UNIVERSIDAD POLITÉCNICA SALESIANA

SEDE CUENCA

CARRERA DE INGENIERÍA DE SISTEMAS

TESIS PREVIA A LA OBTENCIÓN DEL TÍTULO DE:
INGENIERO DE SISTEMAS

TÍTULO DE TESIS

"ANÁLISIS, DISEÑO E IMPLEMENTACIÓN DE UN SISTEMA DE ALARMA PARA EL MONITOREO DEL REGISTRO DE LA LLUVIA EN LA CIUDAD DE CUENCA BASADO EN EL PROTOCOLO GPRS".

AUTORES

JUAN CARLOS AMAY IZQUIERDO

FABIÁN PAÚL TACURI PUMA

DIRECTOR:

ING. RODOLFO BOJORQUE

CUENCA, JUNIO DE 2014.

CERTIFICACIÓN DEL DIRECTOR DE TESIS

Certifico que el presente trabajo de tesis titulado "Análisis, Diseño e Implementación de un Sistema de alarma para el monitoreo del registro de la lluvia en la ciudad de Cuenca basado en el protocolo GPRS", fue dirigido y revisado prolijamente bajo mi dirección en todo su desarrollo, la misma cumple con la reglamentación pertinente y con los objetivos planteados en la denuncia del mismo, y autorizo su presentación.

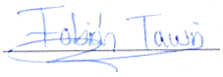
Cuenca, Junio del 2014

Ing. Rodolfo Bojorque
Director de tesis

DECLARATORIA DE RESPONSABILIDAD

El análisis de conceptos y las ideas vertidas en la presente tesis, son de total responsabilidad de los autores y autorizamos a la Universidad Politécnica Salesiana el uso de la misma con fines académicos.

Juan Carlos Amay Izquierdo



Fabián Paúl Tacuri Puma

Dedicatoria

A Dios, reconozco y creo en un ser supremo, el cual tiene el control de todas las cosas, el que con amor enseña y con sabiduría corrige, reconozco a Jesús como el rey de mi vida y quiero dedicarle mi arduo trabajo el que con alegría he podido cumplir, fue un largo camino, un camino muchas veces incierto, un camino como el desierto que pone a prueba hasta las más fuertes convicciones, y en este camino siempre sentí su amor, su apoyo, y nunca me sentí solo.

A mi familia la que también me acompañó, la que estuvo y esta hasta este momento en mi vida, a mi madre Esperanza, siempre serás mi esperanza la que me da fuerzas y me infunde el aliento para hacer las cosas bien, la que siempre me habla del temor a Dios que es el principio de la sabiduría, la que me ha llevado a seguir el camino de la rectitud, el agradecimiento y la humildad.

A mi padre que aunque se encuentre a la distancia, desde pequeño me corrigió, con disciplina me educó, no permitió que siga malos pasos, gracias todavía escucho tu voz papá.

Juan Carlos

Agradecimiento

Agradezco los resultados de este proyecto, a la Universidad Politécnica Salesiana y a todas aquellas personas que, de alguna forma han intervenido en mi desarrollo como estudiante dentro de esta prestigiosa Universidad.

Juan Carlos

Dedicatoria

Al ser ineludible que algunos llaman Dios y otros como yo lo conocemos como ese gran poder que reposa en cada uno de nosotros para poder ser todo lo que queremos en la vida, porque con cada mañana, con cada abrir de ojos él está en nuestro cuerpo que es su templo desde donde podemos obrar conforme a nuestra ética.

Dedico este trabajo a quien con un abrazo profundo de padre un día me enseñó a vivir en el camino del bien, a usted "papá" Eloy (+) que con tus sabias palabras siempre supiste poner en mi la idea de que lo mejor que uno puede hacer es defenderse solo en la vida. A usted querida madre que todos días de mi vida estuvo a mi lado. A mi abuelita Isaura por todo el amor que siempre me ha dado y por ser siempre incondicional conmigo sé que si fuera el peor criminal del mundo o el más grande genio del mundo su amor por mí no cambiaría. A mi abuelita Margarita por ser la gran maestra de mi vida, ella con simples palabras siempre logró que aprendiera a afrontar todo en la vida. A mi abuelito Julio por ser parte esencial en mi vida. A mis tías Mérida, Janeth y Fanny porque más que mis tías han sido mis amigas y hermanas, cada una me ha dado una razón para vivir en el camino del bien. A mi abuelita Blanca que siempre me apoyó incondicionalmente. A mi papá por darme la vida y ser un gran maestro.

Fabián Paúl.

Agradecimiento

Quiero agradecer a mis hermanos Hillary y Christian que han llenado mi vida con su amor y ternura. A mis tíos Geovanny, Mery y Cecilia por todo el apoyo brindado en todo el transcurso de mi vida estudiantil. A mi gran amiga y tía a la vez Patricia por ser la persona que siempre podré acudir sin reparos. A mis amigos que de una u otra manera en el transcurso de mi vida, desde que los conocí hasta el día de hoy hemos compartido grandes momentos de felicidad, tristeza y superación: Efrén, Diego, Mario, Edgar, Celia. A mis amigos de la universidad compañeros de clases y hoy colegas Karina, Jessica, Elisa, Diego, Juan, José y todos aquellos grandes amigos de incontables batallas por conseguir nuestro objetivo.

Fabián Paúl.

Cuenca, 09 de abril de 2014

Ingeniera
Bertha Tacuri
DIRECTORA DE LA CARRERA DE INGENIERIA DE SISTEMAS
DE LA FACULTAD DE INGENIERIAS DE LA UNIVERSIDAD
POLITECNICA SALESIANA
Su despacho.

De mi consideración:

El día de hoy miércoles 09 de abril de 2014, en las oficinas de la Subgerencia de Operaciones de Agua Potable y Saneamiento de ETAPA EP, los señores: Juan Carlos Amay Izquierdo y Fabián Paúl Tacuri Puma, estudiantes de Ingeniería de Sistemas de la Universidad Politécnica Salesiana, realizaron la presentación de su proyecto de grado titulado "Análisis, Diseño e Implementación de un Sistema de Alarma para el monitoreo del registro de la lluvia en la ciudad de Cuenca basado en el protocolo de GPRS", al respecto le informo que este trabajo cumplió con todos los objetivos planteados inicialmente y considero que a futuro podría tener una aplicación directa muy importante que beneficiaría a la empresa ETAPA EP y a la ciudad en general.

Atentamente,

E.T.A.P.A. E.P.
SUBGERENCIA DE
OPERACIONES DE AGUA
POTABLE Y SANEAMIENTO

Ing. Javier Fernández de Córdova W.
INGENIERO DE GESTION DE RIESGOS

Benigno Malo 7-78 y Sucre
Tel.: (593-7) 283 1900 Fax: (593-7) 2833048
Cuenca – Ecuador.
www.etapa.net.ec

Índice de contenidos

Contenido

LISTA DE TABLAS

LISTA DE FIGURAS

RESUMEN

Cuenca es una ciudad que cuenta con cuatro principales ríos Machángara, Sayausí, Tarqui y río Tomebamba los cuales brindan un sin fin de beneficios a sus habitantes, algunos de ellos, agua saludable y limpia, apta para el consumo humano, el cuidado de este importante recurso natural ha llevado al estudio y monitoreo de los afluentes, su comportamiento ante la presencia de lluvia, generando así un área de estudio alrededor de estas vertientes.

Para llevar a cabo el desarrollo de este trabajo fue necesaria la implementación de herramientas que permitan la recolección de datos tales como pluviómetros y/o estaciones meteorológicas, las mismas que han sido ubicadas en puntos estratégicos, puntos justificados con un estudio previo realizado por la empresa AMRA, empresa de consultoría ambiental.

Con los resultados obtenidos fue posible el desarrollo de una aplicación web y de escritorio útiles para interpretar los datos recogidos, generar mapas con la ayuda de tecnologías web como lo son Open ayer, Google Maps, etc.

Para así dar vida a un sistema que interactúa con el usuario, generando reportes, alertas visuales en el mapa o por medio de mensajes de texto usando la red GSM la cual brinda cobertura en los puntos necesarios para nuestro estudio, etc. El sistema además cuenta con un área de administración la que permite al usuario observar los cambios en las estaciones en tiempo real. Es importante resaltar el aporte social hacia las comunidades que se han asentado alrededor de los puntos sensibles ante desbordes de los ríos, comunidades que se sienten apoyadas y atendidas ante la presencia de proyectos que se ocupen del estado de los recursos naturales, esta necesidad de seguridad es atendida por medio de la propuesta de alarmas a las orillas de cada río, para este efecto la tecnología electrónica (Arduino) y sirenas de aviso se han considerado como una solución válida. En conclusión, el desarrollo de aplicaciones de software y electrónicas en conjunto con estudios medio ambientales han dado como resultado la respuesta a un problema que es real, el cual se lo vive fuertemente en familias y comunidades que dependen directamente de la corriente de los ríos. El estudio continúo e integración de todos los datos que se puedan obtener tanto en medición de caudales, medición de la precipitación de lluvias, niveles de los ríos, etc., ayudará a mejorar el rendimiento del sistema y así también hacer más eficaz un sistema de alerta.

Capítulo 1 CICLO INTEGRAL DEL AGUA

1.1 Antecedentes

El agua es un recurso vital que ha influido directamente en la historia de la humanidad pues ha sido una de las bases fundamentales para el desarrollo de la civilización. Es así que grandes civilizaciones como la egipcia o la incaica ya contaban con sistemas de captación y distribución orientados especialmente al riego de sus cultivos. Para esto realizaban la captación de agua desde sus fuentes y también desviaban el curso normal de los ríos, hasta llevar el agua hasta el lugar que necesitaban. Es así como el ciclo del agua o ciclo hidrológico se vio interrumpido por la mano del hombre.

El agua está en un continuo proceso de cambio, es así que la encontramos en la atmósfera, en la superficie terrestre y en el suelo. La cantidad total de agua nunca cambia. Todo este movimiento lo debe a la fuerza de la gravedad y de la incidencia de la energía solar. La repetición de este proceso se denomina ciclo del agua o ciclo hidrológico.

El aumento de la población, el desarrollo económico, la industria, la agricultura son factores determinantes que inciden en el equilibrio del ecosistema. De esta manera el ciclo del agua o ciclo hidrológico se ve también alterado, actividades como la captación de agua que causa el desvío de ríos, el despilfarro del agua puede llegar a reducir el recurso hídrico, la deforestación, etc. Son ejemplos claros que han hecho notoria la mano del hombre en la alteración del ciclo del agua. Esta intervención del hombre cambia los factores necesarios para la vida en el agua por ejemplo en un río donde se vierte desechos industriales no puede existir vida por el grado de contaminación. Por eso es necesario que el agua que se utilizada sea devuelta al medio ambiente en condiciones de calidad y cantidad. Para eso se implementa el ciclo integral del agua. El ciclo integral del agua consiste en cumplir el ciclo hidrológico o ciclo del agua tal como se da en la naturaleza. Se trata de mantener la circulación del agua mediante el uso y devolución al medio en las mejores condiciones de calidad posibles, con el fin de que pueda seguir siendo utilizada. [1]

Entonces parte del ciclo del agua y del ciclo integral del agua es la lluvia y su impacto directo en los caudales de los ríos.
En la ciudad de Cuenca la empresa encargada del abastecimiento de agua es ETAPA EP específicamente la subdirección de Agua Potable y Alcantarillado.

1.2 Justificación

El ciclo del agua es afectado por la intervención del hombre ha causado una alteración en cada una de sus etapas y una de las etapas afectadas es la precipitación, la misma que ha variado constantemente en los últimos años no solo a nivel local sino global. Es así que últimamente se ha registrado grandes cantidades de precipitación en la ciudad de Cuenca llegando a causar grandes inundaciones y desbordes conocidas también como riadas. Los daños causados por estas inundaciones han sido considerables ya que han ocasionado desde pérdidas materiales hasta pérdidas humanas. Es así como se hace presente la necesidad de contar con un sistema de alarma basado en el registro de la lluvia.
En el año 2013 se presentaron fuertes lluvias que hicieron que los niveles del caudal en los ríos llegaran a niveles no registrados en años pasados.
El 2 febrero del 2013 se registró el desbordamiento del río Yanuncay así como el incremento de los caudales de los ríos Tomebamba, Tarqui y Machángara. El Yanuncay provocó estragos especialmente en el sector de Barabón donde se tuvo que evacuar a seis familias. El caudal del río Yanuncay alcanzó 156 metros cúbicos por segundo, cuando lo normal bordea los cinco metros cúbicos por segundo y la alerta de desbordamiento es cuando el nivel alcanza los 50 metros cúbicos por segundo. [2]

Las instituciones como el INAMHI (Instituto Nacional de Meteorología e Hidrología) y el INOCAR (Instituto Oceanográfico de la Armada) han avizorado cambios en la región sur del Ecuador desde septiembre del 2012 hasta mayo del 2013.
En vista de estos cambios climáticos la Secretaria Nacional de Gestión de Riesgos (SGNR) a través de su director provincial Galo Sánchez, especificó que hay zonas

vulnerables de inundación en la ciudad como: Barabón, Puertas del Sol, Av. Primero de Mayo, Universidad del Azuay (UDA), Tres Puentes, Max Uhle, quebrada de Salado, coliseo Jefferson Pérez, entre otras. [3]

En la Foto 1.1 podemos apreciar uno de estos lugares considerados como vulnerables.

Foto 1.1 Sector de la Universidad del Azuay, zona vulnerable. Fuente diario El Mercurio. Publicado con fecha 07/03/2013

El registro INAMHI cuenta que en enero llovió 117.4 milímetros, cuando lo normal del mes es 54.8. El porcentaje estuvo 114% por encima de su promedio. En febrero ocurrió algo similar.

En el mes de mayo se registraron intensas lluvias que llego a incrementar su nivel hasta 26% más de lo pronosticado para los meses de abril, mayo y junio. [4]

En esta imagen podemos apreciar la magnitud del torrencial aguacero que afectó a la ciudad de Cuenca el día miércoles 12 de junio del 2013.

Foto 1.2 Foto tomada del diario El Universo. Publicado con fecha 12/06/2013

La tarde del 23 de octubre del 2013 se registró otra fuerte aguacero que provocó daños materiales en la ciudad de Cuenca esta vez las zonas más afectadas fueron Ucubabamba que soportó del desbordamiento de la quebrada de Molinopamba, en la ciudadela Jardines de los Ríos en el sector de las Cuatro Esquinas también se reportaron inundaciones, así también se registró deslizamiento de piedras en la cabecera norte del aeropuerto sector del puente de Milchichig.

Foto 1.3 Subida del puente de Milchichig. Fuente: Cuenta Twitter de Silvana Estacio con fecha de 13/10/2013.

ETAPA reportó que un total de 11 viviendas fueron afectadas por las lluvias: ocho en Ucubamba y tres en Ricaurte. El personal de ETAPA y del Cuerpo de Bomberos trabajó en el lugar para solucionar el problema. [5]

Foto 1.3 Sector Ucubamba, Fuente cuenta de Twitter de Iván Rodríguez con fecha 23/10/2013.

Estos son algunos de los eventos que se suscitaron en el año 2013, la población de Cuenca necesita de algún tipo de sistema que apoye en la prevención de estas inundaciones y así poder reducir sus pérdidas materiales o de cualquier otra índole que afecte a los ciudadanos.

1.3 Concepto Ecológico

La ecología estudia la relación entre los seres vivos y su medio ambiente. Un ecosistema es la interacción entre una comunidad de organismos como (por ejemplo las plantas) y su medio físico como por ejemplo el agua, el aire, etc. La Tierra es la suma de grandes ecosistemas que contienen millones de organismos. Estos organismos o también llamados seres vivos estamos constituidos por moléculas o biomoléculas, estas pueden ser orgánicas (glúcidos, lípidos, proteínas) o inorgánicas (agua, sales minerales).

Todo ser vivo necesita biomoléculas formadas por átomos, a estos elementos se los conoce como bioelementos. Los bioelementos se dividen de acuerdo a la importancia para el organismo, tenemos: principales o primarios (carbono, hidrógeno, nitrógeno), secundarios (azufre, fósforo, magnesio, calcio, sodio potasio, cloro) y oligoelementos (cobre, cromo, selenio, cobalto, molibdeno, estaño).

Estos elementos son utilizados por los organismos para organizar sus propias biomoléculas y realizar distintas funciones. Los organismos obtienen esta materia para el funcionamiento en su medio, y posteriormente esta es devuelta al medio ambiente y puede pasar a otro organismo. La materia viaja por los ecosistemas en forma cíclica, cumpliendo de esta manera distintas funciones, dependiendo del lugar donde se encuentre localizada. A este recorrido dinámico se le conoce como *ciclo bioquímico.* Existen dos tipos de ciclos bioquímicos:

- Los ciclos gaseosos o atmosféricos (ciclo del agua, ciclo del carbono y ciclo del nitrógeno).
- Los ciclos de nutrientes sedimentarios (ciclo del fósforo y ciclo de azufre). [6]

Entonces desde el punto de vista ecológico el ciclo del agua está clasificado dentro de los ciclos bioquímicos gaseosos o atmosféricos.

El ciclo del agua consiste en cumplir el ciclo hidrológico tal como se da en la naturaleza. El ciclo hidrológico no tiene ni principio ni fin, y su descripción puede comenzar en cualquier punto. El agua que se encuentre sobre la superficie terrestre o muy cerca de ella se evapora bajo el efecto de la radiación solar y el viento. El vapor de agua, que así se forma, se eleva y se transporta por la atmósfera en forma de nubes hasta que se condensa y cae hacia la tierra, el agua precipitada puede volver a evaporarse o ser interceptada por las plantas o las construcciones, luego fluye por la superficie hasta las corrientes o se infiltra. El agua interceptada, una parte de la infiltrada y la que corre por la superficie se evapora nuevamente. De la precipitación que llega a las corrientes, una parte se infiltra y otra llega hasta los océanos y otros grandes cuerpos de agua, como presas y lagos. Del agua infiltrada, una parte es absorbida por las plantas y posteriormente es transpirada, casi en su totalidad, hacia la atmósfera y otra parte fluye bajo la superficie de la tierra hacia las corrientes, el mar u otros cuerpos de agua, o bien zonas profundas del suelo (percolación) para ser almacenada como agua subterránea y después aflorar en manantiales, ríos o el mar. [1]

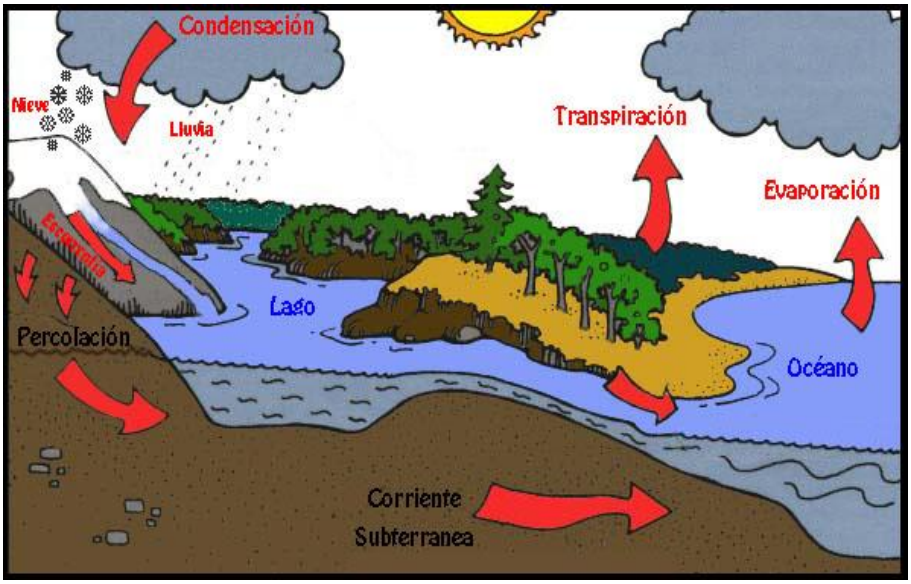


Figura 1.1 Ciclo del agua o ciclo hidrológico. Fuente: http://2.bp.blogspot.com/_RsyKBL2MDYk/ScWvDovAFpI/AAAAAAAAEig/FOGsdqFZJ48/s1600-h/Ciclo+del+Agua+1.jpg

Para simplificar también podemos ver el ciclo del agua en cuatro fases bien definidas las cuales son:

- Evaporación
- Transpiración
- Condensación
- Precipitación

1.4 Concepto humano

El hombre ha alterado el ciclo del agua de manera continua, en los últimos siglos el impacto medioambiental ha sido muy grande. Llegando a causar fenómenos tan dañinos para la Tierra como el calentamiento global. Por esta razón el ciclo integral del agua es denominado como un concepto humano puesto que el hombre de cierta manera trata de reducir el impacto de su intervención en el ciclo del agua siguiendo o cumpliendo con ciertos pasos o fases que observaremos y describiremos brevemente a continuación:

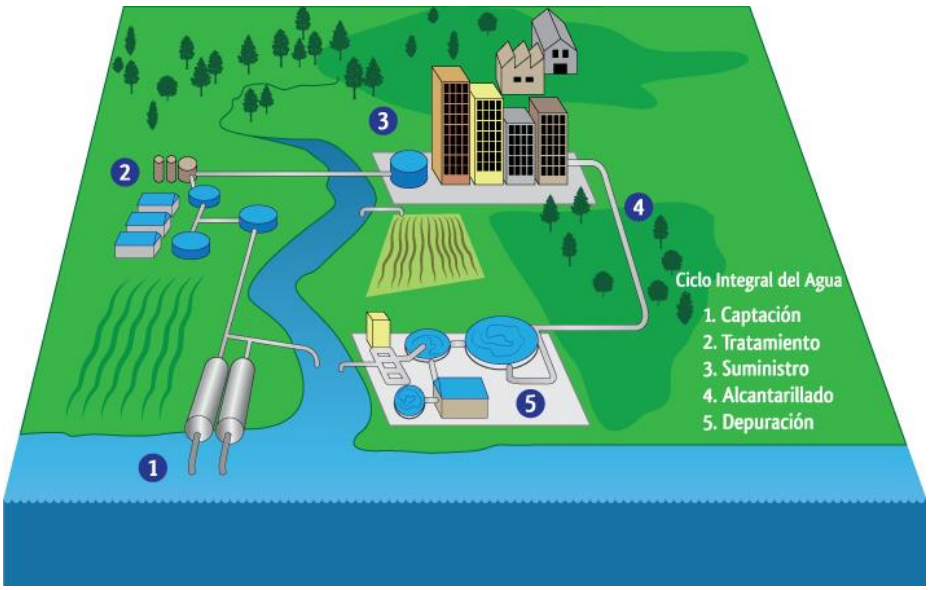


Figura 1.2 Fases del ciclo integral del agua. Fuente: Elaboración de los autores.

- **Captación.-** De fuentes superficiales, que frecuentemente suelen ser embalses y lagos, ríos.
- **Transporte.-** Luego son llevadas a estaciones de tratamiento para su potabilización.
- **Tratamiento.-** Se asegura que el agua sea apta para el consumo humano.
- **Distribución.-** A través de grandes redes de abastecimiento el agua es repartida a gran parte de la población en una ciudad.
- **Evacuación.-** Las aguas servidas son llevadas por medio del alcantarillado para el proceso de depuración y posteriormente son devueltas a los ríos o al mar.

Capítulo 2 MARCO TEORICO

2.1 Generalidades sobre Hidrología

Sobre las generalidades de Hidrología es muy importante mencionar los siguientes conceptos:

2.1.1 Hidrología

En una definición sencilla: la hidrología versa sobre el agua de la Tierra, su existencia y distribución, sus propiedades físicas y químicas, y su influencia sobre el medio ambiente, incluyendo su relación con los seres vivos. El dominio de la hidrología abarcaría la historia completa del agua sobre la Tierra. [7]

2.1.2 Definición de cuencas Hidrográficas

La cuenca hidrográfica (o también conocida como hoyas hidrográficas) es un elemento muy importante dentro de la hidrología. Entre otras definiciones tenemos: Una hoya hidrográfica es un área definida topográficamente, drenada por un curso de agua o un sistema conectado de cursos de agua, tal que todo el caudal efluente es descargado a través de una simple salida. [7]

Otra definición que nos da una idea más clara es la siguiente:

Una cuenca es una zona de la superficie terrestre en donde (si fuera impermeable) las gotas de lluvia que caen sobre ella tienden a ser drenadas por el sistema de corrientes hacia el mismo punto de salida. [1]

2.1.3 Definición de precipitación

La precipitación es el líquido que cae a la tierra desde las nubes, es decir ese cambio del estado (del estado gaseoso al estado líquido). En una definición más científica tenemos: Precipitación es, en general, el termino que se refiere a todas las formas de la humedad emanadas desde atmósfera y depositadas en la superficie terrestre, tales como lluvia, granizo, rocío, neblina, nieve o helada. [7]

La precipitación se puede dar también en forma sólida. El origen de la misma está en la formación de cristales de hielo en las nubes que tienen su tope a grandes alturas y

bajísimas temperaturas (-40ºC). Estos cristales pueden crecer a expensas de gotitas de agua a muy baja temperatura que se congelan sobre ellos (siendo el inicio de la formación del granizo) o bien uniéndose a otros cristales para formar los copos de nieve. Cuando alcanzan un tamaño adecuado y debido a la acción de la gravedad, pueden salir de la nube dando lugar a la precipitación sólida en superficie, si las condiciones ambientales son las apropiadas. A veces los copos de nieve o el granizo que salieron de la nube, si encuentran una capa de aire cálida en su caída, se derriten antes de alcanzar el suelo, dando lugar finalmente a precipitación en forma líquida. [8]

2.2 Descripción y Ubicación de los Equipos

2.2.1Sistemas de telemetría

La palabra Telemetría es el resultado de la unión de las palabras griegas Tele que significa lejos y meter significa medir. En base a esta simple traducción se puede definir a la telemetría como un procedimiento tecnológico para la obtención de datos, datos que son el resultado de mediciones de magnitudes físicas como por ejemplo la temperatura. El envío de información hacia un operador en un sistema de telemetría se realiza típicamente mediante comunicación inalámbrica, aunque también se puede realizar por otros medios. [9]

La telemetría nos permite monitorear procesos, monitoreo de seguridad, encender alarmas, visualización de eventos en tiempo real, análisis estadístico, comparativo de históricos y almacenamiento de información.

Entre muchas de las aplicaciones de la telemetría encontramos:
La telemetría es utilizada en grandes sistemas tales como las naves espaciales o las plantas químicas, debido a que facilita la monitorización automática y el registro de las mediciones, así como el envío de alertas, con el fin de que el funcionamiento sea seguro. [9]

2.2.2Estaciones Meteorológicas

Una estación meteorológica es un lugar donde se puede ubicar instrumentos que nos permitan medir las diferentes variables que pueden afectar la atmósfera.
Estos son algunos de los instrumentos de medición que encontraremos normalmente en una estación meteorológica:

- Termómetro: nos permite medir la temperatura.
- Barómetro: nos mide la presión atmosférica en superficie.
- Pluviómetro: permite medir la cantidad de precipitación.

- Psicrómetro: mide la humedad relativa del aire y la temperatura del punto de roció.
- Heliómetro: mide la intensidad de la luz
- Anemómetro: mide la velocidad del viento.

Figura 2.1 Estación Meteorológica. Fuente: http://www.ign.es/espmap/img/figuras_clima_bach/Clima_fig_01.gif

Instrumento de medida	Variable Meteorológica	Unidad de medida
Termómetro	Temperatura	ºC
Barómetro	Presión atmosférica	hPa
Pluviómetro	Precipitación	l/m²
Higrómetro	Humedad Relativa	%
Evaporímetro	Evaporación	Mm de agua evaporada
Anemómetro	Velocidad del viento	m/s – km/h
Veleta	Dirección del viento	º
Heliógrafo	Horas del sol	h
Radiómetro	Radiación	W/m²

Tabla 2.1 Variables meteorológicas con sus respectivos instrumentos y unidad de medida

2.2.3 Los Equipos

2.2.3.1 Pluviómetro

El instrumento utilizado para medir la precipitación se denomina pluviómetro. Un pluviómetro es un aparato que está formado por una especie de vaso en forma de embudo profundo que envía el agua recogida a un recipiente graduado donde se va acumulando el total de la lluvia caída. [8]

Figura 2.2 Pluviómetro. Fuente: http://4.bp.blogspot.com/_jl42kaCgJ5k/TQpzjuX97EI/AAAAAAAAAdA/WhL7H-ledWs/s1600/pluviometro.jpg

El agua de la lluvia se recoge a través de una superficie cónica y pasa a un balancín con dos cazoletas. Cae el agua hasta que se llena una de las cazoletas que tiene un volumen equivalente a 0.2 l/m² dependiendo de la precisión del pluviómetro. La estación que controla el pluviómetro acumula el número de impulsos cada cinco minutos. [10]

En este proyecto la entidad encargada de la instalación de los pluviómetros y demás equipo de recolección de datos es la empresa SensorVital. El pluviómetro específico que se instalará es el modelo TE525MM-L. El módem para la trasmisión de datos es el

RAVENXTG Celular Digital Modem, este es compatible con la red GPRS. En la figura 2.2 podemos observar un ejemplar.

Figura 2.3 Modem RavenXTG. Fuente: http://s.campbellsci.com/images/x6-3225.jpg.pagespeed.ic.hzLcCiOjkS.jpg

Las estaciones luego de su respectiva instalación las podemos apreciar de mejor manera en las siguientes fotos:

Foto 2.0.1 Estación meteorológica de Chanlud. Fuente: ETAPA EP

Foto 2.0.2 Estación meteorológica de Saucay. Fuente: ETAPA EP.

Foto 1.3 Pluviómetro de Sayausí. Fuente: Los autores.

2.2.4 Coordenadas geográficas

Las coordenadas geográficas de las estaciones meteorológicas en donde se encuentran los pluviómetros son las siguientes:

- Chanlud: -79.067402,-2.658938
- Saucay:-79.009037,-2.749989
- Sayausí: -79.068947,-2.878237

2.2.5 Ubicación geográfica

En el mapa podemos observar las tres estaciones que usaremos para nuestra tesis: la de Chanlud, Saucay y Sayausí. En cada una de estas estaciones se encuentra un pluviómetro que es el encargado de enviar la información correspondiente a la precipitación.

