



ESPE
UNIVERSIDAD DE LAS FUERZAS ARMADAS
INNOVACIÓN PARA LA EXCELENCIA

DEPARTAMENTO DE CIENCIAS DE LA COMPUTACIÓN

CARRERA DE INGENIERÍA EN SISTEMAS E INFORMÁTICA

PROYECTO DE TITULACIÓN PREVIO A LA OBTENCIÓN DEL TÍTULO
DE INGENIERO EN SISTEMAS E INFORMÁTICA

AUTOR: DIEGO MAURICIO CARRERA ARÍZAGA

TEMA: DESARROLLO DE UN PROTOTIPO DE APLICACIÓN WEB EN
COMBINACIÓN CON LA PLATAFORMA ARDUINO PARA CONTROLAR LA
CALIDAD DE AIRE DE LA CIUDAD DE QUITO.

DIRECTOR: ING. WALTER FUERTES PhD

CODIRECTOR: ING. CÉSAR VILLACÍS MSc

SANGOLQUÍ, ENERO 2015

CERTIFICADO

Ing. Walter Fuertes PhD (DIRECTOR DE TESIS)

Ing. César Villacís (CODIRECTOR DE TESIS)

CERTIFICAN

Que el trabajo titulado “DESARROLLO DE UN PROTOTIPO DE APLICACIÓN WEB EN COMBINACIÓN CON LA PLATAFORMA ARDUINO PARA CONTROLAR LA CALIDAD DE AIRE DE LA CIUDAD DE QUITO”, realizado el Sr: Diego Mauricio Carrera Arízaga, ha sido guiada y revisada periódicamente y cumple las normas estatutarias establecidas por la ESPE, en el Reglamento Estudiantes de la Universidad de las Fuerzas Armadas-ESPE.

Debido a que constituye un trabajo con excelente contenido científico, que coadyuvará a la aplicación de conocimientos y desarrollo profesional, se recomienda su publicación.

El mencionado trabajo consta de un documento empastado y un disco compacto el cuál contiene los archivos en formato digital de Acrobat (pdf). Se autoriza al Sr. Diego Mauricio Carrera Arízaga, que el material se entregue al Ing. Mauricio Campaña, en su calidad de Director de Carrera.

Sangolquí, Enero de 2015

Ing. Walter Fuertes PhD
DIRECTOR DE TESIS

Ing. César Villacís
CODIRECTOR DE TESIS

DECLARACIÓN DE RESPONSABILIDAD

Diego Mauricio Carrera Arízaga

DECLARA QUE:

El proyecto de grado denominado: “DESARROLLO DE UN PROTOTIPO DE APLICACIÓN WEB EN COMBINACIÓN CON LA PLATAFORMA ARDUINO PARA CONTROLAR LA CALIDAD DE AIRE DE LA CIUDAD DE QUITO”, ha sido desarrollada en base a una investigación exhaustiva, respetando derechos de autor, conforme a las citas que constan a lo largo del documento de Tesis y que se incorporan en el apartado de Bibliografía.

Consecuentemente este trabajo es de mi autoría.

En virtud de esta declaración, me responsabilizo del contenido, veracidad y alcance científico del proyecto de grado en mención.

Sangolquí, Enero de 2015.

Diego Mauricio Carrera Arízaga

CC: 171319227-4

AUTORIZACIÓN

Autorizo a la UNIVERSIDAD DE LAS FUERZAS ARMADAS – ESPE la publicación, en la biblioteca virtual de la Institución, del trabajo “DESARROLLO DE UN PROTOTIPO DE APLICACIÓN WEB EN COMBINACIÓN CON LA PLATAFORMA ARDUINO PARA CONTROLAR LA CALIDAD DE AIRE DE LA CIUDAD DE QUITO”, cuyo contenido, ideas y criterios son de mi exclusiva responsabilidad y autoría

Sangolquí, Enero de 2015.

Diego Mauricio Carrera Arízaga

CC: 171319227-4

DEDICATORIA

El presente proyecto es el resultado de largas horas de investigación y esfuerzo .

A mi madre, quien desde el cielo me sigue enseñando lo invaluable de ser justo y valiente en la vida. Mami, te extraño mucho.

A mi padre quien con su apoyo y ejemplo nos supo inculcar, a mis hermanos y a mi, responsabilidad, compromiso y el amor por lo que cada uno decidimos hacer en la vida profesional.

A mis hermanos y sobrinos, de quienes siempre he recibido cariño y aliento en cada etapa de mi vida.

A Daniela por ser el impulso final y el pilar principal en la elaboración de este trabajo.

Diego Mauricio Carrera Arízaga

AGRADECIMIENTO

v

En primer lugar quiero agradecer a Dios, ya que sin su infinita bondad nada de esto hubiera sido posible.

A mi familia y amigos, por su paciencia y apoyo en cada momento de mi vida.

A Daniela por enseñarme a ver el mundo desde una perspectiva diferente.

A mi Director del Proyecto de Titulación, Ing. Walter Fuertes PhD, a mi Codirector, Ing. César Villacís, al Dr. Theofilos Toulkeridis y al Ing. Marco Jarrín por compartir conmigo sus conocimientos y conocimientos.

No quisiera caer en la falta de omitir a alguien, por lo que quiero expresar mi más sincero agradecimiento a todas las personas que me apoyaron de una u otra manera. Se los agradezco de corazón.

Diego Mauricio Carrera Arízaga

ÍNDICE DE CONTENIDOS

vi

CERTIFICADO	i
DECLARACIÓN DE RESPONSABILIDAD	ii
AUTORIZACIÓN	iii
DEDICATORIA	iv
AGRADECIMIENTO	v
ÍNDICE DE CONTENIDOS	vi
ÍNDICE DE FIGURAS	ix
ÍNDICE DE CUADROS	xi
RESUMEN	xiii
ABSTRACT	xiv
CAPÍTULO 1	1
INTRODUCCIÓN	1
1.1 Tema.....	1
1.2 Definición del problema	1
1.3 Contextualización del problema.....	1
1.4 Formulación del problema	2
1.5 Justificación del proyecto.....	3
1.6 Objetivos	3
1.6.1 Objetivo General	3
1.6.2 Objetivos Específicos	4
1.7 Alcance del proyecto.....	4
CAPÍTULO 2	6
MARCO TEÓRICO	6
2.1 Metodologías de desarrollo aplicables.....	7
2.1.1 Scrum	7
2.1.2 Extreme Programming (XP)	12
2.2 Scrum en combinación con Extreme Programming (XP)	17

2.2.1	Actividades, artefactos y herramientas tomados de Scrum	vii 18
2.2.2	Practicas tomadas de Extreme Programming (XP).....	18
2.3	Plataforma Arduino	19
2.3.1	Hardware.....	19
2.3.2	Entorno de Programación	20
2.3.3	Módulos de Expansión	21
2.3.4	Sensores	22
2.4	Plataforma de desarrollo.....	22
2.4.1	Plataforma Java Enterprise Edition (JEE).....	23
2.4.2	Modelo Multicapa	23
2.4.3	Servidor Java EE.....	26
2.5	Herramientas de Desarrollo	27
2.5.1	Eclipse.....	27
2.5.2	Servidor WildFly	28
2.5.3	MySQL	28
2.5.4	JQuery y Bootstrap.....	28
CAPÍTULO 3	29
ESPECIFICACIÓN DE REQUERIMIENTOS	29
3.1	Introducción	29
3.2	Propósito.....	29
3.3	Alcance	29
3.4	Limitaciones del Prototipo.....	31
3.5	Personal Involucrado	32
3.6	Definiciones	32
3.7	Acrónimos	33
3.8	Referencias.....	34
3.9	Visión General del Documento	34
3.10	Descripción General	35
3.10.1	Perspectiva del Producto	35
3.10.2	Funciones del Producto.....	35
3.10.3	Características del los Usuarios.....	37
3.10.4	Restricciones.....	38

3.10.5	Suposiciones y Dependencias	viii 38
3.11	Requisitos Específicos	39
3.11.1	Interfaces Externas	41
3.11.2	Funciones	42
3.11.3	Requisitos No Funcionales	44
CAPÍTULO 4	47
DISEÑO DEL PROTOTIPO	47
4.1	Modelo de Datos	47
4.2	Diseño de la Base de Datos	48
4.3	Diagramas de Casos de Uso	49
4.4	Diagramas de Secuencia	54
4.5	Diagramas de Clases	62
4.6	Diagrama de Paquetes	67
4.7	Diagrama de Arquitectura	67
CAPÍTULO 5	69
PLANIFICACIÓN, DESARROLLO Y PRUEBAS DEL PROTOTIPO	69
5.1	Planificación del proyecto utilizando la metodología Scrum	69
5.2	Estándares de programación	71
5.3	Desarrollo y pruebas de la Iteración 1	76
5.4	Sprint Backlog de la iteración 1	77
5.4.4	Revisión y seguimiento de la iteración 1	78
5.4.5	Pruebas de la iteración 1	78
5.4.6	Demo Iteración 1	78
5.5	Desarrollo y pruebas de la Iteración 2	83
5.5.1	Sprint Backlog iteración 2	83
5.5.2	Revisión y seguimiento de la iteración 2	84
5.5.3	Pruebas de la iteración 2	84
5.5.4	Demo iteración 2	85
5.6	Desarrollo y pruebas de la Iteración 3	90
5.6.1	Sprint Backlog iteración 3	91
5.6.2	Revisión y seguimiento iteración 3	91
5.6.3	Pruebas de la iteración 3	91

5.6.4 Demo iteración 3	ix
5.7 Desarrollo y evaluación de la Iteración 4	91
5.7.1 Sprint Backlog de la iteración 4	92
5.7.2 Revisión y seguimiento de la iteración 4	92
5.7.3 Pruebas de la iteración 4	95
5.7.4 Demo de la iteración 4	95
CAPÍTULO 6	100
ANÁLISIS E INTERPRETACIÓN DE RESULTADOS	100
6.1 Introducción	100
6.2 Análisis de parámetros de la Ciudad de Quito	102
6.3 Análisis de parámetros en cuevas de investigación, el Tena	103
CAPÍTULO 7	108
CONCLUSIONES Y RECOMENDACIONES	108
7.1 Conclusiones	108
7.2 Recomendaciones	110
7.3 Trabajo Futuro	110
BIBLIOGRAFÍA	¡Error! Marcador no definido.
ANEXOS	¡Error! Marcador no definido.
ANEXO A	¡Error! Marcador no definido.
MANUAL DE USUARIO	¡Error! Marcador no definido.
ANEXO B	¡Error! Marcador no definido.
EVIDENCIA FOTOGRÁFICA	¡Error! Marcador no definido.

ÍNDICE DE FIGURAS

Figura 1 Ciclo de Aplicación Scrum Fuente: (Álvarez García et al, 2012)	8
Figura 2 Diagrama Burndow Chart (Scrum-Alliance, 2014)	12
Figura 3 Proceso Scrum (Pressman, 2010)	13
Figura 4 Esquema de la placa Arduino Uno. Fuente: (Arduino-Arts, 2014) ..	20
Figura 5 Modelo Multicapa de Java EE. Fuente: (Oracle, 2014)	24

Figura 6 Funciones generales.....	x
Figura 7 Diseño de la Base de Datos.	36
Figura 8 Casos de uso del proyecto.	48
Figura 9 Casos de uso del portal de la aplicación Web.	49
Figura 10 Caso de uso: Registrar Usuario.....	50
Figura 11 Caso de uso: Iniciar Sesión.	50
Figura 12 Caso de uso: Registrar Nuevo Dispositivo.	51
Figura 13 Casos de uso de modificación de información.	52
Figura 14 Caso de uso: Eliminar Dispositivo.	53
Figura 15 Caso de uso: Generar Gráfico.	54
Figura 16 Casos de uso sobre captura y transmisión de datos.	54
Figura 17 Escenario de visita y consulta de información.	55
Figura 18 Escenario de registro de Usuario Avanzado.....	56
Figura 19 Escenario de inicio de sesión	56
Figura 20 Escenario de registro de un nuevo dispositivo electrónico.	57
Figura 21 Escenario de actualización de información.....	59
Figura 22 Escenario de eliminación de dispositivos electrónicos.	60
Figura 23 Generación de gráficos.....	60
Figura 24 Secuencia para capturar, ordenar y transmitir información.	61
Figura 25 Diagrama de clases entidad.	63
Figura 26 Diseño de clases DAO.....	64
Figura 27 Diseño de clases de servicios.....	65
Figura 28 Diagrama de clases de control	66
Figura 29 Diagrama de clases del Servicio Web.	66
Figura 30 Diagrama de paquetes.....	67
Figura 31 Arquitectura de software.	68
Figura 32 Burn-Down Chart de la Iteración 1.....	81
Figura 33 Demo Iteración 1.....	82
Figura 34 Pruebas unitarias de la iteración 1.....	83
Figura 35 Burn-Down Chart de la iteración 2.....	85
Figura 36 Demo de la iteración 2.	88
Figura 37 Pruebas unitarias de la iteración 2.....	89
Figura 38 Prueba unitaria de la iteración 3	92

Figura 39 Burn-Down Chart de la iteración 3	xi
Figura 40 Demo de la iteración 3	94
Figura 41 Burn-Down Chart de la iteración 4	95
Figura 42 Demo de la iteración 4	98
Figura 43 Concentración de Monóxido de Carbono Quito	99
Figura 44 Densidad de Polvo Quito	102
Figura 45 Densidad de Polvo – Cueva Castillo.....	103
Figura 46 Concentración de Dióxido de Carbono – Cueva Castillo	104
Figura 47 Densidad de Polvo – Cueva Gruta de la Virgen	105
Figura 48 Concentración de Dióxido de Carbono – Cueva Gruta de la Virgen	105
.....	106

ÍNDICE DE CUADROS

Cuadro 1 Actividades de cada Sprint.....	9
Cuadro 2 Roles del Scrum Team.....	10
Cuadro 3 Artefactos de Scrum.....	11
Cuadro 4 Actividades de XP	14
Cuadro 5 Roles de XP	15
Cuadro 6 Prácticas de XP	16
Cuadro 7 Capas de la aplicación	25
Cuadro 8 Procesos del prototipo.....	30
Cuadro 9 Personal involucrado.....	32
Cuadro 10 Funciones generales del prototipo	36
Cuadro 11 Características de los usuarios	37
Cuadro 12 Descripción de las restricciones del proyecto	39
Cuadro 13 Requisitos Específicos	40
Cuadro 14 Funciones.....	43
Cuadro 15 Pila de producto o Product Backlog	72
Cuadro 16 Convenciones de código	75
Cuadro 17 Etiquetas XHTML	76
Cuadro 18 Historias de usuario seleccionadas para la iteración 1	77
Cuadro 19 Sprint Backlog de la iteración 1	79

	xii
Cuadro 20 Tareas completadas de la iteración 1	80
Cuadro 21 Historias de usuario seleccionadas para la iteración 2	84
Cuadro 22 Sprint Backlog de la iteración 2	86
Cuadro 23 Tareas completadas de la iteración 2	87
Cuadro 24 Historias de usuario seleccionadas para la iteración 3	90
Cuadro 25 Sprint Backlog de la iteración 3	93
Cuadro 26 Tareas completadas de la iteración 3	93
Cuadro 27 Historias de usuario escogidas para la iteración 4	96
Cuadro 28 Sprint Backlog de la iteración 4	97
Cuadro 29 Tareas completadas de la iteración 4	97
Cuadro 30 Rangos, límites permisibles y valores referenciales	101

RESUMEN

TEMA: DESARROLLO DE UN PROTOTIPO DE APLICACIÓN WEB EN COMBINACIÓN CON LA PLATAFORMA ARDUINO PARA CONTROLAR LA CALIDAD DE AIRE DE LA CIUDAD DE QUITO.

La contaminación atmosférica y la falta de puntos de monitoreo de calidad del aire es uno de los principales retos ambientales y tecnológicos de las ciudades del mundo y del Ecuador. Actualmente, solo Quito, Guayaquil y Cuenca monitorean la calidad del aire de sus urbes mientras otras ciudades apenas están en proceso de desarrollar programas de medición. Este proyecto de titulación presenta el proceso de conceptualización, análisis y desarrollo, de un prototipo tecnológico asequible, compuesto por una aplicación Web, que en combinación con la plataforma Arduino permite medir parámetros referenciales de calidad de aire en la ciudad de Quito. Para la elaboración se utilizó una combinación de metodologías de desarrollo de software como Scrum y Extreme Programming, las mismas que han ordenado y guiado el trabajo no solo de la construcción del sistema de información, si no también el proceso de elaboración del hardware y firmware necesarios para procesar y transmitir las lecturas desde los sensores electrónicos, instalados en la tarjeta Arduino, hacia la aplicación Web, utilizando una serie de elementos de hardware y software que van desde un módulo WiFi, hasta software como la API JAX-RS, la cuál se utiliza para elaborar un servicio Web de tipo RESTful, el mismo que permite la interoperabilidad y almacenamiento de la información recolectada. Al desarrollo de esta solución tecnológica se le efectuó una prueba de concepto la que consistió en un análisis estadístico de los datos medidos, los mismos que fueron comparados con estándares internacionales de calidad del aire.

PALABRAS CLAVE: SCRUM, XP, JAVA, JEE, SERVICIOS WEB, RESTFuL, APLICACIÓN WEB, ARDUINO, SENSORES, CALIDAD DE AIRE.

ABSTRACT

SUBJECT: DEVELOPMENT OF A WEB APPLICATION PROTOTYPE IN COMBINATION WITH ARDUINO PLATAFORM TO CONTROL THE AIR QUALITY OF QUITO.

Air pollution and the lack of air quality monitoring points are one of the most important environmental challenges in cities. In Ecuador, for example, only Quito, Guayaquil and Cuenca monitor air quality of their cities while other are just in the process of developing measurement programs. This project presents the conceptualization, analysis and development, of an affordable technology prototype, consisting of a Web application, which in combination with the Arduino platform, measures air quality parameters in the city of Quito. In order to build the prototype, a combination of software development methodologies such as Scrum and Extreme Programming were used. These have ordered and guided the work to build the information system and to develop the necessary hardware and firmware to process and transmit readings from electronic sensors installed on the Arduino board to the web application. This technology uses a variety of hardware and software, ranging from a WiFi module up to a JAX -RS API software, which is used to develop a RESTful web service, allowing interoperability and storage of the collected information. The development of this technology solution includes a proof of concept, which consists of a statistical analysis of the measured data that later is compared with international standards of air quality to assess the effectiveness of the measurements.

KEY WORDS: SCRUM, XP, JAVA, JEE, WEB SERVICES, RESTFUL, WEB APPLICATION, ARDUINO, SENSORS, AIR QUALITY.

CAPÍTULO 1

INTRODUCCIÓN

1.1 Tema

Desarrollo de un prototipo de aplicación Web en combinación con la plataforma Arduino para controlar la calidad del aire de la ciudad de Quito.

1.2 Definición del problema

La contaminación atmosférica y la falta de puntos de monitoreo de calidad del aire es uno de los principales retos ambientales y tecnológicos de las ciudades del mundo y del Ecuador. Actualmente, solo Quito, Guayaquil y Cuenca monitorean la calidad del aire de sus urbes mientras otras ciudades apenas están en proceso de desarrollar programas de medición (MAE, 2014).

Quito posee una Red de Monitoreo Atmosférica convencional que mide calidad del aire únicamente en 9 puntos de la ciudad. La información captada por este sistema se procesa y publica a través del portal Web de la Secretaría de Ambiente del Municipio de Quito, pero no es lo suficientemente amplia ni representativa de la calidad del aire de la ciudad. (Secretaría de Ambiente Quito, 2014). Por esto, es importante encontrar una alternativa tecnológica versátil que permita mejorar el proceso de medición de la calidad del aire en la ciudad de Quito y brindar valores referenciales en sitios que las redes de monitoreo convencionales no llegan a cubrir.

1.3 Contextualización del problema

Para los gobiernos nacionales y locales es sumamente importante medir la calidad del aire en las ciudades, sobre todo por los efectos que esta

contaminación puede causar en la salud. El 17 octubre del 2013 la Agencia Internacional para la Investigación sobre el Cáncer (IARC, según sus siglas en inglés), que forma parte de la Organización Mundial de la Salud (OMS), en el comunicado de prensa N° 221 declara a la contaminación del aire como agente cancerígeno (International Agency for Reseach on Cancer, 2013).

Para controlar la calidad del aire, la mayoría de las ciudades del mundo poseen redes de monitoreo capaces de determinar la concentración de estos contaminantes a través de estaciones fijas convencionales distribuidas a lo largo y ancho de su geografía. Estas estaciones poseen equipos de medición especializados que permiten determinar las concentraciones de los distintos contaminantes de manera casi exacta, sin embargo no cubren todo el espectro geográfico de las ciudades.

Por esto, las redes de monitoreo necesitan ampliar su ámbito de medición a través de soluciones tecnológicas asequibles y versátiles que permitan obtener información sobre la concentración de los contaminantes para una mejor toma de decisiones a nivel local y para que el ciudadano esté informado.

1.4 Formulación del problema

Según el Ministerio del Ambiente del Ecuador y su programa de Calidad del Aire, a nivel nacional, sólo Quito, Guayaquil y Cuenca poseen los instrumentos necesarios para el monitoreo de la calidad del aire (MAE, 2014).

La Secretaría de Ambiente del Distrito Metropolitano de la ciudad de Quito, es la encargada de gestionar las 9 estaciones de monitoreo que se encuentran distribuidos a lo largo y ancho de la ciudad desde hace 10 años. La información recolectada es presentada en la Web: <http://190.152.144.74>,

de manera informal y poco amigable para los ciudadanos, lo que impide que los habitantes de la ciudad estén debidamente informados sobre los riesgos a los que se encuentran expuestos y que el Municipio pueda gestionar la calidad del aire de forma efectiva.

El crecimiento de la ciudad hace necesario ampliar el ámbito de medición de las estaciones existentes para determinar nuevas zonas de la ciudad con niveles elevados de contaminación atmosférica. Es por esto que este proyecto propone una solución tecnológica versátil para la medición de parámetros de calidad del aire en Quito con sensores atmosféricos, la transmisión de los resultados medidos a través del Internet y su presentación a través de una página Web amigable.

1.5 Justificación del proyecto

Una aplicación Web capaz de interactuar con sensores atmosféricos conectados a Internet ayudaría a:

- Ampliar el ámbito de medición de las redes de monitoreo convencionales de la calidad del aire.
- Obtener fácil acceso a la información sobre la calidad del aire en Quito.
- Mantener un inventario de las concentraciones de contaminantes atmosféricos para su análisis.

1.6 Objetivos

1.6.1 Objetivo General

Desarrollar un prototipo de aplicación Web en combinación con la plataforma Arduino para controlar la calidad del aire de la ciudad de Quito.

1.6.2 Objetivos Específicos

- Revisar el marco teórico referente a las metodologías de SCRUM y Extreme Programming (XP) para el desarrollo de sistemas de información.
- Describir la plataforma Arduino con el uso de sensores para la medición de la calidad del aire.
- Realizar la especificación de requerimientos del prototipo, aplicando la norma IEEE 830.
- Emplear las metodología Scrum y Extreme Programming para la planificación, diseño, desarrollo y pruebas del prototipo.
- Evaluar los resultados obtenidos.

1.7 Alcance del proyecto

Este proyecto consiste en el desarrollo de un prototipo funcional para medición de parámetros de calidad del aire en Quito, como ciudad de referencia.

Este prototipo se desarrollará con recursos propios y en colaboración con la Empresa Omnisoft y contará con 4 módulos, descritos a continuación.

- **Módulo de usuarios:** El prototipo incluirá un módulo de usuarios que brindará el acceso a la información generada por los dispositivos correspondientes a cada cuenta.
- **Módulo de dispositivos:** Este módulo permitirá registrar, configurar y gestionar los dispositivos y la información, que se

recolectará, de las diferentes magnitudes que se deseen medir. Presentará gráficamente la información en función del tiempo y la magnitud medida.

- **Servicio Web:** Se encargará de la interoperabilidad entre los dispositivos físicos y sus sensores, con la aplicación Web que procesará la información recolectada.

- **Portada de la aplicación:** Presentará, en un mapa, la información recolectada por los dispositivos y que esté configurada por el usuario del prototipo como de acceso público.

CAPÍTULO 2

MARCO TEÓRICO

Al momento de construir un sistema informático o un prototipo; es necesario emplear y gestionar un conjunto de elementos, por esta razón es indispensable el uso de métodos, técnicas y herramientas que permitan alcanzar los objetivos esperados en un tiempo determinado, utilizando de forma efectiva los recursos disponibles.

El uso de metodologías ágiles de desarrollo de software se ha popularizado en la última década debido a la evolución rápida y constante de la industria del software. Pressman afirma que: “La ingeniería de software ágil representa una alternativa razonable a la ingeniería de software convencional para ciertas clases de software y en algunos tipos de proyectos. Asimismo, se ha demostrado que concluye con rapidez sistemas exitosos” (Pressman, 2010).

Este enfoque de desarrollo ágil busca responder efectivamente al cambio constante de requerimientos y pone énfasis en la entrega rápida de software totalmente funcional que pueda ser utilizado por el cliente.

Para cumplir con los objetivos propuestos en este proyecto y debido a la naturaleza del mismo, es necesario emplear una combinación de metodologías para su desarrollo que se adapten a dos grandes tipos de necesidades.

La primera necesidad está asociada a la capacidad de adaptación a los cambios de requerimientos del software en todo el ciclo de vida del proyecto. La segunda necesidad se enfoca en facilitar períodos de desarrollo cortos que permitan entregas de prototipos, del software, totalmente funcionales para que puedan ser probados por el cliente.

Adicionalmente, es necesario definir una arquitectura de software que permita la escalabilidad del software, establecer un ecosistema de hardware apropiado según el alcance del proyecto y sobre todo definir las herramientas de desarrollo idóneas que se adapten a las necesidades del sistema o prototipo a desarrollarse.

A continuación se detalla las metodologías, arquitectura de software, ecosistema hardware y herramientas de desarrollo aplicables y utilizadas para la construcción de la solución tecnológica.

2.1 Metodologías de desarrollo aplicables

Se eligió dos metodologías de desarrollo de software combinadas para el elaboración del presente proyecto. La combinación entre Scrum, para la planificación del proyecto y Extreme Programming (XP) para la programación del software.

2.1.1 Scrum

Scrum se define como un marco referencial de trabajo que propone un conjunto de prácticas, principios y valores para organizar y gestionar las tareas en el desarrollo de productos o servicios. Takeduchi, Hirotaka & Nonaka, Ikujiro (1986) mencionan el término Scrum por primera vez en el documento *The New New Product Development Game*, donde se describe un proceso de desarrollo ágil, capaz de responder a la necesidad de ser más competitivos (Álvarez García, de las Heras del Dedo, & Lasa Gómez, 2012).

En 1993 un equipo liderado por Jeff Sutherland en Easel Corporation, aplica Scrum por primera vez con gran éxito. Fue Ken Schwaber, quien en 1996 formaliza el término Scrum en el artículo que lleva por título *Scrum*

Development Process presentado en la conferencia de OOPSLA (Álvarez García, de las Heras del Dedo, & Lasa Gómez, 2012).

Rogert Pressman (2010), en su libro “Ingeniería del Software, un enfoque práctico séptima edición”, sostiene que Scrum ha demostrado ser eficaz para proyectos con tiempo de entrega muy cortos, requerimientos cambiantes y negocios críticos

2.1.1.1 Proceso Scrum

Scrum define dos etapas en su proceso de organización del trabajo. La primera denominada *El Sprint 0* o etapa inicial; y la segunda de iteraciones sucesivas, también llamadas *Sprints* (Álvarez García, de las Heras del Dedo, & Lasa Gómez, 2012). En la Figura 1, se puede observar un esquema gráfico de este ciclo de aplicación de Scrum.

