UNIVERSIDAD MAYOR DE SAN ANDRÉS
FACULTAD DE CIENCIAS PURAS Y NATURALES
CARRERA DE INFORMÁTICA

TESIS DE GRADO

"SISTEMA EMBEBIDO PARA LA PREVENCIÓN DE ACCIDENTES DE TRÁNSITO OCASIONADOS POR CONDUCTORES EN ESTADO DE EBRIEDAD APLICANDO EL INTERNET DE LAS COSAS"

PARA OPTAR AL TÍTULO DE LICENCIATURA EN INFORMÁTICA

MENCIÓN: INGENIERÍA DE SISTEMAS INFORMÁTICOS

POSTULANTE : Univ. ALISON PARISACA QUISPE

TUTORA METODOLÓGICA : Lic. BHYLENIA YHASMYNA RIOS MIRANDA

ASESOR : PhD. YOHONI CUENCA SARZURI

LA PAZ, BOLIVIA 2016

UNIVERSIDAD MAYOR DE SAN ANDRÉS
FACULTAD DE CIENCIAS PURAS Y NATURALES
CARRERA DE INFORMÁTICA

LA CARRERA DE INFORMÁTICA DE LA FACULTAD DE CIENCIAS PURAS Y NATURALES PERTENECIENTE A LA UNIVERSIDAD MAYOR DE SAN ANDRÉS AUTORIZA EL USO DE LA INFORMACIÓN CONTENIDA EN ESTE DOCUMENTO SI LOS PROPÓSITOS SON ESTRICTAMENTE ACADÉMICOS.

LICENCIA DE USO

El usuario está autorizado a:

a) visualizar el documento mediante el uso de un ordenador o dispositivo móvil.
b) copiar, almacenar o imprimir si ha de ser de uso exclusivamente personal y privado.
c) copiar textualmente parte(s) de su contenido mencionando la fuente y/o haciendo la referencia correspondiente respetando normas de redacción e investigación.

El usuario no puede publicar, distribuir o realizar emisión o exhibición alguna de este material, sin la autorización correspondiente.

TODOS LOS DERECHOS RESERVADOS. EL USO NO AUTORIZADO DE LOS CONTENIDOS PUBLICADOS EN ESTE SITIO DERIVARA EN EL INICIO DE ACCIONES LEGALES CONTEMPLADOS EN LA LEY DE DERECHOS DE AUTOR.

Dedicatoria

Esta tesis quiero dedicártela a ti, a ti por prestar atención y tiempo en ella al leer lo que escribí, y debo decirte que estoy honrada por ello.

Aprovecho estas líneas para decirte “se audaz y atrévete”, recuerda que los limites solo están en la mente, se ambicioso (a), mientras tengas aire en los pulmones y un corazón que late, falla la mayor cantidad de veces, solo así podrás conocer de lo que eres capaz.

Agradecimientos

A Dios por darme la oportunidad de vivir y disfrutar de la vida.

A mi mami Lorenza Quispe por su grande, verdadero e incondicional amor, por apoyarme en cada una de mis decisiones, por impulsarme a ser perseverante y valiente, por ser mi amiga y confidente.

A mi papá German Parisaca por ser tan comprensible, por haberme enseñado a hacerle frente a las adversidades, por el apoyo incondicional.

A mis hermanos: Jhon y Mishell, por ser un motor en mi vida, porque cuando los veo simplemente transmiten una alegría y unas ganas de comer al mundo.

A mi asesor: PhD. Yohoni Cuenca y mis tutores: Lic. Bhylenia Rios y Lic. Franz Cuevas, por su tiempo y dedicación para con esta tesis. Gracias por su paciencia, recomendaciones, ideas y observaciones, gracias por haberme guiado en esta etapa de mi vida universitaria.

A mis licenciados y licenciadas: Elizabeth García por su confianza y amistad, Reynaldo Zeballos , Jorge Terán, Menfy Morales, Brígida Carvajal, Celia Tarquino, Marisol Tellez, Carmen Huanca, José Luis Zeballos, por todas sus valiosas enseñanzas, consejos dentro y fuera del aula.

A los estudiantes de Inf-111 e Inf-121, por haberme dado el privilegio de ser parte de su formación académica, por su alegría y entusiasmo de aprender.

A Microsoft y al programa Microsoft Student Partner, por la oportunidad de expandir mis conocimientos fuera de las aulas y compartir lo aprendido a su vez, por haberme dado la oportunidad de explorar nuevas tecnologías.

Al equipo QKIEZ por su confianza en mí y su cofundador Samuel Rene Loza por ser compañero, amigo y comediante, por enseñarme a ver el mundo de diferente manera, por compartir la idea de que la imaginación no límites.

A todos mis amigos por compartir tristezas y más alegrías durante este tiempo.

Resumen

El Internet de las Cosas propone la conexión de cualquier dispositivo de hardware a internet.

Por ello la presente tesis propone la implementación de un sistema embebido que analice el grado alcohólico de un determinado conductor bajo el concepto del Internet de las Cosas, con el objetivo de notificar la ubicación de la movilidad a los familiares/amigos de confianza del propietario de la movilidad si el conductor tuviese un grado alcohólico mayor al establecido por ley en Bolivia.

Para llevar a cabo la construcción e implementación del sistema embebido se utilizó Sensor de gas etílico, modulo GPS, Modulo SIM900 y Arduino, como lenguaje de programación C++, para la comunicación entre el sistema embebido y el internet se usó el protocolo MQTT puesto que es más liviano y rápido que HTTP.

Para visualizarla notificación con la ubicación de la movilidad en un mapa se implementó una aplicación móvil para teléfonos con sistema operativo Android, donde se puede observar el listado de familiares/amigos de confianza del propietario, además de las ultimas notificaciones.

Abstract

The Internet of Things proposes the connection of any hardware device to the Internet.

For this reason, the present thesis proposes the implementation of an embedded system that analyzes the alcoholic level of a certain driver under the concept of Internet of Things, with the objective of notifying the location of the car to trust relatives/owner's friends of the owner's car if the driver had an alcoholic level greater than that established by law in Bolivia.

In order to carry out the construction and implementation of the embedded system we used Ethyl gas sensor, GPS module, SIM900 module and Arduino, C++ as a programming language, for the communication between the embedded system and the internet we used the MQTT protocol since it is Lighter and faster than HTTP.

To visualize the notification with the location of the car in a map a mobile application was implemented for cellphones with Android operating system, where it is possible to observe the list of relatives / owner's friends, in addition to the last notifications.

Índice

CAPÍTULO 1 MARCO REFERENCIAL 6
1.1. Introducción 6
1.2. Antecedentes 7
1.3. Planteamiento del Problema 8
1.3.1. Problema Central 9
1.3.2. Problemas Secundarios 9
1.4. Definición de Objetivos 9
1.4.1. Objetivo General 9
1.4.2. Objetivos Específicos 10
1.5. Hipótesis 10
1.6. Justificación 11
1.6.1. Justificación Económica 11
1.6.2. Justificación Social 11
1.6.3. Justificación Científica 11
1.7. Alcances y Limites 11
1.7.1. Alcances 11
1.7.2. Limites 12
1.8. Aportes 12
1.8.1. Práctico 12
1.8.2. Teórico 12
1.9. Metodología 12
CAPÍTULO 2 MARCO TEÓRICO 14
2.1. Internet de las cosas 14
2.1.1. Internet de las Cosas y su impacto 15
2.1.2. Campos aplicativos del Internet de las Cosas 17
2.1.3. Internet de las Cosas y Software Libre 21
2.2. Sistema embebido 21
2.2.1. Programación en un sistema embebido 22
2.2.2. Evolución de los Sistemas Embebidos 22

2.3. Dispositivos de Hardware 22
2.3.1. Arduino 22
2.3.2. Shields Arduino 24
2.3.3. Sensores 24
2.3.4. Modulo GPS 26
2.3.5. Módulos de conexión a Internet 27
2.4. Arquitectura y patrones de Software 28
2.4.1. Servicios RESTful 28
2.4.2. Patrón Publish/Suscribe 30
2.5. Protocolos de Comunicación MQTT 32
2.6. Metodología de desarrollo Scrum 33
2.6.1. Roles de Scrum 35
2.6.2. Eventos de Scrum 36
2.6.3. Artefactos de Scrum 38
CAPÍTULO 3 MARCO APLICATIVO 41
3.1. Backlog Inicial de Historias de Usuario 41
3.2. Descripción de Historias de Usuario identificadas 41
3.3. Sprints 43
3.3.1. Sprint 0 43
3.3.2. Sprint 1 46
3.3.3. Sprint 2 48
3.4. Modelo entidad-relación E-R 51
3.5. Diagrama de clases 52
CAPÍTULO 4 PRUEBAS Y RESULTADOS 53
4.1. Pruebas Nivel Hardware 53
4.1.1. Casos encontrados 53
4.1.2. Funcionamiento en cada caso 53
4.1.3. Funcionamiento del sistema embebido 55
4.2. Pruebas Nivel Software 56
4.2.1. Resultados de Pruebas de Estrés en los servicios RESTful 56
CAPÍTULO 5 CONCLUSIONES Y RECOMENDACIONES 58
5.1. Conclusiones 58
5.2. Estado de objetivos específicos 58

5.3. Estado de objetivo general.....59
5.4. Recomendaciones.....59
BIBLIOGRAFÍA.....60
ANEXOS.....64

Índice de figuras

Figura 1 Ejemplo Básico del Internet de las Cosas 14
Figura 2 Número de desarrolladores IoT 2014-2020 15
Figura 3 Distribución Global de los desarrolladores IoT 16
Figura 4 IoT en la Agricultura 17
Figura 5 IoT aplicada en el hogar 20
Figura 6 Arduino Uno y sus componentes 24
Figura 7 Algunos sensores y forma de utilizarlos 25
Figura 8 Sensor de gas MQ-3 26
Figura 9 Modulo Shield GPS para Arduino 26
Figura 10 Modulo Ethernet para Arduino 27
Figura 11 Módulo SIM900 GSM/GPRS 28
Figura 12 Funcionamiento de REST 30
Figura 13 Publish/Suscribe 31
Figura 14 Etapas de la metodología Scrum 35
Figura 15 Pantalla de Registro 45
Figura 16 Menú de la Aplicación móvil 45
Figura 17 Amigos/familiares de confianza en la aplicación 47
Figura 18 Pantalla de ubicación de movilidades 48
Figura 19 Pantalla de Últimas 15 notificaciones 48
Figura 20 Análisis de la concentración de alcohol en el aire 50
Figura 21 Conexión con GPS y SIM900 – Prueba con vino 51
Figura 22 Modelo Entidad – Relación E-R del sistema embebido 51
Figura 23 Diagrama de clases – Abstracción 52
Figura 24 Grafica Sensor MQ3 – Gas etílico 54
Figura 25 Esquema de funcionamiento del prototipo 55
Figura 26 Notificación en el teléfono celular 55
Figura 27 Mapa de ubicación de una movilidad con el prototipo 56

Índice de tablas

Tabla 1 Número de desarrolladores IoT 2014-2020 16
Tabla 2 Distribución Global de los desarrolladores IoT 17
Tabla 3 Descripción detallada de las partes de Arduino Uno 23
Tabla 4 Principios de un servicio REST 29
Tabla 5 Métodos del protocolo HTTP más utilizados 29
Tabla 6 Métodos del Protocolo MQTT 33
Tabla 7 Principios del manifiesto ágil 34
Tabla 8 Backlog inicial de Historias de Usuario 41
Tabla 9 H.U. Identificación de tecnologías 41
Tabla 10 H.U. Diseño Modelo E/R de la base de datos. 42
Tabla 11 H.U. Implementación de la Base de datos. 42
Tabla 12 H.U. Diseño inicial de pantallas de la aplicación móvil. 42
Tabla 13 H.U. Servicios RESTful – Añadir usuarios de confianza. 42
Tabla 14 H.U. Conexiones Hardware – Arduino y Sensor MQ3. 42
Tabla 15 H.U. Arduino, GPS y SIM900 – Comunicación con el servidor. 43
Tabla 16 H.U. Integración Servidor – Aplicación móvil 43
Tabla 17 H.U. Visualización de información en la aplicación Android 43
Tabla 18 Sprint 0 – Backlog 44
Tabla 19 Identificación de tecnologías para el desarrollo 44
Tabla 20 Casos de prueba – Sprint 0 44
Tabla 21 Sprint 1 – Backlog 46
Tabla 22 Casos de prueba – Sprint 1 46
Tabla 23 Sprint 2 – Backlog 48
Tabla 24 Componentes para el sistema embebido 49
Tabla 25 Casos de Prueba – Sprint 2 50
Tabla 26 Casos de prueba encontrados 53
Tabla 27 Resultados del sistema embebido 53
Tabla 28 Prueba de estrés de la aplicación móvil 56

CAPÍTULO 1
MARCO REFERENCIAL

1.1. Introducción

Según los datos de la OMS[1] Bolivia está entre los países que más beben en América Latina, antes de Chile, Argentina, Venezuela, Perú, Panamá, Uruguay, Ecuador, México, República Dominicana y Colombia, teniendo como precedente a Costa Rica, Cuba, Nicaragua, Honduras, Guatemala y el Salvador, ocupando así el puesto 13 de la lista. En una entrevista con Maristela Monteiro, asesora principal en abuso de sustancias y alcohol de la Organización Mundial de la Salud, asegura que "el alcohol no solo afecta a quien la bebe", refiriéndose al incremento de índice de violencia y los accidentes de tránsito (Moreno, 2015).

Por otro lado Bolivia establece como grado alcohólico máximo permitido cero punto cincuenta (0.50) grados en cada mil (1000) ml de sangre o su equivalente en mg/l.

Los pasos agigantados que da tecnología es cada vez más sorprendente, en este caso hablamos de IoT[2] que ahora se está aplicando en varios campos como ser la industria, agricultura, medicina, etc., según *Kevin Ashton*, el objetivo del Internet de las Cosas es conectar las cosas/objetos entre sí. Estos objetos/cosas se valen de *sistemas embebidos*, o lo que es lo mismo, hardware especializado que le permite no solo la *conectividad a Internet*, sino que además programa eventos específicos en función de las tareas que le sean dictadas remotamente (Torrez, 2014).

Pero ¿Cómo se aplica la tecnología para con los accidentes de tránsito provocados por conductores en estado de ebriedad?, la presente tesis usa el concepto de Internet de las Cosas, en la construcción de un sistema embebido que podrá medir el grado alcohólico del conductor, si bien el grado alcohólico detectado es mayor al establecido por ley, el sistema no permitirá la conducción de la movilidad, además de proceder a enviar notificaciones a celulares de

[1] OMS – Organización Mundial de la Salud

[2] IoT en ingles *Internet of Things*

familiares/amigos de confianza que mediante la aplicación móvil podrán observar en el mapa la ubicación de la movilidad.

1.2. Antecedentes

En esta sección se muestran trabajos ya existentes en el mundo que brindan un servicio parecido al que se propone.

En la Tesis de Pregrado *"Beehive, una plataforma open-source para el Internet de las Cosas"* de Guillen (2015) de la Universidad Mayor de San Andrés, La Paz, Bolivia, provee canales de comunicación, cuenta con las funcionalidades que una plataforma comercial ofrece, las cuales son la gestión de usuarios, gestión de permisos sobre los dispositivos y librerías para microcontroladores.

En la Tesis de Pregrado *"Control inteligente de seguridad en viviendas mediante agentes móviles"* de Machaca (2015) de la Universidad Mayor de San Andrés, La Paz, Bolivia**,** indaga los elementos computacionales que permitan contar con un modelo informático integrado, que sea capaz de realizar el control inteligente de la seguridad en viviendas controlada a través de un dispositivo móvil.

En la Tesis de Pregrado *"Control de dispensadores de odorización basado en Arduino"* de Alfaro (2015) de la Universidad Mayor de San Andrés, La Paz, Bolivia**,** en este trabajo se presenta un sistema de control del funcionamiento y alerta temprana de los dispensadores de odorización, busca coadyuvar en el objetivo de las empresas de mencionado servicio, implementando un sistema de control constante para los dispensadores de odorización, basado en Arduino, GPS y SMS; para que este de una alerta temprana mediante tecnología móvil, el cual mantendrá a la empresa que ofrece el servicio al tanto de sus clientes y equipos, basados en hardware y software libre.

En la Tesis de Pregrado *"Sistema Demótico para una Casa Inteligente"* De Marcos (2013) de la Universidad Pontificia Comillas ICAI-ICADE, Madrid, España, donde afirma que un importante aspecto de un sistema demótico es que no debería requerir la constante atención del usuario, sobre todo en temas de regulación. Un sistema que regule la temperatura a lo largo del

día en una sala seguramente ahorre más energía que una persona regulando el termostato. La única salvedad es el mantenimiento del sistema, en este trabajo se trata de diseñar un sistema domótica resistente y de bajo consumo de modo que se permita un mantenimiento menos frecuente.

