UNIVERSIDAD NACIONAL DEL CALLAO

FACULTAD DE INGENIERIA ELECTRICA Y ELECTRONICA

ESCUELA PROFESIONAL DE INGENIERIA ELECTRONICA

TESIS PARA OPTAR EL TITULO PROFESIONAL DE INGENIERIO ELECTRONICO

"DISEÑO DE UN SISTEMA DE RUTA CON GPS/4G LTE PARA EL CONTROL DE LAS UNIDADES DE LA EMPRESA NETTELCOM SAC"

BACHILLER CABANA CACERES, GUILLERMO ALEJANDRO
ASESOR: MSC ING JACOBO ASTOCONDOR VILLAR

Callao, 2016
Perú

UNIVERSIDAD NACIONAL DEL CALLAO

FACULTAD DE INGENIERIA ELECTRICA Y ELECTRONICA

ESCUELA PROFESIONAL DE INGENIERIA ELECTRONICA

TESIS PARA OPTAR EL TITULO PROFESIONAL DE INGENIERIO ELECTRONICO

"DISEÑO DE UN SISTEMA DE RUTA CON GPS/4G LTE PARA EL CONTROL DE LAS UNIDADES DE LA EMPRESA NETTELCOM SAC"

BACHILLER CABANA CACERES, GUILLERMO ALEJANDRO
ASESOR: MSC ING JACOBO ASTOCONDOR VILLAR

CALIFICACION: 18 (DIECIOCHO)

Msc. Ing. Armando Pedro Cruz Ramírez
Presidente del Jurado

Ing. Jorge Elías Moscoso Sánchez
Secretario

Ing. Luis Ernesto Cruzado Montañes
Suplente

Callao-Peru
2016

"DISEÑO DE UN SISTEMA DE RUTA CON GPS/4G LTE PARA EL CONTROL DE LAS UNIDADES DE LA EMPRESA NETTELCOM SAC"

DEDICATORIA

A mi familia que siempre me da incentiva para dar lo mejor, y nunca claudicar.

AGRADECIMIENTO

Ante todo quiero darle gracia a Dios que siempre ha sido mi guía, caminado conmigo durante toda mi vida. Gracias por ayudarme a conquistar mis sueños y darme fuerzas para conseguir mis metas.

A mis profesores, mis compañeros y a todas las personas que me facilitaron
El poder concluir mi tesis para mi título de ingeniería.

También quiero extender mi agradecimiento a mi asesor, que ha tenido la Entusiasmo y sensatez para guiar esta investigación.

Quedo muy agradecido con mi alma mater, en la cual además me ha formado para hacer frente en mi camino como profesional. Pondré el nombre de la FIEE-UNAC siempre en alto.

RESUMEN

En la actualidad el mercado no posee desarrollo en sistemas GPS para controlar las unidades de transporte, reparto entre otros teniendo una variedad de rutas lo cual lo define el usuario al momento de desplazarse, se busca una solución mediante el sistema GPS, el cual es un sistema confiable donde el desarrollo de Hardware en conjunto de un Software puedan brindar la solución teniendo como premisa ser Libre, donde la empresa pueda modificar según sus necesidades, obteniendo como resultados rutas confiables, control sobre los usuarios al momento de desplazarse sabiendo en todo momento su ubicación ,se obtuvo resultados favorables donde se pudo verificar mediante una ruta el tiempo, velocidad, gasto de combustible, distancia recorrida.

ABSTRACT

Today the market has no development in GPS systems to control transport units , distribution and others having a variety of routes which defines the user when moving, a solution is sought through the GPS system, which is a reliable system where hardware development together a Software can provide the solution with the premise being Open Source, where the company can modify according to their needs , obtaining as results reliable routes, control over users when scrolling knowing at all times location , favorable results which could be verified by a route time , speed, fuel consumption, distance traveled was obtained.

INDICE DE CONTENIDO

INDICE DE FIGURAS

INDICE DE TABLAS

INTRODUCCIÓN

En nuestro mundo globalizado, el control automático forma parte de cada empresa u organización modernizada. Para esto en la actualidad el sistema GPS (sistema de Posicionamiento Global) es un sistema mundial que nos permite la ubicación, bajo parámetros de navegación y tiempo que ha dado como resultado la localización y posicionamiento global.

Su principal aplicación en sus inicios fue la colocación de puntos geodésicos de alta precisión con errores de ± 2cm. Tomados en forma Diferencial (DGPS), para trabajos de Ingeniería, tales como el control topográfico para proyectos de ferrocarriles, puertos, aeropuertos, carreteras, centrales hidroeléctricas, catastros, entre otros.

Actualmente existen diferentes compañías que fabrican los instrumentos periféricos para poder hacer funcionar el control de los móviles, tales como la ubicación, velocidad, dirección, temperaturas del motor o la carga que necesita refrigeración, control de escotillas y puertas, horas de manejo del piloto y hasta presión de aire de los neumáticos entre otros con lo cual se puede obtener información actualizada del estado del vehículo para su monitoreo.

I. PLANTEAMIENTO DEL PROBLEMA DE INVESTIGACIÓN

1.1. Identificación del Problema.

En nuestro País existe un encarecimiento de cultura en las personas; en el sentido que se tiene el pensamiento erróneo en el cual "el más vivo gana". Lamentablemente, a nuestro parecer, este pensamiento ha sido inculcado desde los valores que le brindan los padres a sus hijos en casa. En esta tesis nos enfocaremos en controlar las rutas de las unidades de trasporte que se desplazan, de manera que el conductor que busca como evadir las reglas viales, maneja como si estuviese solo en la calle, ocasionando accidentes por doquier y sin conciencia alguna, en el tránsito y en diversos factores que afectan el bienestar.

Podemos separar el problema vial en tres grandes categorías: la cultura del conductor al manejar en la ciudad, la excesiva cantidad de vehículos motorizados y la falta de una adecuada señalización en el país. Ambos radican en un desorden descomunal y en el excesivo tiempo que toma llegar de un destino al otro.

Los conductores con frecuencia efectúan cambios de ruta no autorizados en tramos cortos, exceden los límites de velocidad, entran en carreras por el tiempo de entrega y peor aún para control interno disciplinario subsisten anticuados relojes mecánicos para controlar los retrasos acumulados en diversas locaciones de una ruta.

A partir de estos dispositivos, se originan diversos problemas tales como: errores en la interpretación de la tarjeta de marcado, accidentes por apurarse a marcar "su tarjeta de control", relojes sin sincronía horaria, el hecho que la unidad de transporte nunca pase por el reloj de control y finalmente que en cada terminal de transporte se requiere una "máquina" sumadora humana de retrasos según las tarjetas o "el controlador" para dar salidas o castigo a cada unidad de su empresa, el trabajo de los controladores con frecuencia genera suspicacias por favoritismos a las unidades. Siendo estos quienes finalmente terminan de empeorar el servicio de transporte cuando no se actúan en forma transparente.

1.2. Formulación del Problema.

1.2.1 Problema General:

El problema general identificado del transporte donde no tenemos un control, rutas identificadas, la falta de señalización, ni un sistema automatizado donde podamos visualizar y controlar on-line para tomar medidas correctivas en el momento, se realiza la siguiente pregunta:

¿De qué manera el diseño de un sistema de ruta con GPS/4G Lte controla las unidades de la empresa NETTELCOM SAC?

1.2.2 Problemas Específicos:

Los problemas específicos detectados frente a la problemática general, se formulan las siguientes preguntas:

- ¿De qué manera el diseño de un sistema de ruta con GPS/4G Lte controla el tiempo en las unidades de la empresa Nettelcom SAC?
- ¿De qué manera el diseño de un sistema de ruta con GPS/4G Lte controla la distancia en las unidades de la empresa Nettelcom SAC?
- ¿De qué manera el diseño de un sistema de ruta con GPS/4G Lte controla el consumo en las unidades de la empresa Nettelcom SAC?
- ¿De qué manera el diseño de un sistema de ruta con GPS/4G Lte controla la velocidad en las unidades de la empresa Nettelcom SAC?

1.3. Objetivos de la Investigación.

1.3.1 Objetivo General:

El objetivo general que se plantea es: Determinar el diseño de un sistema de ruta con GPS/4G Lte controla las unidades de la empresa NETTELCOM SAC

1.3.2 Objetivos Específicos:

- Determinar la relación que existe entre el diseño de un sistema de ruta con GPS/4G Lte controla el tiempo en las unidades de la empresa Nettelcom SAC.
- Determinar la relación que existe el diseño de un sistema de ruta con GPS/4G Lte controla la distancia en las unidades de la empresa Nettelcom SAC.
- Determinar la relación que existe entre el diseño de un sistema de ruta con GPS/4G Lte controla el consumo en las unidades de la empresa Nettelcom SAC.

- Determinar la relación que existe entre el diseño de un sistema de ruta con GPS/4G Lte controla la velocidad en las unidades de la empresa Nettelcom SAC.

1.4. Justificación del Proyecto de Investigación.

1.4.1. Económico

- Al ser tecnológicas de fácil obtención, y de software libre.
- Al tener rutas más seguras y directas se optimiza los costos de producción.
- La unidad ante una eventualidad podrá ser soluciona en menor tiempo permitiendo su retorno en el menor tiempo posible al siguiente trabajo.
- Se evitaría costos innecesarios en el control de ruta como un personal adicional para el marcaje de ruta.

1.4.2. Tecnológica

El proyecto se justifica tecnológicamente porque permitirá a la empresa contar con un mecanismo automático para el control para la ruta de los vehículos, lo que permitirá tener información precisa en todo momento sobre las unidades conociendo su ubicación y recorrido.

1.4.3. Académica

Este proyecto se justifica académicamente porque se pondrá en práctico los conocimientos aprendidos, se tendrá que realizar investigación de la

nuevas tecnologías el cual nos proporcionara la información al momento de desarrollar nuestra tesis.

1.4.4. Seguridad

Ante cualquier eventualidad en ruta sea por: avería mecánica, falta de combustible, calle bloqueada, alto tráfico, perdida de llave, robo, etc. se podrá tomar la información de la ubicación, en conjunto con el informe del conductor, así dar la solución al problema en forma adecuada reduciendo los tiempos a la eventualidad que se presente on-site.

1.5. Limitaciones y Facilidades

1.5.1. Limitaciones

- Cobertura 4G Lte, por la empresa móvil que vea afectada en el servicio.
- En el desarrollo de un software y/o la implementación de un software desde un Web Server el cual nos presentara limitación según sea el requerimiento.
- Oferta de los Operadores móviles no acorde a la realidad del mercado.

1.5.2. Facilidades

- Tecnología vigente, ya que el sistema GPS se utiliza en la actualidad con gran demanda.
- El Sistema Localización es accesible y gratuito en los receptores GPS.

- Los Beneficios que permita mejora en ahorro de combustible y tiempo/hombre en el desplazamiento de las unidades.

II. MARCO TEÓRICO

2.1. Antecedentes del estudio

De acuerdo con Bisquerra (2004) los antecedentes de investigaciones tienen importancia en la concepción de análisis y avance de la comprensión de las variables en diversos contextos, de ello se extrae las construcciones metodológicas, así como la relación del análisis de las conclusiones que sirven como base fundamental al trabajo de investigación, por ello se proponen los siguientes estudios

2.2.1. Nacionales

BOCANEGRA URETA, RUBEN, tesis **"Desarrollo de una aplicación web para el monitoreo de vehículos con dispositivos GPS que comercializa una empresa de telecomunicaciones"**, tesis de Ingeniera Universidad Ricardo Palma , Lima – Perú, 2012, contiene la solución propuesta que agrupa las funcionalidades representativas de las mejores soluciones de monitoreo del mercado y las presenta de manera objetiva a través de una propuesta completa y puntual donde el usuario final pueda dar uso de la solución a través de un navegador web.

2.2.2. Internacional

GARZON MANUEL, tesis **"Diseño de sistema que permita mantener un estándar de tiempo entre buses consecutivos de una misma ruta"**, tesis de Grado Ingeniería, Universidad Santiago de Cali, Cali - Colombia.2009, contiene las especificaciones y guía de diseño, para

construir un dispositivo que colabore en el proceso de estandarización de tiempos en los recorridos de buses que correspondan a una misma ruta.

CHILAN SOLEDISPA EDGAR, tesis **"Desarrollo de aplicación para presentar reportes gráficos (rutas vehiculares) que se visualicen en Google Maps"**. Tesis Ingenieria, Universidad de Guayaquil, Guayaquil – Ecuador, 2013, En esta tesis se planteó desarrollar una aplicación web que permita visualizar reportes gráficos de los desplazamientos vehiculares en los mapas de Google Maps basándonos en la base de datos del sistema de rastreo. Este trabajo de desarrollo se ha realizado en base a la recopilación de información la cual ha sido obtenida a través de las páginas web oficiales de Microsoft Visual Studio 2010, Rastrac, SQL Server y Google Maps.

2.2. El GPS

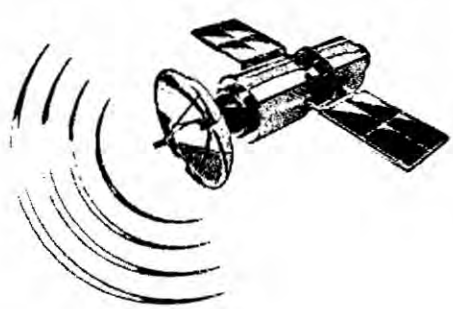
2.2.1. Descripción del GPS

DANIEL VARGAS, El Sistema GPS, disponible en https://vargasdaniel27.wordpress.com/2008/04/28/el-sistema-gps-telecomunicaciones/, articulo web, consultada el 30 de enero del 2016, El GPS o sistema de posicionamiento Global, es un sofisticado sistema de orientación y navegación cuyo funcionamiento está basado en la recepción y procesamiento de las informaciones emitidas por una constelación de 24 a 28 satélites (más 4 satélites de cambio), conocida como NAVSTAR, orbitando en diferentes alturas a unos 20.000 km. por encima de la superficie terrestre. Cada satélite, da dos vueltas diarias al planeta. Las

trayectorias y la velocidad orbital han sido calculadas para que formen una especie de red alrededor de la tierra (debe haber en todo momento cinco satélites a la vista en cualquier zona), de manera que un receptor GPS a cualquier hora del día (24 horas), en cualquier lugar, con independencia de las condiciones meteorológicas, pueda facilitar la posición que ocupa al captar y procesar las señales emitidas por un mínimo de tres satélites.

Los satélites GPS están ubicados en forma de una red alrededor de Globo terrestre a una altura determinada proporcionando información al receptor las 24 horas del día.

Figura N° 2.1 Satélite GPS



Fuente: GPS Global Security Disponible en http://www.gsp.com.co

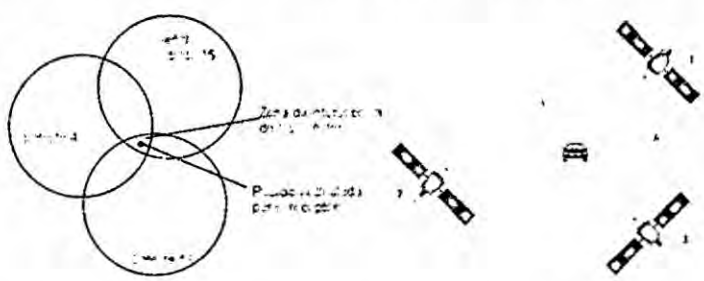
2.2.2. Funcionamiento del GPS

ANONIMO, **La revolución de la comunicación**, disponible en http://revolucioncomunicacion.weebly.com/gps.html, artículo web, consultado el 30 de enero de 2016, La información que es útil al receptor GPS para determinar su posición se llama efemérides. En este caso cada satélite emite sus propias efemérides, en la que se incluye la salud del

satélite (si debe o no ser considerado para la toma de la posición), su posición en el espacio, su hora atómica, información doppler, etc.

Cada satélite indica que el receptor se encuentra en un punto en la superficie de la esfera, con centro en el propio satélite y de radio la distancia total hasta el receptor. Obteniendo información de dos satélites queda determinada una circunferencia que resulta cuando se intersectan las dos esferas en algún punto de la cual se encuentra el receptor. Teniendo información de un tercer satélite, se elimina el inconveniente de la falta de sincronización entre los relojes de los receptores GPS y los relojes de los satélites. Y es en este momento cuando el receptor GPS puede determinar una posición 3D exacta (latitud, longitud y altitud).

Figura N° 2.2 Representación para el cálculo matemático



Fuente: GPS Global Security *Disponible en http://www.gsp.com.co*

Explicaremos desde la triangulación podemos realizar y localizar un punto determinando sobre la capa terrestre.

2.2.2.1.Triangulación

La base del GPS es la "triangulación" desde los satélites

- Distancias. Para "triangular", el receptor de GPS mide distancias utilizando el tiempo de viaje de señales de radio.

- Tiempo. Para medir el tiempo de viaje de estas señales, el GPS necesita un control muy estricto del tiempo y lo logra con ciertos trucos.

- Posición. Además de la distancia, el GPS necesita conocer exactamente donde se encuentran los satélites en el espacio. Orbitas de mucha altura y cuidadoso monitoreo, le permiten hacerlo.

- Corrección. Finalmente el GPS debe corregir cualquier demora en el tiempo de viaje de la señal que esta pueda sufrir mientras atraviesa la atmósfera.

- Fuentes de error.

2.2.2.2. Triangulación desde los satélites

Aunque pueda parecer improbable, la idea general detrás del GPS es utilizar los satélites en el espacio como puntos de referencia para ubicaciones aquí en la tierra.

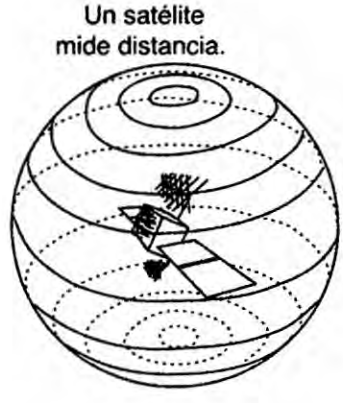
Esto se logra mediante una muy, pero muy exacta, medición de nuestra distancia hacia tres satélites, lo que nos permite "triangular" nuestra posición en cualquier parte de la tierra (Figura N° 2.2 en la página 11). Consideremos primero como la medición de esas distancias nos permite ubicarnos en cualquier punto de la tierra.

La gran idea, Geométricamente, es:

Supongamos que medimos nuestra distancia al primer satélite (véase en la Figura 2.3) y resulta ser de 17 000 Km (10 500 millas)

Sabiendo que estamos a 17 000 Km (10 500 millas) de un satélite determinado no podemos, por lo tanto, estar en cualquier punto del universo sino que esto limita nuestra posición a la superficie de una esfera que tiene como centro dicho satélite y cuyo radio es de 17 000 Km (10 500 millas).

Figura N° 2.3 Ubicación a un solo satélite



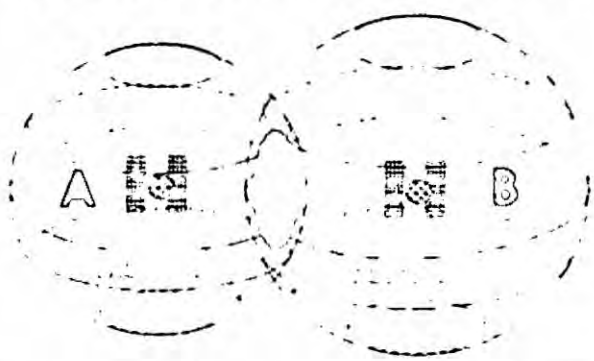
Fuente: Satélites de Posicionamiento Global disponible en http://hyperphysics.phy-astr.gsu.edu/hbasees/gps.html

A continuación medimos nuestra distancia a un segundo satélite (véase la Figura 2.4 en la página 14) y descubrimos que estamos a 19 000 Km (11 800 millas) del mismo.

Esto nos dice que no estamos solamente en la primera esfera, correspondiente al primer satélite, sino también sobre otra esfera que se encuentra a 19 000 km (11 800 millas) del segundo satélite. En otras palabras, estamos en algún lugar de la circunferencia que resulta de la intersección de las dos esferas.

Si ahora medimos nuestra distancia a un tercer satélite (véase la Figura 2.5 en la página 15) y descubrimos que estamos a 13.000 millas del mismo, esto limita nuestra posición aún más, a los dos puntos en los cuales la esfera de 13 000 Km (8 000 millas) corta la circunferencia que resulta de la intersección de las dos primeras esferas.

Figura N° 2.4 Ubicación a dos satélites



Fuente: Satélites de Posicionamiento Global disponible en http://hyperphysics.phy-astr.gsu.edu/hbasees/gps.html

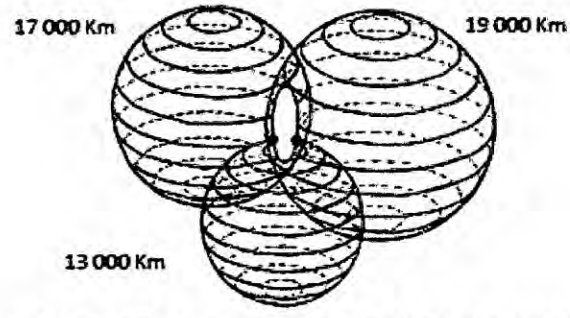
Que midiendo nuestra distancia a tres satélites limitamos nuestro posicionamiento a solo dos puntos posibles.

