplimiento de objetivos del proyecto.

3.2.1 Implementación del hardware del circuito controlador

La implementación se la realizo a partir de la placa principal Arduino Uno siguiendo el diagrama de circuito realizado en el diseño en la Figura 3.1.

3.2.1.1 Materiales Utilizados

Uno de los elementos más importantes para la implementación del prototipo es un módulo de Arduino, para este caso se ha decidido utilizar Arduino UNO, debido a la gran compatibilidad que tiene con varios sensores y módulos. Esto se debe a que es una de las placas con un microcontrolador que más desarrollo ha tenido.

La Placa Arduino Uno, utilizada en la elaboración del circuito controlador se observa en la Figura 3.1

Figura 3.1 - Arduino Uno Utilizado

Fuente: Elaborado por el autor

Para la interacción con la red de telefonía celular se utilizó un Módulo Shield GSM – SIM900, el cual permite la conexión a la red GSM y GPRS.

En la Figura 3.2 se puede observar el shield utilizado para la implementación del prototipo.

Figura 3.2 - Shield GSM – SIM900 utilizado

Fuente: Elaborado por el autor

Para tener acceso al sistema de localización, es importante contar con receptor GPS, para el prototipo se utilizó el Módulo GPS6MV2 que se muestra en la Figura 3.3, el cual cuenta con una antena de cerámica para la recepción y transmisión de señales GPS.

Figura 3.3 - Módulo GPS6MV2 utilizado

Fuente: Elaborado por el autor

Los cables de conexión son importantes para la comunicación de los distintos módulos con la placa de Arduino UNO. En la Figura 3.4 se encuentran los cables de cobre utilizados, con terminales macho a un lado y hembra en el otro lado.

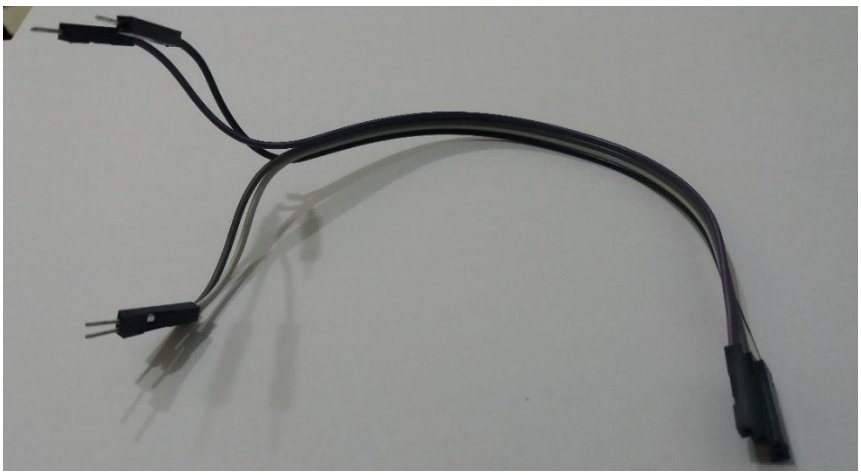


Figura 3.4 - Cables utilizados

Fuente: Elaborado por el autor

Los espadines son importantes para poder conectar adecuadamente el shield GSM900 con la placa de Arduino UNO, y de esta manera darle un soporte fijo al circuito controlador.

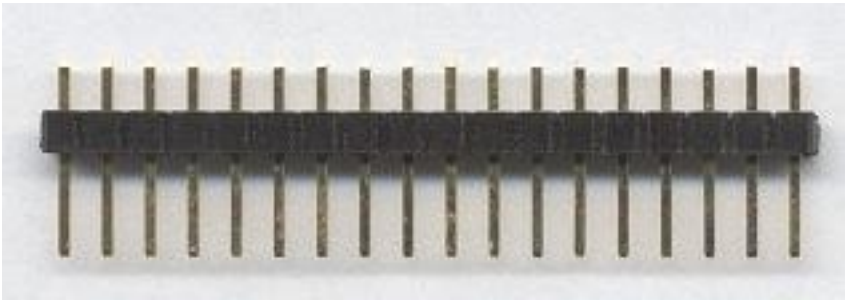


Figura 3.5 - Espadines

Fuente: (Educa Madrid, 2017)

Primero se soldaron los espadines en la placa del ShieldGSM-SIM900 en los pines: 0, 1, 7, 8, 9 y en todo el bloque J14-1 para conectarlo con la placa Arduino Uno en los pines respectivos: 0, 1, 7, 8, 9 y en los primeros 6 entradas del bloque Power respectivamente.

Una vez conectadas estas dos placas se procedió a conectar mediante los cables de conexión, los cuatro pines del módulo GPS6MV2 con los pines respectivos del Arduino uno: transmisión (Tx), recepción (Rx), alimentación (Vcc) y tierra (Gnd).

3.2.2 Implementación del software del circuito controlador

Para la escritura, compilación y carga del programa se utilizó el entorno de desarrollo Arduino IDE con el lenguaje de programación C.

3.2.2.1 Programa Principal

Para iniciar la programación fue necesario incluir ciertas librerías, declarar, definir el tamaño y tipo de variables que fueron utilizadas dentro de nuestro programa.

El código 3.1 del archivo Arduino es responsable de la declaración de las variables que se utilizaron en el programa y de la importación de las librerías:

- *SoftwareSerial*, la cual permite la comunicación serial entre el Arduino y los módulos SIM900 y GPS6MV2 con la placa Arduino a través de los pines 0, 1, 7, 8

- *TinyGPS*, la cual proporciona la funcionalidad de receptor GPS para obtener los distintos valores que el módulo puede obtener como: posición, fecha, hora, altitud, velocidad y curso.

```
TinyGPS gps;                //Declaramos el objeto gps
SoftwareSerial sim900(7,8); //Declaramos el pin 7 rx y 8 tx

//Declaracion de variables para la obtención de datos
int year;
byte month, day, hour, minute, second, hundredths;
unsigned long chars;
unsigned short sentences, failed_checksum;

int bandera=0;
int led=13;
int dato;
char inchar;

//el siguiente bloque define variables de diferente tamaño, sirven para recibir datos seriales
#define uint8 unsigned char
#define uint16 unsigned int
#define uint32 unsigned long int
#define DEBUG_ENABLED 1
char comando;

int x;
int suiche=11;
int g;
```

Código 3. 1 - Declaración y asignación de Variables e Importación de librerías

Fuente: Elaborado por el autor

3.2.2.1.1 Función Setup

El código 3.2 muestra la configuración básica del Arduino Uno para la comunicación con los dos módulos, el encendido y configuración del módulo SIM900 para el envío de mensajes de texto.

```
void setup()
{
  sim900.begin(19200);  //Iniciamos el puerto serie para la comunicacion con el SIM900
  Serial.begin(9600);   //Iniciamos el puerto serie para la comunicacion con el GPS
  SIM900power();        //Encendemos el SIM900
  delay(10000);         //Relizamos un retardo

  for (g = 0;g<10; g++)
  {
    sim900.print("AT+CSCLK=0\r");
    delay(10);
  }

  pinMode(led,OUTPUT);
  digitalWrite(led,LOW);
  pinMode(suiche,INPUT);

  sim900.print("AT+CMGF=1\r");          //Se configura el modo para el envio de SMS
  delay(100);
  sim900.print("AT+CNMI=2,2,0,0,0\r");  // borra el contenido del nuevo SMS al recibirlo en la salida serie del shield-GSM
  delay(100);

  sim900.print("ATe0&w\r");             //instruccion para habilitar el eco sim900
}
```

Código 3.2 - Configuración Básica de los módulos del circuito.

Fuente: Elaborado por el autor

3.2.2.1.2 Método para el encendido del SIM900

El código 3.3 muestra el método *SIM900power()* el cual es utilizado para realizar la función del botón *power* del módulo SIM900 el cual enciende el circuito mediante instrucciones de software.

```
void SIM900power()
// software equivalente de preionar el boton power del ShieldGSM
{
  digitalWrite(9, HIGH);
  delay(1000);
  digitalWrite(9, LOW);
  delay(5000);
}
```

Código 3.3 - Método para el encendido del módulo GSM

Fuente: Elaborado por el autor

3.2.2.1.3 Función Loop

El código 3.4 corresponde a la función *loop* que permite leer los datos del GPS y esperar la solicitud para recibir un mensaje con el código #a1 para responder la solicitud con las coordenadas de la posición recibida del GPS.

```
void loop()
{
  leergps();
  if(bandera==1)
  {
    digitalWrite(led,HIGH);
    enviarsms();
    digitalWrite(led,LOW);
    bandera=0;
  }

  if(sim900.available() >0)
  {
    inchar=sim900.read();
    if (inchar=='#')
    {
      delay(10);
      inchar=sim900.read();
      if (inchar=='a')
      {
        delay(10);
        inchar=sim900.read();
        if (inchar=='0')
        {
          bandera=1;
        }
        if (inchar=='1')
        {
          bandera=1;
        }
      }

      sim900.println("AT+CMGD=1,4"); //borra todos los mensajes
      sim900.print("ATe0&w\r");
    }
  }
  dato=digitalRead(suiche);
  if (dato==HIGH)
  {
    panico();
  }
}
```

Código 3.4 - Programación de la función loop

Fuente: Elaborado por el autor

3.2.2.1.4 Método leergps

El código 3.5 corresponde al método leergps que permite recibir los valores de lectura del GPS mediante el puerto serial para ser almacenados en las distintas variables declaradas al inicio del código.

```
void leergps()
{
  while(Serial.available())
  {
    int c = Serial.read();
    if(gps.encode(c))
    {
      float latitude, longitude;
      gps.f_get_position(&latitude, &longitude);
      gps.crack_datetime(&year,&month,&day,&hour,&minute,&second,&hundredths);
      gps.stats(&chars, &sentences, &failed_checksum);
    }
  }
}
```

Código 3.5 - Método para la obtención de datos del GPS

Fuente: Elaborado por el autor

3.2.2.1.5 Método enviarsms

El código 3.6 corresponde al método enviarsms el cual envía instrucciones al módulo SIM900 indicando el contenido del mensaje que se quiere enviar y el número de destinatario a donde se enviará. Para este caso se enviará un mensaje que contiene un vínculo hacia Google Maps indicando las coordenadas de latitud y longitud obtenidas por el GPS.

```
void enviarsms()
{
  sim900.print("AT+CMGF=1\r");                      // Comando AT para enviar un mensaje SMSmessage
  delay(100);
  sim900.println("AT + CMGS = \"+593998045704\"");  // número de teléfono del destinatario, en formato internacional
  delay(100);
  sim900.print("http://maps.google.com/?q=");       // mensaje a enviar
  delay(100);
  float latitude, longitude;
  gps.f_get_position(&latitude, &longitude);
  sim900.print(latitude,5);
  delay(100);
  sim900.print(",");                                // mensaje a enviar
  delay(100);
  sim900.println(longitude,5);                      // mensaje a enviar
  delay(100);
  sim900.println((char)26);                         // Fin del mensaje a enviar
  delay(100);
  sim900.println();
  delay(5000);                                      // Dar tiempo al módulo para enviar SMS
}
```

Código 3.6 - Método para enviar ubicación mediante un SMS.