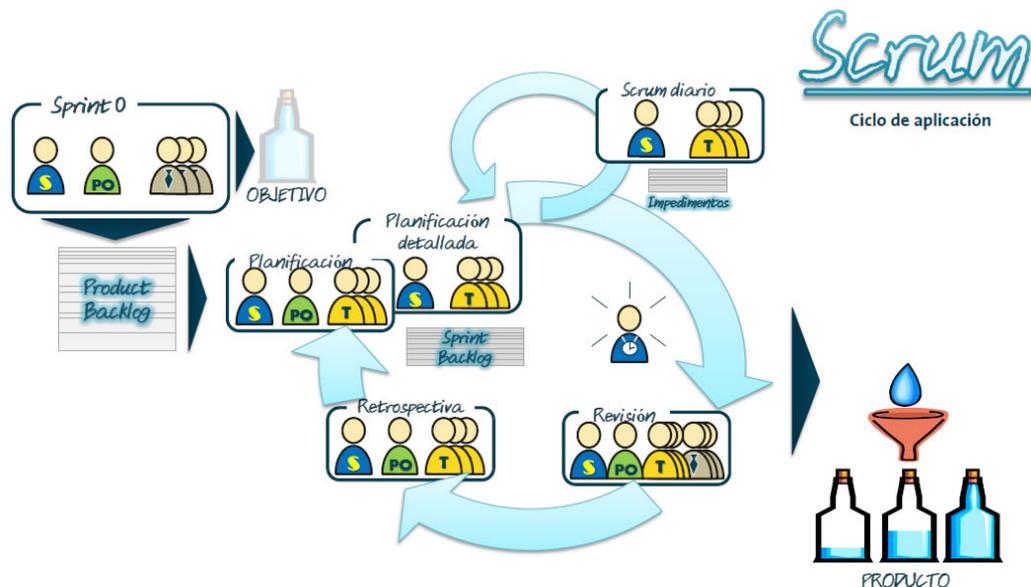


Figura 1 Ciclo de Aplicación Scrum

Fuente: (Álvarez García et al, 2012)

La etapa ***Sprint 0***, de duración variable pero no indefinida, determina la viabilidad del proyecto mediante el estudio de las condiciones y recursos necesarios. También se realiza la primera versión del listado de requerimientos generales del software, listado que se convertirá en el Product Backlog, mismo que es un insumo del proyecto y será refinado en la etapa de iteraciones sucesivas.

La segunda etapa de iteraciones sucesivas o Sprints, se divide en cuatro actividades las cuales se componen de una secuencia de reuniones que buscan garantizar el cumplimiento de los compromisos del Equipo Scrum. El Cuadro 1 detalla las actividades que se deben realizar en cada Sprint.

Cuadro 1

Actividades de cada Sprint

Actividad	Descripción
Planificación	Esta actividad busca ordenar, determinar y entender el trabajo que se va a realizar en el Sprint. Este trabajo se hace en base a un proceso de valoración de cada requerimiento contenido en el Product Backlog, que define o estima el esfuerzo necesario y establece los criterios de aceptación. El resultado es un nuevo listado de tareas que se conoce como: Sprint Backlog. Una vez elegido el número de requerimientos, el equipo de desarrollo procede a desmenuzar cada uno en un conjunto de tareas más manejables.
Scrum diario	Se trata de una reunión corta donde cada miembro del equipo de desarrollo expone las actividades que se han realizado, cuales piensa hacer a continuación y qué impedimentos se han encontrado en el desarrollo del trabajo. Esto permite mantener un ritmo de trabajo constante que facilitará la detección a tiempo de problemas, así como también el conocimiento actualizado del estado de las tareas. Como resultado de esta reunión se obtiene un listado de impedimentos que deberán ser solucionados para evitar retrasos en el trabajo.

Continúa ➔

Actividad	Descripción
Revisión	En esta penúltima actividad se busca evaluar el incremento del producto. El equipo de desarrollo deberá tener “algo” que el cliente pueda usar, probar y decidir si cumple con los criterios de aceptación, definidos en la actividad de Sprint Planning.
Retrospectiva	El propósito de esta última reunión es la de evaluar el camino recorrido en el Sprint, ayuda a identificar los aciertos y los errores cometidos por el equipo de desarrollo.

Fuente: Adaptado, (Scrum-Alliance, 2014)

2.1.1.2 Roles de Scrum

Las actividades que plantea el Proceso Scrum, guían la elaboración del producto que el cliente necesita en base a sus requerimientos. Para dar vida al producto se requiere de un equipo de personas también conocido como: Scrum Team. El Cuadro 2 detalla los roles del Scrum Team.

Cuadro 2
Roles de Scrum

Roles	Descripción
Product Owner	Es la única persona, dentro del Scrum Team, que conoce el giro del negocio del cliente, mantiene la visión del producto, trabaja en estrecha relación con el cliente en el levantamiento de los requerimientos y facilita la comunicación con el equipo de desarrollo. Es también el encargado de mantener el Product Backlog actualizado y se asegura que el equipo de desarrollo construya el producto correcto.
Scrum Master	Esta persona debe dominar el marco de trabajo, que Scrum propone, con el fin de mantener en funcionamiento el proceso, ayudar al Product Owner con la elaboración y actualización del Product Backlog, gestionar los impedimentos que se van presentando en el desarrollo del proyecto y ayudar a que todas las personas involucradas entiendan Scrum.

Continúa ➡

Roles	Descripción
Equipo de desarrollo	Es un grupo de profesionales que deben tener las habilidades necesarias para elaborar y entregar el producto en base a los requerimientos planteados por el cliente. El equipo de desarrollo tiene la autoridad para auto-organizar su trabajo con el fin de alcanzar los objetivos de cada Sprint.

Fuente: Adaptado, (Scrum-Alliance, 2014)

2.1.1.3 Artefactos y Herramientas de Scrum

Con el objetivo de mantener el control y hacer el seguimiento del trabajo, que el equipo de desarrollo va a realizar, Scrum define una serie de artefactos y herramientas. El cuadro 2.3 detalla los artefactos que propone Scrum.

Cuadro 3

Artefactos de Scrum

Artefacto	Descripción
Product Backlog	Es una lista de ideas o requerimientos priorizados del producto, de donde se desprende el trabajo que el Scrum Team debe realizar . El encargado de elaborar y actualizar este listado es el Product Owner.
Sprint Backlog	Es el elemento resultante de la etapa de planificación del Sprint o Sprint Planning. Contiene el número de requerimientos refinados que el equipo de desarrollo se compromete a desarrollar en el período de duración de un Sprint.
Product Increment	Es el artefacto más importante de Scrum porque constituye el incremento de producto u objetivo que cada Sprint debe producir. Debe cumplir con los requerimientos planteados, ser utilizable por parte del usuario y aceptado por el Product Owner.

Fuente: Adaptado, (Scrum-Alliance, 2014)

El diagrama conocido como **Burn-Down Chart** establece la evolución del trabajo durante un Sprint. (Álvarez García, de las Heras del Dedo, & Lasa Gómez, 2012) La Figura 2 representa un ejemplo del diagrama **Burn-Down Chart**, donde la línea recta diagonal marca la evolución del trabajo ideal, el eje horizontal muestra el tiempo desde la fecha inicial hasta el final del Sprint y el eje vertical muestra la cantidad de requerimientos, tareas o historias de usuario que se van a realizar a lo largo del Sprint.



Figura 2 Diagrama Burndow Chart

Fuente: (Scrum-Alliance, 2014)

2.1.2 Extreme Programming (XP)

Kent Beck (1999) define a Extreme Programming (XP) como un método ágil para el desarrollo de software muy útil si tenemos que abordar proyectos con requerimientos cambiantes o vagamente definidos. XP se basa en un conjunto de valores que son los que guían el desarrollo, entre los que destacan: la comunicación, la simplicidad, la retroalimentación, el coraje y el respeto.

Para concretar los valores expuestos, XP propone principios básicos colaborativos como la humanidad, la búsqueda de beneficio mutuo, la mejora continua, la diversidad, el aprendizaje, la responsabilidad compartida, entre otros. (Álvarez García, de las Heras del Dedo, & Lasa Gómez, 2012)

2.1.2.1 Proceso XP

El proceso XP está compuesto por cuatro actividades estructurales: planeación, diseño, codificación y pruebas (Pressman, 2010). La Figura 3 muestra la secuencia de actividades que guían el proceso de Extreme Programming. El Cuadro 4 detalla las actividades estructurales de XP

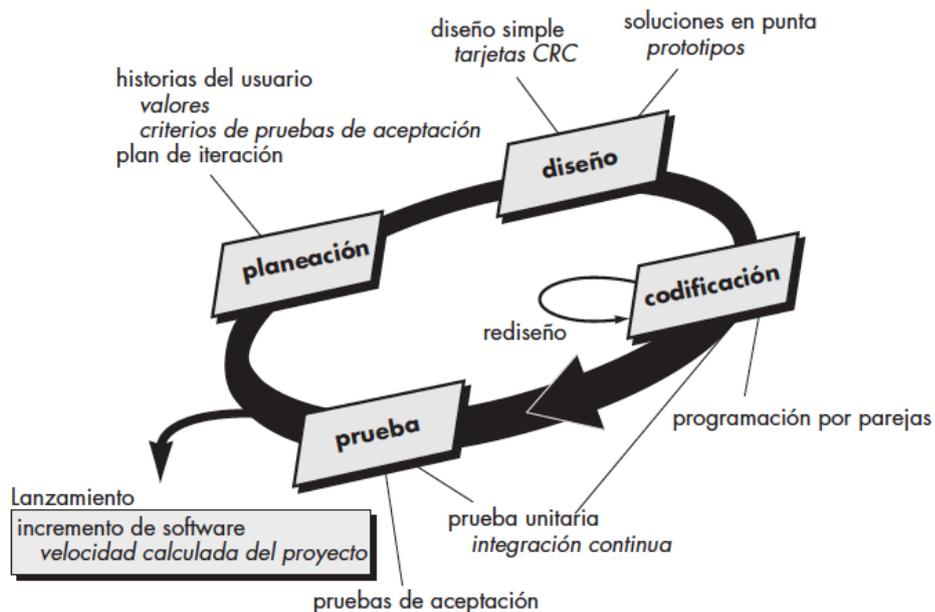


Figura 3 Proceso Scrum

Fuente: (Pressman, 2010).

2.1.2.2 Roles de XP

Extreme Programming propone al menos cuatro roles, siendo el cliente y el desarrollador indispensables dentro del ciclo de vida del proyecto (Chromatic, 2013). El Cuadro 5 detalla los roles dentro del proceso XP.

2.1.2.3 Prácticas de Extreme Programming (XP)

Las prácticas determinan la manera en la que el equipo de desarrollo trabaja día a día. La adopción de una u otra práctica puede variar en función del contexto en el que se esté trabajando. (Álvarez García, de las Heras del Dedo, & Lasa Gómez, 2012). El cuadro 2.6 detalla las practicas de XP.

Cuadro 4

Actividades de XP

Actividad	Descripción
Planeación	<p>Esta actividad tiene como objetivo recabar los requerimientos, conocidas como “historias de usuario”, con las características y funcionalidades necesarias del software para que los miembros del equipo XP puedan entender el contexto del negocio.</p> <p>El cliente asigna un valor a cada requerimiento tomando en cuenta el grado de funcionalidad, que éste, le brindará al negocio. Después, el equipo XP evalúa cada requerimiento y le asigna un costo, medido en semanas de desarrollo.</p> <p>Al finalizar esta actividad, tanto el cliente, como los miembros del equipo XP deberán acordar el contenido de la entrega y el tiempo que tomará construirlo, tiempo que no suele ser superior a tres semanas.</p>

Continúa ➡

Actividad	Descripción
Diseño	<p>XP sigue rigurosamente el principio de un diseño sencillo y estimula el uso de las tarjetas CRC (Clase-Responsabilidad-Colaborador) como único producto que se genera del trabajo de diseño en el proceso XP.</p> <p>El diseño ocurre tanto antes, como después de iniciada la codificación, por lo que es una actividad en transición que puede y debe modificarse a medida que se avanza en la construcción del software, a esto se lo conoce como rediseño.</p>
Codificación	<p>Después que las historias han sido evaluadas y diseñadas, el equipo desarrolla una serie de pruebas unitarias a cada una de las historias incluidas en la entrega pactada.</p> <p>Estas pruebas unitarias permitirán al desarrollador estar mejor capacitado para construir la solución, enfocado siempre en superar la prueba planteada. Una vez terminado el código, se le aplica la prueba unitaria, con lo que se consigue una retroalimentación instantánea para los desarrolladores.</p>
Pruebas	<p>Las pruebas de aceptaciones son especificadas por el cliente y se centran en las “historias” que fueron recabadas en la actividad de Planificación.</p> <p>Las pruebas unitarias individuales brindan una solida base a esta actividad del proceso XP, permitiendo obtener una indicación continua de avance, así como lanzar las señales de alerta oportunas si las cosas marchan mal.</p>

Fuente: Adaptado, (Pressman, 2010)

Cuadro 5

Roles de XP

Rol	Descripción
Cliente	<p>XP siempre se refiere al cliente como una sola persona, que es quien define el proyecto y las metas que se deben cumplir en función al problema del negocio que se quiere atender. El cliente es la voz del final y el equipo XP o incluso ser el cliente final.</p>

Continúa ➡

Rol	Descripción
Desarrollador	El desarrollador es la persona encargada de llevar a cabo la mayoría de las practicas que XP plantea. Se encarga de diseñar las pruebas unitarias y escribir el código en base a las “historias de usuario” que ha evaluado y escogido.
Rastreador	Es el encargado dar el seguimiento al proyecto en base a la velocidad de desarrollo del equipo. Esta medida se obtiene de la relación entre el tiempo estimado para la tarea y el tiempo real invertido en el desarrollo. Otros datos importantes que se pueden utilizar son, el sobretiempo trabajado y la relación entre las pruebas exitosas y las fallidas.
Coach o Tutor	Algunos proyectos que trabajan con XP, cuentan con un tutor, que es quien ayuda al equipo de a adoptar la metodología, sugiriendo o determinando la mejor manera de implementación de las prácticas que XP plantea.

Fuente: Adaptado, (Chromatic, 2013)

Cuadro 6

Prácticas de XP

Práctica	Descripción
Historias de usuario	Los requerimientos deben estar en lenguaje de cliente y ofrecer el mayor valor funcional posible a la aplicación.
Ciclos semanales	Propone atender los requerimientos del cliente en períodos de una semana.
Revisiones trimestrales	En escala de tiempo mayores, es oportuno analizar el estado del equipo, de los progresos y la continuidad de los objetivos.
Trabajar con holgura	Se trata de establecer tiempos de entrega realistas, tomando en cuenta posibles imprevistos.
Sentarse juntos	Para facilitar la comunicación entre los miembros del equipo de desarrollo, es importante sentar a todos en una misma sala y en un espacio abierto.
Equipo completo	El equipo debe estar compuesto por personas con los perfiles idóneos para abordar el desarrollo.

Continúa ➡

Práctica	Descripción
Información en el puesto de trabajo	Desde el puesto de trabajo debe poderse consultar el estado del proyecto y las tareas a realizarse.
Ritmo sostenible	Trabajar de manera productiva, eliminando cualquier distracción para evitar las horas extras de trabajo que desgasten al equipo de desarrollo.
Programación en parejas	Compartir el conocimiento y el código entre dos personas, ayuda a aclarar las ideas en el proceso de desarrollo.
Diseño incremental	El diseño incremental del software permite la evolución de la aplicación. Cada rediseño obliga, al equipo de desarrollo, a refactorizar el código.
Pruebas antes de programar	Las pruebas unitarias obligan, al desarrollador, a enfocarse en la solución del problema y a confirmar el diseño.
Construir en diez minutos	Debe ser posible compilar y probar el sistema en diez minutos sin que esto impacte el ritmo de trabajo. Si algo falla en el proceso, esto permitirá rectificar de inmediato.
Integración continua	Cada vez que se tenga una nueva funcionalidad y que el código sea modificado, se debe recompilar y probar la integración del software.

Fuente: Adaptado, (Álvarez García, de las Heras del Dedo, & Lasa Gómez, 2012).

2.2 Scrum en combinación con Extreme Programming (XP)

Tanto Scrum como XP son marcos de trabajo que se pueden complementar entre sí. Scrum trata de organizar la forma en la que los equipos abordan el trabajo mientras que el objetivo de Extreme Programming (XP) es construir software de manera productiva, con menos defectos y más económico. (Álvarez García, de las Heras del Dedo, & Lasa Gómez, 2012)

Henrik Kniber (2007) plantea que algunas practicas de XP, como por ejemplo “Equipo completo”, “Sentarse juntos”, “Historias de usuario” y “Planificación” son tratadas directamente por Scrum.

En base a estas dos afirmaciones podemos definir que actividades y artefactos utilizaremos de Scrum y que practicas, de XP, se aplicarán en el desarrollo de este proyecto.

2.2.1 Actividades, artefactos y herramientas tomados de Scrum

Las actividades y artefactos elegidos para organizar el trabajo del presente proyecto son:

- **Etapa inicial o Sprint 0:** Determinar la viabilidad del proyecto y elaborar la primera versión del Product Backlog en base a los requerimientos generales expuestos por el cliente.
- **Product Backlog o pila del producto:** Documento que contiene los requerimientos del cliente y que deberá ser refinado y actualizado en cada iteración.
- **Planificación del Sprint:** Reunión con todo el equipo de desarrollo que definirá el trabajo a realizar en el Sprint.
- **Sprint Backlog o pila de tareas:** Documento que contendrá el listado de requerimientos refinados que se atenderán en el Sprint
- Reunión de Scrum diario.
- **Revisión o Sprint Review:** Reunión con el cliente que evaluará el cumplimiento del trabajo pactado en base a los criterios de aceptación.
- **Sprint Increment:** Producto final de cada sprint.
- **Burn-Down Chart:** Diagrama de seguimiento para cada Sprint.

2.2.2 Practicas tomadas de Extreme Programming (XP)

Las practicas fundamentales expuestas por la metodología de desarrollo de software Extreme Programming (XP), para la programación y codificación de un sistema de información son:

- **Pruebas antes de programar:** Programar el software en base a las pruebas unitarias que se diseñaran para cada requerimiento que se necesite desarrollar.
- **Diseño Incremental:** Mejorar el diseño del software conforme se avanza en el desarrollo. Reescribir los métodos y las clases que sean necesarias.
- **Integración Continua:** Compilar el código y desplegarlo cada vez que se tenga una nueva funcionalidad del software.

2.3 Plataforma Arduino

Arduino es un plataforma abierta de computación física compuesta por un microcontrolador montado en una placa electrónica, que puede ser usada para desarrollar objetos interactivos capaces de recolectar datos por medio de sensores y controlar luces, motores u otros elementos físicos (Banzi, 2011). La plataforma Arduino está formada por dos elementos fundamentales:

- La placa electrónica Arduino que vendría a ser el hardware y es con lo que se construye los objetos interactivos
- El software o Ambiente de Desarrollo Integrado (Arduino IDE) que se usa para programar el microcontrolador que está montado en la placa electrónica.

2.3.1 Hardware

La placa Arduino cuenta con un pequeño circuito que incorpora un microcontrolador de la familia ATmega. Existen varias versiones de las placas Arduino, pero todas ellas implementan una serie de entradas y salidas, analógicas y digitales (Banzi, 2011). La Figura 4 detalla las entradas y salidas de la placa Arduino Uno.

La programación del microcontrolador se la realiza por medio del puerto USB, el que debe estar conectado al puerto de computadora debidamente configurado con el driver proporcionado por Arduino. La energía necesaria para que el Arduino funcione, se la puede suministrar, por medio de un adaptador de corriente alterna (AC) que entregue entre 9 y 12 voltios, utilizando el plug de conexión externo o por medio de un cable USB conectado al puerto (Arduino, 2014).

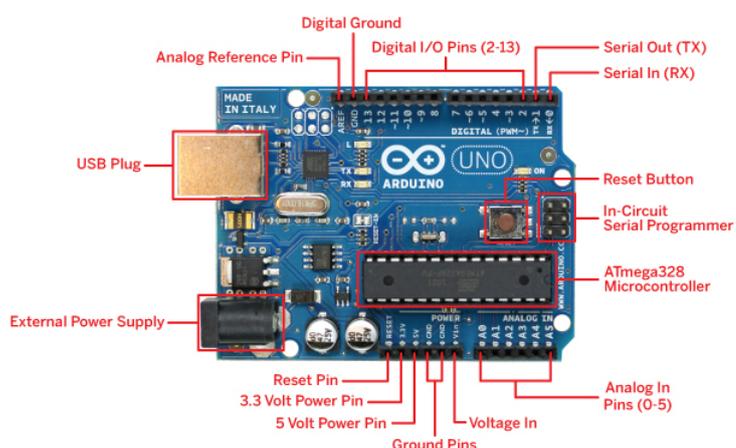


Figura 4 Esquema de la placa Arduino Uno.

Fuente: (Arduino-Arts, 2014)

2.3.2 Entorno de Programación

El Ambiente de Desarrollo Integrado o IDE (Integrated Development Environment), es un programa multiplataforma que permiten escribir las piezas de código, también conocidos como “sketchs”, que van a ser ejecutadas por el microcontrolador. El código escrito en este ambiente de desarrollo es traducido al lenguaje de programación C para luego ser compilado y transmitido a la placa Arduino para que pueda ser ejecutado (Banzi, 2011).

El cuerpo de una pieza de software, o sketch, está compuesto por dos funciones llamadas *setup()* y *loop()*. La función *setup()* contendrá el código que se ejecutará una sola vez, mientras que la función *loop()*, que es el núcleo del programa, se ejecutará una y otra vez mientras la placa se encuentre conectada a la fuente de alimentación.

2.3.3 Módulos de Expansión

Los módulos de expansión o *Arduino Shields*, son circuitos electrónicos que se conectan a la placa Arduino para extender la funcionalidad. Estos circuitos están concebidos para brindar capacidades adicionales a la placa Arduino original como la comunicación de datos y acceso a Internet por medio de redes Ethernet e incluso redes inalámbricas (Arduino Shields, 2014).

2.3.3.4 Xbee Shield

El módulo de expansión *Xbee Shield* permite, a las tarjetas Arduino, la instalación de componentes electrónicos específicos que facilitan la comunicación inalámbrica en distancias de 100 pies en interiores y 300 pies en exteriores. Este tipo de módulos utilizan el estándar ZigBee y se comunican con el microprocesador, de la tarjeta Arduino, por medio de una conexión serial (ArduinoXbeeShield, 2014).

2.3.3.5 Xbee WiFly

Es un módulo WiFi construido por Rovin Network que implementa el estándar IEEE 802.11 b/g para la comunicación con redes inalámbricas y que presenta un bajo consumo de energía. Está especialmente diseñado para simplificar la integración y disminuir el tiempo de desarrollo.

2.3.4 Sensores

Massimo Banzi (2011), define a los sensores como elementos electrónicos capaces de interactuar con el ambiente que los rodea. Así como los ojos convierten la luz en una señal que el cerebro transforma en imágenes, los sensores transforman los eventos físicos en señales eléctricas que pueden ser interpretadas por el microcontrolador de la placa Arduino.

2.3.4.1 Sensor de Densidad de Polvo

El sensor de densidad de polvo utiliza un diodo infrarrojo junto con un fototransistor para detectar el reflejo de la luz que provoca el polvo en el ambiente. Este sensor es especialmente efectivo para detectar partículas finas y es comúnmente utilizado en sistemas de purificación del aire (Sparkfun-Sharp, 2006).

2.3.4.2 Sensor de Monóxido de Carbono

El sensor MG-811 es altamente sensible a la concentración de Dióxido de Carbono (CO₂). El voltaje de salida de este sensor cae cuando la concentración de CO₂ se incrementa (DFRobot, 2014).

2.3.4.3 Sensor de Dióxido de Carbono

El sensor MQ-7 puede detectar la concentración de Monóxido de Carbono en el aire en el rango de 20 hasta 2000 ppm, tiene una alta sensibilidad y un corto tiempo de respuesta. La salida de este sensor es analógica (DFRobot, 2014).

2.4 Plataforma de desarrollo

El desarrollo de este prototipo exige varias características a la plataforma de desarrollo que se va a utilizar, entre los criterios de selección se encuentran:

- a) Grado de madurez
- b) Documentación disponible
- c) Disponibilidad de librerías
- d) Compatibilidad e integración con otros sistemas
- e) Rendimiento y escalabilidad.

2.4.1 Plataforma Java Enterprise Edition (JEE)

La plataforma Java Enterprise Edition o Java EE, por sus siglas en Inglés, es un conjunto de componentes conocidas como API's, cuyo objetivo principal es simplificar el desarrollo de software empresarial y fue diseñada para soportar aplicaciones que implementan servicios empresariales para consumidores, empleados, proveedores y socios estratégicos de la empresa. Este tipo de aplicaciones generalmente son complejas, acceden a varias fuentes de información y son consumidas por varios tipos clientes. (Oracle, 2014). Las principales características que provee esta plataforma son:

2.4.2 Modelo Multicapa

La plataforma Java EE implementa un modelo distribuido multicapa para las aplicaciones empresariales, lo que quiere decir que los elementos que componen el software están divididos de acuerdo a una función y que suelen estar instalados en diferentes computadores. Las aplicaciones de este tipo suelen estar compuestas por la capa cliente, la capa web, la capa de negocio y la capa de datos también llamada Sistema Empresarial de Información o EIS (Oracle, 2014). La Figura 5 representa el modelo multicapa que implementa Java EE.

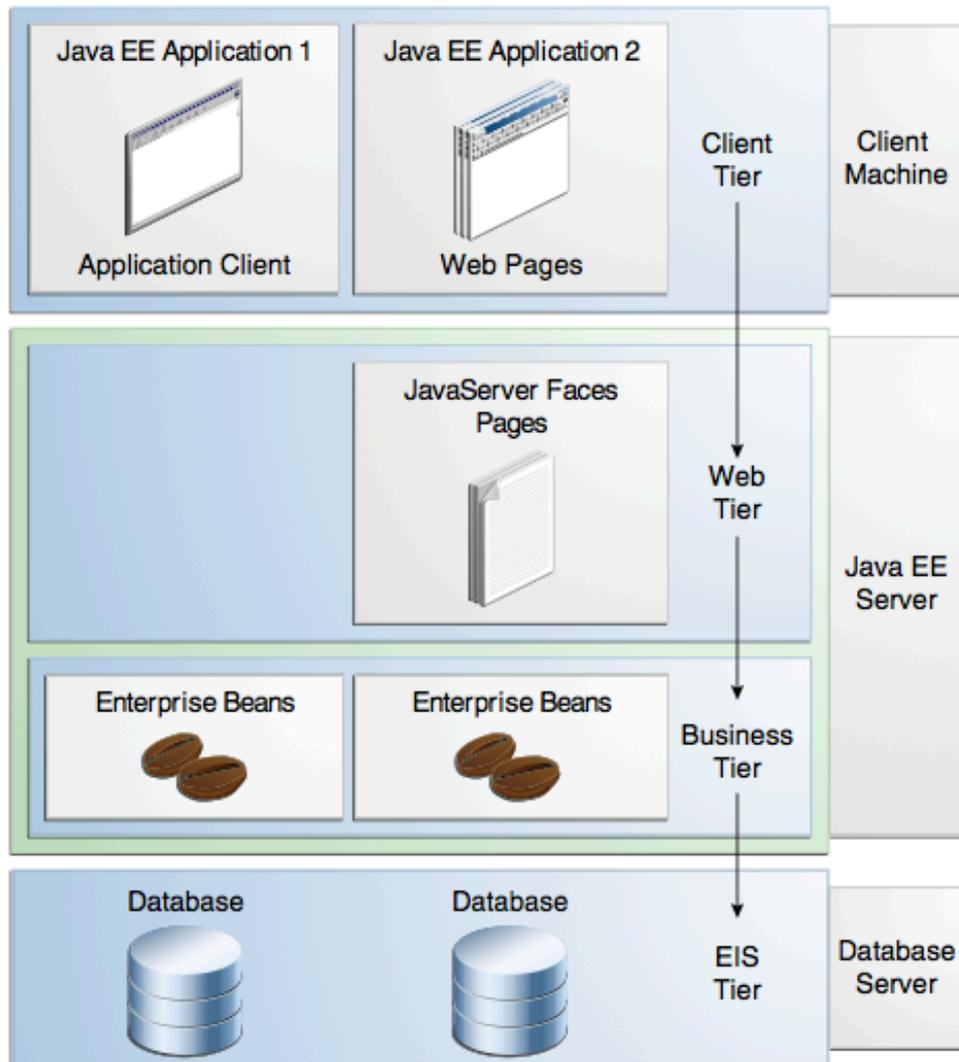


Figura 5 Modelo Multicapa de Java EE.