Para decidir cuál de ellos es nuestra posición verdadera, podríamos efectuar una nueva medición a un cuarto satélite. Pero normalmente uno

de los dos puntos posibles resulta ser muy improbable por su ubicación demasiado lejana de la superficie terrestre y puede ser descartado sin necesidad de mediciones posteriores.

Para tener cálculos más exactos sobre la ubicación es cuando interviene un cuarto satélite o un quinto satélite para ganar más exactitud.

Figura N° 2.5 Ubicación a tres satélites



Fuente: Satélites de Posicionamiento Global disponible en http://hyperphysics.phy-astr.gsu.edu/hbasees/gps.html

Con lo cual podemos concluir.

- Nuestra posición se calcula en base a la medición de las distancias de los satélites.
- Para la realización de los cálculos se necesita cuatro satélites los cuales nos proporciona como dato la distancia, para determinar la posición exacta.

2.2.2.3. Medición de la distancia de los satélites

PEDRO GUTOVNIK, **Como Funciona el GPS**, disponible en http://gutovnik.com/como_func_sist_gps.htm, referencia web, consultada el 30 de enero de 2016, Nuestra posición se calcula a partir de la medición de la distancia como mínimo de tres satélites. Pero, ¿cómo podemos medir la distancia hacia algo que está flotando en algún lugar en el espacio? Lo hacemos midiendo el tiempo que tarda una señal emitida por el satélite en llegar hasta nuestro receptor de GPS.

Matemáticamente, es la ecuación de velocidad y tiempo, la cual nos proporciona como resultante la distancia.

*D(distancia) = V(velocidad) * T(tiempo)*

Recordemos que "Si un auto viaja a 80 kilómetros por hora durante cuatro horas, ¿qué distancia recorrió?

*Velocidad (80 km/h) * Tiempo (4 horas) = Distancia (320 km)*

En el caso del GPS estamos midiendo una señal de radio, que sabemos que viaja a la velocidad de la luz, alrededor de 300.000 km por segundo.

Para medir la distancia que tenemos entre el GPS y su receptor es imprescindible tener el tiempo que demora la señal de radio.

a. Sincronización de relojes

El problema de la medición de ese tiempo es complicado. Los tiempos son extremadamente cortos. Si el satélite estuviera justo sobre nuestras cabezas, a unos 20.000 km de altura, el tiempo total de viaje de la señal hacia nosotros sería de algo más de 0.06 segundos. Estamos necesitando relojes muy precisos.

$$T(tiempo) = \frac{D(distancia)}{V(velocidad)}$$

$$T = \frac{20000Km}{300000\frac{Km}{seg}}$$

$$T = 0.06\ seg$$

Tenemos que lograr una sincronización entre el receptor GPS y el satélite teniendo en cuenta que se tendrá un retraso en la señal dependiendo de la distancia a recorrer.

Si se emite una señal desde el receptor GPS y el satélite, oiríamos dos versiones de la señal. Una de ellas inmediatamente, la generada por nuestro receptor GPS y la otra con cierto atraso, la proveniente del satélite, porque tuvo que recorrer alrededor de 20.000 km para llegar hasta nosotros. Podemos decir que ambas señales no están sincronizadas.

El tiempo de retardo necesario para sincronizar ambas señales es igual al tiempo de viaje de la señal proveniente del satélite. Supongamos que sea

de 0.06 segundos. Conociendo este tiempo, lo multiplicamos por la velocidad de la luz y ya obtenemos la distancia hasta el satélite.

*D (distancia) = Tiempo de Retardo * V (velocidad de la luz)*

*Dist. (18.000 km) = Tiempo de retardo (0.06 seg) * Vel. de la luz (300.000* $\frac{Km}{seg}$*)*

Es necesario tener relojes de alta precisión para lograr una toma de tiempo correcto entre la sincronización entre señales del satélite con el receptor GPS.

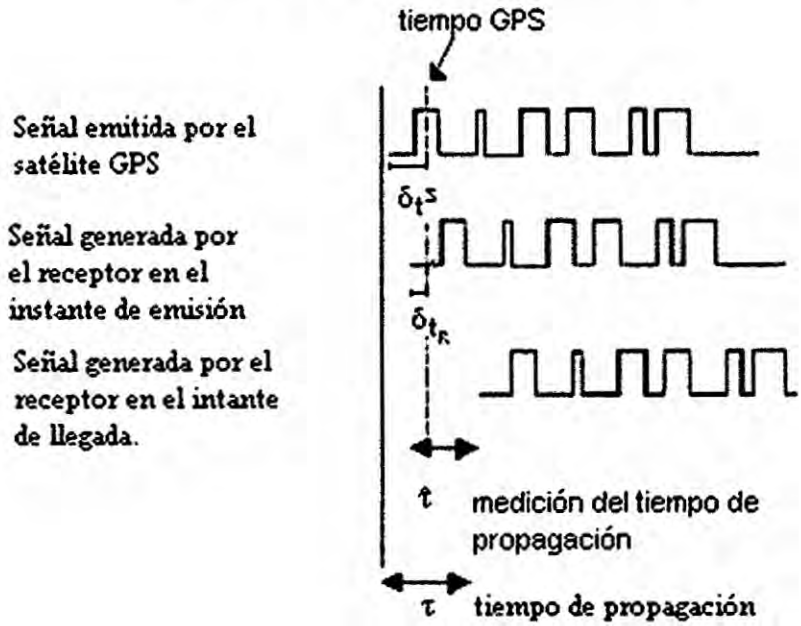
b. Código Aleatorio

Este Código Pseudo Aleatorio es una parte fundamental del GPS. Físicamente solo se trata de una secuencia o código digital muy complicado. O sea una señal que contiene una sucesión muy complicada de pulsos "on" y "off", (véase en la figura 2.6 en la página 19).

La señal es tan complicada que casi parece un ruido eléctrico generado por el azar. De allí su denominación de "Pseudo-Aleatorio".

La complejidad del código ayuda a asegurarnos que el receptor de GPS no se sintonice accidentalmente con alguna otra señal. Siendo el modelo tan complejo es altamente improbable que una señal cualquiera pueda tener exactamente la misma secuencia.

Figura N° 2.6 Ciclos de Sincronismo



Fuente: Sistema NavStart disponible en

http://fcaglp.unlp.edu.ar/referenciacion/index.php?title=Portal:Sistema_NAVSTAR_GPS&redirect=no

Dado que cada uno de los satélites tiene su propio y único Código Pseudo Aleatorio, esta complejidad también garantiza que el receptor no se confunda accidentalmente de satélite. De esa manera, también es posible que todos los satélites trasmitan en la misma frecuencia sin interferirse mutuamente. Esto también complica a cualquiera que intente interferir el sistema desde el exterior al mismo.

Pero hay otra razón para la complejidad del Código Pseudo Aleatorio, una razón que es crucial para conseguir un sistema GPS económico.

El código permite el uso de la "teoría de la información" para amplificar las señales de GPS. Por esa razón las débiles señales emitidas por los satélites pueden ser captadas por los receptores de GPS sin el uso de grandes antenas.

c. Control del tiempo

Si la medición del tiempo de viaje de una señal de radio es clave para el GPS, los relojes que empleamos deben ser muy exactos, dado que si miden con un desvío de un milésimo de segundo, a la velocidad de la luz, ello se traduce en un error de 300 km. Por el lado de los satélites, el timing (frecuencia) es casi perfecto porque llevan a bordo relojes atómicos de increíble precisión.

Por el alto costo que significa implementar un reloj atómico en cada sistema GPS los diseñadores encontraron otra solución la cual consiste en una medición satelital adicional. Resulta que si tres mediciones perfectas pueden posicionar un punto en un espacio tridimensional, cuatro mediciones imperfectas pueden lograr lo mismo.

Una medición adicional remedia el desfasaje del timing.

Si todo fuera perfecto (es decir que los relojes de nuestros receptores GPS lo fueran), entonces todos los rangos (distancias) a los satélites se interceptarían en un único punto (que indica nuestra posición). Pero con relojes imperfectos, una cuarta medición efectuada como control cruzado, NO intersectará con los tres primeros.

De esa manera la computadora de nuestro GPS detectará la discrepancia y atribuirá la diferencia a una sincronización imperfecta con la hora universal. Dado que cualquier discrepancia con la hora universal afectará a las cuatro mediciones, el receptor buscará un factor de corrección único que siendo aplicado a sus mediciones de tiempo hará que los rangos coincidan en un solo punto. Dicha corrección permitirá al reloj del receptor ajustarse nuevamente a la hora universal, una vez que el receptor de GPS aplica dicha corrección al resto de sus mediciones, obtenemos un posicionamiento preciso.

Una consecuencia de este principio es que cualquier GPS decente debe ser capaz de sintonizar al menos cuatro satélites de manera simultánea. En la práctica, casi todos los GPS en venta actualmente, acceden a más de 6, y hasta a 12, satélites simultáneamente.

d. Conocer la ubicación de los satélites.

Para los cálculos hemos tomado una ubicación exacta de los satélites en sus orbitas y de esta manera se podía usar como referencia para la ubicación del Sistema GPS.

¿Pero, cómo podemos saber dónde están exactamente? Todos ellos están flotando a unos 20.000 km de altura en el espacio. Un satélite a gran altura se mantiene estable, La altura de 20.000 km es en realidad un gran beneficio para este caso, porque algo que está a esa altura está bien

despejado de la atmósfera. Eso significa que orbitará de manera regular y predecible mediante ecuaciones matemáticas sencillas.

El Control Constante agrega precisión

Las órbitas básicas son muy exactas pero con el fin de mantenerlas así, los satélites de GPS son monitorizados de manera constante por el Departamento de Defensa de los EEUU. Ellos utilizan radares muy precisos para controlar constantemente con exactitud la altura, posición y velocidad de cada satélite.

Los errores que ellos controlan son los llamados errores de efemérides, o sea evolución orbital de los satélites. Estos errores se generan por influencias gravitacionales del sol y de la luna y por la presión de la radiación solar sobre los satélites.

e. Corrigiendo el mensaje

Una vez que el Departamento de Defensa ha medido la posición exacta de un satélite, vuelven a enviar dicha información al propio satélite. De esa manera el satélite incluye su nueva posición corregida en la información que transmite a través de sus señales a los GPS. Esto significa que la señal que recibe un receptor de GPS no es solamente un Código Pseudo Aleatorio con fines de timing. También contiene un mensaje de navegación con información sobre la órbita exacta del satélite.

De todo lo visto podemos concluir:

- La distancia al satélite se determina midiendo el tiempo que tarda una señal de radio, emitida por el mismo, la cual se enlaza con nuestro receptor de GPS.
- Comparando cuanto retardo existe entre la llegada del Código Pseudo Aleatorio proveniente del satélite y la generación del código de nuestro receptor de GPS, podemos determinar cuánto tiempo le llevó a dicha señal llegar hasta nosotros.
- Multiplicamos dicho tiempo de viaje por la velocidad de la luz y obtenemos la distancia al satélite.
- Para utilizar los satélites como puntos de referencia debemos conocer su ubicación con exactitud en todo momento.
- La información sobre errores es enviada a los satélites para que puedan corregir su ubicación a su vez sea retransmitan su posición corregida junto con sus señales de timing.

2.3. Microcontroladores

VIENNA UNIVERSITY OF TECHNOLOGY, GUNTHER GRIDLING, **Introduction to Microcontrollers**, Disponible en: https://ti.tuwien.ac.at/ecs/teaching/courses/mclu/theory-material/Microcontroller.pdf, articulo web, consultado el 01 de enero del 2016, Nos indica que Un microcontrolador (abreviado µC, UC o MCU) es un circuito integrado programable, capaz de ejecutar las órdenes grabadas en su memoria. Está compuesto de varios bloques funcionales, los cuales

cumplen una tarea específica. Un microcontrolador incluye en su interior las tres principales unidades funcionales de una computadora: unidad central de procesamiento, memoria y periféricos de entrada/salida.

Donde podemos concluir que un microcontrolador es aquel dispositivo que se encarga de controlar los procesos para el cual ha sido diseñado en su programación.

2.3.1 Características

Algunos microcontroladores se caracterizan por poder utilizar 4 bits (ejemplo 1010), y funcionan a velocidad de reloj con frecuencias muy bajas como por ejemplo a 4 kHz, ocasionando un pequeño consumo donde la potencia estaría alrededor de los milivatios (mW).

El microcontrolador puede mantenerse en espera de un evento detectando mediantes sus periféricos algún evento de cambio; el consumo de energía del microcontrolador en espera (reloj de la CPU y los periféricos de la mayoría) es muy pequeño casi siempre está alrededor de los nanovatios, lo que hace que ideal para trabajos donde se tiene una batería dando una larga duración debido a su poco consumo.

Un microcontrolador, no contiene ninguna información, su memoria ROM está vacía en la espera de datos. Para que pueda controlar uno o varios procesos es indispensable crear algún programa en lenguaje ensamblador para ser grabado en la memoria EEPROM del controlador y así poder usarlo en los procesos para el cual ha sido diseñado; sin embargo, para

que el programa pueda ser grabado en la memoria del microcontrolador en la EEPROM, debe ser compilado en sistema numérico hexadecimal que es el sistema que hace funcionar al microcontrolador cuando se ponga en marcha con el resto de sus periféricos de entrada/salida.

El procesador es el parte principal del sistema es el núcleo donde se encarga de analizar y procesar toda la información que pasa por él, entonces cuanto más rápido sea nuestro procesador más rápido ejecutara las instrucciones que tiene contenido en la memoria.

2.3.3.1 Unidad aritmético-lógica (ALU)

JOSE ADOLFO GONZALES, **Introducción a los Microcontroladores**, España, McGraw-Hill, 1994, nos indica que Los Procesadores están compuestos por circuitos que realizan operaciones lógicas y matemáticas, para dicho trabajo se le asigna una unidad completa. Es aquí donde se realizan las operaciones de suma, resta y el álgebra de Boole.

En la actualidad han mejorado su rendimiento y los procesadores ya incorporan una o más ALU, algunas ALU se encargan de realizar operaciones complejas, como en coma flotante, de acuerdo a su trabajo se le otorgado un nombre como por ejemplo el coprocesador matemático.

Su impacto en las prestaciones del procesador es también importante porque, dependiendo de su potencia, tareas más o menos complejas, pueden hacerse en tiempos muy cortos.

a. Registros

Son pequeños espacios de memorias necesarios para un microprocesador, debido que estos espacios de memoria sirven para almacenar los datos para las operaciones que deben realizar los circuitos del procesador. Los registros almacenan los resultados de las instrucciones, que son cargados desde la memoria externa o en ella misma.

VIENNA UNIVERSITY OF TECHNOLOGY, GUNTHER GRIDLING, **Introduction to Microcontrollers**, Disponible en: https://ti.tuwien.ac.at/ecs/teaching/courses/mclu/theory-material/Microcontroller.pdf, articulo web, consultado el 01 de enero del 2016 nos indica que una parte de los registros está destinada a los datos, es la que determina uno de los parámetros más importantes de cualquier microprocesador. Cuando escuchamos que un procesador es de 4, 8, 16, 32, 64 0 128 bits, nos estamos refiriendo a procesadores que realizan sus operaciones con registros de datos de ese tamaño, y por supuesto, esto determina muchas de las potencialidades de dichas máquinas.

A mayor número de bits de los registros de datos del procesador, mayores serán participación para los datos, en cuanto a poder de cómputo y velocidad de ejecución, ya que este parámetro determina la potencia que se puede incorporar al resto de los componentes del sistema.

b. Unidad de control

JARED TIRADO MARTINEZ, **Microcontroladores**, disponible en https://issuu.com/jarettto/docs/loaiza-microcontroladores_-1-, articulo web, consultada el 31de enero del 2016, indica que la unidad de control es las más importantes en el procesador, en ella recae la lógica necesaria para la decodificación y ejecución de las instrucciones, el control de los registros, la ALU, los buses y cuanta cosa más se quiera meter en el procesador.

La unidad de control es un parte fundamental que determina el potencial del procesador, debido a que su estructura determina sus parámetros como el conjunto de instrucciones que pueda ejecutar, velocidad de ejecución, frecuencia de la máquina, el bus que puede tener el sistema, manejo de interrupciones.

2.3.1.2 Buses

JARED TIRADO MARTINEZ, **Microcontroladores**, disponible en https://issuu.com/jarettto/docs/loaiza-microcontroladores_-1-, articulo web, consultada el 31de enero del 2016, Es el medio por la cual se comunica los distintos componentes del procesador para interactuar entre sí, eventualmente los buses o una parte de ellos estarán reflejados en los pines del encapsulado del procesador.

En los microcontroladores, no es normal que los buses estén reflejados en el encapsulado del circuito, debido que estos se destinados principalmente a las E/S de propósito general y periféricos del sistema.

Existen tres tipos de buses:

- Dirección: Se utiliza para seleccionar al dispositivo con el cual se quiere trabajar o en el caso de las memorias, seleccionar el dato que se desea leer o escribir.
- Datos: Se utiliza para mover los datos entre los dispositivos de hardware (entrada y salida).
- Control: Se utiliza para gestionar los distintos procesos de escritura/lectura y controlar la operación de los dispositivos del sistema.

2.3.1.3. Conjunto de instrucciones

Es el grupo de líneas de comando (Instrucciones) que puede realizar el procesador, limitando en sus acciones debidas que las instrucciones son limitadas por cada modelo de procesador.

UNIVERSIDAD NACIONAL DE INGENIERIA, O.BUSTILLOS y C. MARTINEZ, **Microprocesadores vs Microcontroladores**, disponible en define https://electrouni.files.wordpress.com/2010/12/micro-vs-microcon.pdf, articulo web, consultado 31 de enero 2016, indica que las operaciones básicas que puede realizar el procesador, que conjugadas y organizadas forman lo que conocemos como software. El conjunto de instrucciones, vienen siendo como las letras del alfabeto, el elemento básico del lenguaje, que organizadas adecuadamente permiten escribir palabras, oraciones y cuanto programa se le ocurra. Existen dos tipos básicos de almacenaje de instrucciones, que determinan la arquitectura del

procesador estos son, el conjunto de instrucciones complejas (CISC) y Conjunto de Instrucciones Reducidas (RISC).

El CISC, del inglés Complex instruction set Computing, Computadora de Conjunto de Instrucciones Complejo. Los microprocesadores CISC tienen un conjunto de instrucciones que se caracteriza por ser muy amplio y que permiten realizar operaciones complejas entre operandos situados en la memoria o en los registros internos. Este tipo de repertorio dificulta el paralelismo entre instrucciones, por lo que en la actualidad, la mayoría de los sistemas CISC de alto rendimiento convierten las instrucciones complejas en varias instrucciones simples del tipo RISC, llamadas generalmente microinstrucciones.

Conclusiones:

- Los procesadores son los que posibilita el paralelismo en la ejecución.

- Reducen los accesos a memoria.

- Los procesadores más modernos, tradicionalmente basados en arquitecturas CISC, implementan mecanismos de traducción de instrucciones CISC a RISC.

- Los procesadores de los microcontroladores PIC son de tipo RISC.

2.3.2 Memoria

La memoria RAM está destinada al almacenamiento de información temporal que será utilizada por el procesador para realizar cálculos u otro tipo de operaciones lógicas. En el espacio de direcciones de memoria RAM

se ubican además los registros de trabajo del procesador y los de configuración y trabajo de los distintos periféricos del microcontrolador. Es por ello que en la mayoría de los casos, aunque se tenga un espacio de direcciones de un tamaño determinado, la cantidad de memoria RAM de que dispone el programador para almacenar sus datos es menor que la que puede direccionar el procesador.

La memoria en los microcontroladores debe estar ubicada dentro del mismo encapsulado, esto es así la mayoría de las veces, porque la idea fundamental es mantener el grueso de los circuitos del sistema dentro de un solo integrado.

Existen cinco tecnologías, que mayor utilización tienen o han tenido:

2.3.2.1. Máscara ROM (Read Only Memory).

En este caso no se "graba" el programa en memoria sino que el microcontrolador se fabrica un programa determinado.

2.3.2.2 Memoria PROM (Programmable Read-Only Memory).

También conocida como OTP (One Time Programmable). Este tipo de memoria también es conocida como PROM o simplemente ROM. Los microcontroladores con memoria OTP se pueden programar una sola vez, con algún tipo de grabador. Se utilizan en sistemas donde el programa no requiera futuras actualizaciones también para sistemas que requieren serialización de datos, almacenados como constantes en la memoria de programas.

2.3.2.3 Memoria EPROM (Erasable Programmable Read Only Memory).

Los microcontroladores con este tipo de memoria son muy fáciles de identificar porque su encapsulado es de cerámica y llevan encima una ventanita de vidrio desde la cual puede verse la oblea de silicio del microcontrolador.

Se fabrican así porque la memoria EPROM es reprogramable, pero antes debe borrase, y para ello hay que exponerla a una fuente de luz ultravioleta.

2.3.2.4 EEPROM (Electrical Erasable Programmable Read Only Memory).

Son aquellas memorias que pueden ser borrados eléctricamente, por lo que la ventanilla de cristal de cuarzo y los encapsulados cerámicos no son necesarios.