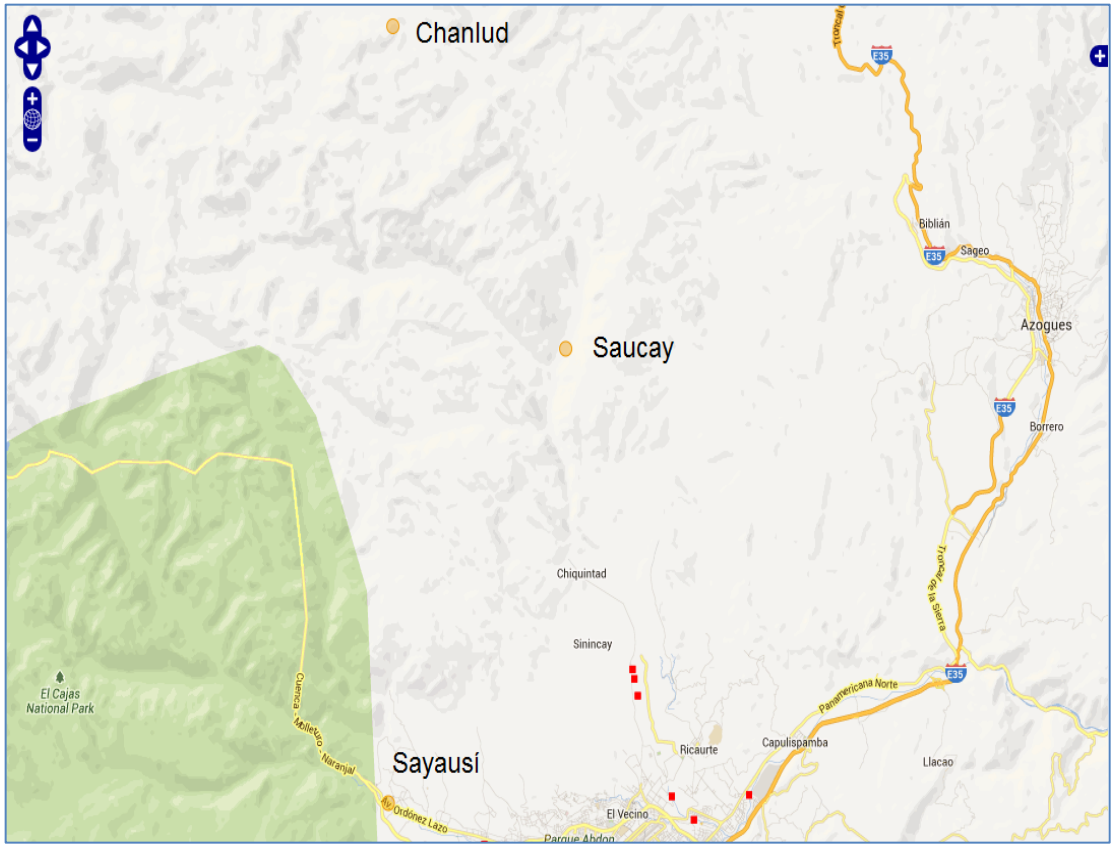


Figura 2.4 Ubicación de las estaciones. Fuente: Google Maps.

2.3 Tecnologías de Geo localización

2.3.1 GPS – Sistema Posicionamiento Global

Nació de una propuesta militar. Es un conjunto de 24 satélites que circulan la Tierra, enviando señales de radio a la superficie. Nos permite localizar la posición de un barco, persona, vehículo u otro objeto que porte un receptor GPS.

El sistema de Posicionamiento Global fue concebido en 1960 y representó la consolidación de otros proyectos para la navegación, su desarrollo se inició bajo los auspicios de la U.S. Air Force, pero fue a partir de 1974 cuando otros cuerpos militares de Estados Unidos aunaron esfuerzos y rebautizaron el proyecto con el nombre NAVSTAR Global Positioning System. Sin embargo, el nombre GPS subsistió. [11]

Se lo define también como: un sistema que tiene como objetivo la determinación de las coordenadas espaciales de puntos respecto de un sistema de referencia mundial. Los

puntos pueden estar ubicados en cualquier lugar del planeta, pueden permanecer estáticos o en movimiento y las observaciones pueden realizarse en cualquier momento del día. [12]

La Unión Europea tiene su propio sistema de ubicación por satélites denominado Galileo, así como la Federación Rusa quienes heredaron el sistema GLONASS de la extinta Unión Soviética.

2.3.1.1 Características y Componentes de GPS

El sistema G.P.S. está compuesto por:

- El segmento espacial conocido como la constelación NAVSTAR actualmente conformado por 24 satélites.
- El segmento de control conformado por estaciones de control master y de alimentación.
- El segmento usuario constituido por los receptores.

Se pueden observar claramente los tres componentes en la figura 2.5.

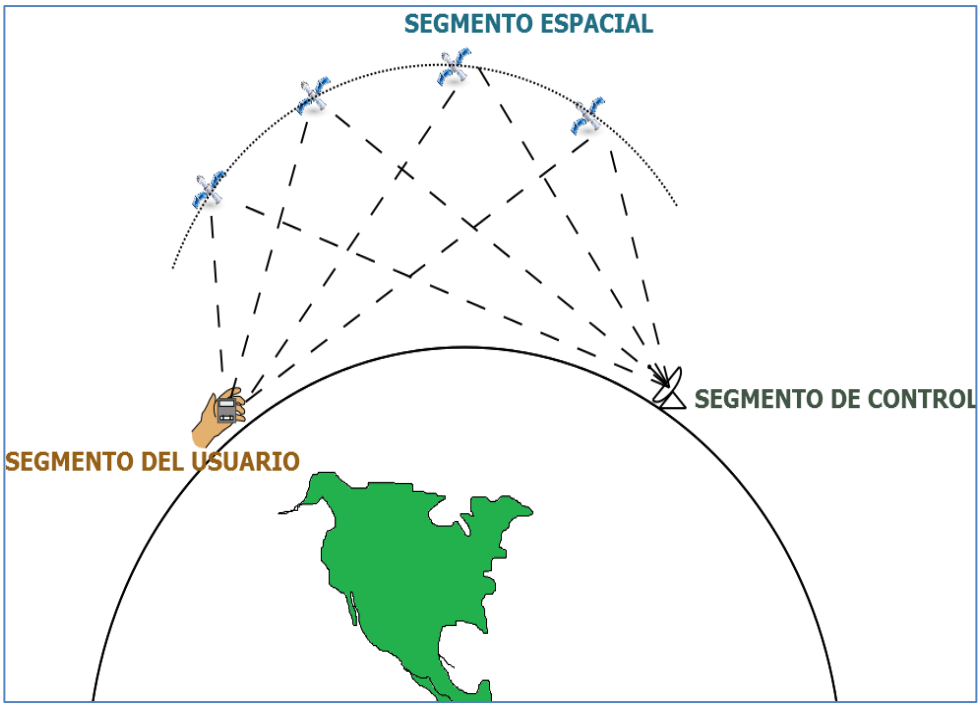


Figura 2.5 Componentes del sistema de posición global. Fuente: Elaboración de los autores.

2.3.1.2 Funcionamiento del GPS

Un receptor GPS es el que recibe las señales de los satélites. Este mismo receptor es el que calcula su posición utilizando las señales de radio emitidas por los satélites.

Los satélites tienen una trayectoria sincronizada para cubrir toda la superficie terrestre. Estos satélites que cuentan con información propia y la que reciben de las estaciones terrestres generan una señal que transmiten a los receptores GPS del usuario. Es decir la ubicación de los satélites es conocida por los receptores GPS gracias a los parámetros transmitidos por los satélites.

Cuando tenemos un objeto con un receptor GPS, lo que hace es medir su distancia a cuatro satélites de los cuales conoce su posición. La distancia *d* de cada satélite al receptor se mide en base a la siguiente ecuación:

$$d = t * c$$

Donde *t* es el tiempo que tarda la señal enviada por el satélite en llegar al receptor, y *c* es la velocidad de la luz a la que viaja la señal.

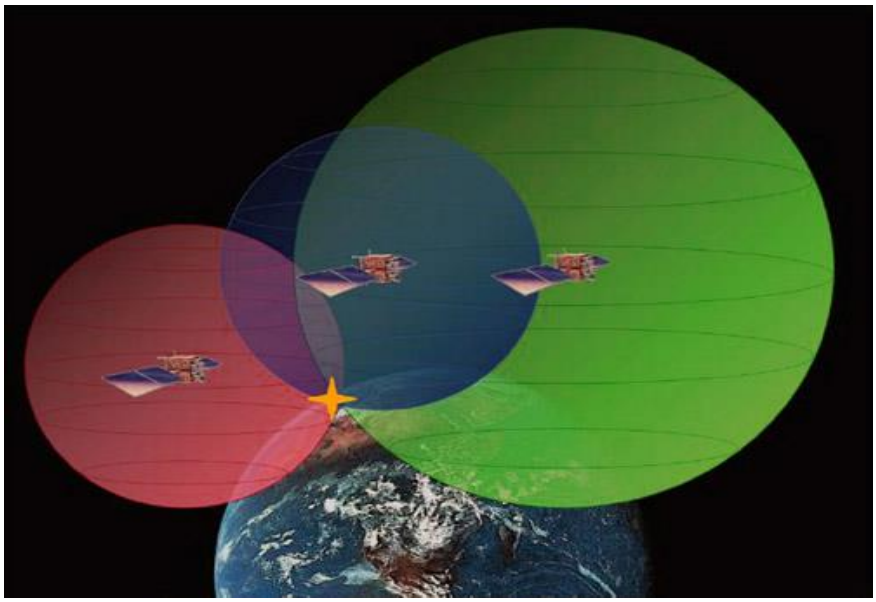


Figura 2.6 Tres satélites determinando una esfera imaginaria. Fuente: http://blogs.20minutos.es/ciencia-para-llevar-csic/files/2014/04/GPS1.jpg.

Como podemos observar en la figura 2.6, cada satélite indica que el receptor GPS se encuentra dentro de la superficie de una esfera que tiene como centro el propio satélite y de radio la distancia al mismo receptor. Con la información transmitida por los tres

satélites se busca el punto de intersección de las tres esferas como se ve claramente en la imagen (esfera roja, azul, verde).

Figura 2.7 Cuatro satélites ubicando el objeto con un receptor GPS. Fuente: http://www.brandseguridad.com/images/stories/nomoomenu1.png

Los receptores GPS deberían tener un reloj atómico como el de los satélites pero no es posible por su elevado costo, solo poseen un reloj de cuarzo. La señal de un cuarto satélite es utilizado para sincronizar los relojes satelitales (reloj atómico) y el reloj de cuarzo del receptor GPS, eliminando así el problema de la sincronización. Permitiendo así que el receptor GPS determine una posición en latitud, longitud y altitud.

2.4 Google Maps

La naturaleza dinámica de Google Maps está basada en HTML, CSS, JavaScript los cuales pueden trabajar juntos. Los mosaicos de mapas (es una combinación o fusión de dos o más imágenes) son imágenes que se cargan en background con llamadas Ajax y luego se insertan en un en la página HTML.

Al navegar en el mapa, el API envía información sobre las nuevas coordenadas y niveles de zoom del mapa en llamadas Ajax que devuelven las imágenes nuevas.

Google Maps fue la primera apuesta por un mashup SIG y hoy en día es el más extendido. [13]

Mashup es una página web o aplicación que combina datos de dos o más fuentes externas en línea. Un claro ejemplo es la página web del Departamento de Policía de Chicago que ha integrado la base de datos de crímenes reportados para combinarla con

Google Maps, donde se puede observar un mapa con los delitos denunciados determinando así áreas peligrosas.

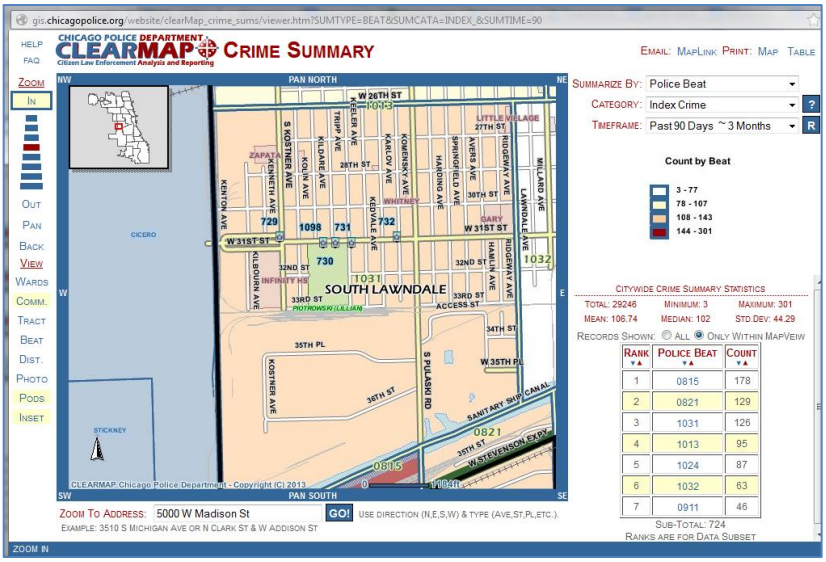


Figura 2.8 Página del Departamento de Policía de Chicago

2.4.1 API Google Maps

El API de Google Maps consiste básicamente en archivos JavaScript que contienen clases con métodos y propiedades que se pueden usar para manipular el mapa.

2.4.1.1 Como cargar Google Maps en una página web

Debemos crear un archivo HTML y una hoja de estilo (CSS). Y por supuesto un archivo de JavaSript.

Para este ejemplo se ha utilizado el IDE NetBeans 6.9.1 en donde la estructura de nuestro proyecto llamado MiMapa es la siguiente:

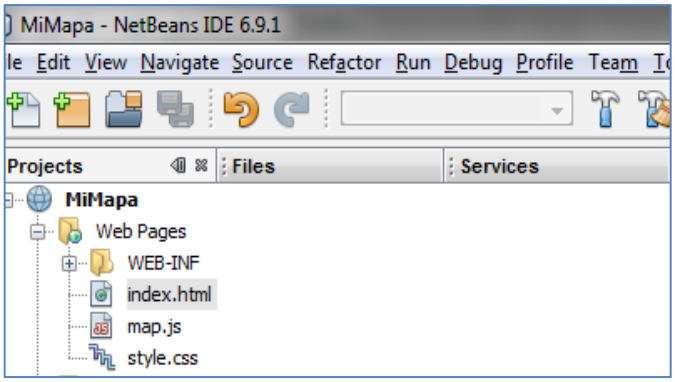


Figura 2.9 Archivos de nuestro ejemplo.

Para este ejemplo hemos creado tres archivos index.html, map.js y style.css. El código del archivo index.html se lo puede apreciar en la figura 2.10.

```
1
2  <html xmlns="http://www.w3.org/1999/xhtml">
3  <head>
4  <meta http-equiv="Content-Type"
5  content="text/html; charset=UTF-8" />
6  <title>Cuenca</title>
7  <link type="text/css" href="style.css" rel="stylesheet" media="all" />
8  <script type="text/javascript"
9  src="http://maps.google.com/maps/api/js?sensor=false"></script>
10 <script type="text/javascript" src="map.js"></script>
11 </head>
12 <body>
13 <h1>Cuenca,Azuay-Ecuador</h1>
14 <div id="map"></div>
15 </body>
16 </html>>
```

Figura 2.10 Código del archivo index.html

El API es un archivo JavaScript que está alojado en los servidores de Google. Lo hemos cargado con una etiqueta <script> en la sección <head> del archivo index.html (Ver figura 2.10, líneas 8 y 9). El elemento <script> se puede utilizar para agregar scripts remotos en una página web.

El elemento (Ver figura 2.10, linea14) contendrá el mapa y tendrá el atributo id = "map". Esto es importante, porque es a través de esta identificación que se hará referencia tanto a la hoja de estilo y al archivo JavaScript.

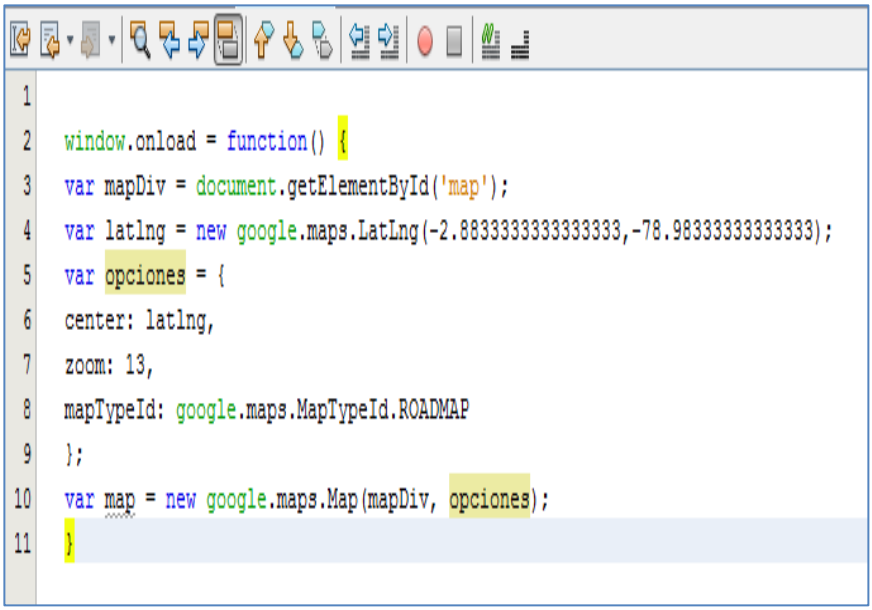
El código fuente de la hoja de estilos llamada style.css lo apreciamos en la figura 2.11.

```
7   */
8
9
10  body {
11  font-family: Verdana, Geneva, Arial, Helvetica, sans-serif;
12  font-size: small;
13  background: #fff;
14  }
15
16  #map {
17  width: 100%;
18  height: 500px;
19  border: 1px solid #000;
20  }
21
22
23
```

Figura 2.11 Código del archivo style.css

En este archivo style.css definimos las dimensiones del con los atributos width y height. El atributo width (anchura) se establece en 100% de manera que se extenderá por toda la página de lado a lado, y height (altura) se establece en 500 píxeles. Un borde de 1 píxel negro al con el atributo border (frontera).

Y finalmente tenemos el código del archivo map.js:



```
1
2   window.onload = function() {
3   var mapDiv = document.getElementById('map');
4   var latlng = new google.maps.LatLng(-2.8833333333333333,-78.98333333333333);
5   var opciones = {
6   center: latlng,
7   zoom: 13,
8   mapTypeId: google.maps.MapTypeId.ROADMAP
9   };
10  var map = new google.maps.Map(mapDiv, opciones);
11  }
```

Figura 2.12 Código del archivo map.js

Dentro del con el atributo <id=map> definido en el archivo index.html incluiremos el mapa. Lo hacemos con el método getElementById (), este método busca en el documento un elemento con el id map y devuelve una referencia a un elemento del código HTML. En la variable mapDiv se asigna la referencia al <div id='map>'. Creamos la variable opciones donde tendremos tres propiedades requeridas para que el mapa se cargue. Estas propiedades son:

- center: Los argumentos se pasan en el siguiente orden: latitud, longitud. En nuestro caso hemos puesto las coordenadas de la ciudad de Cuenca (-2.883,-78.983).
- zoom: Define el nivel de zoom inicial del mapa. Debe ser un número entre 1 y 23, donde 1 es un zoom de acercamiento máximo y 23 es un zoom de alejamiento máximo. Hemos establecido un zoom de 13.
- mapTypeId: Define el tipo de mapa que inicialmente se desea mostrar. Para obtener mapa básico, simplemente definimos google.maps.MapTypeId.ROADMAP.
- Si por el contrario se desea cargar una imagen de tipo satelital, se puede dejar como google.maps.MapTypeId.SATELLITE.

Para crear un mapa e insertarlo en la página web, es necesario inicializarlo. Esto se hace mediante la creación de una nueva instancia del objeto google.maps.Map (Figura 2.12, línea 10). Y listo el resultado es el siguiente:

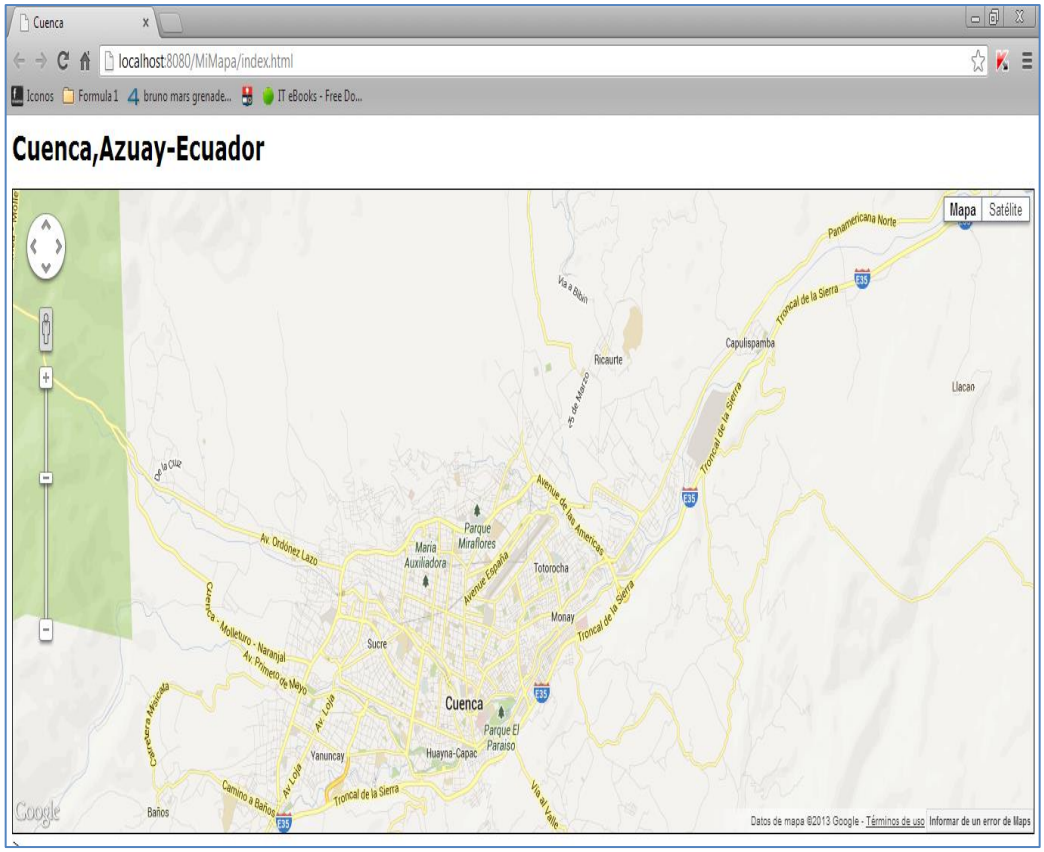


Figura 2.13 Mapa de la ciudad de Cuenca con el API de Google Maps.

2.4.1.2 Atributos y ventanas de información del mapa

2.4.1.2.1 InfoWindow

La API de Google Maps ofrece una herramienta perfecta para esto, y esa es la ventana de información (infoWindow). Se parece a una burbuja de diálogo y por lo general aparece al dar click sobre un marcador.

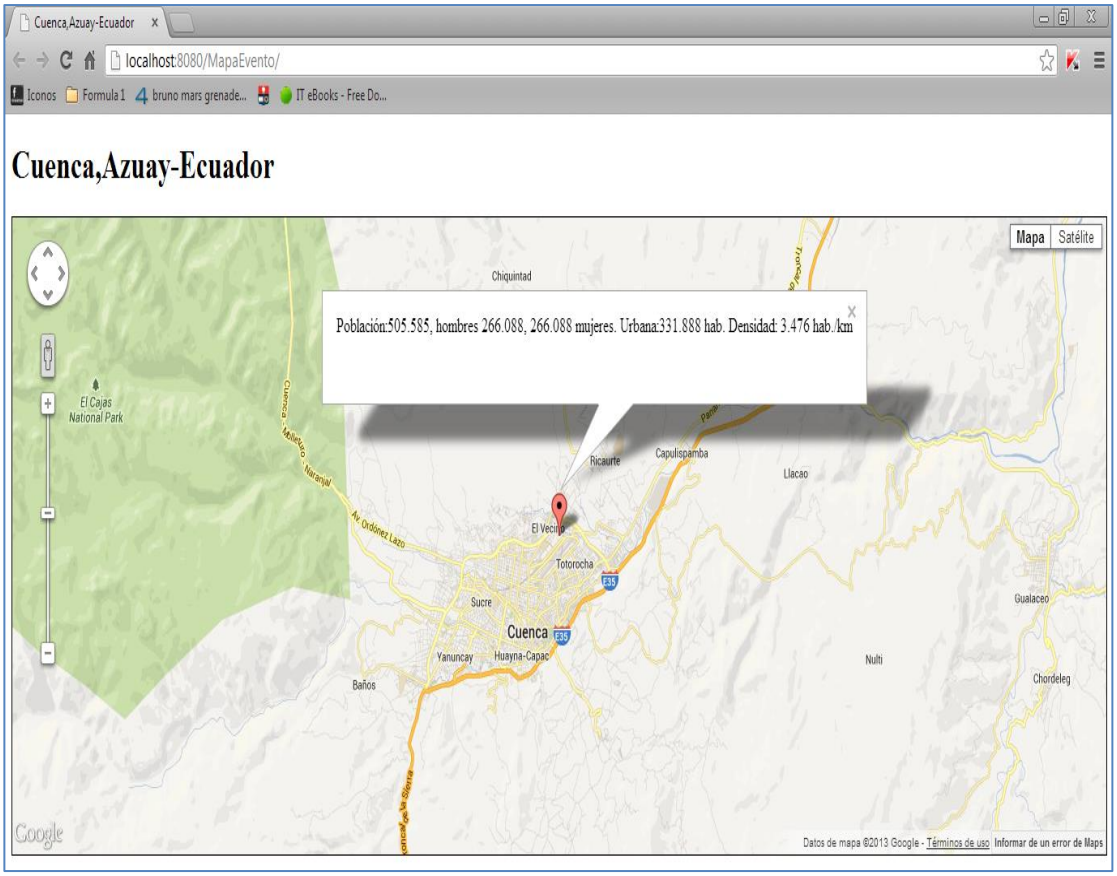


Figura 2.14 InfoWindow sencillo.

El código es muy sencillo, el mismo lo explicaremos a continuación:
Con la variable marcador de tipo marker y con las coordenadas de la ciudad de Cuenca, especificamos el mapa que creamos en anteriormente (Ver figura 2.12, líneas 1-8) y con un título cualquiera. La variable infowindow contiene un texto: *'Población: 505.585, Densidad: 3.476 ha/km'*. Con la función addListener() agregamos un evento al marcador, especificando que al hacer click en el marker éste abra el contenido de la variable infowindow.

```
(function() {
window.onload = function() {
var opciones = {
zoom: 12,
center: new google.maps.LatLng(-2.8833333333333333, -78.98333333333333),
mapTypeId: google.maps.MapTypeId.ROADMAP
};
var map = new google.maps.Map(document.getElementById('map'), opciones);

var marcador = new google.maps.Marker({
position: new google.maps.LatLng(-2.8833333333333333, -78.98333333333333),
map: map,
title: 'Haga Click'
});

var infowindow = new google.maps.InfoWindow({
content: 'Población:505.585, Densidad: 3.476 hab./km '
});

google.maps.event.addListener(
marcador, 'click', function() {infowindow.open(map, marcador);
});
};
})();
```

Figura 2.15 Código para agregar un marker y un infoWindow.

2.4.1.2.2 Eventos de la API Google Maps

Los eventos API los describiremos en una tabla, con su argumento y descripción.

Nombre de Evento	Argumento	Descripción
bounds_changed	Ninguno	Se activa cuando se cambian los límites de la ventana.
center_changed	Ninguno	Se dispara cuando el centro del mapa cambia.
click	MouseEvent	Se activa cuando el usuario da click sobre el mapa, no activa si se da click sobre un marcador o una ventana de información.
dbclick	MouseEvent	Su activación es posterior al doble click pero antes dispara el evento click.
drag	Ninguno	Se activa cuando el usuario este arrastrando el mapa.
dragend	Ninguno	Se activará cuando el usuario deje de arrastrar el mapa.
dragstart	Ninguno	Se activará cuando el usuario empieza a arrastrar

		el mapa.
idle	Ninguno	Se dispara cuando el mapa se vuelve inactivo después de panorámica y zoom.
maptypeid_changed	Ninguno	Se activa cuando la propiedad mapTypeId ha cambiado.
mousemove	MouseEvent	A medida que el usuario mueva una el puntero del ratón sobre el mapa se dispara de igual manera.
mouseout	MouseEvent	Cuando el puntero del mouse salga del contenedor del mapa.
mouseover	MouseEvent	Al entrar el puntero del mouse en el mapa se activara el evento.
projection_changed	Ninguno	Se activará cuando se cambie la proyección.
resize	Ninguno	Se dispara cuando el del mapa cambia de tamaño. Debe ser activado manualmente con esta línea: google.maps.event.trigger (mapa, 'resize')
rightclick	MouseEvent	Se activa cuando el usuario hace click derecho en el mapa.
tilesload	Ninguno	Se activa cuando los títulos visibles del mapa son cargados.
zoom_changed	Ninguno	Se activa cuando la propiedad zoom del mapa es cambiada.

Tabla 2.2 Eventos de API.

2.4.3 Controles de la API Google Maps

La interfaz de usuario consta de varios controles de usuario para controlar o supervisar el mapa, tales como el control del zoom o la escala.

2.4.3.1 ImapTypeControl

```
1
2   window.onload = function() {
3   var mapDiv = document.getElementById('map');
4   var latlng = new google.maps.LatLng(-2.8833333333333333,-78.98333333333333);
5   var opciones = {
6   center: latlng,
7   zoom: 13,
8   mapTypeId: google.maps.MapTypeId.ROADMAP,
9   mapTypeControl: false
10  };
11  var map = new google.maps.Map(mapDiv, opciones);
12  }
```

Figura 2.16 Código mapTypeControl con valor false.

Se coloca en la esquina superior derecha del mapa. Se utiliza para elegir qué tipo de mapa se quiere mostrar.

Si insertamos la línea “mapTypeControl: false”, el resultado es que la opción de cambiar a otros tipos de mapa como satelital u otro ya no aparecerá (ver figura 2.17).

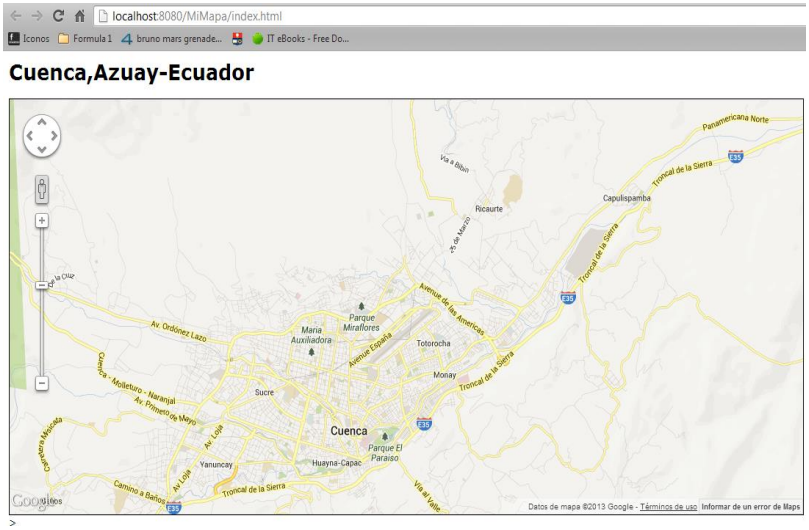


Figura 2.17 Resultado de mapTypeControl: false

2.4.3.2 navigationControlOptions

Con el navigationControlOptions se determina el aspecto del control de navegación.

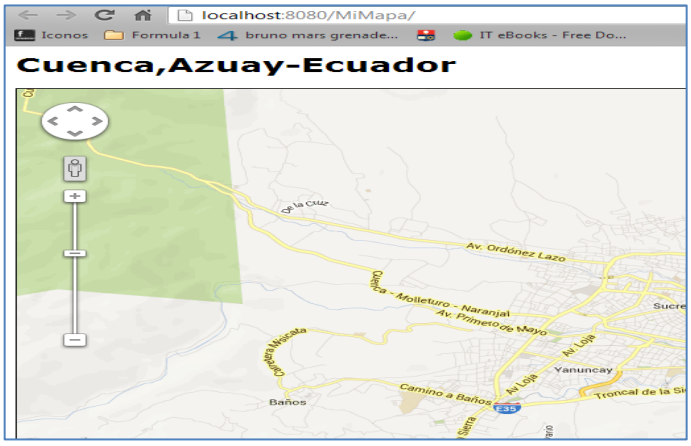


Figura 2.18 La apariencia por defecto del control de navegación

Capítulo 3 PROTOCOLO GPRS

Para hablar del Protocolo GPRS es necesario primero conocer la red GPRS, su definición, un poco de su historia y su funcionamiento.