Fuente: Autor

Una vez escrito el código se procede con la compilación y carga del programa libre de errores tal como se muestra en la Figura 3.6.

Para compilar el programa se presiona el botón  que se encuentra en la parte superior izquierda del IDE

Figura 3.6 - Compilación correcta del código en Arduino IDE

Fuente: Elaborado por el autor

Una vez compilado el código se debe conectar el Arduino Uno vía USB al computador como se muestra en la Figura 3.7 para proceder a la carga del programa.

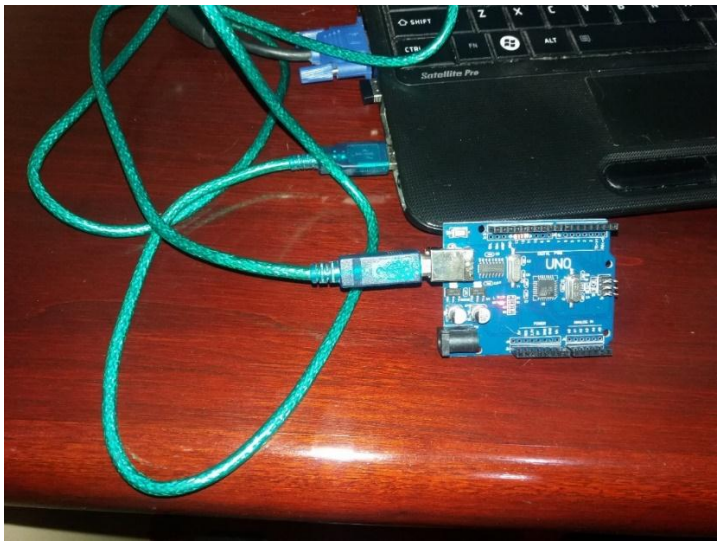


Figura 3.7 - Conexión del Arduino con el PC a través de USB.

Fuente: Elaborado por el autor

Cuando el Arduino se encuentre conectado es necesario identificar el nombre del puerto dentro del computador al cual está conectado utilizando el Administrador de dispositivos de Windows, elegimos Puertos (COM y LTP) y se identifica el nombre del puerto COM al cual se encuentre conectado.

En la Figura 3.8 se muestra un ejemplo donde se puede identificar que la placa Arduino se encuentra conectada a través del puerto COM6.

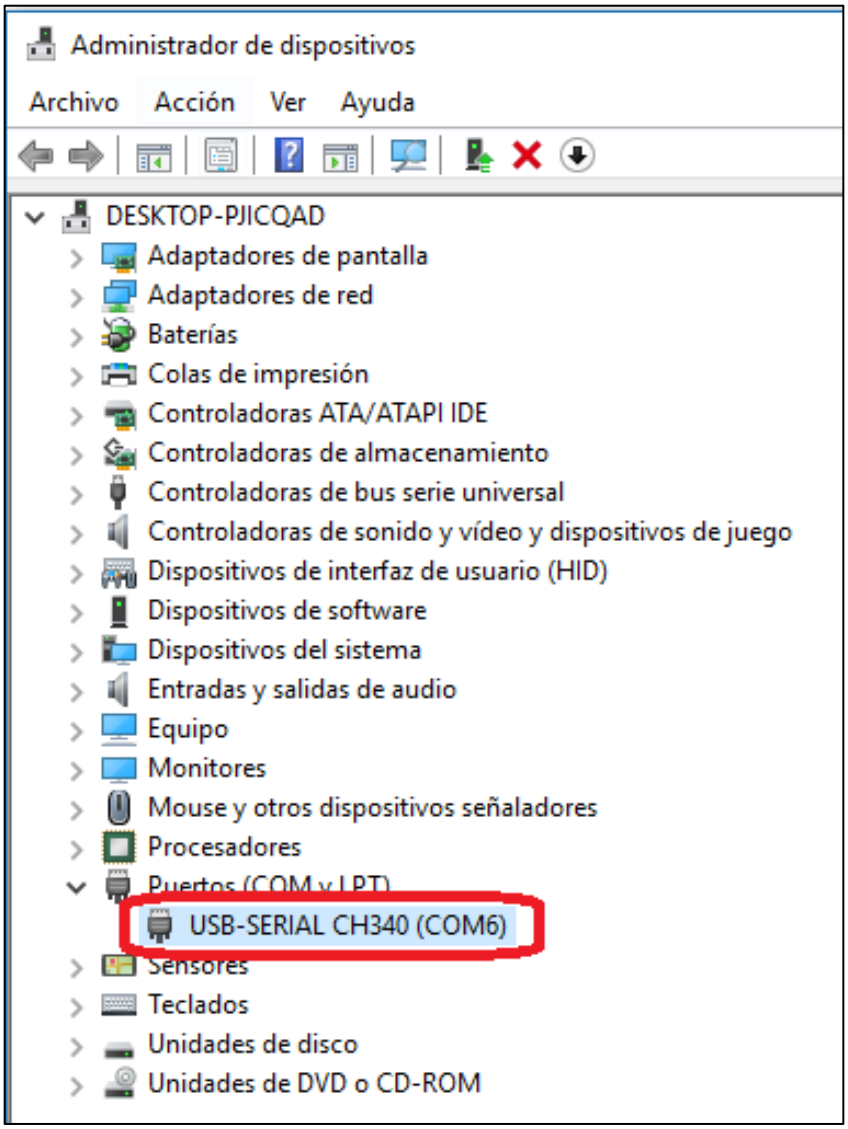


Figura 3.8 - Identificación del puerto de conexión del Arduino.

Fuente: Elaborado por el autor

Una vez compilado el programa, conectado el Arduino e identificado, se comprueba que el IDE reconozca al Arduino en el puerto de conexión. Para esto se observa

que en la esquina inferior derecha del IDE se encuentre el modelo de Arduino y el nombre del puerto como se observa en la Figura 3.9.

Para cargar el programa dentro del Arduino presionamos el botón  que se encuentra en la parte superior izquierda de nuestro IDE.

En la Figura 3.9 se muestra cuando la carga del código se está realizando. Una vez completa la carga se desconecta al Arduino para posteriormente conectarlo con todo el circuito.

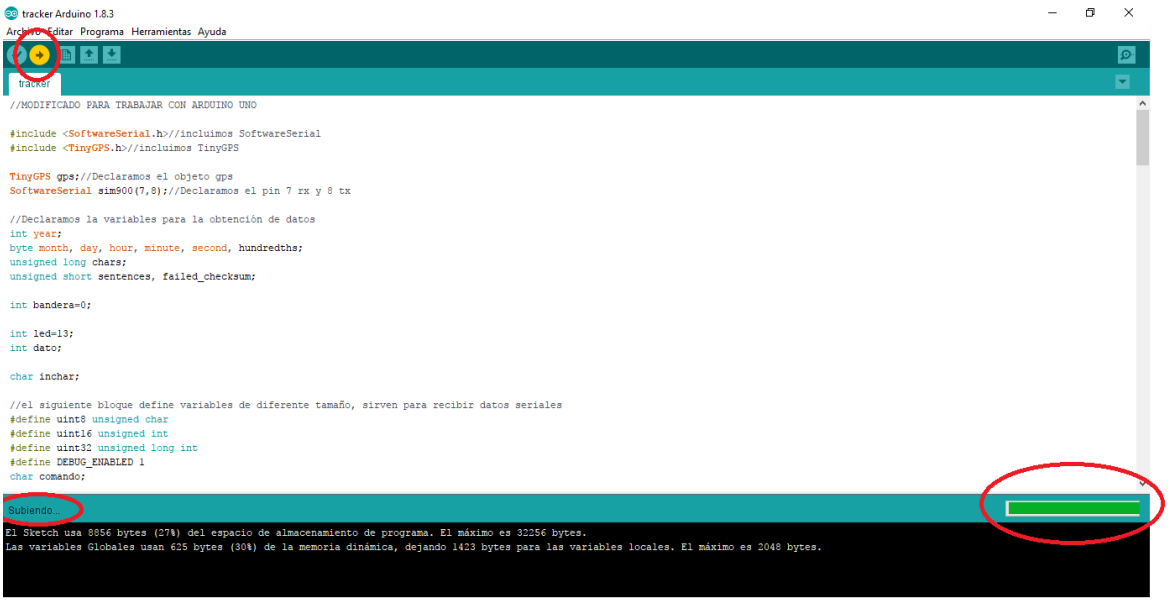


Figura 3.9 - Carga del código en el Arduino Uno

Fuente: Elaborado por el autor

3.2.3 Implementación del *software* de la aplicación móvil

Para la implementación de la aplicación móvil se utilizó el entorno de desarrollo App Inventor para lo cual se realizó la interfaz gráfica de usuario mostrada en la Figura 3.10, en base a los requerimientos del sistema.

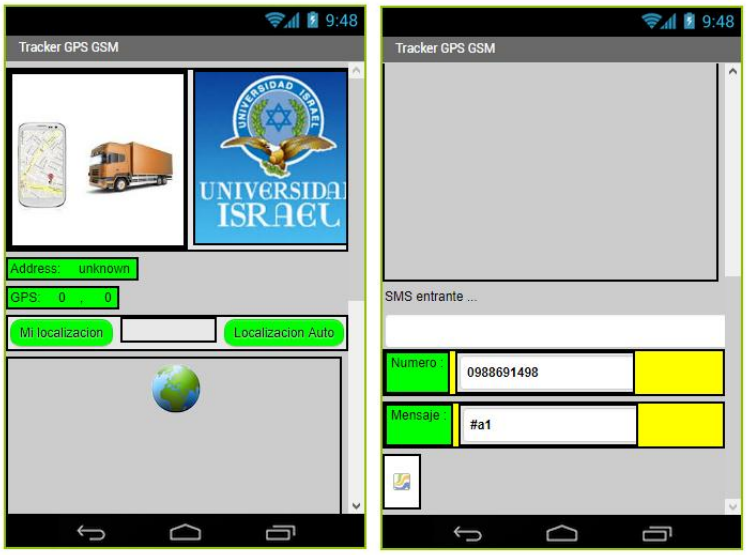


Figura 3.10 - Interfaz gráfica implementada.