Una característica destacable de este tipo de microcontrolador es que fue en ellos donde comenzaron a utilizarse los sistemas de programación en el sistema que evitan tener que sacar el microcontrolador de la tarjeta que lo aloja para hacer actualizaciones al programa.

2.3.2.5 MEMORIA FLASH.

En el campo de las memorias reprogramables para microcontroladores, son el último avance tecnológico en uso a gran escala, y han sustituido a los microcontroladores con memoria EEPROM.

A las ventajas de las memorias flash se le adicionan su gran densidad respecto a sus predecesoras lo que permite incrementar la cantidad de memoria de programas. Pueden además ser programadas con las mismas tensiones de alimentación del microcontrolador, el acceso en lectura y la velocidad de programación es superior.

2.3.3. Interrupciones

Las interrupciones son llamadas a subrutina generadas por los dispositivos externos de tipo físico sea por un sensor detectando algún cambio, al contrario de las subrutinas normales de un programa en ejecución. Como el salto de subrutina no es parte de la programación o secuencia de ejecución programada.

El microcontrolador guarda el cambio en la pila de memoria del procesador y entra a ejecutar un código llamado "manejador de interrupciones" que atiende al periférico externo que ha generado la interrupción. Al terminar la rutina, una instrucción nativa le indica al procesador el fin del proceso de la interrupción. En ese momento el controlador restablece el estado anterior, y continúa ejecutando el programa antes de la interrupción continua como si nada hubiese pasado. Las rutinas que atienden a las interrupciones deben ser lo más cortas posibles para que el rendimiento del sistema sea optimo, debido cuando una interrupción es atendida, todas las demás interrupciones quedan en estado de espera hasta que finalice la interrupción.

Un proceso de interrupción y su atención por parte del procesador, tiene la siguiente secuencia:

- Se produce un evento, para el cual queremos que el procesador ejecute un programa en especial, este proceso tiene la prioridad de ejecutarse en el menor tiempo posible antes de ser atendido o no sabemos en qué momento debe ser atendido.
- El circuito encargado de detectar el estado del evento se activa, y como consecuencia, activa la entrada de interrupción del procesador.
- La unidad de control detecta que se ha producido un estado de interrupción y "levanta" una alerta para registrar dicha situación; de esta forma si las condiciones que provocaron el evento desaparecen y el circuito encargado de detectarlo desactiva la entrada del estado de interrupción del procesador, ésta se producirá de cualquier modo, porque ha sido registrada.
- La unidad de ejecución termina con la instrucción en curso y antes de comenzar con la siguiente instrucción verifica que no se ha ejecutado algún estado de interrupción.
- Se desarrolla un proceso que permite guardar el estado actual del programa en ejecución y saltar a una dirección donde se debe ejecutar algún código que está almacenado en memoria, donde está la primera instrucción de la subrutina de atención a interrupción.

- Se ejecuta el código de atención a interrupción, esta es la parte "consciente" de todo el proceso porque es donde se realizan las acciones propias de la atención al estado de interrupción y el programador desarrolla una tarea específica para dicho estado de interrupción.
- Cuando el código de atención al estado de interrupción se ejecuta la instrucción de retorno, se desencadena el proceso de restauración del procesador retornando al estado en que estaba antes de la atención a la interrupción.

2.3.4. Periféricos

Son dispositivos por los cuales nos informaran de algún estado de cambio registrando una serie de interrupción de esa forma va proporcionando información al sistema. A continuación describiremos algunos de los periféricos que con mayor frecuencia encontraremos en los microcontroladores.

2.3.5. Entrada y salida

También conocidos como puertos de E/S (entrada/salida), generalmente agrupadas en puertos de 8 bits de longitud, donde permite leer o escribir datos del exterior en ellos, desde el interior del microcontrolador, normalmente el trabajo que se realiza seria de on/off (encendido/apagado) lo que se conoce como 0 y 1 lógico.

Algunos puertos de E/S tienen características especiales que le permiten manejar salidas con variaciones de corriente según sea el caso, o incorporan mecanismos propios para realizar una interrupción para el procesador.

Cualquier pin de E/S puede ser considerada E/S de propósito general, pero como los microcontroladores no pueden poseer infinitos pines de E/S, tampoco los pines que necesitemos para el diseño, tenemos que ajustarnos a las características que nos proporciona el fabricante, las E/S de propósito general comparten los pines con otros periféricos. Para usar un pin con una característica particular debemos ajustarnos y configurarlo mediante los registros destinados a él.

2.3.6. Temporizadores y Contadores

Son circuitos sincrónicos para el conteo de los pulsos que llegan a su poder para conseguir la entrada de reloj. Si la fuente de un gran conteo es el oscilador interno del microcontrolador es común que no tengan un pin asociado, y en este caso trabajan como temporizadores. Por otra parte, cuando la fuente de conteo es externa, entonces tienen asociado un pin configurado como entrada, este es el modo contador.

Los temporizadores son uno de los periféricos más habituales en los microcontroladores y se utilizan para muchas tareas, como por ejemplo, la medición de frecuencia, implementación de relojes, para el trabajo de conjunto con otros periféricos que requieren una base estable de tiempo

entre otras funcionalidades. Es frecuente que un microcontrolador típico incorpore más de un temporizador/contador e incluso algunos tienen arreglos de contadores.

2.4. Software Libre

RICHARD STALLMAN, **CONCEPTOS DEL FREE SOFTWARE**, disponible http://e-forma.kzgunea.eus/mod/book/view.php?id=649&chapterid=1285, articulo web, consultado 15 de enero de 2016, indica que, "Software libre" es el software que respeta la libertad de los usuarios y la comunidad, significa que los usuarios tienen "la libertad para ejecutar, copiar, distribuir, estudiar, modificar y mejorar el software". Es decir, el "software libre" es una cuestión de libertad, no de precio. Para entender el concepto, piense en "libre" como en "libre expresión".

Un programa es software libre si los usuarios tienen las cuatro libertades esenciales:

- Libertad 0: La libertad de ejecutar el programa como se desea, con cualquier propósito.
- Libertad 1: La libertad de estudiar cómo funciona el programa, y cambiarlo para que funcione según sus necesidades. El acceso al código fuente es una condición necesaria para ello.
- Libertad 2: La libertad de redistribuir copias para ayudar a su prójimo.
- Libertad 3: La libertad de distribuir copias de sus versiones modificadas a terceros.

Esto le permite ofrecer a toda la comunidad la oportunidad de beneficiarse de las modificaciones. El acceso al código fuente es una condición necesaria para ello.

Bajo el principio de que un software puede ser adquirido, estudiado, modificarlo, y redistribuirlo, cumple con estos requerimientos para ser llamado software libre.

2.4.1. Lenguaje de Programación Java Script

DOUGLAS CROCKFORD, **Java Script**, disponible en http://www.crockford.com/javascript/javascript.html, articulo web, consultado el 15 de enero del 2016, Nos define a JavaScript (abreviado comúnmente JS) como un lenguaje de programación interpretado, dialecto del estándar ECMAScript. Se define como orientado a objeto, basado en prototipos, imperativos, débilmente tipado y dinámico.

Se utiliza dos partes lado cliente y lado servidor, el lado del cliente (client-side), implementado como parte de un navegador web permitiendo mejoras la interface de usuario y páginas web dinámicas. Lado del servidor (Server-side JavaScript o SSJS). Su uso en aplicaciones externas a la web, por ejemplo aplicaciones de escritorio.

DOUGLAS CROCKFORD, **Java Script**, disponible en http://www.crockford.com/javascript/javascript.html, articulo web, consultado el 15 de enero del 2016, indica que todos los navegadores modernos interpretan el código JavaScript integrado en las páginas web.

Una página tiene una implementación del Document Object Model (DOM) donde la web se provee de un lenguaje JavaScript.

Características

- Entorno de ejecución

  JavaScript casi siempre depende del entorno donde va a ser ejecutado (por ejemplo, en un navegador web) para ofrecer objetos y métodos por los que los scripts pueden intercambiar información con el "mundo exterior

- Funciones variádicas

  Un número indefinido de parámetros pueden ser pasados a la función. La función puede acceder a ellos a través de los parámetros o también a través del objeto argumentos locales.

- Funciones como métodos

  A diferencia de muchos lenguajes orientados a objetos, "no hay distinción entre la definición de función y la definición de método". Más bien, la distinción se produce durante la llamada a la función; permite a una función ser llamada como un método. Cuando una función es llamada como un método de un objeto, la palabra clave this, que es una variable local a la función, representa al objeto que invocó dicha función.

- Arrays y la definición literal de objetos

  Al igual que muchos lenguajes de script, arrays y objetos (arrays asociativos en otros idiomas) pueden ser creados con una sintaxis

abreviada. De hecho, estos literales forman la base del formato de datos JSON.

- Expresiones regulares

  JavaScript también es compatible con expresiones regulares como otros lenguajes de programación como Perl, que proporcionan una sintaxis concisa y poderosa para la manipulación de texto que es más sofisticado que las funciones incorporadas a los objetos de tipo string.

- Uso en Páginas Web

  El uso más común de JavaScript es escribir funciones incluidas en el código HTML y que interactúan con el Document Object Model (DOM o Modelo de Objetos del Documento) de la página

  El código JavaScript puede ejecutarse en el cliente es decir desde su navegador del usuario (en lugar de en un servidor remoto), el navegador puede ejecutar las acciones del usuario con rapidez.

2.4.2. Lenguaje ensamblador (Assembler)

WIKIPEDIA, **Lenguaje Ensamblador**, disponible en https://es.wikipedia.org/wiki/Lenguaje_ensamblador, articulo web, consultado 17 de enero de 2016, indica que el lenguaje ensamblador, o assembler, es un lenguaje de programación de bajo nivel para los microprocesadores, microcontroladores y otros circuitos integrados programables.

Implementa una representación simbólica de los códigos de máquina binarios es decir 1 y 0, otra constante necesaria para programar una arquitectura dada de CPU y constituye la representación más directa del código máquina específico para cada arquitectura legible por un programador.

Es el software de bajo nivel utilizado para los procesadores y derivado ya que estos están comunicados en lenguaje máquina de 1 y 0, que están representadas por las instrucciones, los registros del procesador, las posiciones de memoria y otras características del lenguaje. Un lenguaje ensamblador es por lo tanto, específico de cierta arquitectura de computador física (o virtual).

2.5. Hardware Libre

Se llama "hardware libre" o "electrónica libre" aquellos dispositivos de hardware donde las especificaciones y diagramas esquemáticos son de acceso público, ya sea bajo algún concepto de pago o de forma gratuita. La filosofía del software libre es aplicable a la del hardware libre y por eso forma parte de la cultura libre. Algo que tiene en común el hardware con el software es que ambos corresponden a las partes tangibles de un sistema informático.

2.5.1. Arduino

FABRICANTE ARDUINO, Arduino, disponible en https://www.arduino.cc/, blog Arduino, consultada 15 de noviembre del 2015, se define como una plataforma de hardware libre, basada en una placa con

un microcontrolador y un entorno de desarrollo, diseñada para facilitar el uso de la electrónica en proyectos multidisciplinares.

Existen en la actualidad diversas versiones de Arduino IDE, entre las cuales tenemos el Atmega168, Atmega328, Atmega1280 y Atmega8 por su sencillez y bajo coste que permiten el desarrollo de múltiples diseños.

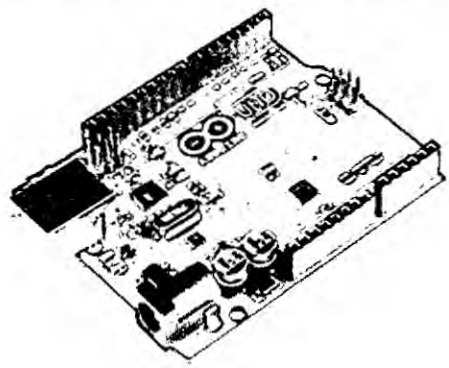
El software de desarrollo de Arduino para la programación en el microcontrolador consiste en un entorno de desarrollo que implementa el lenguaje de programación Processing/Wiring,

- Processing es un lenguaje de programación y entorno de desarrollo integrado de código abierto basado en Java
- Wiring instalación de módulos adicionales.

2.5.1.1. Módulo Arduino Uno o IDE

FABRICANTE ARDUINO, Arduino, disponible en https://www.arduino.cc/, blog Arduino, indica que es una placa electrónica basada en el ATmega328P. Cuenta con 14 pines digitales de entrada / salida (de los cuales 6 se podrán utilizar como salidas como modulación por ancho de pulso PWM) , 6 entradas analógicas, un cristal de cuarzo de 16 MHz , una conexión USB , un conector de alimentación, una cabecera Programación Serial En Circuito ICSP y un botón de reinicio . Contiene todo lo necesario para apoyar el microcontrolador; basta con conectarlo a un ordenador con un cable USB o la corriente con un adaptador de CA a CC o una batería. (Véase la figura 2.7 pagina 42)

Figura N° 2.7 Placa Arduino uno



Fuente: Fabricante Arduino disponible en https://www.arduino.cc/

La característica fundamental de este Arduino es de anexar módulos para mejorar el rendimiento del diseño, estos módulos son de trabajo específicos que traen su configuración adicional (librerías) para hacerlo funcionar, como por ejemplo un display, el cual nos mostrara los datos obtenidos, cualquier información que necesitamos.

a. Características

Estas características son dadas por el fabricante.

Tabla 2.1 Características de Arduino Uno

Microcontroller	ATmega328P
Operating Voltage	5V
Input Voltage (recommended)	7-12V
Input Voltage (limit)	6-20V
Digital I/O Pins	14 (of which 6 provide PWM output)
PWM Digital I/O Pins	6

Analog Input Pins	6
DC Current per I/O Pin	20 mA
DC Current for 3.3V Pin	50 mA
Flash Memory	32KB (ATmega328P) of which 0.5 KB used by bootloader
SRAM	2 KB (ATmega328P)
EEPROM	1 KB (ATmega328P)
Clock Speed	16 MHz
Length	68.6 mm
Width	53.4 mm
Weight	25 g

Fuente: Fabricante Arduino disponible en https://www.arduino.cc/

b. Software Arduino

FABRICANTE ARDUINO, Arduino, disponible en https://www.arduino.cc/en/Main/Software, blog Arduino, nos indica que el, Arduino Software (IDE) es de software abierto y hace que sea fácil de escribir código y subirlo al sistema. Se ejecuta en Windows, Mac OS X y Linux. El entorno está escrito en Java y basadas en el procesamiento y otro software de código abierto.

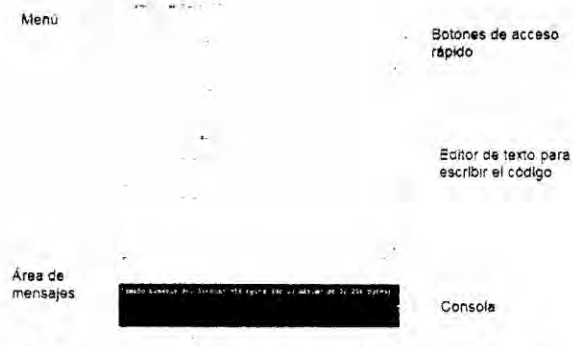
Se puede realizar la descarga del software en el siguiente link: https://www.arduino.cc/en/Main/Software.

Figura N° 2.8 Entorno desarrollo Arduino

Fuente: Software Arduino

El entorno de desarrollo Arduino está constituido por un área edición de texto donde se desarrollara la programación, una área de mensajes, una consola de texto, una barra de menú comunes para facilitar el uso, donde dicha consola es la encarga de cargar los datos necesarios para el diseño entre el hardware y el software Arduino.

Figura N° 2.9 Partes del entorno desarrollo



Fuente: Robotica al Descubierto disponible en

http://solorobotica.blogspot.pe/2012_07_01_archive.html

2.5.1. El lenguaje de programación

2.5.1.1. Funciones

Una función es un bloque de código identificado por un nombre se ejecuta cuando es llamado. La declaración de una función incluye en primer lugar el tipo de datos que devuelve la función. Después del tipo de datos se especifica el nombre de la función.

```
Tipo <nombre de la función> (variables a enviar a la función) {

        Cuerpo de las funciones donde se procesa el código

        Sentencia 1;

        Sentencia 2;

return variable;          // devuelve el valor final de la variable

}

}
```

2.5.2.2. Variables

Una variable debe ser declarada y opcionalmente asignada a un determinado valor. En la declaración de la variable se indica el tipo de datos que almacenará (int, float, long).

```
Tipo <nombre de la Variable>;
```

Una variable puede ser declarada en el inicio del programa antes de setup() a este tipo de variables se le denomina variables globales ya que puede

ser utilizada en cualquier parte del programa, una variable que está declarada dentro de una función solo tendrá el alcance de dicha función a este tipo de variables se le denomina variables locales

Arduino permite manejar los siguientes tipos de variables:

- Byte. Almacena un valor numérico de 8 bits. Tienen un rango de 0-255.
- Int. Almacena un valor entero de 16 bits con un rango de 32,767 a -32,768.
- Long. Valor entero almacenado en 32 bits con un rango de 2,147,483,647 a -2,147,483,648.
- Float. Tipo coma flotante almacenado en 32 bits con un rango de 3.4028235E+38 a -3.4028235E+38.
- Arrays Se trata de una colección de valores que pueden ser accedidos con un número de índice (el primer valor del índice es 0).

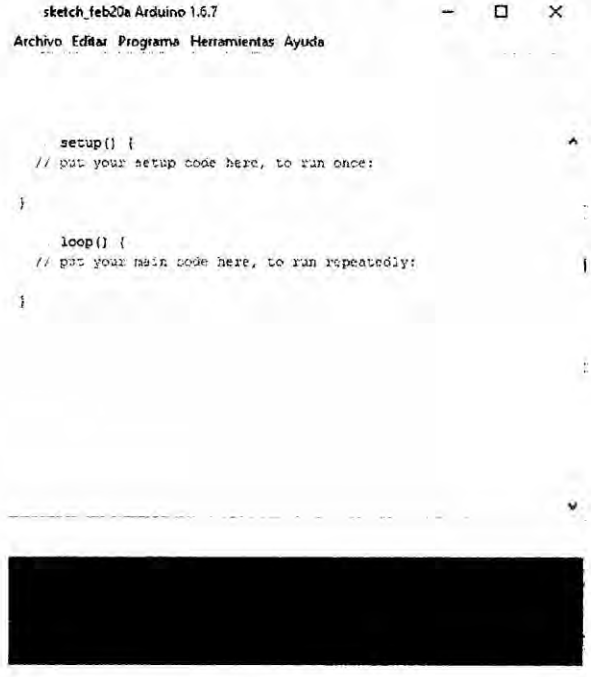
2.5.2.3. Programación en Arduino

El programa se implementará haciendo uso del entorno de programación propio de Arduino y se transferirá empleando un cable USB. Si bien en el caso de la placa USB no es preciso utilizar una fuente de alimentación externa, ya que el propio cable USB la proporciona, para la realización de algunos de los experimentos prácticos sí que será necesario disponer de una fuente de alimentación externa ya que la alimentación proporcionada por el USB puede no ser suficiente. El voltaje de la fuente puede estar entre 6 y 25 Voltios.

Cuando es iniciado el software de desarrollo Arduino nos presenta dos estructuras. (Véase figura 2.10)

- void setup(), se utiliza cuando se inicializa el programa, se utiliza principalmente para iniciar variables, establecer los pines de trabajo e iniciar las lecturas iniciales, esto es ejecutado después de conectar la fuente de alimentación y pulsar el botón reset de la placa Arduino.
- Void loop(), se la parte de código la cual va estar ejecutando continuamente hasta encontrar alguna interrupción en su código, esto permite realizar detectar cambios en los pines de entrada y poder ejecutar algún código para sus pines de salida.

Figura N° 2.10 Estructura del programa



Fuente Software Arduino

a. Entrada/Salida Digitales

Función pinMode()

Debe configurarse en la function setup(), esto indica como un pin dado tiene que comportarse como INPUT/OUTPUT. Ej. pinMode (pin, OUTPUT); configura el pin número 'pin' como de salida se debe declarar el pin como variable y pasarlo a la función, o en se puede asignar directamente la variable, es de preferencia declarar los pines de entrada para llevar un orden a la hora de trabajar con el hardware. (Véase figura 2.11)

Figura N° 2.11 PinMode

Fuente Software Arduino

Función digitalRead()

Lee el valor desde un pin digital específico. Devuelve un valor HIGH o LOW. El pin puede ser especificado con una variable o una constante (0-13). Ej. pin11 = digitalRead(11), (véase figura 2.12 página 49).