GPRS significa General Packet Radio Service o en español Servicio General de Paquetes por Radio. Orientado a la transmisión de datos. Los proveedores de este servicio cobran por la cantidad de datos enviados o recibidos más no por un enlace permanente. Dicho en otras palabras un usuario utiliza un canal solo cuando va a transmitir datos. Luego de la rápida expansión de las redes GSM que utilizan la conmutación de circuitos para el servicio de llamadas de voz, los usuarios necesitaron servicios más avanzados como la transmisión de datos.

3.1 Red GPRS

El sistema GPRS nació para darle al usuario la oportunidad de poder transmitir datos. GPRS comparte el mismo rango de frecuencias que GSM. Repasaremos los componentes principales de una red GSM:

- **MS (Mobile Station).-**Equipo físico usado por el usuario.
- **SIM (Subscriber Identity Module).-** Es el chip identificador del abonado. Contiene información referente solo al usuario.
- **BTS (Base Transceiver Station).-** Establece vía radio conectividad entre las estaciones móviles y la red GSM.
- **BSC (Base Station Controller).-** Actúa como un concentrador de varias estaciones base (BTS).
- **BSS (Base Station Subsystem).-** Está constituido por la BTS y la BSC.
- **MSC (Mobile Switching Center).-** Centro de Conmutación de servicios móviles. Control de llamadas y encargado de la asignación de canales de usuario entre el MSC y el BSC.
- **GMSC (Gateway MSC).-** Es a través del cual se conecta la red GSM con las redes fijas.
- **HLR (Home Location Register).-** base de datos que contiene los datos de los clientes para su gestión. Contiene información como: servicios contratados, limitaciones de servicio y localización del usuario.

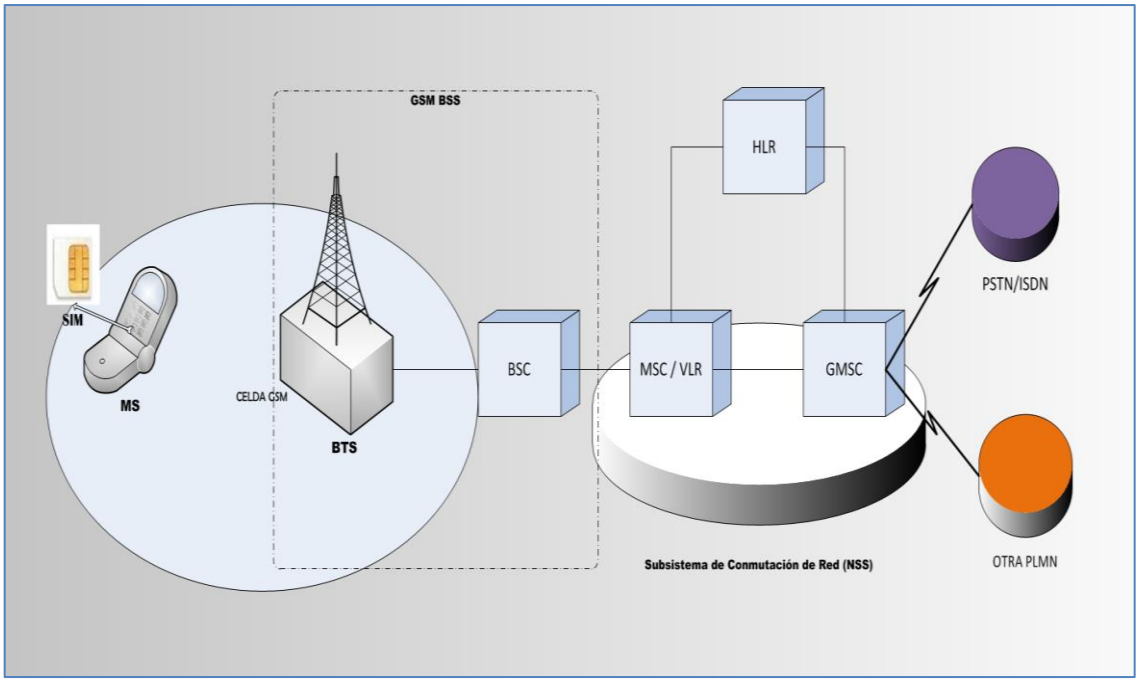


Figura 3.1 Arquitectura de la red GSM. Fuente: Elaboración de los autores.

- **VLR (Visitor Location Register).-** Contiene datos dinámicos que permiten saber la última ubicación del usuario.

La red GPRS agrega algunos componentes a la red GSM tales como el SGSN (responsable de la transferencia de paquetes) y el GGSN (convierte los paquetes al formato IP) que no estaban incluidos en un principio en la red GSM como podemos apreciar en la Figura 3.1. La inclusión de estos nuevos componentes permite la conmutación de paquetes y la utilización de protocolos como IP.

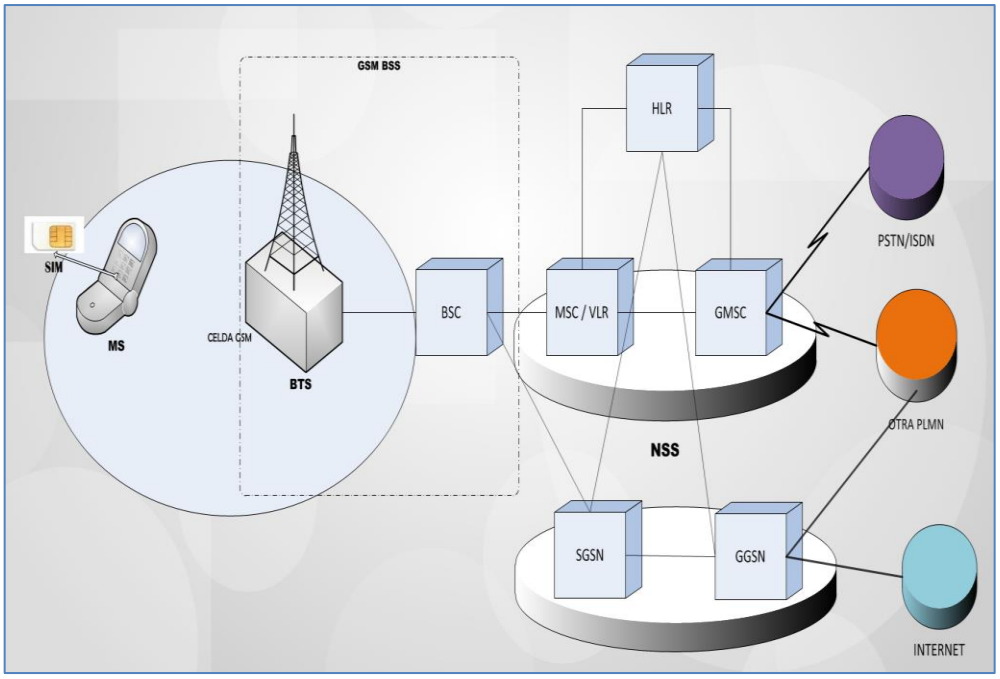


Figura 3.2 Elementos que agrega GPRS. Fuente: Elaboración de los autores.

3.2 Componentes de la Arquitectura de una red GPRS

La utilización de la infraestructura GSM no es suficiente para poder transmitir paquetes de datos a velocidades que están en el rango de 9.6 y 171 Kbps. Es necesario incluir nuevos equipos especializados para esta tarea. Para una mejor apreciación de la arquitectura de una red GPRS podemos observar la Figura 3.2.

La estructura de GSM se ha modificado hasta encontrar un modo de transferencia de conmutación de paquetes de extremo a extremo y los encargados de esta función son los nodos de soporte del servicio denominados GSN (Gateway Support Node).

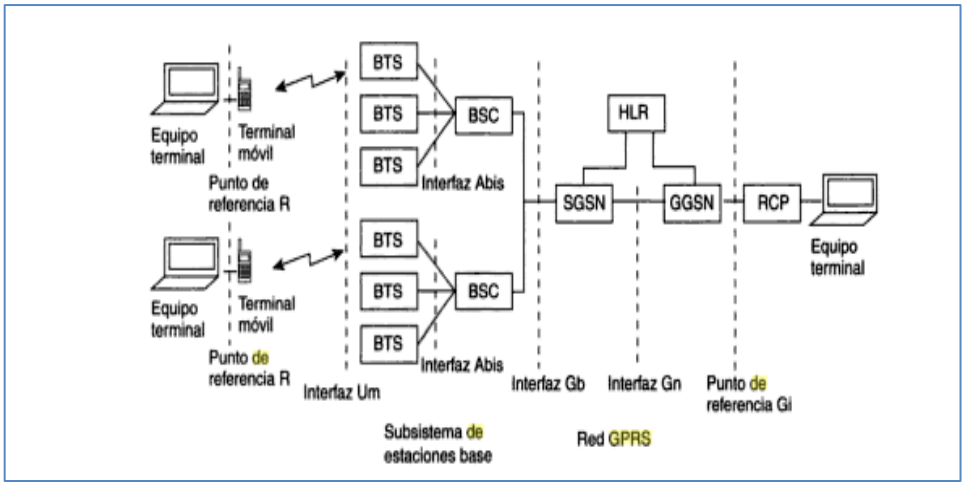


Figura 3.3 Arquitectura del sistema GPRS, Fuente: imagen tomada del libro Servicios Avanzados de Telecomunicación.

Los elementos que a continuación describiremos ayudan a encaminar los paquetes de datos desde y hacia las estaciones móviles:

- **SGSN.-** Nodo de soporte servidor de GPRS (Serving GPRS Support Node), responsable de la transferencia de los paquetes desde y hacia los móviles en su área de servicio. Gestiona la movilidad la autenticación y cifrado. Así como también se encarga de recopilar toda la información necesaria para la facturación.
- **GGSN.-** Nodo de soporte de pasarela de GPRS (Gateway GPRS Support Node), actúa como interfaz lógica entre la red GPRS y las redes públicas de datos (PDN) externas que pueden ser IP o X.25.

Tanto el SGSN Y GGSN son parte del GSN. Todos los nodos GGSN se conectan a través de una troncal GPRS (backbone network) basada en IP. Se diferencian dos clases de troncales: Intra-PLMN GPRS Backbone (Red Troncal GPRS) e Inter-PLMN GPRS Backbone (Inter Operador GPRS). Como podemos observar en la Figura 3.4 la troncal Intra-PLMN permite la conexión entre los GNS de una misma operadora. Mientras que la Inter PLMN servirá para conectar redes troncales GPRS O Intra PLMN de distintas operadoras.

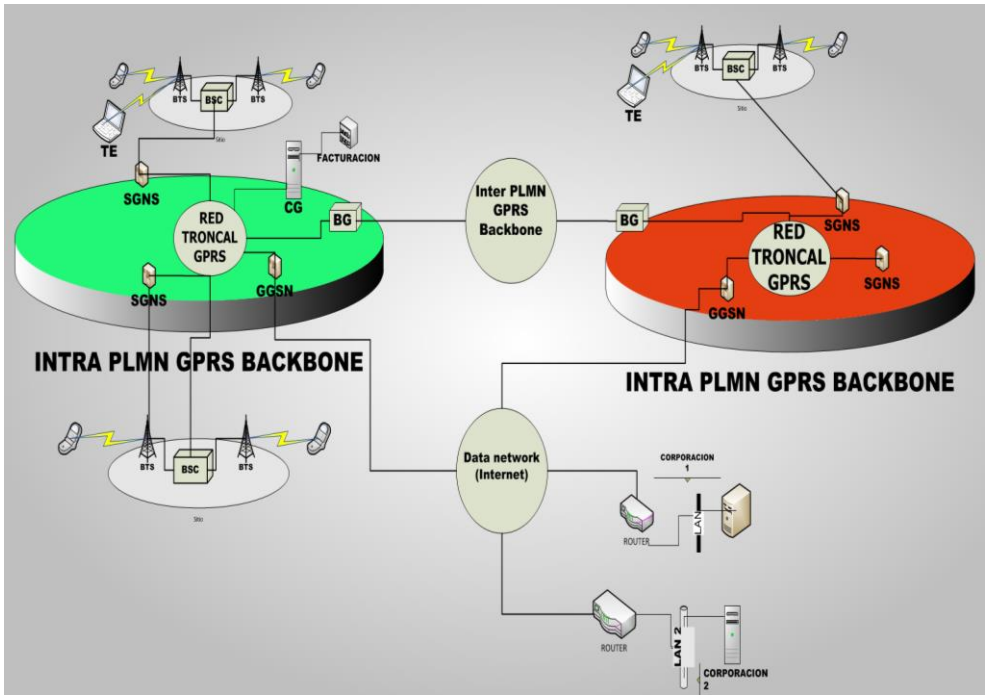


Figura 3.4 Troncales de la Red GPRS. Fuente: Fabián Tacuri.

En la misma Figura 3.4 podemos observar el elemento **BG** (Border Gateway) esta encargado de proporcionar una conexión directa entre una Intra PLMN y una Inter PLMN brindando protección a la Intra PLMN de posibles ataques. El **CG** (Charing Gateway) cumple con la función de recoger los CDR (Call Detailed Records) generados por el SGSN y GGSN para ser luego enviados al sistema de facturación de la operadora.

El término equipo terminal *(TE-TerminalEquipment)* es usado para referirse a una amplia variedad de teléfonos móviles y de estaciones móviles usados en el ambiente GPRS. Existen tres tipos de terminales: Terminales clase A que soportan servicios GPRS y GSM de forma simultánea, terminales clase B, que pueden monitorear canales GSM y GPRS simultáneamente pero que pueden soportar únicamente uno de estos servicios a la vez, y terminales clase C que soportan únicamente un servicio. [14]

3.2.1Interfaces

- **Um.**- Es una interface de radio ubicada entre la estación móvil (MB) y la estación base (BS).
- **Gn.**- Encargado de la transmisión de datos entre la SGNS Y GGSN a través de la troncal GPRS. Aquí se lleva a cabo el GTP (GPRS Tunnelling Protocol) entre los nodos GGSN para llevar datos de los usuarios.
- **Gb.**- conecta los BSC y los SGSN.

- **Gi.-** Conecta la red GPRS con redes de datos externa. Como se observa en la Figura 3.4 lo hace a mediante el GGSN.
- **Gs.-** Utilizada entre el MSC/Registro de Lugares Visitantes (RLV) y el SGSN.

3.3 Protocolo GPRS

El protocolo GPRS es un protocolo de nivel tres, transparente para todas las entidades de red comprendidas entre el terminal móvil MT y el nodo GSN al que el móvil está lógicamente conectado; las entidades entre las que se establece una conexión a este nivel están, de hecho, localizadas en el terminal móvil MT y en el nodo GSN. Este protocolo soporta tanto el intercambio de información de control como de paquetes PDP-PDU (Packet Data Protocol - Protocol Data Unit) entre el móvil y el nodo al que éste está conectado (los PDP-PDU son, de hecho, encapsulados en las tramas GPRS). [15]

3.4 Ventajas y beneficios de GPRS

Entre las ventajas de utilizar GPRS están [16]:

- Al utilizar conmutación de paquetes la tarifa está en relación al tráfico consumido.
- Trabaja sobre la misma estructura de GSM.
- Permite que GMS se pueda comunicar con redes LAN, WAN e Internet.
- Tasas de transmisión hasta de 144kbps.
- Los datos son enviados al mismo tiempo que se puede realizar una llamada.
- Modo de trasmisión asimétrico.

3.5 Desventajas y Limitaciones de GPRS

Las desventajas [16]:

- La cobertura dependerá de la extensión de la red GSM, si en algún lugar no se cuenta con celdas GSM será imposible tener terminales GPRS.
- La velocidad se verá afectada dependiendo del número de usuarios que tenga una celda GSM. Pues GPRS utiliza los mismos canales de GSM, las operadoras no habilitarán todos los canales de una celda solo para GPRS sino solo un porcentaje.

Capítulo 4 TECNOLOGÍA ARDUINO

4.1 Introducción

La tecnología Arduino está incluida dentro de lo que se denomina hardware libre que al igual que el software libre tiene una licencia GPL, sus diagramas y especificaciones son de acceso libre.

Massino Banzi, italiano cofundador del proyecto electrónico Ardunino lo llamó así en honor a un bar ubicado en Ivrea, Italia. [17]

4.2 Concepto

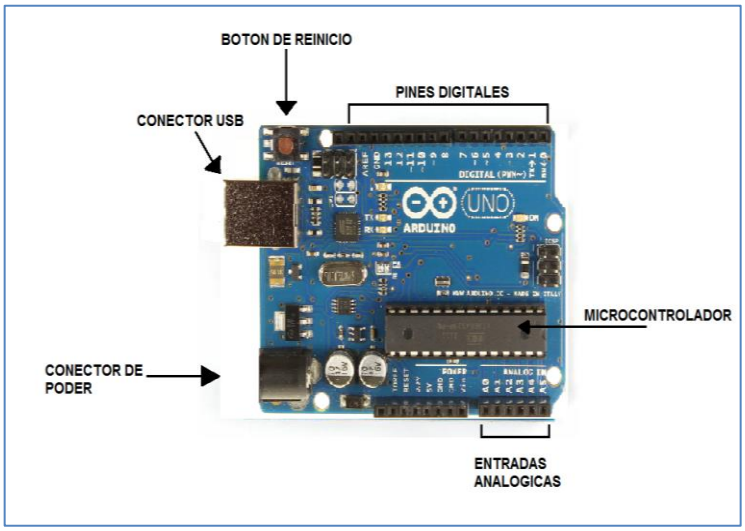


Figura 4.1 Placa Arduino Uno. Imagen tomada de http://arduino.cc

Arduino es una plataforma de electrónica abierta para la creación de prototipos basada en software y hardware flexibles y fáciles de usar. [18] En la figura 4.1 podemos apreciar una placa Arduino Uno, se señala algunas de las principales partes de esta placa. El microcontrolador en la placa Arduino se programa mediante el lenguaje de programación Arduino (basado en Wiring) y el entorno de desarrollo Arduino (basado en Processing). El software incluye librerías de acceso a la placa. [18]

Wiring permite escribir software multiplataforma para controlar dispositivos mediante un microcontrolador.

Processing es un código abierto, enfocado a proyectos multimedia e interactivos. Destacando su manejo de gráficos e imágenes.

Su funcionamiento básico consiste en recibir información de un entorno a través de sus entradas y realizar una acción por medio de sus pines de salida. Por ejemplo se puede tener conectado un sensor de humo a uno de sus pines de entrada al recibir la señal que

indique la presencia de humo por medio de uno de sus pines de salida se puede mandar a activar una alarma.

La placa que utilizaremos en nuestro proyecto es el Arduino Uno R3. Cuyas especificaciones hemos descargado desde la página oficial de Arduino. (Ver tabla 4.1)

Micro controlador	ATmega328
Voltaje de funcionamiento	5V
Voltaje de entrada recomendado	7-12V
Voltaje de entrada limite	6-20V
Pines E/S digitales	14 (6 como salida PWM)
Pines de entrada analógicos	6
Intensidad por pin	40 mA
Intensidad en pin 3.3V	50 mA
Memoria Flash	32 KB (ATmega328)
SRAM	2 KB (ATmega328)
EEPROM	1 KB (ATmega328)
Velocidad de Reloj	16 MHz

Tabla 4.1 Especificaciones de placa Arduino Uno R3.

4.3 Código abierto

Al profundizar en el mundo Arduino podemos afirmar que esta tecnología engloba tres herramientas específicas.

- La primera es el controlador de Arduino que existe en varios tamaños desde pequeños hasta grandes. Cuenta con su esquema de libre acceso. Cualquier persona con los conocimientos necesarios podría ensamblar este controlador.
- En segundo lugar tenemos el lenguaje y el compilador que crea código para el controlador que simplifica muchas de las tareas que se presentan como un desafío para los diseñadores y programadores cuando trabajan con un hardware e interacción física.
- Y por último tenemos el entorno de programación (IDE), es de código abierto desarrollado en Java.

4.4 Multiplataforma

Arduino utiliza librerías de código abierto, escritas en C y C++ y compiladas para diferentes sistemas operativos. Además el entorno de desarrollo está escrito en Java lo que permite tener un software multiplataforma. [19]

El IDE de Arduino puede ser instalado en los sistemas operativos Windows, en MAC y Linux.

4.5 Entorno

Para instalar Arduino descargamos el instalador de la página web oficial de Arduino http://arduino.cc/en/Main/Software. Se nos descargara el archivo ejecutable arduino-1.0.5-windows.exe. Como pre requisito debemos tener instalado el JDK de Java en nuestra computadora las pantallas de instalación son sencillas como podremos apreciar en las figuras. 4.2, 4.3, 4.4 finalmente la figura 4.5 nos muestra el entorno de desarrollo de Arduino.

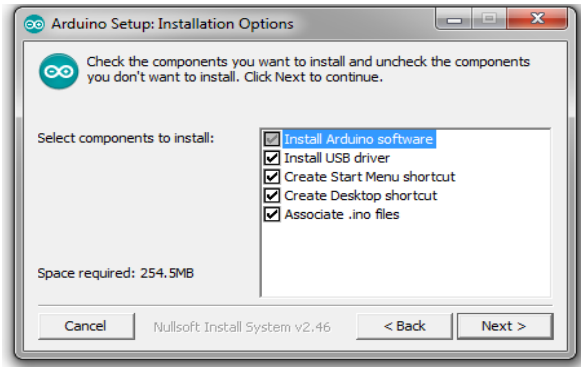


Figura 4.2 Proceso de instalación de IDE para Arduino.

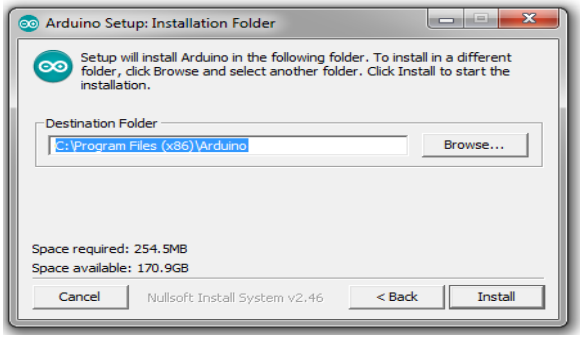


Figura 4.3 Proceso de instalación de IDE para Arduino.

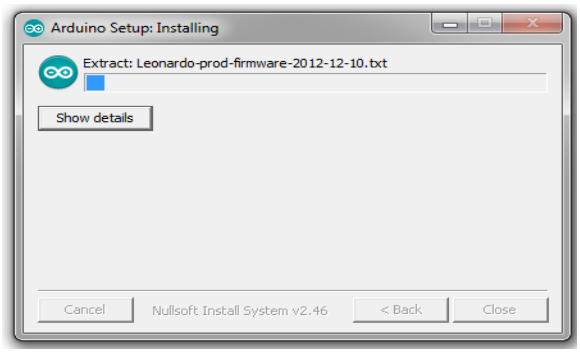


Figura 4.4 Proceso de instalación de IDE para Arduino.

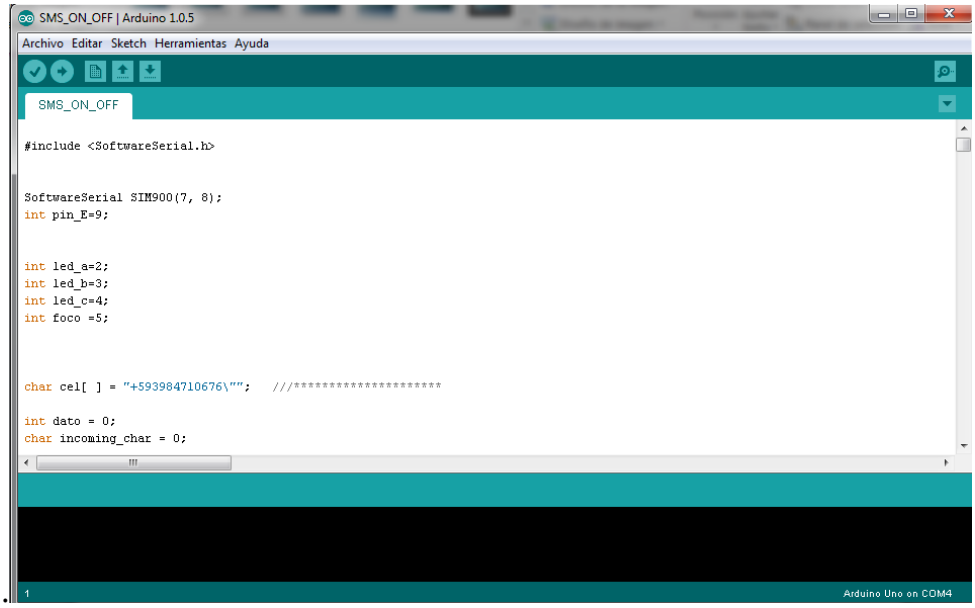


Figura 4.5 IDE de Arduino ejecutado.

4.6 Estructura

4.6.1 Declaración Setup()

Se ejecuta una sola vez al conectar o resetear la placa.

4.6.2 Declaracion loop()

Contiene todo lo que usted necesita hacer en repetidas ocasiones dentro de la aplicación, como por ejemplo chequear un nuevo valor desde un control, enviar información a la computadora, enviar una señal a un pin.

4.6.3 Constantes

Arduino tiene las siguientes constantes que pueden ser utilizadas en cualquier parte del código de una aplicación:

- **HIGH/LOW.-** Estas definen el nivel de voltaje en un pin digital, pudiendo ser 5V ó 0V.

- **INPUT/OUTPUT.-** Estas son constantes que son usadas para definir un pin, entrada o salida.

4.6.4 Métodos

- **pinMode(número_de_pin, mode)**
  Los pines digitales del controlador Arduino pueden ser definidos como entrada (INPUT) o salida (OUTPUT). Debemos establecer si será de entrada o de salida en el setup (). Generalmente solo lo hacemos una vez en la aplicación. Por ejemplo: pinMode (11, OUTPUT) en este ejemplo el pin 11 está puesto como salida, todo esto debemos declararlo dentro de setup ().

- **digitalWrite(valor, pin)**
  Este método coloca un pin digital en HIGH o en LOW dependiendo del uso que le daremos. Para poder ser colocados estos valores primero deben haber sido definidos en el método pinMode (). Por ejemplo:
  void setup().
  {

  pinMode(2, OUTPUT);

  digitalWrite(2, LOW);

  }

- **int digitalRead(numero_de_pin)**
  Este es un método que lee el estado de un pin que está en modo INPUT. Retorna el valor del pin como un integer (valor entero).

- **analogRead(numero_de_pin)**
  Este método lee el valor de un pin analógico. Un pin analógico puede tener un valor desde 0 a 1023 que representa el voltaje del pin. El valor retornado será un

integer. Por ejemplo podemos declarar un int y recibir lo que devuelva el método.
int pin_11=analogRead(11);

4.7 Sintaxis

La página oficial de Arduino nos da como referencia de su sintaxis unos cuatro ejemplos.

- **;(punto y coma).-** Utilizado para terminar una declaración.
- **{} (llaves).-** Una llave de apertura "{" siempre debe ir seguida de una llave de cierre "}". Esta es una condición a la que se suele referir como llaves emparejadas. El IDE (Entorno Integrado de Desarrollo) Arduino incluye una característica para comprobar si las llaves están emparejadas. Sólo tienes que seleccionar una Llave o incluso hacer click en el punto de inserción que sigue inmediatamente a una llave, y su compañera lógica será seleccionada. En la actualidad esta característica tiene un pequeño fallo, el IDE encuentra a menudo (incorrectamente), llaves en el texto que pueden estar situadas dentro de comentarios. [18]
- **// (comentarios en una línea).-** Los comentarios son líneas en el programa para aclarar sobre el funcionamiento del programa. Estas líneas son ignoradas por el compilador y no se exportan al procesador. [18]
- **/**/ (comentarios en múltiples líneas).-** Los comentarios son líneas en el programa para aclaraciones sobre el funcionamiento del programa. Estas líneas son ignoradas por el compilador y no se exportan al procesador.

Para resumir a continuación la tabla 4.1 nos muestra operadores aritméticos, operadores comparativos, tipos de datos con los que se puede trabajar en el IDE de Arduino.

Tipos de datos	Operadores Aritméticos	Operadores comparativos
boolean (booleano)	= (asignación)	== (igual a)
char (carácter)	+ (suma)	!= (distinto de)
byte	- (resta)	< (menor que)

int (entero) unsigned int (entero sin signo) long (entero 32b) unsigned long (entero 32b sin signo) float (en coma flotante) double (en coma flotante de 32b) string (cadena de caracteres) array (cadena) void (vacío)	* (multiplicación) / (división) % (resto)	> (mayor que) <= (menor o igual que) >= (mayor o igual que)

Tabla 4.2 Sintaxis de lenguaje de programación de Arduino.

4.8 GPRS/GSM Shield

El GPRS/GSM Shield provee una conexión para usar la red celular GSM. Con este Shield o Escudo en español, se puede recibir o enviar datos desde un lugar remoto, obviamente el lugar debe tener cobertura del operador GSM que contratemos.

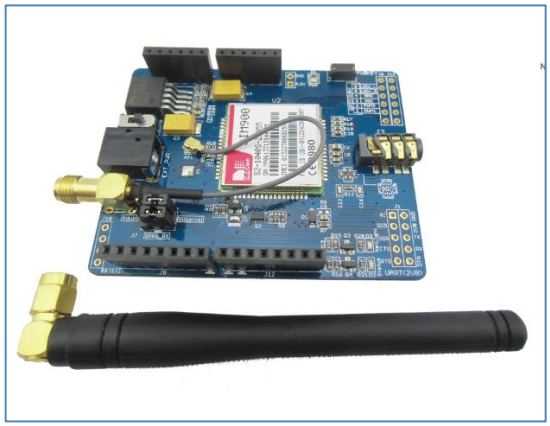


Figura 4.6 Imagen de un GPRS/GSM SHIELD SIM900.

En la figura 4.6 podemos apreciar el modelo del GPRS/GSM SIM900 que utilizaremos para nuestro proyecto. Este módulo nos permitirá enviar y recibir mensajes con ciertas instrucciones para nuestro prototipo de alarma basada en Arduino. Estos escudos interactúan con la placa de Arduino permitiendo de esta manera manejar las mismas salidas y entradas que tiene la placa Arduino.

Capítulo 5 ANÁLISIS Y DISIEÑO

5.1 Análisis

5.1.1 Requerimientos

5.1.1.1 Introducción

Los requerimientos fueron tomados con las autoridades de ETAPA-EP entre ellas el Ing. Fausto Sarmiento, Administrador de Agua Potable, el Ing. Josué Larriva Supervisor de Soporte, Investigación y desarrollo, el Ing. Javier Fernández de Córdova Subgerente de operaciones de agua potable y saneamiento. El tiempo que se tomó en levantar estos requerimientos fue alrededor de 4 meses (mayo, junio, julio, agosto del 2013), dentro de los cuales se llevó a cabo reuniones personales. También se tomó un curso de Data Warehousing en el CIISCA (Colegio de Ingenieros en Informática, Sistemas y Computación del Azuay) que tenía como fin repasar y reforzar conceptos en Almacenes de Datos y Minería de Datos.

5.1.1.1.1 Propósito

Este capítulo recoge los requerimientos obtenidos para poder establecer una línea base desde donde partir con el desarrollo de la aplicación. Esta línea base se ajustará a los requerimientos recogidos.

5.1.1.1.2 Alcance

El proyecto es nombrado como **PluSoftEP** (Software de Pluviómetros de ETAPA).