Fuente: (Oracle, 2014)

2.4.2.1 Capas del Modelo Java Enterprise Edition

El Cuadro 7 describe las capas con las que cuentan la mayoría de las aplicaciones distribuidas desarrolladas bajo la plataforma Java Enterprise Edition (JEE). Pueden tener mayor número de capas dependiendo del patrón de diseño que se vaya a emplear por ejemplo se podría agregar una

capa al momento de construir un servicio Web como el que se plantea para este proyecto.

Cuadro 7

Capas de la aplicación

Capa	Descripción
Capa Cliente	<p>La capa Cliente consiste en una aplicación que hace un petición al servidor Java EE, este procesa la petición y devuelve el resultado a la aplicación cliente.</p> <p>Los clientes pueden ser aplicaciones Web, aplicaciones de escritorio o incluso otros servidores, todas ellas desarrolladas en diferentes lenguajes de programación o en Java.</p>
Capa Web	<p>La capa Web es un conjunto de componentes que manejan la interacción entre los clientes y la capa de negocio. Sus principales tareas son:</p> <ul style="list-style-type: none"> • Generar contenido dinámico para el cliente. • Recoger la información ingresada por el cliente, por medio de la interfaz de usuario y retornar la información apropiada desde la capa de negocio. • Controlar la navegabilidad de las pantallas o paginas del cliente. • Manejar las sesiones del usuario. • Ejecutar lógica básica y guardar datos temporales del usuario. <p>Las tecnologías usadas en la capa Web son:</p> <ul style="list-style-type: none"> • JavaServer Faces (JSF): Framework para construir interfaces de usuario usando páginas XHTML también llamados Facelets. • Expression Language: Conjunto de tags o etiquetas usadas en los Facelets. • Servlets: Clases Java que manejan las peticiones y respuestas. • Inyecciones de Dependencia: Conocidas comúnmente como Dependency Injection, son un conjunto de servicios que permiten el uso sencillo de Java Beans con la tecnología JavaServer Faces.
Capa de Negocio	<p>Esta capa está compuesta por un conjunto de componentes que brindan funcionalidad necesaria a cada una de las reglas del ámbito del negocio. Las tecnologías que forman parte de esta capa son:</p> <ul style="list-style-type: none"> • Componentes JavaBeans • Servicios Web RESTful • Java Persistence API o JPA

Continúa 

Capa	Descripción
	La capa de datos, también conocida como capa de Información Empresarial o Enterprise Information System (EIS), está compuesta por servidores de bases de datos que habitualmente se encuentran instaladas en una máquina diferente al del servidor de aplicaciones.
Capa de Datos	La capa de negocio tiene acceso al Sistema de Información Empresarial utilizando las siguientes tecnologías: <ul style="list-style-type: none"> • Java Database Connectivity (JDBC) • Java Persistence API (JPA) • Java EE Architecture • Java Transaction API (JTA)

Fuente: Adaptado, (Oracle, 2014)

2.4.3 Servidor Java EE

Un servidor Java EE, comúnmente conocido como servidor de aplicaciones, implementa las API's que forman de la plataforma Java EE y brindan el acceso a la información almacenada. Estos servicios son expuestos en forma de contenedores, que son la interface entre los componentes de la aplicación y la funcionalidades de bajo nivel. La funcionalidad del contenedor está determinada por los diferentes tipos de componentes soportados entre los que se destacan los contenedores Web, EJB, RESTful Web Services (Oracle, 2014).

2.4.3.1 Contenedores Web

El contenedor Web es la interfaz entre los componentes Web y el servidor Web. Gestionan el ciclo de vida de las páginas, manejan los pedidos y son el nexa con el contexto de datos. Estos componentes pueden ser páginas JavaServer Faces (Oracle, 2014).

2.4.3.2 Contenedor EJB

El contenedor EJB es la interfaz entre los componentes de las reglas del negocio o Enterprise Java Beans (EJB). Maneja la ejecución de los EJB de la aplicación.

2.4.3.3 Contenedor de Aplicación Cliente

Este contenedor es la interfaz entre las aplicaciones cliente Java EE y el servidor de aplicaciones Java EE. El contenedor de aplicación cliente se ejecuta en la maquina cliente y es la puerta de enlace entre la aplicación cliente y los componentes del servidor de aplicaciones Java EE.

2.4.3.4 Servicio Web RESTful

Los Servicios Web de tipo REST o Transferencia de Estado Representacional, son un conjunto de principios arquitectónicos enfocados en los recursos que pueden ser abordados y transferidos sobre el protocolo de comunicaciones http (IBM, 2008)

2.5 Herramientas de Desarrollo

Para la construcción del prototipo se ha tomado en cuenta una combinación de herramientas completamente compatibles con la plataforma descrita anteriormente. Las herramientas de desarrollo escogidas para desarrollar la aplicación Web se detallan a continuación.

2.5.1 Eclipse

Es un entorno de desarrollo integrado o IDE, por sus siglas en Inglés, de código abierto y que puede ser utilizado para desarrollar aplicaciones Java

EE. Debido a su arquitectura de complementos, Eclipse cuenta con herramientas para la edición gráfica de HTML, JavaServer Pages, JavaServer Faces y la gestión de base de datos. Soporta la mayoría de servidores de aplicaciones. (Daum, 2004)

2.5.2 Servidor WildFly

WildFly es la evolución del servidor de aplicaciones JBoos desarrollado por la empresa RedHat. Implementa el modelo Java EE 7 y está concebida para soportar aplicaciones orientadas a los servicios Web (WildFly, 2014).

2.5.3 MySQL

Es el sistema de base de datos relacional de código abierto más popular alrededor del mundo. Ampliamente usado para aplicaciones Web por su confiabilidad, fácil uso y alto rendimiento. Actualmente está desarrollado y mantenido por la empresa Oracle (MySQL, 2014).

2.5.4 JQuery y Bootstrap

Son librerías escritas en JavaScript, que permiten manipular los elementos del HTML para modificar su comportamiento y mejorar la experiencia del usuario al momento de usar la aplicación web. Orientadas a mejorar la interfaz de usuario, ambas librerías son ampliamente usadas junto con el estándar HTML5.

ESPECIFICACIÓN DE REQUERIMIENTOS

3.1 Introducción

La Ingeniería de Requerimientos es un amplio conjunto de tareas y técnicas que proporcionan el mecanismo apropiado para analizar, entender y especificar las necesidades del cliente (Pressman, 2010)

A continuación se elaborará una Especificación de Requerimientos de Software (ERS) para el proyecto planteado y que automatizará los procesos de medición, transmisión, tratamiento, almacenamiento y presentación de la información sobre la concentración de contaminantes en la ciudad de Quito.

3.2 Propósito

La presente Especificación de Requerimientos de Software pretende identificar el contexto del proyecto y determinará los requerimientos funcionales y no funcionales.

Esta especificación se basará en el estándar del Instituto de Ingenieros Eléctricos y Electrónicos (IEEE por sus siglas en Inglés), Practicas Recomendadas para los Requerimientos de Software IEEE/ANSI 830-1998 (IEEE, 2014).

3.3 Alcance

El proyecto denominado “GreenAir”, atenderá los procesos de medición, transmisión, tratamiento ,almacenamiento y presentación de la información

concerniente a la concentración de contaminantes en la atmosfera de la ciudad de Quito. El Cuadro 8 detalla estos procesos.

Cuadro 8

Procesos del prototipo

Proceso	Descripción
Medición	Este primer proceso deberá medir la concentración de los contaminantes en el ambiente con el uso de sensores electrónicos debidamente montados, configurados y calibrados en la placa electrónica Arduino Uno.
Transmisión	La placa Arduino Uno procesará la información que los sensores entreguen y la transmitirá a la aplicación Web. La información deberá cumplir la sintaxis y semántica definida por el protocolo HTTP para la transferencia de los datos.
Tratamiento	Se encargará de recibir y procesar la información enviada por la placa electrónica Arduino. Para atender a la solicitud del sistema cliente, se utilizará un Servicio Web de tipo RESTful que procesará la información y facilitará su verificación.
Almacenamiento	Una vez verificada la información, este proceso la almacenará, en un sistema de base de datos, para su posterior análisis. La estructura, del sistema de base de datos, garantizará la integridad de la información recibida.
Presentación	De los datos almacenados anteriormente, la aplicación Web entregará información de fácil lectura que evidencie la calidad del aire en el ambiente, los medios para cumplir con este requerimiento serán: <ul style="list-style-type: none"> • Mapa de ubicación geográfica de los dispositivos electrónicos y sus sensores. • Gráfico de las lecturas en un período de tiempo dado. • Estado de los sensores.

3.4 Limitaciones del Prototipo

El proyecto implementará las funciones principales de un sistema completo de este tipo, por lo que se han determinado algunas limitaciones que se detallan a continuación:

- No encriptará la información que se transmite de la plataforma Arduino a la aplicación Web.
- No implementará el protocolo HTTPS para la transmisión de datos segura.
- No incluirá algoritmos de Inteligencia Artificial que permitan el análisis e interpretación de los datos medidos.
- No consultará información de otros sistemas de calidad del aire.
- No incluirá un desarrollo específico para dispositivos móviles como tabletas o teléfonos inteligentes.
- No implementará ningún tipo de alerta en base a los parámetros medidos, ya que la información que se espera recolectar es referencial y no exacta.

Los beneficios de una aplicación Web, capaz de interactuar con sensores de calidad del aire, para procesar información sobre la concentración de contaminantes en el ambiente son:

- Ampliar el ámbito de medición de las redes de monitoreo fijas
- Conocer la concentración de contaminantes en el ambiente.
- Compartir la información sobre la concentración de contaminantes.
- Almacenar la información para estudios posteriores.

3.5 Personal Involucrado

El equipo de personas que llevarán a cabo este proyecto está compuesto por profesionales en distintos campos del conocimiento. Es necesario contar un experto en el campo del medio ambiente que valide la información que los dispositivos electrónicos recolectarán y transmitirá a la aplicación Web para ser procesada y compartida. El Cuadro 9 detalla los roles que cumplirán cada uno de los miembros del equipo de trabajo.

Cuadro 9

Personal involucrado

Nombre	Campo Profesional	Rol	Información de Contacto
Daniela Torres	Ingeniera Ambiental	Experto ambiental	alisdaniela@gmail.com
Walter Fuertes	Ingeniero en Sistemas	Director del proyecto	wmfuertes@espe.edu.ec
César Villacís	Ingeniero en Sistemas	Co-director del proyecto	cvillacis@espe.edu.ec
Marco Jarrin	Ingeniero en Sistemas	Sponsor del proyecto	mjarrin@omnisoft.cc
Diego Carrera	Egresado de la Carrera de Ingeniería en Sistemas	Analista, diseñador y programador	diegomcarrera@gmail.com

3.6 Definiciones

Prototipo: Es una primera versión o modelo del software en el que se ha incorporado algunas características del producto final.

Contaminantes: Sustancias químicas que alteran negativamente el estado natural del ambiente.

Dispositivo electrónico: Placa Arduino capaz de conectarse a Internet y que se encuentra dotada de sensores.

Arduino: Placa electrónica que facilita el desarrollo de prototipos, existen varias versiones de placas con diferentes características.

Sensores: Elementos electrónicos capaces de interactuar con el ambiente que los rodea.

Servicio Web: Es una tecnología que utiliza un conjunto de protocolos para el intercambio de datos entre aplicaciones.

Base de Datos: Conjunto de datos relacionados entre sí.

Sistemas de Bases de Datos: Software que facilita la gestión de una base de datos.

3.7 Acrónimos

IR: Identificación de requerimiento.

RE: Requerimiento específico.

NR: Nombre del requerimiento.

ERS: Especificación de Requerimientos de Software.

TCP/IP: (Transfer Control Protocol - Internet Protocol), hace referencia al conjunto de protocolos en los que se basa el Internet.

HTTP: (HyperText Transfer Protocol), protocolo de transferencia de hipertexto usado en cada transferencia sobre Internet.

HTTPS: (HyperText Transfer Protocol Secure), protocolo seguro de transferencia de hipertexto.

RESTful: Transferencia de Estado Representacional (Representational State Transfer por sus siglas en Inglés), es un estilo arquitectónico que permite la comunicación de datos entre sistemas usando el protocolo HTTP.

IEEE: Instituto de Ingenieros Eléctricos y Electrónicos.

IEEE 802.11 b/g: Norma de funcionamiento para redes inalámbricas.

3.8 Referencias

IEEE-STD-830. (1998). Especificación de Requerimientos de Software. (IEEE, 2014)

Pressman, R. (2010). Ingeniería del software: Un enfoque práctico. México: Interamericana Editores S.A. McGraw-Hill

3.9 Visión General del Documento

Esta Especificación de Requerimientos (ERS) está dividida en tres secciones. La primera proporciona una introducción al ERS y una visión general del proyecto que se desarrollará.

En la segunda sección del ERS se describirá, de manera general, las funciones de la aplicación Web y el dispositivo electrónico, los datos

asociado, las características de los distintos tipos de usuarios, restricciones, suposiciones y dependencias que afectan al desarrollo, sin entrar en excesivos detalles. Por último, la tercera sección del ERS recogerá detalladamente los requerimientos funcionales y no funcionales del proyecto

3.10 Descripción General

En esta sección de la especificación de requerimientos, se hará una descripción general del producto que se pretende construir para brindar una solución al problema planteado.

3.10.1 Perspectiva del Producto

El prototipo de aplicación Web, los dispositivos electrónicos y sus sensores, usaran un conjunto de tecnologías idóneas y que deberán interactuar entre sí, para medir la concentración de contaminantes en el ambiente. Esta información será recolectada, transmitida, procesada, almacenada y compartida, con el objetivo de mantener, a la comunidad de la ciudad de Quito, correctamente informada sobre la calidad del aire que la rodea.

3.10.2 Funciones del Producto

La Figura 6 ilustra las funciones generales que deberán implementarse tanto por el prototipo de aplicación Web, como por la tarjeta electrónica Arduino y sus componentes. Estas funciones están relatadas en el Cuadro 10, en función a los procesos que se ejecutarán con el objetivo de cumplir con la perspectiva del producto.

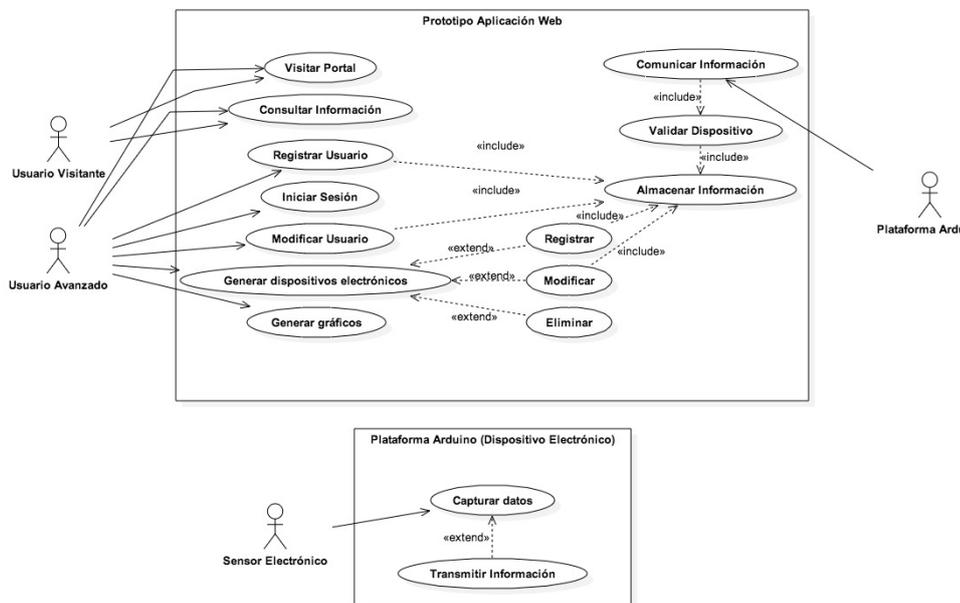


Figura 6 Funciones generales

Cuadro 10

Funciones generales del prototipo.

Proceso	Funciones
Medir	La función principal en este proceso será la de recolectar los valores de la concentración de contaminantes que producirán los sensores de calidad del aire. Cada lectura generada será almacenada en una variable temporal definida en el software que ejecuta la placa Arduino.
Transmitir	Una vez capturada la información, se la procesará para ser transmitida a la aplicación Web. Esta transmisión de datos deberá cumplir con la semántica establecida por el protocolo HTTP y se la hará en períodos de tiempo debidamente definidos y programados en el software que se ejecuta en el dispositivo electrónico
Procesar	La aplicación Web recibirá la información y la procesará para que pueda ser almacenada. Se incluye una función de autenticación para el dispositivo electrónico, el cuál debe estar registrado, configurado y habilitado para transmitir la información hacia el prototipo Web.

Continúa ➡

Proceso	Funciones
Almacenar	Una vez autenticado el dispositivo electrónico, la aplicación Web permitirá el almacenaje de la información captada en un sistema de base de datos. La estructura de la base de datos deberá garantizar la integridad de los datos y se incluirá información como hora y fecha de la lectura.
Presentar	La presentación de la información almacenada deberá ser amigable y de fácil lectura, incluyendo un mapa de ubicación de los dispositivos electrónicos, estado y lectura actual. Adicionalmente se deberá generar un gráfico en función al tiempo y las lecturas.

3.10.3 Características del los Usuarios

El prototipo de aplicación Web tendrá tres tipos de usuarios, dos de los cuales son humanos y el restantes es un dispositivo electrónico capaz de captar la información requerida. Cada dispositivo electrónico, podrá contar con uno o más sensores, los que a su vez serán usuarios del software que lo controla. Este pequeño programa, que se construirá usando el entorno de desarrollo que provee Arduino, procesará la información captada por cada sensor y permitirá su transmisión hacia la aplicación Web. El Cuadro 11 describe las características generales de cada tipo de usuario.

Cuadro 11

Características de los usuarios

Usuario	Descripción	Capacidad Técnica
Visitante	Podrá consultar la información sobre la concentración de contaminantes desde el portal de la aplicación Web, sin necesidad de registrarse como usuarios.	No necesitará una capacidad técnica especializada y solo deberá estar familiarizado con el uso de un navegador Web.

Continúa ➡

Usuario	Descripción	Capacidad Técnica
Avanzado	Podrá registrarse como usuario de la aplicación web para conectar y gestionar los dispositivos electrónicos desarrollados o mantenidos por él.	Deberá tener formación especializada y estar familiarizado con la Plataforma Arduino, electrónica básica, sensores electrónicos, protocolos de transmisión de datos y servicios web RESTful.
Dispositivo Electrónico	Procesará la información entregada por los sensores electrónicos y podrá enviarla, haciendo uso de los protocolos de transmisión de datos, hacia la aplicación Web.	Deberá tener la capacidad suficiente para captar información, procesarla y transmitirla usando los protocolos de comunicación de datos http y TCP/IP.
Sensor Electrónico	Captará las variación de contaminantes y entregará esta información al dispositivo electrónico para que sea procesada y enviada.	Deberá reaccionar a los cambios físicos del ambiente para los que está diseñado y construido.

3.10.4 Restricciones

Las restricciones detalladas a continuación, permitirán conocer las limitaciones del hardware que se empleará en el desarrollo del proyecto, las interfaces con otras aplicaciones, los protocolos de comunicación de datos y los lenguajes de programación que se utilizarán. El Cuadro 12 describe cada una de las restricciones.

3.10.5 Suposiciones y Dependencias

El equipo donde se ejecutará la aplicación Web, deberá cumplir con los requisitos necesarios para soportar el servidor de aplicaciones y el sistema de gestión de base de datos. Deberá también, estar conectado

adecuadamente al Internet para permitir la transmisión de datos desde los dispositivos electrónicos.

Cada dispositivo electrónico tendrá una fuente de alimentación de energía adecuada y estarán conectados a Internet, sin estas características no se podrá transmitir la información recolectada hacia la aplicación Web.

Cuadro 12

Descripción de las restricciones del proyecto

Restricciones	Descripciones
Limitaciones del hardware	La placa electrónica Arduino, en su modelo UNO R3, tiene 32 Kilobytes de capacidad de almacenamiento, lo que limita el tamaño de los programas que se ejecutan para medir y transmitir la información a la aplicación Web.
Interfaces con otras aplicaciones	El prototipo deberá implementar un Servicio Web de tipo RESTful para la integración con cualquier plataforma electrónica o informática.
Protocolos de comunicación	Los protocolos de comunicación de datos serán HTTP y TCP/IP.
Lenguajes de Programación	La necesidad de construir una aplicación Web para compartir la información y que sea capaz de interactuar con dispositivos electrónicos mediante el uso de Servicios Web, restringe el uso de un lenguaje de programación. Por otro lado, la plataforma Arduino utiliza el lenguaje "Processing" para la construcción de las piezas de código que ejecuta el microcontrolador. El lenguaje C++ se utilizará para crear las librería de comunicación de datos.

3.11 Requisitos Específicos

El Cuadro 13 recoge los requisitos que permitirán planificar, diseñar, desarrollar y probar el prototipo de aplicación Web y el software que ejecutará cada dispositivo electrónico.

Cuadro 13

Requisitos Específicos

RE	NR	Características
RE01	Consultar Información	Los visitantes podrán consultar la información de los sensores de calidad del aire.
RE02	Iniciar sesión	El usuario avanzado deberá identificarse para acceder a la gestión de los dispositivos electrónicos que le pertenecen.
RE03	Registrar Usuarios Avanzados	El usuario avanzado podrá registrarse para acceder a la gestión de sus dispositivos electrónicos.
RE04	Modificar Usuarios Avanzados	El usuario avanzado podrá modificar la información de su cuenta.
RE05	Gestión de dispositivos electrónicos	Se permitirá, al usuario avanzado, registrar y configurar nuevos dispositivos electrónicos y sus sensores, para que se puedan comunicar con la aplicación Web.
RE06	Gestión de dispositivos electrónicos	El usuario avanzado podrá modificar las configuraciones del dispositivo electrónico y sus sensores.
RE07	Gestión de dispositivos electrónicos	Se permitirá, al usuario avanzado, eliminar el dispositivo electrónico del sistema de base de datos.
RE08	Capturar y procesar las lecturas	El dispositivo electrónico podrá capturar las lecturas de los sensores y procesarlas.
RE09	Transmitir la información	El dispositivo electrónico deberá transmitir, hacia la aplicación Web, la información captada por los sensores electrónicos
RE10	Autenticar dispositivos electrónicos	Permite identificar los dispositivos electrónicos registrados.
RE11	Almacenar información	Permite almacenar la información recolectada y transmitida por los dispositivos electrónicos y sus sensores.
RE12	Gestionar gráficos	Permite generar gráficos de la concentración de contaminantes en función del tiempo.

3.11.1 Interfaces Externas

A continuación, se describirán los requisitos que afectan a las interfaces de usuario, hardware, software y comunicaciones. También se incluye el detalle de la interfaz que necesariamente se tendrá con el software que se ejecutará en el dispositivo electrónico Arduino.

3.11.1.1 Interfaces de Usuario

La interfaz, con el usuario visitante, será construida en base a un mapa de la ciudad de Quito, donde estarán geolocalizados, cada uno de los dispositivos electrónicos, y presentará la información de la concentración de contaminantes en tiempo real. Incluirá información sobre la unidad de medida y símbolo, así como también el estado de los sensores. Deberá ser de fácil consulta y de atractivo diseño.

La interfaz para el usuario avanzado estará constituida por un conjunto de ventanas, campos de texto, botones, listas y mapas que faciliten la gestión de los dispositivos electrónicos, sus sensores y la información que generarán. Se buscará que el diseño sea intuitivo y atractivo.

3.11.1.2 Interfaces de Hardware

La aplicación Web se ejecutará en un equipo de computo que cubra las características necesarias para soportar la ejecución del servidor de aplicaciones y el sistema de base de datos. Las características recomendadas son:

- Procesador 2.7Ghz
- Memoria RAM de 4Gb
- Disco duro 500gb
- Adaptador de red

- Monitor, teclado y mouse

Será necesario contar con una placa electrónica Arduino UNO R3 equipada con un módulo de expansión para conexión con redes inalámbricas IEEE 802.11 b/g, que permita la conexión a Internet. También son necesarios los sensores de Monóxido de Carbono (CO), Dióxido de Carbono (CO₂) y densidad de polvo.

3.11.1.3 Interfaces de Software

Tanto el servidor de aplicaciones, como el sistema de base de datos, deberán ser independientes del sistema operativo para asegurar su portabilidad. En este proyecto en particular se utilizará Linux como sistema operativo y el acceso a la aplicación Web será posible mediante el uso de un navegador de Internet moderno, que soporte el estándar HTML5.

3.11.1.4 Interfaces de Comunicación

La comunicación entre los usuarios, la aplicación Web y los dispositivos electrónicos se la hará en base a los protocolos de comunicación de datos TCP/IP y HTTP. El Servicio Web deberá soportar estos protocolos para lograr la interoperabilidad entre todos los componentes.

3.11.2 Funciones

En el Cuadro 14 se detallan las funciones que tendrán, tanto la aplicación Web como el dispositivo electrónico Arduino. Se han tomado en cuenta los requerimientos específicos anteriormente expuestos en el Cuadro 13

Cuadro 14

Funciones

RF	Nombre	Descripción
RF01	Consultar Información	La aplicación Web presentará en un mapa la ubicación geográfica de cada dispositivo electrónico junto con la información sobre concentración de contaminantes y en tiempo real.
RF02	Iniciar Sesión	La aplicación Web ofrecerá acceso a la gestión de los dispositivos electrónicos, solo a los usuarios avanzados debidamente registrados.
RF03	Registrar Usuarios Avanzados	La aplicación Web permitirá el registro como usuario avanzado suministrando datos como nombre, apellido, correo electrónico, etc.
RF04	Modificar Usuarios Avanzados	La aplicación Web permitirá modificar la información de registro del usuario avanzado.
RF05	Gestión de dispositivos electrónicos	La aplicación Web permitirá: <ul style="list-style-type: none"> • El registro y configuración de nuevos dispositivos electrónicos y sus sensores electrónicos. • Modificar el nombre del dispositivo electrónico y su localización, así como también la configuración de nombre, unidad de medida y símbolo de los sensores electrónicos que equipan al dispositivo electrónico. • Eliminar el dispositivo electrónico, y su información, del sistema de base de datos.
RF06	Capturar y procesar las lecturas	El software que se ejecuta en el dispositivo electrónico, capturará y procesará la información que generan los sensores electrónicos. Cada sensor tendrá su propia rutina para lograr este objetivo.
RF07	Transmitir la información	El dispositivo electrónico conectado a Internet transmitirá la información, captada por los sensores electrónicos, hacia la aplicación Web haciendo uso de los protocolos de comunicación de datos.