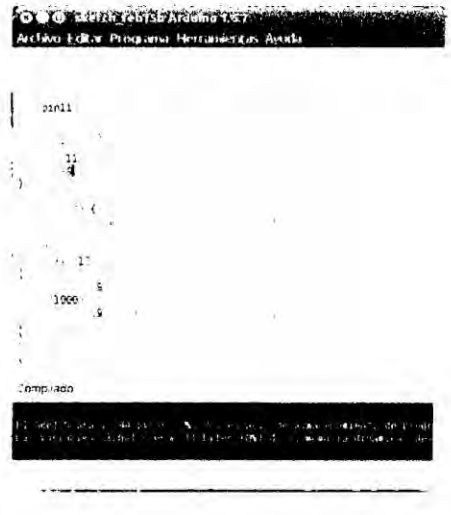
Figura N° 2.12 DigitalRead

Fuente Software Arduino

Función digitalWrite()

Introduce un nivel alto (HIGH) o bajo (LOW) en el pin digital especificado. De nuevo, el pin puede ser especificado con una variable o una constante 0-13. Ejemplo digitalWrite (pin, HIGH). (Véase figura 2.13)

Figura N° 2.13 DigitalWrite

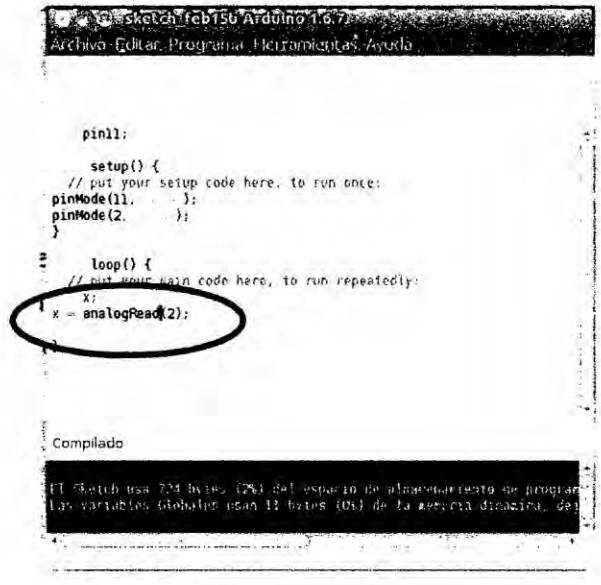


Fuente Software Arduino

Función analogRead(pin)

Lee el valor desde el pin analógico especificado con una resolución de 10 bits. Esta función solo funciona en los pines analógicos (0-5). El valor resultante es un entero de 0 a 1023. Los pines analógicos, a diferencia de los digitales no necesitan declararse previamente como INPUT o OUTPUT. (véase figura 2.14)

Figura N° 2.14 analogRead



Fuente: Software Arduino

Función analogWrite()

Escribe un valor pseudo-analógico usando modulación por ancho de pulso (PWM) en un pin de salida marcado como PWM. Ejemplo

analogWrite(pin, v); // escribe 'v' en el 'pin' analógico.

Puede especificarse un valor de 0 - 255. Un valor 0 genera 0 V en el pin especificado y 255 genera 5 V. Para valores de 0 a 255, el pin alterna rápidamente entre 0 V y 5 V, cuanto mayor sea el valor, más a menudo el pin se encuentra en HIGH (5 V). Por ejemplo, un valor de 64 será 0 V tres cuartas partes del tiempo y 5 V una cuarta parte. Un valor de 128 será 0 V la mitad del tiempo y 5 V la otra mitad. Un valor de 192 será 0 V una cuarta parte del tiempo y 5 V tres cuartas partes. (Véase figura 2.15)

Figura N° 2.15 analogWrite

Fuente: Software Arduino

b. Puerto Serie

Serial.begin(rate)

Abre un Puerto serie y especifica la velocidad de transmisión. La velocidad de comunicación con el ordenador es de 9600 aunque depende del dispositivo a conectar. (Véase figura 2.16 página 52)

Serial.println(data)

Imprime datos al puerto serie seguido por un retorno de línea automático.

Figura N° 2.16 serial.begin

Fuente: Software Arduino

Serial.print()

Imprime datos al puerto serie sin el salto de línea al final. Este comando puede emplearse para realizar la depuración de programas, activando el "Serial Monitor" se puede observar el contenido del puerto serie. (Véase figura 2.17 página 53)

Figura N° 2.17 serial.print

Fuente: Software Arduino

Serial.read()

Lee o captura un byte (un carácter) desde el puerto serie. Devuelve -1 si no hay ningún carácter en el puerto serie. (Véase figura 2.18)

Figura N° 2.18 serial.read

Fuente: Software Arduino

Serial.available()

Devuelve el número de caracteres disponibles para leer desde el puerto serie. (Véase figura 2.19)

Figura N° 2.19 serial.avaliable

Fuente: Software Arduino

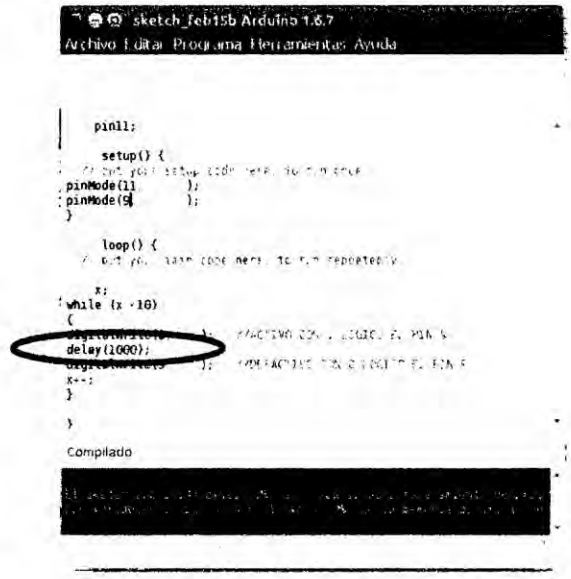
c. Función de tiempo

Delay(ms)

Realiza una pausa en el programa esta expresada en milisegundos, tiene como máximo 1000 y como mínimo 10. (Véase figura 2.20 página 55)

Son rutinas que prediseñadas para diferente hardware, esto permite ampliar y mejorar el uso del Arduino extendiendo a más dispositivos.

Figura N° 2.20 delay

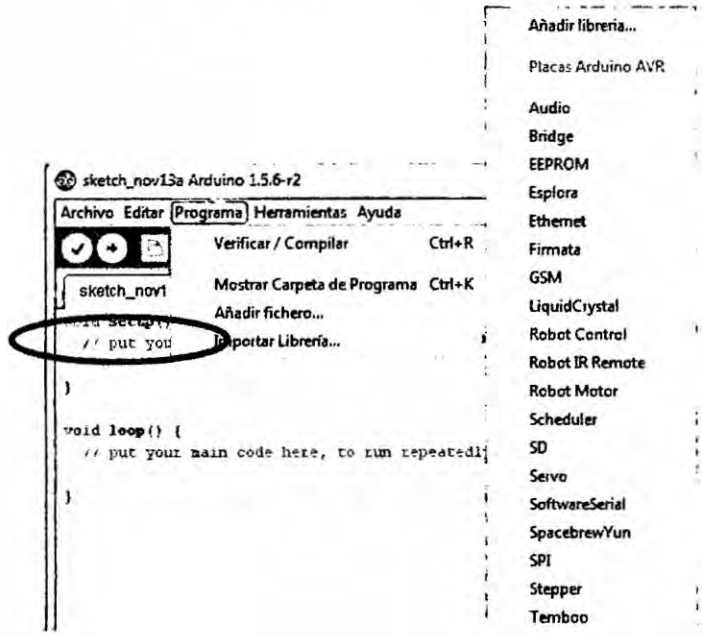


Fuente: Software Arduino

Fabricante Arduino lo define que las librerías estándar nos ayudan a manejar ciertos componentes sin necesidad de entrar en el detalle fino (y molesto) de tener que conocer cómo funciona cada una de las señales de ese componente o las características específicas del bus de control con el que se maneja.

Pero también cuantas más librerías instalamos más complicado se hace mantenerlas y ordenarlas, hasta que al fin, inevitablemente tendremos que borrar alguna para poder simplemente ver lo que tenemos instalado en un momento dado. Teniendo en cuenta que la librería ocupa memoria de nuestro microcontrolador es decir a más librerías menos memoria para programar. (Véase figura 2.21 página 56)

Figura N° 2.21 Librerías



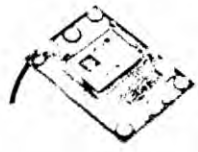
Fuente: Software Arduino

2.5.2. Módulo Arduino GPS

Es un circuito electrónico que nos proporcionara la información de GPS, latitud, longitud, altura, velocidad entre otros. La cual es enviada en tramas que debemos interpretar para obtener la información que necesitamos.

Las tramas viene codificada, esto puede varias dependiendo del fabricante y del modelo a utilizar. (Véase Figura 2.23 página 57)

Figura N° 2.22 Modulo GPS Modelo GY-GPS6MV2



Fuente: ElectroTec Perú disponible en http://electrotec.pe/blog/GPS

Figura N° 2.23 Tramas del GPS

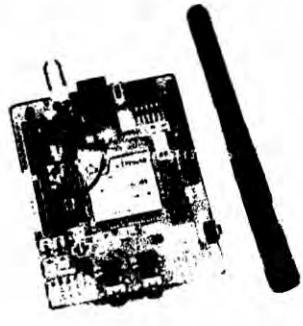
```
GPGGA -> Posicionamiento Global de Datos Fijos del Sistema
GPVTG -> Velocidad respecto al suelo
GPGSA -> GPS DOP (calidad de la señal) y satélites activos
GPGSV -> Información de cada satélite
GPGLL -> La posición geográfica, latitud / longitud
```

Fuente ElectroTec Perú disponible en http://electrotec.pe/blog/GPS

2.5.2. Módulo Arduino GPRS/GSM/4GLte

Este módulo está basado para trabajar con GSM/GPRS/Lte, con el chip GSM/GPRS SIM900 Quad-band. Se controla a través de comandos AT, es compatible con el módulo Arduino IDE.

Figura N° 2.24 Modulo Sim900



Fuente: Arduino disponible en

https://forum.arduino.cc/index.php?topic=296244.0

Debemos tener siempre en cuenta para la configuración del módulo lo siguiente.

- Utilice mayúsculas para los comandos AT.
- Enviar CR (retorno de carro) y LF (salto de línea) después de que el comando AT.
- Coloque los puentes de comunicación en serie en la posición correcta.
- Utilice una fuente de alimentación externa y colocar los puentes de potencia en la posición correcta. Si el Modulo se alimenta desde el Arduino, el puente de alimentación debe estar en la posición de Arduino 5V. Si el Modulo se alimenta desde la entrada Vin (en el módulo), el puente de alimentación debe estar en la posición Vext.

Tabla 2. 2 Comandos AT de comprobación

Mando	Respuesta	Descripción
A	Ok	Si obtiene OK, la comunicación con el módulo está funcionando
AT + CPIN = "****"	Ok	Si la tarjeta SIM está bloqueada con el PIN (**** es el número de identificación personal)
AT + COPS?		información del operador

Fuente Electrónica Estudio disponible en

http://www.electronicaestudio.com/docs/ISTD-034.pdf

Para interconectar el módulo Arduino Uno con el módulo sim900, lo primero que vamos a hacer con el módulo es para conectar el módulo a un PC directamente (utilizando un Arduino como puerta de enlace) y comprobar la base de los comandos AT. En este caso, los puentes de comunicación en serie tienen que ser fijado en la posición de la puerta de enlace USB.

- Retirar el microcontrolador del módulo Arduino Uno.
- Interconectar ambos módulos.
- A continuación, conecte el cable USB y la tarjeta SIM.
- Si se utiliza el monitor serie Arduino Uno para enviar comandos AT, a que asegúrese de que está enviando CR (retorno de carro) y LF (Line Feed).
- Ajuste la velocidad de transmisión a 115200 bps y abrir el puerto serie, a continuación.
- pulse el botón de encendido durante dos segundos.
- A continuación, si escribe AT obtendrá OK, esto significa que la comunicación con el módulo está trabajando en forma correcta.

2.6. Base de Datos

HUITRON HERNANDEZ, Base de Datos, disponible en http://docencia.fca.unam.mx/CAACS/bases-datos.php, articulo web, consultada el día 20 de diciembre del 2015, Una base de datos es un "almacén" que nos permite guardar grandes cantidades de información de forma organizada para que luego podamos encontrar y utilizar fácilmente.

Actualmente, y debido al desarrollo tecnológico de campos como la informática y la electrónica, la mayoría de las bases de datos están en formato digital (electrónico), y por ende se ha desarrollado y se ofrece un amplio rango de soluciones al problema del almacenamiento de datos.

Con lo cual lo concluimos que una base de datos o banco de datos es un conjunto de datos pertenecientes a un mismo contexto y almacenados sistemáticamente para su posterior uso.

Existen programas denominados sistemas gestores de bases de datos, abreviado DBMS, que permiten almacenar y posteriormente acceder a los datos de forma rápida y estructurada. Las propiedades de estos DBMS, así como su utilización y administración, se estudian dentro del ámbito de la informática.

Podemos mencionar algunas características que tiene el uso de una base de datos

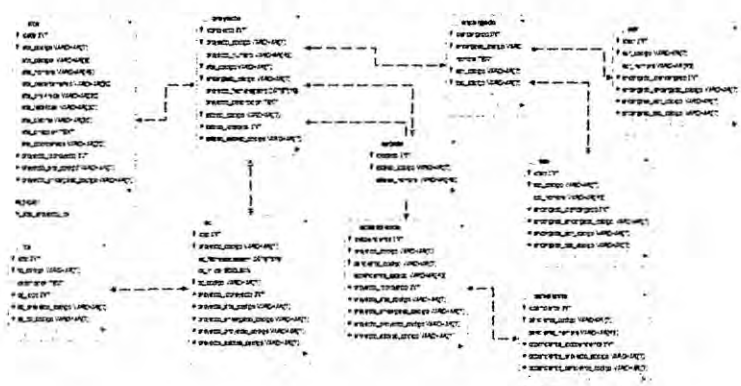
- Uso de reglas lógicas para expresar las consultas.
- Permite responder consultas recursivas.
- Redundancia mínima.
- Acceso concurrente por parte de múltiples usuarios.
- Integridad de los datos.
- Seguridad de acceso y auditoría.
- Respaldo y recuperación.
- Acceso a través de lenguajes de programación estándar.

2.6.1. Base de Datos MySQL

MySQL es un gestor de base de datos del tipo relacional, multihilo y multiusuario.

- Relacional, se refiere Permite establecer interconexiones o relaciones entre los datos (que están guardados en tablas), y a través de dichas conexiones relacionar los datos de ambas tablas, se le conoce como modelación de una base de datos.

Figura N° 2.25 Modelación de Base de Datos

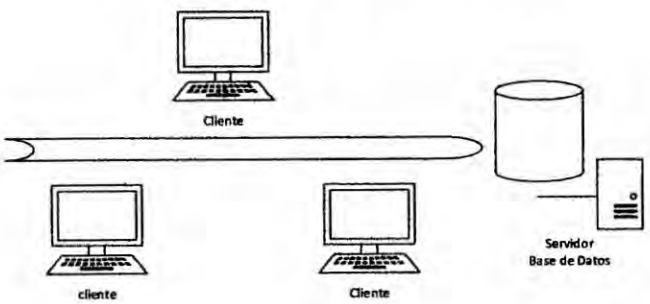


Fuente Software Workbeanch

- Multihilo, la base de datos utiliza un elemento finito de trabajo, (el hilo) para una variedad de operaciones (instrucciones de usuarios, bloqueos de datos, E/S de datos, administración del caché, etc.) en vez de utilizar aplicaciones especializadas para cada tarea.

- multiusuario, debe ser accesible por todos los ordenadores a través de la red, a estos ordenadores los llamaremos clientes.

Figura N° 2.26 Tipo Multiusuario



Fuente: Auditoria Propia

La base de datos tienes sus sentencias de consultas, están representadas con el formato T-Sql.

2.6.1.1. Tipos de sentencias en MySql

a. Por manipulación de datos

Las cuales nos permiten realizar consultas, agregar, eliminar y actualizar registros en la base de datos.

SELECT se utiliza cuando quieres seleccionar los datos.

INSERT se utiliza cuando quieres añadir nuevos datos.

UPDATE se utiliza cuando quieres actualizar datos existentes.

DELETE se utiliza cuando quieres eliminar datos.

b. Por Definición de datos

Aquí nos permitirá crear la base de datos con sus tablas de contenido,

CREATE DATABASE se utiliza para crear una nueva base de datos.

DROP DATABASE se utiliza para eliminar completamente una base de dato. CREATE TABLE se utiliza para crear una nueva tabla.

ALTER TABLE se utiliza para modificar una tabla ya existente.

DROP TABLE se utiliza para eliminar una tabla existente.

Ejemplo:

```
create database mibasededatos;

create table tabla (id int primary key, nombre varchar(20) );

insert into tabla value (1, registro1);

select id, nombre from tabla;

select id, nombre from tabla where id=1;

update tabla set nombre = registro2 where id=1;

delete from tabla where id =1;

drop database mibasededatos;
```

c. Comandos Transaccionales y bloqueo

Por defecto, MySQL se ejecuta con el modo autocommit activado. Esto significa que en cuanto ejecute un comando que actualice (modifique) una tabla, MySQL almacena la actualización en disco.

Sintaxis de START TRANSACTION, inicia los cambios en disco

Sintaxis de COMMIT almacenar los cambios en disco

Sintaxis de ROLLBACK ignora los cambios hechos desde el comienzo de la transacción.

Sintaxis de LOCK TABLES y UNLOCK TABLES bloqueo y desbloqueo de las tablas

2.7. Aplicación Web

En la ingeniería de software se denomina aplicación web a aquellas herramientas que los usuarios pueden utilizar accediendo a un servidor web a través de Internet o de una intranet mediante un navegador. En otras palabras, es una aplicación software que se codifica en un lenguaje soportado por los navegadores web en la que se confía la ejecución al navegador.

2.7.1. Estructura De Las Aplicaciones Web

En la actualidad existe muchas variantes posibles, para una aplicación web, una aplicaciones web está diseñada o estructurada en tres capas (negocios, datos y presentación).

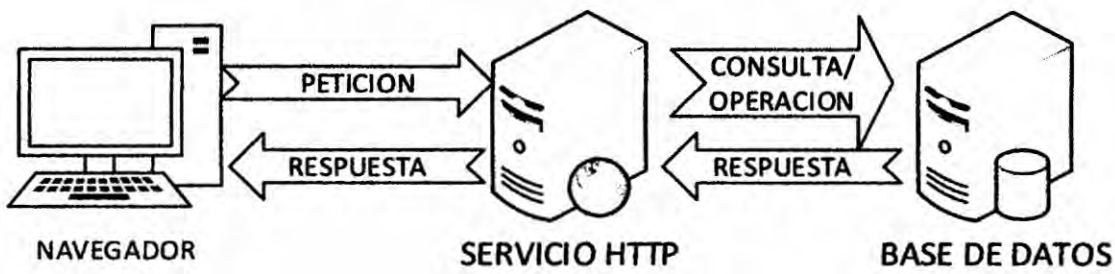
- Capa de Presentación: La capa de presentación utiliza el navegador web (Internet Explorer, google Chrome, Mozilla, Opera, etc.) para interface entre el usuario y el sistema. basadas en un motor capaz de usar alguna tecnología web dinámica (PHP, Java Servlets o ASP, ASP.NET, CGI, etc.) esta capa debe ser de fácil uso e intuitiva para el usuario ya que aquí realiza ingreso, consulta, salida, actualización de la información.
- Capa de negocios: Es donde se establece las reglas a ejecutar y estas deben cumplirse, se reciben las peticiones del usuario y se envían las respuestas tras el proceso. Esta capa es la intermediaria con la capa de presentación, para recibir las solicitudes y presentar los resultados, y con la capa de datos, para solicitar al gestor de base de datos la información a procesar esta puede ser almacenamiento o solo de consulta.
- Capa de Datos, Es donde se encuentran los datos, se encarga de procesar al mismo. Está formada por uno o más gestores de bases de datos que realizan todo el almacenamiento de datos, reciben

solicitudes de almacenamiento o recuperación de información desde la capa de negocio. (Véase figura 2.27).

2.7.1. Ventajas

Las aplicaciones Web para Internet e Intranet presentan una serie de ventajas y beneficios con respecto al software de escritorio, con lo cual se logra aprovechar y acoplar los recursos de una empresa de una forma mucho más práctica que el software tradicional.

Figura N° 2.27 Estructura de un sistema web



Fuente: Auditoria Propia

Entre los beneficios que las aplicaciones desarrolladas para la web tienen respecto a las aplicaciones de escritorio se encuentran:

- Ahorra tiempo: Se pueden realizar tareas sencillas sin necesidad de descargar ni instalar ningún programa.
- Compatibilidad: Basta tener un navegador actualizado para poder ser utilizado.

- Consumo de recursos bajo: Dado que toda (o gran parte) de la aplicación no se encuentra en nuestro ordenador, no ocupan espacio en nuestro disco duro.
- Multiplataforma: Se pueden usar desde cualquier sistema operativo porque sólo es necesario tener un navegador.
- Disponibilidad: Suele ser alta porque el servicio se ofrece desde múltiples localizaciones para asegurar la continuidad del mismo.
- Flexibilidad: Los navegadores ofrecen cada vez más y mejores funcionalidades para crear aplicaciones web ricas, permitiendo actualizaciones inmediatas.