El sistema interactúa con una base de datos desde donde se toman valores de las estaciones pluviométricas de: Chanlud, Saucay y Sayausí. Luego a partir de procesos se definirá si el acumulado de niveles de lluvia son peligrosos en determinadas zonas sensibles a inundaciones, que están definidas para las riveras de los ríos Tomebamba y Machángara.

5.1.1.1.3 Definiciones, Siglas y Abreviaturas

PluSoftEP.- Software de Pluviómetros de ETAPA

ETAPA-EP.-EMPRESA PÚBLICA MUNICIPAL DE TELECOMUNICACIONES, AGUA POTABLE, ALCANTARILLADO Y SANEAMIENTO DE CUENCA.

Password.- Clave para autentificar el ingreso al sistema.

RF.- Requerimiento funcional.

5.1.1.1.4 Visión General

El documento está dividido en dos partes la primera trata de todo lo referente a la especificación de requerimientos y la segunda parte hace referencia al diseño de la aplicación.

5.1.1.2 Descripción General

5.1.1.2.1 Perspectiva

Se realizará un sistema para que obtenga datos de las estaciones pluviométricas de la empresa ETAPA-EP. A partir de estos datos realizará un proceso en el cual se determinará si los niveles de lluvia son peligrosos para ciertas riveras de ríos denominados zonas susceptibles.

5.1.1.2.2 Funcionalidades

En una breve descripción el sistema deberá realizar tareas como:

Análisis y filtrado de datos: Los datos que lleguen a nuestra base tendrán un formato específico. De este formato tomaremos información necesaria para llenar las tablas de nuestro sistema.

Gestión de Estaciones pluviométricas: El usuario podrá buscar y listar las estaciones que necesite revisar.

Gestión de Alarmas: el usuario recibirá notificaciones de cuando una alarma se haya activado. Se mostrarán en un mapa los puntos sensibles por medio de una animación gráfica.

Generar reportes: Cada estación tiene un histórico. Este histórico se generará de acuerdo a los parámetros que solicite el usuario. También se visualizará un gráfico de los históricos.

5.1.1.2.3 Características de los Usuarios

Tipo de usuario	Administrador.
Formación	Universitario con título de tercer nivel.
Habilidades	Conocimientos avanzados en sistemas informáticos.
Actividades	Podrá tener acceso a los reportes del registro historio del registro de la precipitación de lluvia.

Tipo de usuario	Común.
Formación	Universitario con título de tercer nivel.
Habilidades	Conocimientos avanzados de sistemas informáticos.
Actividades	Podrá visualizar el mapa con los puntos sensibles a inundaciones. Así como la ubicación de las estaciones pluviométricas.

5.1.1.2.4 Restricciones u/o Condiciones

El sistema estará basado en roles donde cada usuario tendrá definido su nivel de acceso a la información.

5.1.1.2.5 Suposiciones y Dependencias

El sistema se desarrollará con varios frameworks como Spring. La base de datos será de licencia GPL.

5.1.1.2.6 Requisitos Futuros

Para futuras versiones se puede implementar una interfaz web para que el usuario pueda controlar las alarmas.

5.1.1.3 Requerimientos Específicos

Los requerimientos funcionales están basados en lo que se necesita que haga el sistema y lo que los usuarios desean obtener. Así mismo se especifica lo que se necesita que se muestre en la interfaz Web.

- Mostrar un mapa con la ciudad de Cuenca que permita ubicar las estaciones pluviométricas.
- Visualización en un mapa con las zonas de riesgo de inundación en la ciudad de Cuenca y aledañas al río Tomebamba y al río Machángara.
- Al ver el mapa de la ciudad de Cuenca con la ubicación de las tres estaciones pluviométricas: Chanlud, Saucay y Sayausí. El usuario podrá dar click sobre una de ellas y podrá observar el histórico del registro de lluvia.
- Gráficas y reportes de los históricos.

5.1.1.3.1 Requerimientos Funcionales

RF-1	Loggin en el sistema		
Versión	2.0	Fecha	19/06/2013
Autores	Juan Carlos Amay		
Fuentes	Administrador		
Actor	Administrador		
Descripción	Permite ingresar al sistema		
Precondición	Usuario y Password		
Secuencia normal	1.- El administrador digita usuario y contraseña. 2.- El sistema validad datos y le da acceso al administrador.		
Post Condición	Se mostrará la página de administrador		
Excepciones	Ninguna		
Prioridad	Alta		
Estado	Implementado		
Estabilidad	Alta		
Comentarios	No		

RF-2	Ver mapa con estaciones		
Versión	2.0	Fecha	19/19/2013
Autores	Fabián Tacuri Puma		
Fuentes	Administrador, usuario común		

Actor	Administrador
Descripción	Permite visualizar la ubicación de los pluviómetros
Precondición	El usuario debe estar logueado
Secuencia normal	1.- El administrador/usuario debe entrar en el menú de mapas. 2.- El sistema ejecuta una ventana donde se visualiza el mapa con las estaciones. 3.- El administrador/usuario podrá dar click sobre una de estas estaciones y ver alguna referencia.
Post Condición	Ninguna
Excepciones	Ninguna
Prioridad	Alta
Estado	Implementado
Estabilidad	Alta
Comentarios	No

RF-3	Ver alarmas gráficas en el mapa		
Versión	2.0	Fecha	19/06/2013
Autores	Fabián Tacuri Puma		
Fuentes	Administrador, usuario común		
Actor	Administrador		
Descripción	Permite visualizar si un evento activó la alarma en las zonas vulnerables a inundaciones. Naranja significa caudal del río ha sufrido algún cambio. Amarillo se ha percibido un aumento en el caudal. Rojo en la zona susceptible se puede producir un desborde el río.		
Precondición	La base de datos debe haber sufrido un cambio.		
Secuencia normal	1.- El sistema presenta para el administrador en la ventana del mapa las zonas vulnerables a inundaciones con color rojo, naranja o amarillo. 2.- El administrador recibirá una notificación en el celular con		

	información sobre el tipo de alarma que se produjo.
Post Condición	Ninguna
Excepciones	Ninguna
Prioridad	Alta
Estado	Implementado
Estabilidad	Alta
Comentarios	No

RF-4	Recibir notificación de alarma vía sms		
Versión	2.0	Fecha	12/08/2013
Autores	Fabián Tacuri Puma, Juan Carlos Amay		
Fuentes	Administrador, usuario común		
Actor	Administrador		
Descripción	Un dispositivo de alarma se activará en las zonas susceptibles a inundaciones.		
Precondición	Los niveles de caudal han sobrepasado los límites permitidos.		
Secuencia normal	1.- El sistema envía una señal al dispositivo que se encuentra en la zona susceptible a inundaciones. 2.- Se activará una alarma en la zona afectada. 3.- Un mensaje llegará al celular del administrador del sistema. 4.- El usuario podrá apagar la alarma luego de que sea atendida.		
Post Condición	Ninguna		
Excepciones	Ninguna		
Prioridad	Alta		
Estado	Implementado		
Estabilidad	Alta		
Comentarios	No		

5.1.1.3.2 Requerimientos No Funcionales

- **Requerimiento del producto**

  Se deberá llevar un respaldo de todo el histórico pues ante cualquier fallo debe haber un respaldo.

- **Requerimiento de organización**

  La empresa ETAPA-EP ha dispuesto la necesidad de desarrollar aplicaciones con software libre.

- **Requerimiento disponibilidad**

  El sistema debe estar disponible las 24 horas del día, 7 días de la semana y los 365 días del año.

- **Requerimientos de Seguridad**

  El acceso a la información será manejada por distintos usuarios pues cierta información es sensible como los históricos del registro de la intensidad de la lluvia. Estos históricos deben mantener su confidencialidad.

- **Facilidad de uso**

  La información de salida se observará de forma muy ilustrativa. Los reportes se presentarán en gráficos estadísticos.

- **Flexibilidad**

  Los usuarios deben poder acceder a la aplicación con su navegador de preferencia pudiendo ser Explorer, Firefox, Opera, etc.

5.1.1.4 Casos de Uso

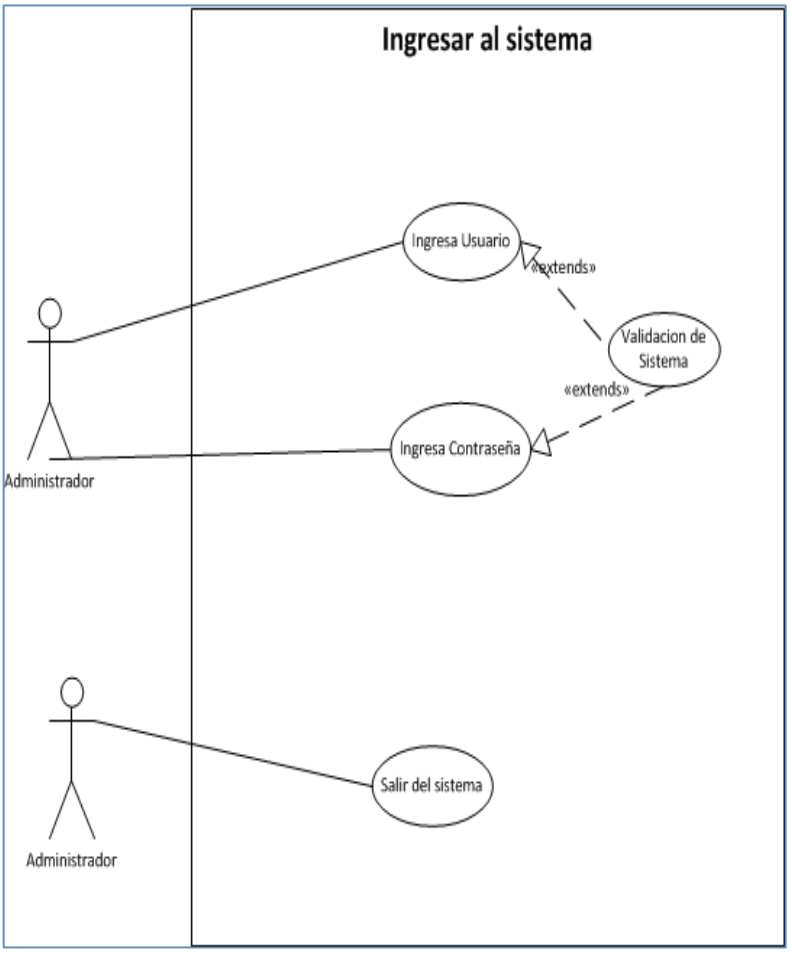


Figura 5.1 Ingresar al sistema. Fuente: Los autores.

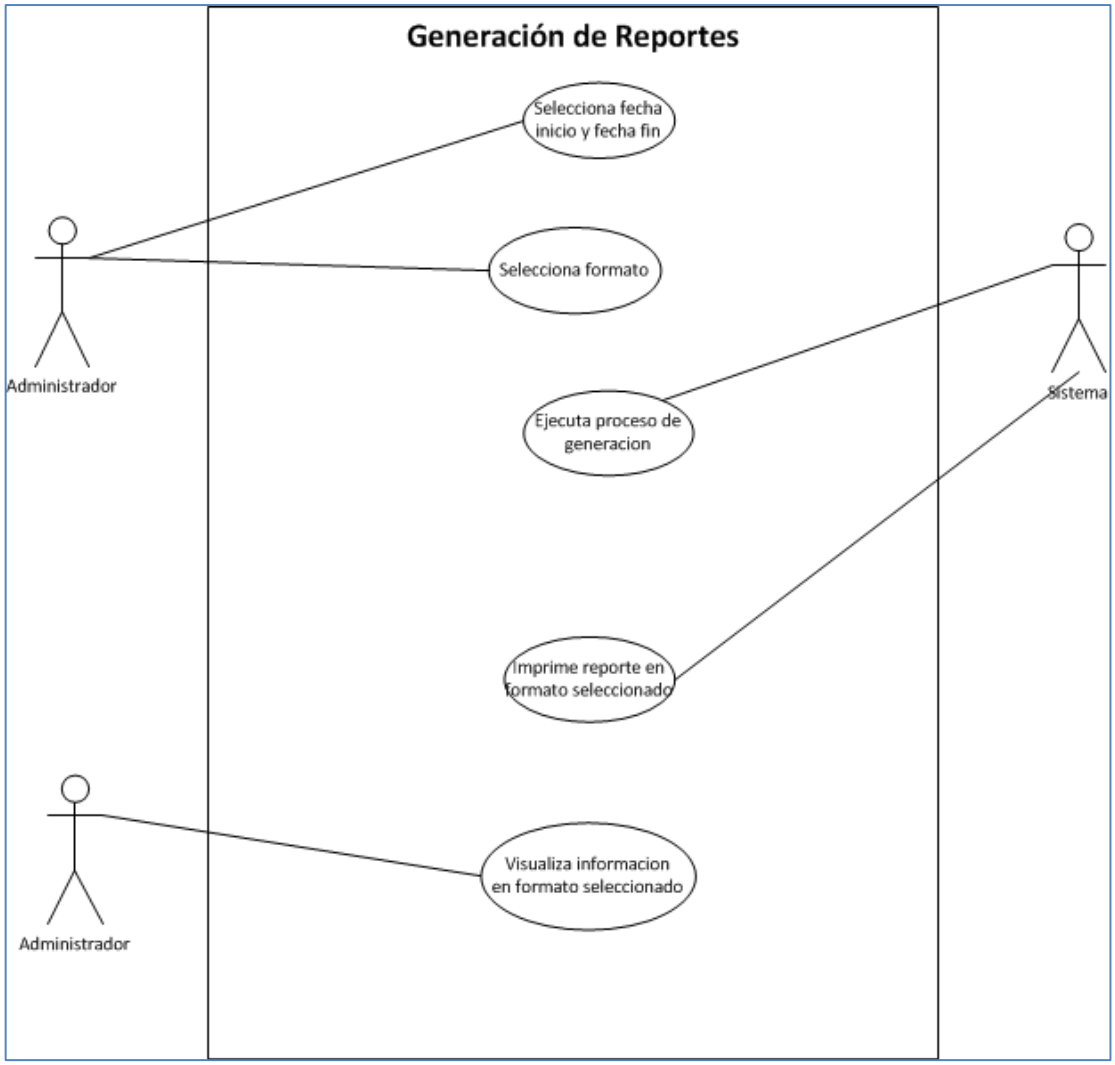


Figura 5.2 Generación de reportes. Fuente: Los autores.

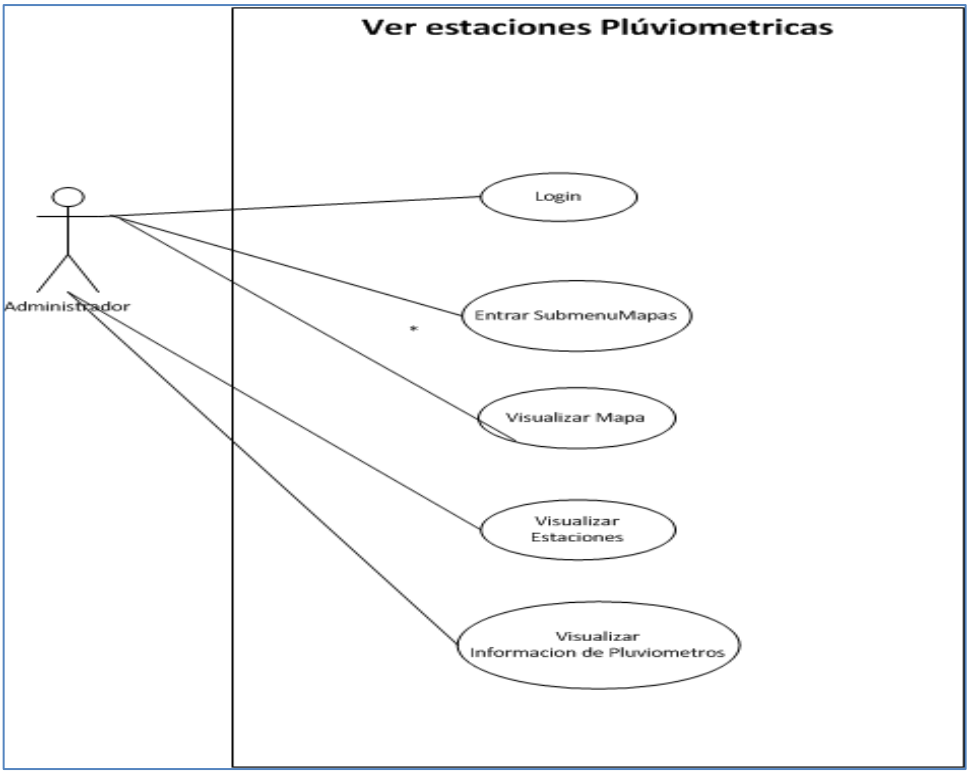


Figura 4.3 Ver estaciones pluviométricas.

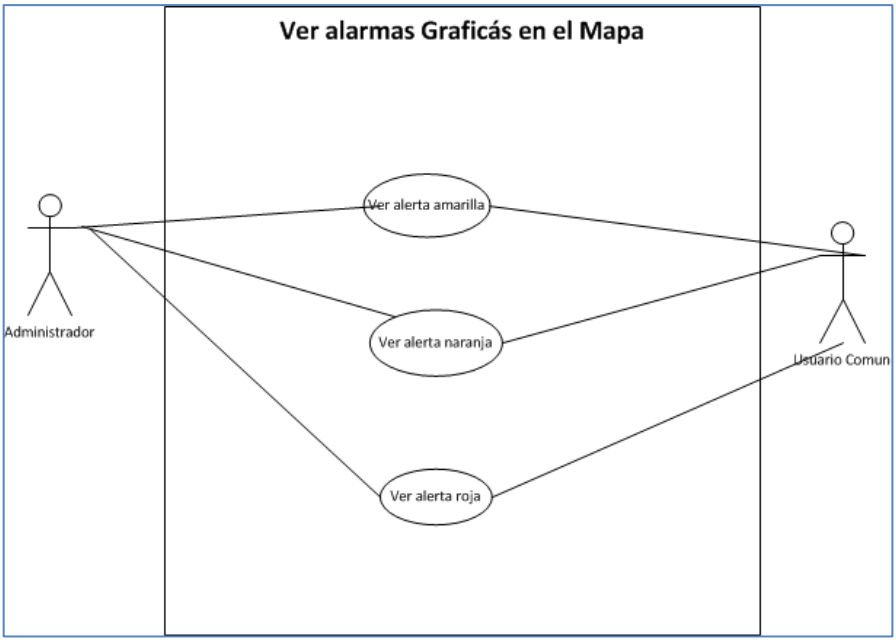


Figura 4.4 Ver alarmas gráficas en Mapa. Fuente: Los autores.

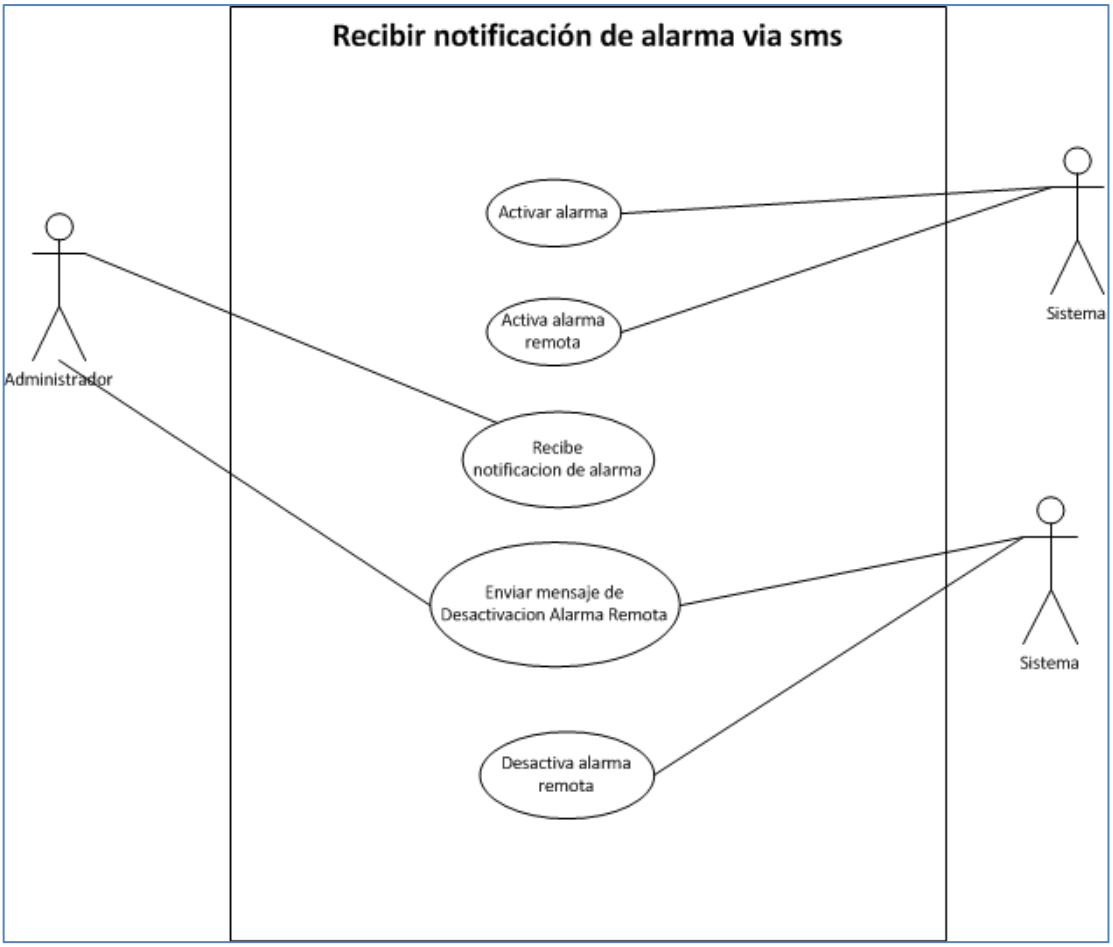


Figura 4.5 Recibir notificación de la alarma vía SMS.

5.2 Diseño

El diseño nos permitirá definir las herramientas que utilizaremos en nuestro sistema. El objetivo es realizar en base a este diseño una aplicación que cumpla con todos los requerimientos antes definidos.

5.2.1 Diseño de Datos

5.2.1.1 Arquitectura de Base de Datos

5.2.1.1.1 Modelo Lógico

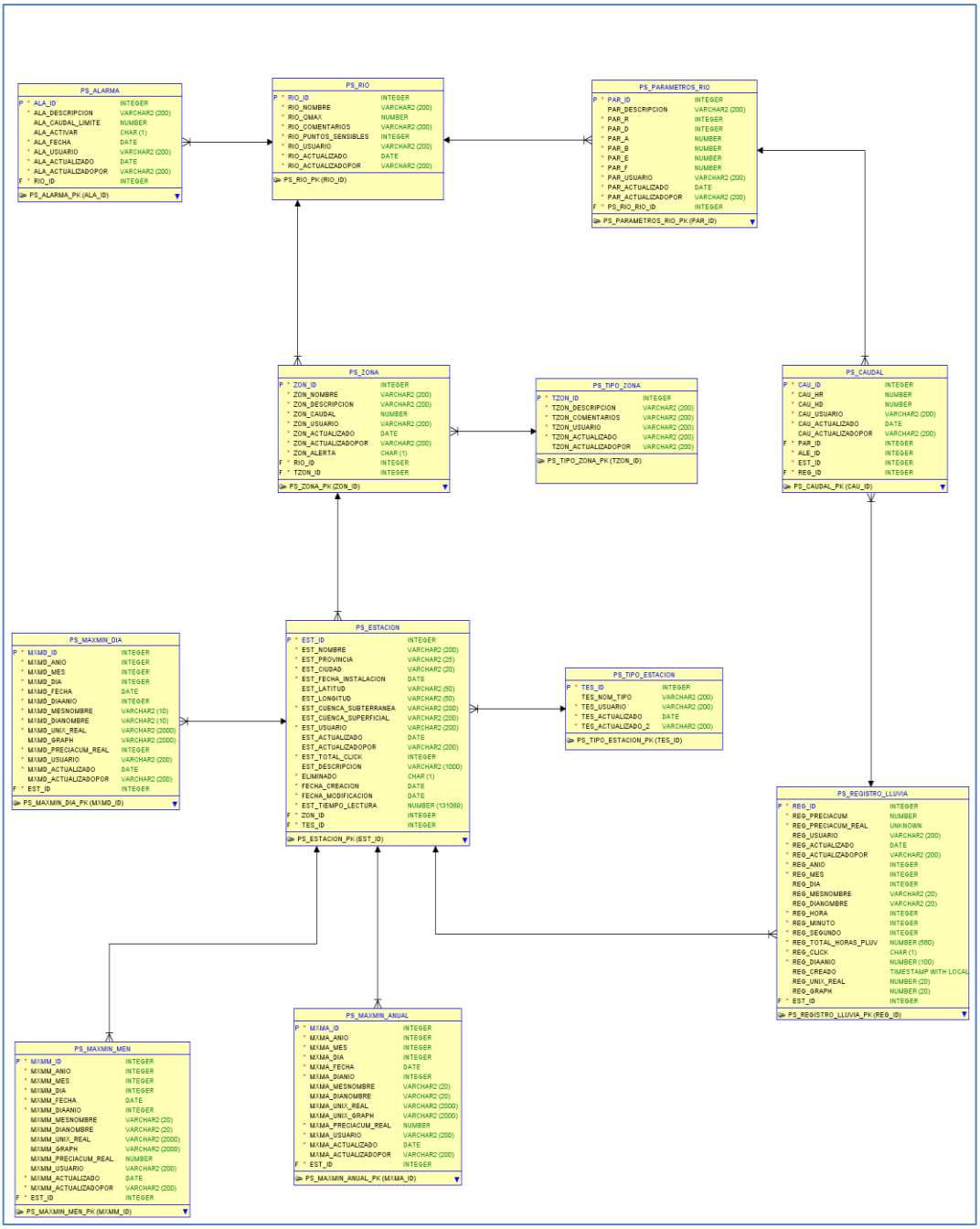


Figura 4.6 Diagrama Entidad Relación. Tablas que utilizaremos para el almacenamiento de datos

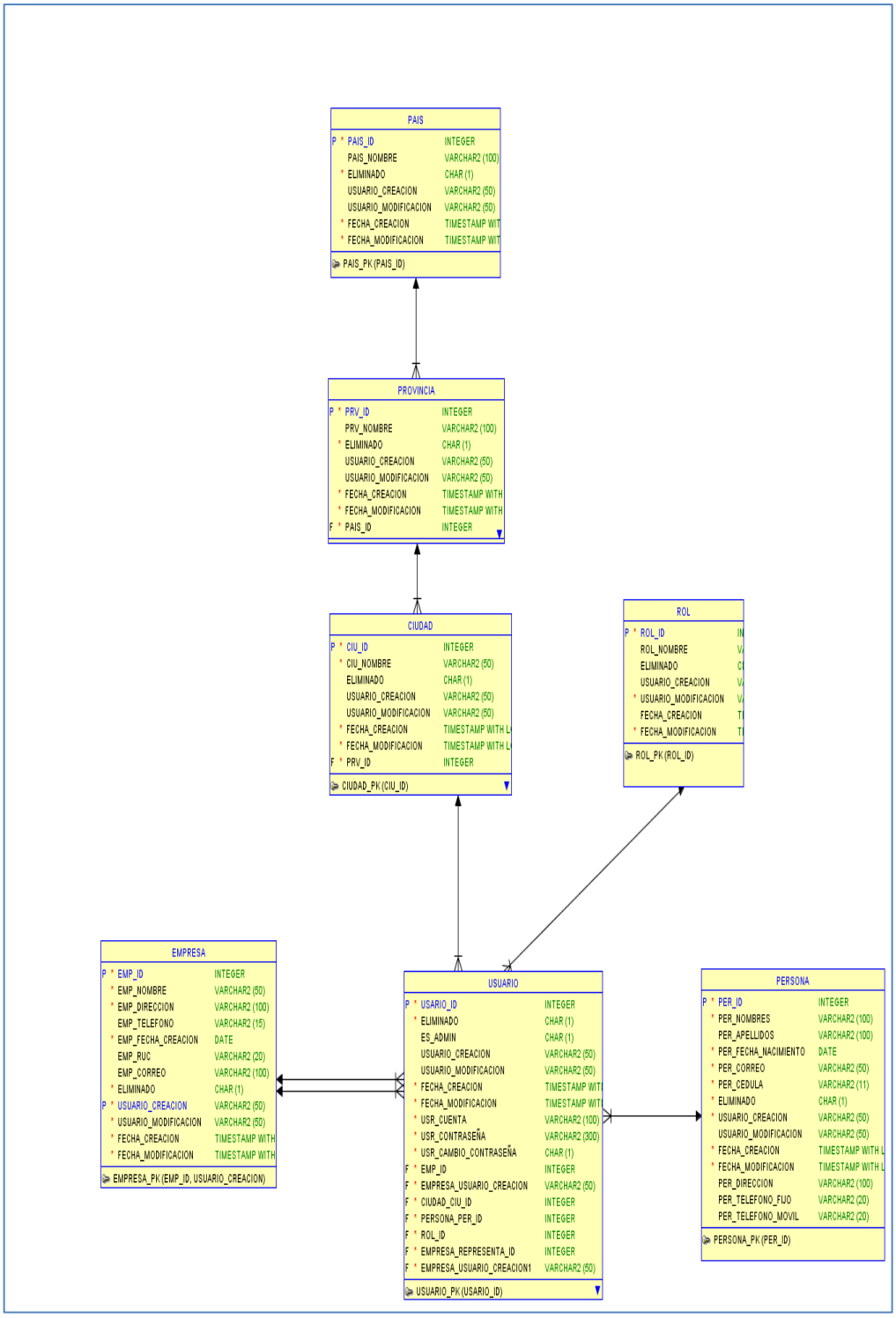


Figura 5.7 Diagrama Entidad-Relación para la administración de los usuarios.

5.2.1.1.1 Modelo Físico

Tablas.- El prefijo para definir las tablas será PS seguido de un guión bajo, a continuación el nombre de la entidad.

PS_: es el prefijo de PluSoftEP.

Ejemplo: PS_CAUDAL

Columnas de las Tablas.- se tomará en consideración los siguientes enunciados:

Si el nombre de la tabla tiene una sola palabra se tomará como prefijo las tres primeras letras del nombre de la tabla.

Ejemplo:

Tabla: PS_ZONA. Columnas: ZON_ID, ZON_NOMBRE, ZON_DESCRIPCION.

Si el nombre de la tabla contiene más de dos palabras se tomarán en consideración las tres primeras letras de la primera palabra pero quedará a libre elección el orden de las mismas y la selección de la cuarta letra que provendrá de la segunda letra.

Ejemplo:

Tabla: PS_MAXMIN_ANUAL. Columnas: MXMA_ID, MXMA_ANIO, MXMA_MES.

Estructura del esquema:

TIPO	ELEMENTO
Almacenamiento de datos del sistema	PS_ALARMA, PS_RIO, PS_PARAMETROS_RIO, PS_ZONA, PS_TIPO_ZONA, PS_CAUDAL, PS_ESTACION, PS_TIPO_ESTACION, PS_MAXMIN_DIA, PS_MAXMIN_MES, PS_MAXMIN_ANUAL, PS_REGISTRO_LLUVIA.
Administración de usuarios	PAIS, PROVINCIA, CIUDAD, ROL,

	EMPRESA, USUARIO, PERSONA.