En la Tesis de Pregrado *"Diseño e implementación de un sistema automático de Alumbrado Led Publico Inteligente controlado vía Wireless e instalado en una Casa de Don Bosco de Guayaquil"* de Chacho, Sotomayor & Delgado (2013) de la Universidad Politécnica Salesiana, Guayaquil, Ecuador**,** este trabajo busca un ahorro de energía y reducir la contaminación ambiental, donde se reducen gastos por el consumo de energía en iluminación invertidos por un determinado albergue, para ello se utilizó dispositivos de adquisición de datos para controlar, monitorear, supervisar la ubicación de Lámparas Led mediante el modulo inteligente GSM *Sistema Global para las comunicaciones móviles.*

Empresa dedicada a la tecnología de alcoholemia "*Lifesafer"* en los Estados Unidos, es un dispositivo que funciona como un alcoholímetro donde el conductor debe soplar o pasar la prueba de alcoholemia, este dispositivo está conectado al sistema eléctrico de una movilidad, impide que esta sea conducida por un conductor que ha estado bebiendo y que ha pasado el nivel máximo de alcohol permitido.

1.3. Planteamiento del Problema

Bolivia es catalogado como uno de los países que más beben (Moreno, 2015) donde señala que es una de las principales causas de accidentes de tránsito siendo provocados por conductores en estado de ebriedad.

De enero a mayo de 2014, La Paz tuvo 4484 accidentes de tránsito, de las cuales 526 accidentes fueron provocados por conductores en estado de ebriedad, el cual representa un 11.73%, siendo la segunda causa principal de accidentes de tránsito (Villa, 2014).

De enero a julio de 2015, en la ciudad de El Alto se registraron 2.034 accidentes de tránsito y 140 muertes, donde el consumo del alcohol influye aproximadamente en un 40% de accidentes de tránsito (Rivas, 2015).

De enero a febrero de 2016, se produjeron 4422 accidentes de tránsito en Bolivia de donde aproximadamente 8% viene a ser causado por consumo de bebidas alcohólicas, de las cuales en la ciudad de La Paz se registró 1696 hechos la cual representa el 38%, quedando como la ciudad que tiene la mayor tasa de accidentes de tránsito, teniendo como una de las causas principales el estado de ebriedad de los conductores (Valdés, 2016).

1.3.1. Problema Central

¿Cómo contribuir a prevenir accidentes de tránsito provocados por conductores en estado de ebriedad?

1.3.2. Problemas Secundarios

- Accidentes se provocan durante la noche y madrugada, cuando se tiene una deficiencia de policías controlando a conductores.
- Policía Boliviana hace prueba de alcoholemia solo en puntos establecidos.
- Conductores en estado de ebriedad provocan lesiones inclusive podría llegar hasta la muerte de los mismos conductores y/o terceras personas.
- Conductores en estado de ebriedad pueden provocar daños materiales.
- Familiares preocupados, ya que no tienen conocimiento de la ubicación del conductor en estado de ebriedad y de su automóvil en altas horas de la noche y/o madrugada hasta inclusive durante el día.

1.4. Definición de Objetivos

1.4.1. Objetivo General

Implementar un sistema embebido que analice el grado alcohólico de un conductor basado en Arduino bajo el concepto del Internet de las Cosas, el cual envie notificaciones a los familiares/amigos del mismo en estado de ebriedad que mediante una aplicación móvil los familiares puedan visualizar la ubicación de una determinada movilidad para poder ayudar a prevenir accidentes de tránsito.

1.4.2. Objetivos Específicos

- Diseñar e implementar una base de datos, donde se registren la georreferenciación, placa de las movilidades que están siendo conducidas o a punto de ser conducidas por conductores en estado de ebriedad con un grado alcohólico mayor al límite establecido por ley (0.5 grados de alcohol por mililitro de sangre), así como los contactos de confianza que recibirán las notificaciones.
- Construir el sistema embebido que analice el grado alcohólico del conductor.
- Realizar el módulo de envío de mensajes/notificación de manera automática en el sistema embebido aplicando el concepto del Internet de las Cosas cuando el conductor tenga un grado alcohólico mayor al permitido.
- Realizar el envío de mensajes automatizados, cuando la movilidad este en marcha y el conductor este bebiendo.
- Realizar un indicador que muestre que la movilidad no se puede conducir.
- Realizar la aplicación móvil con una interfaz amigable y de fácil uso, donde se mostrara la ubicación de la movilidad en cuestión a familiares/amigos.
- Aplicar la Metodología Ágil Scrum para el desarrollo del prototipo.

1.5. Hipótesis

“Con la construcción e implementación del sistema embebido basado en Arduino bajo el concepto del Internet de las Cosas, ayudará a prevenir accidentes de tránsito ocasionados por conductores en estado de ebriedad.”

A. Variables independiente

Sistema embebido basado en Arduino bajo el concepto del Internet de las Cosas.

B. Variables dependiente

Ayudará a prevenir accidentes de tránsito ocasionados por conductores en estado de ebriedad.

1.6. Justificación

1.6.1. Justificación Económica

El presente sistema embebido propuesto en la tesis, analiza constantemente el grado alcohólico de un conductor ya que estará situado en la parte del volante, si este sistema detecta que existe un grado alcohólico permitido por ley pues evitara la conducción del motorizado, por consiguiente evitara pérdidas materiales, como ser daños a la misma movilidad en accidentes, atropellos a peatones y/o propiedades.

1.6.2. Justificación Social

Mediante el desarrollo del prototipo tanto en Hardware y Software aplicando el Internet de las Cosas para la prevención de accidentes de tránsito provocados por conductores en estado de ebriedad, se pretende que sirva de apoyo a los desarrolladores de sistemas embebidos, para poder tener en futuro investigaciones más amplias y con mayor aceptación, hecho que beneficia en nuestro ámbito.

1.6.3. Justificación Científica

El presente trabajo se justifica científicamente, debido a que mediante esta investigación se amplían las aplicaciones del Internet de las Cosas, desde la construcción del sistema embebido que detecta el grado alcohólico de un conductor hasta la interacción con una aplicación móvil.

1.7. Alcances y Limites

1.7.1. Alcances

- Se logrará la interactividad del sistema embebido con la aplicación móvil.
- El sistema analiza constantemente el grado alcohólico que fluye en el aire hasta 50 centímetros desde el volante de una determinada movilidad.
- El sistema solo enviara notificaciones a familiares del conductor si detecta un grado alcohólico mayor al establecido por ley en Bolivia.

- Aplicación móvil en Android, donde los familiares y/o amigos de confianza podrán visualizar en un mapa la localización del vehículo.

1.7.2. Limites

- Cuando detecta al conductor en estado de ebriedad y la movilidad está en marcha no la detiene.
- El sistema no envía mensajes cuando la movilidad está en una zona donde se carezca de señal.
- El sistema embebido no detecta el grado alcohólico de los pasajeros.
- El sistema embebido no detecta presencia del conductor.
- La aplicación no está disponible para celulares que no tengan Sistema Operativo Android.
- No se puede obtener la ubicación de una determinada movilidad desde la aplicación móvil.

1.8. Aportes

1.8.1. Práctico

La presente investigación reportara la ubicación de una movilidad que este siendo conducida por un conductor en estado de ebriedad o quiera conducir uno a los familiares y/o amigos del mismo conductor, así estos podrán actuar para evitar mayores accidentes.

1.8.2. Teórico

Aplicar el Internet de Las Cosas, construcción de un sistema embebido y una aplicación móvil para la solución de problemas, en este caso social como es “los accidentes de tránsito provocados por conductores en estado de ebriedad”.

1.9. Metodología

Se utiliza la metodología de investigación Sistémica ya que es uno de los instrumentos lógicos más contemporáneos en el ámbito de la metodología, orientado a la percepción holística es decir

la totalidad de la realidad de donde se extraerá la propia problemática y las soluciones correspondientes.

El Enfoque Sistémico (ES) es para la Informática una herramienta conceptual y de acción, interpreta lo concreto y facilita el pasaje de la teoría a la práctica, logrando mejores resultados.

En el marco del Enfoque Sistémico, el análisis de sistemas, la modelización y la simulación se constituyen en las tres etapas fundamentales para el estudio del comportamiento dinámico de los sistemas complejos (Barchini, Sosa & Herrera, 2014).

Las principales etapas de esta metodología son:

- Reconocimiento de problemas sistémicos: extracción de los aspectos relacionales y estructurales del problema.
- Abstracción del problema: identificación del problema dentro de un marco conceptual particular.
- Aplicación propia: uso de una herramienta metodológica apropiada para resolver el problema en su formulación abstracta.
- Interpretación de los resultados en términos del problema específico.

CAPÍTULO 2
MARCO TEÓRICO

2.1. Internet de las cosas

El Internet de las Cosas podría ser definida de la siguiente manera: *"El Internet De Las Cosas pretende ofrecer una conectividad avanzada entre dispositivos, sistemas y servicios que va más allá de una comunicación M2M (Machine to Machine) y que cubre una variedad de protocolos, dominios y aplicaciones"* (Höller, Tsiatsis, Mulligan, Karnouskos, Avesand & Boyle, 2014).

Internet de las cosas es el espacio donde el cálculo físico colisiona con Internet, donde se combina el diseño y el hardware, el software y la Web para crear objetos interactivos y prácticos que puedan sentir y responder a estímulos online o a cambios de su entorno (Mcewen & Cassimally, 2014).

Teniendo en cuenta los conceptos mencionados, se define la siguiente fórmula para el internet de las Cosas (McEwen & Cassimally, 2014).

Internet de las Cosas = Objeto Físico + Controlador, Sensor y Actuadores + Internet

En la Figura 1 se ilustra algunas aplicaciones del Internet de las Cosas.

Figura 1 Ejemplo Básico del Internet de las Cosas

Fuente: (Carrero, 2016)

2.1.1. Internet de las Cosas y su impacto

A medida que pasa el tiempo el interés hacia el Internet De Las Cosas crece, la empresa Gartner realiza un estudio en el que se estima que para el año 2020 la cantidad de dispositivos electrónicos conectados a internet será aproximadamente de 26 billones, excluyendo computadoras personales, teléfonos inteligentes, y tablets. Además que los proveedores de productos y servicios para el Internet De Las Cosas generará un ingreso de aproximadamente 300 billones de dólares (Gartner, 2013).

Se estima que gran parte de los ingresos que generará el Internet de las Cosas será por parte de los servicios que se brinden y no por los productos de hardware creados. Al ser los productos de hardware, en general sensores que generarán una gran cantidad de información se necesitarán varios desarrolladores para poder procesar dicha información (Asay, 2014).

Un estudio realizado por la consultora Research and Markets hay más de 6,2 millones de desarrolladores en el mundo enfocados en IoT (Olmo, 2016), dato que supera el estudio realizado por Visión Mobile en 2014 como se muestra en la Figura 2.

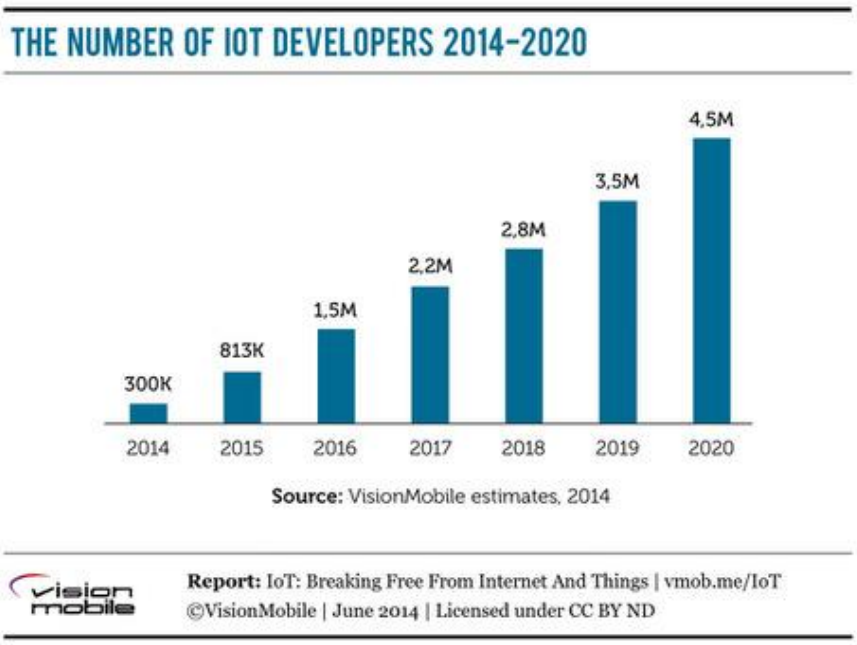


Figura 2 Número de desarrolladores IoT 2014-2020

Fuente: (Olmo, 2016)

La Tabla 1 Muestra los valores identificados en la Figura 2.

Tabla 1 Número de desarrolladores IoT 2014-2020

Año	Desarrolladores IoT
2014	300 Mil
2015	813 Mil
2016	1,5 Millones
2017	2,2 Millones
2018	2,8 Millones
2019	3,5 Millones
2020	4,5 Millones

Un estudio realizado por Visión Mobile muestra la distribución global de los desarrolladores para el Internet de las Cosas como se observa en la Figura 3, donde estos datos muestran que no existe un área que domine totalmente el campo del Internet de las Cosas en cuanto a desarrolladores.

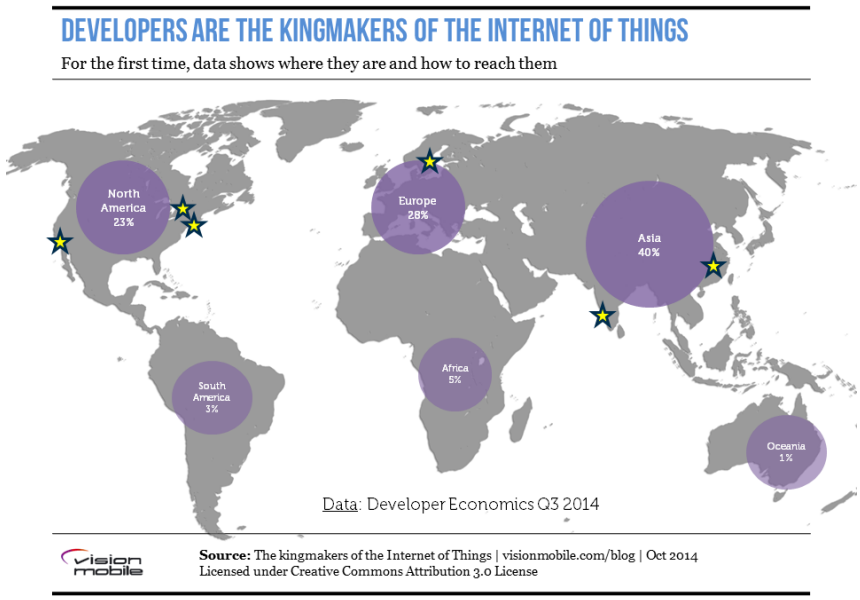


Figura 3 Distribución Global de los desarrolladores IoT

Fuente: (Schuermans S. , 2014)

En la Tabla 2 se observa los valores identificados en la Figura 3.

Tabla 2 Distribución Global de los desarrolladores IoT

Región	**Porcentaje de desarrolladores**
Asia	40%
Europa	28%
América del norte	23%
África	5%
América del Sur	3%
Oceanía	1%

2.1.2. Campos aplicativos del Internet de las Cosas

Al existir diferentes tipos de dispositivos que podrán interactuar con seres humanos se generará una cantidad increíble de datos a lo que se denomina BigData y el uso apropiado de estos datos puede ser aplicado para el beneficio del ser humano. Entre algunas áreas que serán beneficiadas por el Internet De Las Cosas son la agricultura, salud y domótica (Dacosta, 2013).

Agricultura

El manejo y control de la industria agrícola es dificultoso ya que existen muchos factores en juego como ser: factores naturales y factores humanos. Hoy en día a pesar que existe un gran interés de inversionistas sobre la agricultura y la tecnología es un hecho no muy conocido que la agricultura fue y sigue siendo uno de los sectores que proveerá al Internet De Las Cosas una adopción a gran escala (Mohammed, 2014).

En la Figura 4 muestra una abstracción de lo mencionado en el anterior párrafo.