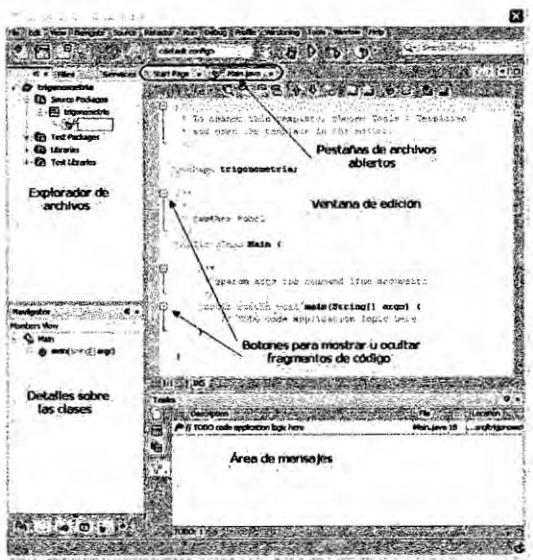
2.7.2. Netbeans

NETBEANS, **Netbeans**, disponible en https://netbeans.org/, pagina web, consultada el 03 de febrero del 2016, se definen como un entorno de desarrollo integrado libre, hecho principalmente para el lenguaje de programación Java. Existe además un número importante de módulos para extenderlo. NetBeans IDE es un producto libre y gratuito sin restricciones de uso.

El entorno de desarrollo está compuesto por la ventana de edición (véase figura 2.28 página 68), aquí es donde realizamos la programación, explorador de archivos donde podemos visualizar nuestros proyectos, ventana de detalles donde visualizares la estructura de las clases y el área

de mensajes, donde nos informa sobre cualquier error o advertencia en nuestro programa.

Figura N° 2.28 Entorno Netbeans



Fuente: Observatorio Tecnológico disponible en http://recursostic.educacion.es/observatorio/web/ca/software/programacion/911-monografico-java?start=3

2.7.2.1. Estructura del Lenguaje Programación Java

GARCIA BELTRAN, **Programación Java**, Madrid, Consorcio OpenCourseWare, segunda edición, 2009. Nos explica que, existen dos elementos típicos del código fuente: los comentarios y los identificadores. La estructura de un programa de Java es similar a la de un programa de C/C++. Por su diseño, permite a los programadores de cualquier otro lenguaje leer código en Java sin mucha dificultad. Donde concluimos que

Java emplea siempre la Programación Orientada a Objetos por lo que todo el código se incluye dentro de las clases. Las clases son combinaciones de datos (constantes y variables) y rutinas (métodos).

La estructura de un programa simple en Java es la siguiente:

```
public class ClasePrincipal {

public static void main(Strinsg[] args)

    {

    sentencia_1;  //comentario

    sentencia_2;

    sentencia3:  //comentario

    // ... sentencia_N;

    }

}
```

a. Comentarios

Los comentarios se emplean para dejar notas o apuntes dentro del código para señalar alguna observación durante el desarrollo del software.

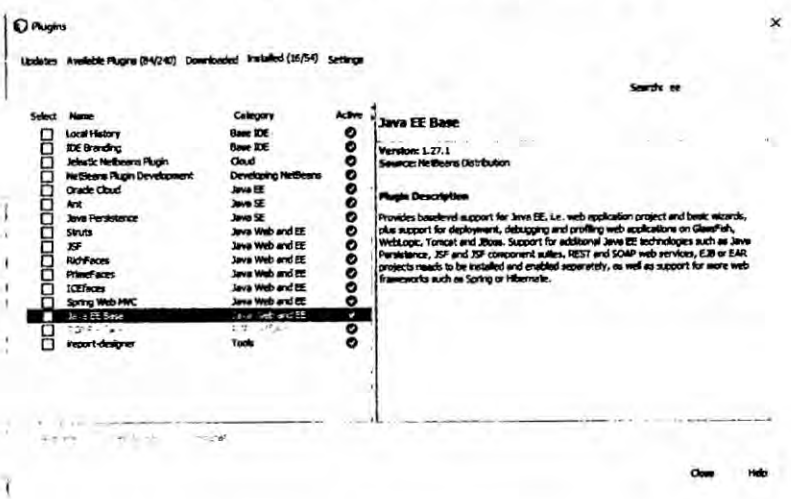
b. Identificadores

Los identificadores son nombres que se les asignan a variables, métodos, clases en el código fuente de un programa. Los identificadores sólo existen en el código del programa fuente y no en el programa objeto.

2.7.2.2. Creación de un Proyecto Web

Para la creación web vamos tenemos que instalar la librerías Java EE (entreprice edition), este plugin se agrega en >tool>plugins (véase figura 2.29)

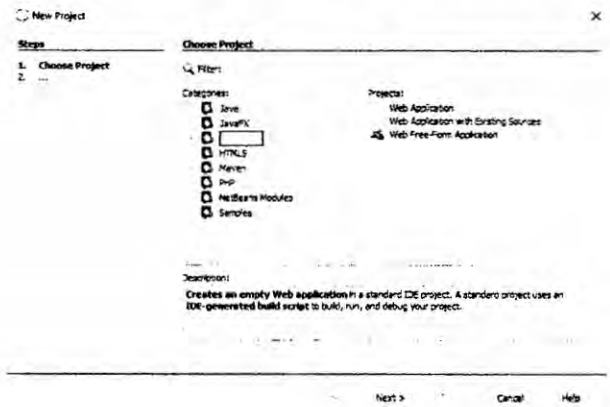
Figura N° 2.29 Plugins en Netbeans



Fuente: Software Netbeans

Ahora crearemos un nuevo proyecto Java Web en ">File>New Proyect" nos aparecerá un cuadro de dialogo donde escogeremos ">Java Web>Web Aplication". (Véase figura 2.30 página 71)

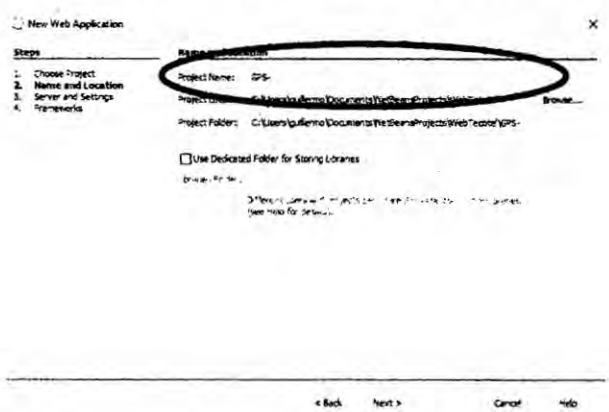
Figura N° 2.30 Ventana New Proyect



Fuente: Software Netbeans

Daremos nombre al nuevo proyecto (véase figura 2.31)

Figura N° 2.31 Nombre del proyecto

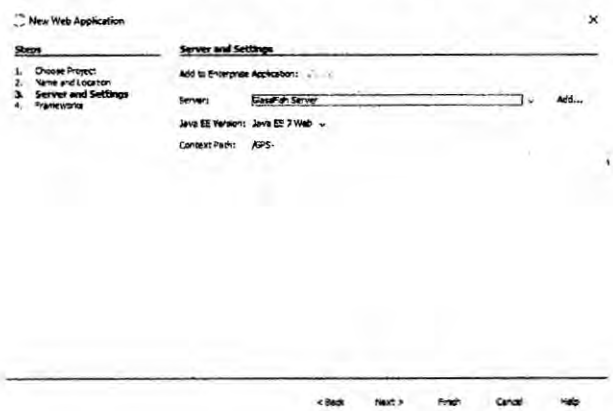


Fuente: Software Netbeans

Netbeans cuenta con un servidor incorporado para realizar las pruebas de nuestros proyectos webs, utiliza dos servidores web, el Apache Tomcat y el Glassfish, cuando hemos dado nombre a nuestro proyecto nos preguntara que tipo de servidor usaremos para publicar nuestro proyecto,

para nuestro caso usaremos el Glassfish server la versión java EE 7 web. (Véase figura 2.32)

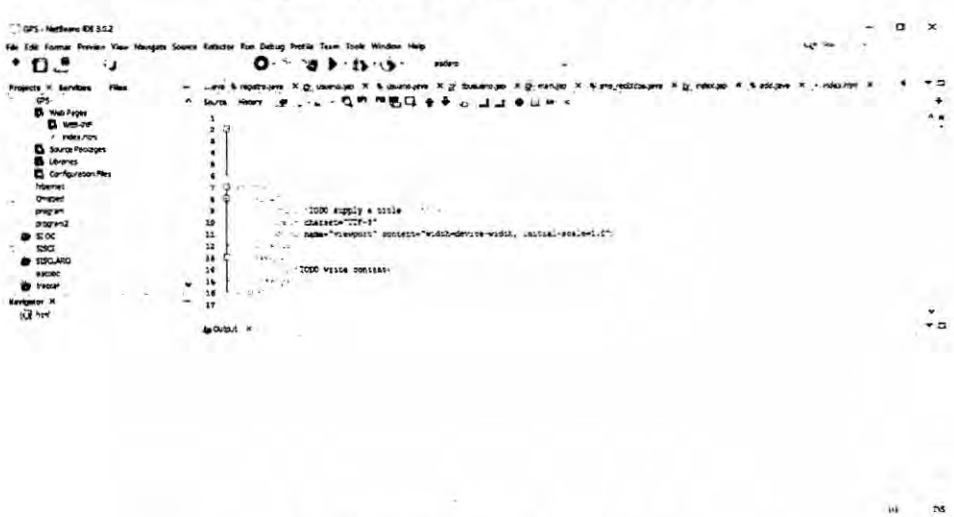
Figura N° 2.32 Escogemos El server web



Fuente: Software Netbeans

Daremos finalizar donde tendremos el proyecto en espera de la programación por parte del usuario. (Véase figura 2.33)

Figura N° 2.33 proyecto creado



Fuente: Software Netbeans

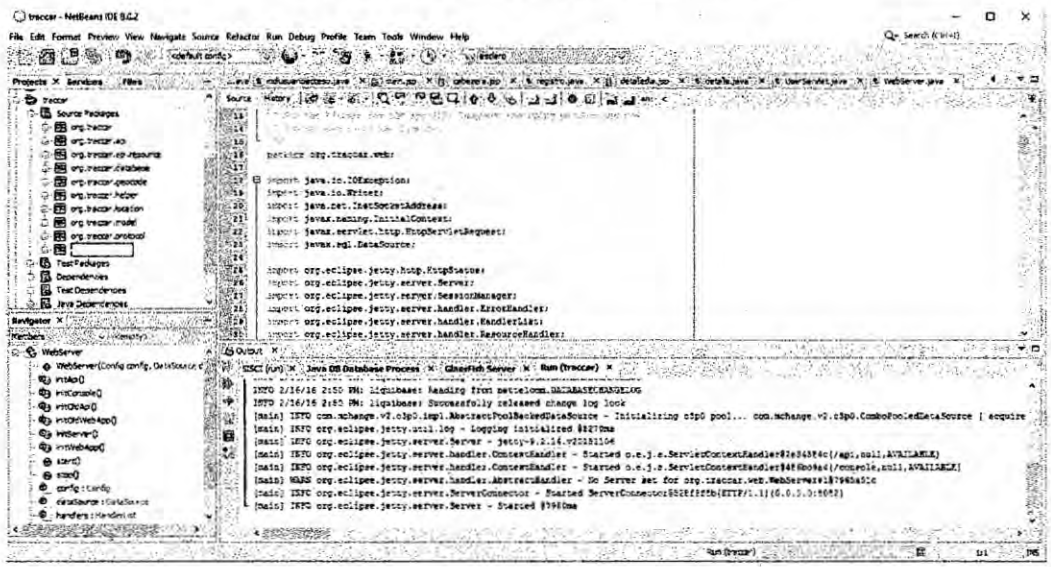
2.7.3. Traccar

TRACCAR, **Traccar**, disponible en https://www.traccar.org/, sitio web, consultado el 1 de diciembre del 2015, se define como un sistema de localización por GPS de código abierto para varios dispositivos de localización GPS. El sistema soporta más de 80 protocolos de comunicación diferentes de los vendedores populares. Incluye interfaz web para gestionar los dispositivos de seguimiento en línea. Traccar es un software libre la cual nos proporciona la herramienta para ser utilizado, modificado según el dispositivo GPS inclusive nos proporciona un software cliente para caso de móviles para realizar pruebas.

El software Traccar puede ser descargado desde https://www.traccar.org/download/ es compatible con diferentes sistemas operativos (Windows, Linux, MAC, etc.)

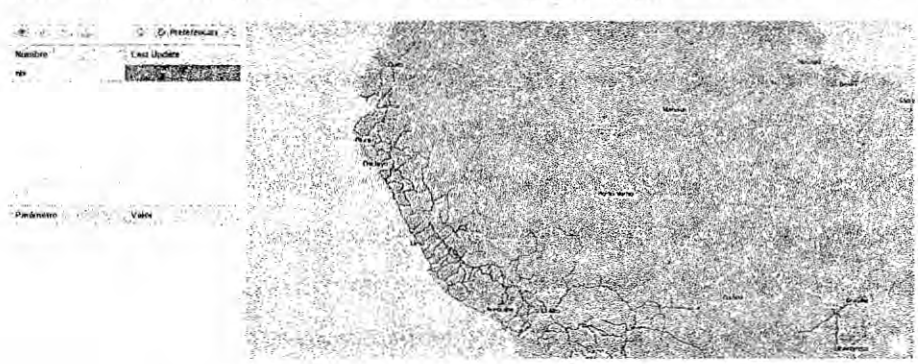
Traccar está desarrollado en java bajo, bajo web-servers en este caso las de google las cuales nos proporciona los mapas para la ubicación geográfica, mediante el uso del software Netbeans (figura 2.34 página 74) podremos visualizar y editar su programación para adecuarla al desarrollo de nuestra tesis. Este software incorpora a las base de datos MySql la cual podrá ser analizada, teniendo como resultado una página web la cual nos proporciona la información que necesitemos: http://localhost:8082 (véase figura 2.35 página 74)

Figura N° 2.34 Proyecto Traccar



Fuente: Software Netbeans

Figura N° 2.35 Proyecto Traccar Ejecutándose



Fuente: Navegador Web Internet Explorer

2.8. Sistema de Móvil de Telefonía 4G Lte

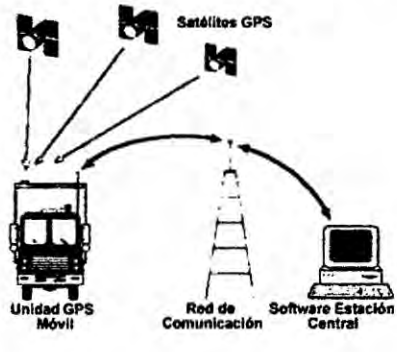
En telecomunicaciones, 4G son las siglas utilizadas para referirse a la cuarta generación de tecnologías de telefonía móvil. Es la sucesora de las tecnologías 2G y 3G.

La 4G Lte está basada completamente en el protocolo IP, siendo un sistema y una red, que se alcanza gracias a la convergencia entre las redes de cables e inalámbricas. Esta tecnología podrá ser usada por módems inalámbricos, móviles inteligentes y otros dispositivos móviles.

La principal diferencia con las generaciones predecesoras será la capacidad para proveer velocidades de acceso mayores de 100 Mbit/s en movimiento y 1 Gbit/s en reposo, manteniendo una calidad de servicio (QoS) de punta a punta de alta seguridad que permitirá ofrecer servicios de cualquier clase en cualquier momento.

Con el sistema de móvil podremos enviar la data, logrando un monitoreo on-line así mantener informado en cada momento que lo necesite el programa para así ser almacenado en la base de datos.

Figura N° 2.36 Esquema del sistema GPS y 4G Lte



Fuente ElectroTec Perú disponible en http://electrotec.pe/blog/GPS

III. VARIABLES E HIPOTESIS

3.1. VARIABLES

Tabla 3.1 Variables Dependiente

VARIABLES	DIMENSIONES	INDICADORES
Independiente: La implementación de un sistema de ruta.	Ubicación	Ubicación en coordenadas
	Fecha/Hora	horas, minuto, segundo, fecha

3.2. Operacionalidad de Variables

Tabla 3. 2 Variables Dependiente

VARIABLES	DIMENSIONES	INDICADORES
Dependiente: Control en las unidades de la empresa Nettelcom SAC	Tiempo	Diferencial de Tiempo, hora minuto segundo
	Distancia	En km
	Consumo	galones, litros, m^3
	Velocidad	k/h, m/s

3.2. Hipótesis de la Investigación

3.2.1. Hipótesis General

Existe una relación significativa entre el diseño de un Sistema de ruta con GPS/4G Lte y el control de las unidades en la empresa NETTELCOM SAC.

3.2.2. Hipótesis Específica

- Existe una relación significativa entre el diseño de un sistema de ruta con GPS/4G Lte controla el tiempo en las unidades de la empresa Nettelcom SAC.
- Existe una relación significativa entre el diseño de un sistema de ruta con GPS/4G Lte controla la distancia en las unidades de la empresa Nettelcom SAC.
- Existe una relación significativa entre el diseño de un sistema de ruta con GPS/4g Lte controla el consumo en las unidades de la empresa Nettelcom SAC.
- Existe una relación significativa entre el diseño de un sistema de ruta con GPS/4G Lte controla la velocidad en las unidades de la empresa Nettelcom SAC.

IV. METODOLOGIA

El método de análisis Documental, permitirá realizar el sustento teórico del trabajo de investigación permitiendo describir los fundamentos del trabajo y obtener información sobre modelos de autoevaluación escrita por algún profesional.

4.1. Tipo de Investigación

El diseño de la investigación es, no experimental, ya que no existe manipulación activa de alguna variable. Además, se trata de un diseño correlacional, transaccional transversal, ya que se busca establecer la relación de variables medidas en la muestra, en un único momento del tiempo, tal como lo sostiene Hernández, Fernández y Baptista (2006, p. 267) La toma de datos del GPS que son las coordenadas y Fecha/Hora en donde las variables no serán manipuladas, dicho trabajo se define como una investigación no experimental, correlacional.

4.2. Diseño de la Investigación

Para el diseño de un sistema de ruta con GPS/4G Lte para el control de las unidades de transporte de la empresa Nettelcom SAC el cual se tendrá que tomar ciertos criterios.

- Puertos de entrada y salida, sean analógicos o digitales, los cuales permitirán tomar información sobre el sistema.

- Utilización de cualquier elemento electrónico para mejorar nuestro diseño.
- Utilización de microcontroladores los cuales nos permitirá realizar procesos complejos, para esto nos ayudaremos del sistema Arduino el cual nos permite manejar la información de forma más sencilla.
- Diseño de una plataforma web el cual nos permita almacenar la información y luego procesarla.

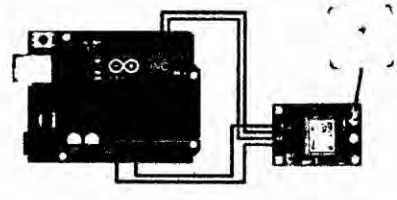
Diseño del Módulo de Detección GPS

Para el diseño de la plataforma Arduino debemos tener presente lo que deseamos que realice dicha plataforma

- Captura de señal GPS y 4G Lte
- Procesamiento de señal GPS para poder determinar la información
- Envió hacia a internet (servidor) el cual almacenara la información para ser procesada.

Teniendo en cuenta que estamos utilizando el Arduino Uno, con el modulo GPS 6MV2 tenemos la siguiente conexión. (Véase figura 4.1)

Figura 4.1 Esquemático Arduino – GPS-6MV2



Fuente: ElectroTec Perú disponible en http://electrotec.pe/blog/GPS

La forma de conexión estaría orientada en forma serial, donde tenemos un pin de Tx y un pin Rx.

El modulo GPS nos envía información cuando se enlaza con los satélites y da referencia de su ubicación.

Figura 4.2 Trama GPS

```
$GPGGA,142651.000,4138.39329,N,00053.28085,W,1,05,3.5,272.35,M,51.6,M,,*76
$GPVTG,0.0,T,,M,0.1,N,0.3,K*62
$GPGSA,A,3,10,08,07,02,04,,,,,,,,9.8,3.5,9.2*36
$GPGSV,3,1,10,13,49,047,,10,80,000,30,23,21,061,,16,07,043,*7E
$GPGSV,3,2,10,08,42,174,37,07,65,121,39,05,24,303,,02,51,277,35*7C
$GPGSV,3,3,10,04,52,213,41,120,71,269,,,,,,,,,,,,*46
$GPGLL,4138.393,N,00053.281,W,142651.000,A*29
```

Fuente: ElectroTec Perú disponible en http://electrotec.pe/blog/GPS

Aquí hay que analizar la salida real. La salida por ejemplo que vamos a trabajar con una lectura de latitud 1234.5678 y una lectura de longitud de la 12345.6789.

Si la lectura de latitud = 1234.5678 latitud real se lee como 12°34.5678'

Si la longitud de lectura = 12345,6789 longitud es 123°45.6789'

Que la parte flotante requiere un procesamiento de datos en función de lo necesita, por lo tanto convertiremos todo a minutos:

Latitud 12°34.5678 = 12*60 +34.5678 = 754.5678'

Longitud 123°*60 + 45.6789 = 7425,6789'

Esto se puede convertir en metros sabiendo que 1' ≈ 1.852 metros (van fuera del hecho de cada grado es de aproximadamente 111 kilómetros).