Fuente: Elaborado por el autor

Las acciones que realizarán cada componente grafico del diseño es decir la parte lógica de la aplicación se la realizo mediante la programación por bloques que se detalla a continuación.

3.2.3.1 Programa principal

En esta sección se explica el programa realizado en App Inventor detallando cada uno de los bloques de programación utilizados para el correcto funcionamiento de la aplicación.

3.2.3.2 Bloque 1 y 2

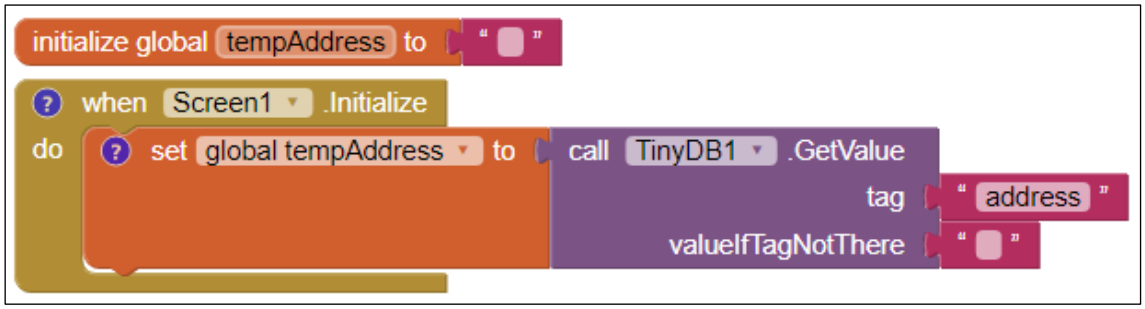


Figura 3.11 - Bloque 1 y 2

Fuente: Elaborado por el autor

La Figura 3.11 muestra los dos primeros bloques que son utilizados para la configuración inicial de nuestra aplicación móvil. Realizan la función de inicializar la interfaz gráfica y realizar la conexión hacia la base de datos TinyDB que utilizará nuestra aplicación.

3.2.3.3 Bloque 3

En la Figura 3.12 se muestra el bloque asociado al sensor GPS cuando se determine que el teléfono cambió de ubicación.

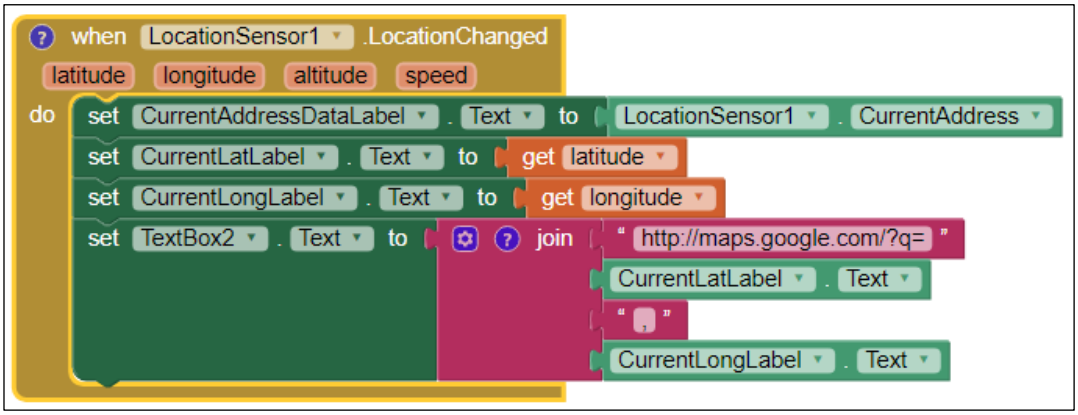


Figura 3.12 - Bloque 3

Fuente: Elaborado por el autor

Cuando la aplicación a través del sensor GPS detecte que ha cambiado la ubicación del *smartphone*, este modificará la propiedad Text de las variables: CurrentAddressDataLabel, CurrentLatLabel, CurrentLongLabel y TextBox2 con los valores: CurrentAddress, latitud, longitud y con el mensaje creado a partir del texto "http://maps.google.com/?q=" concatenando con los valores de la propiedad Text del CurrentLatLabel y CurrentLongLabel modificados anteriormente para obtener una dirección web que nos mostrará la ubicación actual del *smartphone*.

3.2.3.4 Bloque 4

En la Figura 3.13 se muestra el bloque asociado a la acción de presionar el botón "Mi Localización" que se detallará a continuación.

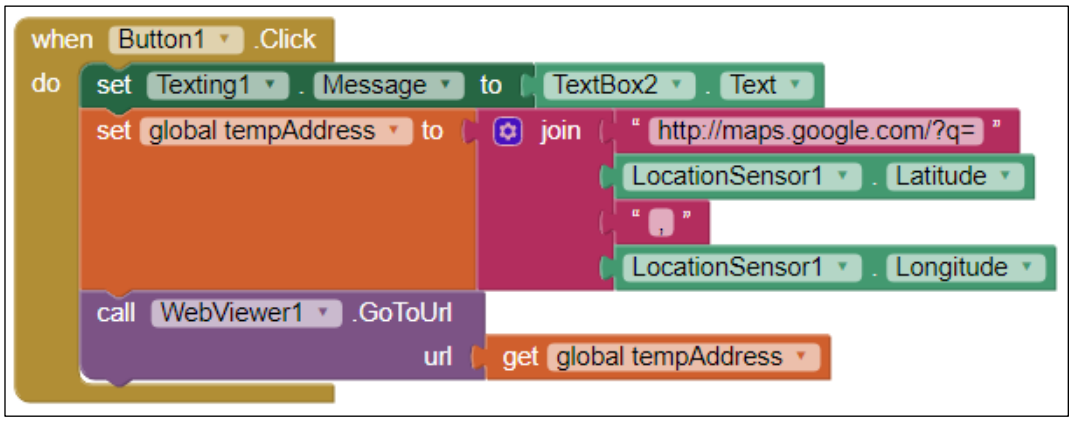


Figura 3.13 - Bloque 4

Fuente: Elaborado por el autor

Cuando el botón "Mi localización" haya sido presionado, la aplicación reemplazará la propiedad Text de Message con el valor del texto que se encuentra en el textBox2, luego creará una dirección web a partir del texto "http://maps.google.com/?=" concatenando con los valores de latitud y longitud del sensor GPS. Posteriormente se desplegará esta dirección en nuestro navegador web, esta dirección mostrará el mapa de Google con un marcador en nuestra posición actual.

3.2.3.5 Bloque 5

En la Figura 3.14 se muestra el bloque asociado a la acción de presionar el botón "Localización Auto".

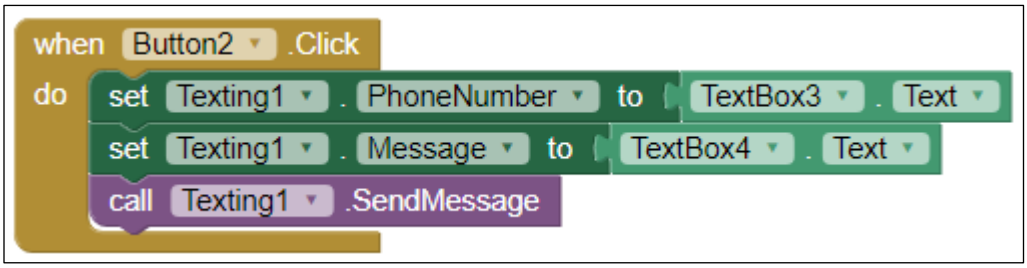


Figura 3.14 - Bloque 5

Fuente: Elaborado por el autor

Cuando el botón "Localización Auto" haya sido presionado, la aplicación creará un mensaje de texto usando el número destinario que esté escrito en el TextBox3 y el contenido del mensaje que este escrito en el TextBox4, luego enviará dicho mensaje a través de la red celular realizando la petición de ubicación del circuito controlador con el objetivo de esperar una respuesta que permitirá localizarlo.

3.2.3.6 Bloque 6

La Figura 3.15 muestra el bloque de programación asociado a la acción de recibir un mensaje de texto.

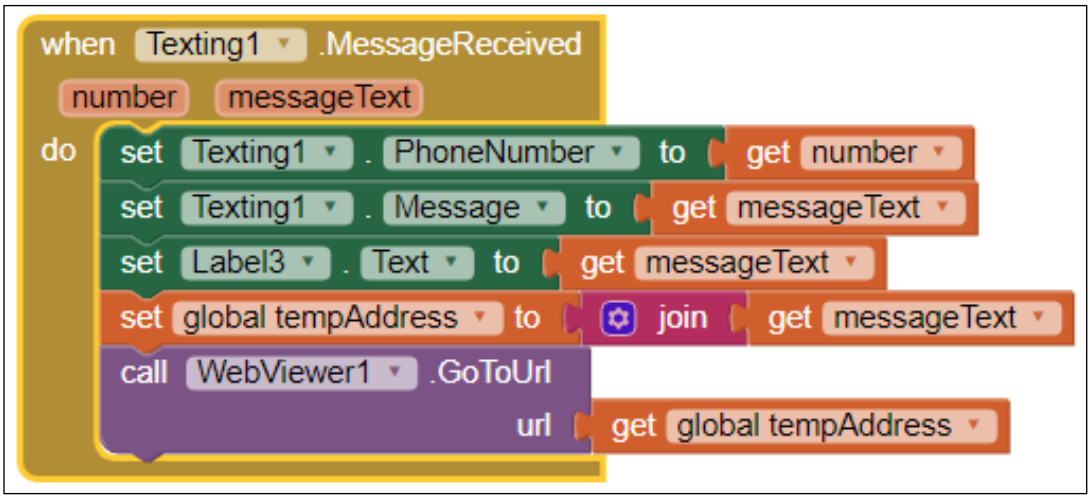


Figura 3.15 – Bloque 6

Fuente: Elaborado por el autor

Cuando el *smartphone* haya recibido un mensaje de texto la aplicación obtendrá dos variables: el número y el mensaje en sí, luego modificará la propiedad Text del Label3 es decir el texto que muestra este componente con el contenido del SMS, crea una dirección web usando el contenido del mensaje y finalmente mostrará esta dirección web a través del navegador web de la aplicación.