Continúa ➡

RF	Nombre	Descripción
RF08	Autenticar dispositivos electrónicos	La aplicación Web autenticará al dispositivo electrónico, el cual deberá estar registrado, habilitado y debidamente configurado.
RF09	Almacenar información	Una vez autenticada la información, se procederá a almacenar la información, proveniente del dispositivo electrónico, en el sistema de base de datos de manera estructurada para garantizar su integridad.
RF10	Generar gráficos	En base a la información almacenada en el sistema de base de datos, la aplicación Web presentará gráficamente la información recolectada. El gráfico permitirá conocer las concentraciones en función del tiempo.

3.11.3 Requisitos No Funcionales

La siguiente sección tratará sobre los atributos no funcionales que deberá cumplir el prototipo Web y el software que controla los dispositivos electrónicos. Estos factores garantizarán el rendimiento, seguridad, fiabilidad y disponibilidad del producto que se construirá.

3.11.3.1 Requisitos de Rendimiento

Tanto la aplicación Web, como el software que se ejecuta en el dispositivo electrónico Arduino, garantizarán un correcto desempeño. Todos procesos que se realizarán con el sistema de base de datos deberán estar diseñados para que la respuesta sea rápida y no comprometa el rendimiento del software. Con la ayuda del complementos de Eclipse: JUnit y Arquillian, se podrá definir con exactitud el tiempo de respuesta de la ejecución de los métodos programados, garantizando de esta forma un rendimiento óptimo del prototipo de aplicación Web.

3.11.3.2 Seguridad

Los visitantes solo podrán consultar la información pública que estará disponible en el portal de la aplicación Web, información que deberá estar dispuesta en una interfaz intuitiva y de fácil manejo.

La aplicación Web también garantizará la seguridad de la información de los dispositivos electrónicos pertenecientes a los usuarios avanzados, mediante las funciones de registro y autenticación. Estos usuarios solo podrán acceder a los datos de los dispositivos electrónicos que le pertenecen y que han sido registrados y configurados por este.

En cuanto a la seguridad en la transmisión de información desde los dispositivos electrónicos hacia la aplicación Web, se diseñará e implementará una función de verificación que permitirá identificar el origen de los datos y su composición.

3.11.3.3 Fiabilidad

Todas las funciones identificadas en este proyecto deberán operar de manera fiable. El almacenamiento de la información en el sistema de base de datos deberá garantizar la integridad y correcta relación entre los datos, esto facilitará su consulta posterior.

3.11.3.4 Disponibilidad

La aplicación Web será diseñada para siempre estar disponible para los usuarios visitantes, usuarios avanzados y dispositivos electrónicos. Esta

condición dependerá del equipo donde será instalada la aplicación y del servicio de Internet.

CAPÍTULO 4

DISEÑO DEL PROTOTIPO

4.1 Modelo de Datos

Un modelo es la representación de una porción de la realidad, que elimina los detalles irrelevantes para centrarse en aquellos que ayudan a la solución de un problema. Por lo tanto, un modelo de datos es un conjunto de herramientas que describen la estructura de una base de datos, las relaciones, las restricciones y en ocasiones incluye semántica o significado de los datos. (Hansen, 1997)

Para almacenar la información que generarán los dispositivos electrónicos es necesario diseñar una base de datos en base a un modelo que responda a los requerimientos especificados en el tercer capítulo. El modelo entidad-relación se ajusta a las características del proyecto ya que incorpora la representación explícita de objetos, atributos y relaciones, categorización de objetos, abstracciones y restricciones. También proporciona un método gráfico para mostrar la estructura conceptual de la base de datos y se basa en la identificación de objetos o entidades que son representaciones de la realidad como personas, lugares, eventos o conceptos acerca de los cuales se recopilan datos. Las entidades son descritas por atributos y conectadas mediante relaciones. (Ricardo, 2009)

Es importante establecer un estándar en cuanto a los nombres de las tablas y atributos que conformarán la base de datos, con el objetivo de facilitar su comprensión y mantenimiento. Los estándares propuestos para el diseño de la base de datos serán:

- El nombre de las tablas deberán escribirse en idioma Inglés y mayúsculas, en el caso que el nombre esté compuesto por dos palabras, estas deberán estar separadas por un guión bajo.
- Todas las tablas de la base de datos deberán tener una clave primaria ubicada al inicio, serán campos numéricos autogenerados y llevarán el nombre: SERIAL seguido por un guión bajo y el acrónimo de la tabla a la que pertenecen.
- La relación entre tablas estará dada por la claves foráneas para lograr la integridad referencial de los datos almacenados.

4.2 Diseño de la Base de Datos

El diseño de la base de datos se construirá tomando en cuenta los requerimientos establecidos en el tercer capítulo del presente proyecto y se utilizará el modelo entidad – relación para lograr una representación de la realidad que ayude a la solución del problema planteado. La correcta normalización y relación entre tablas, campos y tipos de datos, permitirán estructurar la información de manera que el rendimiento sea óptimo. La Figura 7 presenta las entidades, atributos y relaciones que conformarán la base de datos del prototipo de aplicación Web.

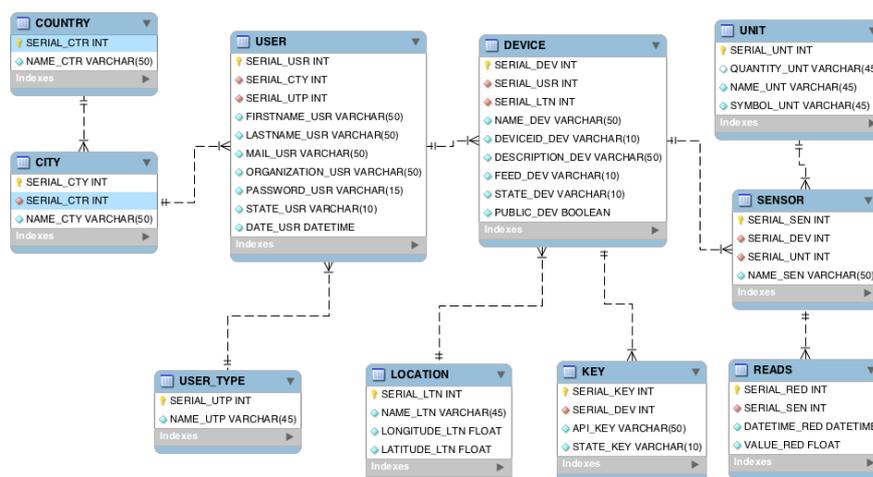


Figura 7 Diseño de la Base de Datos.

4.3 Diagramas de Casos de Uso

El objetivo de un diagrama de casos de uso es describir la manera en la que se utilizará un sistema (Kime1, 2006). La Figura 8 ilustra los casos de uso tanto del prototipo de aplicación Web, como de la placa Arduino o dispositivo electrónico. Será necesario descomponer cada uno de los casos de uso y detallarlos adecuadamente con el objetivo de entender mejor las funcionalidades y comportamiento que tendrán los distintos componentes que conforman el presente proyecto (Ricardo, 2009; IEEE, 2014).

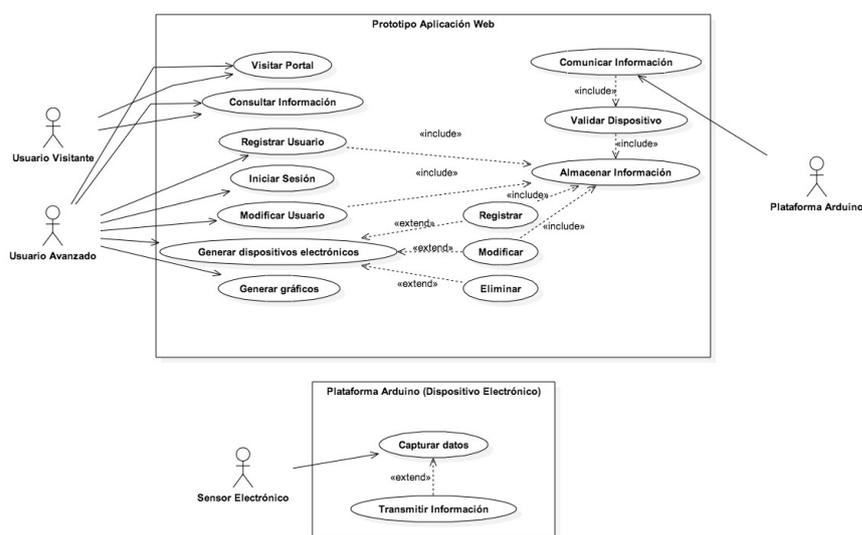


Figura 8 Casos de uso del proyecto.

En base a lo requerimientos especificados en el tercer capítulo, se identificó la necesidad de desarrollar un portal que pueda ser visitado tanto por el Usuario Visitante, como por el Usuario Avanzados. Estos dos actores podrán conocer la información generada, por los dispositivos electrónicos, y transmitida hacia el prototipo de aplicación Web. La Figura 9 agrupa los casos de uso “Visitar Portal” y “Consultar Información”.

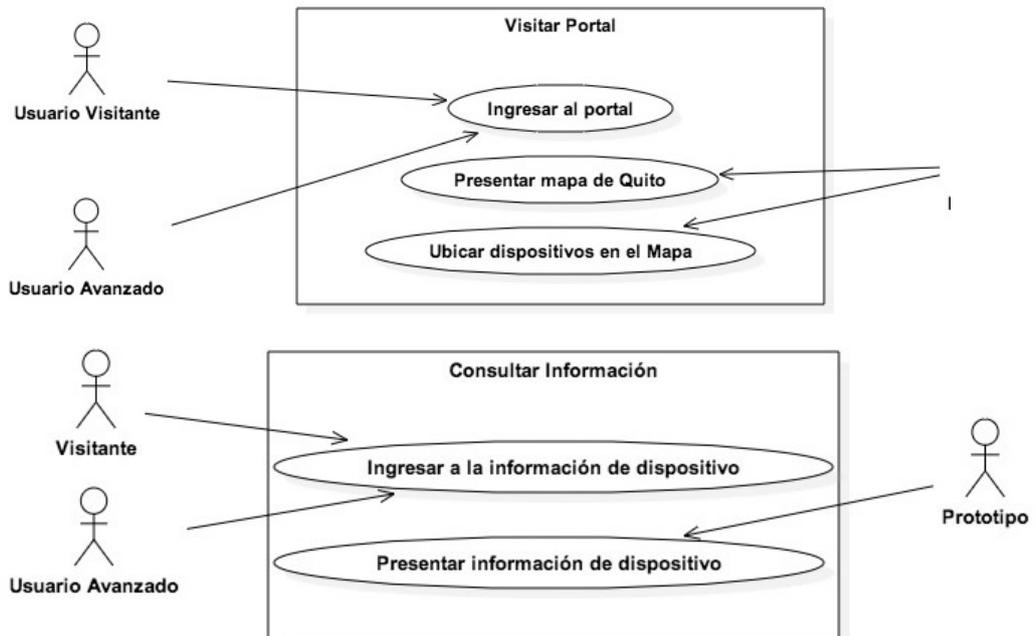


Figura 9 Casos de uso del portal de la aplicación Web.

El Usuario Avanzado deberá estar registrado en la aplicación Web para ingresar a la sección de gestión de los dispositivos electrónicos, un proceso de verificación de la información entregada será necesario para evitar duplicidad de usuarios. La Figura 10 presenta el caso de uso “Registrar Usuario”.

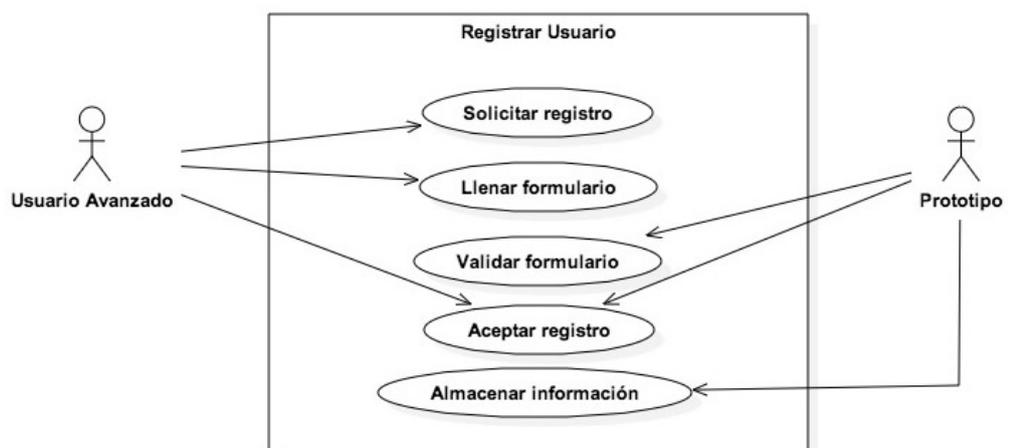


Figura 10 Caso de uso: Registrar Usuario.

Una vez completado el registro como Usuario Avanzado de la aplicación Web, este podrá registrar y configurar dispositivos electrónicos, modificar su información o eliminarlos. El acceso a estas funcionalidades se lo obtiene luego de una verificación de la identidad del Usuario Avanzado, este proceso corresponde al caso de uso: Iniciar sesión y que se detalla en la Figura 11.

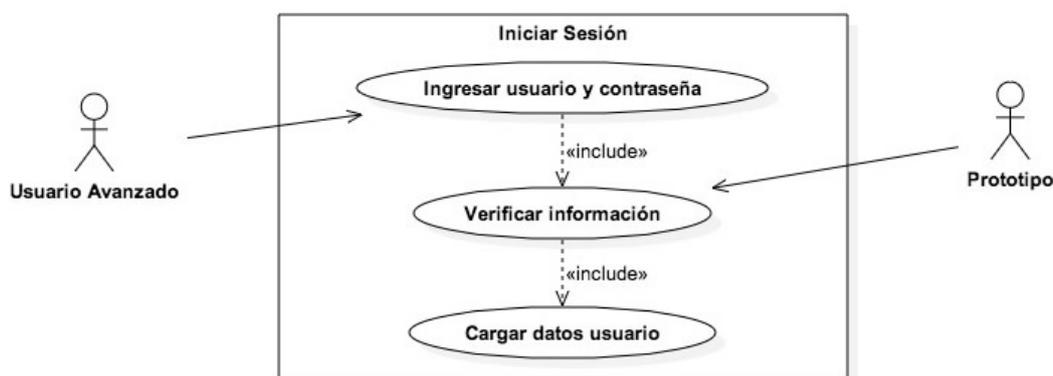


Figura 11 Caso de uso: Iniciar Sesión.

La Figura 12 muestra la descomposición del caso de uso “Registrar Nuevo Dispositivo”, proceso que es fundamental para lograr la interacción entre los dispositivos electrónicos y el prototipo de aplicación Web. Adicionalmente al registro de nuevos dispositivos, es necesario configurar su ubicación y los sensores electrónicos que componen el dispositivo electrónico. Toda esta información servirá como base para generar el mapa que se presentará a los Usuarios Visitantes.

Los procesos de actualización o modificación de las configuraciones de los dispositivos electrónicos, están separados en tres grupos de casos de uso y se encuentran ilustrados en la Figura 13. Se podrá modificar información como el nombre del dispositivo electrónico, su ubicación y la configuración de los sensores que lo componen. Cada modificación estará sujeta a un proceso de validación que lo ejecutará el prototipo de aplicación Web.

Eliminar un dispositivo electrónico sería el último caso de uso que restaría por abordar en lo que se refiere a la gestión de dispositivos electrónicos. La Figura 14 modela los casos de uso que componen a la eliminación de un dispositivo electrónico del prototipo de aplicación Web.

El sistema ofrecerá la posibilidad de generar un gráfico de la información recolectada por cada sensores electrónico, esta funcionalidad se detalla en el caso de uso “Generar Gráfico” y se encuentra ilustrado en la Figura 15. El acceso a la generación de gráficos estará disponible solo para los usuario avanzados por tratarse de un prototipo de aplicación Web. El gráfico presentará la concentración de contaminantes en función del tiempo.

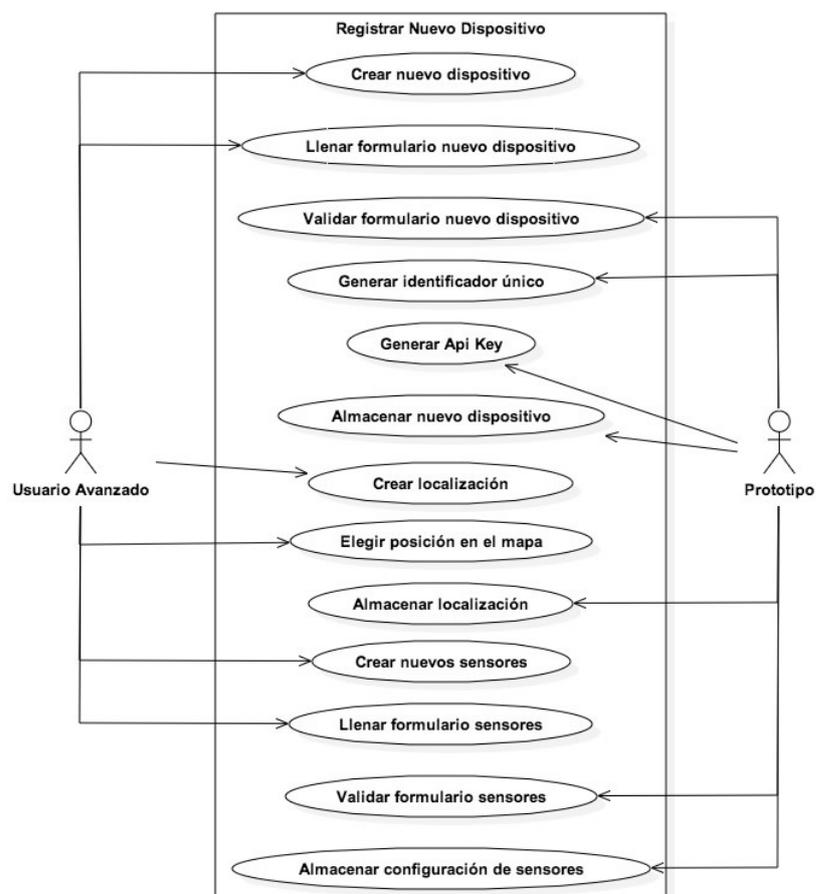


Figura 12 Caso de uso: Registrar Nuevo Dispositivo.

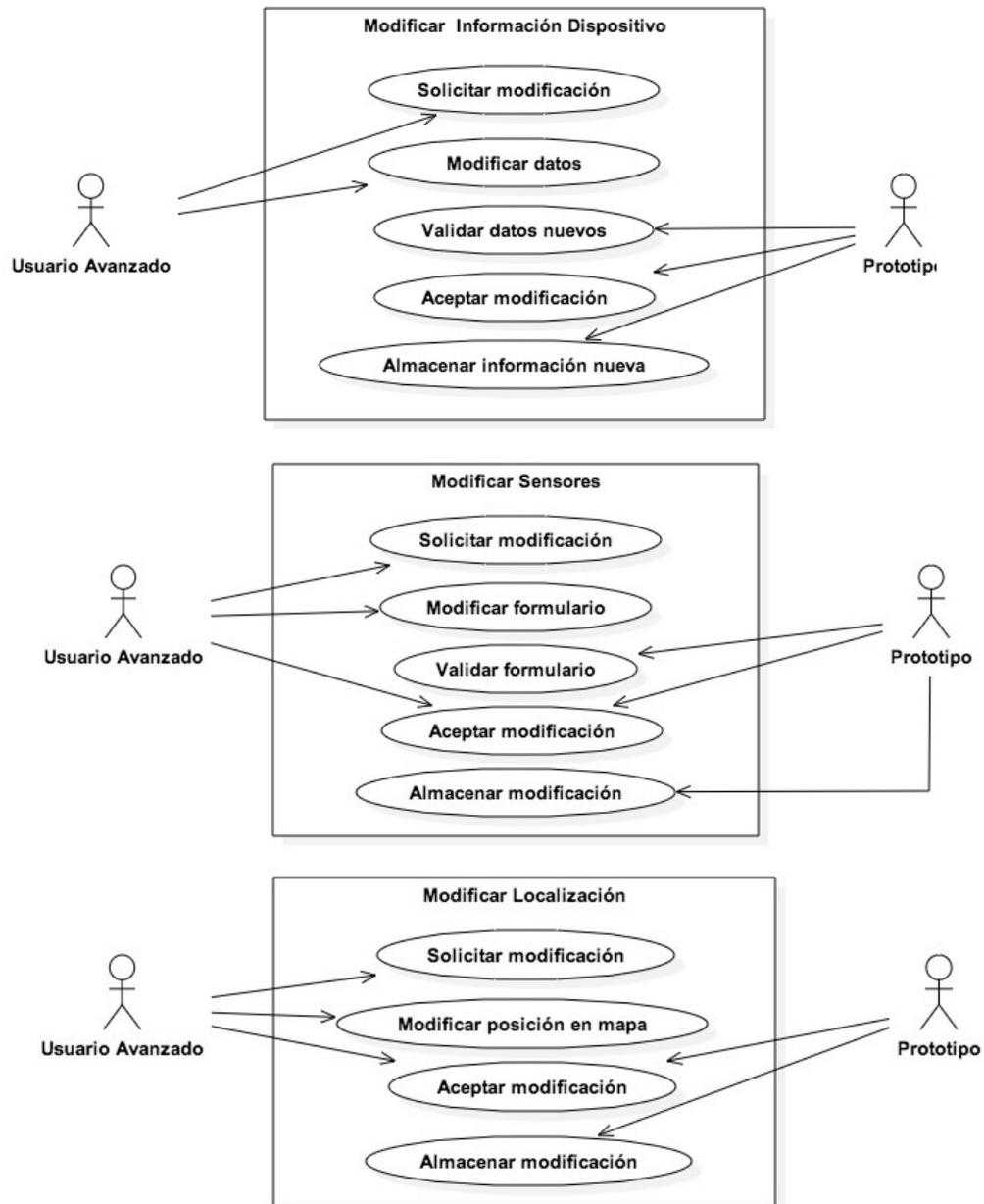


Figura 13 Casos de uso de modificación de información.

Los casos de uso de captura de datos y envío de información, desde los dispositivos electrónicos hacia el prototipo de aplicación Web, se detallan en la Figura 16 y están divididos en dos grupos “Comunicar Información” y “Capturar Datos”.

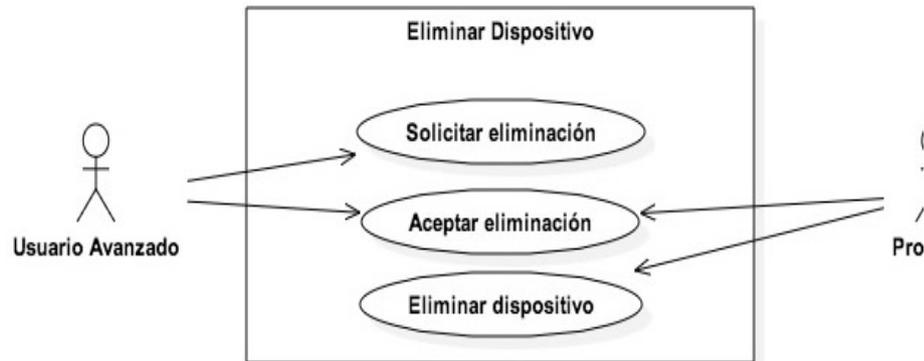


Figura 14 Caso de uso: Eliminar Dispositivo.

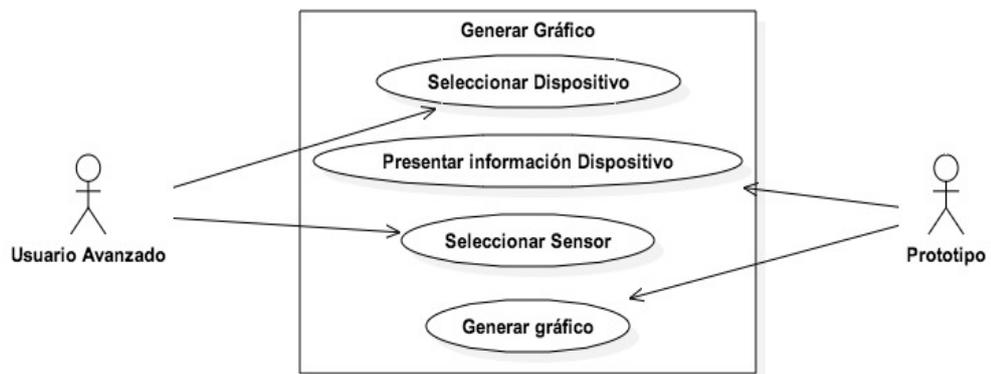


Figura 15 Caso de uso: Generar Gráfico.

4.4 Diagramas de Secuencia

Los diagramas mas comunes de interacción dentro de UML son los de secuencia y colaboración. Los diagramas de secuencia muestran como los objetos, de un sistema funcional, interactúan entre sí en el tiempo y están organizados linealmente (Kime1, 2006). De los diagramas de casos de uso elaborados anteriormente, se puede distinguir cinco actores que interactuarán en cada uno de los escenarios posibles. Estos escenarios pueden agrupar a dos o más casos de uso.

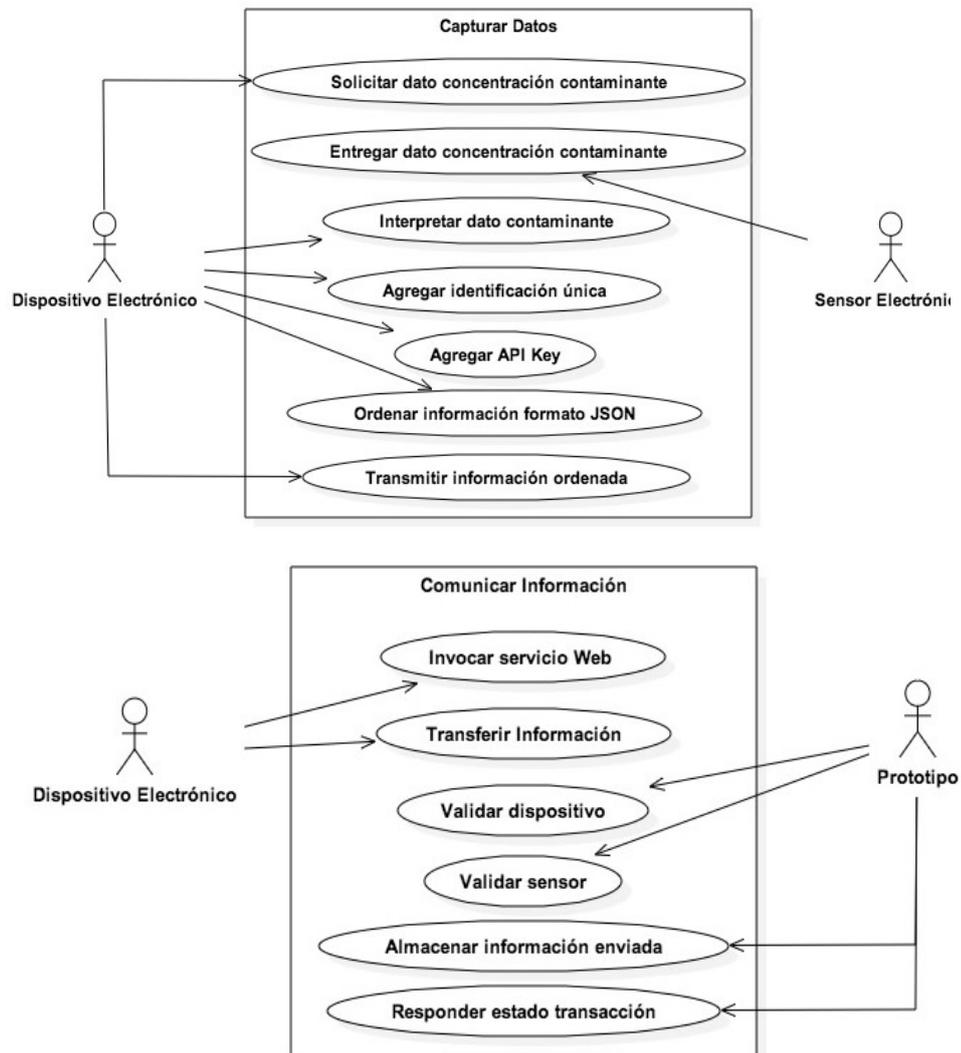


Figura 16 Casos de uso sobre captura y transmisión de datos.