Figura 4 IoT en la Agricultura
Fuente: (Tharrington, 2015)

La siguiente lista muestra algunas áreas de la agricultura que serán campos de prueba para el Internet De Las Cosas:

- Productividad

  El campo de agricultura de precisión, una actividad que usa el análisis de datos para optimizar decisiones agrícolas, es un gran campo de oportunidades para el Internet De Las Cosas. En la actualidad, es más crítico que nunca el maximizar el rendimiento de cada acre de tierra que se encuentra dedicado a la producción de alimentos. Con la ayuda de dispositivos inalámbricos conectados a la nube será posible maximizar la producción de la cosecha automatizando operaciones de agricultura como ser el riego, control de humedad y otros, además de proveer un monitoreo en tiempo real y el análisis de datos para una toma de decisiones inteligente.
- Control de Plagas

  A medida que el movimiento orgánico gana más popularidad, el interés por alternativas más efectivas y relativamente económicas a los pesticidas por parte de la industria agrícola y de alimentos ha aumentado considerablemente. Una de estas alternativas son las trampas de feromonas que, combinadas con el Internet De Las Cosas pueden resultar útiles ya que será posible monitorear la cantidad de alguna plaga y en caso que esta población aumente, se podrá activar una trampa de feromonas para poder acabar con dicha plaga.
- Conservación del Agua

  Desde el punto de vista de la escasez del agua, la agricultura históricamente ha sido un reto que demanda un gran conocimiento técnico, dominio en colección de datos y de sistemas de riego. Para una respuesta efectiva, granjeros requieren información precisa en tiempo real que ayude a minimizar el desperdicio, exceso y falta de irrigación, de esta forma es posible manejar los costos del agua de una forma más efectiva. Combinando el Internet De Las Cosas con dispositivos inalámbricos y sistemas de monitoreo del suelo, granjeros pueden medir humedad, detectar fugas de agua y controlar con más eficiencia el uso de energía, todo en tiempo real.

Salud

En países de Europa o Norte América, existen nuevos dispositivos que permiten realizar el diagnóstico y tratamiento. Algunos de estos dispositivos se encuentran conectados a Internet y al ser así, se convierten en un importante elemento para el Internet De Las Cosas. Entre algunos beneficios de tener conectado equipamiento de prueba de laboratorio y otros dispositivos de diagnóstico se tiene: (Isaacs, 2014)

- Reducir el Tiempo de inactividad a través de monitoreo y soporte remoto

  Un dispositivo conectado a Internet puede ser probado y diagnosticado remotamente, como ejemplo un técnico puede conectarse desde su oficina y realizar el diagnóstico de un equipo de imagen por resonancia magnética que este fallando. El técnico podría encontrar la raíz del problema además que este técnico puede conectarse con los técnicos del hospital al que pertenece el equipo para realizar el soporte técnico. Cuando la causa del problema sea identificada, una nueva pieza puede ser entregada incluyendo las instrucciones para reemplazarla.
- Cumplimiento proactivo mediante la reposición de suministros antes de que sean necesarios

  Un equipo médico conectado a Internet puede reportar cuando componentes críticos de funcionalidad se encuentran en niveles bajos. Un ejemplo son los niveles de helio en un equipo de imagen por resonancia magnética que necesitan ser monitoreados para asegurar que el equipo se encuentra funcionando correctamente. Esto permite que técnicos en el área reciban notificaciones de visitar un hospital antes que los niveles de helio de estos dispositivos se hayan agotado, evitando así, el apagado total del equipo además de re-programar exámenes a los pacientes.
- Programación eficiente para atender más pacientes

  En el campo médico, existe una gran cantidad de equipos que son muy caros como para ser manejados de manera ineficiente. Un equipo médico conectado a Internet puede proveer estadísticas diarias de uso que pueden ser aprovechadas para la programación de pacientes. Siguiendo con el ejemplo del equipo de imagen por resonancia magnética, en algún área este equipo podría ser utilizado solo en un 20 por ciento mientras que en otra área este mismo equipo puede encontrarse sobre-utilizado. Siendo el caso, doctores

pueden reasignar a los pacientes a utilizar el otro equipo. Esto a través de una aplicación basada en la nube que realice la programación de uso de dichos equipos.

Domótica

Por muchos años el mercado de domótica fue dominado por empresas que realizan instalaciones en el hogar, equipando clientes adinerados con instalaciones entre $10.000 y $100.000 para poder controlar luces, seguridad, aire acondicionado, el audio de la casa y otros como se muestra en la Figura 5. En la actualidad surgen nuevas formas de automatización más simples gracias a los productos DIY[3] que existen en el mercado haciendo innecesarios varios servicios de las empresas de automatización del hogar. Algunas marcas que proveen esto son Nest, Yale, Iris, Apigy, Sonos, Lutron y Belkin que individualmente controlan la corriente, luces, cerraduras, puertas de garaje, cortinas y cualquier cosa que pueda conectarse (Moorhead, 2013).

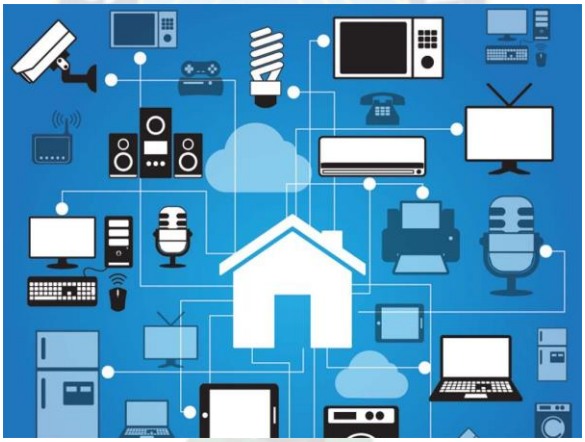


Figura 5 IoT aplicada en el hogar
Fuente: (Palma, s.f.)

Al crecer un mercado más amplio de productos DIY un nuevo reto surge. Un usuario puede comprar una variedad de estos productos (termostatos Nest, cerraduras de Apigy, conectores Belkin, interruptores de luz Lutron) entonces este tendrá la obligación de utilizar cuatro aplicaciones diferentes, es decir una por producto DIY, además de adaptarse al funcionamiento de las plataformas que pueden diferir unas de otras. En adición, no existen formas fáciles para que estos dispositivos trabajen como un solo sistema, es decir, programación de eventos y otros

[3] DIY – Do it yourself en español Hazlo por ti mismo

ya que estos productos requieren aplicaciones distintas. Una de las soluciones a este problema es la de crear una aplicación superior para las aplicaciones que difieren (Moorhead, 2013).

2.1.3. Internet de las Cosas y Software Libre

El software libre y los estándares abiertos ya se encuentran presentes en el contexto del Internet De Las Cosas. Una cantidad considerable de aplicaciones que obtienen y analizan datos dependen de software Open-Source y estándares abiertos. Los dispositivos de hardware con software embebido en el contexto del Internet De Las Cosas ejecutan un sistema operativo GNU/Linux en su gran mayoría, es por ello que los fabricantes de hardware adoptaron de un 40 a 50 por ciento sistemas operativos Open-Source, sin embargo, existe un gran porcentaje de software propietario dentro de estos sistemas embebidos (Korolov, 2015).

La diferencia entre Software libre y privativo radica en como los dispositivos mantendrán comunicación, los estándares de mensajería y las aplicaciones que implementen estos estándares. En una encuesta realizada por Visión Mobile a desarrolladores en el área del Internet de las Cosas muestra que un 91% prefiere usar Software Libre (Schuermans, 2016).

2.2. Sistema embebido

Un Sistema Embebido es un sistema electrónico diseñado para realizar pocas funciones en tiempo real es decir se los diseñan para cubrir necesidades específicas, al contrario de lo que ocurre con las computadoras, las cuales tienen un propósito general, ya que están diseñadas para cubrir un amplio rango de necesidades. Sin embargo los sistemas embebidos ya que por lo general pertenecen a máquinas que se espera que funcionen por años, y en algunos casos se programan para que se recuperen de posibles errores.

En un Sistema Embebido la mayoría de los componentes se encuentran incluidos en la placa base (la tarjeta de video, audio, módem) y muchas veces los dispositivos resultantes no tienen el aspecto de lo que se suele asociar a una computadora, sin embargo suelen tener en una de sus partes una computadora con características especiales conocida como **microcontrolador** que viene a ser el cerebro del sistema, este no es más que un microprocesador que incluye interfaces

de entrada/salida en el mismo chip. Normalmente estos sistemas poseen una interfaz externa para efectuar un monitoreo del estado y hacer un diagnóstico del sistema.

Algunos ejemplos de Sistemas Embebidos podrían ser dispositivos como un taxímetro, un sistema de control de acceso, la electrónica que controla una máquina expendedora o el sistema de control de una fotocopiadora entre otras múltiples aplicaciones (SemanticWebBuilder, 2015).

2.2.1. Programación en un sistema embebido

Los Sistemas Embebidos se pueden programar directamente en el lenguaje ensamblador del microcontrolador o microprocesador incorporado sobre el mismo, o también, utilizando los compiladores específicos que utilizan lenguajes como C o C++ y en algunos casos, cuando el tiempo de respuesta de la aplicación no es un factor crítico, también pueden usarse lenguajes interpretados como Java la cual está siendo muy usada ya que la capacidad cada vez mayor del nuevo hardware disponible en el mercado (SemanticWebBuilder, 2015).

2.2.2. Evolución de los Sistemas Embebidos

Se está planteando la evolución de los Sistemas Embebidos a Sistemas Inteligentes, en donde la principal diferencia para considerarlos como inteligentes es que deben estar conectados a otro dispositivo M2M Comunicación Maquina a Maquina o en especial a Internet, lo cual llevaría una reducción en el precio de los componentes hardware y software debido a la gran cantidad de computadoras en todo el mundo (SemanticWebBuilder, 2015).

2.3. Dispositivos de Hardware

2.3.1. Arduino

Arduino es un plataforma física Open-Source basada en una placa electrónica simple, con la lógica de “Hardware y Software fácil de utilizar” además de un entorno de desarrollo para crear software para esta placa. Esta puede ser utilizada para crear diferentes tipos de proyectos que incluyan lectura de diferentes tipos de sensores o el control de diferente tipos de salidas físicas como luces, motores y otros. Al ser hardware Open-Source, el esquema de Arduino puede ser utilizado para crear clones de esta placa. Actualmente existe una gran variedad de clones los

cuales son construidos bajo los esquemas de un Arduino, al ser así, estos clones pueden ser utilizados como si fuera un Arduino Original (Arduino, n.d.).

"Arduino puede tomar información del entorno a través de sus pines de entrada de toda una gama de sensores y puede afectar aquello que le rodea controlando luces y motores" (SemanticWebBuilder, 2015).

La siguiente Tabla 3 muestra las partes de Arduino Uno de manera detallada según (Herrador, 2009).

Tabla 3 Descripción detallada de las partes de Arduino Uno

Nro.	Partes de Arduino
1	Conector USB para el cable Tipo AB
2	Pulsador de Reset
3	Pines de E/S digitales y PWM
4	LED verde de placa encendido
5	LED naranja conectado al pin13
6	ATmega 16U2 encargado de la comunicación con el PC
7	LED TX (Transmisor) y RX (Receptor) de la comunicación serial
8	Puerto ICSP para programación serial Microcontrolador
9	ATmega 328, cerebro del Arduino
10	Cristal de cuarzo de 16Mhz
11	Regulador de voltaje
12	Conector hembra 2.1mm con centro positivo
13	Pines de voltaje y tierra
14	Entradas análogas

Arduino Uno la cual es ilustrada por la Figura 6 es considerado el modelo más popular dentro de la familia de Arduino, esta placa usa el micro controlador ATMega328 el cual dota a este modelo de 32 kb de memoria para almacenar programa, cuenta con 14 pines de entrada y salida digital, de los cuales los numerados como 0 y 1 se pueden utilizar como bus de comunicación para interactuar con otros dispositivo, 6 pines como entradas analógicas, un resonador cerámico de 16 Mhz, conexión USB, un alimentador de poder, botón de reinicio y cabecera ICSP.

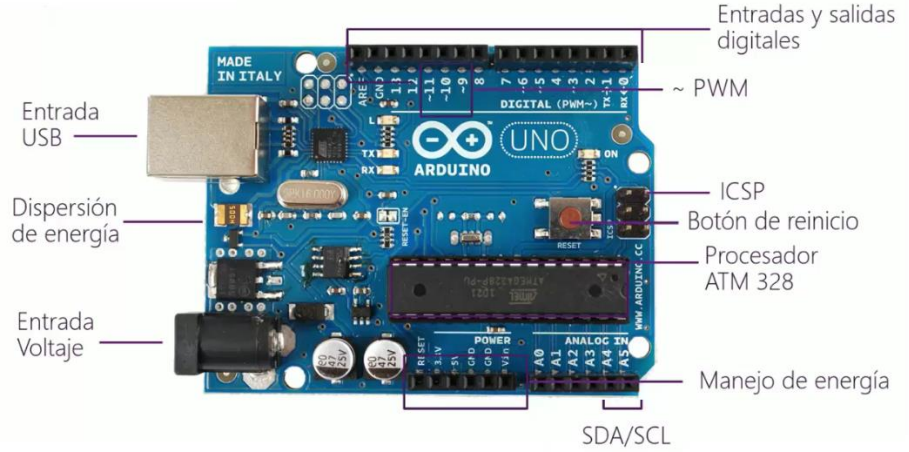


Figura 6 Arduino Uno y sus componentes

2.3.2. Shields Arduino

Un shield es una placa impresa que se pueden conectar en la parte superior de la placa Arduino para ampliar sus capacidades, pudiendo ser apilada una encima de la otra.

Las shields suelen ser diseños bastante simples y en general de código abierto y publicados libremente (Anderson & Cervo, 2013).

2.3.3. Sensores

Un sensor es un dispositivo capaz de detectar acciones o estímulos externos y responder a los mismos. Los sensores pueden transformar las magnitudes físicas o químicas, en variables eléctricas.

"Los sensores son las maneras de conseguir información en el dispositivo, descubrir cosas acerca de su entorno" (McEwen & Cassimally, 2014).

Los sensores siempre que estén activados estarán tomando continuamente la situación actual de una habitación y es el servidor o la placa Arduino quien leerá esta información y decidirá cómo actuar de acuerdo a la programación. Capturando los datos que nosotros deseemos.

Existen muchos sensores, para diferentes aplicaciones, la Figura 7 muestra algunos sensores, algunos de ellos son: sensor magnético, sensor de impacto, sensor de movimiento, sensor de humo, sensor de gas, sensor de humedad y temperatura.

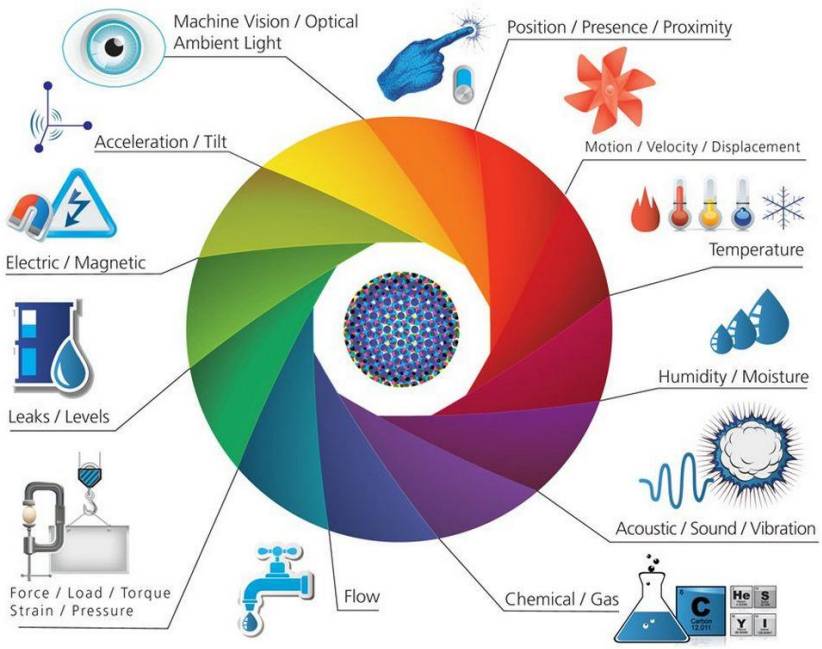


Figura 7 Algunos sensores y forma de utilizarlos

Fuente: (Lopez, s.f.)

Sensor MQ-3

El sensor MQ-3 que se muestra en la Figura 8 pertenece a la serie MQ, es muy sensible al alcohol y de menor sensibilidad a la bencina, también es sensible a gases como Gas Licuado de Petróleo GLP, Hexano, Monóxido de Carbono CO, Metano CH4 pero con sensibilidad muy baja, la cual se puede despreciar si hay poca concentración de estos (NaylampMechatronics, 2015).

El sensor de gas MQ-3 reacciona cuando un poco de alcohol está presente en el medio ambiente. Fue seleccionado porque es de alta sensibilidad tiene una respuesta rápida. Cuando se detecta el olor, la resistencia entre dos terminales varía y porque la tensión de alimentación se mantiene la corriente durante los cambios de carga que genera diferentes tensiones en él. Este voltaje $V_{entrada}$ es la que se mide y se analiza (Pavon, Duque & Fuentes, 2012).

"Este sensor se puede implementar con cualquier microcontrolador incluyendo Arduino UNO, además de ser adecuado para su uso como alcoholímetro" (NaylampMechatronics, 2015).