3.2.4 Instalación de la aplicación en el *smartphone*

Una vez terminada la implementación de la aplicación se debe verificar que no existan errores observando las alertar que indica el IDE App Inventor, una vez comprobado vamos al menú -> Built el cual creará el archivo instalador de la aplicación, para la descarga de la aplicación se tiene dos opciones como se muestra en la figura las cuales detallaremos a continuación:

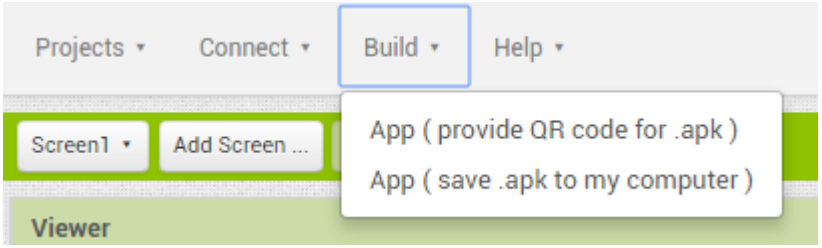


Figura 3.16 – Opciones de descarga del archivo .apk

Fuente: Elaborado por el autor

La Figura 3.16 muestra dos opciones para la descarga del archivo instalador de la aplicación móvil.

3.2.5 Opción 1 de descarga: App (provide QR code for .apk)

Al elegir esta opción del menú Build el entorno mostrará un código QR (Quick Response) como el mostrado en la Figura 3.17, para lo que es necesario tener instalado un lector de códigos QR dentro de nuestro teléfono móvil. Este código redireccionará hacia un enlace el cual iniciara la descarga del *.apk* de la aplicación.

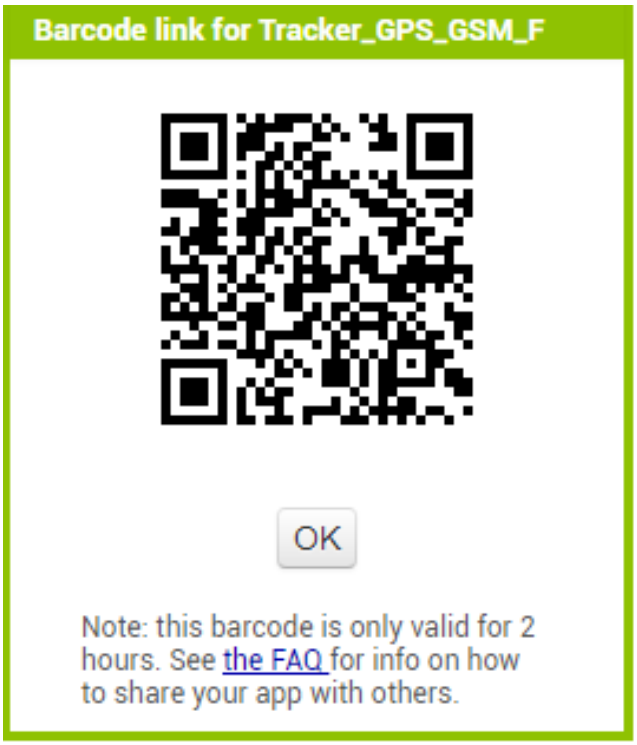


Figura 3. 17 - Código QR generado para descargar la aplicación.

Fuente: Elaborado por el autor

3.2.6 Opción 2 de descarga: App (save .apk to my computer)

Cuando se elige esta opción se iniciará una descarga de un archivo .apk dentro de nuestro navegador. Este archivo es necesario copiarlo en el *smartphone* para proceder con la instalación.

Una vez descargado el archivo .apk en el *smartphone* se debe permitir la instalación de aplicaciones de orígenes desconocidos para lo cual se irá a configuración, Bloqueo y seguridad y activar la opción Orígenes desconocidos.

Posteriormente instalamos la aplicación a través del archivo .apk descargado anteriormente y se procede a usarla.

3.3 Pruebas de funcionamiento

En primer lugar, se verificaron que todos los requerimientos del prototipo se cumplan correctamente, después se instaló el circuito controlador en la parte interior del vehículo, tomando en cuenta que el sensor GPS tenga línea de vista para poder obtener los datos.

Las pruebas de funcionamiento realizadas a la aplicación móvil fueron las que mencionamos a continuación

- La aplicación determina la ubicación del *smartphone* y muestra en el mapa la dirección.
- La aplicación envía la solicitud de ubicación al circuito controlador.
- El circuito controlador determina su ubicación.
- El circuito controlador responde a una solicitud a través de un SMS.
- La aplicación a través del envío de SMS recibirá la ubicación del vehículo.
- La aplicación muestra la ubicación recibida del circuito controlador.

3.3.1 La aplicación determina mi ubicación y muestra mi dirección.

Al iniciar la aplicación se muestra la interfaz gráfica con los valores de Address: unknown y GPS: 0,0 como se muestra en la Figura 3.18 lo que significa que la aplicación aún no reconoce la ubicación del dispositivo. Al pasar unos segundos cuando la aplicación ha recibido los datos del GPS estos campos muestran la dirección actual y las coordenadas de latitud y longitud de la ubicación del *smartphone* como muestra la Figura 3.18

Figura 3.18 - Aplicación recién iniciada.

Fuente: Elaborado por el autor

Figura 3. 19 – Aplicación después de recibir los datos GPS

La siguiente prueba realizada fue si la aplicación muestra mi ubicación. La Figura 3.20 muestra a la aplicación después de haber presionado el botón "Mi localización" en donde se muestra en el navegador web la ubicación actual del *smartphone* a través del mapa de Google.

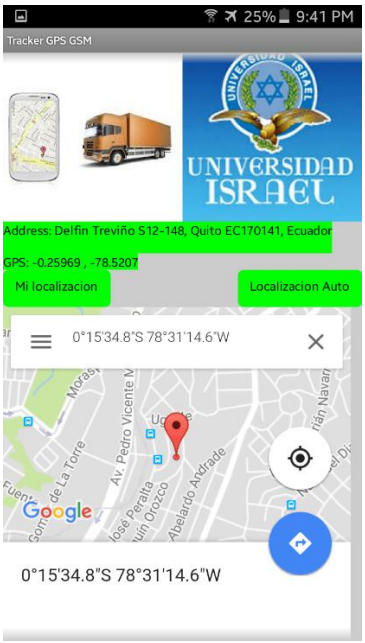


Figura 3.20 - Aplicación luego de presionar el botón "Mi localización"

Fuente: Elaborado por el autor

3.3.2 Pruebas del Sistema

En el vehículo se instaló el prototipo del circuito controlador en la parte delantera conectándose mediante el puerto USB del radio como se muestra en la Figura 3.21.

Figura 3.21 - Instalación del circuito en un vehículo

Fuente: Elaborado por el autor

Una vez instalado el circuito en un vehículo el cual tomó una ruta de un bus de transporte para poder realizar las pruebas de funcionamiento.

A continuación, se describirán las pruebas de funcionamiento realizadas. Se determinó la ubicación del vehículo en tres ocasiones determinando en cada ocasión la ruta que debe tomar y tiempo de llegada estimado.

En la Figura 3.23 se muestra a la aplicación después de haber presionado el botón "Localizar Auto", El resultado como era de esperarse mostro la ubicación del vehículo en un mapa de Google. Previamente a esto, se recibió por mensaje de texto la ubicación como se observa en la Figura 3.22.

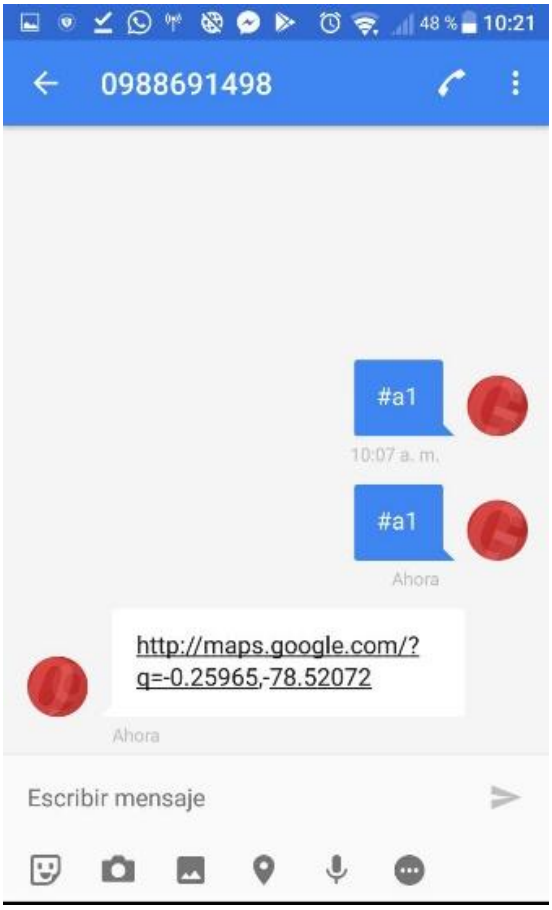


Figura 3.22 - Envío y recepción de mensajes entre el circuito y la aplicación

Fuente: Elaborado por el autor

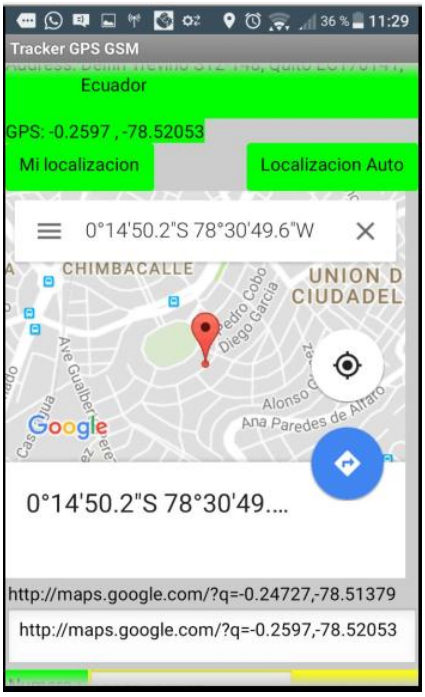


Figura 3. 23 - Primera localización del vehículo.

Fuente: Elaborado por el autor

En la Figura 3.24 indica la ruta, tiempo de llegada, y distancia en la que se encuentra el vehículo.