El primer escenario está ilustrado en la Figura 17 y detalla la secuencia de mensajes que existirán cuando los usuarios visitantes y avanzados acceden al portal del prototipo de aplicación web para consultar la información generada por los dispositivos electrónicos. Se ha hecho hincapié en la necesidad de estar registrado como Usuario Avanzado para tener acceso a la sección de gestión de dispositivos electrónicos, la Figura 18 recoge la secuencia necesaria para cumplir con este requisito.

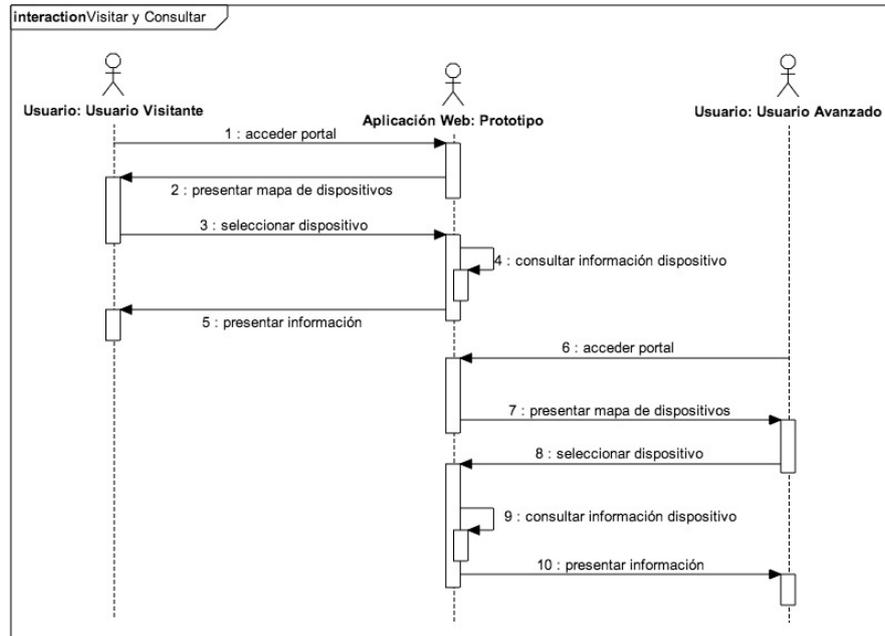


Figura 17 Escenario de visita y consulta de información.

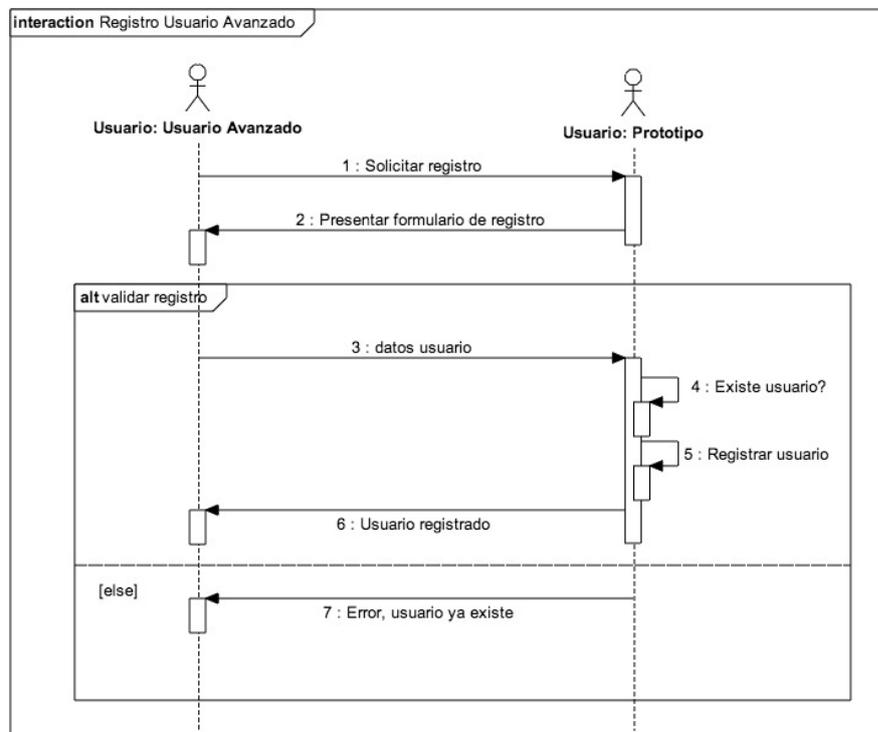


Figura 18 Escenario de registro de Usuario Avanzado.

La secuencia que valida las credenciales de un Usuario Avanzado se denomina “Iniciar de Sesión” e incluye un bloque de comprobación de la información entregada por el usuario. Este bloque se lo conoce como *marco de interacción o fragmento combinado*, son regiones rectangulares que se usan para organizar los diagramas de secuencia y pueden transmitir información específica. (KimeI, 2006)

La Figura 19 ilustra el escenario de inicio de sesión por parte de un Usuario Avanzado, previamente registrado. Es importante notar el marco de interacción que en este caso contendrá la lógica condicional encargada de validar la información entregada al prototipo Web.

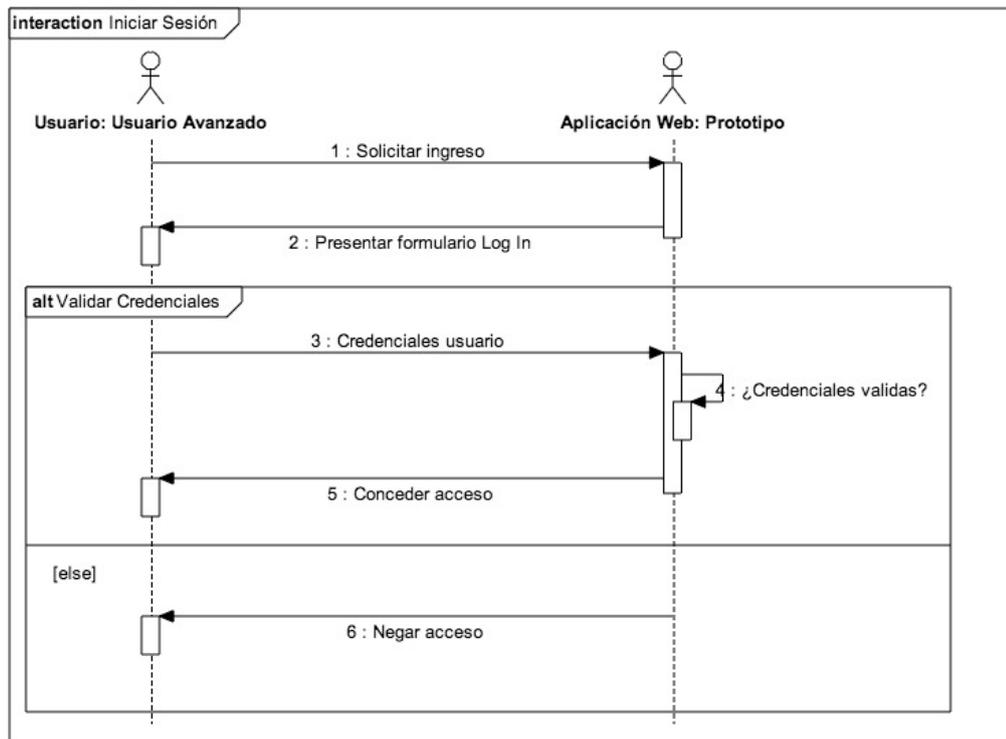


Figura 19 Escenario de inicio de sesión

Una vez cumplido exitosamente el escenario de inicio de sesión, el prototipo Web concederá el acceso a la sección de gestión de los dispositivos electrónicos. Estos dispositivos pueden contener uno o más

sensores que generen la información relacionada a la concentración de contaminantes ambientales.

El Usuario avanzado solo podrá gestionar los dispositivos electrónicos que sean registrados por él. El prototipo de aplicación Web generará un identificador único y una clave API para cada dispositivo registrado, esto con el fin de determinar y validar la procedencia de la información cuando los dispositivos comience a consumir el servicio web que se construirá.

El escenario de registro de un nuevo dispositivo se encuentra detallado en la Figura 20, contiene la secuencia necesaria para cumplir con este caso de uso.

Siempre podrá ser posible modificar las configuraciones iniciales con las que se registraron a un dispositivo electrónico. Sin embargo, información como el identificador único o la llave API no se podrá modificar.

Las configuraciones de sensores y localización del dispositivo serán perfectamente modificables, entendiendo que un dispositivo electrónico puede ser cambiado de ubicación o podrá incrementar o disminuir el número de magnitudes físicas o químicas capaz de capturar. El escenario de actualización de información de un dispositivo electrónico se presenta en la Figura 21.

El último escenario posible dentro de la gestión de dispositivos electrónicos es la eliminación y se encuentra ilustrado en la Figura 22. La secuencia de mensajes o acciones para generar un gráfico a partir de la información recolectada por los sensores electrónicos, se encuentra representada en la Figura 23.

El escenario de captura de datos y transmisión de la información cuenta como actores al prototipo de aplicación Web, al dispositivo electrónico que

se desarrollará bajo la plataforma Arduino y a los sensores electrónicos que entregarán las lecturas sobre la presencia de contaminantes en el ambiente.

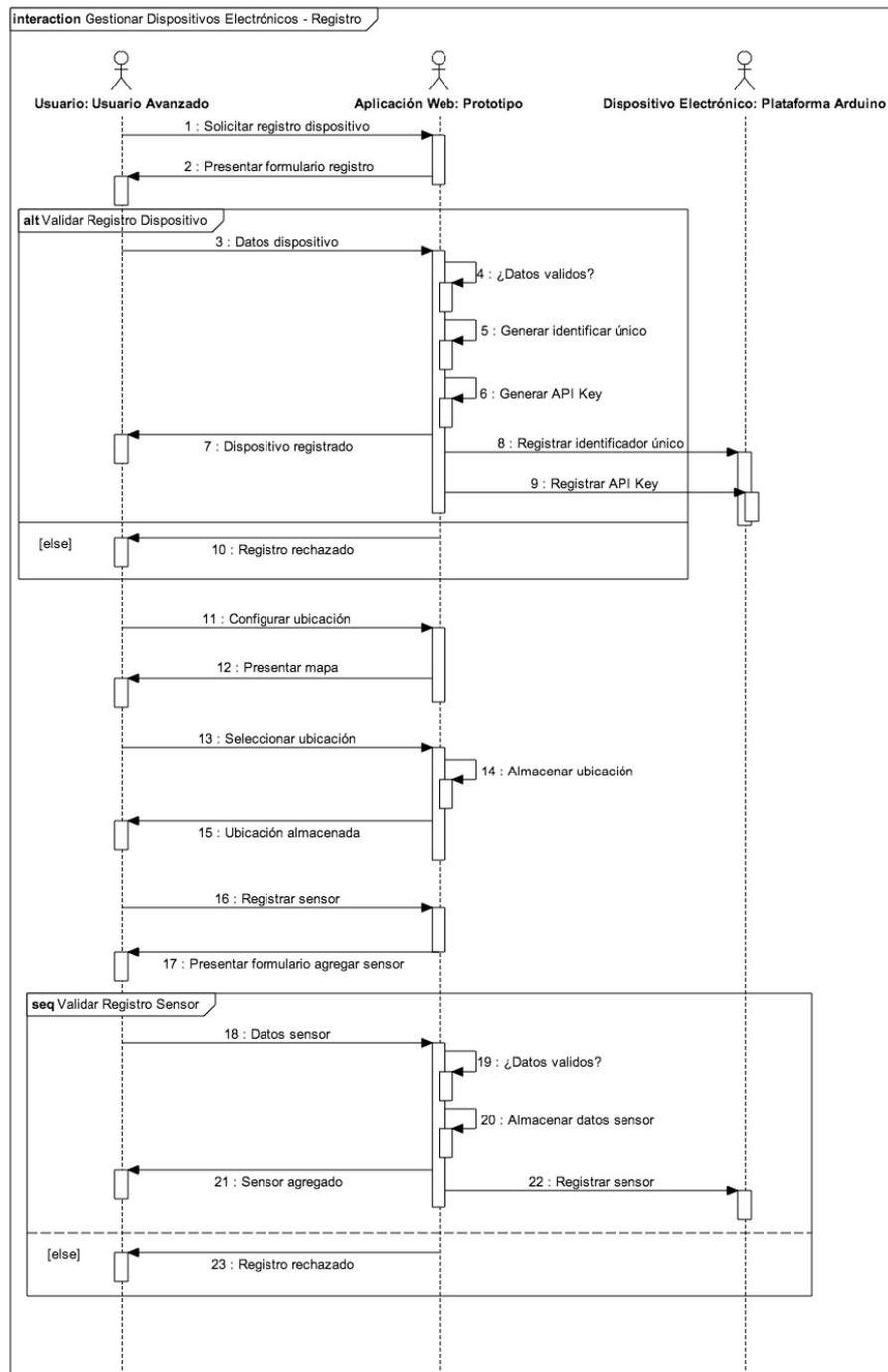


Figura 20 Escenario de registro de un nuevo dispositivo electrónico.

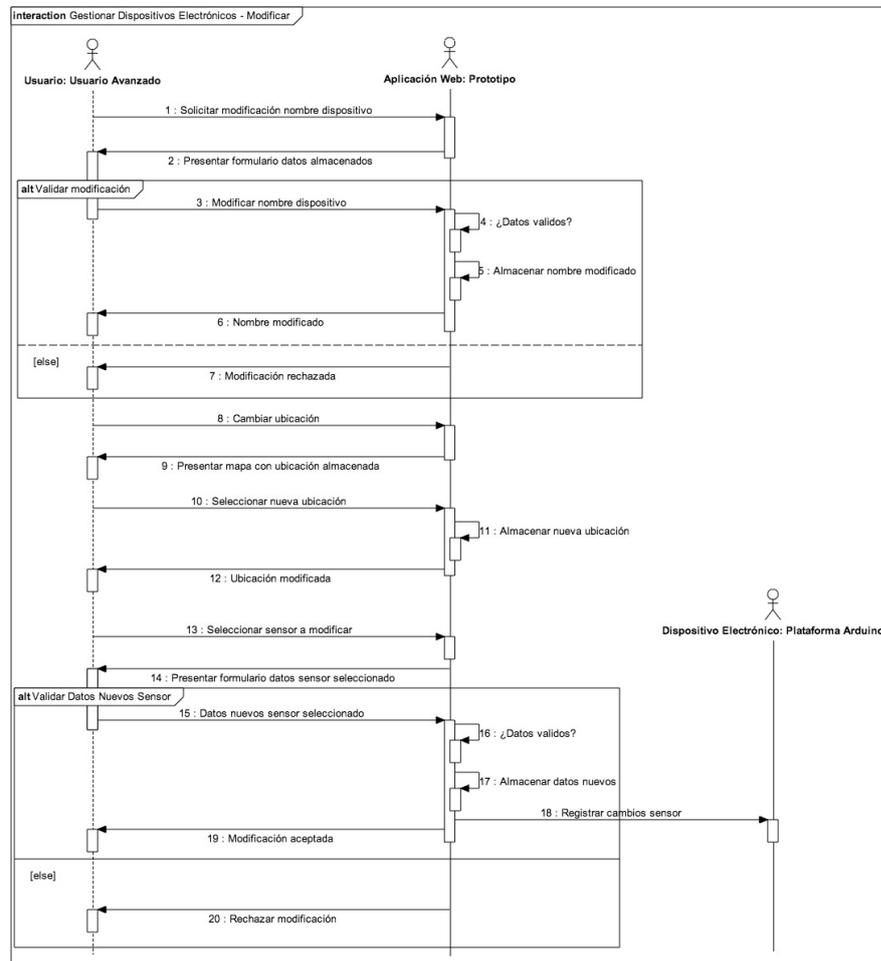


Figura 21 Escenario de actualización de información.

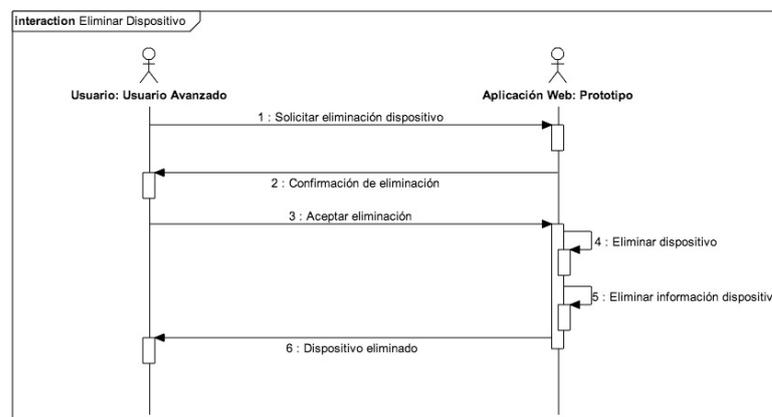


Figura 22 Escenario de eliminación de dispositivos electrónicos.

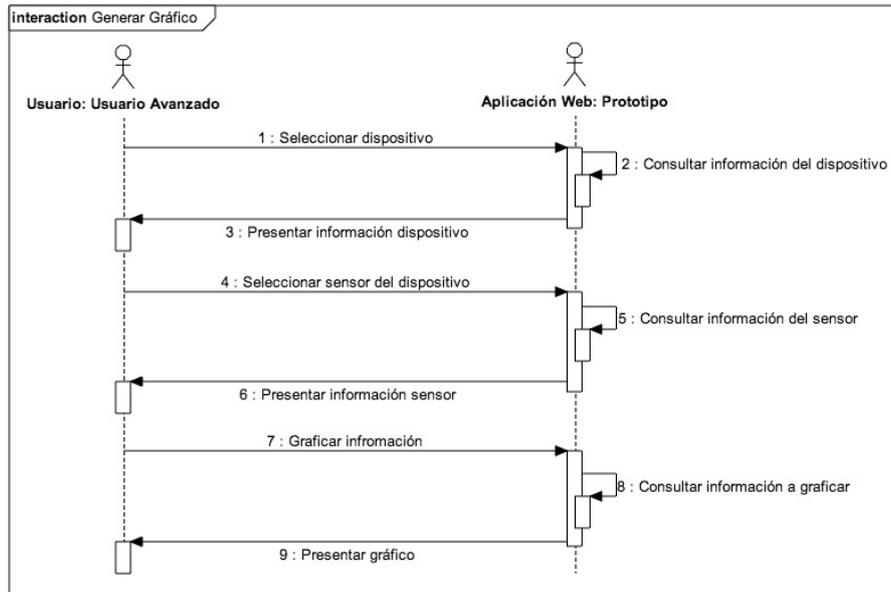


Figura 23 Generación de gráficos.

La Figura 24 ilustra la secuencia de acciones necesarias para conseguir recolectar, ordenar y transmitir esta información. Es importante señalar que este escenario agrupa varios casos de usos detallados anteriormente.

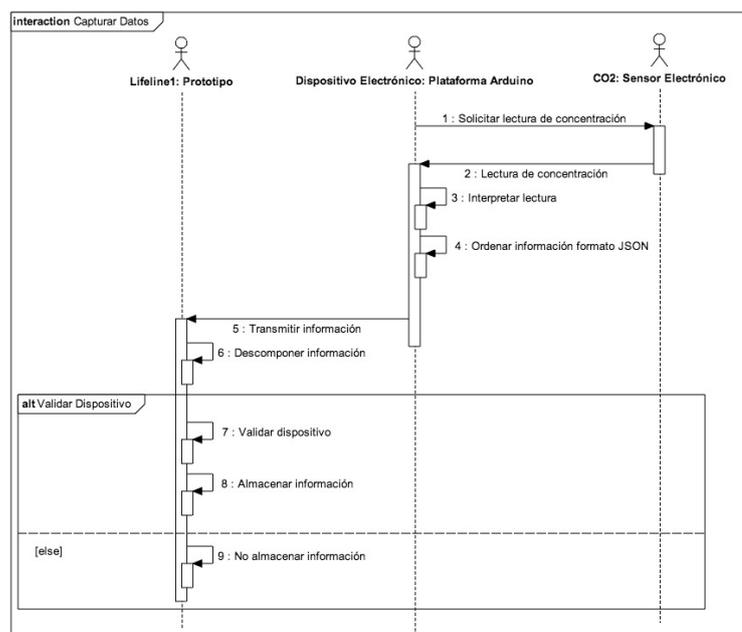


Figura 24 Secuencia para capturar, ordenar y transmitir información.

4.5 Diagramas de Clases

Los diagramas de clases están diseñados para mostrar todas las piezas de la solución y sus relaciones. Es la vista más importante en el diseño de un sistema y debe guardar sentido con las necesidades que se quieren atender. (Kimel, 2006)

El diseño de clases del prototipo de aplicación Web está dividido en cinco paquetes. Cada paquete contiene un conjunto de clases que brindarán una funcionalidad específica. El primer paquete que se ilustra en la Figura 25, contiene las clases entidad, son las que modelan la información que a menudo será persistida.

El segundo paquete de clases utiliza el patrón de diseño DAO (Data Acces Object por sus siglas en Inglés) y también se lo conoce como Servicios de Información Empresarial (EIS por sus siglas en Inglés). Este patrón define una interfaz y su respectiva implementación para ejecutar las operaciones más comunes sobre la base de datos en relación con las clases Entidad. La Figura 26 presenta el diseño de la clases DAO.

El tercer paquete de clases empleará un diseño similar al que se presenta en el segundo paquete, esto con el afán de conseguir un diseño orientado a objetos con alta cohesión entre clases y de bajo acoplamiento. La Figura 27 recoge el diseño de las clases de servicio.

Las clases controladoras conforman el cuarto paquete y serán las encargadas de manejar las peticiones que harán los usuarios a través de las vista del prototipo de aplicación Web. Estas clase están asociadas a las clases de servicios que a su vez se encuentran asociadas a las clases DAO presentadas en los diagramas anteriores.

Visual Paradigm for UML Community Edition [not for commercial use]

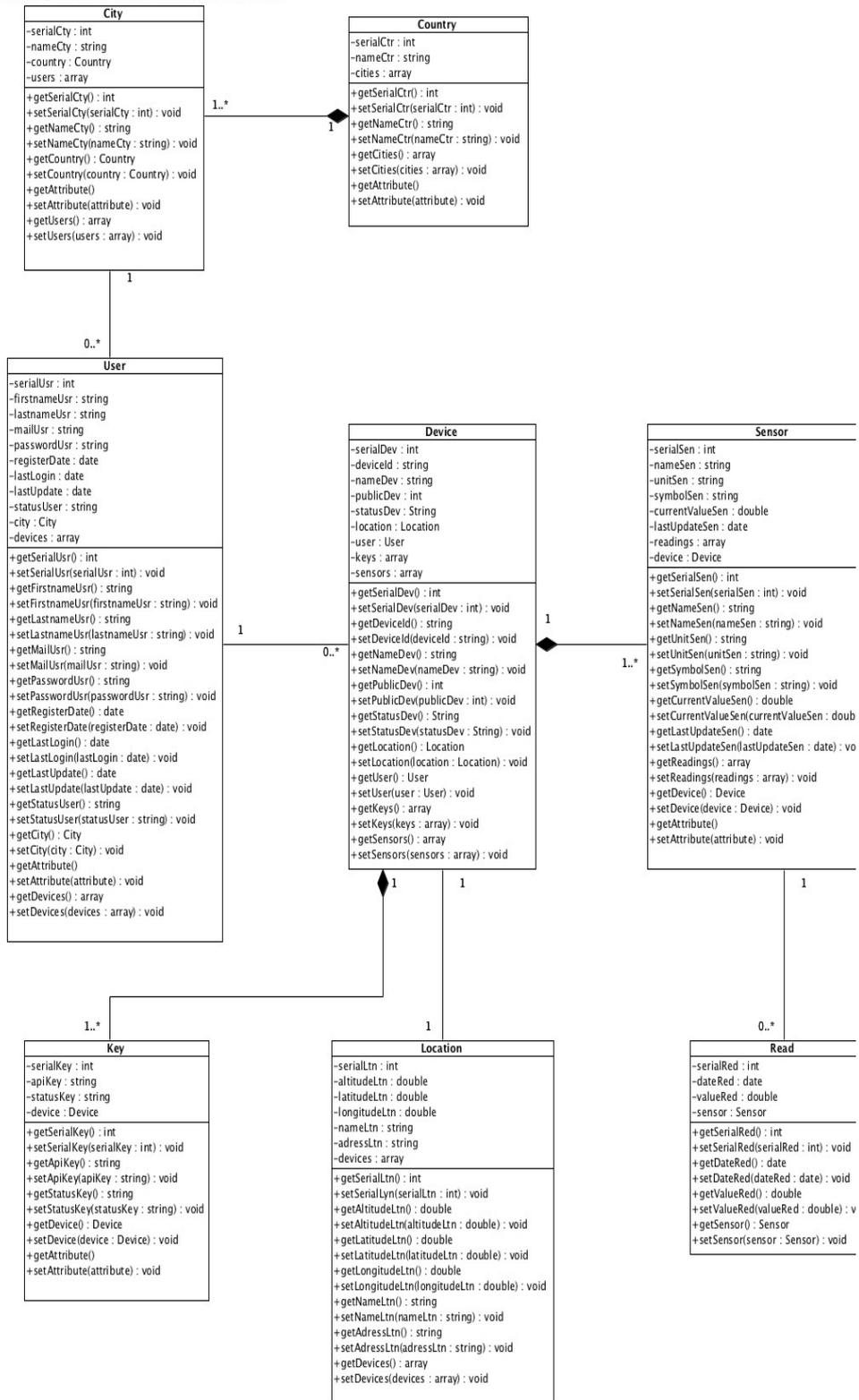


Figura 25 Diagrama de clases entidad.