Figura 8 Sensor de gas MQ-3
Fuente: (NaylampMechatronics, 2015)

2.3.4. Modulo GPS

GPS[4], es un sistema de navegación por satélite basado en el espacio diseñado por el ejército estadounidense a finales de la década de 1960. Sin embargo, la tecnología fue liberada para uso civil recién en el mes de marzo del año 1996 (Library of Congress, 2011).

La principal tarea del GPS es devolver información sobre la posición, hora y/o tiempo de una entidad que se encuentre en la tierra. Para devolver esta información, al menos cuatro de los veintiocho satélites actualmente funcionando deben tener acceso visual no obstruido a la entidad en cuestión.

Figura 9 Modulo Shield GPS para Arduino
Fuente: (Arduino, s.f.)

[4] GPS – Global Positioning System

El shield GPS presentada en la Figura 9 es un dispositivo que te permitirá dotar a su Arduino de la capacidad de recibir señales de GPS y de almacenar las coordenadas recibidas en una tarjeta de memoria micro SD. El dispositivo es compatible con las tarjetas Arduino UNO y Mega. Los pines de comunicación del módulo GPS pueden seleccionarse para ser conectados a los pines digitales D0 a D7 las cuales con señales de Transmisión y Recepción TX y RX. El Shield GPS, además de proporcionar la ubicación geográfica con una precisión de unos cuantos metros, también proporciona información sobre la hora con gran precisión, por lo que podemos utilizar este Shield en aplicaciones donde se requiere una base de tiempo muy precisa (GeekFactory, s.f.).

2.3.5. Módulos de conexión a Internet

Existen diferentes módulos de hardware de que permiten a un micro controlador efectuar una conexión a Internet. Estos módulos permiten conectarse a través de diferentes tecnologías de conexión como ser Ethernet, Wifi, Red GSM y otros.

Modulo Ethernet para Arduino

El módulo Ethernet que se muestra en la Figura 10 permite a un dispositivo Arduino conectarse a una red local o a Internet. Basado en el chip Ethernet Wiznet W5100, este módulo permite trabajar con protocolos TCP y UDP soportando a lo mucho cuatro conexiones simultáneas. Este módulo posee un zócalo para tarjetas micro-SD, el cual puede ser utilizado para almacenar archivos para compartir en la red. Los datos de esta tarjeta micro-SD pueden ser utilizada por la librería SD que provee Arduino (Arduino , 2013).

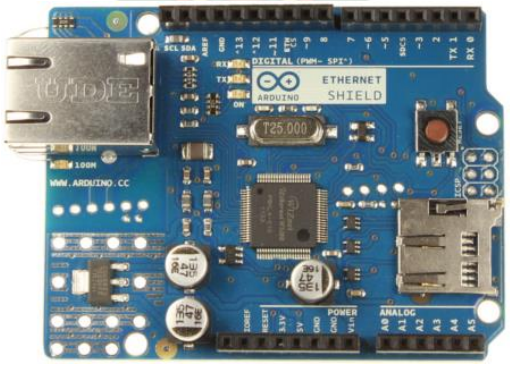


Figura 10 Modulo Ethernet para Arduino
Fuente: (Arduino , 2013)

Módulo SIM900 para Arduino

GPRS - *General Packet Radio Service*, es una extensión de la tecnología de comunicaciones móviles GSM - *Sistema Global para las comunicaciones Móviles*. En ella la información es dividida en pequeños bloques, los que posteriormente se reagrupan al llegar a destino. Este tipo de transmisión permite una mayor capacidad y velocidad. GPRS permite la conexión a Internet (Entel, s.f.).

Módulo SIM900 que se presenta en la Figura 11 es una tarjeta GPRS ultra compacta de comunicación inalámbrica a comparación de la Shield Ethernet. La tarjeta es compatible con todos los modelos de Arduino con el formato UNO, también se puede controlarla con otros microcontroladores (Lara, 2015).

Figura 11 Módulo SIM900 GSM/GPRS
Fuente: (Itead, s.f.)

2.4. Arquitectura y patrones de Software

2.4.1. Servicios RESTful

REST[5] es una abstracción de la arquitectura de la World Wide Web. Este es un estilo de arquitectura que contempla todos los componentes dentro de un sistema hipermedia distribuido (Fielding, 2000).

[5] REST - Representational State Transfer, en español Transferencia de Estado Representacional

REST define un set de principios arquitectónicos por los cuales se diseñan servicios web haciendo foco en los recursos del sistema, incluyendo cómo se accede al estado de dichos recursos y cómo se transfieren por HTTP hacia clientes escritos en diversos lenguajes.

Una implementación concreta de un servicio web REST sigue cuatro principios de diseño fundamentales, las cuales se muestra en la Tabla 4 (De Seta, 2008).

Tabla 4 Principios de un servicio REST

Nro.	**Principio**
1	Utiliza los *métodos HTTP* de manera explícita.
2	*No mantiene estado*, los servicios web REST necesitan escalar para poder satisfacer una demanda en constante crecimiento.
3	Expone URIs con forma de directorios.
4	Transfiere XML, JSON[6], o ambos.

La Web funciona debido al protocolo HTTP[7], ya que este brinda muchas facilidades al cliente que realiza una conexión con un servidor web. El protocolo HTTP permite realizar diferentes acciones en los diferentes recursos gracias al uso de métodos que se muestra en la Tabla 5, de este modo el servidor interpreta una petición en base al método que está presente como se muestra en la Figura 12 (HTTP, 1999).

Tabla 5 Métodos del protocolo HTTP más utilizados

Método	**Descripción de Método**
GET	La petición obtiene un recurso.
POST	La petición crea un recurso en el servidor.
PUT	Indica que la petición realizará cambios a un recurso existente. Se pueden entender como una petición que actualizará un recurso.
DELETE	Indica que la petición eliminará un recurso existente.

[6] JSON - JavaScript Object Notation

[7] HTTP – Hypertext Transfer Protocol

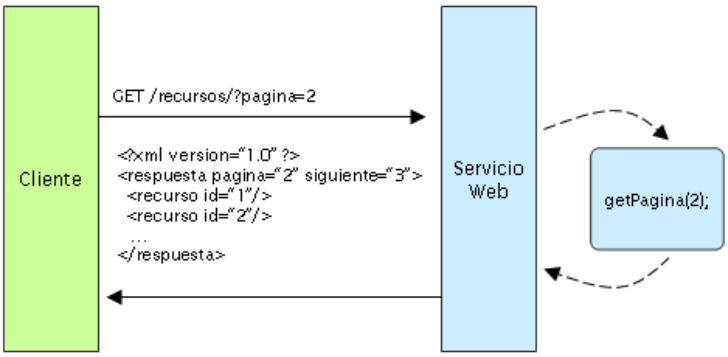


Figura 12 Funcionamiento de REST

Fuente: (De Seta, 2008)

2.4.2. Patrón Publish/Suscribe

El envío de mensajes haciendo uso del patrón Publish/Subscribe permite que varios usuarios puedan enviar mensajes a una entidad en el servidor, esta entidad por lo general se llama tópico y cada tópico puede tener varios suscriptores. Cada suscripción recibe una copia de cada mensaje que se envía al tópico suscrito (HiveMQ, 2015).

El patrón Publish/Suscribe cuyo funcionamiento básico se muestra en la Figura 13 es una alternativa a un modelo cliente-servidor tradicional donde el cliente se comunica directamente con un punto final (endpoint). El objetivo principal de desacoplar a un cliente el cual envía un mensaje al servidor (publisher) con uno o más clientes (subscribers) que reciben mensajes a través de un tópico. Esto significa que el usuario que envía el mensaje (publisher) y los usuarios que reciben este mensaje (subscribers) no tienen un conocimiento directo de la existencia del usuario remitente (publisher). Hay un tercer componente, el cual se lo denomina "broker" o agente que es conocida por el por el remitente y el receptor, que filtra todos los mensajes entrantes en base al tópico y luego distribuir a los usuarios que estén suscritos a este tópico.

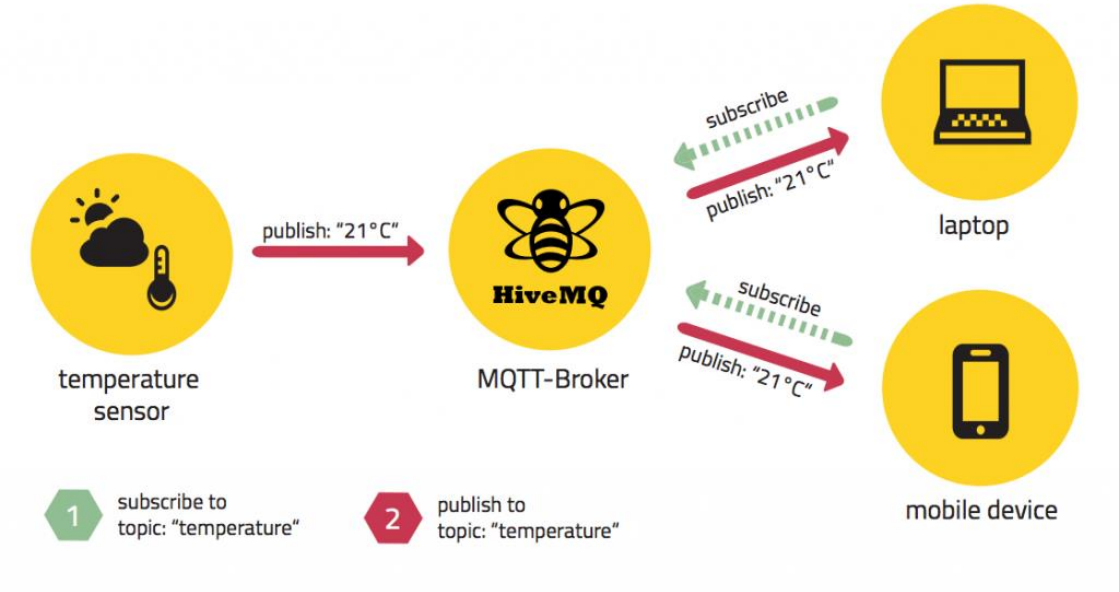


Figura 13 Publish/Suscribe
Fuente: (HiveMQ, 2015)

Para realizar el desacoplamiento entre remitentes y suscriptores el "bróker" contempla tres dimensiones:

- Espacio; Remitentes y Suscriptores no necesitan conocerse entre sí (dirección IP o puerto).
- Tiempo; Remitentes y Suscriptores no necesitan estar en ejecución o conectados al mismo tiempo.
- Sincronización; Las operaciones en ambos elementos (remitentes y suscriptores) no son interrumpidas durante la publicación o suscripción.

Filtrado de Mensajes:

Para que el "broker" pueda filtrar los mensajes entrantes, es posible hacer uso de las siguientes opciones de filtrado dependiendo del caso.

Opción 1; Basados por el tópico, el filtrado se basa en un tema o tópico, que es parte de cada mensaje. El cliente receptor se adhiere sobre los temas que le interesa con el corredor ya partir de ahí se pone toda mensaje en función de los temas suscritos. Los tópicos están en cadenas

generales con una estructura jerárquica, que permiten el filtrado basado en un número limitado de expresión.

Opción 2; Basados por el contenido, el "bróker" realiza el filtrado en base al contenido del mensaje, ya que siendo así existe una gran desventaja de esto es, que el contenido del mensaje debe ser conocida de antemano y no se puede cifrar o modificar fácilmente.

Opción 3; Basados por el tipo, generalmente en lenguajes orientados a objetos es una práctica común filtrar los mensajes en base al tipo o clase. Siendo así, un suscriptor puede esperar que el broker filtre todos los mensajes de cierta clase.

El uso del patrón publish/subscribe proporciona mayor escalabilidad, gracias al desacoplamiento entre remitentes y suscriptores. Esta ventaja genera algunos desafíos que deben ser tomados en cuenta. En el caso de filtrado por tópico, es importante que tanto remitentes y suscriptores conozcan con anterioridad los tópicos con los que trabajaran. Otro aspecto muy importante es la entrega de mensajes y que el remitente no puede asumir que el mensaje haya sido recibido por algún suscriptor, es posible que el mensaje nunca haya sido recibido por un suscriptor.

2.5. Protocolos de Comunicación MQTT

MQTT[8] es un protocolo usado para la comunicación máquina a máquina M2M[9] que actualmente es considerado el protocolo para el Internet De Las Cosas, ya que está orientado a la comunicación de sensores, debido a que consume muy poco ancho de banda y puede ser utilizado en la mayoría de dispositivos embebidos con pocos recursos en cuanto a CPU y RAM. Creado por el Dr. Andy Stanford-Clark de IBM y Arlen Nipper de Arcom en 1999, este protocolo es utilizado en diferentes industrias. El año 2013 este protocolo inicia el proceso de estandarización hasta noviembre de 2014 donde pasa a ser un estándar por parte de la

[8] MQTT – Message Queue Telemetry Transport es un Protocolo de mensajería, diseñado para las conexiones en lugares con ancho de banda limitada.

[9] M2M – Machine to Machine.

organización OASIS. Sin embargo la especificación del protocolo se encontraba publicada bajo una licencia libre de regalías durante varios años.

MQTT funciona bajo el protocolo TCP/IP y está diseñado para transportar mensajes de la manera más ligera posible haciendo uso del patrón "publish/subscribe". Este protocolo es útil para conexiones en lugares remotos en los que el ancho de banda de la red es limitado (MQTT, 2004).

MQTT está diseñado para aplicaciones que envíen datos de telemetría[10] o desde sondas espaciales en consecuencia use poco ancho de banda y reduzca el consumo de batería. Por ello que Facebook lanza la aplicación Facebook Messenger haciendo uso del protocolo MQTT para solucionar problemas de lentitud (Zhang, 2011).

La Tabla 6 muestra los métodos del protocolo MQTT y su respectiva descripción.

Tabla 6 Métodos del Protocolo MQTT

Método	**Descripción**
Connect	Espera a que la conexión sea establecida con el servidor.
Disconnect	Espera que el cliente MQTT termine el trabajo que tiene que hacer, para que la sesión TCP/IP sea desconectada.
Subscribe	Espera la finalización del método Suscribe o UnSuscribe.
UnSubscribe	Solicita al servidor que cancele la suscripción del cliente de uno o más temas.
Publish	Devuelve inmediatamente al subproceso de la aplicación después de pasar la solicitud al cliente MQTT.

2.6. Metodología de desarrollo Scrum

Una buena práctica de organización de proyectos es hacer uso de una metodología de desarrollo para poder así organizar al equipo, los tiempos y recursos. En la actualidad por el gran nivel competente entre empresas es necesario producir software de la manera más rápida posible, de esta forma es posible obtener una retro alimentación de los usuarios para poder realizar mejoras.

[10] Telemetría - Tecnología que permite la medición remota de magnitudes físicas y el posterior envío de la información hacia el operador del sistema.

En la actualidad, las diferentes metodologías ágiles ganan popularidad entre empresas al permitir la entrega constante de productos funcionales.

La metodología Scrum fue desarrollada por Ken Schwaber y Jeff Shuterland en los años 90 Scrum es un marco de trabajo utilizado para la construcción de productos complejos. Scrum al no ser un proceso o técnica, pero este cuenta con diferentes procesos y técnicas. Scrum muestra de forma clara la relación eficaz entre la administración de un producto y las buenas prácticas de desarrollo (Schwaber, K. & Sutherland, J., 2016).

Scrum como se observa en la Figura 14, es un marco de trabajo con el cual es posible trabajar con problemas complejos con los que lidian diferentes equipos haciendo entrega de productos de valor alto de una forma productiva y creativa.

Scrum al ser un marco de trabajo ágil, sigue los principios planteados en el manifiesto ágil. La

Tabla 7 muestra estos principios (Beck, K.; Beedle, M.; Bennekum, A.; Cockburn, A.; Cunningham, W.; Fowler, M.; Grenning, J.; Highsmith, J.; Hunt, A.; Jeffries, R.; Kern, J.; Marick, B.; Martin, R.; Mellor, S.; Schwaber, K.; Sutherland, J. & Thomas, D., 2001).

Tabla 7 Principios del manifiesto ágil

Nro.	Principio de manifiesto ágil
1	Satisfacer al cliente mediante la entrega temprana y continua de software con valor.
2	Es aceptable que los requisitos cambien, incluso en etapas tardías de desarrollo. Los procesos aǵiles aprovechan el cambio para proporcionar una ventaja competitiva al cliente.
3	Entrega frecuente de software funcional, entre dos semanas y dos meses, con preferencia al periodo de tiempo más corto posible
4	Los responsables de negocio y desarrolladores trabajan juntos de forma cotidiana durante todo el proyecto
5	Los proyectos se desarrollan en torno a individuos motivados. Es necesario brindarles el entorno y apoyo que estos requieren así confiarles la ejecución del trabajo
6	El método más eficiente y efectivo de comunicar información al equipo de desarrollo y entre sus miembros es la conversación cara a cara
7	El software funcional es la medida principal de progreso
8	Los procesos ágiles promueven el desarrollo sostenible. Los promotores, desarrolladores y usuarios deben ser capaces de mantener un ritmo constante de forma indefinida
9	La atención continua a la excelencia técnica y al buen diseño mejora la agilidad
10	La simplicidad, o el arte de maximizar la cantidad de trabajo no realizado es esencial

11	Las mejores arquitecturas, requisitos y diseños surgen de equipos auto-organizados
12	A intervalos regulares el equipo reflexiona sobre cómo ser más efectivo para poder así ajustar y perfeccionar su comportamiento en consecuencia.