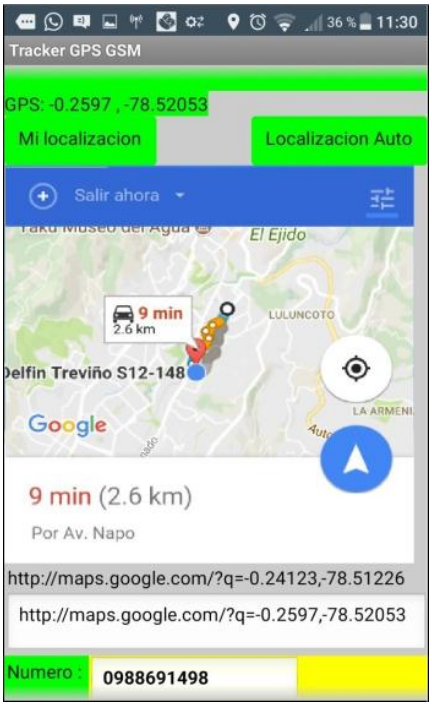


Figura 3.24 - Determinación de la distancia, tiempo y ruta 1

Fuente: Elaborado por el autor

Después de cierto tiempo se volvió a solicitar la ubicación del vehículo y cómo se puede observar en la Figura 3.25 el vehículo ya se encuentra en otra ubicación y se volvió a determinar la ruta, tiempo de llega y distancia a la que se encuentra.

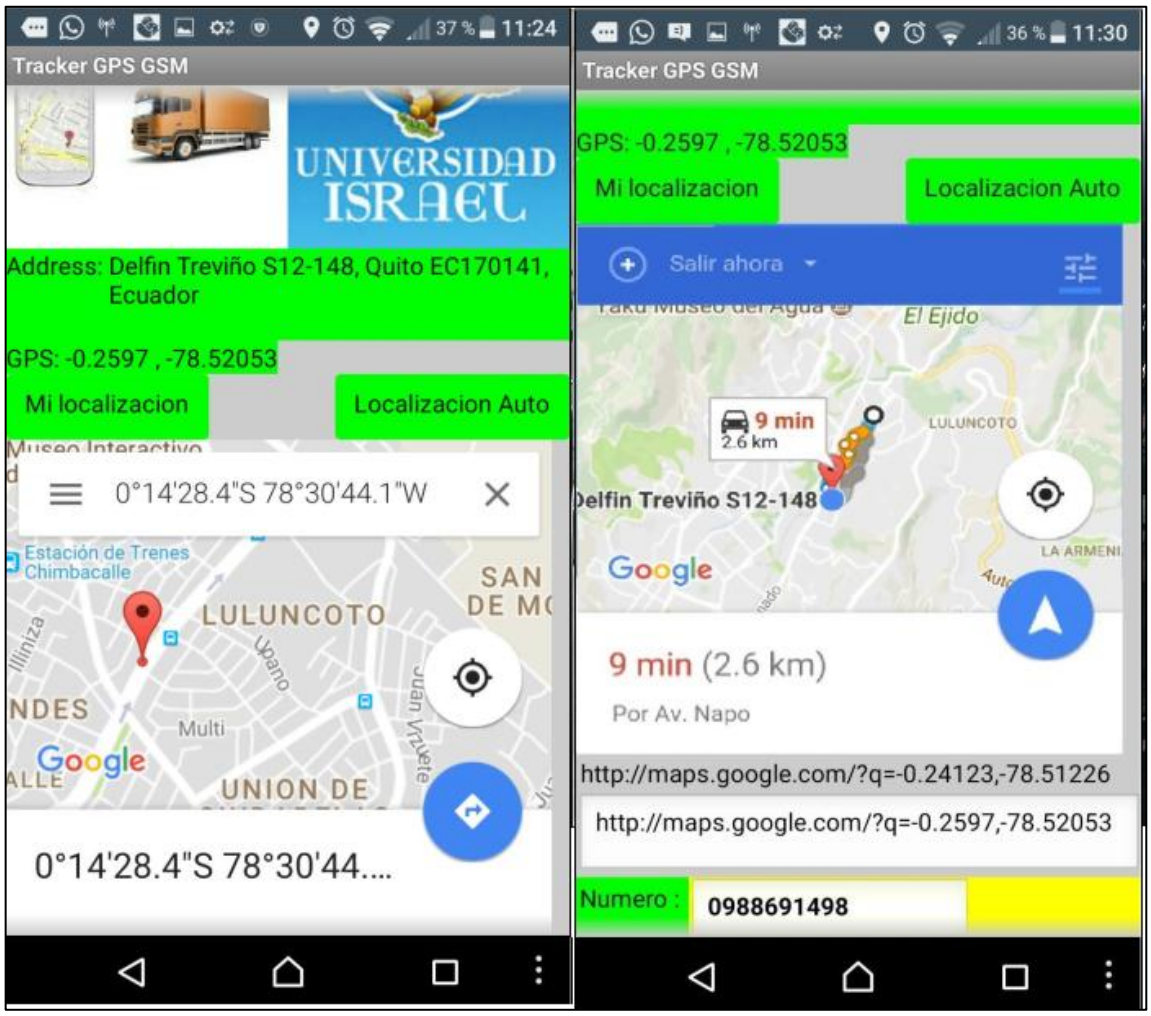


Figura 3.25 - Determinación de posición, distancia, tiempo y ruta

Fuente: Elaborado por el autor

Se volvió a solicitar una tercera vez la ubicación del vehículo y se obtuvo los resultados mostrados en la figura.

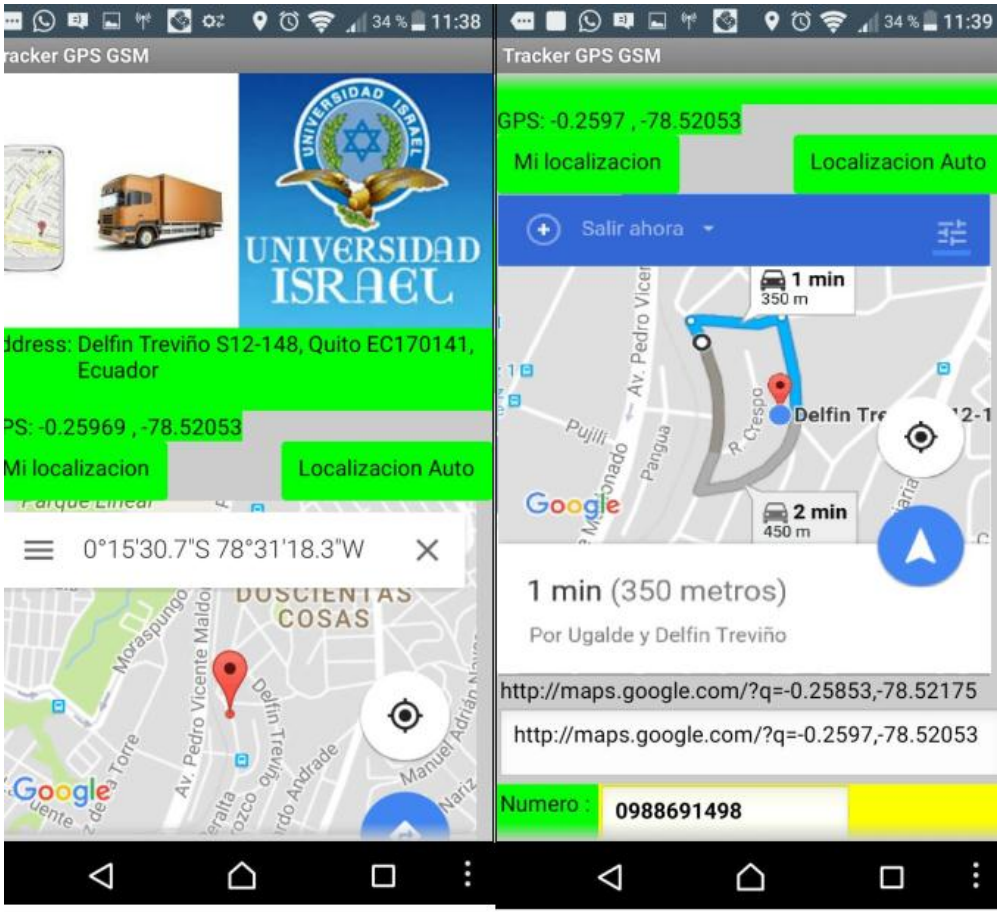


Figura 3.26 - Tercera prueba de localización y ubicación

Fuente: Elaborado por el autor

3.4 Análisis de resultados

Para el análisis de resultados se verificará los parámetros detallados a continuación.

3.4.1 Funcionamiento de la autenticación

Antes de realizar las pruebas del funcionamiento del prototipo se debe encender el GPS en el dispositivo móvil, además se debe solicitar a la aplicación que determine la ubicación del *smartphone*, como se observa en la Figura 3.27, para de esta manera tener un punto de referencia para el trazado de las rutas.

Figura 3.27 – Determinación de la ubicación del dispositivo móvil

Fuente: Elaborado por el autor

Para realizar la autenticación se utiliza una clave en texto plano. La cuál es configurada en la aplicación de Android y en el programa Arduino. Para las pruebas la clave utilizada fue #a1. En la Figura 3.28 se observa que al solicitar la ubicación el teléfono envía la clave no encriptada.

Figura 3.28 – Envió de un SMS con la contraseña

Fuente: Elaborado por el autor

3.4.2 Recepción de las coordenadas de ubicación

Después que la contraseña se verifique en el circuito de control, este procede a enviar la información que le proporciona el módulo GPS, a través de un mensaje de texto que envía el shield GSM900, con un enlace que redirige a los mapas de Google, como se puede observar en la Figura 3.29. En este enlace se encuentra la información de longitud y latitud para localizar el autobús.

El formato del mensaje recibido es:

http://maps.google.com/?q= coordenadas de latitud, coordenadas de longitud

El formato que se utiliza para el procesamiento de las coordenadas es grados y minutos decimales.

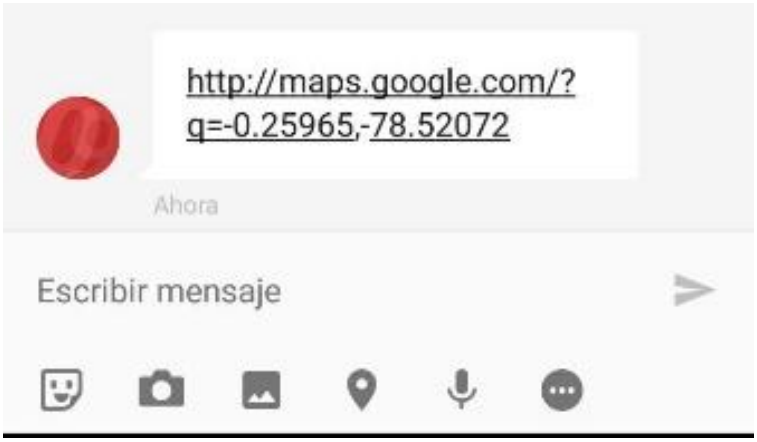


Figura 3.29 - Recepción del SMS con las coordenadas

Fuente: Elaborado por el autor

La aplicación de Android permite la redirección automática del enlace y va almacenando las ubicaciones señaladas. Esto se puede observar en la Figura 3.30. Una vez obtenida la última localización esta se muestra en el mapa con un puntero de color rojo.