Visual Paradigm for UML Community Edition [not for commercial use]

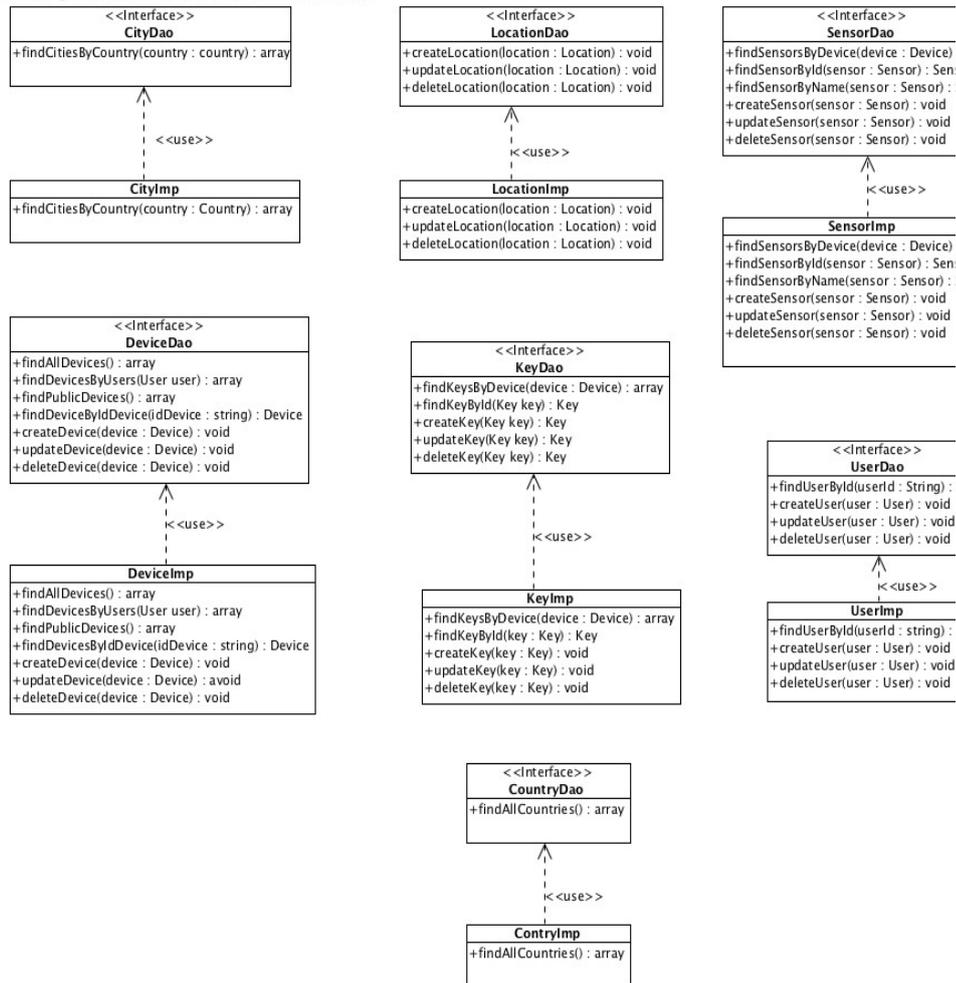


Figura 26 Diseño de clases DAO.

Todo este conjunto de relaciones y asociaciones entre clases permitirán implementar los diferentes escenarios presentados en los diagramas de secuencia. La Figura 28 muestra el diagrama de clases controladoras.

El quinto paquete se refiere a las clases que harán posible la comunicación por medio de un servicio Web RESTful, La Figura 29 muestra el diagrama de clases y sus relaciones que manejará el Servicio Web.

Visual Paradigm for UML Community Edition [not for commercial use]

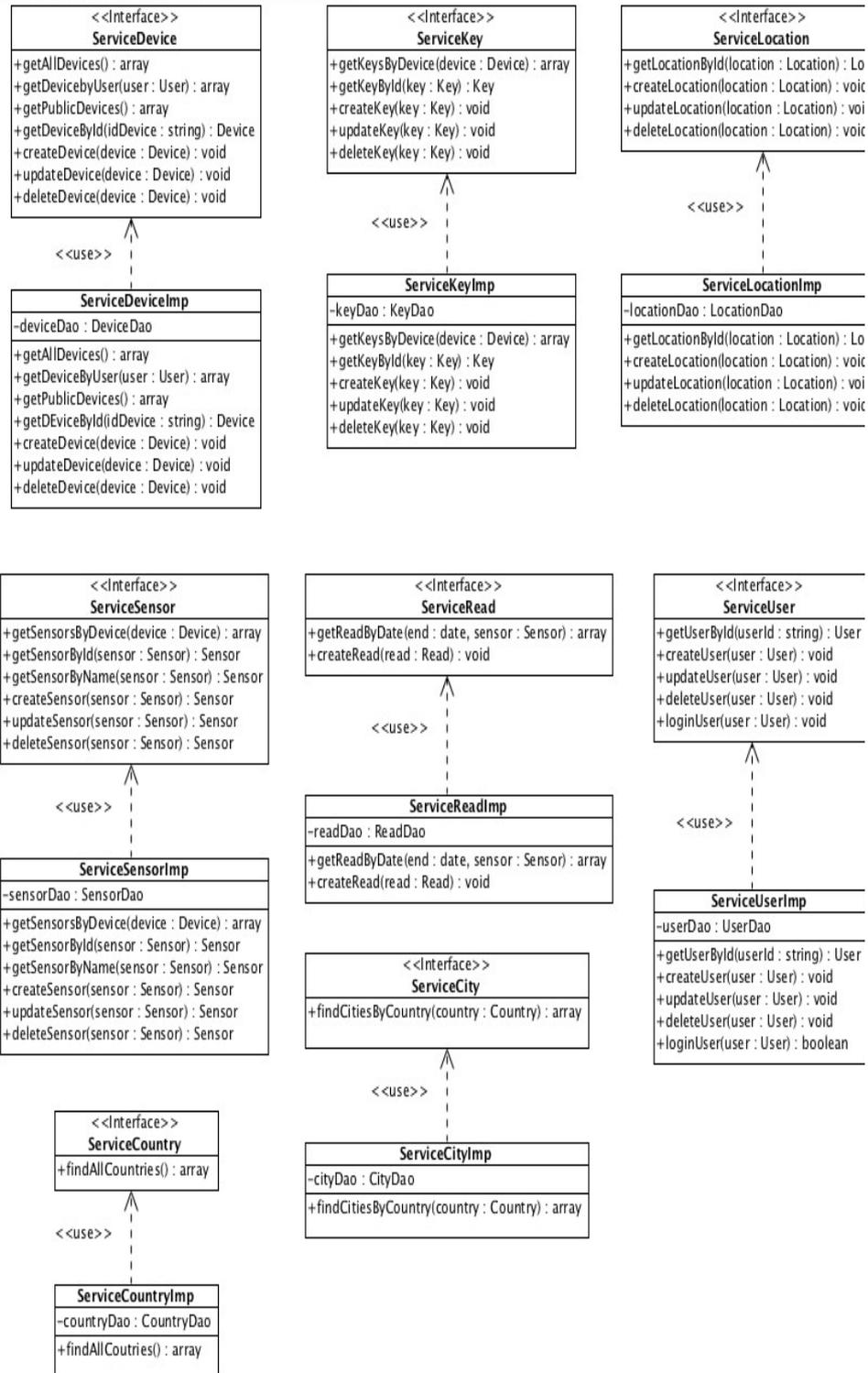


Figura 27 Diseño de clases de servicios

Visual Paradigm for UML Community Edition [not for commercial use]

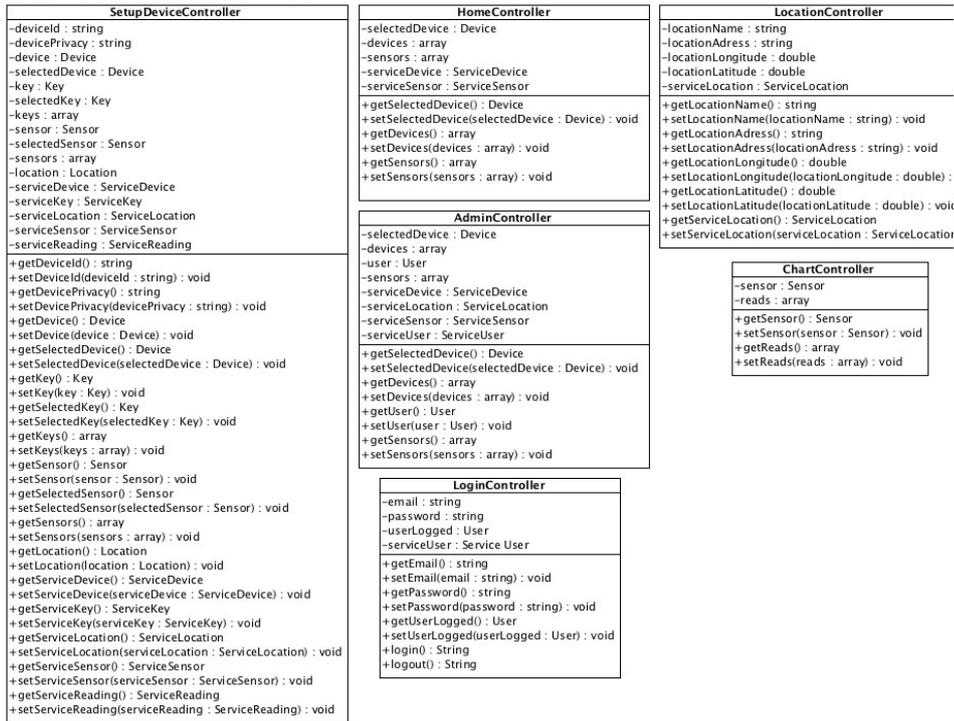


Figura 28 Diagrama de clases de control

Visual Paradigm for UML Community Edition [not for commercial use]

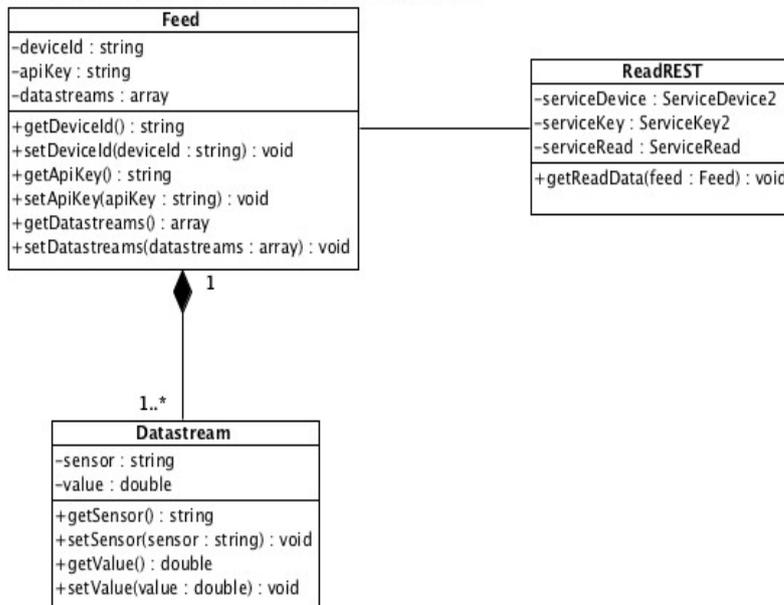


Figura 29 Diagrama de clases del Servicio Web.

4.6 Diagrama de Paquetes

Para entender mejor la relación entre los diferentes paquete de clases podemos recurrir a un diagrama de paquetes. Estos diagramas representan un nivel de abstracción superior al diagrama de clases. La Figura 30 presenta el diagrama de paquetes del prototipo de aplicación Web.

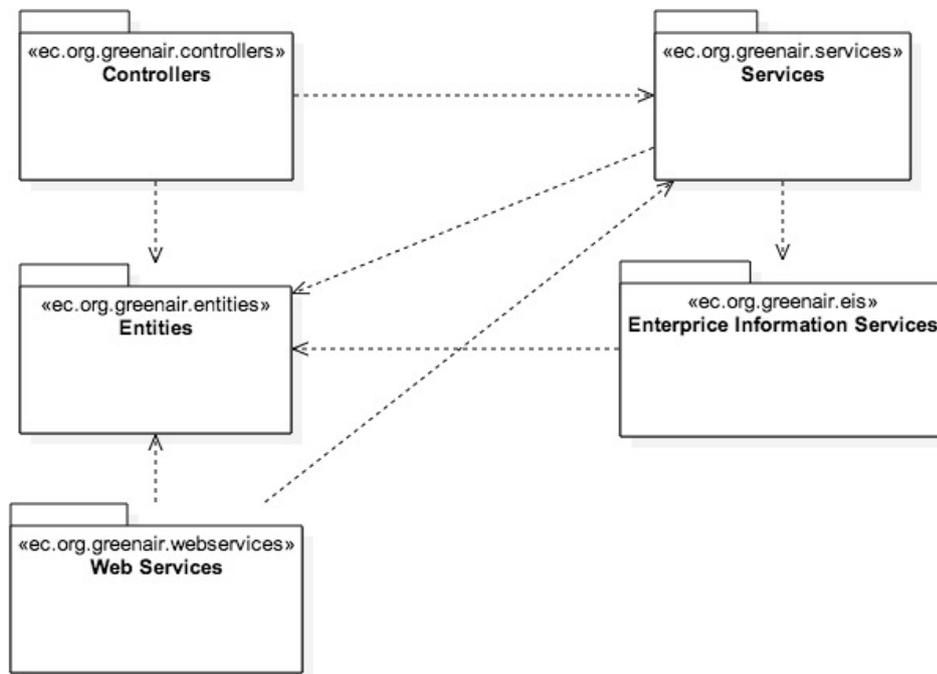


Figura 30 Diagrama de paquetes.

4.7 Diagrama de Arquitectura

La Figura 31 ilustra la arquitectura del software, donde se puede observar como se encuentran relacionadas las diferentes capas con las que contará la el prototipo de aplicación Web. También se señala que capas utilizarán los dos tipos de usuarios, anteriormente identificados, y el dispositivo electrónico.

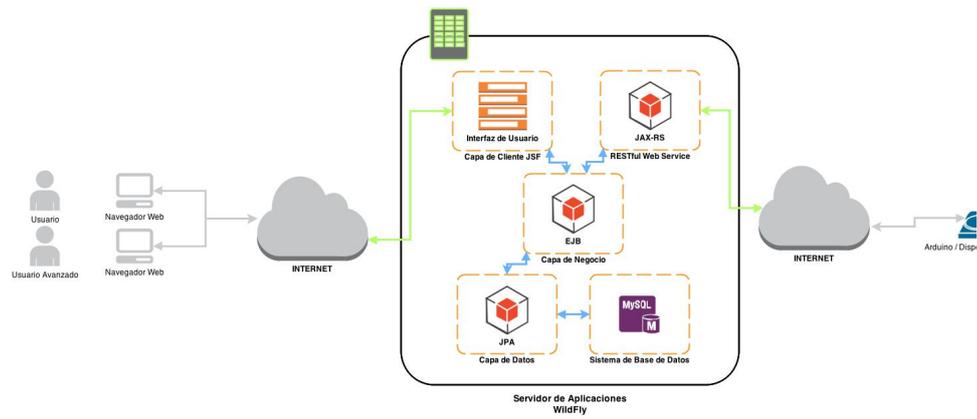


Figura 31 Arquitectura de software.

PLANIFICACIÓN, DESARROLLO Y PRUEBAS DEL PROTOTIPO

Para la planificación, desarrollo y evaluación de este proyecto se empleará una combinación de las metodologías Scrum y Extreme Programming (XP). Scrum abordará la manera en la que se organizará el trabajo mientras que Extreme Programming se utilizará para la codificación de software.

5.1 Planificación del proyecto utilizando la metodología Scrum.

El punto de partida, de acuerdo a lo que menciona la metodología Scrum, es la construcción de la pila de producto o Product Backlog y que se hará en base a los requerimientos expuestos en el tercer capítulo del presente proyecto.

Es necesario definir un formato del documento, que se obtendrá del proceso de construcción de la pila del producto, para que pueda ser fácilmente entendido. Los campos elegidos para identificar a cada ítem del Product Backlog son:

- **ID:** Identificador único del ítem del listado.
- **Historia de usuario:** Descripción general del requerimiento
- **Estimación:** Tiempo necesario
- **Prioridad:** Es el grado de importancia que tendrá el ítem en relación al resto. Esto será definido por el propietario del producto o Product Owner.
- **Criterio aceptación:** Descripción de la funcionalidad que se espera tenga el demo del producto.

El Cuadro 15 detalla los once requerimientos o historias de usuario generales iniciales que fueron identificados en el tercer capítulo de este proyecto, cada historia engloba una serie de tareas que serán abordadas en la etapa de iteraciones sucesivas o Sprints y que se documentarán en la pila de tareas o Sprint Backlog. Cada tarea deberá estar priorizada y debidamente estimada en relación al tiempo que se necesitará para completarla.

La metodología también señala que la pila de producto o Product Backlog deberá ser refinada para facilitar la planificación de las iteraciones sucesivas. Para cumplir con esta directriz, se ha definido una escala tanto para el tiempo estimado que tomará elaborar una historia de usuario, como para la prioridad que cada historia tendrá en el desarrollo del prototipo de aplicación Web. Las escalas que se emplearán son:

- Escala tiempo estimado: 1 a 4 semanas.
- Escala de prioridad: 1 a 10, siendo 10 la menos importante.

Este proceso de refinamiento de la pila del producto o Product Backlog contempla la aplicación de varios criterios lo que ayudarán a ordenar y priorizar las historias de usuarios. Los criterios aplicables son:

- Cada ítem debe representar un incremento significativo a la funcionalidad del prototipo.
- El equipo de desarrollo debe ser capaz de construir cada ítem en un solo Sprint.
- Todos los participantes en el proyecto deben entender lo que se va a desarrollar para cada requerimiento.

Una vez organizado y priorizado el Product Backlog, se iniciará la etapa de iteraciones sucesivas o Sprints. En esta etapa se llevará a cabo la actividad de planificación o Sprint Planning, donde los miembros del equipo

de desarrollo escogen una o un grupo de historias de usuario del Product Backlog para desarrollarlo en una sola iteración. A este nuevo listado de tareas se lo conoce como pila de tareas o Sprint Backlog.

5.2 Estándares de programación.

El empleo de estándares de programación o convenciones de código son importantes por un algunas razones. El 80% del costo de un proyecto de software corresponde al mantenimiento de la aplicación, una tarea que en pocas ocasiones es realizada por la persona que lo creó. Las convenciones de código mejoran la lectura del software, permitiendo a los programadores entenderlo y modificarlo en el caso que se quiere implementar nuevas funcionalidades o cubrir nuevos requerimientos. Si la entrega del código fuente de la aplicación forma parte del producto final, este debe estar correctamente empaquetado y ser lo suficientemente claro (Sun Microsystems, Oracle, 1997).

En tal virtud, la codificación del prototipo del lado del servidor empleará las convenciones de código para el lenguaje Java, publicadas por Sun Microsystems el 12 de septiembre de 1997. El Cuadro 16 recoge las convenciones de código que se utilizarán.

Para la construcción de las interfaces de usuario se planificó emplear el framework Java Server Faces (JSF), el que está constituido por una serie de etiquetas extensible (Hypertext Markup Language, XHTML); este conjunto de tecnologías se complementará con el uso de las librerías Bootstrap y JQuery compatibles con el estándar HTML 5 para mejorar la experiencia del usuario al momento de utilizar la aplicación. El Cuadro 17 detalla cada una de las etiquetas HTML y XHTML que se utilizarán en la construcción de la capa de cliente del prototipo de aplicación Web.

Pila de producto o Product Backlog

ID	Historia de usuario	Estimación (semanas)	Prioridad	Criterio de aceptación
1	Registrar usuarios avanzados	2	1	Seleccionar la opción: Regístrate, de la portada de la aplicación, desplegar el formulario de registro, ingresar la información solicitada, validar los datos ingresados y comunicar el registro exitoso o fallido del nuevo usuario avanzado. Verificar en el tracking del servidor de aplicaciones.
2	Iniciar sesión	2	2	Seleccionar la opción: Iniciar Sesión, de la portada de la aplicación, ingresar las credenciales del usuario. Si las credenciales son correctas, permitir el acceso a la gestión de dispositivos electrónicos, caso contrario, comunicar el mensaje de error de inicio de sesión. Verificar en el tracking del servidor de aplicaciones
3	Modificar usuarios avanzados	2	3	Ingresar al prototipo Web, escoger la opción: Perfil, modificar la información deseada, validar los datos nuevos ingresados y comunicar la actualización correcta o fallida. Verificar en el tracking del servidor de aplicaciones.
4	Gestión de dispositivos electrónicos – Agregar dispositivos	2	4	Ingresar al prototipo Web, seleccionar la opción: Agregar Dispositivo, desplegar el formulario de registro de dispositivo, llenar el formulario, validar la información ingresada. Luego desplegar el formulario de configuración de dispositivo, elegir la ubicación y configuración de sensores, validar información y almacenarla.

ID	Historia de usuario	Estimación (semanas)	Prioridad	Criterio de aceptación
5	Gestión de dispositivos electrónicos – Editar configuraciones	2	4	Ingresar al prototipo Web, seleccionar la opción: Editar Dispositivo, desplegar el formulario de configuración de dispositivo electrónico, modificar las configuraciones necesarios, validar la nueva información ingresada y almacenarla. Verificar en el tracking del servidor de aplicaciones.
6	Capturar y procesar lecturas	2	5	Ingresar a la plataforma de desarrollo de Arduino, conectar el dispositivo electrónico por medio del puerto USB y comprobar la información captada por los sensores electrónicos.
7	Transmitir información	1	6	Ingresar a la plataforma de desarrollo de Arduino, conectar el dispositivo electrónico con la computadora por medio del puerto USB y comprobar la conexión entre la placa Arduino y la red inalámbrica. Verificar el código HTTP de respuesta del servidor de aplicaciones.
8	Autenticar dispositivos electrónicos	1	7	Ingresar a la aplicación HttpRequister, complemento del navegador Web Firefox, comprobar el funcionamiento del servicio Web RESTful del prototipo de aplicación Web. Verificar en el tracking del servidor de aplicaciones.
9	Almacenar información	1	8	Ingresar a la aplicación MySQL Workbench, comprobar la cantidad de lecturas almacenadas en la base de datos y que fueron procesadas por el Servicio Web del prototipo. Verificar en el tracking del servidor de aplicaciones las sentencias ejecutadas.

Continúa ➡

ID	Historia de usuario	Estimación (semanas)	Prioridad	Criterio de aceptación
10	Generar mapa y gráficos	1	9	Ingresar al prototipo Web y comprobar la ubicación de los dispositivos electrónicos, pertenecientes al usuario que los registró, en el mapa que se despliega en la página inicial de la gestión de dispositivos. Para la generación de gráficos, escoger la opción de edición de dispositivos, seleccionar el icono de generación de gráfico dispuesto para cada sensor y comprobar el gráfico de la información de las lecturas recogidas en función del tiempo.
11	Consultar Información	1	10	Ingresar al portal del prototipo de aplicación Web y verificar la ubicación de los dispositivos electrónico, que están compartiendo su información públicamente, en el mapa que se desplegará. Seleccionar un dispositivo y comprobar la información, en tiempo real, de cada uno de los sensores con los que está equipado un dispositivo electrónico.

Cuadro 16

Convenciones de código

Elementos	Reglas	Ejemplos
Paquetes	El nombre de los paquetes deben estar escritos en minúsculas para evitar conflictos con el nombre de las clases. Se puede utilizar el nombre del dominio de Internet en el que funcionará la aplicación.	ec.org.greenair.entities ec.org.greenair.ies ec.org.greenair.services ec.org.greenair.controllers
Clases	El nombre de las clases serán sustantivos y la primera letra se escribirá en mayúsculas. En el caso de nombres compuestos la primera letra de cada palabra se escribirá en mayúsculas.	User Device Read Location
Interfaces	El nombre de las interfaces siguen la misma regla de la clases	UserDao DeviceDao ReadDao LocationDao
Métodos	Los métodos serán verbos escritos en minúsculas, en el caso de ser compuestos, la primera letra de la segunda palabra deberá ser mayúscula.	findAllDevices() createUser() updateUser()
Variables	Los nombres de las variables deben ser cortos y con significado, se escribirán en minúsculas y en el caso de ser compuestas se seguirá la regla de nombres de los métodos.	serialUsr serialDevice serialLocation

Fuente: Adaptado, (Sun Microsystems, Oracle, 1997)

Cuadro 17

Etiquetas XHTML

Etiquetas	Tipo Etiqueta	Descripción
h:head	XHTML	Etiqueta para delimitar el espacio de head en un documento XHTML.
h:body	XHTML	Etiqueta para delimitar el espacio body en un documento XHTML.
h:form	XHTML	Etiqueta para crear un formulario dentro de un documento XHTML.
input	HTML	Etiqueta simple de ingreso de texto.
h:commandButton	XHTML	Etiqueta para botones de eventos.
f:ajax	XHTML	Etiqueta que habilita el procesamiento de información con AJAX.
p:gmap	XHTML	Etiqueta de Primefaces para crear mapas con la API de Google Maps
p:gmapInfoWindow	XHTML	Etiqueta de Primefaces para generar ventanas desplegables sobre mapas.
ui:repeat	XHTML	Etiqueta para crear un componente de repetición personalizado en JSF.
label	HTML	Etiqueta simple de salida de texto
h:commandLink	XHTML	Etiqueta de un enlace asociado a un botón
h:link	XHTML	Etiqueta de enlace simple

Fuente: Adaptado, (Sun Microsystems, Oracle, 1997)

5.3 Desarrollo y pruebas de la Iteración 1

Las tres primeras historias de usuario han sido escogidas para ser abordadas en esta primera iteración que tendrá una duración de 10 días laborables, es decir, 2 semanas de trabajo. El Cuadro 18 recoge las historias de usuario seleccionadas y que contemplan la implementación de las funcionalidades del registro de usuarios, inicio de sesión y del mantenimiento de la información del usuario registrado

Cuadro 18

Historias de usuario seleccionadas para la iteración 1

ID	Historia de usuario	Prioridad	Criterio de aceptación
1	Registrar usuario avanzado	1	Seleccionar la opción: Regístrate, de la portada de la aplicación, desplegar el formulario de registro, ingresar la información solicitada, validar los datos ingresados y comunicar el registro exitoso o fallido del nuevo usuario avanzado. Verificar en el tracking del servidor de aplicaciones.
2	Iniciar sesión	1	Seleccionar la opción: Iniciar Sesión, de la portada de la aplicación, ingresar las credenciales del usuario. Si las credenciales son correctas, permitir el acceso a la gestión de dispositivos electrónicos, caso contrario, comunicar el mensaje de error de inicio de sesión. Verificar en el tracking del servidor de aplicaciones
3	Modificar usuario avanzado	1	Ingresar al prototipo Web, escoger la opción: Perfil, modificar la información deseada, validar los datos nuevos ingresados y comunicar la actualización correcta o fallida. Verificar en el tracking del servidor de aplicaciones.

5.4 Sprint Backlog de la iteración 1

Es necesario determinar las tareas que se deberán cumplir para lograr atender las historias de usuario seleccionadas, de la pila de producto, para esta primera iteración y que serán detalladas en el Sprint Backlog. El Cuadro 19 describe el conjunto de tareas, su responsable, el elemento del

Product Backlog al que pertenece y la fecha de entrega para cada uno de los requerimientos seleccionados.

5.4.4 Revisión y seguimiento de la iteración 1

Para la revisión del Sprint es necesario listar las tareas completadas y pendientes de cada desarrollador, esto con el objetivo de medir el avance en las historias de usuario seleccionadas. El Cuadro 20 detalla las tareas completadas en el Sprint.

El gráfico Burn-Down Chart se emplea como herramienta de seguimiento para medir el avance del proyecto en el Sprint en ejecución. Se cuantifica el número de tareas del Sprint Backlog en relación al tiempo de entrega planificado para determinar el grado de cumplimiento del cronograma. La Figura 32 ilustra el Burn-Down Chart de la primera iteración o Sprint 1.