Figura 14 Etapas de la metodología Scrum
Fuente: (ScrumAlliance, s.f.)

2.6.1. Roles de Scrum

El Equipo Scrum consiste en un Product Owner[11], Development Team[12] y un Scrum Master[13]. Los Equipos Scrum son auto organizados y multifuncionales. Los equipos auto organizados eligen la mejor forma de llevar a cabo su trabajo y no son dirigidos por personas externas al equipo. Los equipos multifuncionales tienen todas las competencias necesarias para llevar a cabo el trabajo sin depender de otras personas que no son parte del equipo. El modelo de equipo en Scrum está diseñado para optimizar la flexibilidad, la creatividad y la productividad.

Los Equipos Scrum entregan productos de forma iterativa e incremental, maximizando las oportunidades de obtener retroalimentación. Las entregas incrementales de producto "Terminado" aseguran que siempre estará disponible una versión potencialmente útil y funcional del producto.

[11] Product Owner – Dueño del producto.

[12] Depelopment Team - Equipo de Desarrollo.

[13] Scrum Master – Lider del equipo Scrum

Product Owner, este miembro es el responsable de maximizar el valor del producto y el trabajo del Equipo de Desarrollo. El Product Owner es el único miembro responsable del backlog del producto. Por lo general este miembro es una sola persona y no un comité, sin embargo, este se encarga de plasmar los requerimientos de algún comité en el backlog del producto. Las decisiones realizadas por el Product Owner se hacen visibles en el backlog del producto. De esta forma, nadie más que el Product Owner puede ordenar que el Equipo de Desarrollo trabaje en otro tipo de requerimientos, y el Equipo de desarrollo no tiene permitido en trabajar en base a que otros digan.

Development Team, es el Equipo de Desarrollo consiste de profesionales que realizan el trabajo de realizar entregas de un producto hecho al final de cada sprint o "iteración". Un Equipo de Desarrollo es auto-organizado y funcional para poder así maximizar la efectividad del equipo.

Por lo general un equipo de desarrollo debe estar formado entre tres y nueve elementos, de esta forma no tener conflictos de organización ni minimizar los resultados en cada sprint.

Scrum Master, es responsable de asegurar que todas las actividades de Scrum se lleven de forma correcta. Es necesario que este rol muestre y haga entender la teoría, reglas y prácticas de Scrum. Es necesario que el Scrum Master ayude a los elementos fuera del equipo a entender cuáles son las interacciones con el equipo son útiles y cuáles no, de este modo el Scrum Master ayudará a modificar y adaptar estas interacciones para poder maximizar el valor de todo el equipo en conjunto.

2.6.2. Eventos de Scrum

En Scrum existen eventos predefinidos con el fin de crear regularidad y minimizar la necesidad de reuniones no definidas en Scrum. Todos los eventos son bloques de tiempo, de tal modo que todos tienen una duración máxima. Una vez que comienza un Sprint, su duración es fija y no puede acortarse o alargarse. Los demás eventos pueden terminar siempre que se alcance el objetivo del evento, asegurando que se emplee una cantidad apropiada de tiempo sin permitir desperdicio en el proceso.

Además del propio Sprint, que es un contenedor del resto de eventos, cada uno de los eventos de Scrum constituye una oportunidad formal para la inspección y adaptación de algún aspecto.

Estos eventos se diseñaron específicamente para habilitar los pilares vitales de transparencia e inspección. La falta de alguno de estos eventos da como resultado una reducción de la transparencia y constituye una oportunidad perdida de inspección y adaptación.

Sprint – El corazón de Scrum se yace en el Sprint o iteración, la cual cuenta con un lapso de tiempo menor o igual a un mes en el cual es necesario presentar una versión incrementada, utilizable, con el potencial de ser el producto final en base a una meta inicial. Un Sprint es consistente por los esfuerzos de desarrollo empleados. Un Sprint inicia al finalizar un Sprint anterior. Un Sprint consiste de la Planificación de Sprint, Daily Scrum, Revisión del Sprint y la Retrospectiva del Sprint.

Planificación de Sprint – Este evento realiza la planificación de todas las metas a cumplirse durante el Sprint, la duración de esta planificación debe ser menor o igual a ocho horas, sin embargo, la duración de la planificación es directamente proporcional a la duración del Sprint, es decir, la planificación de un Sprint corto debe ser corta. En este evento el Scrum Master debe encargarse de que el equipo entienda el propósito del Sprint a realizarse.

La meta del Sprint es un conjunto de objetivos que pueden hacerse gracias al backlog del producto. La meta es una guía para el equipo de desarrollo para la auto-organización de este. Estos objetivos son un conjunto de historias de usuario a desarrollarse que se obtienen del backlog del producto a las cuales se asignan tareas, puntos de complejidad, encargados que realizaran la historia de usuario.

Daily Scrum – Es una reunión que se realiza todos los días con una duración máxima de quince minutos en la que todos los miembros del equipo tocan los siguientes puntos:

- Que se realizó el día anterior.
- Que se realizará en la jornada presente.
- La existencia de algún elemento que bloquee al miembro del equipo.

Revisión del Sprint – Este evento se realiza al haber finalizado el Sprint para poder inspeccionar las mejoras en el producto trabajado y verificar los cambios que se realizaron al backlog del producto. Para un Sprint de un mes se recomienda que este evento dure a lo mucho cuatro horas.

Los diferentes miembros del equipo comparten los avances que se realizaron durante el Sprint con las partes interesadas en el proyecto, por lo general clientes que son invitados previamente por el Product Owner.

El Product Owner debe los elementos del backlog del producto que fueron realizados, basados en eso, el Equipo de Desarrollo demuestra el trabajo que realizó y responde a las preguntas respecto al trabajo realizado en el Sprint actual. Es posible que el mercado al que apunta la aplicación haya cambiado, es por ello que en este evento es necesario aclarar este tipo de tópicos para poder aumentar el valor de la siguiente iteración. En general, el grupo completo colabora de tal forma que se consiguen puntos a tocar en la planificación del siguiente Sprint que pueda aumentar el valor del producto.

Retrospectiva del Sprint – Este evento permite que el equipo encuentre falencias o propuestas para mejorar el siguiente Sprint. Este evento se lleva a cabo tras haber concluido la revisión del Sprint y antes de comenzar la planificación del siguiente Sprint. Se recomienda que la duración de este evento sea a lo mucho de tres horas para sprints de un mes.

Durante este evento el equipo debe planificar diferentes formas para poder incrementar la calidad del producto. Para ello es necesario inspeccionar como fueron las personas del equipo, relaciones y uso de las herramientas de tal forma que sea posible identificar falencias o mejoras. De este modo, al finalizar la Retrospectiva del Sprint el equipo habrá encontrado mejoras que puedan ser aplicadas al siguiente Sprint. No es obligatorio que estas mejoras sean encontradas y aplicadas durante y después de la Retrospectiva del Sprint, estas mejoras pueden ser aplicadas en cualquier momento de un Sprint.

2.6.3. Artefactos de Scrum

Los artefactos de Scrum representan trabajo o valor en diversas formas que son útiles para proporcionar transparencia y oportunidades para la inspección y adaptación. Los artefactos definidos por Scrum están diseñados específicamente para maximizar la transparencia de la información clave, necesaria para asegurar que todos tengan el mismo entendimiento del artefacto.

Backlog del Producto – El Backlog del Producto es una lista los requerimientos, funciones, características, mejoras y arreglos necesarios para el producto, además de ser la única fuente de requerimientos al cual pueden realizarse cambios. El Product Owner es el encargado de mantener y gestionar este artefacto incluyendo su contenido, disponibilidad y orden.

Al ser Scrum una metodología iterativa, el backlog del producto nunca se encontrará finalizada, mientras exista el producto existirá el backlog del producto. Inicialmente el contenido de este artefacto solo muestra los requerimientos que fueron comprendidos. Este artefacto evoluciona a medida que el producto y el entorno en el que el producto funcionará evolucionan, es decir, cualquier cambio en requerimientos de negocio o condiciones de mercado puede causar cambios en el backlog del producto.

Es necesario refinar el backlog del producto adicionando estimados, detalles y orden a los elementos que yacen en este. Durante este proceso en el que trabajan el Product Owner y el Equipo de desarrolla los elementos del backlog del producto son revisados.

Backlog del Sprint – El backlog del sprint es un conjunto de elementos del backlog del producto que son elegidos para llevarse a cabo durante un Sprint, además de tener un plan para entregar la mejora del producto y cumplir la meta del Sprint. El backlog del sprint es un pronóstico del equipo de desarrollo sobre que funcionalidad tendrá la siguiente mejora del producto y el trabajo necesario para poder finalizar dichas funcionalidades. Este artefacto hace visible todo el trabajo necesario identificado por el Equipo de Desarrollo para cumplir la meta del Sprint.

A medida que más trabajo es necesario, es necesario adicionarlo al backlog del sprint. Por otro lado, a medida que se realice el trabajo o se lo complete, el trabajo restante estimado se actualiza.

A diferencia del backlog del producto, el backlog del sprint solo es administrado por el Equipo de Desarrollo, de tal forma que solo este equipo puede realizar cambios en este artefacto, adicionar elementos necesarios o remover elementos innecesarios.

Incremento o mejora del producto – Al final un sprint el incremento o mejora del producto es el total de todos los elementos completados que pertenecen al backlog del sprint y el valor de las mejoras del producto de sprints anteriores. Al finalizar un sprint un nuevo incremento está

hecho lo que significa que este incremento debe encontrarse en una condición útil independientemente que el Producto Owner decida que el incremento se publicará o no.

CAPÍTULO 3
MARCO APLICATIVO

3.1. Backlog Inicial de Historias de Usuario

La Tabla 8 muestra el backlog inicial de historias de usuario para el sistema embebido, por orden de importancia.

Tabla 8 Backlog inicial de Historias de Usuario

Nro.	Nombre de historia de Usuario
1	Identificación de tecnologías
2	Diseño de modelo de datos E/R[14] de la base de datos
3	Implementación de la Base de Datos
4	Diseño inicial de pantallas de la aplicación móvil – Autenticación
5	Servicios con RESTful – Añadir usuarios de confianza
6	Conexión Arduino y Sensor MQ3 – Sistema Embebido
7	Conexión Arduino, GPS y SIM900 – comunicación con el servidor
8	Integración servidor – aplicación móvil
9	Realización de servicios con RESTful – Obtención de datos relevantes para su visualización en la aplicación móvil.

3.2. Descripción de Historias de Usuario identificadas

Para lograr los objetivos planteados en el Capítulo 1, es necesario tener todas las historias de usuario, así poder tener clara las características que se desean realizar.

Como punto de partida se identifican 10 historias de usuario H.U.[15] divididas en categorías identificadas, desde la Tabla 9 hasta la Tabla 17.

Tabla 9 H.U. Identificación de tecnologías

Historia de Usuario		
ID: 1	**Nombre:** Identificación de Tecnologías	
	Usuario: Equipo Desarrollo	**Categoría:** Ninguna
Descripción: Identificación de tecnologías para la realización del prototipo en cuanto a hardware y software.		

[14] E/R – Entidad/Relación.

[15] H.U. – Historias de Usuario.

Tabla 10 H.U. Diseño Modelo E/R de la base de datos.

Historia de Usuario		
ID: 2	**Nombre:** Diseño de modelo de datos E/R de la base de datos	
	Usuario: Equipo Desarrollo	**Categoría:** Ninguna
Descripción: Diseño inicial del modelo de datos teniendo en cuenta el dispositivo y la aplicación móvil.		

Tabla 11 H.U. Implementación de la Base de datos.

Historia de Usuario		
ID: 3	**Nombre:** Implementación de la Base de Datos	
	Usuario: Equipo Desarrollo	**Categoría:** Ninguna
Descripción: Implementación del diseño inicial del modelo de datos, con sus respectivas pruebas.		

Tabla 12 H.U. Diseño inicial de pantallas de la aplicación móvil.

Historia de Usuario		
ID: 4	**Nombre:** Diseño inicial de pantallas de la aplicación móvil – Registro usuario nuevo	
	Usuario: Usuario Móvil	**Categoría:** Aplicación Móvil
Descripción: Como usuario móvil, deseo una interfaz de usuario entendible y fácil de usar, visualizar mapas y ver que amigos/familiares de confianza tengo.		

Tabla 13 H.U. Servicios RESTful – Añadir usuarios de confianza.

Historia de Usuario		
ID: 5	**Nombre:** Servicios con RESTful – Añadir usuarios de confianza	
	Usuario: Usuario Aplicación Móvil	**Categoría:** RestFul API
Descripción: Como usuario de la aplicación móvil, se desea poder añadir a otros usuarios como amigos/familiares de confianza, quienes podrán ver la ubicación de la movilidad.		

Tabla 14 H.U. Conexiones Hardware – Arduino y Sensor MQ3.

Historia de Usuario		
ID: 6	**Nombre:** Conexión Arduino y Sensor MQ3	
	Usuario: Propietario de movilidad	**Categoría:** Sistema Embebido
Descripción: Como propietario de una movilidad, deseo un dispositivo que mida la concentración de alcohol cerca el volante de una movilidad de manera constante y que a su vez no permita la conducción del vehículo si supera el límite establecido por ley.		

Tabla 15 H.U. Arduino, GPS y SIM900 – Comunicación con el servidor.

Historia de Usuario		
ID: 7	**Nombre:** Conexión Arduino, GPS y SIM900 – comunicación con el servidor	
	Usuario: Propietario de movilidad	**Categoría:** Sistema Embebido
Descripción: Como propietario de una movilidad, deseo que mi movilidad envíe notificaciones con su ubicación a familiares/amigos de confianza.		

Tabla 16 H.U. Integración Servidor – Aplicación móvil

Historia de Usuario		
ID: 8	**Nombre:** Integración servidor – aplicación móvil	
	Usuario: Usuario móvil, dueño del movilidad	**Categoría:** Aplicación Móvil
Descripción: Como usuario móvil, deseo poder guardar mi información en la nube como ser las ultimas notificaciones y los datos de amigos de confianza.		

Tabla 17 H.U. Visualización de información en la aplicación Android

Historia de Usuario		
ID: 9	**Nombre:** Realización de servicios con RESTful – Obtención de datos relevantes para su visualización en la aplicación móvil.	
	Usuario: Usuario Aplicación Móvil	**Categoría:** RestFul API
Descripción: Como usuario de la aplicación móvil, se desea tener acceso a información de la ubicación de familiares u amigos que intenten o estén conduciendo en estado de ebriedad.		

3.3. Sprints

En esta sección se realiza los sprints completando así los alcances propuestos en la sección 1.7.1, cada Sprint cuenta con su Backlog detallado y estas a su vez con las tareas a realizar para cada historia de usuario, los casos de prueba para verificar la funcionalidad de todo el sistema.

3.3.1. Sprint 0

En este Sprint se define un esquema inicial como estructura base del proyecto, diseño del modelo de datos, definición de herramientas y tecnologías con las cuales se desarrolla el prototipo.

Este Sprint se realizó desde el 25 de septiembre hasta 10 de octubre de 2016, en la Tabla 18 se muestra las historias de usuario *"Sprint Backlog"* con respecto al Sprint 0 y para cada una sus respectivas tareas a realizar.

Tabla 18 Sprint 0 – Backlog

ID	Nombre de Historia de Usuario
1	Identificación de tecnologías
Tareas: - Identificar tecnologías para el Backend - Identificar tecnologías para el Frontend	
2	Diseño de modelo de datos E/R de la base de datos
Tareas: - Diseñar el modelo de datos con el cual inicia	
3	Implementación de la Base de Datos
Tareas: - Implementar la base de datos previa selección de tecnologías	
4	Diseño inicial de pantallas de la aplicación móvil – Registro usuario nuevo
Tareas: - Crear bosquejos del posible diseño de la aplicación móvil - Plasmar los bosquejos en Android	

La Tabla 19 muestra las tecnologías identificadas para la implementación del prototipo.

Tabla 19 Identificación de tecnologías para el desarrollo

Tecnologías Backend
- Framework Laravel para el desarrollo web y creación de la API RESTful - NodeJS con la librería SockJS para el uso de Websockets en el lado del servidor - Mosquitto Broker para la comunicación con el hardware a través del protocolo MQTT - MySql para almacenar el modelo de datos - Arduino para la programación de Hardware - Android Studio
Tecnologías Frontend
- Material Design para la creación de la aplicación móvil

En la Tabla 20 se observa los casos de prueba del Sprint.