5.2.2 Diseño del sistema

Para visualizar mejor al sistema lo hemos definido por módulos. Como podemos ver en la siguiente imagen donde podemos ver:

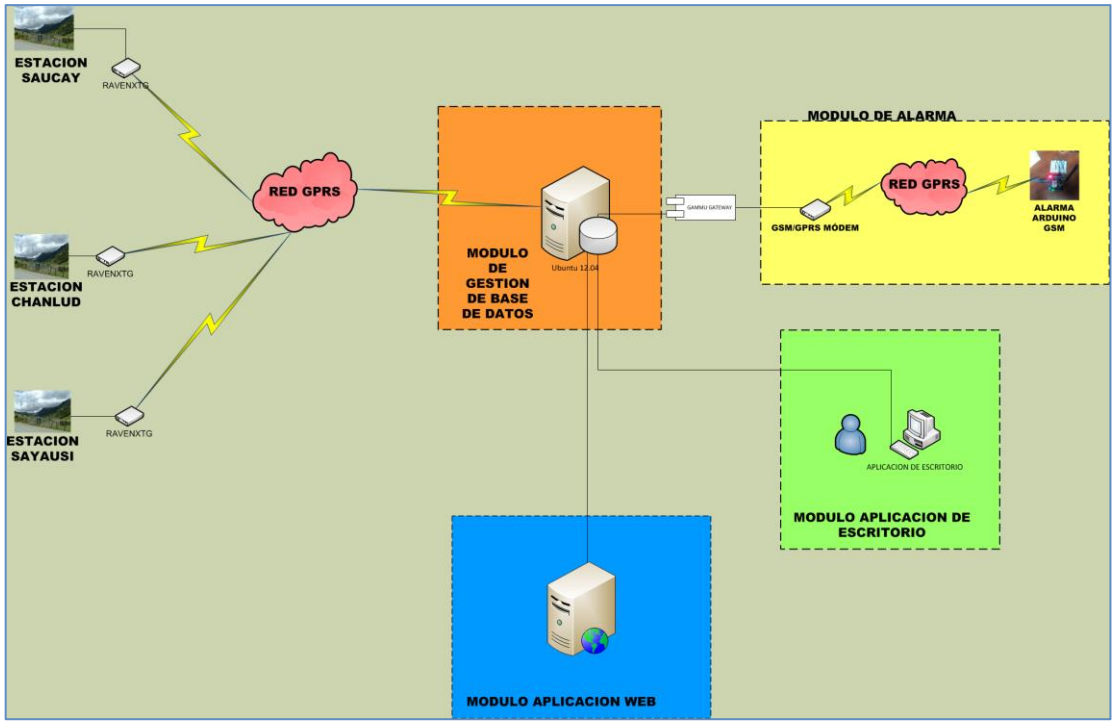


Figura 5.8 Diagrama con la identificación de los módulos que conformarán nuestro sistema PluSoftEP. Fuente: Elaboración de los autores.

El módulo de la gestión de la base de datos.- Almacena los datos referentes a las estaciones, históricos, las alarmas, etc. Además almacena los datos para la administración de los usuarios.

El módulo de alarma.- Utilizará tecnología Arduino para el desarrollo del prototipo. Para la comunicación con la base de datos se utilizará Gammu Gateway.

El módulo de la aplicación web.- Se encargará de visualizar las alarmas en la web. El usuario podrá visualizar a través de este módulo los reportes.

El módulo de la aplicación de escritorio.- Se encargará de ciertos parámetros que el administrador podrá realizar como por ejemplo la modificación de los límites máximos de caudal de las orillas de los ríos.

En el siguiente capítulo describiremos las herramientas que utilizamos para desarrollar cada uno de estos módulos así como su correcta instalación y configuración.

Capítulo 6 IMPLEMENTACIÓN Y PRUEBAS

6.1 Implementación

La empresa pública ETAPA-EP tiene como prioridad el desarrollo de aplicaciones basadas en software libre. Nuestros conocimientos y fuerte envergadura orientada a la programación con herramientas CON licencia GPL nos han llevado a la búsqueda, aprendizaje e implementación de este proyecto de tesis en herramientas justamente orientadas hacia esta rama. Dichas herramientas describiremos a continuación.

6.1.1 Herramientas de Implementación

6.1.1.2 PostgresSQL

PostgreSQL es una base de datos relacional de alta disponibilidad, con capacidad de almacenar gran cantidad de información, con varios años en el mercado mundial de bases de datos, ha ganado madurez y robustez convirtiéndose en una de las más conocidas y mejores bases de datos relacionales. Cuenta con amplio soporte y brinda herramientas de administración comparables a las que ofrecen bases de datos privativos como Oracle.
La página web oficial de PostgreSQL en español tiene como definición: PostgreSQL es un sistema de gestión de bases de datos objeto-relacional, distribuido bajo licencia BSD y con su código fuente disponible libremente. Es el sistema de gestión de bases de datos de código abierto más potente del mercado y en sus últimas versiones no tiene nada que envidiar a otras bases de datos comerciales. [20]

Por estas razones y por muchas más se escogió PostgreSQL como la base de datos del Sistema.

PostgreSQL almacenará información sensible y que no varíe demasiado en el tiempo, información como:

- Nombre de las estaciones
- Zonas
- Nombre de Ríos o afluentes
- Lugares
- Nombres y datos de Personas
- Datos de Usuarios

- Registro de actividad de Usuario

- Registro de eventos - Alarmas

6.1.1.3 Node.js

Node.js es un servidor de aplicaciones web basado en el compilador V8 de Google, su concepto principal es el de tener JavaScript del lado del servidor, haciendo de este un servidor rápido, flexible y compacto. Es un servidor orientado a eventos, debido a esto fue diseñado para soportar aplicaciones en tiempo real. Debido a que maneja un solo hilo de ejecución para atender a múltiples usuarios lo vuelve un servidor muy escalable, con capacidad de atender hasta miles de usuarios, una ventaja bastante favorable en comparación a otro tipo de servidores. Debido a que sus características primordiales se ajustan a los requerimientos del sistema este srevidor es el que se ha implementado para soportar al mismo.

> La meta número uno declarada de Node.js es "proporcionar una manera fácil para construir programas de red escalables". Lo que representa un problema con los lenguajes de programación convencionales. En lenguajes como Java y PHP, cada conexión genera un nuevo hilo que potencialmente viene acompañado de 2 MB de memoria. En un sistema que tiene 8 GB de RAM, esto da un número máximo teórico de conexiones concurrentes de cerca de 4.000 usuarios. A medida que crece su base de clientes, si se desea que una aplicación soporte más usuarios, se necesita agregar más y más servidores. Desde luego, esto suma en cuanto a los costos de servidor del negocio, a los costos de tráfico, los costos laborales, y más. Además de estos costos están los costos por los problemas técnicos potenciales — un usuario puede estar usando diferentes servidores para cada solicitud, así que cualquier recurso compartido debe almacenarse en todos los servidores. Por todas estas razones, el cuello de botella en toda la arquitectura de aplicación Web (incluyendo el rendimiento del tráfico, la velocidad de procesador y la velocidad de memoria) era el número máximo de conexiones concurrentes que podía manejar un servidor.
>
> Node.js resuelve este problema cambiando la forma en que se realiza una conexión con el servidor. En lugar de generar un nuevo hilo de OS para cada conexión (y de asignarle la memoria acompañante), cada conexión dispara una ejecución de evento dentro del proceso del motor de Node.js
>
> Node.js también afirma que nunca se quedará en punto muerto, porque no se permiten bloqueos y porque no se bloquea directamente para llamadas E/S. Un servidor que lo ejecute puede soportar decenas de miles de conexiones concurrentes.

Entonces, con este tipo de aplicaciones ahora se pueden manejar cientos de miles de conexiones concurrentes. A continuación veremos cómo trabaja Node.js y también como está diseñado.

Node.js es un programa de servidor. Sin embargo, el producto base de Node.js definitivamente *No* es como Apache o Tomcat. Esos servidores básicamente son productos para servidor listos para instalar y que están listos para implementar aplicaciones instantáneamente. Node definitivamente no trabaja así. De forma similar como en Apache se pueden agregar módulos PHP para permitir a los desarrolladores crear páginas Web dinámicas, y un módulo SSL para conexiones seguras, Node.js también tiene el concepto de módulos que se pueden agregar a su núcleo. Literalmente hay cientos de módulos de los que se puede escoger con Node.js, y la comunidad es bastante activa en cuanto a producir, publicar y actualizar docenas de módulos por día. [21]

El tipo de arquitectura que se maneja es el de tener un back-end (lado del servidor) y un front-end (lado del usuario)

Usar JavaScript del lado del servidor con Node.js específicamente permite mostrar actualizaciones en tiempo real sobre la marcación de niveles de lluvia en las estaciones, un ambiente propicio para aplicar esta tecnología que es hoy por hoy usada en muchos proyectos de gran producción de información.

6.1.1.3.1 LAMP (Linux Apache MySQL PHP) vs JavaScript

Llevar JavaScript fuera del navegador ha sido un gran salto en la programación web, todo para satisfacer la necesidad de navegación asíncrona (Actualizaciones dinámicas en la página web).

Años atrás la forma habitual de desarrollar páginas web estáticas con tecnologías como: html, php, aspx, funcionaba correctamente para páginas publicitarias o no interactivas. Aproximadamente desde el año 2009 esto ha ido cambiando paulatinamente, llevando la interacción y experiencia de usuario a otro nivel en la navegación web.

LAMP está considerada como una de las mejores herramientas disponibles para que cualquier organización o individuo pueda emplear un servidor web versátil y potente. Aunque son tecnologías creadas por separado, cada una de las tecnologías que lo forman dispone de una serie de características comunes. Especialmente interesante es el hecho que estos cuatro productos pueden funcionar en una amplia gama de hardware, con requerimientos relativamente pequeños sin perder estabilidad. [22]

6.1.1.4 Spring Framework

Es considerado uno de los frameworks más populares en J2EE, Spring es una plataforma MVC que proporciona una infraestructura ordenada, fuerte y segura para desarrollar aplicaciones basadas en Java, usa el lenguaje Groovy el cual difiere un poco en sintaxis y esto lo hace aún más flexible.

6.1.1.4.1 Módulos

Spring está estructurado por módulos de tal forma que se delega un trabajo específico a cada uno de ellos, de esta manera Spring ofrece una plataforma de trabajo por capas, aprovechando los recursos del sistema. Spring consta de 20 módulos. [23]

En la siguiente figura podemos apreciar de mejor manera sus módulos:

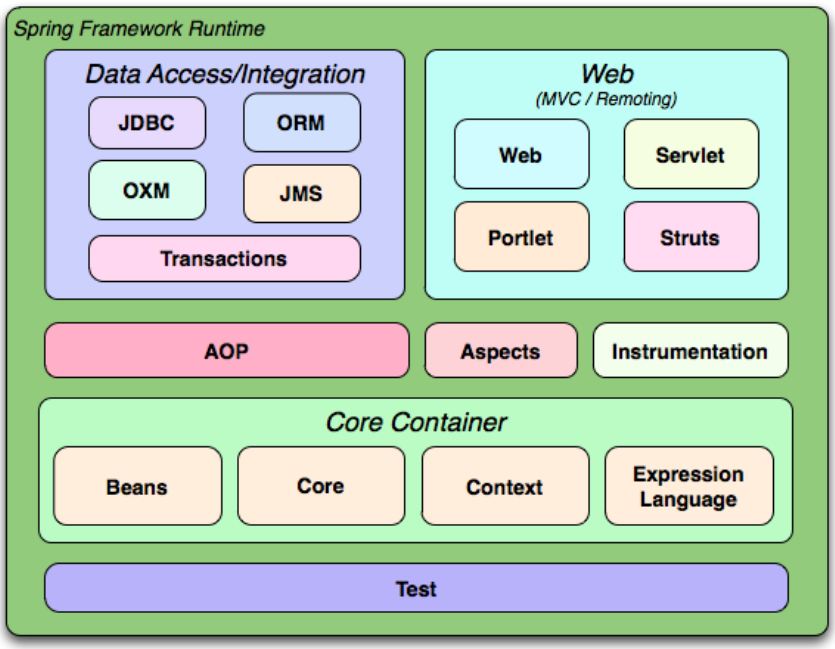


Figura 6.1 Módulos de Spring. Fuente: http://img.genbetadev.com/2011/06/spring-modules.png

6.1.1.5 Grails

Grails es en realidad una aplicación Spring MVC disfrazada, utiliza el lenguaje de programación Groovy similar en realidad a Java, pero difiere en que se omiten varias características del lenguaje padre que es Java como “;” (punto y coma) al final de cada declaración, y la declaración de variable y métodos se realiza por medio de la palabra reservada def. [24]

Grails y Spring apoyan conjuntamente el mapeo de la base de datos PostgreSQL, por medio de controllers se apoyan servicios REST en formato JSON que son consumidos tanto para el módulo de mapas y módulo de administración del sistema, la facilidad en la implementación de dichos controladores hacen de estas herramientas una buena opción para cubrir los requerimientos en cuanto a comunicación dentro del sistema.

6.1.1.6 Hibernate

Hibernate es un framework de persistencia para la plataforma Java y también disponible para .net como NHIbernate, que facilita el mapeo de entidades de una base de datos relacional y también el modelo de objetos de una aplicación, utiliza spring-orm (Spring-Mapeo Objeto Relacional). Hibernate es un software libre que es distribuido bajo los términos de la licencia GNU LGPL.

Una de las comunidades de web lo define como: Hibernate es una herramienta de mapeo objeto/relacional para entornos Java. El término de mapeo objeto/relacional (ORM) se refiere a la técnica de mapear una representación de datos desde un modelo de objeto a un modelo de datos relacional con un esquema basado en SQL. Hibernate no solamente se ocupa del mapeo desde las clases Java a las tablas de las bases de datos (y desde los tipos de datos de Java a los tipos de datos de SQL), sino que también facilita la consulta y recuperación de datos. Esto puede reducir de manera importante el tiempo de desarrollo que se tomaría con el manejo de datos de forma manual en SQL y JDBC. [25]

Su arquitectura la podemos observar en la figura 6.2.

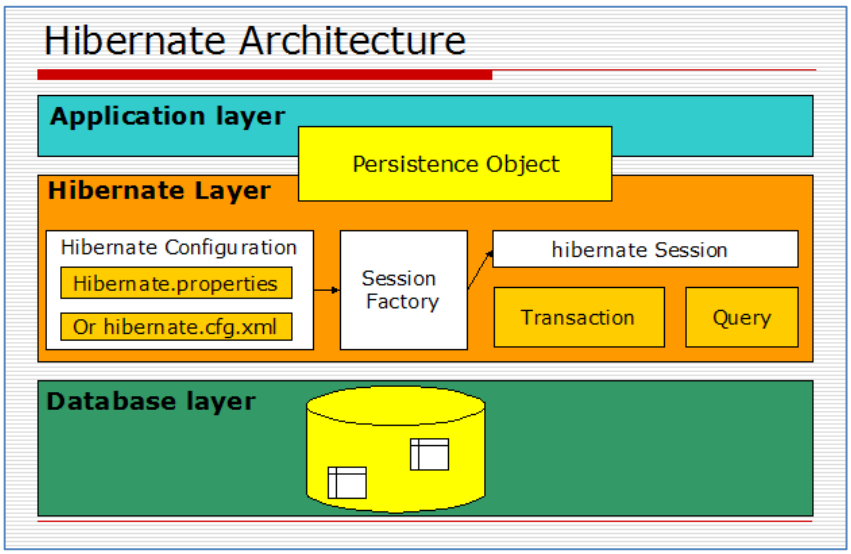


Figura 6.2 Arquitectura de Hibernate. Fuente: http://www.cloudsopedia.com/tutorial/hibernate/images/hibernate2.png

6.1.1.7 APACHE KAFKA

Desarrollado por LinKedln, Kafka es un sistema distribuido basado en PUB-SUB (Publicación y suscripción) de mensajería, ofrece una solución capaz de manejar toda la actividad del flujo de datos y procesar estos datos en un sitio web de gran consumo.

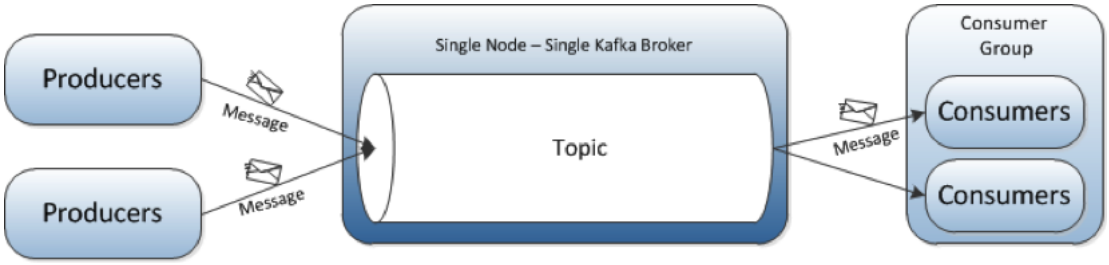


Figura 6.3 Arquitectura de APACHE KAFKA.
Fuente: http://proskurnia.in.ua/blog/wp-content/uploads/2013/08/ApacheKafkaExample-300x70.png

Está diseñado con estos objetivos:

- Alto rendimiento: incluso con hardware muy modesto Kafka puede soportar cientos de miles de mensajes por segundo.
- Soporte para la partición de mensajes a través de los servidores de Kafka y consumo distribuido en un clúster de máquinas consumidores, manteniendo la ordenación por partición
- Soporte para la carga de datos en paralelo en Hadoop.

Kafka pretende unificar el procesamiento online y offline, proporcionando un mecanismo para la carga paralela en Hadoop, así como la capacidad de partición en tiempo real del consumo en un clúster. [26]

En esta ocasión Kafka ha sido implementado como una solución para la comunicación de sistemas de producción como lo es la aplicación de escritorio basada en código puramente Java que actúa como PRODUCTOR, luego se maneja la actividad del flujo de datos que es enviado hacia Spring/Grails configurado como CONSUMIDOR y PRODUCTOR a su vez el productor Kafka en Grails transmite el flujo de información hacia el Back-end en Node.js que actúa como un consumidor del flujo de la información.

6.1.1.7 APACHE ZOOKEEPER

Es un servicio de coordinación de alto rendimiento para aplicaciones distribuidas.

ZooKeeper es muy sencillo, permite distribuir servicios en cientos de máquinas, además soporta jerarquías: [27]

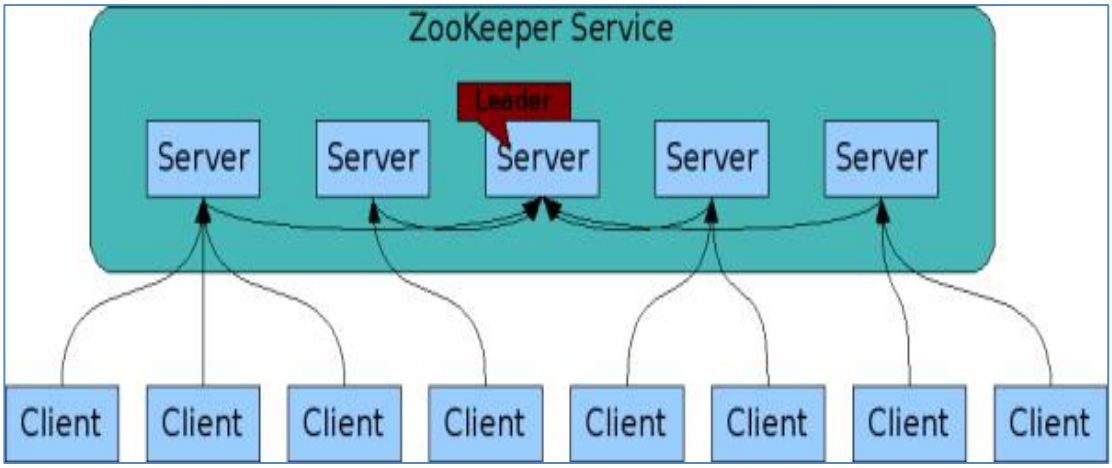


Figura 6.4 Servicio de ZooKeeper. Fuente: http://zookeeper.apache.org/doc/r3.4.6/images/zkservice.jpg

Componentes de ZooKeeper:

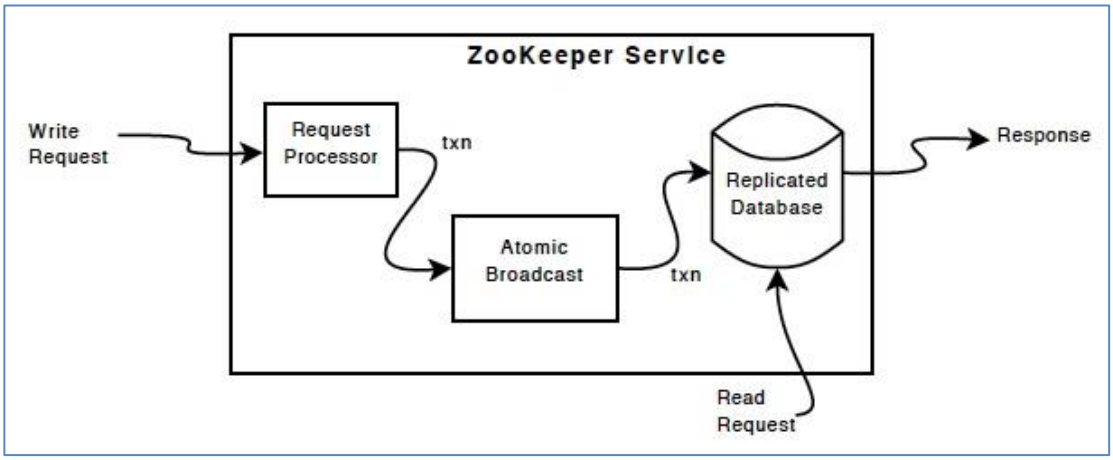


Figura 6.5 Componentes de ZooKeeper. Fuente: http://zookeeper.apache.org/doc/r3.4.6/images/zkcomponents.jpg

6.1.1.8 Gammu Gateway

Como parte del Sistema de Alerta Temprana se implementó un Gateway de mensajería para confirmar que se activó una alerta utilizando un módem, para lograr este objetivo se investigaron varios sistemas pero el que cumplió con los requerimientos iniciales fue Gammu. El cual es un proyecto de código abierto que tiene como utilidad primordial controlar un teléfono celular o módem por medio de línea de órdenes o comandos AT, fue desarrollado en C y se basa en la librería libGammu.

Las funcionalidades Gammu son invocadas desde Java, de esta forma se automatizó la funcionalidad de enviar un mensaje de texto.

La utilidad de línea de órdenes Gammu provee acceso a una amplia variedad de características. Sin embargo, el nivel de soporte varía de un teléfono a otro. Por lo general las siguientes características tienen soporte: [28]

- Listado, inicio y manejo de llamadas
- Recuperación, copia de respaldo y envío de SMS
- Recuperación MMS
- Listado, importación y exportación de contactos (también de formatos estándares tales como vCard)
- Listado, importación y exportación de calendario y tareas (también de formatos estándares tales como vCalendar o iCalendar)
- Recuperación de teléfono e información de red
- Acceso a sistema de archivos del teléfono (nótese que algunos teléfonos funcionan también como dispositivos de almacenamiento USB y no se puede acceder a estos a través de Gammu)

6.1.2. Configuración e implementación

Para la implementación del *módulo de gestión de la base de datos* y *el módulo de aplicación web* seguimos los siguientes procesos.

6.1.2.1 Instalación y configuración Grails

Abrimos el terminal en Ubuntu y escribimos lo siguientes comandos:

Paso 1 Agregando repositorio

sudo add-apt-repository ppa:groovy-dev/grails

Paso 2 Actualizando repositorio del sistema

sudo apt-get update

Paso 3 get Ppa

sudo apt-get install grails-ppa

Paso 4 Instalación

Para agregar grails 2.2.1

sudo apt-get install grails 2.0.x

#to add grails 1.3.9

sudo apt-get install grails-1.3.9

#to add grails 1.2.5

sudo apt-get install grails-1.2.5

#Cambiando entre versiones

sudo update-alternatives --config grails

6.1.2.2 Instalación y configuración PostgreSQL

Paso 1 Actualizar repositorio de paquetes

> sudo apt-get update

De ser necesario actualizamos también el sistema con:

> sudo apt-get upgrade

Paso 2 Instalación

> sudo apt-get install postgresql-9.1 libpq-dev

Paso 3

> Instalación PgAdmin

PgAdmin es una interfaz de administración para PostgreSQL, se la puede instalar de la siguiente manera:

> sudo apt-get install pgadmin3

Paso 4 Crear Usuario y contraseña

Procedemos a crear una cuenta de usuario con una contraseña para vincularla a PostgreSQL y poder acceder a su administración.

> sudo su postgres -c psql

postgres=# CREATE ROLE nombredeusuario SUPERUSER LOGIN;

postgres=# \q

De esta manera se ha instalado la base de datos para el Sistema y está lista para su administración.

6.1.2.3 Instalación y configuración Kafka

Inicio rápido con Kafka, pasos para su instalación, configuración y prueba:

Paso 1 Descargar

Se procede a descargar el Software, en pruebas realizadas se detectó que Kafka está diseñado para sistemas operativos de 64 bits, en sistemas operativos de 32 bits el software presenta errores de compatibilidad principalmente con el JDK i86.

Para sistemas Linux corremos los siguientes comandos:

> **tar -xzf kafka_2.9.2-0.8.1.tgz**

> **cd kafka_2.9.2-0.8.1**

Paso 2 Arrancar el servidor

Kafka usa ZooKeeper como coordinador de procesos, así que necesitamos iniciar su servicio antes, si no se tiene instalado un servidor ZooKeeper podemos utilizar el que viene embebido en el paquete Kafka.

> **bin/zookeeper-server-start.sh config/zookeeper.properties**

Ahora arrancamos el servidor Kafka

> **bin/kafka-server-start.sh config/server.properties**

Paso 3 Crear un topic o tema

Un topic o tema quiere decir un espacio de discusión el cual pueden escuchar uno o varios usuarios.

Al crear un topic simulamos la creación de una sala a la cual varios Usuarios podrán acceder, entonces escucharan solo lo que se transmita en dicha sala, esto beneficia al sistema ya que se pueden implementar n topics pudiendo emitirse diferente información en cada una de ellas.

Es importante definir los topic que usaremos en el sistema, para el caso de nuestro sistema serán dos:

a) Topic:mapa

b) Topic:biblioteca_estaciones

Vamos a crear un topic llamado test.

> **bin/kafka-topics.sh --create --zookeeper localhost:2181 --replication-factor 1 --partitions 1 --topic test**

Podemos ver la lista de tópicos corriendo el siguiente comando:

> **bin/kafka-topics.sh --list --zookeeper localhost:2181**

test

Paso 4 Enviar algunos mensajes - productor

Kafka viene con un cliente por línea de comandos que podemos utilizar para enviar mensajes por separado al clúster instanciado en Kafka.

Corriendo el siguiente mensaje podemos ingresar el mensaje que queremos enviar por consola:

> **bin/kafka-console-producer.sh --broker-list localhost:9092 --topic test**

Éste mensaje:

mapa
biblioteca_estaciones

Paso 5 Iniciar un consumidor

Kafka también viene con un consumidor por línea de comandos que podemos usar para probar los productores que tengamos inicializados.

> **bin/kafka-console-consumer.sh --zookeeper localhost:2181 --topic test --from-beginning**

Este mensaje:

Listen for Topic instance

Se obtiene el mensaje que se implementó en nuestro productor.

Se puede configurar un multi-broker clúster en Kafka pero para nuestro caso esto no será necesario.

6.1.2.4 Instalación y configuración Node.js

Paso 1 Npm library manager

Para sistemas Linux, abrimos una terminal y escribimos lo siguiente:

> sudo apt-get install nodejs npm

npm es un gestor de librerías para Node.js, esto facilitará la instalación de librerías y versiones.

Por ejemplo podremos instalar socket.io de la siguiente manera:

> sudo npm install socket.io

Paso 2 Repositorio

Posteriormente actualizamos, vamos a la terminal y digitamos:

> sudo add-apt-repository ppa:chris-lea/node.js

Una vez que tengamos nuestro repositorio actualizado procedemos a instalar nuevamente.

Paso 3 Actualización e Instalación

> sudo apt-get update && sudo apt-get install nodejs npm

Paso 4 Verificación

Por último verificaremos si la instalación fue exitosa probando la versión de Node.js digitando:

> node -v

Listo ahora tenemos instalado el servidor Node.js

6.1.2.5 Formato de datos

A continuación tenemos una muestra de la información que está llegando a la base de datos. En ella podemos observar que cada campo está separado por un “;”. Según la información proporcionada por la empresa ETAPA-EP la información que nos interesa son los datos que hemos resaltado con negrita. A continuación tenemos una muestra:

8582;0;**8**;**"2011-01-29 00:00:00-05"**;**"1899-12-30 07:52:20-05:14"**;0;232;3;129;16;2134;32918;**20**;97**;**98;99;99;100;100;101;101;102;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0

El primer dato que es de suma importancia para nuestro proyecto es el tercero (contando de izquierda a derecha) en este ejemplo es el número **8**, este dato nos indica la estación. El siguiente dato es el cuarto, que es **2011-01-29 00:00:00-05** éste nos indica la fecha en este caso viene siendo 2011-01-29 el resto solo es parte del formato GMT que no interesa. El siguiente dato es **1899-12-30 07:52:20-05:14** de igual manera éste tiene formato GMT pero la parte que nos interesa es 07:52:20 que es la hora. Por último el treceavo dato en este caso **20** es acumulado total de lluvia en esa hora. El resto de datos forma parte del sistema de adquisición de datos (pluviómetros) que en nuestro proyecto no se contempla.

6.1.2.5 Modulo de alarma: Instalación y configuración Gammu

Como siguiente paso veremos la instalación del Gammu:

Paso 1 Actualizar nuestras librerías

> sudo apt-get update

Paso 2 Instalar Gammu

> sudo apt-get install gammu-smsd

Paso 3 Verificar puertos

> ls -l /dev/tty*

Paso 4 Identificar nuestro dispositivo

> Gammu -c /etc/gammu-smsdrc –identify

Respuesta:
Dispositivo: /dev/ttyUSB0
Fabricante: Huawei
Modelo: unknown (E173)
Firmware: 11.126.85.00.197
IMEI: ##############
IMSI SIM: ############

Es recomendable instalar los paquetes Gammu por medio de Synaptic Package Manager, pues según las pruebas realizadas al instalar Gammu por este medio no dio problemas. Mientras que al instalar los paquetes por línea de comandos desde la terminal los problemas fueron múltiples.

Nota: Por cuestiones de seguridad la información importante del módem ha sido reemplazada por el signo #.