5.4.5 Pruebas de la iteración 1

En el segundo capítulo se señalaron las prácticas que se tomarán de la metodología Extreme Programming. El diseño de las pruebas unitarias antes de codificar se constituyen en una actividad fundamental en el desarrollo del software. La Figura 34 muestra las pruebas unitarias que se le han aplicado a las clases relacionadas con el manejo de los usuarios.

5.4.6 Demo Iteración 1

El demo de la primera iteración presenta las ventanas de registro de usuarios avanzados, la funcionalidad de inicio de sesión y el mantenimiento de la información de los usuarios previamente registrados. También se definirá el diseño gráfico base del prototipo de aplicación Web. La Figura 33 muestra los detalles del demo.

Cuadro 19

Sprint Backlog de la iteración 1

ID	Tarea	Inicio		Duración (semanas)	Responsable	Product Backlog	30	
		1	30-Junio-2014				4	7-11
1	Implantación del modelo de datos diseñado			2	Diego Carrera	1	X	
2	Diseño de la prueba unitaria para el manejo de usuarios.				Diego Carrera	1	X	
3	Codificación de las clases entidad (JPA) en base al modelo de datos				Diego Carrera	1	X	
4	Codificación de las interfaces DAO y su implementación				Diego Carrera	1	X	
5	Codificación de las interfaces de servicio y su implementación				Diego Carrera	1	X	
6	Diseño de la interfaz de usuario				Diego Carrera	1	X	
7	Diseño del formulario de registro				Diego Carrera	1		X
8	Codificación del controlador del registro de usuario.				Diego Carrera	1		X
9	Diseño del formulario de inicio de sesión				Diego Carrera	2		X
10	Codificación del controlador del inicio de sesión				Diego Carrera	2		X
11	Diseño del formulario de mantenimiento de usuario				Diego Carrera	3		X
12	Codificación del controlador del mantenimiento de usuarios				Diego Carrera	3		X

Cuadro 20

Tareas completadas de la iteración 1

Tareas asignadas a: Diego Carrera		Número de tareas: 12			
ID	Tarea	Iteración	Estado	Fecha	Horas
1	Implantación del modelo de datos diseñado	1	Completada	30 - Junio - 2014	8
2	Diseño de la prueba unitaria para el manejo de usuarios.	1	Completada	1 - Julio - 2014	8
3	Codificación de las clases entidad (JPA) en base al modelo de datos	1	Completada	2 - Julio - 2014	8
4	Codificación de las interfaces DAO y su implementación	1	Completada	3 - Julio - 2014	8
5	Codificación de las interfaces de servicio y su implementación	1	Completada	4 - Julio - 2014	8
6	Diseño de la interfaz de usuario	1	Completada	7 - Julio - 2014	8
7	Diseño del formulario de registro	1	Completada	8 - Julio - 2014	4
8	Codificación del controlador del registro de usuario.	1	Completada	8 - Julio - 2014	4
9	Diseño del formulario de inicio de sesión	1	Completada	9 - Julio - 2014	4
10	Codificación del controlador del inicio de sesión	1	Completada	9 - Julio - 2014	4
11	Diseño del formulario de mantenimiento de usuario	1	Completada	10 - Julio - 2014	8
12	Codificación del controlador del mantenimiento de usuarios	1	Completada	11 - Julio - 2014	8

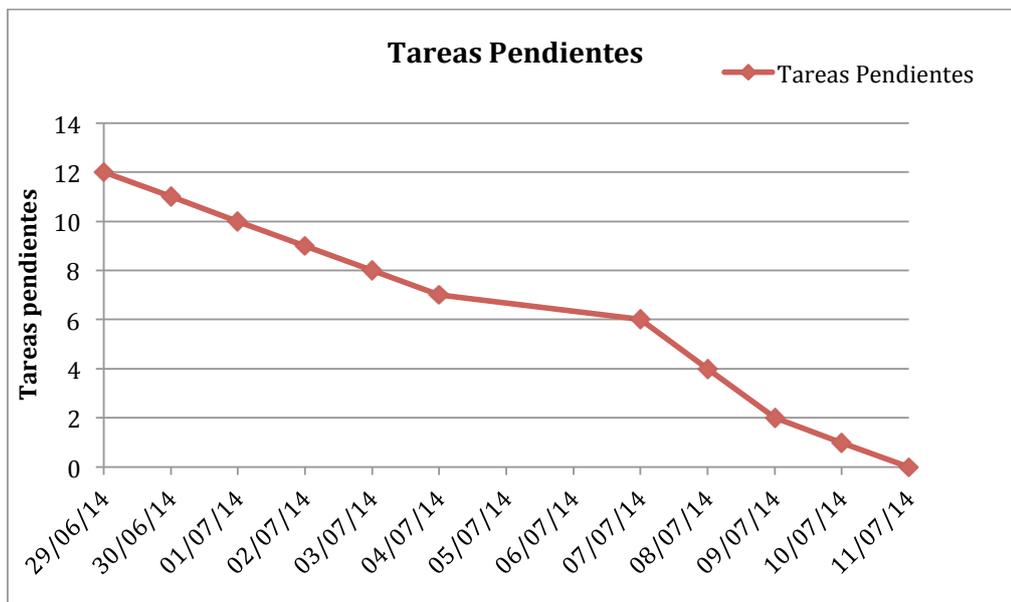
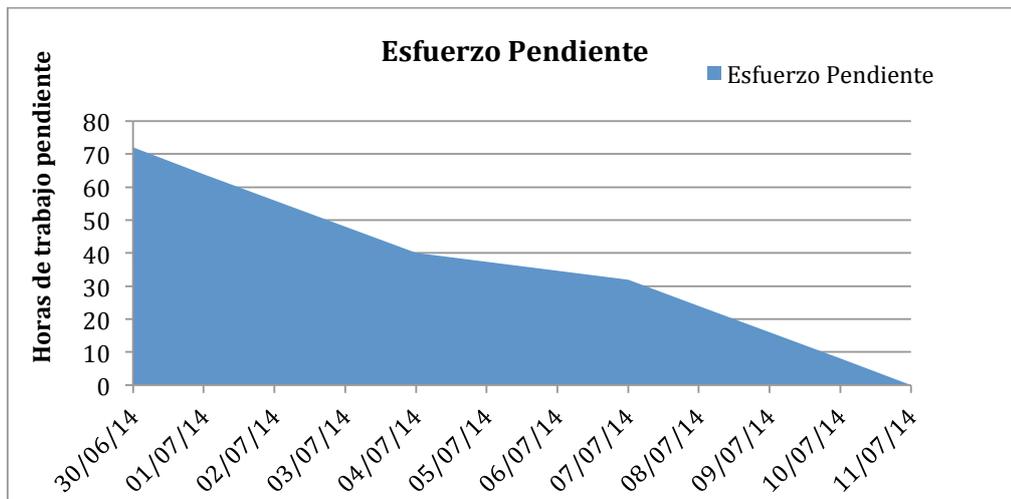


Figura 32 Burn-Down Chart de la Iteración 1.

ESPACIO EN BLANCO DEJADO
INTENCIONALMENTE

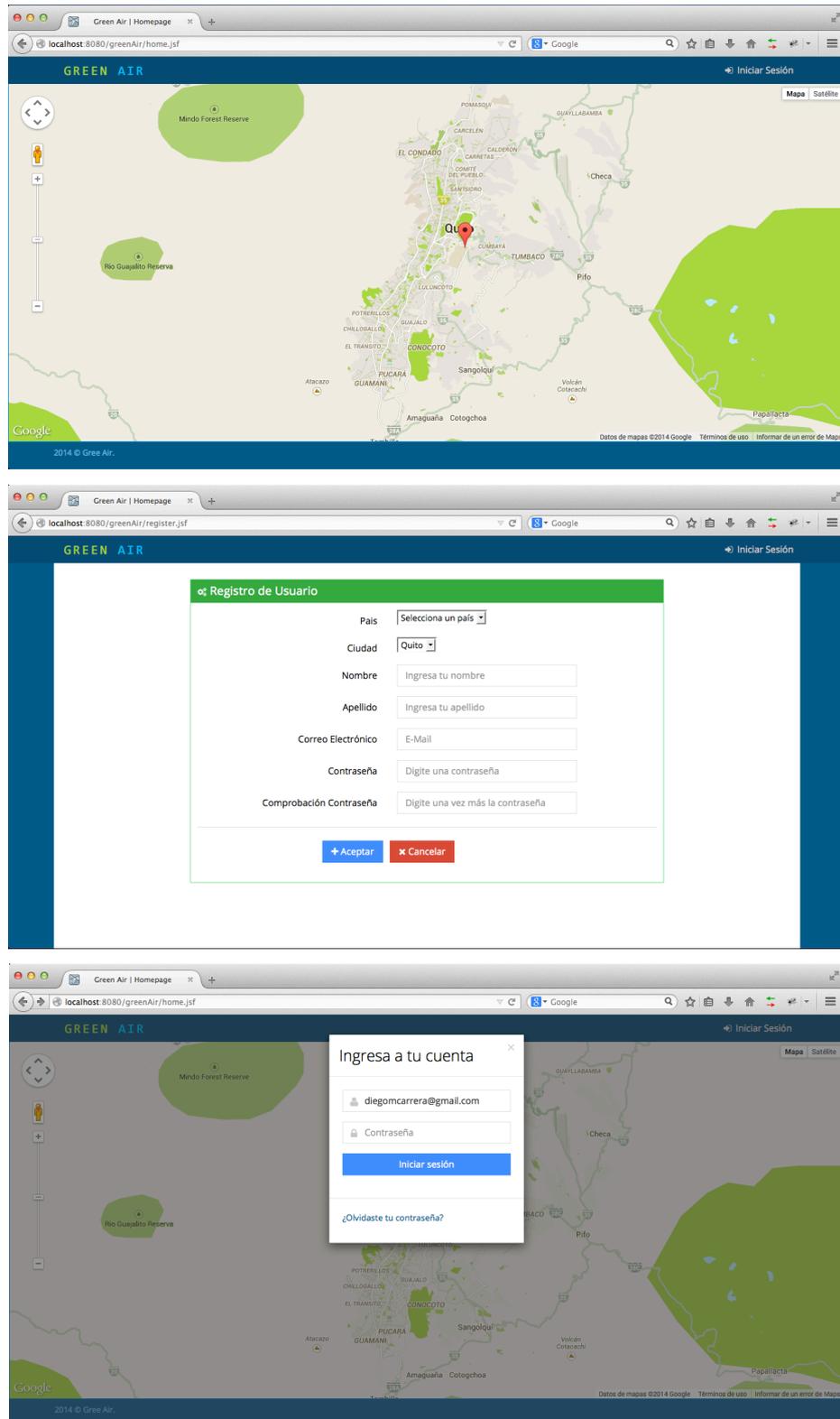


Figura 33 Demo Iteración 1.

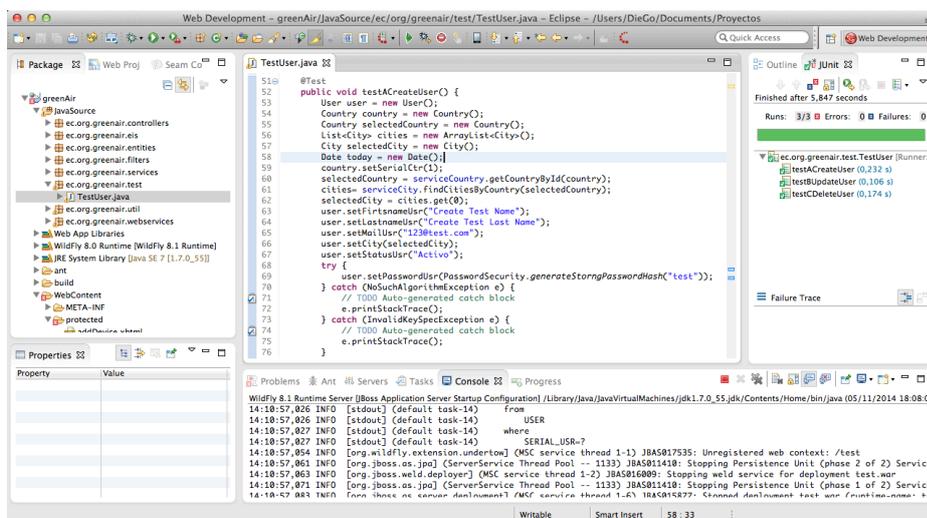


Figura 34 Pruebas unitarias de la iteración 1

5.5 Desarrollo y pruebas de la Iteración 2

Para esta segunda iteración se tomarán las siguientes tres historias de usuario, las mismas que tienen una estimación en tiempo de 2 semanas. Estas historias contemplan la construcción de las pantallas de gestión de dispositivos electrónicos, geolocalización, clave API o API Key y sensores. El Cuadro 21 presenta la historias de usuario elegidas para ser abordadas en la ejecución de la segunda iteración dentro del desarrollo del prototipo de aplicación Web.

5.5.1 Sprint Backlog iteración 2

Al igual que en la primera iteración, será necesario definir las tareas requeridas para cubrir las historias de usuario seleccionadas del Product Backlog detalladas en el Cuadro 15. El Cuadro 22 detalla las tareas que componen el Sprint Backlog de la segunda iteración.

5.5.2 Revisión y seguimiento de la iteración 2

El Cuadro 23 recoge las tareas planteadas y su estado. La Figura 35 ilustra el avance en la ejecución de tareas con el empleo del gráfico Burn-Down Chart.

Cuadro 21

Historias de usuario seleccionadas para la iteración 2

ID	Historia de usuario	Prioridad	Criterio de aceptación
4	Agregar dispositivos electrónicos	4	Ingresar al prototipo Web, seleccionar la opción: Agregar Dispositivo, desplegar el formulario de registro de dispositivo, llenar el formulario, validar la información ingresada. Luego desplegar el formulario de configuración de dispositivo, elegir la ubicación y configuración de sensores, validar información y almacenarla. Verificar en el tracking del servidor de aplicaciones.
5	Editar configuración es de dispositivos electrónicos	4	Ingresar al prototipo Web, seleccionar la opción: Editar Dispositivo, desplegar el formulario de configuración de dispositivo electrónico, modificar las configuraciones necesarios, validar la nueva información ingresada y almacenarla. Verificar en el tracking del servidor de aplicaciones.
6	Capturar y procesar lecturas	5	Ingresar a la plataforma de desarrollo de Arduino, conectar el dispositivo electrónico por medio del puerto USB y comprobar la información captada por los sensores electrónicos.

5.5.3 Pruebas de la iteración 2

La Figura 37 muestra la pruebas unitarias aplicadas a las clases empleadas en el desarrollo de esta segunda iteración con el objetivo de garantizar su correcto funcionamiento.

5.5.4 Demo iteración 2

El demo de esta segunda iteración presenta la pantalla inicial de la gestión de dispositivos electrónicos. Se destaca la posibilidad de agregar sensores a cada dispositivo y especificar su localización. La Figura 36 muestra las pantallas desarrolladas en la iteración 2.

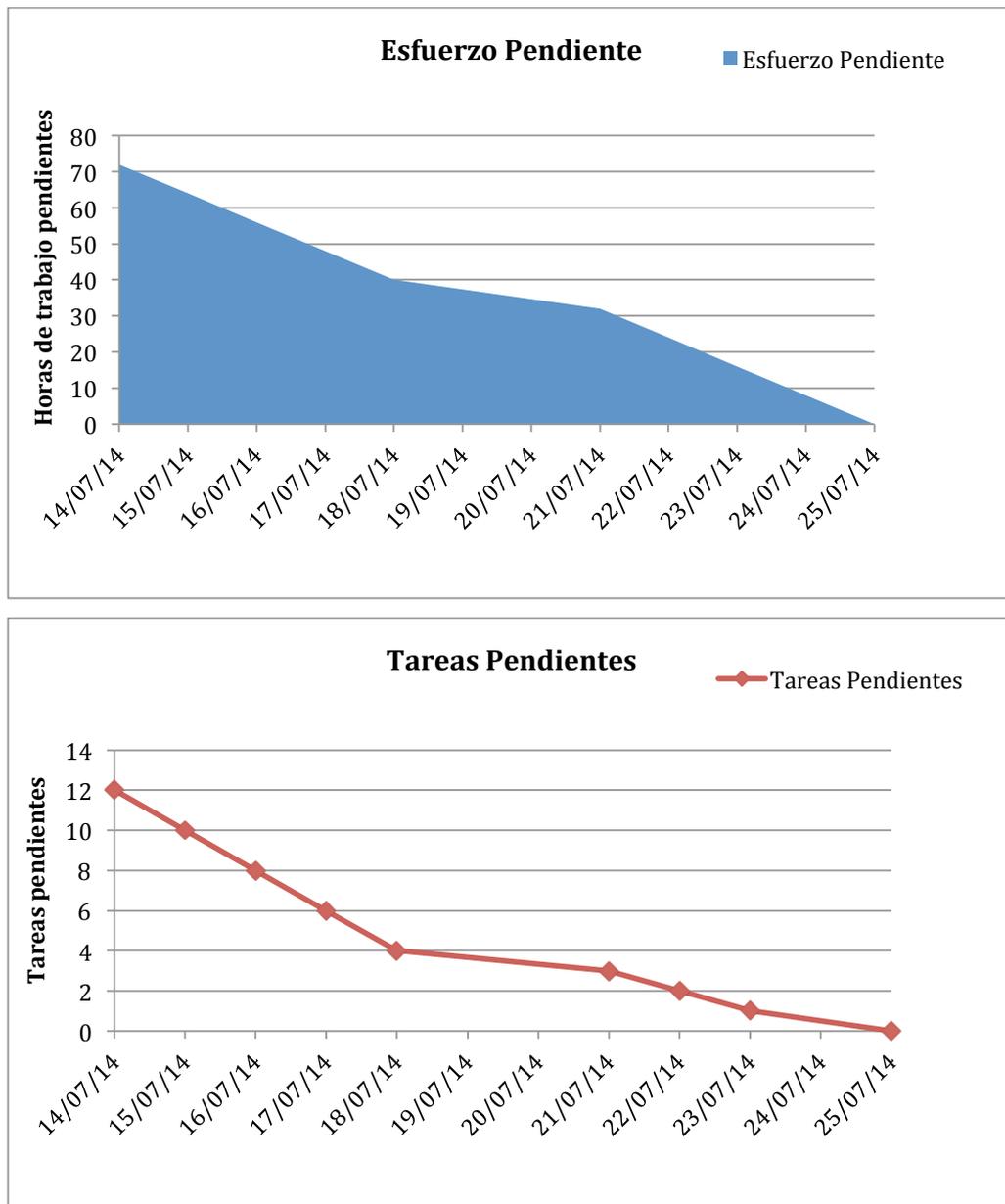


Figura 35 Burn-Down Chart de la iteración 2

Cuadro 22

Sprint Backlog de la iteración 2

Sprint		Inicio	Duración (semanas)				
2		14-Julio-2014	2				
ID	Tarea	Responsable	Product Backlog	14-18 Julio	21-25 Julio		
1	Diseño de las pruebas unitarias para el manejo de dispositivos	Diego Carrera	4	X			
2	Diseño de la interfaz para gestión de dispositivos	Diego Carrera	4	X			
3	Diseño formulario para agregar dispositivos	Diego Carrera	4	X			
4	Codificación del controlador para agregar dispositivos	Diego Carrera	4	X			
5	Diseño formulario para configuración de dispositivos	Diego Carrera	4	X			
6	Codificación controlador para configuración de dispositivos	Diego Carrera	4	X			
7	Diseño del formulario de configuración de sensores	Diego Carrera	5	X			
8	Codificación del controlador de configuración de sensores	Diego Carrera	5	X			
9	Diseño del formulario de geolocalización de dispositivos	Diego Carrera	5		X		
10	Codificación del controlador de geolocalización de dispositivos	Diego Carrera	5		X		
11	Diseño del formulario de llave API o API Key	Diego Carrera	5		X		
12	Codificación del controlador del formulario API Key	Diego Carrera	5		X		
13	Ensamblaje del dispositivo electrónico	Diego Carrera	6		X		
14	Codificación del software que controla el dispositivo electrónico Arduino	Diego Carrera	6		X		

Cuadro 23

Tareas completadas de la iteración 2

Tareas asignadas a: Diego Carrera		Número de tareas: 14			
ID	Tarea	Iteración	Estado	Fecha	Horas
1	Diseño de las pruebas unitarias para el manejo de dispositivos	2	Completada	14 - Julio - 2014	4
2	Diseño de la interfaz para gestión de dispositivos	2	Completada	14 - Julio - 2014	4
3	Diseño formulario para agregar dispositivos	2	Completada	15 - Julio - 2014	4
4	Codificación del controlador para agregar dispositivos	2	Completada	15 - Julio - 2014	4
5	Diseño formulario para configuración de dispositivos	2	Completada	16 - Julio - 2014	4
6	Codificación controlador para configuración de dispositivos	2	Completada	16 - Julio - 2014	4
7	Diseño del formulario de configuración de sensores	2	Completada	17 - Julio - 2014	4
8	Codificación del controlador de configuración de sensores	2	Completada	17 - Julio - 2014	4
9	Diseño del formulario de geolocalización de dispositivos	2	Completada	18 - Julio - 2014	4
10	Codificación del controlador de geolocalización de dispositivos	2	Completada	18 - Julio - 2014	4
11	Diseño del formulario de llave API o API Key	2	Completada	21 - Julio - 2014	8
12	Codificación del controlador del formulario API Key	2	Completada	22 - Julio - 2014	8
13	Ensamblaje del dispositivo electrónico	2	Completada	23 - Julio - 2014	8
14	Codificación del software que controla el dispositivo electrónico Arduino	2	Completada	25 - Julio - 2014	16

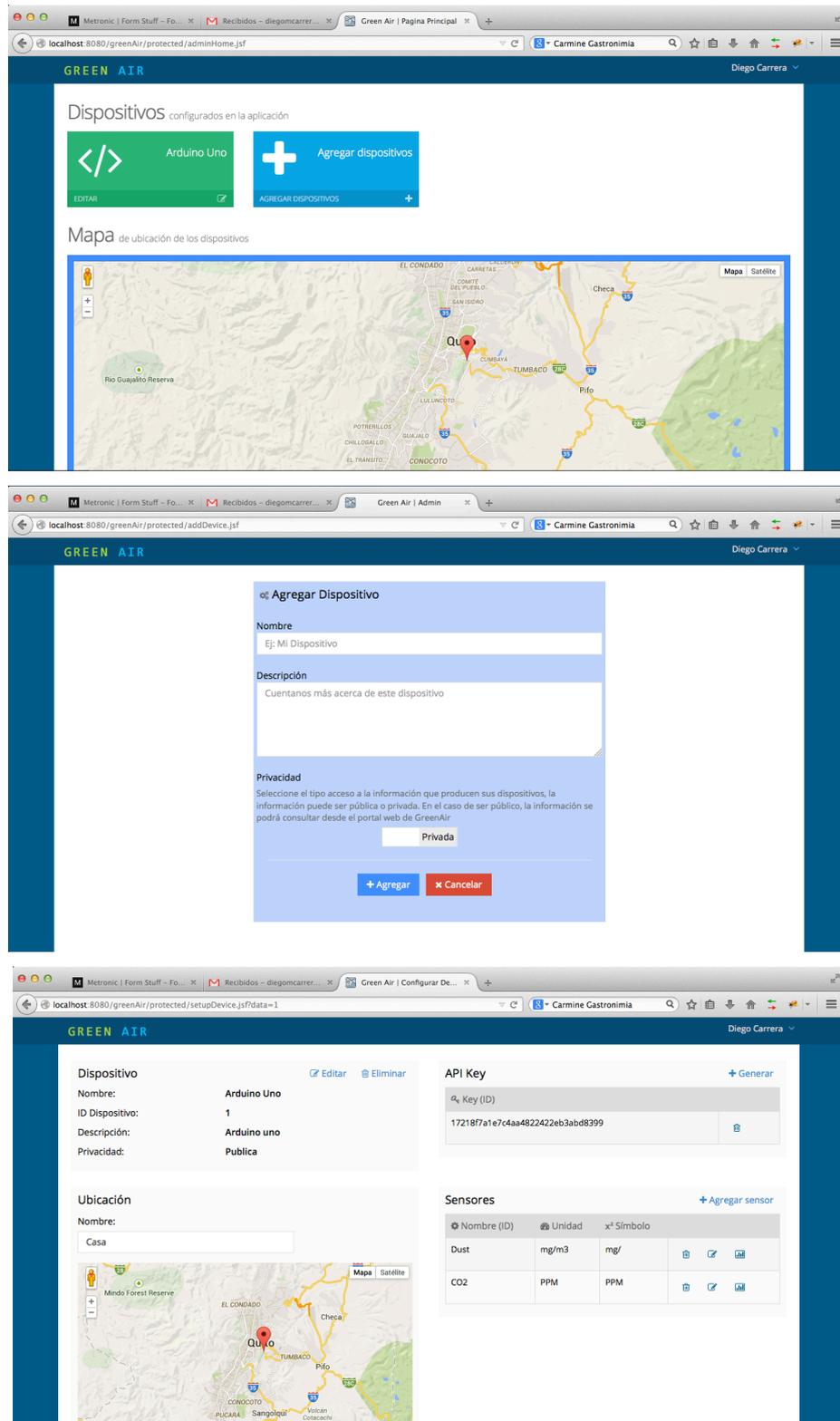


Figura 36 Demo de la iteración 2.

The figure displays three screenshots of the Eclipse IDE, each showing a different unit test being executed. The IDE interface includes a Package Explorer on the left, a central editor for the test code, and a right-hand pane for test results and console output.

Top Screenshot: TestDevice.java
 The editor shows the `TestDevice` class with methods `testA()`, `testB()`, and `testC()`. The test runner window shows the test finished after 4,271 seconds with 3/3 runs, 0 errors, and 0 failures. The console shows the test results for `testA` (0.181 s), `testB` (0.068 s), and `testC` (0.169 s).

Middle Screenshot: TestSensor.java
 The editor shows the `TestSensor` class with a `testA()` method. The test runner window shows the test finished after 4,525 seconds with 3/3 runs, 0 errors, and 0 failures. The console shows the test results for `testA` (0.582 s), `testB` (0.077 s), and `testC` (0.160 s).

Bottom Screenshot: TestLocation.java
 The editor shows the `TestLocation` class with methods `testA()`, `testB()`, and `testC()`. The test runner window shows the test finished after 4,814 seconds with 3/3 runs, 0 errors, and 0 failures. The console shows the test results for `testA` (0.435 s), `testB` (0.071 s), and `testC` (0.165 s).

Figura 37 Pruebas unitarias de la iteración 2.

5.6 Desarrollo y pruebas de la Iteración 3

La tercera iteración abordará las siguientes tres historias de usuario contenidas en la pila de producto o Product Backlog, las mismas que se estiman durarán cinco días laborables. El cuadro 5.10 recoge las historias de usuario seleccionadas para esta tercera iteración.

Cuadro 24

Historias de usuario seleccionadas para la iteración 3

ID	Historia de usuario	Prioridad	Criterio de aceptación
7	Transmitir información	4	Ingresar al prototipo Web, seleccionar la opción: Agregar Dispositivo, desplegar el formulario de registro de dispositivo, llenar el formulario, validar la información ingresada. Luego desplegar el formulario de configuración de dispositivo, elegir la ubicación y configuración de sensores, validar información y almacenarla. Verificar en el tracking del servidor de aplicaciones.
8	Autenticar dispositivos electrónicos	4	Ingresar al prototipo Web, seleccionar la opción: Editar Dispositivo, desplegar el formulario de configuración de dispositivo electrónico, modificar las configuraciones necesarios, validar la nueva información ingresada y almacenarla. Verificar en el tracking del servidor de aplicaciones.
9	Almacenar información	5	Ingresar a la plataforma de desarrollo de Arduino, conectar el dispositivo electrónico por medio del puerto USB y comprobar la información captada por los sensores electrónicos.