Con este conocimiento, podemos tomas dos latitud y longitud, dos lecturas y determinar la diferencia en metros entre las coordenadas. Esto nos servirá para formar el mapa y ver la ruta y el tiempo que le toma en llegar de un lugar a otro.

La programación a realizar en el Arduino para detectar la señal GPS es la siguiente (véase figura 4.3)

Figura 4.3 Rutina de Detección GPS

Fuente: Software Arduino

A partir de esta rutina de detección tenemos que enviarlo a nuestro servidor, recordemos que el usuario no necesita visualizar esta información solo le enviar su información de ubicación cada ciclo de tiempo.

En el siguiente programa se utiliza para realizar un test de detección de un módulo GPS con la librería TinyGPS, donde escogemos los pines de trasmisión y recepción, la velocidad de comunicación, se puede realizar la descarga de esta librería https://www.dropbox.com/s/46tydow03wqviy7/TinyGPS.zip?dl=0

```
#include <SoftwareSerial.h>

#include <TinyGPS.h>

TinyGPS gps;

SoftwareSerial ss(4, 3); //recordar que los pines a conectar son 4(rx) and 3(tx)

static void smartdelay(unsigned long ms);

static void print_float(float val, float invalid, int len, int prec);

static void print_int(unsigned long val, unsigned long invalid, int len);

static void print_date(TinyGPS &gps);

static void print_str(const char *str, int len);

void setup() {

  Serial.begin(115200);
```

```
  Serial.print("Testing TinyGPS library v. ");
 Serial.println(TinyGPS::library_version());
  Serial.println("by Mikal Hart");
  Serial.println();
  Serial.println("Sats HDOP Latitude Longitude Fix Date Time Date
Alt Course Speed Card Distance Course Card Chars Sentences
Checksum");
  Serial.println("          (deg)    (deg)     Age                      Age  (m)  --- from
GPS ---- ---- to London  --- RX  RX       Fail");
  Serial.println("----------------------------------------------------------------------
--------------------------------------------");
  ss.begin(9600);
}
void loop()
{
  float flat, flon;
  unsigned long age, date, time, chars = 0;
```

```
 unsigned short sentences = 0, failed = 0;

 static const double LONDON_LAT = 51.508131, LONDON_LON = -
0.128002;

  print_int(gps.satellites(), TinyGPS::GPS_INVALID_SATELLITES, 5);

 print_int(gps.hdop(), TinyGPS::GPS_INVALID_HDOP, 5);

 gps.f_get_position(&flat, &flon, &age);

 print_float(flat, TinyGPS::GPS_INVALID_F_ANGLE, 10, 6);

 print_float(flon, TinyGPS::GPS_INVALID_F_ANGLE, 11, 6);

 print_int(age, TinyGPS::GPS_INVALID_AGE, 5);

 print_date(gps);

 print_float(gps.f_altitude(), TinyGPS::GPS_INVALID_F_ALTITUDE, 7, 2);

 print_float(gps.f_course(), TinyGPS::GPS_INVALID_F_ANGLE, 7, 2);

 print_float(gps.f_speed_kmph(), TinyGPS::GPS_INVALID_F_SPEED, 6,
2);

 print_str(gps.f_course() == TinyGPS::GPS_INVALID_F_ANGLE ? "*** " :
TinyGPS::cardinal(gps.f_course()), 6);
```

```
 print_int(flat == TinyGPS::GPS_INVALID_F_ANGLE ? 0xFFFFFFFF :
(unsigned long)TinyGPS::distance_between(flat, flon, LONDON_LAT,
LONDON_LON) / 1000, 0xFFFFFFFF, 9);

 print_float(flat == TinyGPS::GPS_INVALID_F_ANGLE ?
TinyGPS::GPS_INVALID_F_ANGLE : TinyGPS::course_to(flat, flon,
LONDON_LAT, LONDON_LON), TinyGPS::GPS_INVALID_F_ANGLE, 7,
2);

 print_str(flat == TinyGPS::GPS_INVALID_F_ANGLE ? "*** " :
TinyGPS::cardinal(TinyGPS::course_to(flat, flon, LONDON_LAT,
LONDON_LON)), 6);

 gps.stats(&chars, &sentences, &failed);

 print_int(chars, 0xFFFFFFFF, 6);

 print_int(sentences, 0xFFFFFFFF, 10);

 print_int(failed, 0xFFFFFFFF, 9);

 Serial.println();

  smartdelay(1000);

}

static void smartdelay(unsigned long ms)
```

```
{
  unsigned long start = millis();
  do
  {
    while (ss.available())
      gps.encode(ss.read());
  } while (millis() - start < ms);
}
static void print_float(float val, float invalid, int len, int prec)
{
  if (val == invalid)
  {
    while (len-- > 1)
      Serial.print('*');
    Serial.print(' ');
  }
```

```
  else
  {
    Serial.print(val, prec);
    int vi = abs((int)val);
    int flen = prec + (val < 0.0 ? 2 : 1); // . and -
    flen += vi >= 1000 ? 4 : vi >= 100 ? 3 : vi >= 10 ? 2 : 1;
    for (int i=flen; i<len; ++i)
      Serial.print(' ');
  }
  smartdelay(0);
}
static void print_int(unsigned long val, unsigned long invalid, int len)
{
  char sz[32];
  if (val == invalid)
    strcpy(sz, "********");
```

```
  else
    sprintf(sz, "%ld", val);
  sz[len] = 0;
  for (int i=strlen(sz); i<len; ++i)
    sz[i] = ' ';
  if (len > 0)
    sz[len-1] = ' ';
  Serial.print(sz);
  smartdelay(0);
}
static void print_date(TinyGPS &gps)
{
  int year;
  byte month, day, hour, minute, second, hundredths;
  unsigned long age;
```

```
  gps.crack_datetime(&year, &month, &day, &hour, &minute, &second,
&hundredths, &age);
  if (age == TinyGPS::GPS_INVALID_AGE)
    Serial.print("********** ******** ");
  else
  {
    char sz[32];
    sprintf(sz, "%02d/%02d/%02d %02d:%02d:%02d ",
        month, day, year, hour, minute, second);
    Serial.print(sz);
  }
  print_int(age, TinyGPS::GPS_INVALID_AGE, 5);
  smartdelay(0);
}
static void print_str(const char *str, int len)
{
```

```
 int slen = strlen(str);

 for (int i=0; i<len; ++i)

  Serial.print(i<slen ? str[i] : ' ');

 smartdelay(0);

}
```

Hasta ahora hemos realizado la conexión solo Arduino Uno - GPS, pero eso nos daría la información solo en el GPS donde está instalado pero en realidad lo que necesitamos que se envié al servidor para guardar y procesar la información, dicho envió lo realizaremos con el módulo GSM/GPRS/Lte con algún chip de una operadora chip de forma que podamos enviar la información sea en mensaje de texto o a una página web determinada. Donde se encargue de almacenar la información para luego ser procesada.

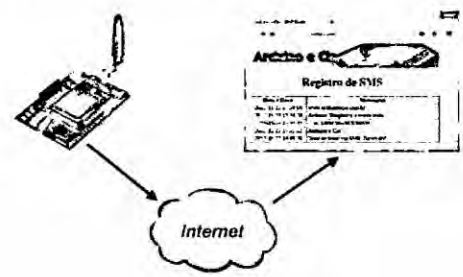
Diseño Modulo Arduino-Web-Base de Datos

Los que queremos realizar en este módulo es la configuración del módulo Arduino y pueda conectarse con la base de datos en internet, y grabar la información en la base de datos donde podremos almacenar la información del sistema para luego ser procesada.

Los requisitos para esta etapa:

- Módulo Arduino Lte/GPRS/GSM
- Instalación y configuración de un servidor MySql para almacenar la información enviada por el modulo.
- Instalación y configuración de un servidor web, que contenga una Ip estática para conectarse.
- Realizar un código capaz de realizar la gestión con la base de datos.
- Envió de la información mediante el Arduino.

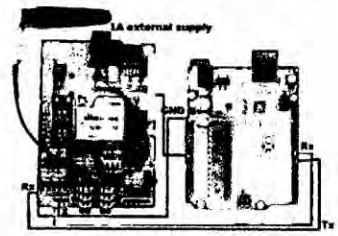
Figura 4.4 Visualización de Datos mediante una URL



Fuente: Arduino Cia disponible en http://www.arduinoecia.com.br/2015/11/acessar-internet-arduino-gsm-shield-sim-900.html

Esta función la realizaremos mediante el Arduino GSM/GPRS/Lte shield SIM900 o el Cheap Arduino to LAN/WiFi/3G/3.5G/4G connection. Es el acceso a la Internet, utilizando el Arduino para enviar información a un servidor web mediante que almacenara la información en una base de datos MySql.

Figura 4.5 Conexión Modulo GPRS/GSM/Lte



Fuente: Stack Exchange Inc. Disponible en http://arduino.stackexchange.com/questions/9483/how-to-communicate-the-arduino-board-with-sim900

Los pasos descritos implican mucho más la configuración de Java y base de datos MySQL que parte de la Arduino, debemos seguir la siguiente secuencia:

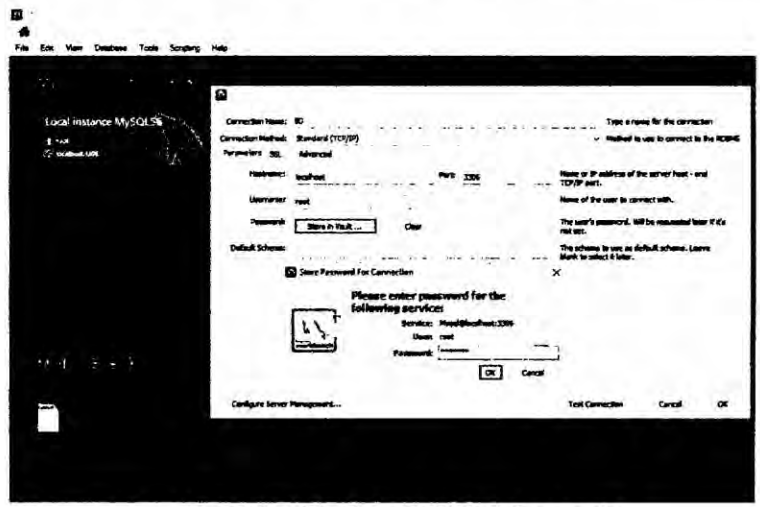
Configuración de Servidor de Base de Datos

Para el servidor de base de datos tenemos la base de datos MySql el cual nos permitirá el almacenar la información proporcionada por Arduino, para realizar esto necesitamos la creación de la tabla de recepción de mensajes de texto.

- Instalación: la instalación se puede realizar en Windows o Linux lo que tenemos tener presente es el usuario y contraseña de conexión a la base de datos.

- Configurando la base de datos: para ingresar y gestionar en la base de datos debemos realizar mediante un gestor MySql, como el workbeanch, query Mysql, PhpMyAdmin, entre otros que nos facilitara la utilización de la misma. (véase figura 4.6)

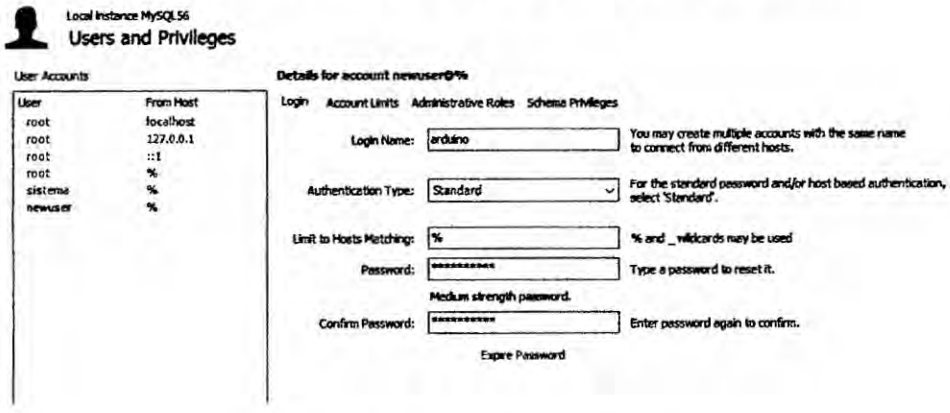
Figura 4.6 Conectado a la base de datos Con WorkBeanch



Fuente: Software Workbeanch

- Una vez validados en la base de datos crearemos un usuario y password de conexión para proteger nuestra base de dato, recordemos que el usuario root de la instalación es el administrador principal de nuestra base de datos por lo cual no es recomendable conectarnos con dicho usuario. Debemos crear un usuario y password para la base de datos a trabajar. (Véase figura 4.7 página 94).

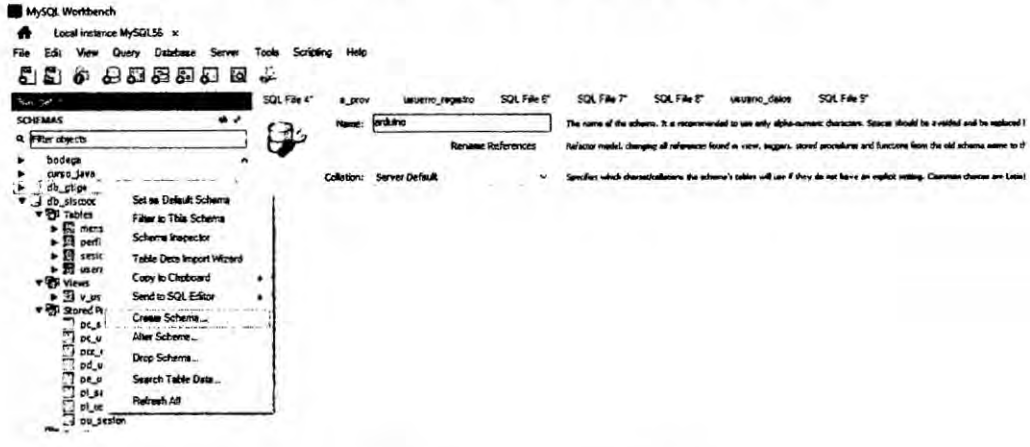
Figura 4.7 Creación de usuario para la base de datos



Fuente: Software Workbeanch

- Paremos a la creación de la base de datos, la base de datos llamaremos Arduino. (véase figura 4.8)

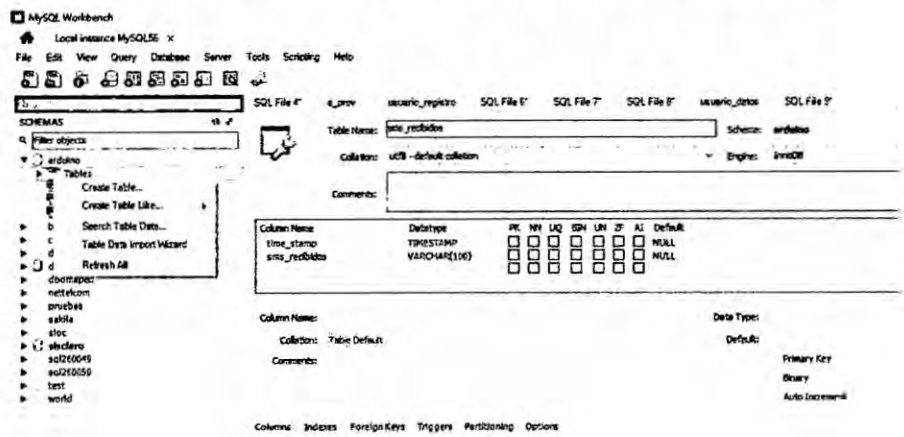
Figura 4.8 Creación de la Base de Datos



Fuente: Software Workbeanch

- creación de la tabla, la tabla a crear es sms_recibidos, con dos campos uno time_stamp y sms_recibido. (Véase figura 4.9 página 95)

Figura 4.9 Creación de tablas en la base de datos



Fuente: Software Workbeanch

Configuración de Entorno Java web

Ahora vamos a crear el proyecto para acceder y actualizar la base de datos. Mediante un entorno web para esto vamos a utilizar la Plataforma de Netbeans mediante la programación Java Web:

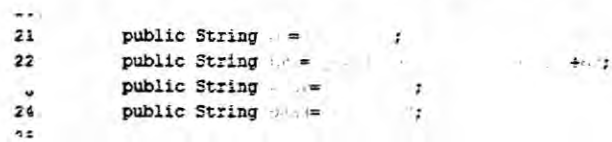
- Conectado con la base de datos, creamos el archivo conexión.java, donde se *tiene la configuración de conexión a la base de datos (usuario y contraseña)*

Figura 4.10 Función de conexión

```
67
68      public Connection conectar()
69          {
70          Connection link = null;
71          try
72          {
73              //cargamos drivers
74              Class.forName("com.mysql.jdbc.Driver");
75              //creamos un enlace hacia la base de datos
76              link = DriverManager.getConnection(this.url,this.user,this.pass);
77          }
78          catch (Exception ex)
79          {
80          JOptionPane.showMessageDialog(null, "Mensaje Error Base de Datos"+ex);
81          }
82      return link;
83      }
84  }
85
```

Fuente: Software Netbeans

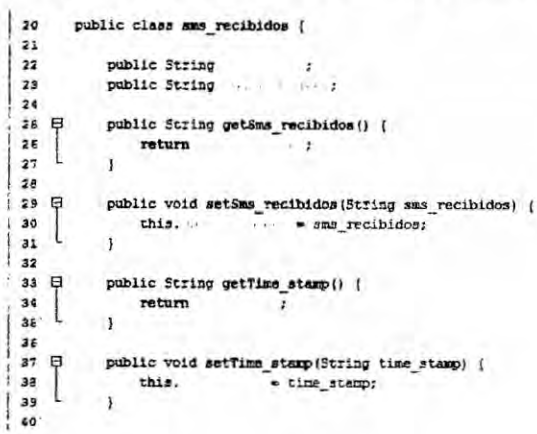
Figura 4.11 variables para la cadena de conexión



Fuente: Software Netbeans

- Creación de modelos según las tablas que tengamos, debemos definir nuestras variables de entorno.

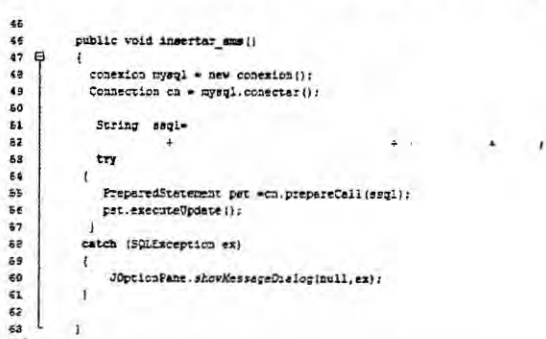
Figura 4.12 modelo de sms_recibido



Fuente: Software Netbeans

- Nuestra función de inserción de datos, es la encargada de insertar los datos en la base de datos.

Figura 4.13 sms_recibido.java función insertar_sms



Fuente: Software Netbeans

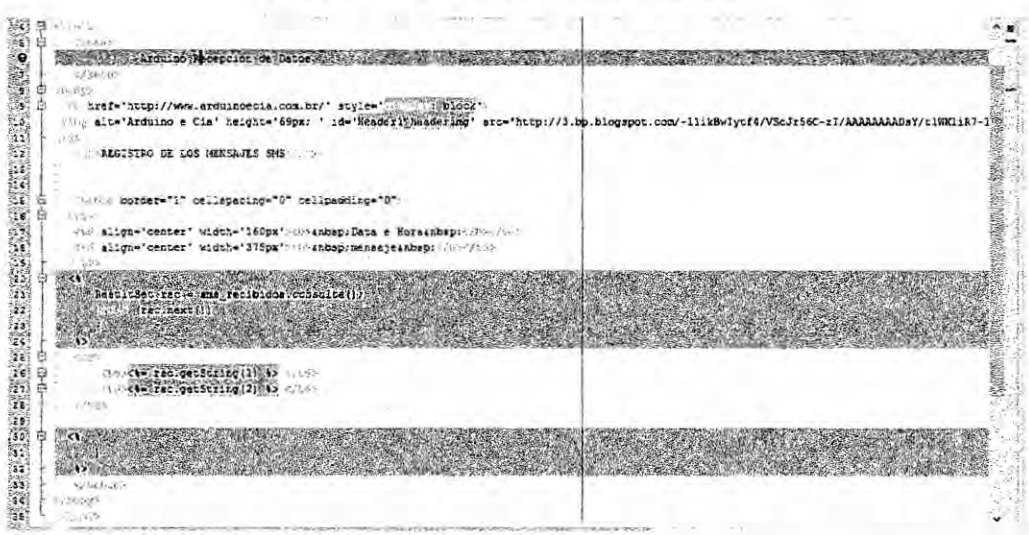
Ahora la consulta para poder visualizar los datos.