6.1.2.5.1 Configuración de Arduino

```
/*
*CONFIGURACION DE ALARMA ARDUIDO
*/
#include <SoftwareSerial.h>


SoftwareSerial SIM900(7, 8); // PINES 7 Y 8 EN ESTE CASO VAN A HACER LA COMUNICACION SERIAL.
int pin_E=9; // PARA ENERGIZAR EL MODEM
int led_a=2;// PIN 2
int led_b=3;// PIN 3
int led_c=4;// PIN 4
```

Figura 6.6 Código para la alarma Arduino.

```
void loop()
{
  if(SIM900.available() >0)//SI LLEGO ALGO AL SIM
  {
    incoming_char=SIM900.read(); // ASIGNAR VALOR QUE CONTIENE SIM900 A VARIABLE incoming_char
    if(incoming_char == 'X')(alerta_naranja();)// SI EL DATO QUE LLEGO AL SIM ES X IR A PROCESO alerta_naranja()
    if(incoming_char == 'x')(apagar_alerta_naranja();)//SI EL DATO QUE LLEGO AL SIM ES x IR AL PROCESO apagar_alerta_naranja()
    if(incoming_char == 'Y')(alerta_amarilla();)//SI EL DATO QUE LLEGO AL SIM ES Y IR AL PROCESO alerta_amarilla()
    if(incoming_char == 'y')(apagar_alerta_amarilla();)//SI EL DATO QUE LLEGO AL SIM ES y IR AL PROCESO apagar_alerta_amarilla()
    if(incoming_char == 'Z')(alerta_roja();)//SI EL DATO QUE LLEGO AL SIM ES Z IR AL PROCESO alerta_roja()
    if(incoming_char == 'z')(apagar_alerta_roja();)//SI EL DATO QUE LLEGO AL SIM ES z IR AL PROCESO apagar_alerta_roja()
  }
}

void alerta_naranja()
{
digitalWrite(led_a, HIGH); //PRENDE EL led_a
dato = analogRead(A0);// ASIGNAMOS EL VALOR DEL VOLTAJE A0 A LA VARIABLE dato
if(dato > 500)// COMPARAMOS SI ESTE VALOR ES MAYOR A 500 ES QUE ESTA ARRIBA DE 2.5V
{
  enviar("Alerta Naranja Led3 ON");//POR LO TANTO ENVIAMOS EL MENSAJE
}
Serial.print(dato);
```

Figura 6.7 Dentro del método loop () leeremos lo que llegue al SIM900. Con el proceso alerta naranja () enviaremos el mensaje de alerta y encendemos el led0 respectivo.

```
void apagar_alerta_naranja()
{
digitalWrite(led_a, LOW); //SE LEE EL led_a
dato = analogRead(A0); //SE ASIGNA EL VALOR DE A0 A dato
if(dato < 500)// SI EL VALOR ES MENOR A 500 QUIERE DECIR QUE TIENE UN VOLTAJE MENOR A 2.5V
{
enviar("Alerta Naranja Led3 OFF");
)Serial.print(dato)
;
}
```

Figura 6.8 Este proceso es igual para los otros leds del prototipo.

Para activar el prototipo se utilizará mensajes de texto y para desactivar el mismo de igual forma. Fue necesario definir letras para identificar el tipo de alarma, “X;Y,Z” para la alerta naranja, amarilla y roja respectivamente, y para desactivar el prototipo se utilizan las mismas letras pero en minúsculas es decir, “x,y,z”. El mensaje de respuesta ira acompañado de un texto de aviso sobre la alarma que fue activada o desactivada. Por ejemplo: “La alarma roja fue activada”.

En la siguiente foto podemos observar el prototipo en funcionamiento:

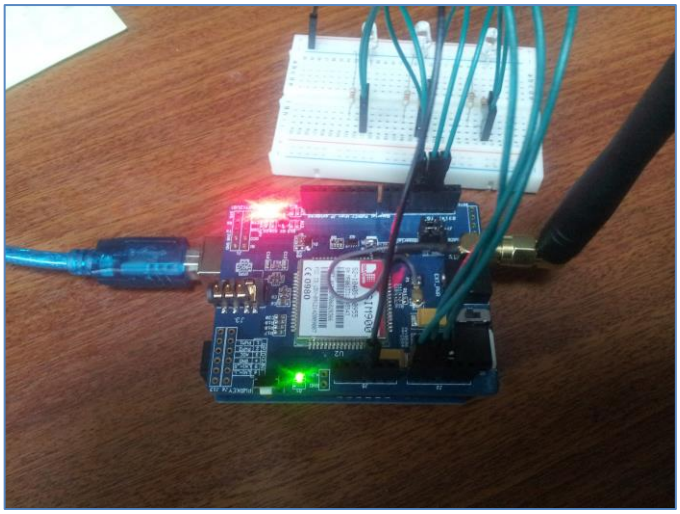


Foto 6.1 Circuito Arduino en funcionamiento. Fuente: Los autores.

6.2 Plan de emergencias

Dentro del plan de emergencias que está basado en el estudio realizado por la empresa italiana AMRA se ha definido la siguiente ecuación para obtener el caudal máximo:

$$Q = f(h_r) + f(h_d)$$

Ecuación 6.1 Ecuación que define el caudal máximo.

Dónde:

- la función $f(h_r)$ tiene en cuenta las contribuciones pluviométricas recientes (h_r);
- la función $f(h_d)$ depende de las lluvias lejanas en el tiempo (h_d). [29]

Para el cálculo de los caudales máximos de los ríos Machángara y Tomebamba se toma los valores del siguiente cuadro de valores:

Río	Caudal máximo para alerta de desborde
Tomebamba	160 m3/s,
Yanuncay	160 m3/s,
Tarqui	90 m3/s
Machángara	125m3/s
Cuenca	300 m3/s

Tabla 6.1 Valor de variables para cálculo de Caudal máximo para alerta de desborde.

En el mismo estudio se han definido las zonas susceptibles a inundaciones.

- Por el río Tomebamba: 4 zonas susceptibles, además de la sección de Matadero en Sayausí.(Ver anexo A)
- Por el río Machángara: 5 zonas susceptibles. (Ver anexo A)
- Por el río Yanuncay: 4 zonas susceptibles.
- Por el río Cuenca: 1 zonas susceptibles. (Tomebamba en Ucubamba).

Coordenadas puntos sensibles

Río	Coordenadas zonas susceptibles
Machángara	PUNTO 1M -2.840432,-78.986489 Paseo Río Machángara Cuenca 1.1 km Sur
	PUNTO 2M Paseo Río Machángara Cuenca 914 km Sur -2.84249,-78.985784
	PUNTO 3M Paseo Río Machángara Cuenca 412 km Sur -2.847119,-78.984969
	PUNTO 4M

	-2.876308,-78.973117 Cornelio Veintimilla Cuenca 241 km Sur
	PUNTO 5M -2.882844,-78.9658 Cuenca 139 km Sur-Oeste

Tabla 6.2 Coordenadas zonas susceptibles río Machángara.

Río	**Coordenadas**
Tomebamba	PUNTO 1T -2.878237, -79.068947 Av. Ordóñes Lazo. Altura de la Vía a Buenos Aires.
	PUNTO 2T -2.889809,-79.036553 Calle Paseo Tres de Noviembre. Al frente del Campus Balzay de la Universidad de Cuenca Cuenca 80 km Este
	PUNTO 3T -2.893067,-79.021962 Av. Tres de Noviembre Altura de las Piscinas Olímpicas. Cuenca 88 km Oeste
	PUNTO 4T -2.90721,-78.983896 Av. Pumapungo - Urbanización Villanueva. A la Altura del redondel de la intersección de la Av. Max Uhle y Av. Veinticuatro de Mayo. Cuenca 52 km Noroeste

Tabla 6.3 Coordenadas zonas susceptibles río Tomebamba.

Río	Coordenadas
Yanuncay	PUNTO1Y -2.895424,-79.070756 Cuenca - Molleturo - Naranjal 1.3 km E
	PUNTO2Y -2.907039,-79.023807 Av. Primero de Mayo Cuenca 49 m SO
	PUNTO3Y -2.912825,-79.013207 Av. Primero de Mayo Cuenca 22 m S
	PUNTO4Y -2.915954,-78.9978 Av. 27 de Febrero Cuenca km 36 NO

Tabla 6.4 Coordenadas zonas susceptibles río Yanuncay

Río	Coordenadas
Yanuncay	PUNTO1Y -2.895424,-79.070756 Cuenca - Molleturo - Naranjal 1.3 km E
	PUNTO2Y -2.907039,-79.023807 Avenida Primero de Mayo Cuenca 49 m SO
	PUNTO3Y -2.912825,-79.013207 Avenida Primero de Mayo

	Cuenca 22 m S
	PUNTO4Y -2.915954,-78.9978 Av. 27 de Febrero Cuenca 36 m NO

Tabla 6.6 Coordenadas zonas susceptibles del río Yanuncay.

Río	**Coordenadas**
Cuenca	-2.875879,-78.94716 Vía a Paccha Cuenca 56 km Sur

Tabla 6.7 Coordenadas zonas susceptibles del río Cuenca.

Las zonas las podremos apreciar de mejor manera en el Anexo A donde veremos las zonas susceptibles marcadas en el Google Maps. En nuestro proyecto se han incluido las zonas susceptibles del río Tomebamba y Machángara. Pues los pluviómetros instalados (Chanlud, Saucay, Sayausí) envían datos validos solo para calcular el caudal máximo de estos dos ríos. En el Anexo B (Río Machángara) Anexo C (Río Tomebamba) observaremos las fotos de las zonas susceptibles y los efectos causados por los desbordamientos. Se pudo observar la falta de trabajos que puedan brindar seguridad en estos sitios especialmente en las zonas del río Machángara.

Para la definición de alarmas se toma en cuenta los siguientes enunciados tomados del análisis de zonas sensibles definidas en el mismo informe de AMRA:

a) Atención altura de agua correspondiente al 30% del caudal de inundación

b) Pre-alarma altura de agua correspondiente al 50% del caudal de inundación

c) Alarma altura de agua correspondiente al 65% del caudal de inundación

d) Emergencia altura de agua correspondiente al 80% del caudal de inundación. [29]

Es así que se define como alarma naranja al enunciado b, alarma amarilla al enunciado c y alarma roja al enunciado d. El resultado final de cómo se observarán dichas alarmas en la web las podemos ver en el Anexo D para las orillas del río Tomebamba y Anexo C para las orillas del río Machángara.

La emergencia será coordinada con las distintas instituciones de socorro como Cruz Roja, Policía Nacional, Bomberos la alarma sería desactivada una vez que la emergencia haya sido atendida.

6.3 Aplicaciones y Servidores en Producción

A continuación es necesario poner en marcha los servidores los cuales estarán escuchando las solicitudes del Usuario, generando el flujo de datos, y verificando el estado de la información dentro del sistema.

6.3.1 Iniciando PluSoftEP Desktop Application

PluSoftEP es una aplicación de escritorio desarrollada puramente en Java, el propósito de esta aplicación es permitir una administración de los datos que llegan desde las estaciones a un más bajo nivel, además permite dar una mantenimiento a los datos como son los parámetros, información sobre las estaciones, etc.

Como se trata de una aplicación de escritorio lo único que se debe hacer es ejecutarla dando doble click sobre el icono de la misma.

Para entornos de desarrollo Linux en este caso Ubuntu nos ubicamos dentro un terminal y accedemos al directorio dist de nuestra aplicación y escribimos el siguiente comando:

> java -jar PluSoftEPSimulacion

El resultado se mostrará como sigue a continuación:

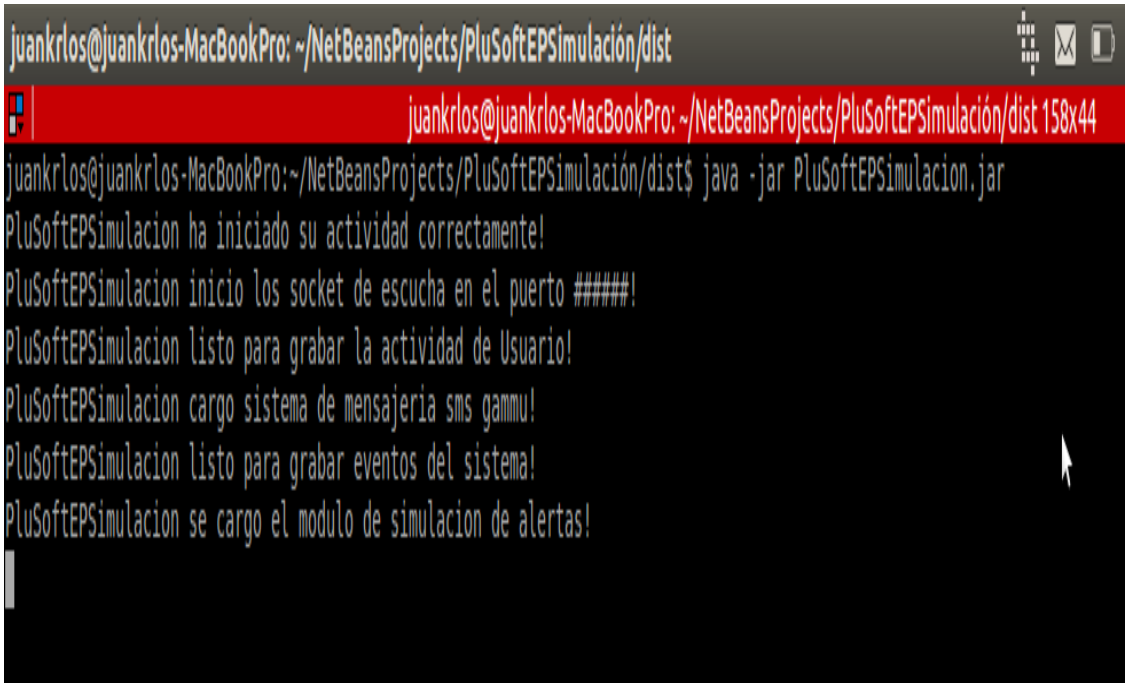


Figura 6.9 Arranque de aplicación de escritorio.

La aplicación de escritorio debe arrancar con normalidad, los controles de usuario están activos y listos para operar.

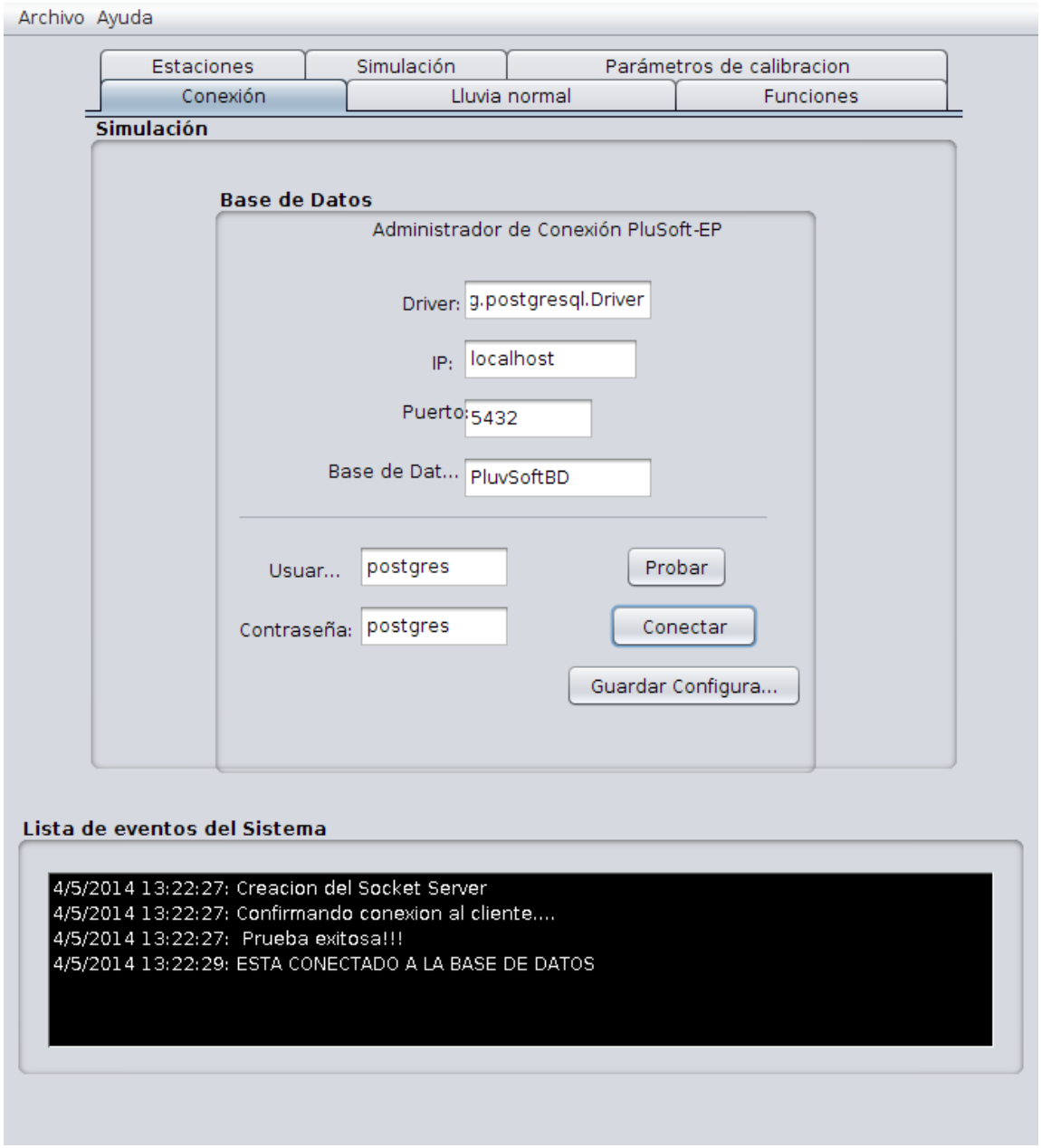


Figura 6.10 Ejecución de la aplicación de escritorio PluSoft.

6.3.3 Arrancando KAFKA Service

Es necesario iniciar nuestro servicio Kafka para esto corremos los comandos antes descritos en el Paso 2 Arrancar servidor.

El resultado debe ser el siguiente:

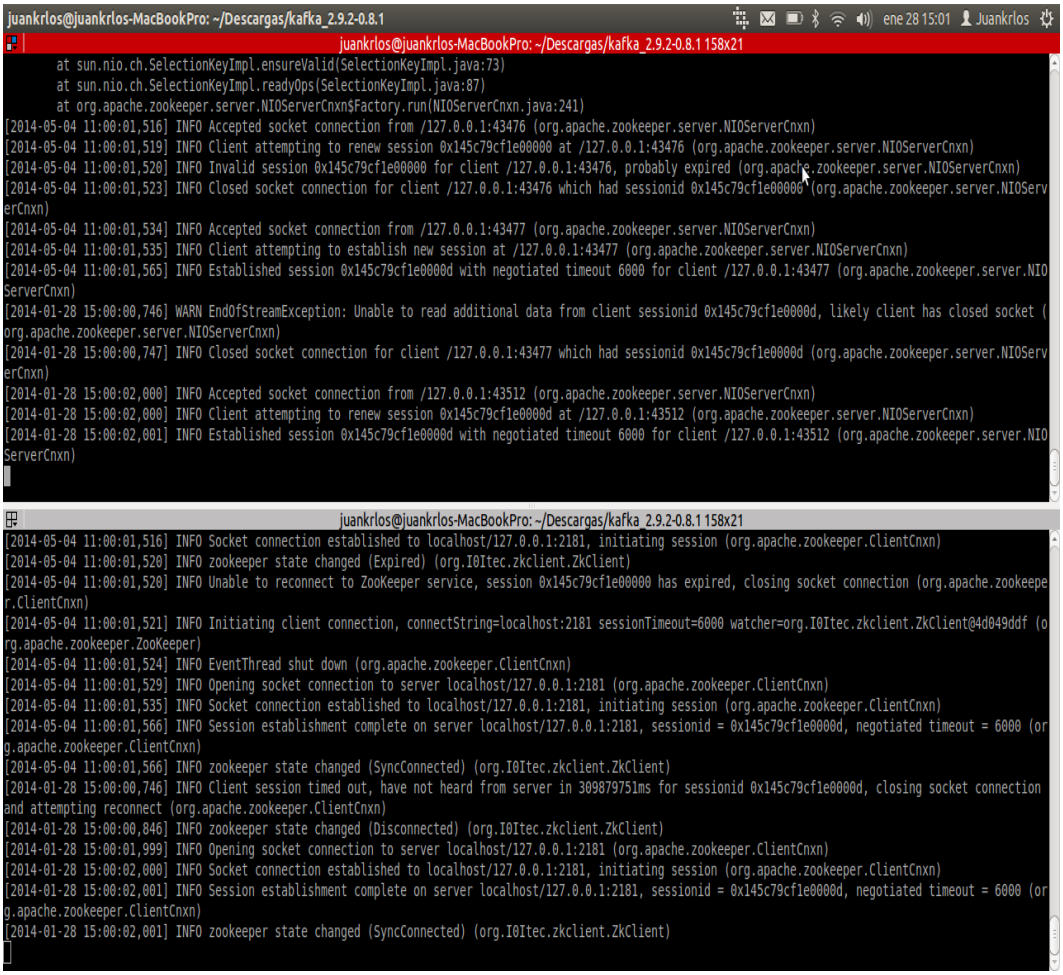


Figura 6.11 Arrancando el servicio de Kafka.

Descripción: En la parte superior el servicio embebido ZooKeeper arranca primero escuchando conexiones futuras por medio de Kafka que se aprecia en la parte inferior.

6.3.4 Iniciando Grails app PluSoft

Para poner nuestra aplicación basada en Spring MVC framework lo que debemos hacer es abrir una terminal, ubicarnos en el directorio de ubicación de nuestra aplicación y escribir el siguiente comando:

> grails run-app

Si la aplicación ha sido desarrollada correctamente el resultado será el siguiente:

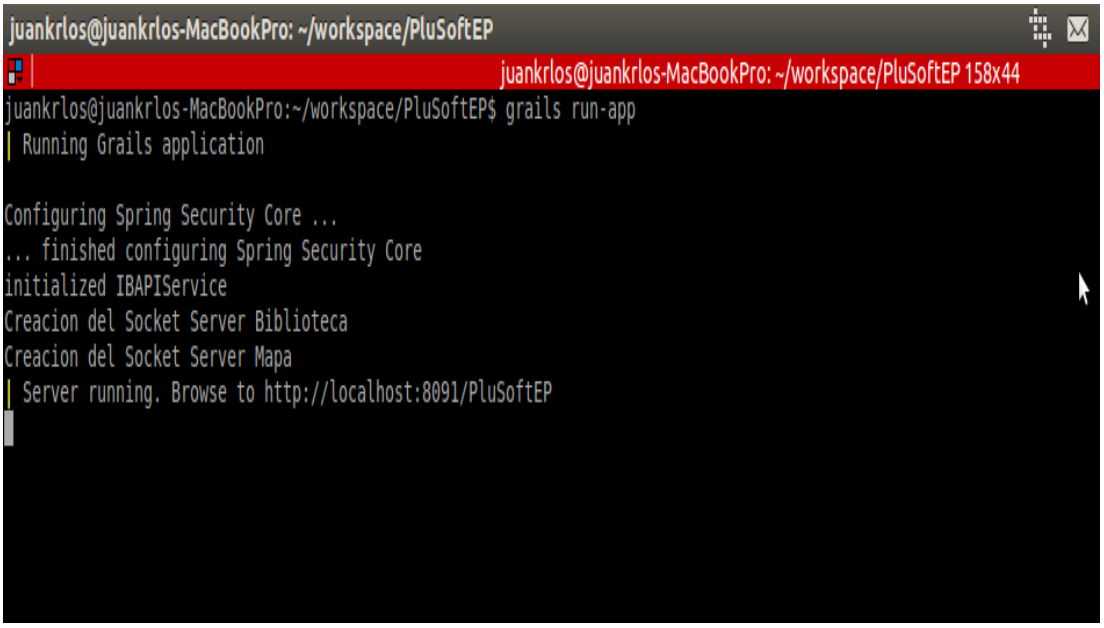


Figura 6.12 Iniciando Grails Services Aplication.

Una vez que la aplicación en Grails ha arrancado los servicios para nuestro Back-End en Node.js tanto para el mapa y administración podrán conectarse y consumir estos servicios.

6.3.5 Iniciando Node.js Server

Para arrancar nuestro servidor Node.js basta con ubicarnos en el path de nuestro proyecto y teclear el siguiente comando:

> node app.js

Donde **node** hace referencia al servidor y **app.js** a un nombre reservado para la aplicación main de un proyecto en Node.js, esto se debe a estándares que se manejan en el mundo de programación con JavaScript.

Si todo marcha bien y todas las librerías han sido instaladas correctamente y el servidor correctamente configurado el resultado debería ser el siguiente:

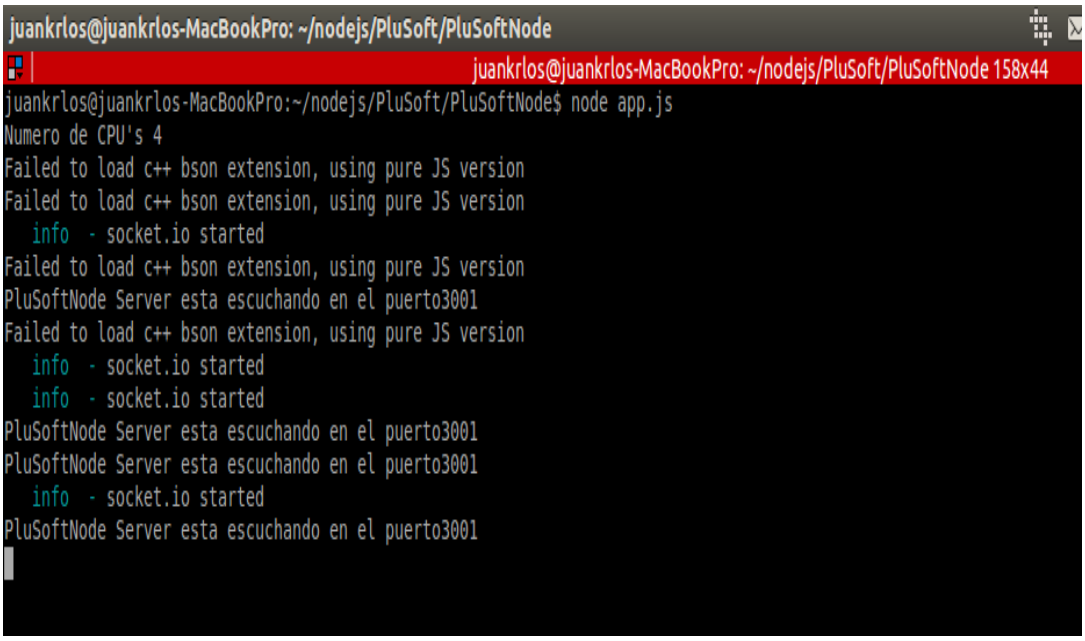


Figura 6.13 Arrancando PLuSoftNode Server.

Una vez que el servidor en Node.js arranca inmediatamente establece conexión con el Core de la aplicación en Grails consumiendo los servicios y escribiendo datos en la base de datos.

6.4 Pruebas

6.4.1 Pruebas de rendimiento y análisis

La fase de pruebas y rendimiento es de suma importancia ya que aquí se ve reflejado que tan óptimo ha sido el desarrollo de la aplicación, tanto en instalación, configuración y codificación.

Para la parte de pruebas y rendimiento se ha seleccionado una herramienta propiamente diseñada para un entorno de desarrollo basado en JavaScript, específicamente bajo el servidor Node.js.

La aplicación tiene como nombre NodeTime una app perteneciente a AppDynamics Company.

6.4.1.2 Nodetime

Nodetime es un servicio de monitoreo de rendimiento de la aplicación y análisis. Es una solución completa para el monitoreo, perfiles y solventar problemas de rendimiento de aplicaciones Node.js

El perfilador de funcionamiento dentro de la aplicación envía de forma segura los datos del perfil en el servidor Nodetime, que lo envía a un explorador en tiempo real, AppDynamics Company es una multinacional con sede en varios países como USA, Alemania, Francia, etc. Provee software de análisis de rendimiento y pruebas de sistemas en tiempo real. Provee aplicaciones de gran escalabilidad con soporte mundial. Nodetime es una aplicación con licencia privativa pero con un soporte especial para los usuarios registrados con periodos de prueba, el periodo de prueba es de 14 días, suficiente para probar el rendimiento de aplicaciones pequeñas y/o mediana. [30]

Suscripción

Nodetime ofrece suscripciones gratuitas y de pago. Todos los planes tienen el mismo conjunto de características y funcionalidades. La diferencia entre los planes es el número de agentes permitidos por cuenta para enviar datos simultáneamente a Nodetime, en donde el agente es el módulo Nodetime de funcionamiento dentro de un proceso de nodo.

Versión Free

El periodo de prueba en la versión Free es de 14 días con opción a probar el rendimiento de **1** aplicación solamente, si se registra el **Key** de la cuenta de prueba en más de una app de Node.js la cuenta se bloquea automáticamente, así que es importante saber que por cada cuenta se puede comprobar el rendimiento de una aplicación, de esta forma nos evitamos inconvenientes.

Una vez que el usuario se ha registrado los pasos a seguir para el monitoreo de nuestro sistema son muy sencillos y se los detallan a continuación:

Paso 1 Instalación

En una terminal de Linux escribimos el siguiente comando:

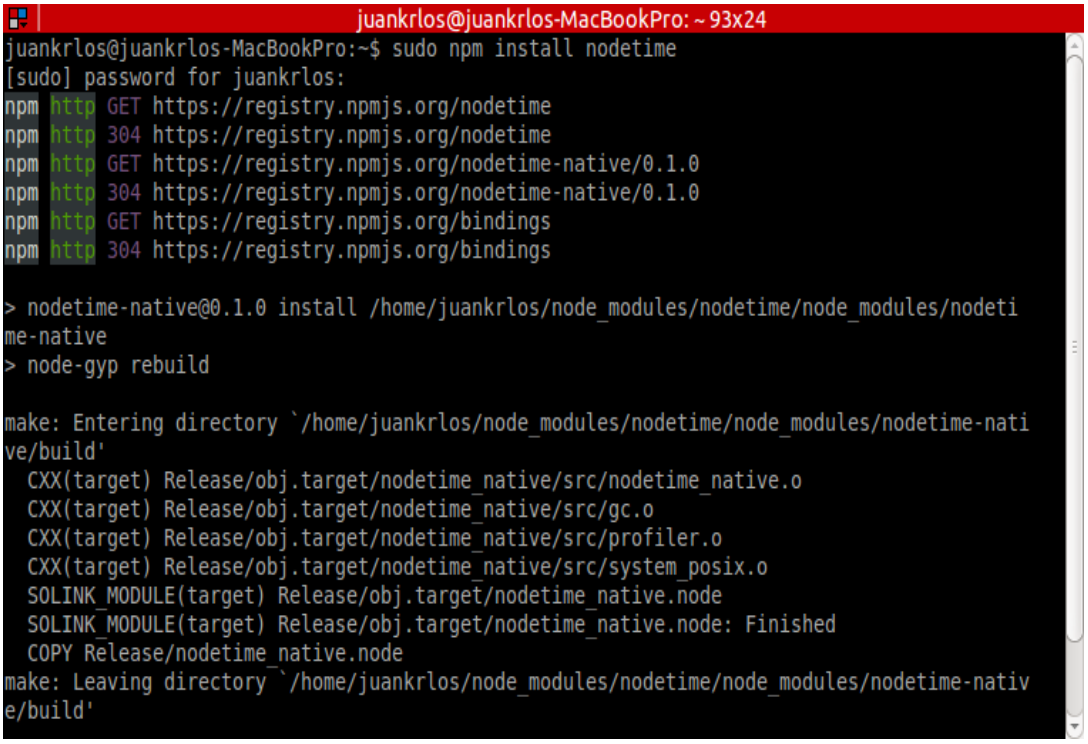


Figura 6.14 Ejecución de comando sudo npm install nodetime

Paso 2 Adherir código al proyecto

Antes de cualquier require o al inicio de cada clase en nuestra aplicación y principalmente en el main (app.js en este caso) hay que colocar el siguiente fragmento de código.

```
require('nodetime').profile({
  accountKey:   your_account_key
});
```

De esta forma el programa invocará a Nodetime con el accountKey correspondiente.

Paso 3 Analizar

El paso final es acceder a nuestra cuenta Nodetime con el correo como nombre de Usuario y su contraseña, automáticamente se ha creado un link con el nombre de nuestra aplicación, el mismo que nos llevará a la consola de monitoreo.