Tabla 20 Casos de prueba – Sprint 0

Detalle
Posibilidad de crear una cuenta en la aplicación móvil
Precondiciones
- Usuario nuevo, es decir no haya registrado su número de celular ni su correo electrónico en la aplicación móvil

Nro.	Pasos a realizar	Resultados a esperar
1	Entrar en la aplicación donde está la pantalla de inicio de sesión, hacer click en “register”	Ver la pantalla de registro
2	Ingresar datos requeridos por los campos.	Visualizar el menú de la aplicación.
Detalle		
Posibilidad Iniciar y cerrar sesión en la aplicación móvil		
Precondiciones		
- Ningún otro usuario haya iniciado sesión en la aplicación móvil		
Nro.	**Pasos a realizar**	**Resultados a esperar**
1	Entrar a la pantalla de inicio de sesión	Visualizar la pantalla con los campos de usuario y contraseña.
2	Ingresar usuario y contraseña valida	Haber redirigido a la pantalla de inicio de la aplicación donde se encuentra el menú.
3	En el menú hacer click en “logout” para finalizar sesión	

El incremento en este Sprint consta de la identificación de tecnologías a utilizar, registro de usuarios, control de sesiones de los mismos.

La Figura 15 se muestra la pantalla de registro y la Figura 16 el menú principal de la aplicación móvil.

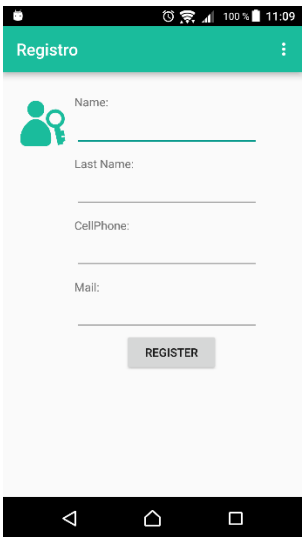


Figura 15 Pantalla de Registro

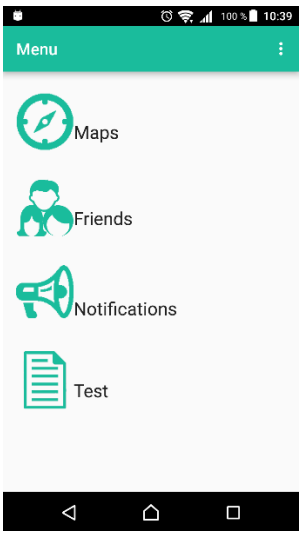


Figura 16 Menú de la Aplicación móvil

3.3.2. Sprint 1

En este Sprint se realiza la aplicación móvil en su totalidad, con todas las funcionalidades necesarias, como ser la adición de amigos/familiares de confianza, mostrar datos en una mapa y un historial de las últimas notificaciones.

Este Sprint se realizó desde el 11 de octubre hasta 10 de noviembre de 2016, en la Tabla 21 detalla las historias de usuario *"Sprint Backlog"* con referente al Sprint 1 junto a las tareas a realizar en cada una de las historias de usuario.

Tabla 21 Sprint 1 – Backlog

ID	**Nombre de Historia de Usuario**
8	Integración servidor – aplicación móvil
Tareas: - Realizar la conexión la Aplicación Móvil con el servidor mediante un API - Realizar la conexión segura mediante HTTPS - Guardar información en la base de datos desde la Aplicación Móvil	
5	Servicios con RESTful – Añadir usuarios de confianza
Tareas: - Realizar el modulo para añadir Amigos/familiares de confianza con Laravel, datos que se enviaran desde la aplicación móvil	
9	Realización de servicios con RESTful – Obtención de datos relevantes para su visualización en la aplicación móvil
Tareas: - Realizar métodos con Laravel que retorne datos relacionados con un determinado usuario - Realizar métodos en Android Studio para mostrar en la aplicación móvil del usuario.	

En la Tabla 22 se observa los casos de prueba realizados en el Sprint.

Tabla 22 Casos de prueba – Sprint 1

Detalle		
Añadir Usuarios de confianza		
Precondiciones		
- Usuario tendrá que haber iniciado sesión		
Nro.	**Pasos a realizar**	**Resultados a esperar**
1	En el menú principal de la aplicación ingresar a "Friend"	Se podrá ver una lista de contactos amigos/familiares de los cuales se podrá seleccionar cuales son de su entera confianza. Esta selección de contactos se

		guarda de manera automática en la base de datos del prototipo
Detalle		
Obtener información de ultimas notificaciones		
Precondiciones		
- Usuario tendrá que haber iniciado sesión		
Nro.	**Pasos a realizar**	**Resultados a esperar**
1	En el menú principal de la aplicación móvil, ingresar a “Notifications”	Se redirige a una pantalla con un listado de las ultimas 10 notificaciones o alertas que se haya recibido.
2	En la pantalla con la lista de notificaciones, hacer click en la notificación de interés	Se desplegara la ventana de Maps, donde se podrá observar la ubicación de la movilidad la cual envió la notificación
Detalle		
Obtener datos de mapas		
Precondiciones		
- Usuario tendrá que haber iniciado sesión		
Nro.	**Pasos a realizar**	**Resultados a esperar**
1	En el menú principal de la aplicación móvil hacer click en “Maps”	Sera redirigido a la pantalla Maps donde se podrá observar la ubicación de un determinado vehículo.

El incremento en este Sprint consiste en la aplicación móvil que recibe notificaciones, muestra en un mapa la ubicación de movilidades y la posibilidad de añadir amigos/familiares, la aplicación está preparada para recibir mensajes del dispositivo embebido, el cual se detallara en el siguiente Sprint.

La Figura 17 muestra un listado de amigos/familiares del propietario de una movilidad a quienes le llegará notificaciones.

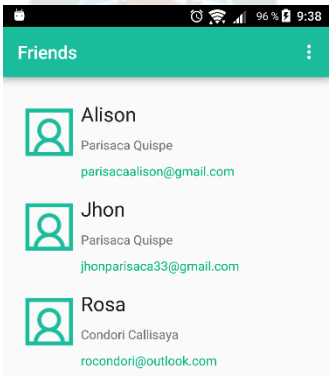
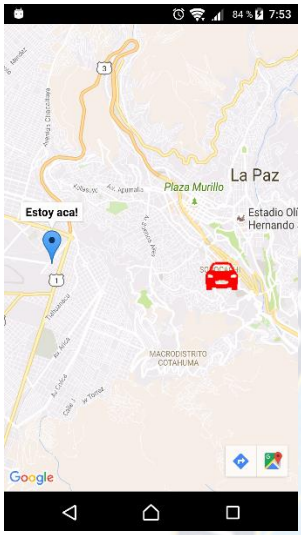


Figura 17 Amigos/familiares de confianza en la aplicación

La Figura 18 muestra una pantalla con la ubicación de una determinada movilidad, dato que fue enviado desde el sistema embebido y en la Figura 19 se muestra un listado de las últimas notificaciones.



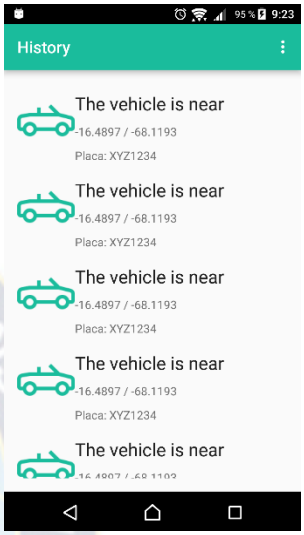


Figura 18 Pantalla de ubicación de movilidades *Figura 19 Pantalla de Últimas 15 notificaciones*

3.3.3. Sprint 2

En este Sprint se realiza el sistema embebido o dispositivo, el cual estará en constante funcionamiento, dicho dispositivo estará analizando la concentración del alcohol etílico cerca del volante, enviado datos relevantes de una movilidad si el grado alcohólico fuese mayor al establecido por ley.

Este Sprint se realizó desde el 11 de noviembre hasta 22 de noviembre de 2016, en la Tabla 23 muestra las historias de usuario *"Sprint Backlog"* del Sprint 2 con sus respectivas tareas a realizar en cada historia de usuario.

Tabla 23 Sprint 2 – Backlog

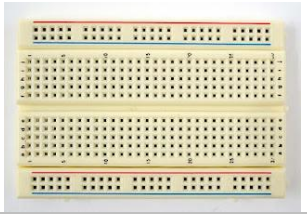
ID	**Nombre de Historia de Usuario**
6	Conexión Arduino y Sensor MQ3
Tareas: - Realizar la conexión de los componentes 1,2, 3 y 6 de la , que será el ***sistema embebido inicial***. - Realizar el programa para el análisis de concentración de alcohol en el aire, teniendo así el grado alcohólico del conductor - Realizar un indicador que muestre que una movilidad no puede ser conducida	

7	Conexión Arduino, GPS y SIM900 – comunicación con el servidor
Tareas: - Realizar la conexión del ***sistema embebido inicial*** con los componentes 4 y 5 de la logrando así el sistema embebido en su totalidad. - Realizar el programa para la obtener datos del módulo GPS - Realizar el envió de datos al servidor el indicador sea activado mediante el módulo SIM900 haciendo uso del protocolo MQTT.	

La Tabla 24 muestra los componentes principales y necesarios para la construcción y programación del sistema embebido.

Tabla 24 Componentes para el sistema embebido

Número	Nombre de componente	Figura de componente
1	Arduino	
2	Sensor MQ3	
3	Leds	
4	Modulo GPS	
5	Módulo SIM900	

6	Protoboard	

En la Tabla 25 se observa los casos de prueba del Sprint.

Tabla 25 Casos de Prueba – Sprint 2

Detalle		
Dispositivo tendrá que ser capaz de medir el grado alcohólico del conductor		
Precondiciones		
- Dispositivo conectado en el volante		
Nro.	**Pasos a realizar**	**Resultados a esperar**
1	Colocar el dispositivo a una distancia de 25 a 35 centímetros cerca de la persona que hará el papel de conductor y este haya ingerido bebidas alcohólicas.	El dispositivo analizará el aire, en cuanto a concentración de alcohol, cuando se haya pasado el límite establecido por ley, se activara el indicador que señala que la movilidad no puede ser conducida. Además se espera que el sistema embebido envié notificaciones a amigos/familiares de confianza.

El incremento en este Sprint está orientado a Hardware, por lo que el incremento es el sistema embebido que analiza el aire a un radio de 30 centímetros aproximadamente, si el límite de grado es sobrepasado, el sistema embebido enviara notificaciones a familiares/amigos del dueño del vehículo.

La Figura 20 muestra el monitoreo del aire desde el IDE de Arduino y Figura 21 muestra la conexión de Arduino, el sensor, el modulo GPS y Modulo SIM900, quedando así el sistema embebido terminado.

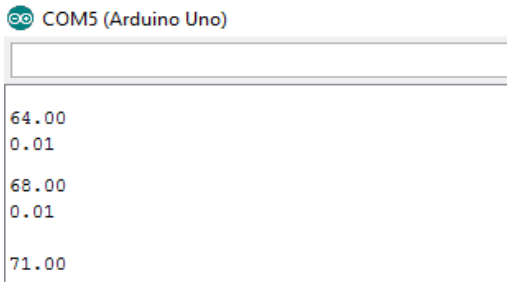


Figura 20 Análisis de la concentración de alcohol en el aire

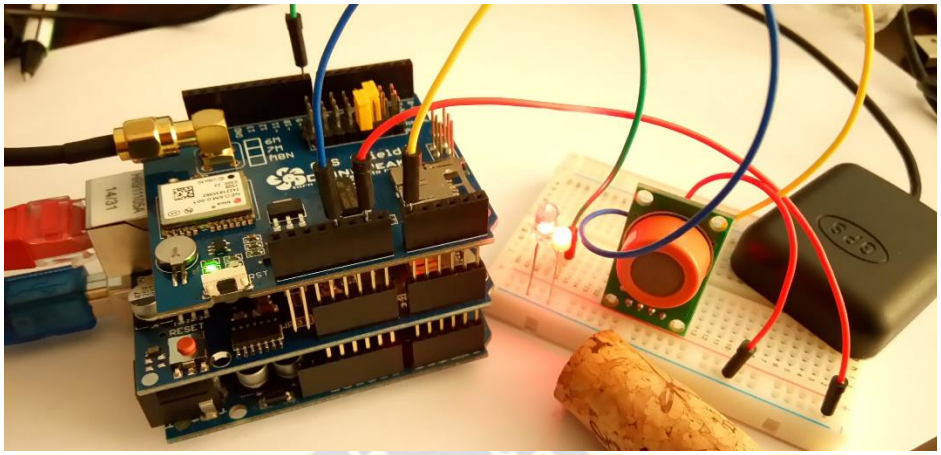


Figura 21 Conexión con GPS y SIM900 – Prueba con vino

3.4. Modelo entidad-relación E-R

En la Figura 22 se muestra el modelo entidad-relación del prototipo.

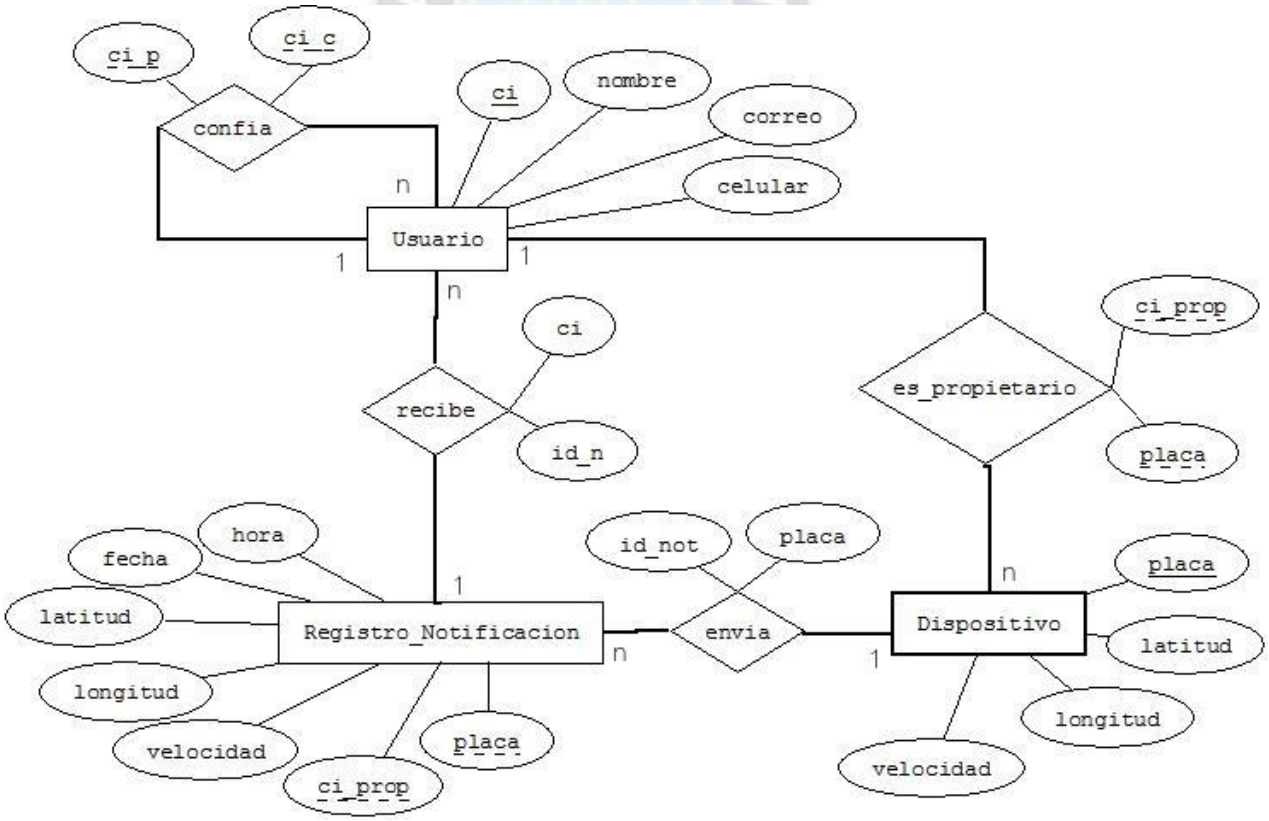


Figura 22 Modelo Entidad – Relación E-R del sistema embebido

3.5. Diagrama de clases

En la Figura 23 se muestra el diagrama de clases del prototipo teniendo en cuenta la abstracción.