Figura 4.14 sms_recibido función para mostrar los datos recolectados

Fuente: Software Netbeans

- Crearemos la página web la cual nos mostrara la información que es esta almacenando en la base de datos. Dicha página tendrá que ser grabada con extensión jsp

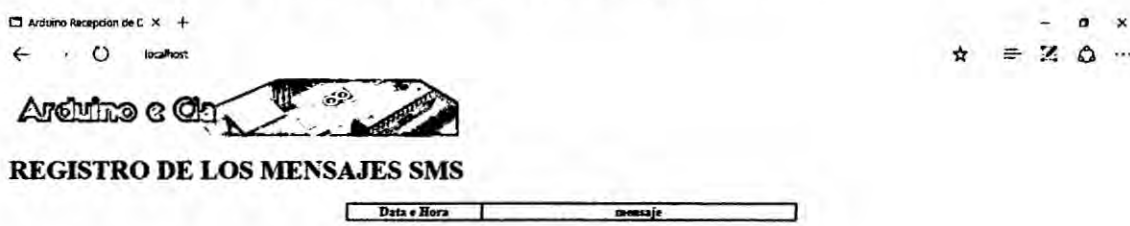
Figura 4.15 Programación del JSP



Fuente: Software Netbeans

Al ejecutar el programa tendríamos la siguiente presentación, aperturamos nuestro explorador web y en la url pondremos http://localhost:8080/GPS/ se cargara la página web donde visualizaremos la información registrada por el sistema web. (Véase figura 4.16)

Figura 4.16 Página web en ejecución



Fuente: Navegador Web Internet Explorer

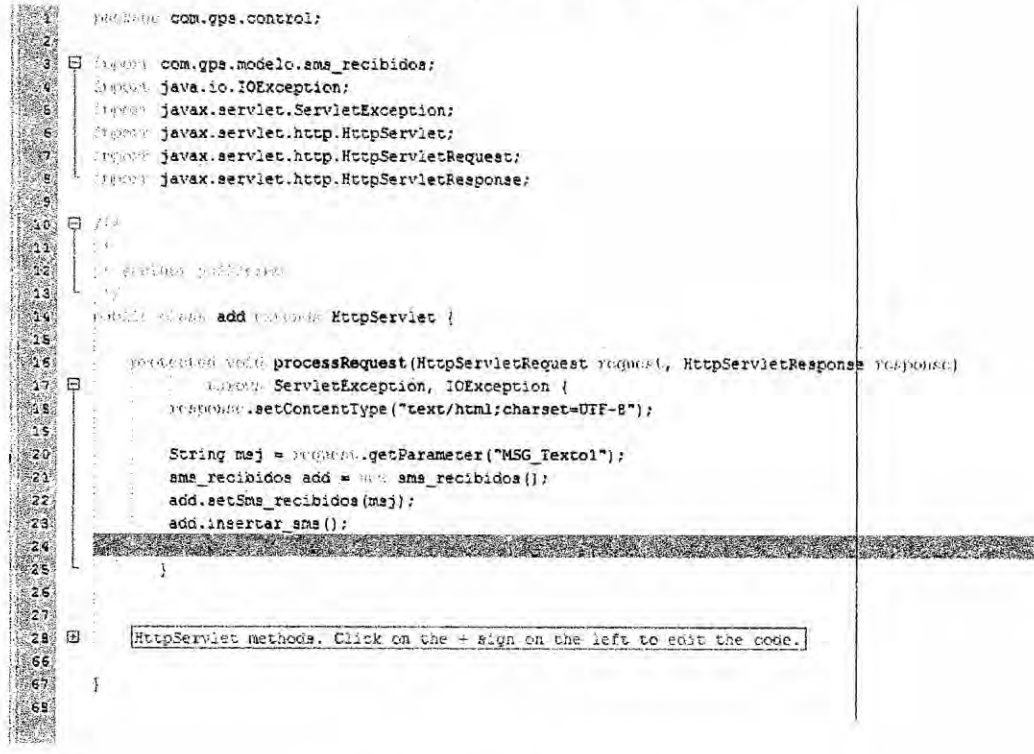
- Ahora nos queda por configurar el servelt que es el encargado de insertar los datos, para luego ser analizados. (véase Figura 4.17 pagina 99)

Quedando toda la estructura de la programación para la lectura de datos de la siguiente forma.

Figura 4.17 Árbol de archivos del proyecto GPS

Fuente: Software Netbeans

Figura 4.18 add.java programación de servelt



Fuente: Software Netbeans

Configurando en el Modulo Arduino

En el siguiente programa, vamos a utilizar la biblioteca GSM_GPRS_GPS_Shield disponible en https://github.com/MarcoMartines/GSM-GPRS-GPS-Shield.

Dentro del bucle, se ejecuta la rutina que espera la llegada de un nuevo SMS. Una vez que se recibe el mensaje, la rutina se realiza envia_GSM, que se conectará a Internet a través de la red de telefonía móvil, y enviará al servidor información mediante una URL con el método _POST. Se necesita establecer la APN, nombre de usuario y la contraseña de acuerdo a su chip de celular: sintaxis:

Sintaxe: *inet.attachGPRS("<APN>", "<USUARIO>, "<SENHA>")*

Claro: *inet.attachGPRS("claro.pe", "claro", "claro")*

Movistart:

inet.attachGPRS("movistart.pe",

"movistar@datos", "movistart")333

El proceso se demora al conectar, tiene que identificar a su portadora, porque por defecto la biblioteca realiza una serie de rutinas de prueba y conexión.

```
//Programa: Arduino GSM Shield SIM900 - Acesso internet
//#include "SIM900.h"
#include <SoftwareSerial.h>
#include "inetGSM.h"


InetGSM inet;


boolean started = false;
char smsbuffer[160];
char n[20];


byte valor;
```

```
void setup()
{
  Serial.begin(9600);
  powerUpOrDown();
  Serial.println(F("Testando GSM Shield SIM900"));
  if (gsm.begin(2400))
  {
    Serial.println(F("\nstatus=READY"));
    started = true;
  }
  else Serial.println(F("\nstatus=IDLE"));
}

void loop()
{
  if (started) {
    //Aguarda novo SMS e envia para o servidor web
    if (gsm.readSMS(smsbuffer, 160, n, 20)) {
      String str(smsbuffer);
      envia_GSM(smsbuffer);
      delay(10000);
    }
    delay(1000);
```

```
  }
}

void powerUpOrDown()
{
  //Liga o GSM Shield
  Serial.print(F("Funsiona GSM..."));
  pinMode(6, OUTPUT);
  digitalWrite(6, LOW);
  delay(1000);
  digitalWrite(6, HIGH);
  delay(1000);
  Serial.println(F("OK!"));
  digitalWrite(6, LOW);
  delay(500);
}
void envia_GSM(String texto)
{
  char temp_string[55];
  char msg[10];
  int numdata;
  if (inet.attachGPRS("claro.pe", "claro", "claro"))
    Serial.println(F("status=Conectado..."));
```

```
    else Serial.println(F("status=No conectado !!"));
    delay(100);
    String valor = "MSG_Texto1=" + texto;
    valor.toCharArray(temp_string, 55);
    numdata = inet.httpPOST("milppublica", 8080, "/add",temp_string, msg, 50);
    delay(5000);
  }
```

Después de la transmisión de información entre el navegador y nos conectaremos con la dirección web http://localhost:8080/GPS/. Podrá visualizar los envíos de datos que ha generado. (Véase figura 4.19)

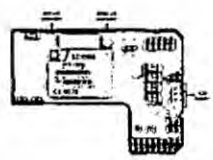
Figura 4.19 Pagina web Visualizando Datos

Etapa de integración Modulo Uno - GPS – BASE DE DATOS

Hasta ahora hemos utilizados los módulos por separado es decir por un lado el GPS y por otro lado el GPRS/GSM/4GLte, ahora necesitamos que se integre en un solo conjunto, actualmente existen módulos que integran ambas tecnologías.

Tenemos el módulo sim908 Arduino, (véase figura 4.20)

Figura 4.20 de Modulo Shield V3.0 SIM 900

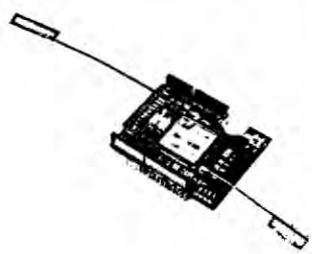


Fuente: Stack Exchange Inc. Disponible en

http://arduino.stackexchange.com/

También tenemos el módulo sim808 Arduino, (véase figura 4.21)

Figura 4.21 Modulo Shield V3.0 SIM880



Fuente: Stack Exchange Inc. Disponible en

http://arduino.stackexchange.com/

Para el desarrollo de nuestra tesis usaremos el módulo sim908 ya que incorpora todas las tecnologías de señal móvil en su versión 2.

Módulo Arduino SIM908

Es un módulo que ofrece una completa solución para comunicaciones y ubicación mediante satélites GPS. Ocupa solo dos pines del

microcontrolador, y podemos activar cada función acorde a nuestras necesidades

Descripción:

Este shield está diseñado por AND Tecnologies, con el chip SIM908, el cual posee un motor GSM/GPRS/Lte de cuatro bandas, además de tener integrado un receptor de GPS (Sistema Global de Navegación por Satélite). Es compatible con todas las tarjetas de Arduino.

El combinar GSM/GPRS/GPS nos permite diseñar sistemas de control y monitoreo de vehículos y personas para realizar un seguimiento en cualquier momento y en cualquier lugar con cobertura de la señal. Además podemos descargar la librería están disponible en http://www.gsmlib.org/download.html

Procedimiento:

Conecte la Antena GSM y GPS. Conecte la Fuente de alimentación (9V/1A).Conecte el cable USB

Características:

- Quad-Band 850/900/1800/1900MHz
- GPRS multi-slot class 10
- Control por commandos AT (GSM 07.07,07.05 and SIMCom enhanced AT Commands)
- Fuente de alimentación: 6-12V (Conectar una Fuente externa de 9V/1A para suministrar la corriente necesaria.)

- Dimensiones: 68.6 x 53.3mm (la misma que la tarjeta Arduino UNO)
- TCP/IP stack integrado.
- 100uF capacitor para RTC
- Tecnología GPS para navegación satelital.
  - 42-channel, GPS L1 C/A code
  - Sensibilidad de tracking: -160 dBm
  - Precisión en la posición: <2.5m CEP

Comandos AT

Son comandos para la parte de comunicación móvil, están permitirán controlar la parte de señal móvil donde controlaremos la conexión apn (conexión a internet), y el envío de mensajes url a internet, los comandos AT están disponible en: http://www.electronicaestudio.com/docs/ISTD-034.pdf

Tabla 4. 1 Lista de comando AT

Comandos para envio/recepción de mensajes

Command	Response	Description
AT+CMGF=	OK	Specifies the input and output format of the short messages. 0 for PDU mode and 1 for text mode.

AT+CMGS	Sends a message.
AT+CMGR=*	Reads a message. * is the number of the message.

Comandos para el establecer conexión APN en tcp/udp

AT command	Response	Description
AT+CIPMUX=	OK	Selects single connection (0) or multiple connection (1)
AT+CSTT="myAPN"	OK	Sets APN
AT+CIICR		Brings up wireless connection
AT+CIFSR		Get local IP address
AT+CIPSTART		Establishes a connection with a server.
AT+CIPSEND		Sends data when the a connection is established.

AT+CIPCLOSE	Closes the connection

Comandos para comunicación HTTP

AT command	Response	Description
AT+SAPBR	OK	Configures GPRS profile
AT+HTTPINIT	OK	Initializes HTTP service
AT+HTTPPARA	OK	Configures HTTP parameters
AT+HTTPACTION=0	OK	Sets HTTP Method Action , GET in this chase.
AT+HTTPREAD		Reads HTTP data
AT+HTTPTERM	OK	Closes the opened HTTP session.

Comando para utilización de GPS

AT command	Response	Description
AT+CGPSPWR	OK	Powers the GPS

AT+CGPSRST	OK	Sets the reset mode
AT+CGPSSTATUS		Gets the status: Unknown, Not Fix, 2D Fix and 3D Fix
AT+CGPSINF	NMEA string	Gets NMEA strings

Fuente: Electrónica Estudio disponible en

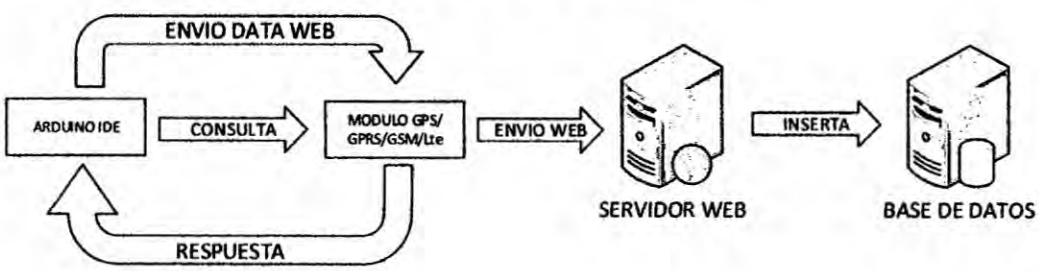
http://www.electronicaestudio.com/docs/ISTD-034.pdf

De todo lo diseñado, ahora usaremos lo aprendido de los diferentes modulo, agrupándolo en uno solo pero con la misma idea, es decir el Arduino Uno, es el que va a gestionar toda la información obtenida del módulo sim908,

- cuando el Arduino Uno, tienes que realizar dos tareas una es de consulta y la otra de envió, la rutina consulta al módulo sim908, proporcionando la información del GPS dando los datos de coordenadas, fecha, hora, según lo que necesitemos almacenar en la base de datos. La rutina de envió de datos es la recolección de datos que será enviada por el módulo sim908 pero ahora por la parte del GSM/GPRS, según lo indicamos este envió tiene que ser en formato web mediante el método post el cual será decepcionado por la página web y almacenado en la base de datos para luego ser depurada y analizada.

La programación a realizar es una consecuencia de lo ya visto pero con los comandos AT ya que estos comandos son del módulo sim908, donde el Arduino Uno tendrá que enviar dichos comandos veamos la programación base.

Figura 4.22 Diagrama de Flujo



Fuente: Auditoria Propia

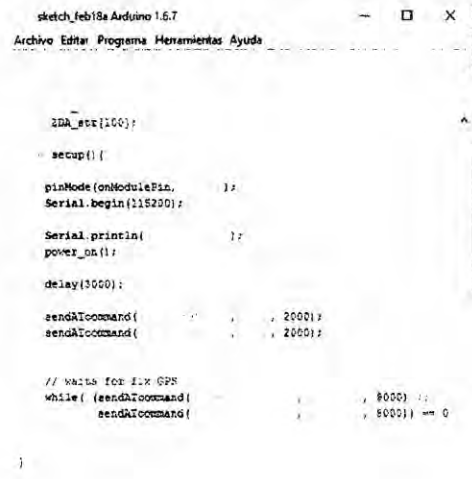
Realizaremos la declaración de las variables para poder trabajar dentro del programa Arduino.

Figura 4.23 Declaración de variables

Fuente: Software Arduino

Fijaremos de los puertos y velocidades de comunicación entre los módulos. Esto lo realizamos en el void setup(), el pin a utilizar será el 2, declarado entre las variables.

Figura 4.24 Fijando Parámetros



Fuente: Software Arduino

Ahora viene la programación en void loop(), donde tenemos que realizar las consultas al GPS y el envió http mediante el método post a través del APN del proveedor del chip conectado.

```
void loop(){

  while( Serial.available() > 0) Serial.read();

  delay(100);

  // request Basic string
```

```
sendATcommand("AT+CGPSINF=0", "AT+CGPSINF=0\r\n\r\n", 2000);

counter = 0;

answer = 0;

memset(Basic_str, '\0', 100);    // Initialize the string

previous = millis();

// el loop esta en espera por el NMEA string

do{

  if(Serial.available() != 0){

    Basic_str[counter] = Serial.read();

    counter++;

    if (strstr(Basic_str, "OK") != NULL)

    {

      answer = 1;

    }

  }

  }while((answer == 0) && ((millis() - previous) < 2000));
```

```
    Basic_str[counter-3] = '\0';
    //************************************************************
// GGA
// Clean the input buffer
while( Serial.available() > 0) Serial.read();
delay(100);
// request GGA string
sendATcommand("AT+CGPSINF=2", "AT+CGPSINF=2\r\n\r\n", 2000);
counter = 0;
answer = 0;
memset(GGA_str, '\0', 100);    // Initialize the string
previous = millis();
// this loop waits for the NMEA string
do{
   if(Serial.available() != 0){
      GGA_str[counter] = Serial.read();
```

```
      counter++;
      // check if the desired answer is in the response of the module
      if (strstr(GGA_str, "OK") != NULL)
      {
        answer = 1;
      }
    }
    // Waits for the asnwer with time out
}while((answer == 0) && ((millis() - previous) < 2000));
GGA_str[counter-3] = '\0';
//*************************************************************
// GLL
// Clean the input buffer
while( Serial.available() > 0) Serial.read();
delay(100);
// request GLL string
```

```
sendATcommand("AT+CGPSINF=4", "AT+CGPSINF=4\r\n\r\n", 2000);

counter = 0;

answer = 0;

memset(GLL_str, '\0', 100);    // Initialize the string

previous = millis();

// this loop waits for the NMEA string

do{

  if(Serial.available() != 0){

      GLL_str[counter] = Serial.read();

      counter++;

      // check if the desired answer is in the response of the module

      if (strstr(GLL_str, "OK") != NULL)

      {

         answer = 1;

      }

  }
```

```
    // Waits for the asnwer with time out

  }while((answer == 0) && ((millis() - previous) < 2000));

  GLL_str[counter-3] = '\0';

  //************************************************************

  // RMC

  // Clean the input buffer

  while( Serial.available() > 0) Serial.read();

  delay(100);

  // request RMC string

  sendATcommand("AT+CGPSINF=32",          "AT+CGPSINF=32\r\n\r\n",
2000);

  counter = 0;

  answer = 0;

  memset(RMC_str, '\0', 100);    // Initialize the string

  previous = millis();

  // this loop waits for the NMEA string
```

```
do{
if(Serial.available() != 0){
      RMC_str[counter] = Serial.read();
      counter++;
      // check if the desired answer is in the response of the module
      if (strstr(RMC_str, "OK") != NULL)
      {
         answer = 1;
      }
  }
  // Waits for the asnwer with time out
}while((answer == 0) && ((millis() - previous) < 2000));
RMC_str[counter-3] = '\0';
//**********************************************************
// VTG
// Clean the input buffer
```

```
  while( Serial.available() > 0) Serial.read();

  delay(100);

  // request VTG string

  sendATcommand("AT+CGPSINF=64",        "AT+CGPSINF=64\r\n\r\n",
2000);

  counter = 0;

  answer = 0;

  memset(VTG_str, '\0', 100);    // Initialize the string

  previous = millis();

  // this loop waits for the NMEA string

  do{

    if(Serial.available() != 0){

      VTG_str[counter] = Serial.read();

      counter++;

      // check if the desired answer is in the response of the module

      if (strstr(VTG_str, "OK") != NULL)
```

```
        {
            answer = 1;
        }
      }
      // Waits for the asnwer with time out
    }while((answer == 0) && ((millis() - previous) < 2000));

    VTG_str[counter-3] = '\0';
    //*************************************************************
    // ZDA
    // Clean the input buffer
    while( Serial.available() > 0) Serial.read();
    delay(100);
    // request ZDA string
    sendATcommand("AT+CGPSINF=128",     "AT+CGPSINF=128\r\n\r\n",
2000);
```

```
counter = 0;

answer = 0;

memset(ZDA_str, '\0', 100);    // Initialize the string

previous = millis();

// this loop waits for the NMEA string

do{

  if(Serial.available() != 0){

    ZDA_str[counter] = Serial.read();

    counter++;

    // check if the desired answer is in the response of the module

    if (strstr(ZDA_str, "OK") != NULL)

    {

      answer = 1;

    }

  }

  // Waits for the asnwer with time out
```

```
    }while((answer == 0) && ((millis() - previous) < 2000));


    ZDA_str[counter-3] = '\0';

    // Aquí debemos realizar el envio atravez de l protocolo AT

    Serial.println("**************                                    DISPLAY
GPS**********************************");

    Serial.print("Basic string: ");

    Serial.println(Basic_str);

    Serial.print("GGA string: ");

    Serial.println(GGA_str);

    Serial.print("GLL string: ");

    Serial.println(GLL_str);|

    Serial.print("RMC string: ");

    Serial.println(RMC_str);

    Serial.print("VTG string: ");

    Serial.println(VTG_str);
```

```
    Serial.print("ZDA string: ");
    Serial.println(ZDA_str);


    delay(15000);
}
void power_on(){
    uint8_t answer=0;
    // checks if the module is started
    answer = sendATcommand("AT", "OK", 2000);
    if (answer == 0)
    {

      // power on pulse
      digitalWrite(onModulePin,HIGH);
      delay(3000);
      digitalWrite(onModulePin,LOW);
```

```
    // waits for an answer from the module
    while(answer == 0){
      // Send AT every two seconds and wait for the answer
      answer = sendATcommand("AT", "OK", 2000);
    }
  }
}
int8_t sendATcommand(char* ATcommand, char* expected_answer1,
    unsigned int timeout)
{
  uint8_t x=0,  answer=0;
  char response[100];
  unsigned long previous;
  memset(response, '\0', 100);    // Initialize the string
  delay(100);
  // Clean the input buffer
```

```
while( Serial.available() > 0) Serial.read();

Serial.println(ATcommand);    // Send the AT command

x = 0;

previous = millis();

// this loop waits for the answer

do{

  if(Serial.available() != 0){

    response[x] = Serial.read();

    x++;

    // check if the desired answer is in the response of the module

    if (strstr(response, expected_answer1) != NULL)

    {

      answer = 1;

    }

  }

  // Waits for the asnwer with time out
```

```
}while((answer == 0) && ((millis() - previous) < timeout));

return answer;

}
```

Una vez que tengamos dichos datos en la página web debemos ir depurando cada dato, el cual nos dará la información de cada punto de envió que se ha envía por el sistema Arduino, dando como información las coordenadas, fecha/hora, las cuales podemos analizar el tiempo, distancia, velocidad, consumo, entre otras cosas que se podrán ir depurando en la programación.