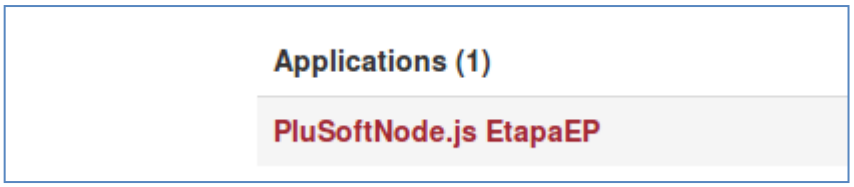


Figura 6.15 Nuestra aplicación en la cuenta de Nodetime.

Con eso estamos listos para monitorear la actividad del sistema y su rendimiento.

6.4.2 Nodetime Transacciones y Operaciones

El perfilador de funcionamiento dentro de la aplicación envía de forma segura los datos del perfil en el servidor Nodetime y son enviados a un explorador en tiempo real.

El área para la apreciación del rendimiento de la aplicación es amigable y a su vez avanzada, intuitivamente se puede navegar por las opciones que ofrece en pantalla.

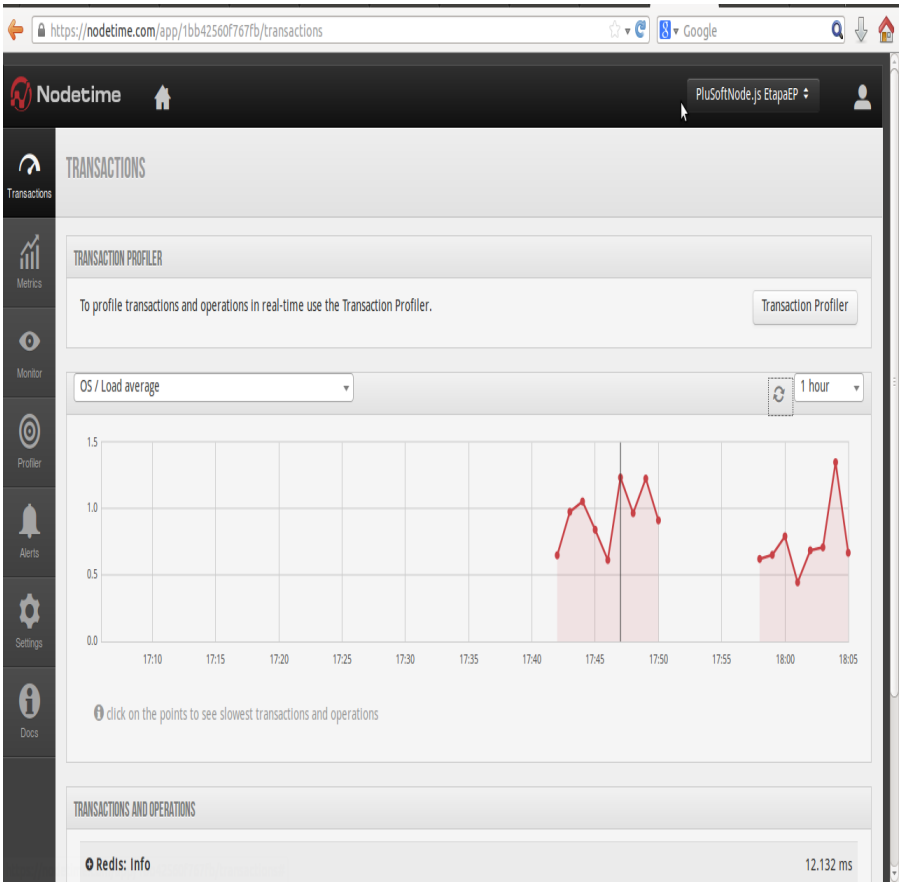


Figura 6.16 Nodetime página de análisis

6.4.2.1 Áreas

Nodetime ofrece un abanico de opciones para análisis muy interesante, a continuación se describen algunas de sus principales características.

6.4.2.2Transaction Teacing (Rastreo Transaccional)

Una característica fundamental de Nodetime, es la que da visibilidad de lo que está sucediendo dentro de la aplicación, por una parte se muestran las solicitudes y las operaciones, por ejemplo, llamadas a la API (con una gran cantidad de información incluyendo el tiempo de respuesta), tiempo de CPU, operaciones (como las llamadas a bases de datos), las solicitudes de cliente http y otras llamadas a la API que ocurrieron al mismo tiempo. Todo esto se completa con información de contexto relacionado y el estado de la aplicación.

6.4.2.3CPU Profiling (Perfilado de CPU)

Como siguiente paso lógico después de detectar un uso de CPU, Nodetime hace que sea increíblemente fácil de usar el muestreo incorporado de CPU sobre el perfilador de V8 para analizar los puntos calientes (gran procesamiento) y localizar funciones ineficientes.

6.4.2.4 Perfilado de Memoria

Nodetime usa una función como generadora de perfiles para generar gran cantidad de datos y a su vez hace que esta gran cantidad de datos pase sobre el compilador de Node.js que es V8, el perfilador permite tomar instantáneas del procesamiento de esta información.

Esto es muy importante para ver cómo responde la memoria del sistema ante el consumo de recursos de la aplicación.

6.4.2.5 Alertas

Uno de los módulos más importantes es el de alertas, una opción bastante útil al momento de evaluar sistemas en producción. Nodetime también ofrece esta opción.

Es importante que se notifique cuando una aplicación está experimentando problemas de rendimiento con el fin de evitar tiempos de inactividad y ser capaz de localizar rápidamente la causa raíz del problema, mientras que los síntomas del problema identifican perfiles exactos, los cuales podrían desaparecer más tarde.

6.4.3 Pruebas de Rendimiento

En Ingeniería de Software el objetivo primordial que se persigue es el de producir un sistema, api, o producto en general, de alta calidad. Para alcanzar este objetivo los Ingenieros de Software hacen uso de herramientas modernas con métodos de alta efectividad.

Al igual que para cualquier desarrollador de software es importante ofrecer aplicaciones de calidad que cumplan con los estándares de desarrollo y además que cumplan con las expectativas del cliente y/o Usuario.

Habría que encontrar entonces una definición de calidad, calidad se la puede definir como la característica de algo, o una serie de atributos que distinguen a algo.

La calidad de un sistema o producto se lo mide también por la capacidad que tiene para satisfacer las necesidades y resolver los problemas planteados desde un inicio, incluso antes de desarrollar el sistema en sí.

Aunque existen muchas medidas para la calidad del software, la facilidad de mantenimiento, integridad, disponibilidad, etc. Son consideradas como primordiales en un sistema.

6.4.3.1 Equipo que actuó como Servidor

El equipo que actuó como un servidor centralizado fue el siguiente:

MacBookPro core i7

Dual Core de 2.9 Ghz

8 GB Ram

DDR 3 de 1600 Mhz

SO Linux Ubuntu 12.04

Para el sistema propuesto se han realizado pruebas de rendimiento puntuales con la respectiva herramienta antes especificada, cuyos resultados son expuestos a continuación.

6.4.3.2 Nodetime Transaction

Nodetime se basa en pruebas de enganche hacia el servidor haciendo llamadas a la API y estas a su vez con sus respectivas devoluciones por medio de callbacks.

Un callback es una función que se encarga de determinar si un proceso se ha cumplido y no libera el hilo del proceso hasta que el mismo haya sido completado para esto envía un mensaje de confirmación y se procede a ejecutar la tarea o proceso que sigue.

Nodetime mide el tiempo de respuesta, agrega variables y crea objetos y todo esto de forma natural y transparente para la aplicación. Un proceso significativo en el proceso de sobrecarga es la lectura de trazas de la pila de operaciones incluidas en cada muestra.

6.4.3.3 Métricas

Los resultados obtenidos al ejecutar el módulo Métricas, para un límite de una hora de análisis de rendimiento fue la siguiente: Hay que recalcar que esta prueba se realizó en periodos cortos para una cantidad de usuarios limitada de hasta 10.

Memoria libre durante la ejecución del servidor:

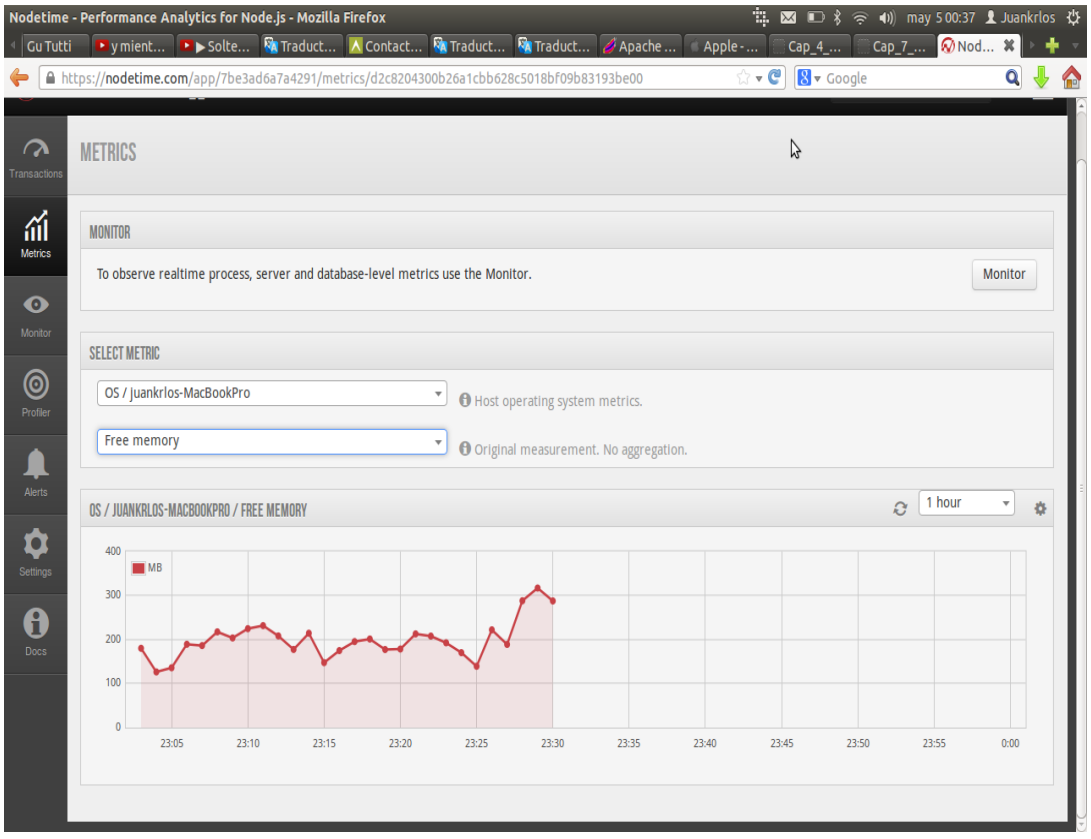


Figura 6.17 Resultados de aplicar el módulo de métricas.

Tomando en cuenta que la máquina que actúa como servidor se encarga de todos los servicios del sistema tales como Apache Kafka, Apache ZooKeeper, Spring frame, Node.js, etc. El resultado obtenido en memoria libre muestra que se está consumiendo cerca de 300 MB en promedio, esto es positivo para la disponibilidad de recursos en el sistema.

6.4.3.4 Monitor Server

El registro de la actividad para el servidor de aplicaciones también fue positivo, tomando en cuenta que la mayor cantidad de solicitudes al servidor se da por Socket.io y .net para la comunicación del Back-end en Node.js con el Back-end en grails, los valores obtenidos muestran un registro de actividad por debajo del 100% de consumo y en otros casos muy debajo de este valor.

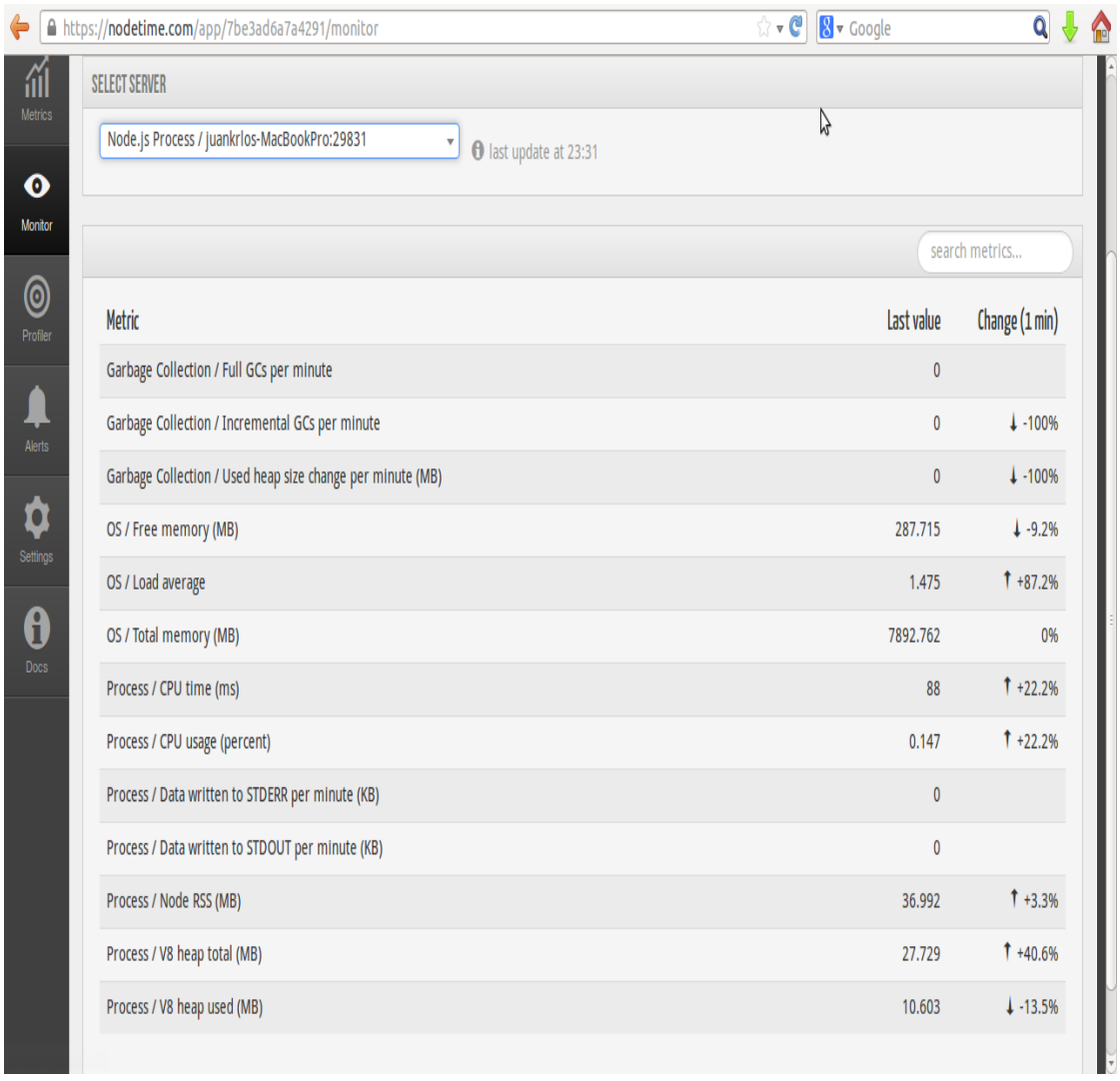


Figura 6.18 Resultados con Monitor Server.

Los resultados que se destacan en este análisis son:

Process / CPU time (ms): con Last Value de 88 y un tiempo de respuesta positivo de +22,2%.

El uso del procesador en la atención de cada solicitud **Process / CPU usage (percent)** fue de también +22,2% corroborando la del tiempo de respuesta.

Mientras tanto los callback procesados por el compilador V8 de Node.js muestran también una actividad favorable con el uso de cabecera por debajo del 10% y el **heap total** con +40.6%, esto demuestra una alta disponibilidad en el sistema.

Process / Node RSS (MB)	36.992	+3.3%
Process / V8 heap total (MB)	27.729	+40.6%
Process / V8 heap used (MB)	10.603	-13.5%

Tabla 6.7 Resumen de resultados.

6.4.3.5 Transaction Profiler (Analizador de Transacciones)

Como parte del análisis de resultados en las pruebas realizadas los resultados en el analizador de transacciones fue positivo, los tiempos de respuesta están dentro del límite tolerable y la actividad no muestro errores en el tiempo de ejecución.

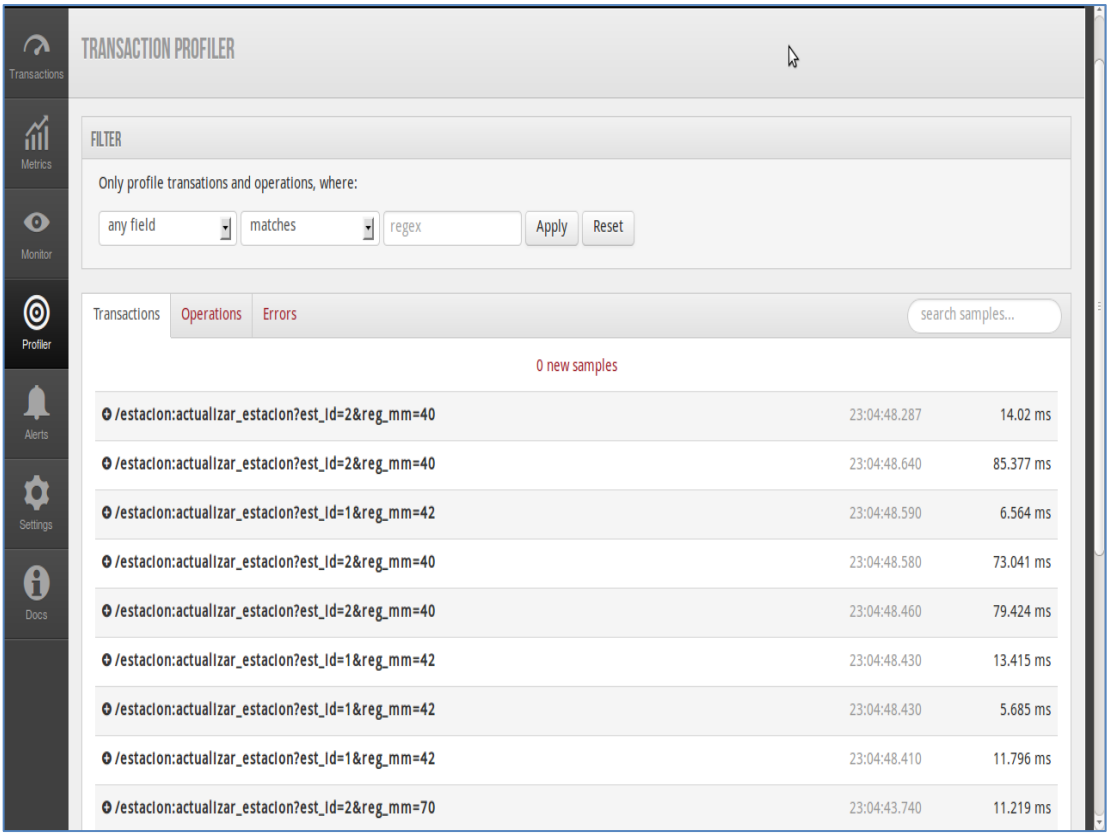


Figura 6.19 Análisis de transacciones.

6.4.3.6 Solicitudes y tiempos de respuesta

Para la estación con id 2 un value reg_mm=40 el tiempo de respuesta es de 14.02 ms con un Request Header apuntando al puerto 3001 se obtiene un Status code 200, esto quiere decir que el recurso está disponible y la solicitud fue atendida correctamente.

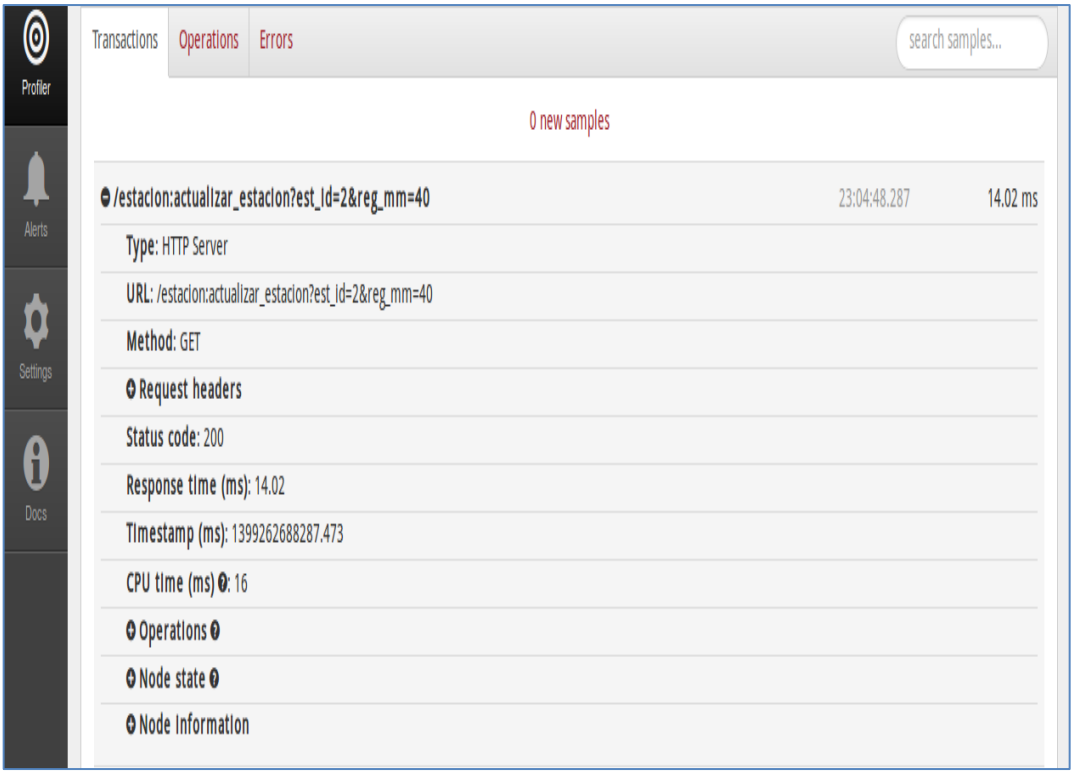


Figura 6.20 Tiempo de respuesta para estación con id=2.

Un CPUtime (ms) igual a 16 refleja los resultados que se obtuvieron en la prueba anterior, al parecer el tiempo de procesamiento en el servidor está siendo algo lenta en comparación a los resultados obtenidos, pero no dejan de ser favorables para el sistema.

En el mismo marco de prueba en esta ocasión para la estación con id=1 con un valuereg_mm=42 el tiempo de respuesta aumenta considerablemente hasta los 6.564 ms, es una respuesta que se esperaba debido a que el flujo de datos es ligero y no se transporta mayor cantidad de información por medio de la red.

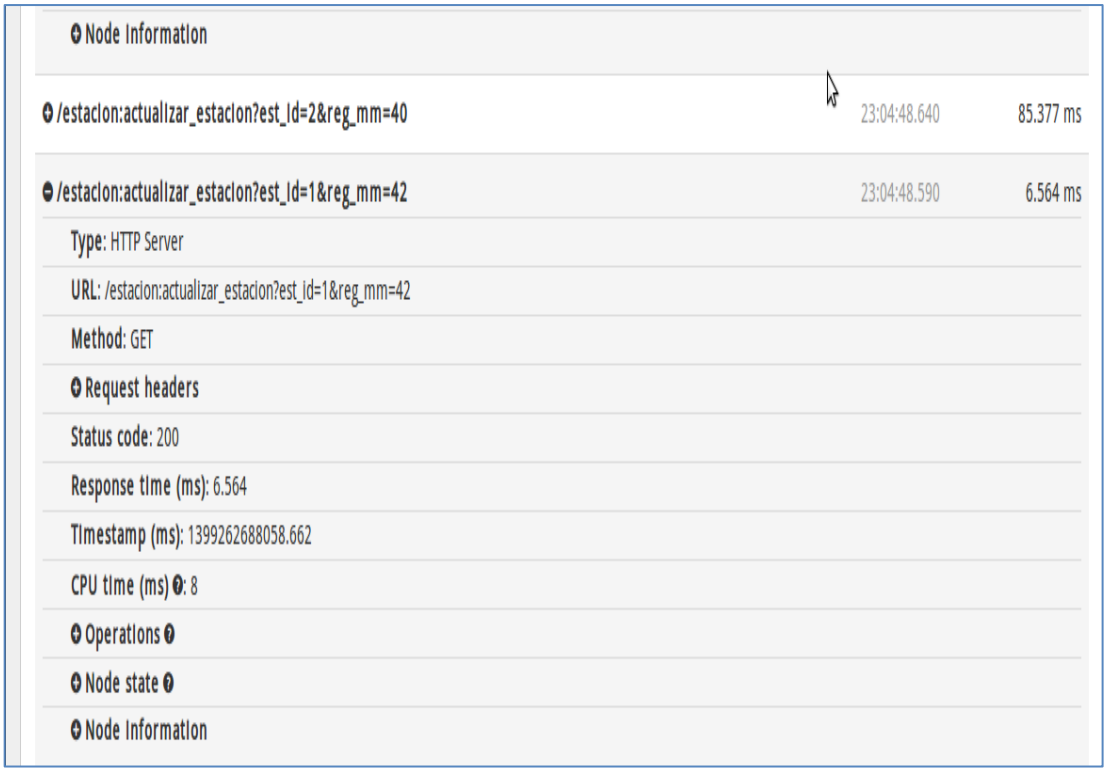


Figura 6.21 Estación con id=1.

6.4.3.7 Operaciones

En el monitoreo de operaciones dentro del analizador de transacciones los resultados han sido favorables con tiempos de respuesta casi óptimos, los métodos PUT y GET tienen tiempos de respuesta de entre 1 a 4 ms para el transporte de información ligera, como es el caso de push en la biblioteca de estaciones, y en cambio el tiempo de respuesta disminuye cuando se realizan consultas de históricos cuyo tiempo bordea los 42ms de respuesta, los resultados adicionales se los puede apreciar a continuación:

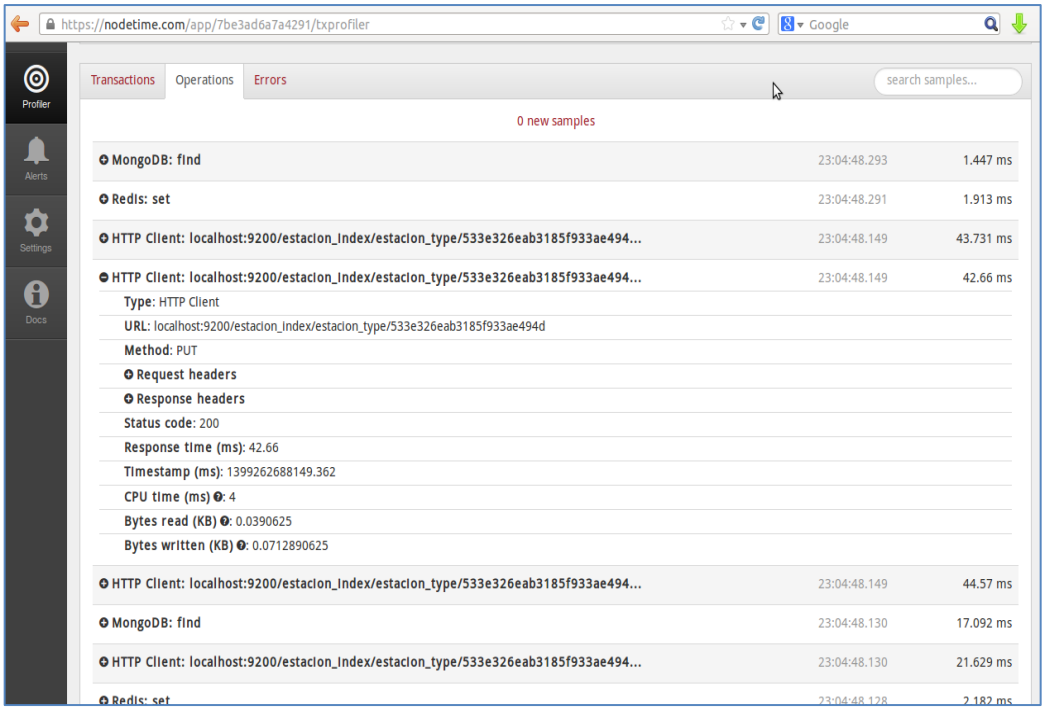


Figura 6.22 Tiempo de respuesta disminuye cuando se realizan consultas de históricos.

6.4.3.8 Errores

No se han registrado errores en las pruebas realizadas, la prueba de errores corresponde al monitoreo de transporte de datos en el uso de los métodos PUT, GET y POST dentro de la aplicación, siendo este resultado positivo.

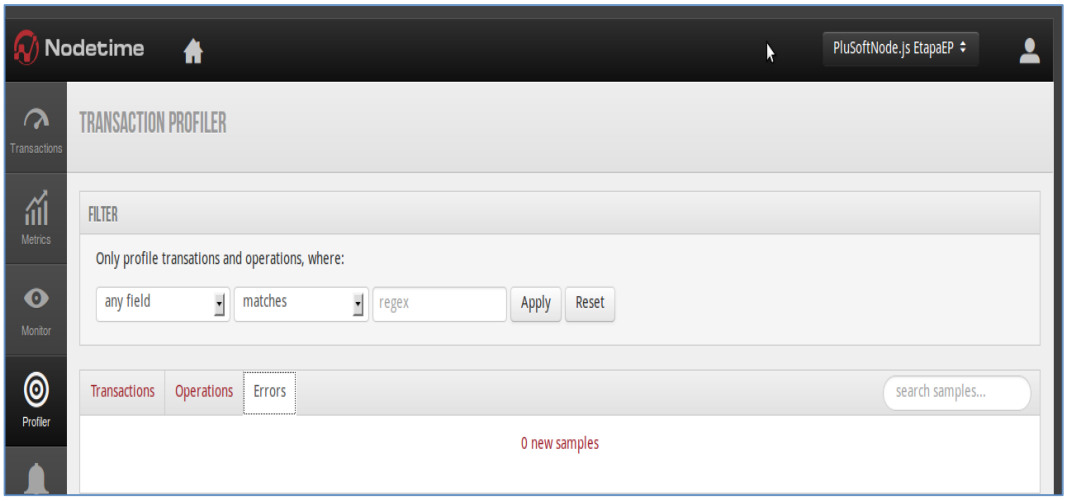


Figura 6.23 Resultados de errores.

6.4.3.9 Alertas

El software que se seleccionó como herramienta de pruebas **Nodetime** ofrece un servicio de alerta, área que se aprecia a continuación:

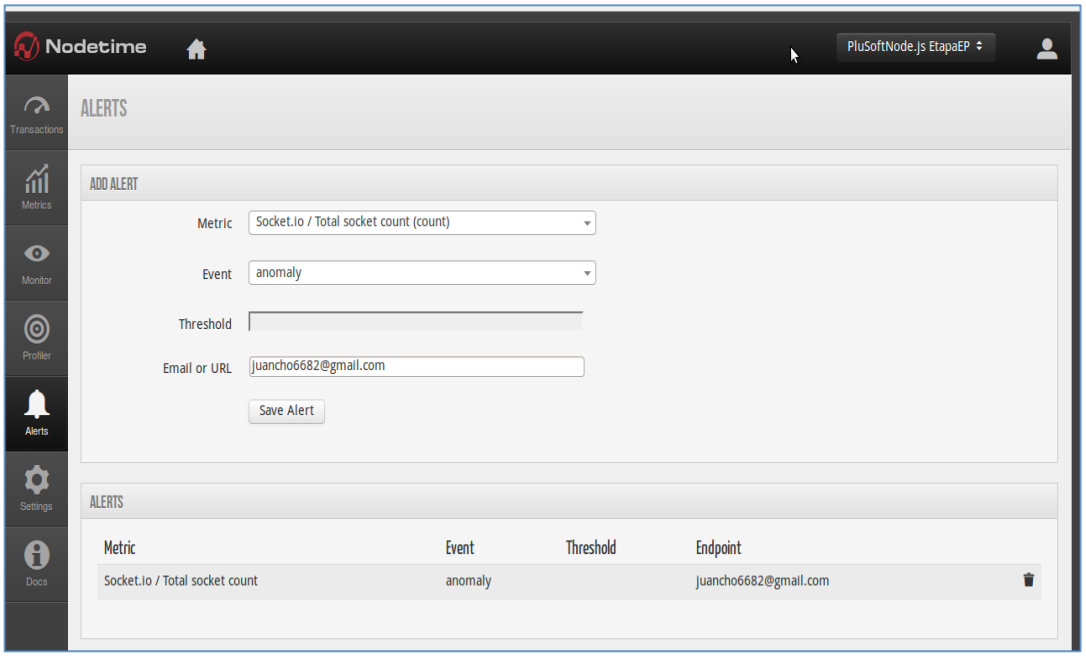


Figura 6.24 Alertas

En el área de Alertas se permite configurar la métrica a monitorear. La métrica que interesa monitorear dentro del sistema es la de Socket.io, debido a que se transporta información en tiempo real, el evento es **anomaly** y los avisos de una alerta generada serán enviados a un correo personal.