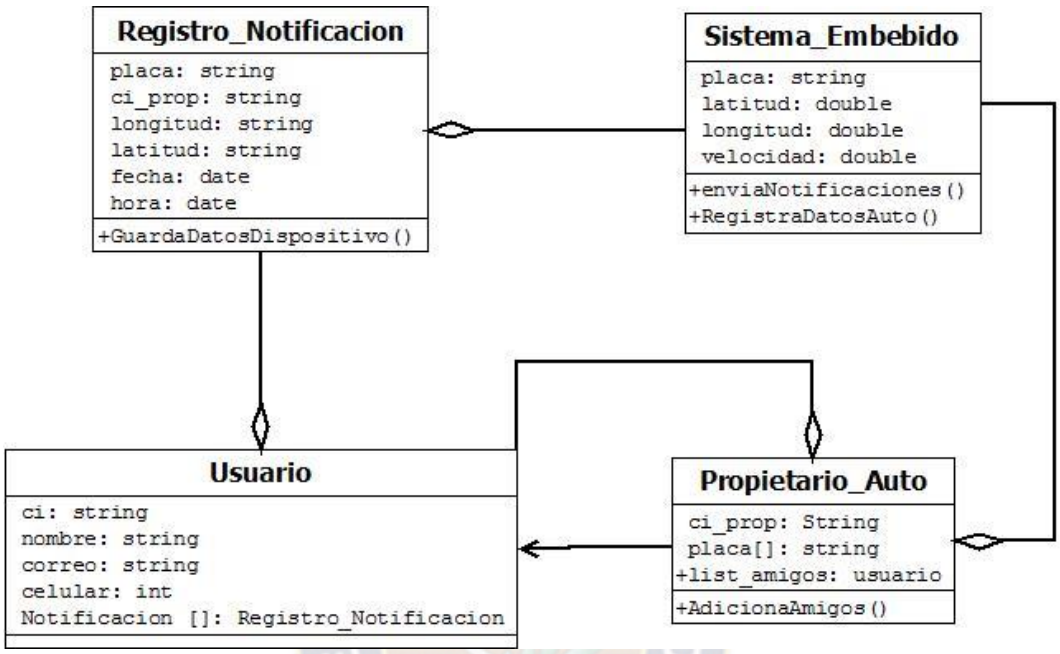


Figura 23 Diagrama de clases – Abstracción

CAPÍTULO 4
PRUEBAS Y RESULTADOS

4.1. Pruebas Nivel Hardware

El sistema embebido se encuentra en el volante donde es más proclive el análisis de gas etílico en el aire por consiguiente el grado alcohólico del conductor, ya que el sensor tiene una sensibilidad de aproximadamente 25 a 35 centímetros de distancia y está en constante monitoreo.

4.1.1. Casos encontrados

Es conveniente tomar casos de prueba las cuales se presentan en la Tabla 26.

Tabla 26 Casos de prueba encontrados

Nro.	Descripción de caso
1	Conductor no se encuentra en estado de ebriedad
2	Conductor ha ingerido bebidas alcohólicas antes de conducir pero no ha sobrepasado el límite establecido en Bolivia
3	Conductor bebe mientras conduce
4	Conductor en estado de ebriedad intenta conducir una movilidad.

4.1.2. Funcionamiento en cada caso

La Tabla 27 muestra los resultados del sistema embebido ante cada caso.

Tabla 27 Resultados del sistema embebido

Nro de caso	Descripción	Resultados
1	No existe presencia de gas etílico	El Software desarrollado en Arduino determina que no hay ningún problema, por tanto no envía ninguna notificación.
2	Presencia de gas etílico por poco tiempo	El tiempo promedio de grado alcohólico por cada sorbo de bebida alcohólica es de 2 a 3 minutos como se muestra en la Figura 24, por lo que antes de enviar notificaciones el sistema embebido analiza este tiempo evitando que la movilidad sea

		conducida, así el conductor puede intentar nuevamente conducir pasado esos 3 minutos.
3	Presencia de gas etílico mientras una movilidad está en marcha.	El sistema embebido tiene la capacidad de medir la velocidad de una movilidad, si en este proceso verifica que el conductor tiene grado alcohólico tiende a enviar notificaciones a familiares cada 2 minutos.
4	Conductor en estado de ebriedad	Si el grado alcohólico persiste en 2 intentos como se mencionó en el caso 2 y la movilidad no está en marcha, el sistema embebido procede a activar el indicador el cual señala que el conductor no es apto para conducir, además de enviar las notificaciones.

La Figura 24 muestra los datos captados por el sensor, en un tiempo de aproximadamente 2 a 3 minutos, prueba realizada con 20ml de vino. El grado alcohólico persiste por más tiempo si una persona supera el límite establecido en Bolivia.

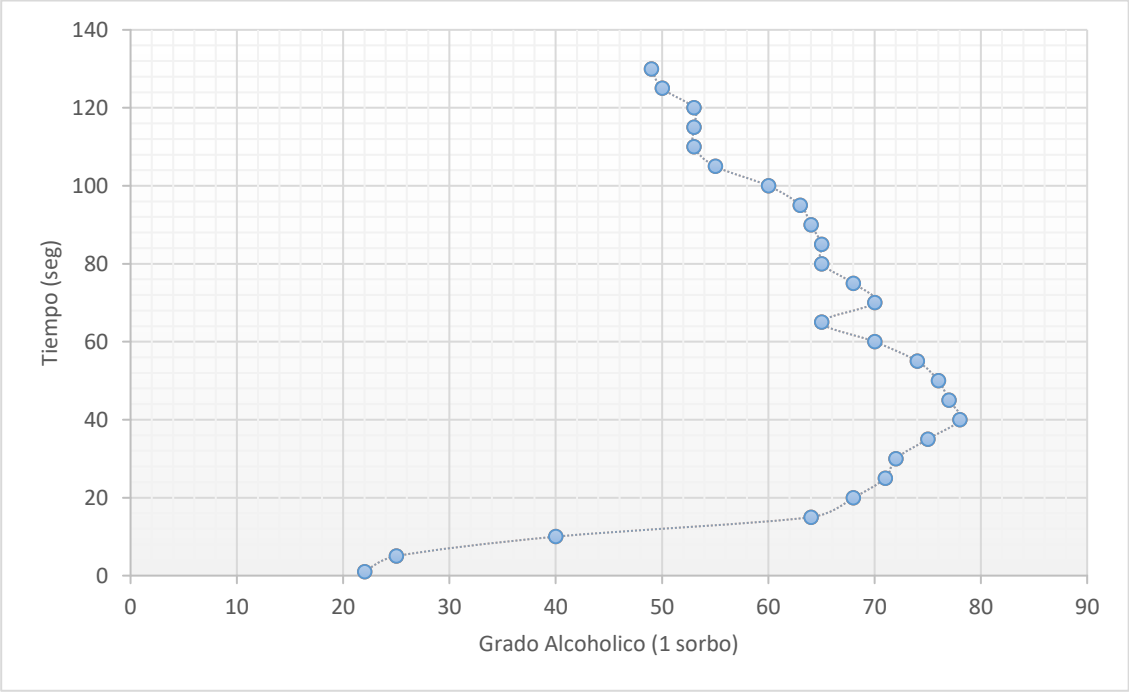


Figura 24 Grafica Sensor MQ3 – Gas etílico

4.1.3. Funcionamiento del sistema embebido

En la Figura 25 se muestra el funcionamiento del prototipo, donde el sistema embebido monitorea el grado alcohólico del conductor (1), procediendo a enviar notificaciones con la posición (2) a familiares/amigos (3) si se encuentra en estado de ebriedad.

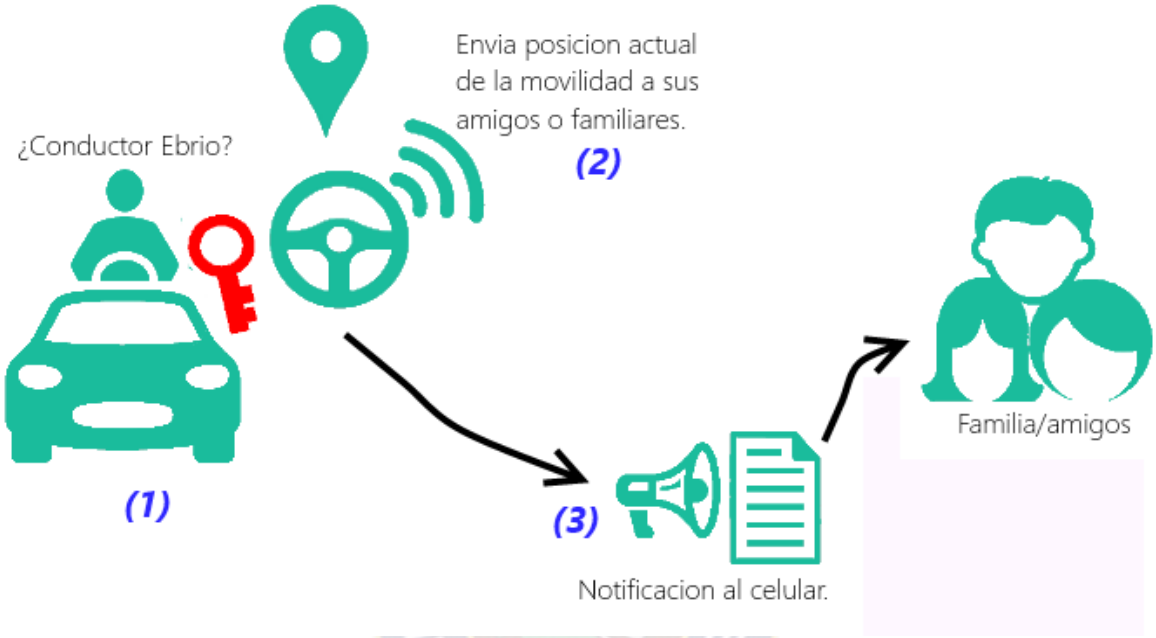


Figura 25 Esquema de funcionamiento del prototipo

Se observa en la Figura 26 la notificación recibida en el teléfono celular, el cual redirige a la aplicación mostrando en el mapa la ubicación de la movilidad como se ve en la Figura 27.

Figura 26 Notificación en el teléfono celular

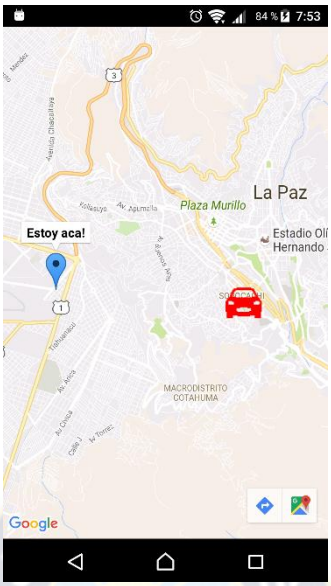


Figura 27 Mapa de ubicación de una movilidad con el prototipo

4.2. Pruebas Nivel Software

Las pruebas de estrés se realizan a los diferentes recursos de la API RESTful. Estas pruebas son realizadas haciendo uso del software Apache Bench para probar los servicios RESTful. Los elementos de prueba se realizan en un servidor local y un servidor remoto.

4.2.1. Resultados de Pruebas de Estrés en los servicios RESTful

La Tabla 28 muestra las pruebas de estrés a estos servicios constan de 5000 peticiones y un nivel de concurrencia de 1000, es decir, se realizarán 5000 peticiones con 1000 conexiones concurrentes. Las peticiones HTTP a realizar son del tipo de lectura con el método GET, ya que esta operación es la que realiza más procesamiento en el servidor debido a que se obtiene datos para mostrarlas en la aplicación móvil.

Tabla 28 Prueba de estrés de la aplicación móvil

Campo	Servidor Local	Servidor Remoto
Peticiones por segundo [seg]	23.43	27.28
Tiempo por petición (media) [ms]	42671.766	36658.560
Tiempo por petición (media en todas las peticiones concurrentes) [ms]	42.672	36.659
Velocidad de transferencia [Kbyts/seg]	39.65	41.72

El tiempo total en realizar las 5000 peticiones con un nivel de concurrencia de 1000 en el servidor local y remoto es de 213.359 y 183.293 segundos respectivamente. En el servidor local y remoto se identifica que el tiempo de respuesta comienza a demorar aproximadamente desde la petición 2250 y 4000 respectivamente.

CAPÍTULO 5
CONCLUSIONES Y RECOMENDACIONES

5.1. Conclusiones

- La lógica del sistema embebido responde adecuadamente a los requerimientos de los casos de estudio mostrada en la Sección 4.1.2
- El Internet de las Cosas influye en el sistema embebido ya que se comunica con los familiares del conductor con grado alcohólico mayor al establecido conectándose a Internet, el cual se muestra en la Sección 3.3.3
- La aplicación móvil muestra la ubicación de movilidades que han sido notificadas por el sistema embebido.

5.2. Estado de objetivos específicos

- Se implementó la base de datos donde el prototipo registra datos relevantes como ser la placa, ubicación, velocidad de una movilidad inclusive las notificaciones que esta envié cuando el grado alcohólico del conductor haya pasado el límite permitido por ley en Bolivia.
- Se realizó el sistema embebido el cual analiza la concentración de gas etílico en el aire y por consiguiente el grado alcohólico del conductor, el cual tiene un indicador que muestra que la movilidad no puede ser conducida si el grado alcohólico del conductor pasa el límite.
- Se realiza el envío de notificaciones satisfactoriamente, de manera que el sistema se conecta a internet, siendo aplicado en concepto del Internet de las Cosas.
- El sistema embebido tiene la capacidad de calcular la velocidad de la movilidad, por lo que si el conductor se encuentra en estado de ebriedad y conduciendo, el sistema procede a activar el indicador y enviar notificaciones.
- Se realizó la aplicación móvil en Android donde se muestra en un mapa la ubicación de una determinada movilidad, si y solo si sistema embebido envía datos de localización.

5.3. Estado de objetivo general

Se ha implementado y programado un sistema embebido basado en Arduino bajo el concepto del Internet de las cosas que mide el grado alcohólico de un conductor el cual envía notificaciones a los familiares/amigos si el grado alcohólico es mayor al establecido.

5.4. Recomendaciones

- Buscar metodologías de seguridad ya que al exponer datos muy importantes, se pone en riesgo la privacidad.
- Para hacer pruebas en una movilidad se recomienda usar un LM7805 que es un regulador de voltaje dado que las movilidades trabajan con 12 voltios y el Arduino con 5 voltios.
- A medida que pasa el tiempo la tecnología va avanzando de manera significativa, por ello se recomienda actualizar el hardware tanto como el software y/o adaptarla.
- A medida que la cantidad de dispositivos de hardware a ser conectados crece, es necesario que las características de hardware del servidor en que se ejecuta el servicio del sistema embebido sean aumentadas en cuanto a memoria RAM, almacenamiento, procesamiento y otros.

BIBLIOGRAFÍA

Alfaro, N. (2015). *Control de dispensadores de odorizacion basado en Arduino (Tesis de Pregrado).* La Paz, Bolivia: Universidad Mayor de San Andres.

Anderson & Cervo. (2013). *Pro Arduino.* Technology in Action.

ANF. (13 de junio de 2016). *El Dia*. Obtenido de https://www.eldia.com.bo/index.php?cat=386&pla=3&id_articulo=200865

Arduino . (2013). Obtenido de Arduino Ethernet Shield: http://arduino.cc/en/Main/ArduinoEthernetShield

Arduino. (s.f.). *Arduino Concept*. Recuperado el Octubre de 2016, de http://arduino.cc

Asay. (27 de Junio de 2014). *ReadWrite*. Obtenido de The Internet Of Things Will Need Millions Of Developers By 2020: http://readwrite.com/2014/06/27/internet-of-things-developers-jobs-opportunity/

Barchini, Sosa & Herrera. (2014). *La informática como disciplina científica.* Argentina: Universidad Nacional de Santiago del Estero.

Beck, K.; Beedle, M.; Bennekum, A.; Cockburn, A.; Cunningham, W.; Fowler, M.; Grenning, J.; Highsmith, J.; Hunt, A.; Jeffries, R.; Kern, J.; Marick, B.; Martin, R.; Mellor, S.; Schwaber, K.; Sutherland, J. & Thomas, D. (2001). *Principios del Manifiesto Ágil*. Recuperado el Octubre de 2016, de http://agilemanifesto.org/iso/es/principles.html

Carrero, A. (19 de Noviembre de 2016). *IncubaWeb*. Obtenido de El internet de las cosas y sus mayores riesgos: https://www.incubaweb.com/internet-de-las-cosas-mayores-riesgos/

Chacho, Sotomayor & Delgado. (2013). *Diseño e implementación de un sistema automático de Alumbrado Led Publico Inteligente controlado vía Wireless e instalado en una Casa de Don Bosco de Guayaquil (Tesis de pregrado).* Guayaquil, Ecuador: Universidad Politecnica Salesiana.

Congress, L. o. (Junio de 2011). Obtenido de http://www.loc.gov/rr/scitech/mysteries/global.html

daCosta, F. (2013). *Rethinking The Internet Of Things, A Scalable Approach to Connecting Everything.* Appress Open .

Dacosta, F. (2013). *Rethinking The Internet Of Things, A Scalable Approach to Connecting Everything.* Appress Open.