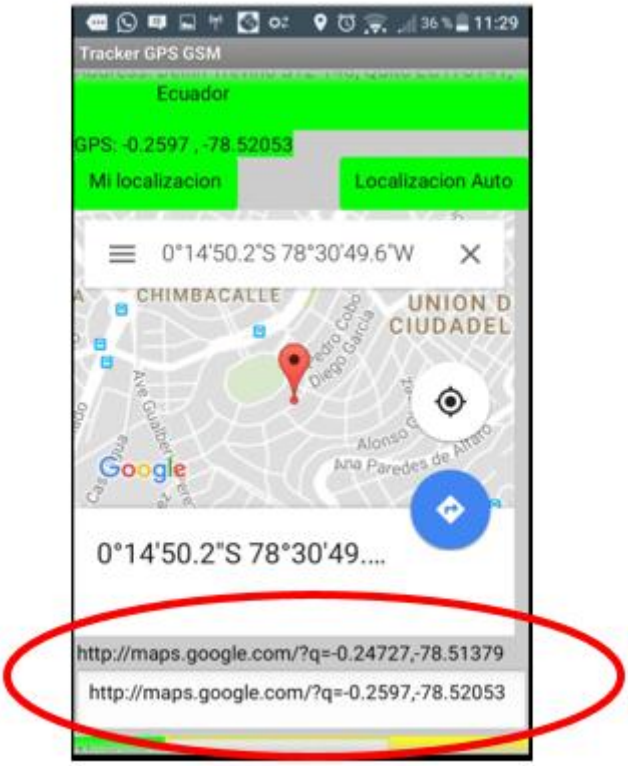


Figura 3.30 – Redirección de la información del SMS a la aplicación Android.

Fuente: Elaborado por el autor

3.4.3 Conversión de grados sexagesimales a decimales.

La notación de grados sexagesimales para identificación de las coordenadas es usada en los mapas tradicionales, impresos, plegables, libros, etc. Consiste en tres grupos de valores: grados, minutos y segundos. Un ejemplo sería 25°08”S 82°12’33”W.

En cambio, el sistema decimal para la determinación de la ubicación es muy empleado en los programas y aplicaciones de mapas, en especial en el sistema GPS. En esta notación se separa la parte entera de la parte fraccionaria por medio de un punto; convirtiéndose los minutos y segundos en números decimales. Un ejemplo sería 0.158777, -23.138884

Para realizar manualmente la conversión de la notación de grados sexagesimales a decimales se tendría, por ejemplo:

(0° 14’ 50.2”) = (0 + (14/60)) + (50.2/3600)) = 0.247277

El proceso manual es muy largo y tedioso cuando se realizan varias mediciones, existen varios convertidores online que permiten optimizar este cálculo. En la aplicación esto se realiza de manera automática, gracias a los mapas de Google, esto se puede observar en la Figura 3.31.

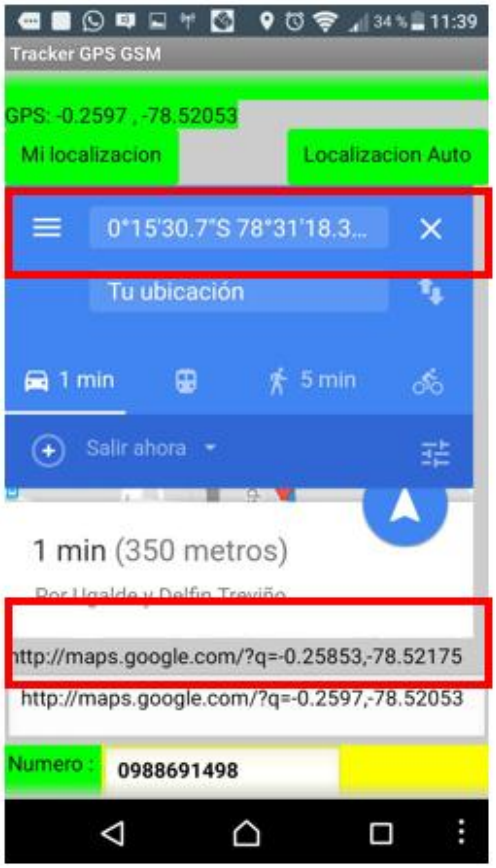


Figura 3.31 – Conversión automática de grados sexagesimales a decimales

Fuente: Elaborado por el autor

3.4.4 Resumen análisis de resultados.

En la Tabla 3.1 se exponen de manera resumida las razones por la que se desarrolló el prototipo en lugar de comprar uno existente en el mercado y se puede verificar las ventajas que trajo el desarrollo del prototipo. Es importante recalcar que la mayor ventaja además del ahorro en costos es la flexibilidad que tiene el prototipo para poder añadir características personalizadas porque se posee el código fuente.

Tabla 3.1 – Prototipo desarrollado vs uno existente en el mercado.

Característica	Prototipo en el mercado	Prototipo desarrollado
Es posible personalizar rutas, número de vehículos en un desarrollo futuro	✗	✓
Tiene un costo considerablemente bajo teniendo en cuenta la calidad de sus materiales.	✗	✓
Se cubre las necesidades generales y especificas adecuados para el uso en un bus de transporte urbano.	✗	✓
Permite determinar la ubicación en tiempo real del vehículo.	✓	✓
Permite determinar la ruta y el tiempo de llegada del vehículo.	✗	✓
Es posible reemplazar o aumentar módulos para agregar características al sistema.	✗	✓

Fuente: Elaborado por el autor

Todos los elementos, el sistema operativo escogido, los módulos utilizados permiten el correcto funcionamiento del sistema implementado, teniendo una correcta interacción entre la aplicación Android y el sistema de control ubicado en el bus de transporte urbano y logrando obtener la ubicación, la ruta y la distancia de este, cumpliendo los objetivos del proyecto.

A continuación, en la Tabla 3.2 se observa de manera resumida las razones por las cuales durante el estudio teórico se decidió utilizar los siguientes componentes.

Tabla 3. 2 - Razones para el uso de los elementos del prototipo.

Elementos del Prototipo	Descripción de los elementos
Sistema Operativo Android	Se eligió el sistema operativo Android debido a que está basado en el paradigma de software libre y tiene un porcentaje de utilización mayor al 80%.
App Inventor	App Inventor permite realizar aplicaciones Android evitando al desarrollador la escritura excesiva de código, simplificando así el desarrollo.
Arduino UNO	Este modelo de Arduino es la placa cual ha servido de base para desarrollar los diferentes modelos por lo cual consta con variedad de elementos tanto hardware como software compatibles y fáciles de conseguir para esta versión de Arduino.
Módulo GPS6MV2	Este receptor GPS tiene alta precisión además de tener un tamaño reducido tiene gran facilidad de uso y compatibilidad con Arduino Uno.
Módulo Shield GSM SIM900	Este módulo además de ser de una buena calidad tiene gran compatibilidad con Arduino Uno el cual es la base del prototipo, facilitando la conexión y la programación del proyecto.

Fuente: Elaborado por el autor

CONCLUSIONES

Se estimó las coordenadas de la ubicación del bus urbano, a través de la definición de las variables de tiempo y distancia. Lo que permitió adquirir la ubicación de un bus urbano.

El sistema GPS diseñado proporciona mucha información acerca de la localización del objeto, para el desarrollo de la tesis; se realizó un filtro que solo envié las coordenadas de latitud y longitud.

La aplicación Android nos permite visualizar la ruta, la distancia y el tiempo estimado de llegada del vehículo. A través del envío de SMS. Se eligió Android porque es el sistema operativo que acapara mayor mercado en la venta de aplicaciones móviles.

Se creó un sistema completo para el monitoreo de un bus urbano, a través de la identificación de este y de su ruta. Esto se logró a través de la interacción de la aplicación Android y el prototipo que se ubica dentro del vehículo.

Se realizó un análisis de las pruebas obtenidas al implementar el prototipo completo para verificar su correcto funcionamiento. Las pruebas fueron satisfactorias y permitieron la correcta implementación del sistema

RECOMENDACIONES

Se podría implementar la aplicación de Android en otro entorno de programación diferente a APP Inventor, así, por ejemplo, Basic4Android, InDesig CS6, HTLM5, Ruboto, Rhomobile Rodes, entre otros.

Para futuras versiones se podría utilizar una placa de Arduino más compacta y de esta manera economizar espacio, para desarrollar un sistema de alimentación que sea autosustentable, a través de la reutilización de energía, un ejemplo sería la utilización de pequeños paneles solares.

El código se puede mejorar para que no solo realice el monitoreo de una unidad a la vez, si no que a través de una red de sensores poder monitorear una cooperativa completa.

Otra mejora al prototipo implicaría recolectar la información de las rutas y tiempos durante el día y almacenarla en una base de datos, de esta manera se lograría la automatización del servicio urbano, y con esa información recaudada mejorar la calidad del servicio de transporte público.

El prototipo podría integrarse al sistema de conteo de pasajeros que muchos buses poseen, de esta manera la información recolectada será más eficiente. Debido a que permitirá determinar si el número de unidades disponibles a determinadas horas es suficiente para cubrir la demanda en la ciudad.