No existieron alertas durante los periodos de prueba, esto es un resultado positivo al verificar la calidad del producto.

6.4.3.10 Lista de alertas programadas

Para explotar el sistema de monitoreo al máximo se programó una lista de alertas las cuales enviarán un mensaje de correo electrónico en caso de que alguna de una alerta en la lista se dispare, la lista se la detalla a continuación:

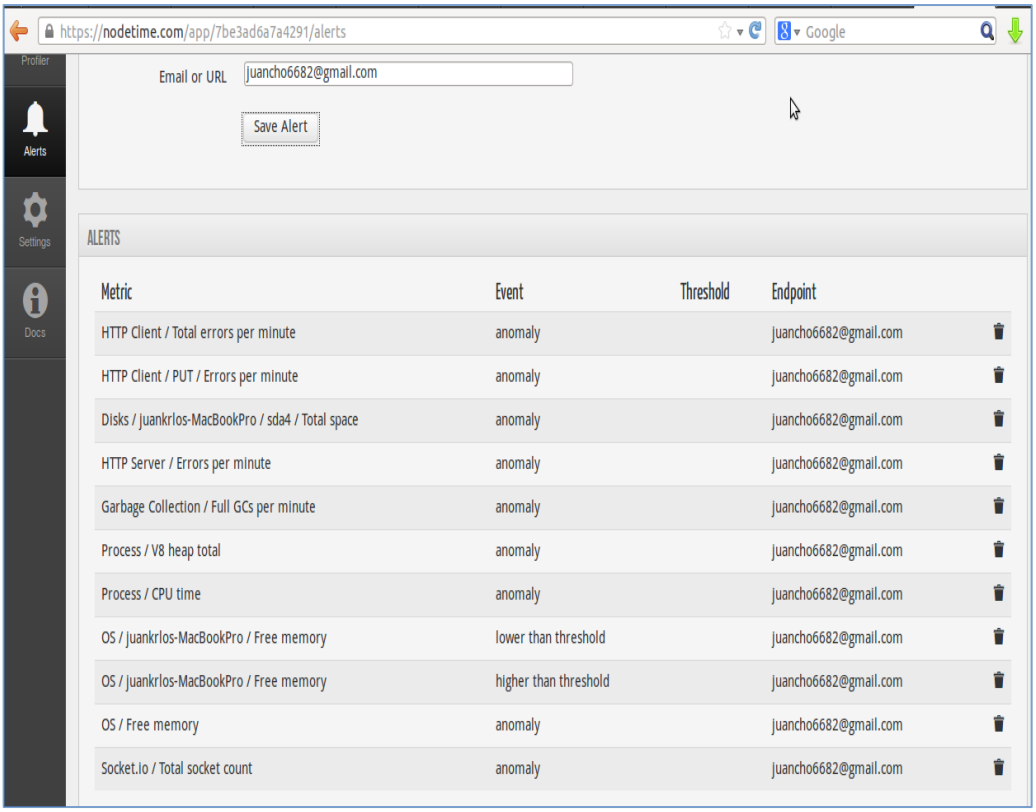


Figura 6.25 Lista de alertas programadas.

- Como conclusión de las pruebas realizadas al Sistema PluSoftNode entregaron resultados positivos con tiempos de respuesta que se encuentran dentro del límite tolerable por debajo de 50 ms.

- Los tiempos elevados de respuesta se deben a la gran cantidad de información que se maneja en el sistema, específicamente en la consulta de los datos históricos para cada estación.

- El sistema de alertas no presentó ningún mensaje de error y esto supone un buen desempeño de la aplicación, es decir la utilización de buenas prácticas en la escritura del código fuente.

Conclusiones

El trabajo de investigación aquí expuesto, el cual tuvo como objetivo principal analizar, diseñar e implementar un Sistema de alarma para el monitoreo del registro de la lluvia en la ciudad de Cuenca, basado en el protocolo GPRS ha sido alcanzado y podemos concluir:

- La experiencia y el conocimiento que deja haber recorrido un gran camino de investigación y de haber trabajado con personas especializadas en el tema, nos ha permitido comprender mejor el panorama del ámbito profesional y técnico, además desarrollar las aptitudes que permitan llevar a cabo un trabajo de investigación netamente científico y poder solucionar problemas reales de la sociedad.

- El sistema desde su concepción hasta su elaboración y previa aprobación cumplió con los objetivos planteados, tanto desde la recolección de la información hasta el tratamiento de la misma, por medio de tecnología aprendida como la encontrada en la fase de investigación la cual se fue acoplando a nuestras necesidades y al mismo tiempo despertando nuestro interés y haciendo que la curva de aprendizaje se reduzca en comparación al reto que implica aprender e implementar dichas herramientas en este proyecto de investigación.

- La tecnología expuesta en este trabajo es netamente investigativa y que va ganando un amplio espacio en el desarrollo de software. En el área del desarrollo web es una tecnología de vanguardia. En base a estas características confiamos que la implementación de este sistema será exitosa.

- El uso de software y hardware libre que ayudaron a concluir con éxito el trabajo amplía el alcance del proyecto en el ámbito legal y profesional, en vista a futuras implementaciones en el campo del tratamiento medioambiental.

- La motivación que llevó a la implementación y conclusión del Sistema fueron siempre la sociedad, la responsabilidad con la naturaleza y el cumplir el sueño de aportar a la seguridad en nuestra comunidad, las motivaciones se cumplieron en un deber y el deber ha sido realizado con éxito.

Del prototipo de alarma electrónica podemos concluir que:

- La implementación de nuestro prototipo basado en la tecnología Arduino ha sido una experiencia satisfactoria, un componente fundamental para cumplir con los objetivos planteados ya que se trata de una herramienta flexible y confiable, no presentó problemas en su funcionamiento, el rendimiento obtenido fue óptimo además su característica de hardware libre permiten una implementación sin barreras.
- Cuando se llega a dominar esta tecnología, llegamos a tener en nuestras manos la posibilidad de desarrollar prototipos con un amplio campo de aplicaciones.

De la relación con la empresa:

- La idea de implementar un Sistema con las características aquí presentadas fue concebida en una reunión de amigos, presentada a varias empresas y entes públicas que creíamos podían intervenir para que este proyecto no quedara solamente en ideas, la Empresa Pública ETAPA-EP a través de la Subgerencia de Operaciones de Agua Potable y Saneamiento, depositó su confianza en nuestra visión y creyeron hasta el final del gran alcance y beneficio que puede traer consigo la implementación real de nuestro prototipo, y así fue, la relación luego de la presentación del trabajo realizado ha sido gratificante y augura grandes éxitos para ambas partes en relación a un gran proyecto que pudo consolidarse y finalizarse.

- Se alcanzó con éxito el objetivo de ofrecer a la empresa una alternativa al software privativo, una alternativa de alta calidad con software libre y que cubra expectativas, además de ofrecer un significativo ahorro en inversión para la empresa pública.

Recomendaciones

Las recomendaciones que se deben tomar en cuenta son las siguientes:

- No cabe duda que existen muchas cosas por mejorar, ya que todo es mejorable, nuestro anhelo es que con el trabajo conjunto y el apoyo mutuo se alcancen muchos logros y que el beneficio abarque a todos en la ciudad y demás lugares que puedan necesitar un tipo de sistema como el que se ha sido expuesto aquí.

- Si el sistema llegara a crecer, sería muy conveniente implementar un sistema de mensajería mucho más completo como un servidor de mensajería, que debido al alto costo de un equipo Gateway emisor de mensajería a gran escala aquí no pudo ser implementado.

- Se deben continuar los esfuerzos para prolongar el estudio de los afluentes que rodean la ciudad, involucrar a más estaciones recolectoras de datos y hacer crecer el repositorio de información, de esta manera se podría implementar un sistema especializado de alerta temprana.

- Se recomienda seguir utilizando las herramientas de desarrollo aquí expuestas, ya que son libres y permiten seguir creciendo sin obstáculos financieros y legales.

- Es recomendable buscar el financiamiento y apoyo para implementar este Sistema en lugares que son vulnerables a inundaciones y desbordes de ríos, zonas costeras que se ven afectadas por este tipo de eventos naturales, darle continuidad a este proyecto el cual tiene un gran alcance y es muy prometedor.

Glosario de términos

ETAPA EP: Empresa Municipal de Telecomunicaciones, Pública de Agua Potable y Saneamiento.

INAMHI: Instituto Nacional de Meteorología e Hidrología.

INOCAR: Instituto Oceanográfico de la Armada.

HTML: Lenguaje de marcas de Hipertexto.

CSS: Hoja de estilos en casada. Describe aspecto y formato de un documento.

IDE: Entorno de desarrollo integrado. Interfaz de programación.

NetBeans: Es un IDE, desarrollado principalmente para el lenguaje de programación Java.

API: Interfaz de programación de aplicaciones. Es un conjunto de funciones y procedimientos. Ofrece cierta biblioteca de la cual se puede tomar dichas funciones o procedimientos.

GPRS: significa General Packet Radio Service o en español Servicio General de Paquetes por Radio.

GSM: Sistema Global para las telecomunicaciones móviles.

Gateway: Puerta de enlace. Permite interconectar redes con protocolos y arquitecturas diferentes.

LAN: Red de Área Local.

WAM: Red de área metropolitana.

GPL: Licencia publica general. Licencia de software libre.

Microcontrolador: circuito integrado programable, ejecuta ordenes gravadas en su memoria.

C: Lenguaje de programación orientado a la implementación de sistemas operativos.

AMRA: Analysis and Monitoring of Environmental Risk.

SIG: Sistema de Información Geográfica.

SSL: Secure Sockets Layer.

GNU LGPL: Lesser General Public License (Licencia Pública General Reducida)

Bibliografía

[1] Franciso J Aparicio Mijares, *Fundamentos de Hidrología de Superficie*. Mexico, D.F.: LIMUSA , 1992.

[2] Diego Cáceres. (2013, Febrero) eltiempo.com.ec. [Online]. (Recuperado: 10 de Mayo del 2013) http://www.eltiempo.com.ec/noticias-cuenca/115529-ra-o-yanuncay-se-desborda-por-lluvias/

[3] C.S.M. (2013, Marzo) El Mercurio. [Online]. (Recuperado: 29 de Marzo del 2013)http://www.elmercurio.com.ec/371617-latente-riesgo-de-inundaciones-en-la-urbe/#.UxfbrIXGVSE

[4] CSM. (2013, Mayo) El Mercurio. [Online]. (Recuperado: 22 de Septiembre del 2013) http://www.elmercurio.com.ec/382392-previsiones-para-prevenir-inundaciones/#.UxgFooXGVSE

[5] El Tiempo. (2013, Octubre) ELTiempo. [Online]. (Recuperado: 5 de Enero del 2014) http://www.eltiempo.com.ec/noticias-cuenca/131524-fuerte-aguacero-en-la-ciudad-provoca-inundaciones/

[6] Raúl Calixto Flores and Lucila Herrera Reyes, *Ecología y Medio Ambiente*. México D.F.: Editorial Progreso S.A.

[7] German Monsalve Sáenz, *Hidrología en la Ingeniería*. Bogotá: Tercerr Mundo Editores, 1995.

[8] Rosa María Rodríguez Jiménez, Águeda Benito Capa, and Adelaida Portela Lozano, *Meteorología y Climatologíia*. Madrid: FECYT, 2004.

[9] Oscar Francisco Fayad Ospina, ESTUDIO TECNICO ECONOMICO DE SISTEMAS DE UBICACION Y RASTREO VEHICULAR APLICADOS AL MANTENIMIENTO DE FLOTAS, 2006.

[10] CONFEDERACIÓN HIDROGRÁFICA DEL EBRO. (2002, Enero) SAIHEbro. [Online]. (Recuperado: 10 de Enero del 2014). http://www.saihebro.com/saihebro/img/flash/funcionamiento_pluviometro.swf

[11] Lawrence Letham, *GPS fácil*. Barcelona: Paidrotibo, 2001.

[12] Eduardo Huerta, Aldo Mangiaterra, and Gustavo Noguera, *GPS Posicionamiento Satelital*. Rosario: UNR Editora, 2005.

[13] Antonio Pérez Navarro, *Introducción a los sistemas de información geográfica y geotelemática*. Barcelona: Editorial UOC, 2011.

[14] Álvaro Pachón de la Cruz, "Evolución de los sistemas móviles celulares GSM," Universidad ICESI, Cali, Ensayo 2004.

[15] Juan Andrés Sánchez Wevar, Análisis y Estudio de Redes GPRS, 2005, Trabajo de Titulación para optar al Titulo de Ingeniero Electrónico.

[16] Ardilla M. Jorge, CARACTERÍSTICAS DE LA TECNOLOGÍA GPRS (GENERAL PACKET RADIO SERVICES), COMO EVOLUCIÓN DE LAS ACTUALES REDES GSM DE 2G, APLICADO A LAS REDES DE 3G, 2003, Tesis para Ingeniéro Eléctrico, Universidad de los Andes. Mérida-Venezuela.

[17] David Kushner. (2011, Octubre) IEEE SPECTRUM. [Online].(Recuperado: 20 de Febrero del 2014). http://spectrum.ieee.org/geek-life/hands-on/the-making-of-arduino

[18] Massimo Banzi, Tom Igoe, Gianluca Martino, and David Mellis. (2013, Noviembre) Arduino. [Online].(Recuperado: 20 de Marzo del 2014). http://arduino.cc/es/#.Uw1iIYXGVSE

[19] Jose Antonio Hernández Martínez, SISTEMAS Y SOLUCUINES PARA EL REGADIO, 2012, Tesis de Máster.

[20] Rafael Martinez Guerrero. (2013, Enero) PostgreSQL-es. [Online]. (Recuperado: 10 de Abril del 2014). http://www.postgresql.org.es/sobre_postgresql

[21] Michael Abernethy. (2011, Junio) developerWorks. [Online].(Recuperado: 20 de Marzo del 2014). http://www.ibm.com/developerworks/ssa/opensource/library/os-nodejs/#authorN1001D

[22] Ciberaula International Training. (2014, Enero) Ciberaula. [Online].(Recuperado: 5 de Abril del 2014). http://www.ciberaula.com/curso/lamp/que_es/

[23] Juergen Hoeller , Keith Donald , Colin Sampaleanu , Rob Harrop , Thomas Risberg , Alef Arendsen , Darren Davison , Dmitriy Kopylenko , Mark Pollack , Thierry Templier , Erwin Vervaet , Portia Tung , Ben Hale , Adrian Colyer , John Lewis Rod Johnson. (2014, Enero) Spring. [Online]. (Recuperado: 16 de Abril del 2014).

http://docs.spring.io/spring/docs/4.1.0.BUILD-SNAPSHOT/spring-framework-reference/htmlsingle/#overview-modules

[24] Grails. (2013, Enero) GRAILS by Pivotal. [Online]. (Recuperado: 10 de Abril del 2014). https://grails.org/learn

[25] Christian Bauer, Max Rydahl Andersen, Emmanuel Bernard, y Steve Ebersole Gavin King. (2004, Enero) HIBERNATE Community Documentation. [Online]. (Recuperado: 22 de Agosto del 2014). http://docs.jboss.org/hibernate/core/3.5/reference/es-ES/html/preface.html

[26] Luis Miguel Gracia. (2012, Diciembre) Un poco de Java. [Online]. (Recuperado: 11 de Abril del 2014). http://unpocodejava.wordpress.com/?s=APACHE+KAFKA

[27] The Apache Software Foundation. (2013, Enero) Zookeeper.apache. [Online]. (Recuperado: 21 de Marzo del 2014). http://zookeeper.apache.org/doc/r3.4.6/index.html

[28] Michal Čihař. (2014, Enero) [GW]ammu. [Online]. (Recuperado: 22 de Abril del 2014). http://es.wammu.eu/gammu/

[29] AMRA, "SERVICIOS DE CONSULTORÍA PARA EL PLAN DE GESTIÓN DE RIESGOS EN LOS SISTEMAS DE AGUA POTABLE Y ALCANTARILLADO DE LA CIUDAD DE CUENCA," ETAPA-EP, Nápoles, Informe de Consultoría EC-L1019, 2011.

[30] Nodetime. (2012, Enero) Nodetime. [Online].(Recuperado: 11 de Marzo del 2014). http://docs.nodetime.com/

[31] Pedro Ramos Castellanos, *Uso eficiente y sostenible de los recursos naturales*. Salamanca, 2007.

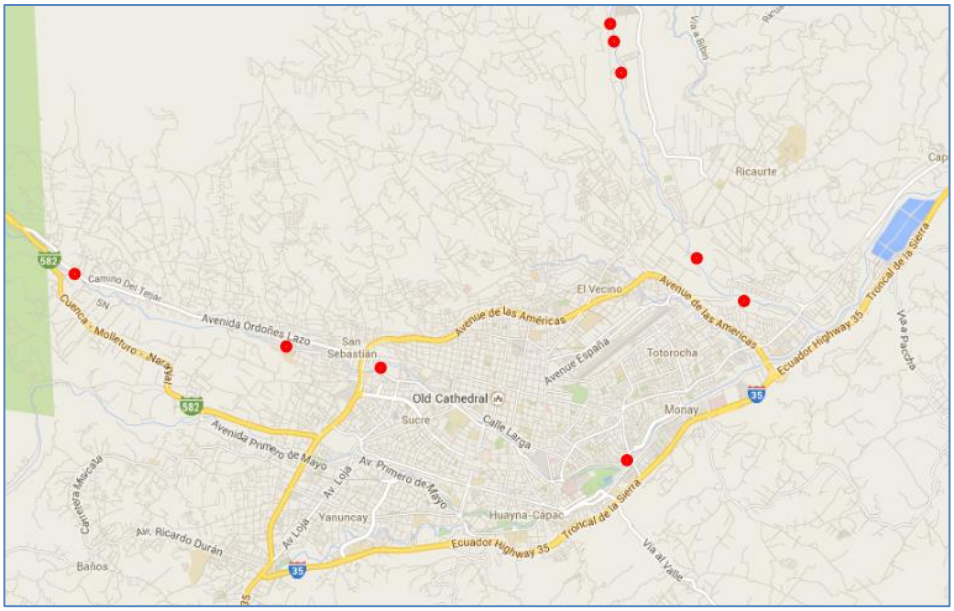
[32] María Carmen España Bosquera, *Servicios Avanzados de Telecomunicación*, 2003rd ed., Juan Bravo, Ed. Madrid, España: Díaz de Santos, S.A., 2003.

Anexos

A. Zonas susceptibles.

Río Tomebamba y río Machángara marcadas en Google Maps, imagen de satélite.

Río Tomebamba y río Machángara marcadas en Google Maps, imagen de Street.



B. Fotos de zonas susceptibles Río Machángara.

Fotos de PUNTO 1M,-2.840432,-78.986489. Paseo Río Machángara

Fotos de PUNTO 2M, -2.84249,-78.985784. Paseo Río Machángara

Fotos de PUNTO 3M, -2.847119,-78.984969. Paseo Río Machángara

Fotos de PUNTO 4M, -2.876308,-78.973117, Cornelio Veintimilla

Fotos de PUNTO 5M, -2.882844,-78.9658. Victoria del Portete.

C. Fotos de zonas susceptibles Río Tomebamba

Fotos de PUNTO 1T,- 2.878237,-79.068947, Av. Ordóñes Lazo.

Fotos de PUNTO 2T, -2.889809,-79.036553, Calle Paseo Tres de Noviembre.

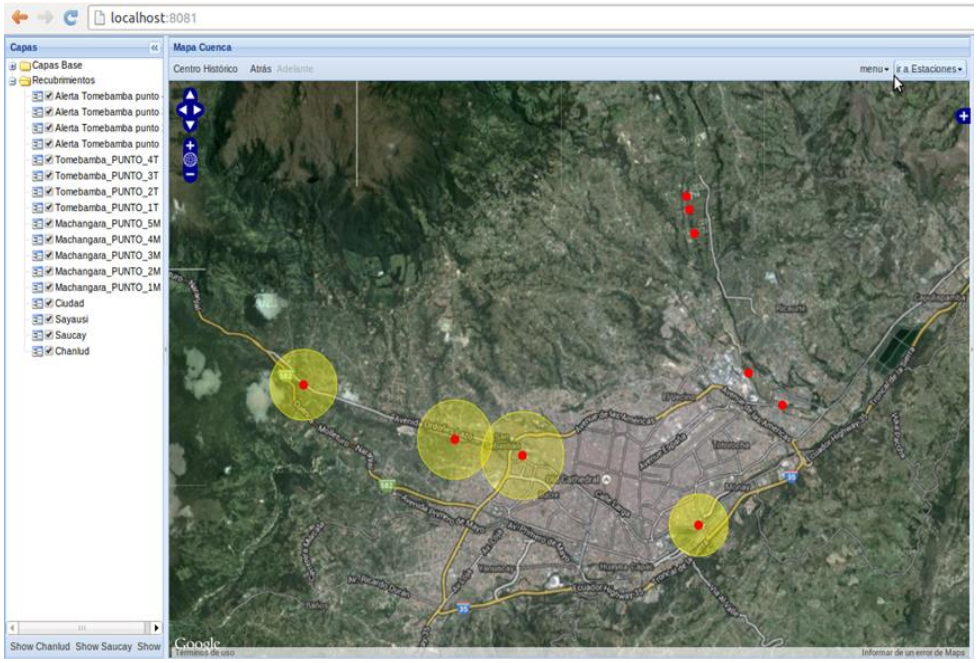
Fotos de PUNTO 3T, -2.893067,-79.021962. Av. Tres de Noviembre.


MUEBLES

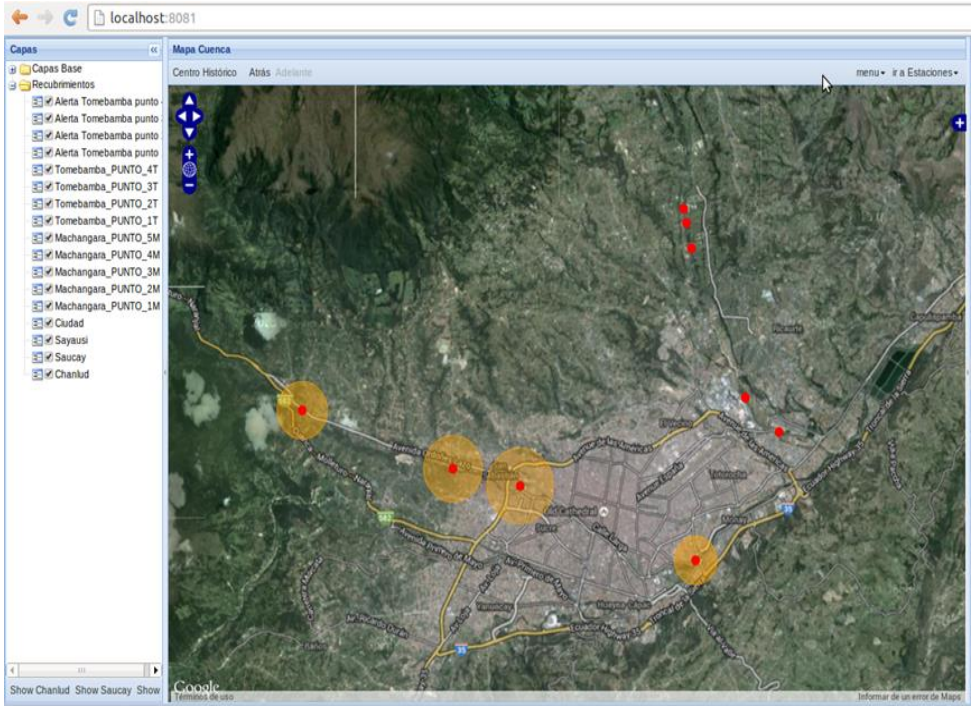
Fotos de PUNTO 4T, -2.90721,-78.983896, Av. Pumapungo - Urbanización Villanueva

D. Alarmas Río Tomebamba

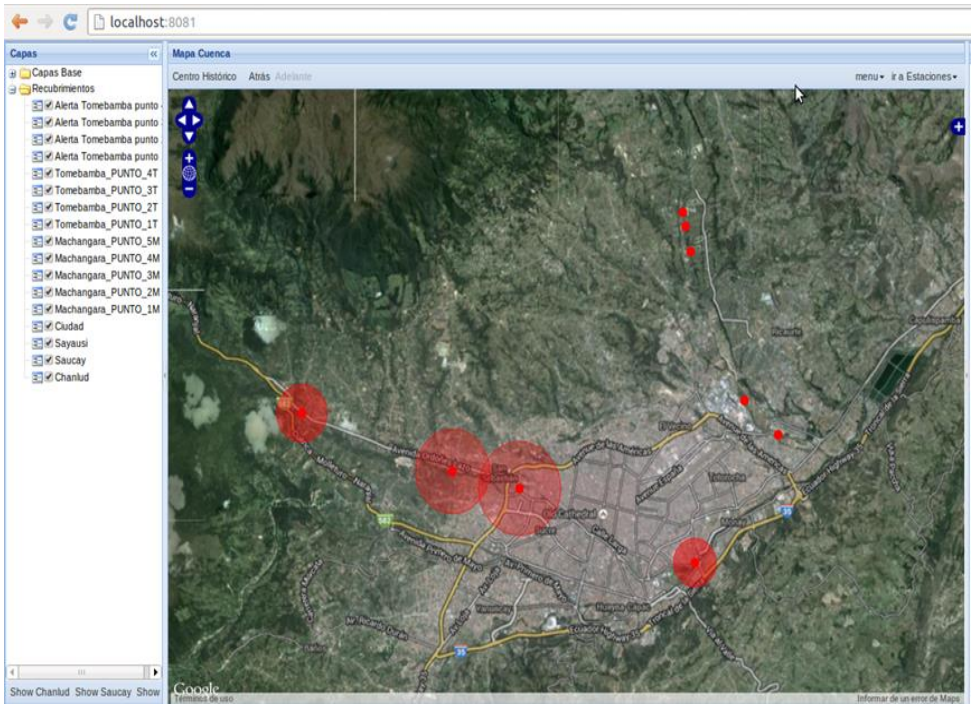
Alerta amarilla en las zonas susceptibles del río Tomebamba



Alerta naranja en las zonas susceptibles del río Tomebamba

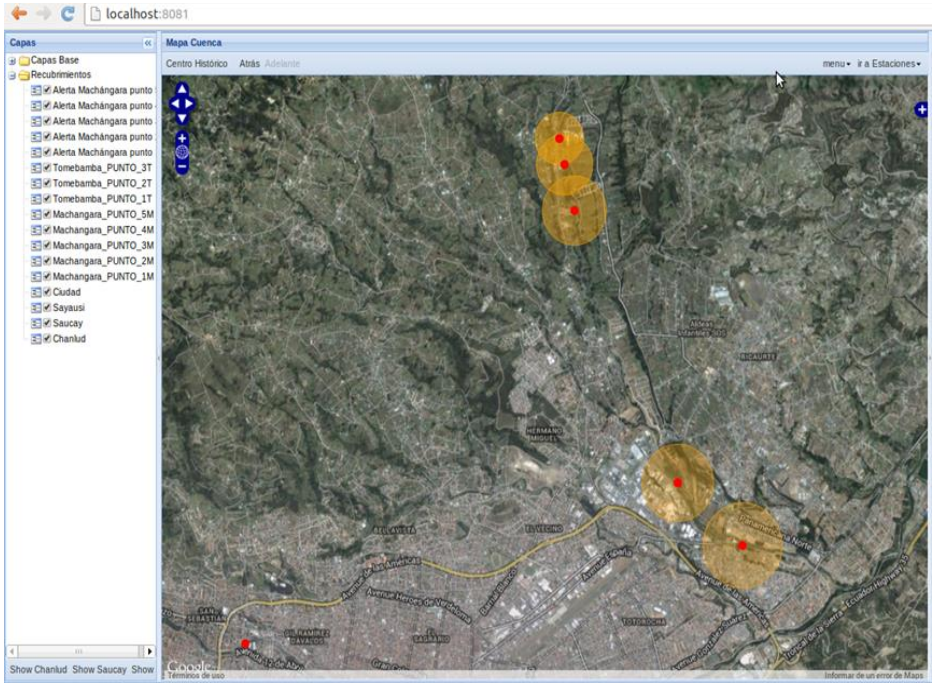


Alerta roja en las zonas susceptibles del río Tomebamba

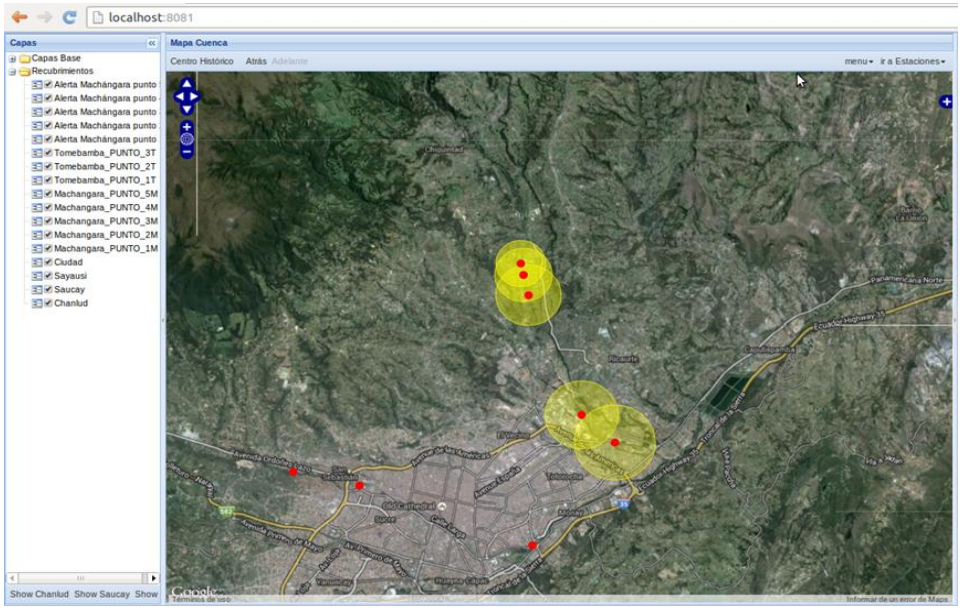


E. Alarmas Río Machángara

Alerta naranja en las zonas susceptibles del río Machángara



Alerta amarilla en las zonas susceptibles del río Machángara



Alerta roja en las zonas susceptibles del río Machángara