De Marcos, R. (2013). *Sistema domótico para una Casa Inteligente (Tesis de pregrado).* Madrid, España: Universidad Pontificia Comillas ICAI-ICADE.

De Seta, L. (13 de Noviembre de 2008). *DosIdeas*. Obtenido de Introducción a los servicios web RESTful: http://www.dosideas.com/noticias/java/314-introduccion-a-los-servicios-web-restful

Entel. (s.f.). *Vivir mejor conectado*. Recuperado el 25 de Octubre de 2016, de GPRS: http://personas.entel.cl/PortalPersonas/appmanager/entelpcs/personas?_nfpb=true&_pageLabel=P11800567291273156038130

Fielding, R. (2000). , *Architectural Styles and Design of Networkbased Software Architectures (Capítulo 5 – Representational State Transfer REST).* Recuperado el Octubre de 2016, de http://www.ics.uci.edu/~fielding/pubs/dissertation/rest_arch_style.htm

Gartner. (12 de diciembre de 2013). *Gartner Says the Internet of Things Installed Base Will Grow to 26 Billion Units By 2020*. Obtenido de http://www.gartner.com/newsroom/id/2636073

GeekFactory. (s.f.). *Geek Factory*. Recuperado el 25 de Octubre de 2016, de Shield GPS for Arduino: http://www.geekfactory.mx/tienda/shields-arduino/shield-gps/

Guillen, S. (2015). *Beehive, una plataforma Open-Source para el Internet de las Cosas (Tesis Pregrado).* La Paz, Bolivia: Universidad Mayor de San Andres.

Herrador. (2009). *Guia de Usuario de Arduino.* Cordoba: Universidad de Cordoba.

HiveMQ. (2015). *HiveMQ*. Recuperado el Octubre de 2016, de MQTT Essentials Part2, Publish & Subscribe: http://www.hivemq.com/mqtt-essentials-part2-publish-subscribe/

Höller, Tsiatsis, Mulligan, Karnouskos, Avesand & Boyle. (2014). *From Machine-to-Machine to the Internet of Things: Introduction to a New Age of Intelligence.* USA: Elsevier.

HTTP. (Junio de 1999). *RFC26126*. Recuperado el Octubre de 2016, de Hypertext Transfer Protocol -- HTTP/1.1: https://tools.ietf.org/html/rfc2616#section-9

Isaacs. (3 de septiembre de 2014). *3 Ways The Internet Of Things Is Revolutionizing Health Care*. Obtenido de 3 Ways The Internet Of Things Is Revolutionizing Health Care

Itead. (s.f.). *Itead*. Obtenido de SIM900/SIM900A GSM/GPRS Minimum System Module: https://www.itead.cc/sim900-sim900a-gsm-gprs-minimum-system-module.html

Korolov. (30 de Marzo de 2015). *Network World*. Obtenido de Will open source save the Internet of Things?: http://www.networkworld.com/article/2902231/internet-of-things/will-open-source-save-the-internet-of-things.html

Lara, E. (13 de Octubre de 2015). *HETPRO*. Obtenido de SIM900 GSM GPRS SHIELD CON ARDUINO UNO: http://hetpro-store.com/TUTORIALES/sim900-gsm-shieldarduino/

Library of Congress. (June de 2011). *What is GPS? Everyday Mysteries.* Obtenido de Library of Congress: http://www.loc.gov/rr/scitech/mysteries/global.html

Lifesafer. (2016). *Lifesafer*. (LifeSafer) Obtenido de https://www.lifesafer.com/devices/what-is-an-interlock/

Lopez, J. (s.f.). *Tuataratech*. Obtenido de Sensores (Sensors) vs Actuadores (Actuators): http://www.tuataratech.com/2015/06/sensores-sensors-vs-actuadores-actuators_8.html

Machaca, B. (2015). *Control Inteligente de seguridad en viviendas mediante Agentes Móviles (Tesis de pregrado).* La Paz, Bolivia: Univerdidad Mayor de San Andre.

McEwen & Cassimally. (2014). *Designing The Internet Of Things.* John Wiley and Sons, Ltd.

Mcewen & Cassimally. (2014). *Internet de las Cosas, la teconologia revolucionaria que todo lo conecta.* Madrid: Anaya Multimedia.

Microsoft. (s.f.). *Model-View-Controller*. Obtenido de MSDN: http://msdn.microsoft.com/en-us/library/ff649643.aspx

Mohammed, J. (7 de Diciembre de 2014). *VB*. Obtenido de Surprise: Agriculture is doing more with IoT Innovation than most other industries: http://venturebeat.com/2014/12/07/surprise-agriculture-is-doing-more-with-iot-innovation-than-most-other-industries/

Moorhead. (26 de septiembre de 2013). *Forbes*. Obtenido de The Problem With Home Automation's Internet Of Things (IoT) : http://www.forbes.com/sites/patrickmoorhead/2013/09/26/the-problem-with-home-automations-iot/

Moreno, J. (24 de julio de 2015). *BBC*. Obtenido de Los países que más beben en América Latina: la dramática radiografía del consumo de alcohol en la región: http://www.bbc.com/mundo/noticias/2015/07/150723_consumo_alcohol_latinoamerica_muertes_paises_jm

MQTT. (2004). *MQTT*. Recuperado el Octubre de 2016, de Protocol Definition: http://mqtt.org/

NaylampMechatronics. (2015). *Sensores de gas MQ2, MQ3, MQ7 y MQ135*. Obtenido de http://www.naylampmechatronics.com/blog/42_Tutorial-sensores-de-gas-MQ2-MQ3-MQ7-y-MQ13.html

Olmo, L. (7 de Julio de 2016). *TICbeat*. Obtenido de Ya hay 6,2 millones de desarrolladores de Internet de las Cosas: http://www.ticbeat.com/tecnologias/ya-hay-62-millones-de-desarrolladores-de-internet-de-las-cosas/

Palma, N. (s.f.). *La Conversacion* . Obtenido de El Internet de las Cosas: hay más cosas conectadas a la red que gente : http://laconversacion.net/2016/08/29/el-internet-de-las-cosas-hay-mas-cosas-conectadas-a-la-red-que-gente/

Pavon, Duque & Fuentes. (2012). *Advances in Artificial Intelligence - IBERAMIA 2012*. Springer.

Rivas, M. (19 de agosto de 2015). *La Razon*. Obtenido de Consumo de alcohol influye en 40% de accidentes de tránsito: http://www.la-razon.com/ciudades/El_Alto-consumo-alcohol-influye-accidentes-transito_0_2328967115.html

Sampieri, Fernández & Baptista. (2010). *Metodología de la investigación* (5 ed.). Mexico, D. F.: McGraw-Hill Interamericana.

Schuermans. (16 de Mayo de 2016). *Vision Mobile*. Obtenido de 91% of IoT developers use Open source: https://www.visionmobile.com/blog/2016/05/open-source-in-iot

Schuermans, S. (15 de Octubre de 2014). *Vision Mobile*. Obtenido de The Kingmakers of the Internet Of Things: http://www.visionmobile.com/blog/2014/10/kingmakers-internet-things

Schwaber, K. & Sutherland, J. (Julio de 2016). *The Scrum Guide.* Obtenido de http://scrumguides.org/docs/scrumguide/v1/Scrum-Guide-US.pdf

ScrumAlliance. (s.f.). *ScrumAlliance*. Obtenido de Learn About Scrum: https://www.scrumalliance.org/why-scrum

SemanticWebBuilder. (2015). *Sistemas Embebidos: Innovando hacia los Sistemas Inteligentes*. Obtenido de http://www.semanticwebbuilder.org.mx/es_mx/swb/Sistemas_Embebidos_Innovando_hacia_los_Sistemas_Inteligentes_

Tharrington, M. (5 de Noviembre de 2015). *DZone*. Obtenido de The Future of Smart Farming With IoT and Open Source Farming: https://dzone.com/articles/the-future-of-smart-farming-with-iot-and-open-sour

Torrez. (20 de octubre de 2014). *Hipertextual*. Obtenido de ¿Qué es y cómo funciona el Internet de las cosas?: https://hipertextual.com/archivo/2014/10/internet-cosas/

Valdés, K. (30 de abril de 2016). *La Razon*. Obtenido de La Paz tiene la mayor tasa de accidentes de tránsito en 2016: http://la-razon.com/ciudades/Datos-La_Paz-mayor-tasa-accidentes-transito_0_2481951820.html

Villa, M. (14 de julio de 2014). *La Razon*. Obtenido de Imprudentes y ebrios causan 8 de cada 10 hechos viales: http://www.la-razon.com/index.php?_url=/ciudades/seguridad_ciudadana/Imprudentes-ebrios-causan-hechos-viales_0_2089591024.html

Wikipedia. (9 de septiembre de 2016). *Sistema Embebido*. Obtenido de https://es.wikipedia.org/wiki/Sistema_embebido

Wikipedia. (16 de agosto de 2016). *Wikipedia*. Obtenido de Internet de las Cosas: https://es.wikipedia.org/wiki/Internet_de_las_cosas

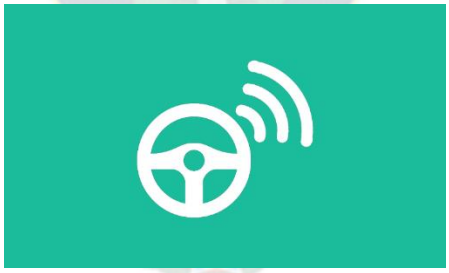
Zhang, L. (12 de Agosto de 2011). *Building Facebook Messenger*. Recuperado el Octubre de 2016, de Facebook: https://www.facebook.com/notes/facebook-engineering/building-facebook-messenger/10150259350998920

Zkoss. (s.f.). *Zkoss*. Obtenido de Category:ZK Developer's Reference MVC: https://www.zkoss.org/wiki/ZK_Developer's_Reference/MVC

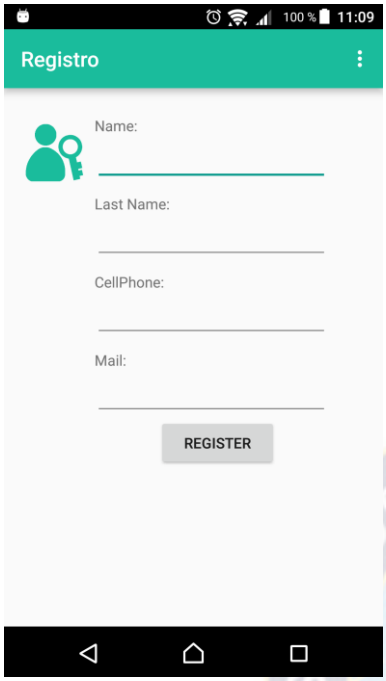
ANEXOS

Manual de usuario

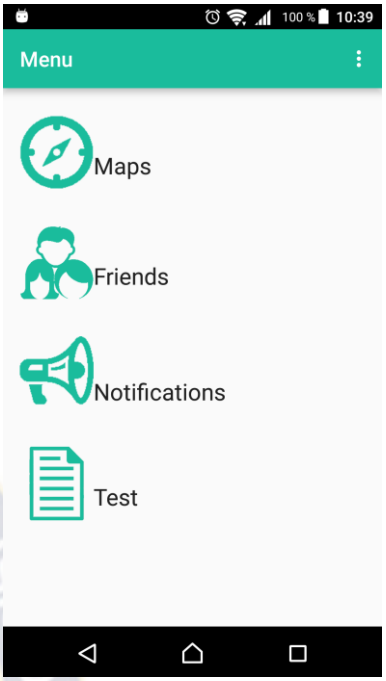
SISTEMA EMBEBIDO PARA LA PREVENCIÓN DE ACCIDENTES DE TRÁNSITO OCASIONADOS POR CONDUCTORES EN ESTADO DE EBRIEDAD APLICANDO EL INTERNET DE LAS COSAS



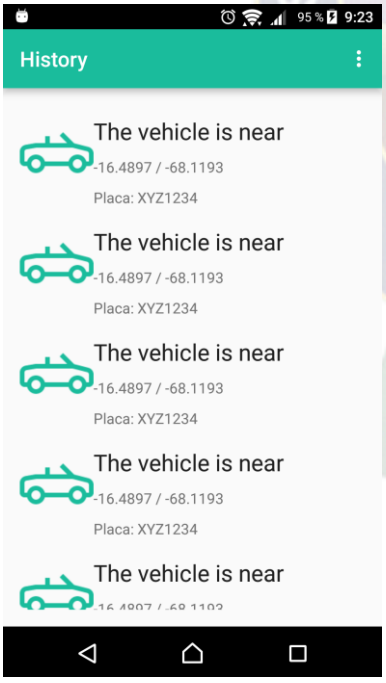
El Sistema Embebido es un dispositivo que está en un vehículo, el cual mide el grado alcohólico del conductor al momento de encender el vehículo, si este sistema verifica que el conductor esta ebrio procede a enviar la placa del vehículo y la posición del vehículo a los familiares o amigos de confianza del propietario de la movilidad, por otro lado si el motorizado está en marcha, el sistema solo enviara notificaciones con la posición y la placa cada cierto tiempo.



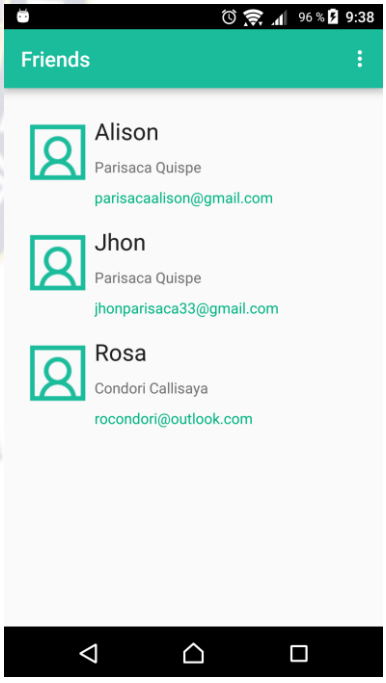
Registro de usuarios



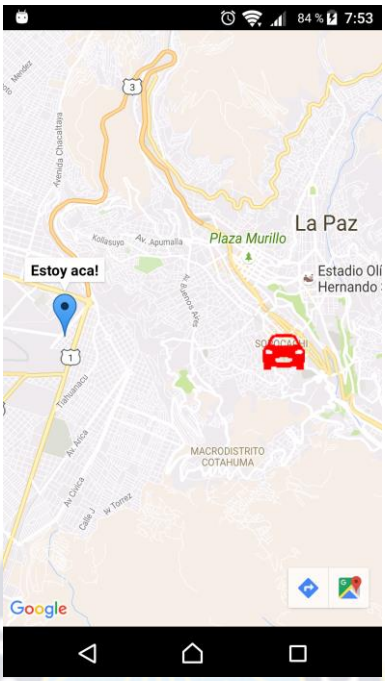
Menú principal de la App



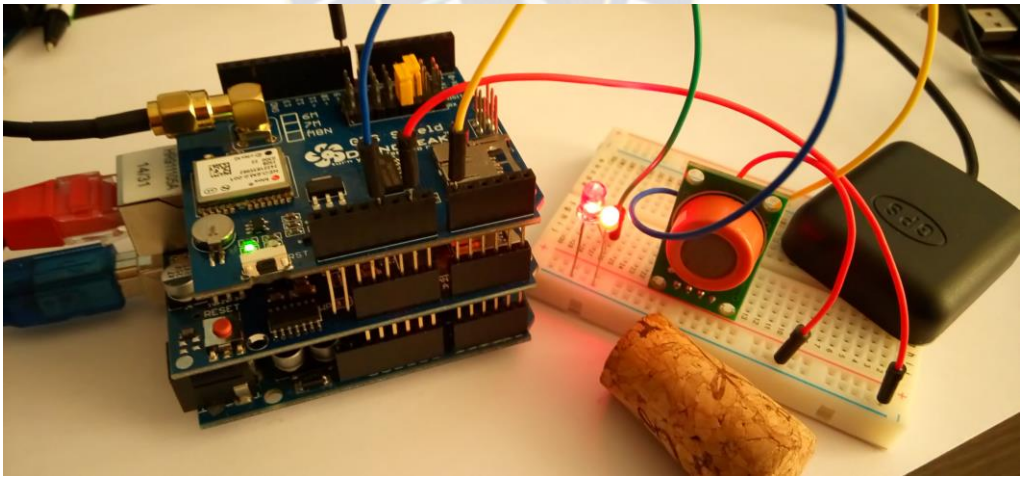
Historial de las ultimas notificaciones



Posibilidad de tener Familia/amigos de confianza



Cuando se tiene la posición de un amigo o familiar ebrio, podemos observar la distancia y lugar en la cual se encuentra, el auto rojo es la posición del amigo o familiar ebrio y el punto azul es la posición en la que te encuentras.



Dispositivo consta de Sensor Etílico, Modulo Sim900, Modulo GPS. Tiene la capacidad de conectarse a internet desde lugares que carecen de velocidad, debido a que esta implementada con el protocolo MQTT.

DOCUMENTACION