BIBLIOGRAFÍA

Aprendiendo Arduino. (2017). Obtenido de IDE Arduino y Configuración: https://aprendiendoarduino.wordpress.com/category/ide/

Arduino Nano- User Manual. (2017). Obtenido de https://www.arduino.cc/en/uploads/Main/ArduinoNanoManual23.pdf

Arduino.stack. (2017). Obtenido de How to communicate the Arduino board with SIM900: https://arduino.stackexchange.com/questions/9483/how-to-communicate-the-arduino-board-with-sim900

Baz, A., Ferreira, I., Álvarez, M., & García, R. (s.f.). *Dispositivos móviles.* Obtenido de http://isa.uniovi.es/docencia/SIGC/pdf/telefonia_movil.pdf

Blanco, P., Camarero, J., Fumero, A., Werterski, A., & Rodríguez, P. (2016). *Metodología de desarrollo ágil para sistemas móviles. Introducción al desarrollo con Android y el iPhone.* Obtenido de https://www.researchgate.net/profile/Antonio_Fumero/publication/267795011_Metodologia_de_desarrollo_agil_para_sistemas_moviles_Introduccion_al_desarrollo_con_Android_y_el_iPhone/links/577009d108ae842225aa444b/Metodologia-de-desarrollo-agil-para-sistemas-m

Designers and Blocks Editor. (2015). Obtenido de http://appinventor.mit.edu/explore/designer-blocks.html

DSpace, C. (2017). *CZECH TECHNICAL UNIVERSITY DIGITAL LIBRARY.* Obtenido de https://dspace.cvut.cz/bitstream/handle/10467/65602/F2-BP-2016-Lexmann-Robert-priloha-4-ArduinoMega2560.pdf?sequence=-1&isAllowed=y

Educa Madrid. (2017). Obtenido de Espadines: https://mediateca.educa.madrid.org/imagen/2qdrh9knzyu6qbky

Enríquez, R. (2009). *Guía de Usuario de Arduino.* Obtenido de http://www.uco.es/aulasoftwarelibre/wp-content/uploads/2010/05/Arduino_user_manual_es.pdf

ETSI. (1996). *Digital cellular telecommunications system (Phase 2+); Technical realization of the Short Message Service (SMS) Point-to-Point (PP).* Obtenido de

http://www.etsi.org/deliver/etsi_gts/03/0340/05.03.00_60/gsmts_0340v050300p.pdf

Force, U. A. (2 de agoto de 2017). *GPS.gov*. Obtenido de Control Segment: http://www.gps.gov/systems/gps/control/

Force, U. A. (2017). *GPS.gov*. Obtenido de GPS Accuracy: http://www.gps.gov/systems/gps/performance/accuracy/

Gete-Alonso Roldán, Ó. (2008). *Estudio de disponibilidad de señales de localización GPS/GSM*. Obtenido de La arquitectura de GSM: http://www.galeon.com/seguridadengprs/tema2.htm#2

Gisiberica. (2017). Obtenido de GPS TOPOGRÁFICO: http://www.gisiberica.com/GPS%20Topografia%20geodesia/GPS%20TOPOGRAFICOS.htm

HETPRO. (2017). Obtenido de SIM900 GSM GPRS SHIELD con arduino UNO: https://hetpro-store.com/TUTORIALES/sim900-gsm-shieldarduino/

Hillebrand, F., Trosby, K., & Harris, L. (2010). *Short Message Service-The Creation of Personal Global Text Messaging.* London.

Hofmann, B., Collins, J., & Lichtenegger, H. (2012). *Global Positioning System: Theory and Practice.* New York.

Inteligencia artificial. (2017). Obtenido de GPSMV2: http://www.inteligenciaartificialyrobotica.com/esp/item/496/9/gy-gps6mv2-modulo-gps-rx-tx

JAMECO Electrónica. (2017). Obtenido de https://www.jameco.com/Jameco/Products/ProdDS/2163778.pdf

leantec-robotics&electronics. (2017). Obtenido de Tutorial Arduino: Módulo GPS GPS6MV2: http://www.leantec.es/blog/54_Tutorial-Arduino--Modulo-GPS-GPS6MV2.html

LilyPad Arduino. (2017). Obtenido de http://www.mouser.com/catalog/specsheets/LilyPad.pdf

LinuxGizmos.com. (s.f.). Obtenido de Arduino adds wireless-ready "Arduino Uno WiFi": http://linuxgizmos.com/arduino-srl-adds-wireless-ready-arduino-uno-wifi/

Lizana, C. (2017). *INTEGRACIÓN DE TÉCNICAS GPS EN AEROTRIANGULACIÓN.* Obtenido de ftp://ftp.unsj.edu.ar/agrimensura/Fotogrametria/.../Aerotriangulacion_TEMA_7.doc

Romero, R. (2017). *Arquitectura de Android*. Obtenido de https://persefoneblog.wordpress.com/formacion/android_introduccion/tema-1/1-1-arquitectura-de-android/

Ruiz, J. (2007). *Manuel de Programación Arduino.* Obtenido de https://arduinobot.pbworks.com/f/Manual+Programacion+Arduino.pdf

UOC. (2017). Obtenido de Tecnología y desarrollo en dispositivos móviles: http://cv.uoc.edu/web/~mgalicia/practica_final/modulo5_2_2%20.html

Wolber, D. (s.f.). *App Inventor and Real-World Motivation.* Obtenido de http://cs.millersville.edu/~rwebster/cs406MDD/stuff/ACMSIGCSE2011Papers/p601.pdf

YouBioit.com. (2017). Obtenido de Qué es la trilateración satelital: https://www.youbioit.com/es/article/15626/que-es-la-trilateracion-satelital&size=_original

ANEXOS

ANEXO 1: MANUAL TÉCNICO

INTRODUCCIÓN

La finalidad del presente manual es proporcionar la lógica con lo que se desarrolló el sistema, debido a que la lógica de programación y diseño puede variar entre desarrolladores por lo cual se considera necesario documentar.

Se aclara que el presente documento no pretende ser un curso de aprendizaje de las herramientas utilizadas al momento de desarrollar el sistema.

OBJETIVO

Proporcionar una guía para que el lector del presente documento comprenda la lógica del sistema implementado.

CONTENIDO

Descripción del sistema

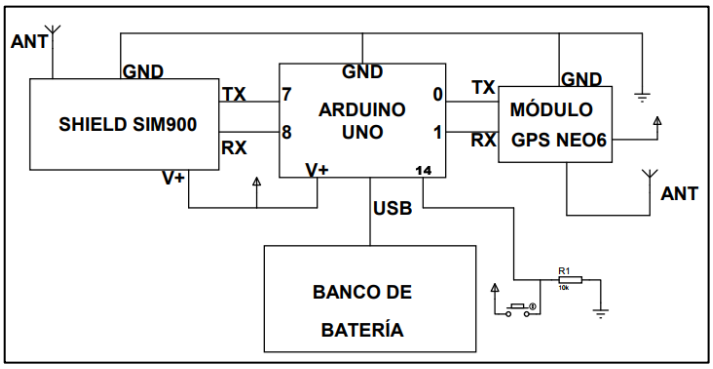
Es un sistema prototipo que sirve para la localización GPS de un bus urbano mediante una aplicación móvil donde se podrá verificar el recorrido de la ruta asignada.

El sistema consta de dos componentes los cuales son un circuito controlador y una aplicación móvil.

Circuito controlador

El cual consta de una placa Arduino Uno, un módulo SIM900 y un módulo GPS6MV2. El circuito será instalado dentro de un vehículo el cual realizará lecturas de su posición mientras espera una solicitud de ubicación mediante mensajes de texto. Una vez recibida la solicitud este componente responde con un mensaje de texto indicando los datos de sus coordenadas actuales.

Conexión del circuito



Software utilizado: Arduino IDE.

El código quemado en el Arduino sirve para leer datos del GPS, esperar una solicitud de ubicación vía SMS y responder a dicha solicitud con las coordenadas obtenidas del GPS.

El código utilizado consta de las siguientes funciones y métodos:

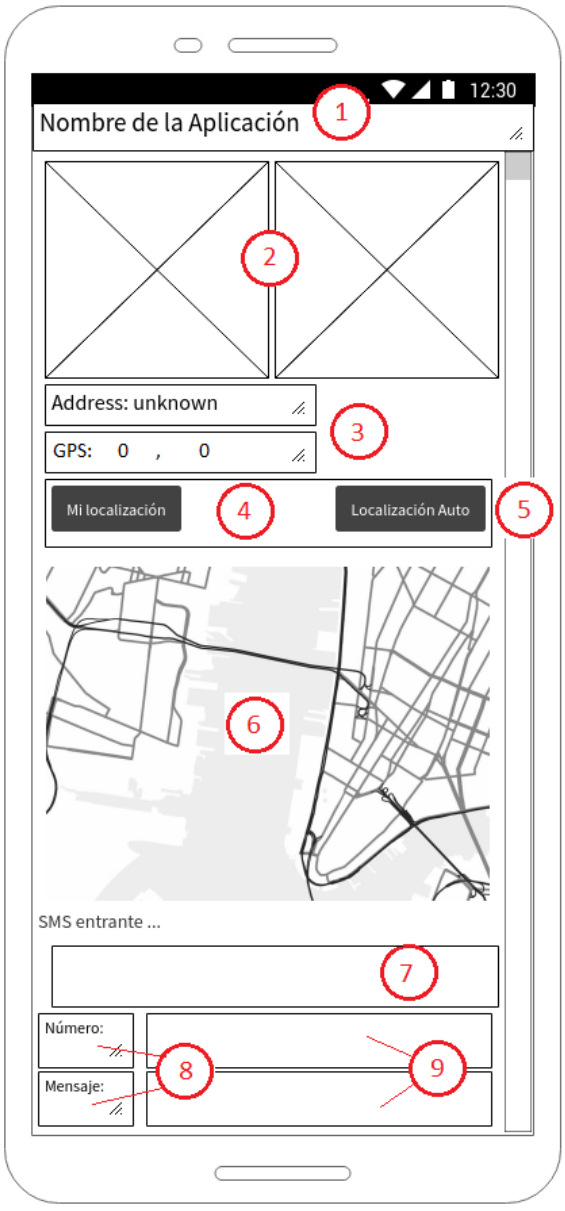
- **Función Setup:** Configuración básica del Arduino Uno para la comunicación con los dos módulos, el encendido y configuración del módulo SIM900 para el envío de mensajes de texto.
- **Método para el encendido del SIM900:** Utilizado para realizar la función del botón *power* del módulo SIM900 mediante instrucciones de *software*.
- **Función Loop:** Permite leer los datos del GPS y estar esperando una solicitud esperando recibir un mensaje con el código con el mensaje #a1 para responder la solicitud con las coordenadas de la posición recibida del GPS.
- **Método *leergps:*** Permite recibir los valores de lectura del GPS mediante el puerto serial para ser almacenados en las distintas variables declaradas al inicio del código.
- **Método *enviarsms*:** Envía instrucciones al módulo SIM900 indicando el contenido del mensaje que se quiere enviar y el destinatario. Para este caso se enviará un mensaje que contiene un vínculo hacia Google Maps indicando las coordenadas de latitud y longitud obtenidas por el GPS.

Aplicación Móvil

La cual podrá determinar la ubicación del smartphone, hacer una solicitud de ubicación hacia el circuito controlador, leer la respuesta del circuito para posteriormente determinar la ubicación del vehículo y mostrar la ruta y el tiempo de llegada desde su ubicación hacia la ubicación del smartphone.