Figura 4.25 Visualización del dato enviado

Fuente: Pagina Web Traccar

4.3. Población y Muestra

4.3.1. Población

Según Rodrigo & Molina un colectivo objeto de estudio está formado por un conjunto de elementos con características similares y sobre el que se pretende inferir regularidades.

Para la tesis se contó con 120 unidades de transporte de la empresa Nettelcom.

Especificación técnica de la muestra

Hino Dutro serie 300

- Motor Diesel / Eléctrico
- Cabina simple

Figura 4.26 Camion Hino serie 300

Fuente: Hino Japon disponible en http://www.pacrodidacol.com

Resultados Acumulados:

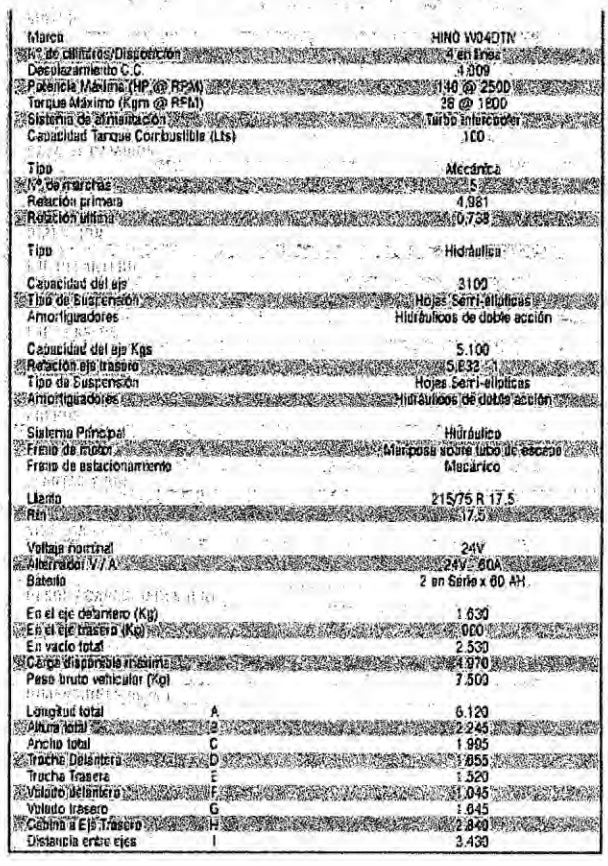
Hino Recorrido Total 19,669 km.

Consumo de Combustible 3,790 lts

Rendimiento 5.19 km / lts

Hoja técnica:

Figura 4.27 Hoja Técnica Hino serie 300



Marca		HINO W04DTN
Nº de cilindros/Disposición		4 en línea
Desplazamiento C.C.		4.009
Potencia Máxima (HP @ RPM)		140 @ 2500
Torque Máximo (Kgm @ RPM)		28 @ 1800
Sistema de alimentación		Turbo intercooler
Capacidad Tanque Combustible (Lts)		100
Tipo		Mecánica
Nº de marchas		5
Relación primera		4,981
Relación última		0,758
Tipo		Hidráulica
Capacidad del eje		3100
Tipo de Suspensión		Hojas Semi-elípticas
Amortiguadores		Hidráulicos de doble acción
Capacidad del eje Kgs		5.100
Relación eje trasero		5,833
Tipo de Suspensión		Hojas Semi-elípticas
Amortiguadores		Hidráulicos de doble acción
Sistema Principal		Hidráulico
Freno de motor		Mariposa sobre tubo de escape
Freno de estacionamiento		Mecánico
Llanto		215/75 R 17.5
Rin		17,5
Voltaje nominal		24V
Alternador V / A		24V - 60A
Bateria		2 en Serie x 60 AH.
En el eje delantero (Kg)		1.630
En el eje trasero (Kg)		900
En vacio total		2.530
Carga disponsible maxima		4.970
Peso bruto vehicular (Kg)		7.500
Longitud total	A	6.120
Altura total	B	2.245
Ancho total	C	1.995
Trocha Delantera	D	1.655
Trocha Trasera	E	1.520
Voladio delantero	F	1.045
Voladio trasero	G	1.645
Cabina a Eje Trasero	H	2.840
Distancia entre ejes	I	3.430

Fuente: Praco Didacol S.A. disponible http://www.pacrodidacol.com

4.3.2. Muestra

Según Rodrigo & Molina, la muestra es el sub conjunto de la población o colectivo de la población.

Para determinar el tamaño de la muestra se utilizó la siguiente formula.

$$n = \frac{Z_{\alpha}^{2}.N.p.q}{i^{2}(N-1)+Z_{\alpha}^{2}.p.q}$$

Donde:

n: tamaño muestral

N: tamaño de la población

Z: valor correspondiente a la distribución de gauss

p: prevalencia esperada del parámetro a evaluar, en caso de desconocerse (p=0.5), que hace mayor el tamaño muestral.

q: 1-p (si p =70%, q =30%)

i: error que se prevé cometer si el del 10%, i=0.1

Calculo de la muestra

Se evaluara para una población de 120 unidades tendremos los siguientes datos:

N=120

P=0.7

q=0.3

Z=1.986

i =10%=0.1

Realizando los cálculos respectivos tendríamos el siguiente resultado

$$n = \frac{1.96^2 * 120 * 0.7 * 0.3}{0.1^2(120-1)+1.96^2 * 0.7 * 0.3} = 48.15$$

Por lo tanto la muestra para el desarrollo de la tesis fue de 50 unidades de transporte, con los cuales se realizaron las pruebas del sistema de control.

4.3. Delimitación

4.3.1. Ubicación espacio

En el presente trabajo se diseñó un sistema de ruta con GPS/4G Lte para el control y seguridad en las unidades de la empresa Nettelcom S.A.C. la ubicación espacial fue en Lima Metropolitana.

4.4. Técnica e Instrumentación de recolección de datos.

4.4.1. Técnicas de recolección de datos:

Ficha técnica de obtención de los datos enviados por el GPS, para la evolución de la utilizo que el gps envie un trama cada 30 segundos

4.4.2. Instrumentos:

El GPS instalado en cada unidad de transporte.

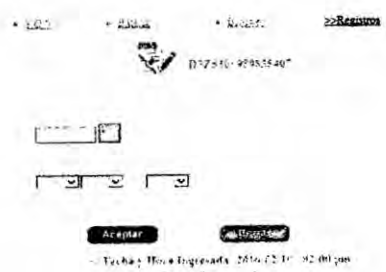
4.5. Procedimiento estadístico y análisis de datos

El trabajo se desarrolló mediante la estadística descriptiva en SPSS ver.22 (paquete estadístico para la ciencia sociales).

V. RESULTADOS.

Se obtuvo una serie de puntos consecutivos los cuales nos da la ruta de conducción, conociendo la distancia y el tiempo entre punto y punto se puede calcular la velocidad referencial, esto también se puede nos permitirá realizar cálculos referentes como consumo de combustible, entre otras cosas. Esto se almacena en la base de datos donde podemos consultarlo

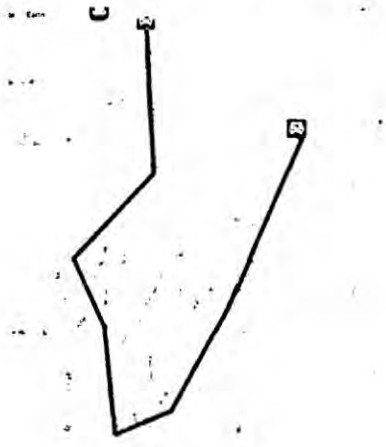
Figura 5.1 Tabla de recorrido



Fuente: Pagina Traccar

Mediante el Webserver Traccar donde almacenaremos las tramas tendremos la ruta establecida.

Figura 5.2 ruta creada por los datos



Fuente: Pagina Traccar

5.1. Pruebas

Todo el sistema se probó antes de utilizarlo, ya que el costo es menor si se detectan los problemas antes de entrar en funcionamiento, se hizo una serie de pruebas con datos reales del sistema.

Se planteó una ruta para realizar las pruebas del punto a (jr los amautas 1277) al punto B (hospital de San Juan Lurigancho), donde se envió a tres unidades que tomaran rutas distintas logrando recorridos y tiempos distintos.

Las cuales no proporciona la información de distancia y tiempo que han demorado en cada ruta, esto nos permite evaluar cuál es la mejor ruta con menor distancia y menor tiempo, para establecer como una ruta confiable

Figura 5.3 Ruta Establecidas

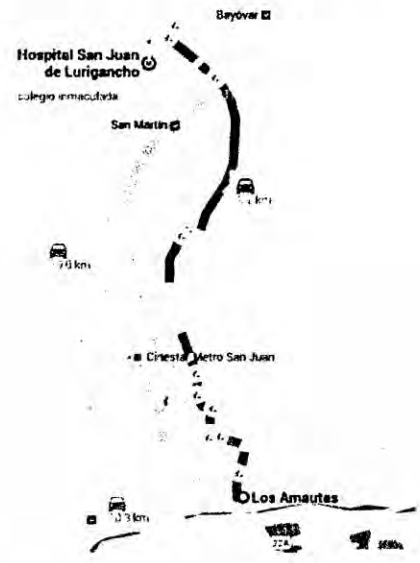


Tabla 5. 1 Resultados

CAR-0001	Tiempo Hora/min	Distancia km	Consumo $5.19\ ^{km}/_{lts}$	Velocidad Prom $^{km}/_{hora}$
RUTA 1	25 min	10	1.92	24
RUTA 2	27 min	10.3	1.98	22.8
RUTA 3	25 min	9.7	1.88	23.28

VI. DISCUSIONES DE LOS RESULTADOS

6.1. Contrastación de la Hipótesis con los resultados

Se ha obtenido los resultados esperados ya que al observar los diferentes recorridos se puede tomar decisiones para establecer la una ruta confiable como además nos permite observar cambios de dicha ruta en todo momento, donde siempre estará en evaluación para su mejora ya que puede intervenir factores externo como tráfico, accidente en carretera o alguna obra que lleve a modificar la ruta pre- establecida.

6.2. Contrastación de la Hipótesis con otros estudios Similares

El Ingeniero Rubén, Bocanegra Ureta, en su trabajo de tesis "Desarrollo de una aplicación web para el monitoreo de vehículos con dispositivos GPS que comercializa una empresa de telecomunicaciones", utiliza herramientas de software con licencia, para el desarrollo del software como también para la base de datos, esto ocasiona un inversión extra para la empresa donde se implemente.

Edgar Chilan Soledispa tesis "Desarrollo de aplicación para presentar reportes gráficos (rutas vehiculares) que se visualicen en Google Maps". Estoy de acuerdo en el uso del api de google Maps ya que están de uso libre, permitiendo su compatibilidad con cualquier software de desarrollo.

VII. CONCLUSIONES

Después de analizar la información que se ha recopilado sobre este tipo de desarrollo se puede anotar varias conclusiones:

- Las tecnologías modular en Arduino, donde los sistemas de microcontroladores llevados a un entorno de programación más amigable donde nos permita integrar los módulos que estemos necesitando para el desarrollo de nuestro proyecto.
- Se encuentra una gran información sobre el desarrollo de la aplicación web en una plataforma Php, se ha hecho un cambio realizando trabajo en Java donde podemos darle una mayor seguridad y estabilidad, bajo software libre.
- La programación que trabaja arruino software es java la cual al seguir el desarrollo a una aplicación web en Java, todo trabaja bajo un solo lenguaje nativo.
- Este proyecto permite establecer un marco teórico de referencia para nuevas investigaciones dentro de la línea de la aplicación de las tecnologías Web y los sistemas de localización adaptables para cualquier entorno.

VIII. RECOMENDACIONES

Después de analizar la información que se ha recopilado sobre este tipo de desarrollo se puede anotar varias conclusiones:

- Los módulos Arduino, tienen una gran variedad para desarrollo de la aplicación, es conocer la parte aplicativa para escoger el modulo correcto y lograr un mejor diseño para nuestra aplicación.
- Este de tipo de desarrollo no debe estar sujeto a un medio de información como es caso de TRACCAR; sino que debe acoplarse a cualquier base de datos que contenga información sobre coordenadas de ubicación, las mismas que pueden en cualquier momento ser presentadas en cualquier tipo de mapas digitales.
- El desarrollo de hardware & software no se detienen siempre están haciendo innovaciones lo que nos lleva a estar buscando nuevas tecnologías para poder actualizar nuestros trabajo y lograr una plataforma moderna y estable.

IX. REFERECIA BIBLIOGRAFICA.

Bibliografía

1. BOCANEGRA URETA, RUBEN, tesis **"Desarrollo de una aplicación web para el monitoreo de vehículos con dispositivos GPS que comercializa una empresa de telecomunicaciones"**, tesis de Ingeniera Universidad Ricardo Palma , Lima – Perú, 2012.
2. GARZON MANUEL, tesis **"Diseño de sistema que permita mantener un estándar de tiempo entre buses consecutivos de una misma ruta"**, tesis de Grado Ingeniería, Universidad Santiago de Cali, Cali - Colombia.2009.
3. CHILAN SOLEDISPA EDGAR, tesis **"Desarrollo de aplicación para presentar reportes gráficos (rutas vehiculares) que se visualicen en Google Maps"**. Tesis Ingeniería, Universidad de Guayaquil, Guayaquil – Ecuador, 2013
4. DANIEL VARGAS, El Sistema GPS, disponible en https://vargasdaniel27.wordpress.com/2008/04/28/el-sistema-gps-telecomunicaciones/, articulo web, consultada el 30 de enero del 2016
5. GPS Global Security Disponible en http://www.gsp.com.co
6. ANONIMO, **La revolución de la comunicación**, disponible en http://revolucioncomunicacion.weebly.com/gps.html, articulo web, consultada el 30 de enero de 2016

7. GPS Global Security Disponible en http://www.gsp.com.co

8. Satélites de Posicionamiento Global disponible en http://hyperphysics.phy-astr.gsu.edu/hbasees/gps.html

9. PEDRO GUTOVNIK, **Como Funciona el GPS**, disponible en http://gutovnik.com/como_func_sist_gps.htm, referencia web, consultada el 30 de enero de 2016

10. Sistema NavStart disponible en http://fcaglp.unlp.edu.ar/referenciacion/index.php?title=Portal:Sistema_NAVSTAR_GPS&redirect=no

11. VIENNA UNIVERSITY OF TECHNOLOGY, GUNTHER GRIDLING, **Introduction to Microcontrollers**, Disponible en: https://ti.tuwien.ac.at/ecs/teaching/courses/mclu/theory-material/Microcontroller.pdf, artículo web, consultado el 01 de enero del 2016.

12. JOSE ADOLFO GONZALES, **Introducción a los Microcontroladores**, España, McGraw-Hill, 1994

13. JARED TIRADO MARTINEZ, Microcontroladores, disponible en https://issuu.com/jarettto/docs/loaiza-microcontroladores_-1-, articulo web, consultada el 31de enero del 2016.

14. UNIVERSIDAD NACIONAL DE INGENIERIA, O.BUSTILLOS y C. MARTINEZ, **Microprocesadores vs Microcontroladores**,

disponible en disponible https://electrouni.files.wordpress.com/2010/12/micro-vs-microcon.pdf.

15. RICHARD STALLMAN, CONCEPTOS DEL FREE SOFTWARE, disponible en http://eforma.kzgunea.eus/mod/book/view.php?id=649&chapterid=1285, artículo web, consultado 15 de enero de 2016.
16. DOUGLAS CROCKFORD, **Java Script**, disponible en http://www.crockford.com/javascript/javascript.html, articulo web, consultado el 15 de enero del 2016.
17. WIKIPEDIA, **Lenguaje Ensamblador**, disponible en https://es.wikipedia.org/wiki/Lenguaje_ensamblador, articulo web, consultado 17 de enero de 2016.
18. FABRICANTE ARDUINO, **Arduino**, disponible en https://www.arduino.cc/, blog Arduino, consultada 15 de noviembre del 2015.
19. Robotica al Descubierto disponible en http://solorobotica.blogspot.pe/2012_07_01_archive.html
20. ElectroTec Peru disponible en http://electrotec.pe/blog/GPS
21. Electrónica Estudio disponible en http://www.electronicaestudio.com/docs/ISTD-034.pdf

22. HUITRON HERNANDEZ, Base de Datos, disponible en http://docencia.fca.unam.mx/CAACS/bases-datos.php, articulo web, consultada el dia 20 de diciembre del 2015

23. NETBEANS, Netbeans, disponible en https://netbeans.org/, pagina web, consultada el 03 de febrero del 2016,

24. GARCIA BELTRAN, **Programación Java, Madrid**, Consorcio OpenCourseWare, segunda edición, 2009

25. TRACCAR, Traccar, disponible en https://www.traccar.org/, sitio web, consultado el 1 de diciembre del 2015

26. Stack Exchange Inc. Disponible en http://arduino.stackexchange.com/questions/9483/how-to-communicate-the-arduino-board-with-sim900

27. Electronica Estudio disponible en http://www.electronicaestudio.com/docs/ISTD-034.pdf

28. Hino Japon disponible en http://www.pacrodidacol.com

29. Praco Didacol S.A. disponible http://www.pacrodidacol.com

ANEXOS

Anexo 01 Matriz de consistencia

DISEÑO DE UN SISTEMA DE RUTA CON GPS/4G LTE PARA EL CONTROL DE LAS UNIDADES DE LA EMPRESA NETTELCOM SAC

PROBLEMAS	OBJETIVO	HIPÓTESIS	VARIABLE E INDICADORES			
			VARIABLES	DIMENSIONES	INDICADORES	ESCALA
Problema General: ¿De qué manera el diseño de un sistema de ruta con GPS/4G Lte controla las unidades de la empresa Nettelcom SAC?	**Objetivo General:** Determinar el diseño de un sistema de ruta con GPS/4G Lite en el control de las unidades de la empresa Nettelcom SAC	**Hipótesis General:** Existe una relación significativa entre el diseño de un sistema de ruta con GPS/4G Lite y el control de las unidades de la empresa Nettelcom SAC.	Vx EL DISEÑO DE UN SISTEMA DE RUTA	Ubicación	Ubicación de coordenadas,	
Problemas Específicos: ¿De qué manera el diseño de un sistema de ruta con GPS/4G Lte controla el tiempo en las unidades de la empresa Nettelcom SAC?	**Objetivos Específicos:** Determinar la relación que existe entre el diseño de un sistema de ruta con GPS/4G Lte controla el tiempo en las unidades de la empresa Nettelcom SAC.	**Hipótesis Específicas:** Existe una relación significativa entre el diseño de un sistema de ruta con GPS/4G Lte controla el tiempo en las unidades de la empresa Nettelcom SAC.		Fecha/Hora	Fecha, días, horas, min, segundo	
¿De qué manera el diseño de un sistema de ruta con GPS/4G Lte controla la distancia en las unidades de la empresa Nettelcom SAC?	Determinar la relación que existe entre el diseño de un sistema de ruta con GPS/4G Lte controla la distancia en las unidades de la empresa Nettelcom SAC.	Existe una relación significativa entre el diseño de un sistema de ruta con GPS/4G Lte controla la distancia en las unidades de la empresa Nettelcom SAC.	Vy ONTROL EN LAS UNIDADES DE LA EMPRESA NETTELCOM	1. Tiempo	Intervalos de Tiempo de la unidad, hora min segundo	Bueno Malo regular
¿De qué manera el diseño de un sistema de ruta con GPS/G Lte controla el consumo en las unidades de la empresa Nettelcom SAC?	Determinar la relación que existe entre el diseño de un sistema de ruta con GP/4G Lte controla el consumo en las unidades de la empresa Nettelcom SAC.	Existe una relación significativa entre el diseño de un sistema de ruta con GPS/4G Lte controla el consumo en las unidades de la empresa Nettelcom SAC.		2. distancia	km,metros	
¿De qué manera el diseño de un sistema de ruta con GPS/4G Lte controla la velocidad en las unidades de la empresa Nettelcom SAC?	Determinar la relación que existe entre el diseño de un sistema de ruta con GPS/4G Lte controla la velocidad en las unidades de la empresa Nettelcom SAC.	Existe una relación significativa entre el diseño de un sistema de ruta con GPS/4G Lte controla la velocidad en las unidades de la empresa Nettelcom SAC.		3.Consumo	galones, litros, m3	
				4 velocidad	k/h, m/s	