Software utilizado: App Inventor

La aplicación móvil consta de una interfaz visual descrita a continuación:



Identificador	Descripción
1	Muestra el nombre de la aplicación
2	Muestra imágenes de presentación de la aplicación
3	Etiqueta que muestra la dirección y coordenadas GPS del teléfono
4	Botón para mostrar mi ubicación en el mapa
5	Botón para mostrar la ubicación del vehículo en el mapa
6	Visor Web utilizado para la visualización del mapa.
7	Texto donde se muestra el mensaje de respuesta del circuito controlador
8	Etiqueta para identificar lo que debo ingresar en los TextBox
9	Textos que pueden ser modificados por el usuario donde va el número de teléfono asociado al circuito controlador y el mensaje a enviar

ANEXO 2: MANUAL DE USUARIO

INTRODUCCIÓN

El siguiente manual tiene como objetivo aprender a usar todas las funcionalidades del prototipo de geolocalización.

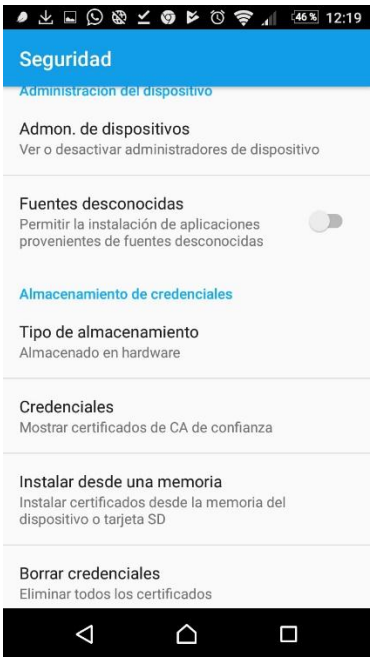
Este manual está enfocado a los usuarios con poco o ningún conocimiento técnico, que harán uso del prototipo y de su aplicación de Android.

USO DE LA APLICACIÓN EN EL SMARTPHONE

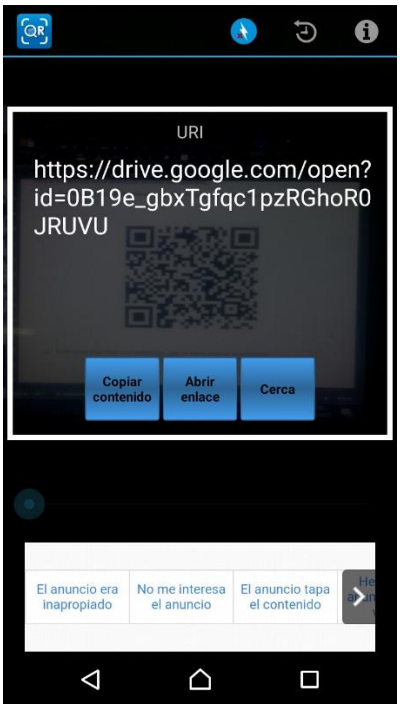
Instalación de la aplicación

Descarga directa de la aplicación en el celular a través del código QR que se indica a continuación.

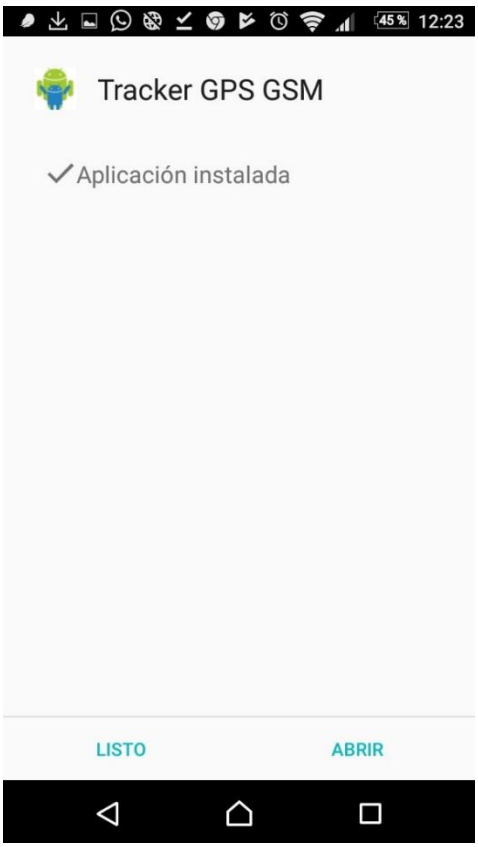
Antes de escanear el código QR es necesario que en Ajustes o Configuración se dé permiso para aplicaciones de origen desconocido, como se indica en la figura.



A continuación, debemos aceptar ir al enlace que redirige el código QR



Luego la aplicación se comenzará a instalar, una vez finalizada la instalación, se visualizará en la pantalla que se ha realizado con éxito el proceso.



Luego de que la aplicación se haya instalado correctamente se selecciona abrir y se visualizará la siguiente pantalla.

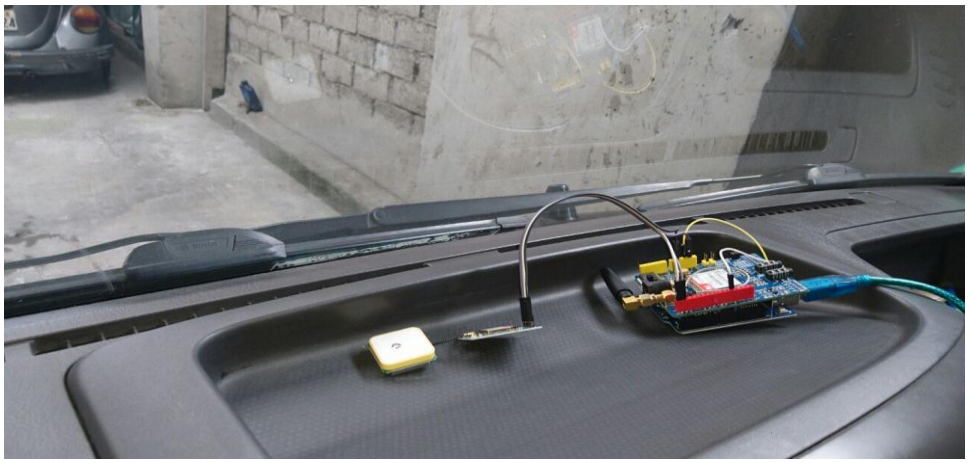
Al inicializar la aplicación buscará la localización del móvil, si es que el GPS se encuentra encendido

Descripción de la interfaz gráfica

Identificador	Descripción
1	Muestra el nombre de la aplicación
2	Muestra imágenes de presentación de la aplicación
3	Etiqueta que muestra la dirección y coordenadas GPS del teléfono
4	Botón para mostrar mi ubicación en el mapa
5	Botón para mostrar la ubicación del vehículo en el mapa
6	Visor Web utilizado para la visualización del mapa.
7	Texto donde se muestra el mensaje de respuesta del circuito controlador
8	Etiqueta para identificar los campos a ser llenados.
9	Textos que pueden ser modificados por el usuario donde va el número de teléfono asociado al circuito controlador y el mensaje a enviar

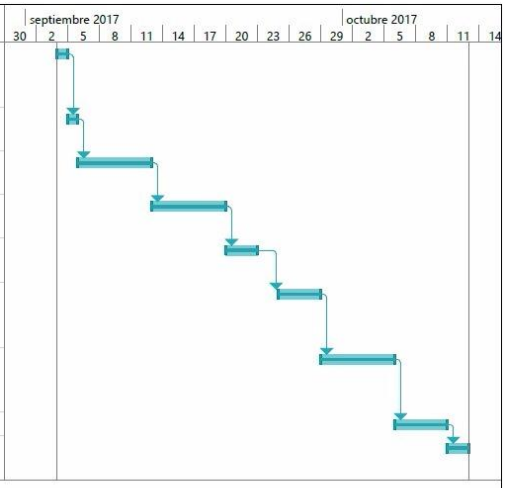
USO DEL CIRCUITO CONTROLADOR

Al alimentar el circuito controlador con 5V y una corriente mínima de 2A, este se enciende automáticamente. Y después de un momento el GPS comienza a enviar y a recibir señales de los satélites. Es importante considerar que la antena de cerámica de GPS debe tener línea de vista con los satélites para su mejor funcionamiento.



ANEXO 3: CRONOGRAMA

Id	Modo de tarea	Nombre de tarea	Duración	Comienzo	Fin
1		Consulta de los sensores necesarios para Arduino UNO	1 día	lun 4/9/17	lun 4/9/17
2		Compra de los materiales necesarios	1 día	mar 5/9/17	mar 5/9/17
3		Desarrollo de la aplicación en APP Inventor	5 días	mié 6/9/17	mar 12/9/17
4		Desarrollo del programa en Arduino	5 días	mié 13/9/17	mar 19/9/17
5		Implementación del circuito controlador	3 días	mié 20/9/17	vie 22/9/17
6		Pruebas de funcionamiento integrando todas las características	4 días	lun 25/9/17	jue 28/9/17
7		Correcciones necesarias en la aplicación y en el circuito controlador	5 días	vie 29/9/17	jue 5/10/17
8		Pruebas finales.	3 días	vie 6/10/17	mar 10/10/17
9		Análisis de resultados	2 días	mié 11/10/17	jue 12/10/17



Proyecto: Cronograma.mpp
Fecha: mar 28/8/18

Tarea	Resumen inactivo	Tareas externas
División	Tarea manual	Hito externo
Hito	solo duración	Fecha límite
Resumen	Informe de resumen manual	Progreso
Resumen del proyecto	Resumen manual	Progreso manual
Tarea inactiva	solo el comienzo	
Hito inactivo	solo fin	

DECLARACIÓN Y AUTORIZACIÓN

Yo, Rommel Ufredo Hurtado Suárez, CI 1206312140 autor/a del trabajo de graduación :**PROTOTIPO DE SISTEMA DE LOCALIZACIÓN DE UN BUS DE TRANSPORTE URBANO MEDIANTE GPS Y UNA APLICACIÓN ANDROID**, previo a la obtención del título de **Ingeniería en Electrónica Digital y Telecomunicaciones** en la UNIVERSIDAD TECNOLÓGICA ISRAEL.

1.- Declaro tener pleno conocimiento de la obligación que tienen las instituciones de educación superior, de conformidad con el Articulo 144 de la Ley Orgánica de Educación Superior, de difundir el respectivo trabajo de graduación para que sea integrado al Sistema Nacional de Información de la Educación Superior del Ecuador para su difusión pública respetando los derechos de autor.

2.- Autorizo a la SENESCYT a tener una copia del referido trabajo de graduación, con el propósito de generar un repositorio que democratice la información, respetando las políticas de propiedad intelectual vigentes.

Quito, Septiembre del 2018

Atentamente.

_____

Rommel Ufredo Hurtado Suárez.

C.I. 1206312140