5.6.1 Sprint Backlog iteración 3

Siguiendo con la dinámica del trabajo, planteada por la metodología, será necesario definir las tareas requeridas para cumplir con las historias de usuario seleccionadas. El Cuadro 25 detalla cada una de las tareas para cada ítem escogido de la pila de producto.

5.6.2 Revisión y seguimiento iteración 3

Para verificar el avance y cumplimiento se listarán el estado y grado de cumplimiento de las tareas propuestas. El Cuadro 26 muestra el estado y cumplimiento de cada una de las tareas necesarias para cumplir con las historias de usuario. La Figura 39 ilustra el avance del proyecto mediante el empleo de los gráficos Burn-Down Chart.

5.6.3 Pruebas de la iteración 3

Al igual que en las dos iteraciones anteriores, se aplicarán las pruebas unitarias diseñadas para esta iteración con el fin de garantizar el correcto funcionamiento del prototipo de aplicación Web. La Figura 38 muestra las pruebas unitarias aplicadas a cada clase empleada en la tercera iteración.

5.6.4 Demo iteración 3

El demo de la tercera iteración mostrará el funcionamiento del dispositivo electrónico interactuando con el prototipo de aplicación Web mediante el consumo del servicio Web RESTful. La Figura 40 ilustra la salida de tracking del servidor de aplicaciones donde se puede apreciar las sentencias ejecutadas por el consumo del servicio Web construido.

5.7 Desarrollo y evaluación de la Iteración 4

Esta cuarta y última iteración abordará las dos últimas historias de usuario contenidas en la pila de producto o Product Backlog. El Cuadro 27 detalla las historias de usuario para esta cuarta iteración.

5.7.1 Sprint Backlog de la iteración 4

El Cuadro 28 lista las tareas necesarias para la construcción de los mapas de ubicación de los dispositivos electrónicos y los gráficos de las lecturas tomadas por los sensores electrónicos.

5.7.2 Revisión y seguimiento de la iteración 4

Para controlar el avance de la cuarta iteración se listarán las tareas planteadas en el Sprint Backlog, con su respectivo estado y grado de cumplimiento, el Cuadro 29 detalla estas características. La Figura 41 ilustra los gráficos de seguimiento de la iteración Burn-Down Chart.

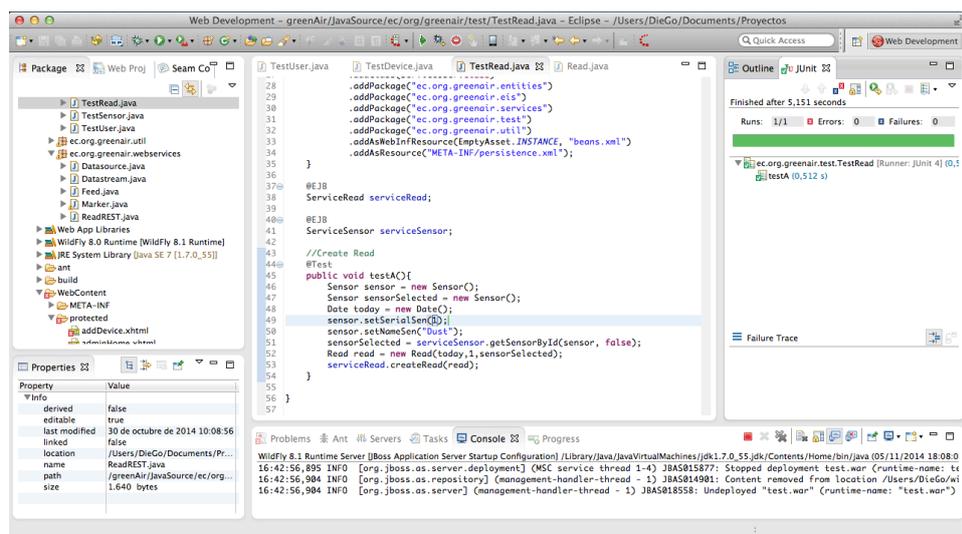


Figura 38 Prueba unitaria de la iteración 3

Cuadro 25

Sprint Backlog de la iteración 3

Sprint	Inicio	Duración (semanas)		
3	28-Julio-2014	1		
ID	Tarea	Responsable	Product Backlog	
1	Codificar la librería en C++, para crear cadenas JSON con la información recolectada	Diego Carrera	7	X
2	Configurar el módulo RN-XV WiFi	Diego Carrera	7	X
3	Codificar el servicio Web RESTful	Diego Carrera	8	X
4	Diseñar la prueba unitaria para el manejo de los datos de los sensores electrónicos	Diego Carrera	8	X
5	Codificar función para autenticar los dispositivos electrónicos	Diego Carrera	8	X
6	Codificar función para almacenar los datos transmitidos desde los dispositivos electrónicos	Diego Carrera	9	X

Cuadro 26

Tareas completadas de la iteración 3

Tareas asignadas a: Diego Carrera		Número de tareas: 6			
ID	Tarea	Iteración	Estado	Fecha	Horas
1	Codificar la librería en C++, para crear cadenas JSON	3	Completada	28 - Julio - 2014	4
2	Configurar el módulo RN-XV WiFi	3	Completada	28 - Julio - 2014	4
3	Codificar el servicio Web RESTful	3	Completada	29 - Julio - 2014	8
4	Diseñar la prueba unitaria para el manejo de los datos de los sensores electrónicos	3	Completada	29 - Julio - 2014	8

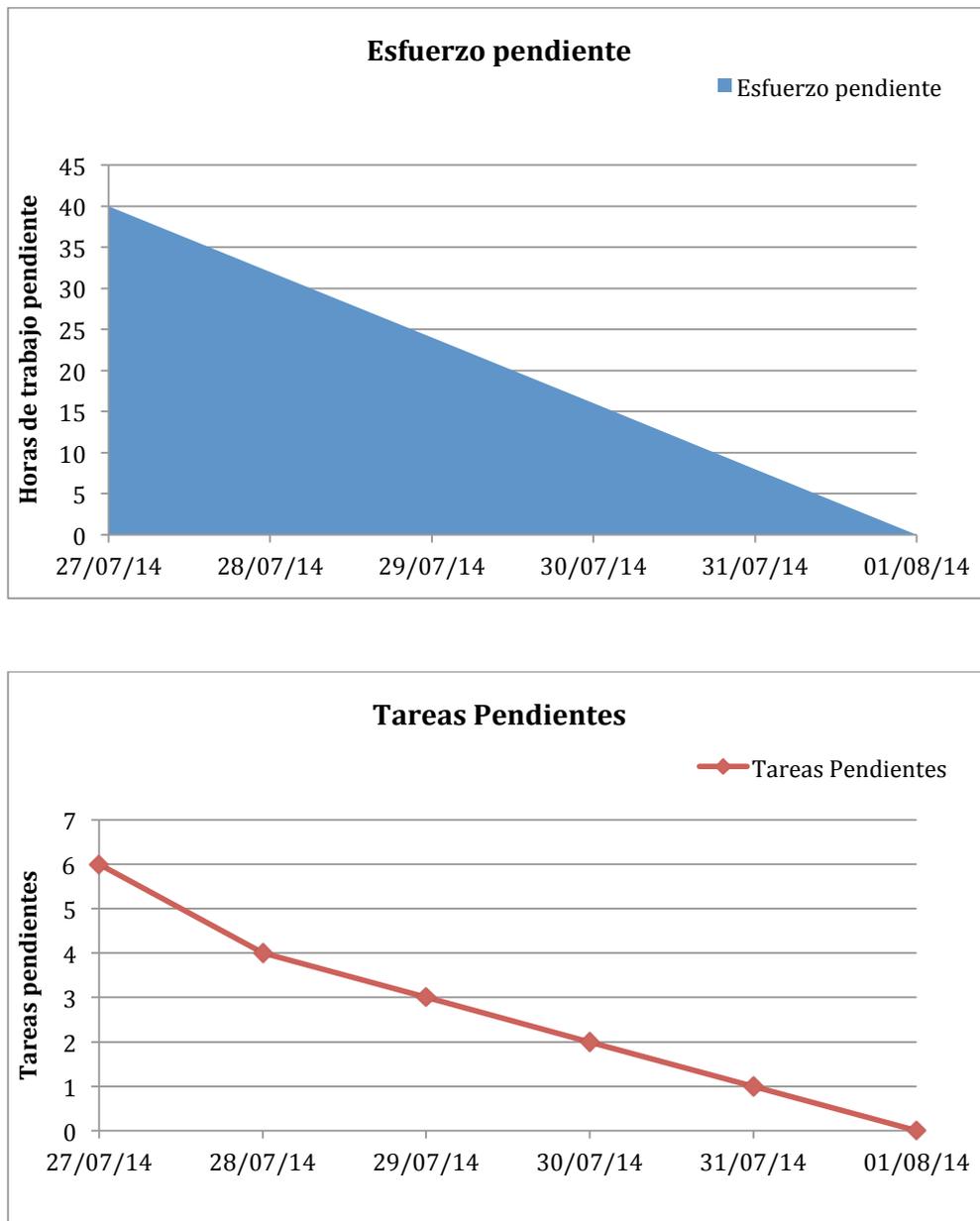
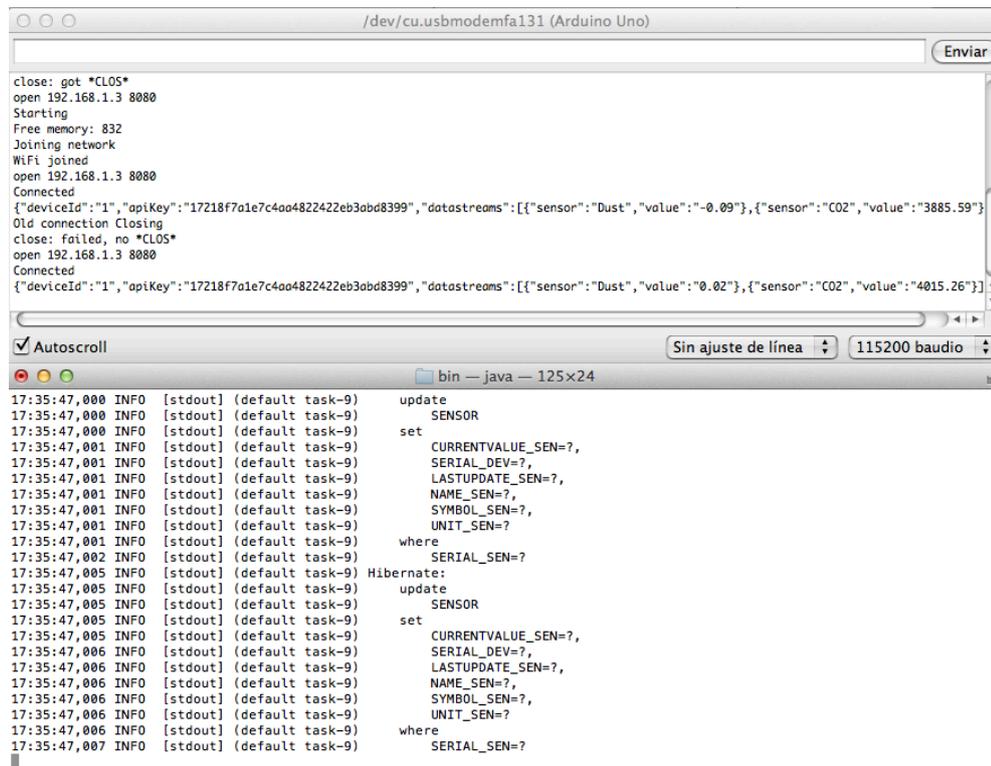


Figura 39 Burn-Down Chart de la iteración 3

ESPACIO EN BLANCO DEJADO
INTENCIONALMENTE



```

/dev/cu.usbmodemfa131 (Arduino Uno)
Enviar
close: got *CLOS*
open 192.168.1.3 8080
Starting
Free memory: 832
Joining network
WiFi joined
open 192.168.1.3 8080
Connected
{"deviceId":"1","apiKey":"17218f7a1e7c4aa4822422eb3abd8399","datastreams":[{"sensor":"Dust","value":"-0.09"}, {"sensor":"CO2","value":"3885.59"}]}
Old connection Closing
close: failed, no *CLOS*
open 192.168.1.3 8080
Connected
{"deviceId":"1","apiKey":"17218f7a1e7c4aa4822422eb3abd8399","datastreams":[{"sensor":"Dust","value":"0.02"}, {"sensor":"CO2","value":"4015.26"}]}

Autoscroll
Sin ajuste de línea 115200 baudio
bin — java — 125x24
17:35:47,000 INFO [stdout] (default task-9) update
17:35:47,000 INFO [stdout] (default task-9)   SENSOR
17:35:47,000 INFO [stdout] (default task-9)   set
17:35:47,001 INFO [stdout] (default task-9)     CURRENTVALUE_SEN=?,
17:35:47,001 INFO [stdout] (default task-9)     SERIAL_DEV=?,
17:35:47,001 INFO [stdout] (default task-9)     LASTUPDATE_SEN=?,
17:35:47,001 INFO [stdout] (default task-9)     NAME_SEN=?,
17:35:47,001 INFO [stdout] (default task-9)     SYMBOL_SEN=?,
17:35:47,001 INFO [stdout] (default task-9)     UNIT_SEN=?
17:35:47,001 INFO [stdout] (default task-9)   where
17:35:47,002 INFO [stdout] (default task-9)     SERIAL_SEN=?
17:35:47,005 INFO [stdout] (default task-9) Hibernate:
17:35:47,005 INFO [stdout] (default task-9)   update
17:35:47,005 INFO [stdout] (default task-9)     SENSOR
17:35:47,005 INFO [stdout] (default task-9)   set
17:35:47,006 INFO [stdout] (default task-9)     CURRENTVALUE_SEN=?,
17:35:47,006 INFO [stdout] (default task-9)     SERIAL_DEV=?,
17:35:47,006 INFO [stdout] (default task-9)     LASTUPDATE_SEN=?,
17:35:47,006 INFO [stdout] (default task-9)     NAME_SEN=?,
17:35:47,006 INFO [stdout] (default task-9)     SYMBOL_SEN=?,
17:35:47,006 INFO [stdout] (default task-9)     UNIT_SEN=?
17:35:47,006 INFO [stdout] (default task-9)   where
17:35:47,007 INFO [stdout] (default task-9)     SERIAL_SEN=?

```

Figura 40 Demo de la iteración 3

5.7.3 Pruebas de la iteración 4

A lo largo de todas las iteraciones, se han venido ejecutando las pruebas unitarias de las clases que componen el prototipo de aplicación Web. Para esta última iteración se ejecutarán las pruebas unitarias de los métodos de consulta.

5.7.4 Demo de la iteración 4

El demo de esta última iteración mostrará los mapas de ubicación de los dispositivos, así como los gráficos de las lecturas recogidas, por los sensores electrónicos, en tiempo real. La Figura 42 muestra el demo de la cuarta y última iteración.

Cuadro 27

Historias de usuario escogidas para la iteración 4

ID	Historia de usuario	Prioridad	Criterio de aceptación
10	Generar mapa y gráficos	4	Ingresar al prototipo Web y comprobar la ubicación de los dispositivos electrónicos, pertenecientes al usuario que los registró, en el mapa que se despliega en la página inicial de la gestión de dispositivos. Para la generación de gráficos, escoger la opción de edición de dispositivos, seleccionar el icono de generación de gráfico dispuesto para cada sensor y comprobar el gráfico de la información de las lecturas recogidas en función del tiempo.
11	Consultar Información	4	Ingresar al portal del prototipo de aplicación Web y verificar la ubicación de los dispositivos electrónico, que están compartiendo su información públicamente, en el mapa que se desplegará. Seleccionar un dispositivo y comprobar la información, en tiempo real, de cada uno de los sensores con los que está equipado un dispositivo electrónico.

ESPACIO EN BLANCO DEJADO
INTENCIONALMENTE

Cuadro 28

Sprint Backlog de la iteración 4

Sprint		Inicio	Duración (semanas)				4 - 8 Agosto	
4		4-Agosto-2014	1					
ID	Tarea	Responsable	Product Backlog					
1	Codificar las consultas necesarias para presentar dispositivos e información de sus sensores	Diego Carrera	10	X				
2	Codificar el controlador para la presentación de dispositivos sobre el mapa	Diego Carrera	10	X				
3	Codificar el controlador para la generación de gráficos	Diego Carrera	10	X				
4	Construir las ventanas que presentan los gráficos al seleccionar un sensor	Diego Carrera	10	X				
5	Utilizar y personalizar el API de Google Maps para presentar los dispositivos sobre el mapa	Diego Carrera	11	X				

Cuadro 29

Tareas completadas de la iteración 4

Tareas asignadas a: Diego Carrera		Número de tareas: 5						
ID	Tarea	Iteración	Estado	Fecha	Horas			
1	Codificar las consultas necesarias para presentar dispositivos e información de sus sensores	4	Completada	4 - Agosto - 2014	8			
2	Codificar el controlador para la presentación de dispositivos sobre el mapa	4	Completada	5 - Agosto - 2014	8			
3	Codificar el controlador para la generación de gráficos	4	Completada	6 - Agosto - 2014	8			

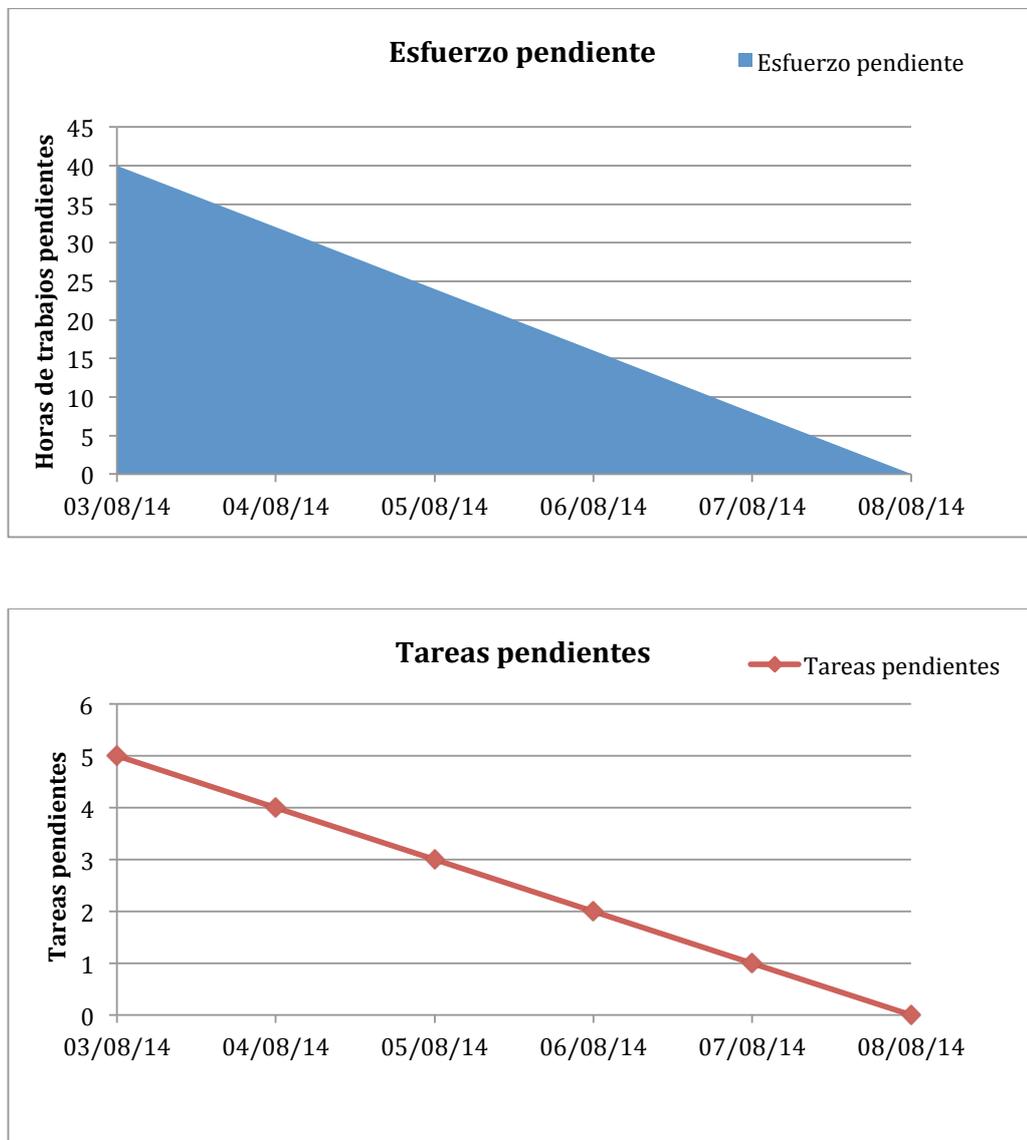


Figura 41 Burn-Down Chart de la iteración 4

ESPACIO EN BLANCO DEJADO
INTENCIONALMENTE

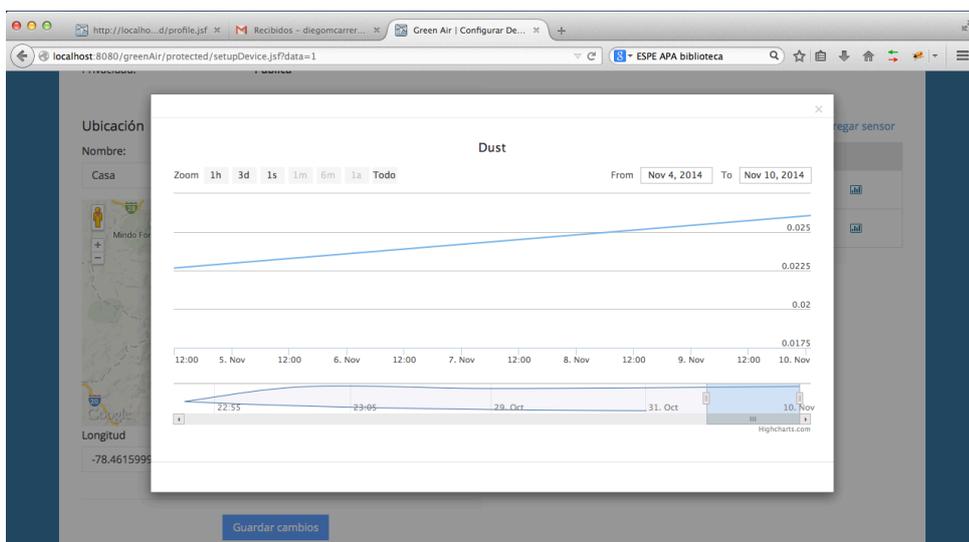
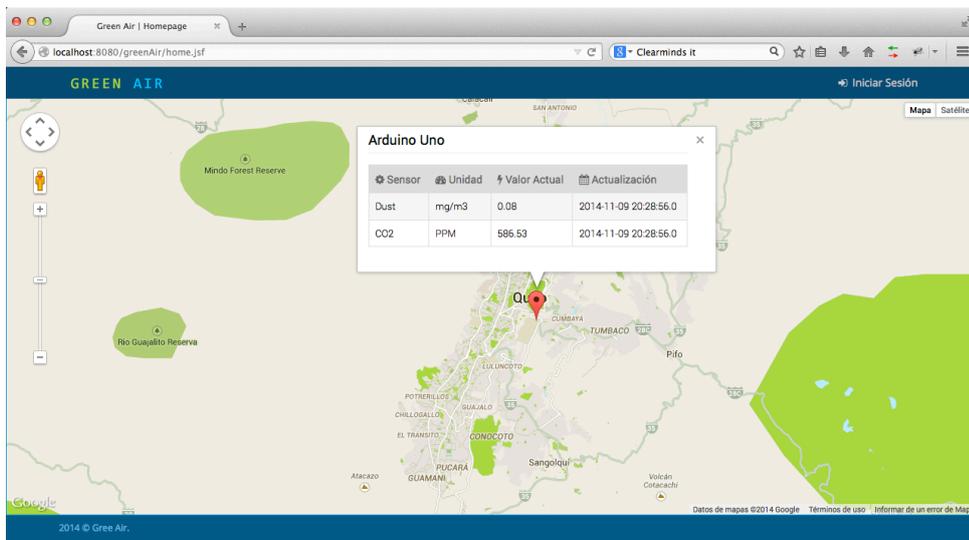


Figura 42 Demo de la iteración 4

ESPACIO EN BLANCO DEJADO
INTENCIONALMENTE

CAPÍTULO 6

ANÁLISIS E INTERPRETACIÓN DE RESULTADOS

6.1 Introducción

Para elaborar la prueba de concepto, se obtuvieron muestras representativas de la concentración de los tres contaminantes que los sensores del dispositivo electrónico son capaces de medir. Las mediciones se realizaron en la Parroquia Jipijapa, en la ciudad de Quito; y en las cuevas de investigación científica en la ciudad del Tena. En el Anexo 2, se encuentra la evidencia fotográfica de las mediciones realizadas en la ciudad de Quito y en la ciudad del Tena.

La medición de los parámetros en la ciudad de Quito, se realizaron en la parroquia mencionada, debido a que en ésta se encuentra localizada la estación principal de monitoreo de la calidad del aire en Quito, la que es operada y mantenida por la Secretaría de Medio Ambiente del Distrito Metropolitano. Los parámetros medidos en las cuevas de investigación de la ciudad del Tena, se realizaron siguiendo las recomendaciones del Grupo de Investigación en Geodinámica Interna y Externa de la Universidad de las Fuerzas Armadas - ESPE.

El Cuadro 30 describe la relación entre los parámetros medidos por los sensores del prototipo versus los límites permisibles de concentración de Monóxido de Carbono y de valores referenciales para la Densidad de Polvo y Dióxido de Carbono, provenientes de estándares internacionales definidos por la USEPA y ASRAEH.

Esto permitirá realizar una primera aproximación a un análisis detallado para realizar una correcta interpretación de los datos obtenidos en las distintas mediciones efectuadas.

Cuadro 30

Rangos, límites permisibles y valores referenciales aplicables a los sensores

Sensor	Definición	Parámetro de medición	Unidades de Medición	Rango de Medición	Límite permisible o Valor Referencial	Fuente Valor Referencial
Polvo respirable	Partículas sólidas mas grandes del tamaño coloidal capaces de estar suspendidas en el aire.	Densidad de polvo en viviendas, humo de tabaco, etc.	miligramos/m ³ (mg/ m ³)	0 - 0,8 mg/m ³	Calidad de Aire Exterior Concentración promedio: 0,26 mg/m ³ Calidad de Aire Interior 0,2 mg/m ³ en áreas de no fumadores 0,2 – 0, 6 mg/m ³ en áreas de fumadores	USEPA. Calidad de Aire Exterior. Estándar ASHRAE -62.- Ventilación para niveles aceptables de calidad de aire interna
	Gas incoloro que proviene de procesos de combustión naturales y antropogénicos	Concentración de CO en la atmósfera	microgramos/m ³ (ppm)	10 - 10000 ppm	35 ppm Periodo de exposición de 1 hora	USEPA. Calidad de Aire Exterior
CO2	Gas proveniente de procesos de combustión de combustibles fósiles, deforestación,	Concentración de CO2 en la atmósfera	microgramos/m ³ (ppm)	350 - 10000 ppm	Concentración media atmosférica 400 ppm	Concentración Global de CO2 en la atmósfera en el tiempo.

6.2 Análisis de parámetros de la Ciudad de Quito

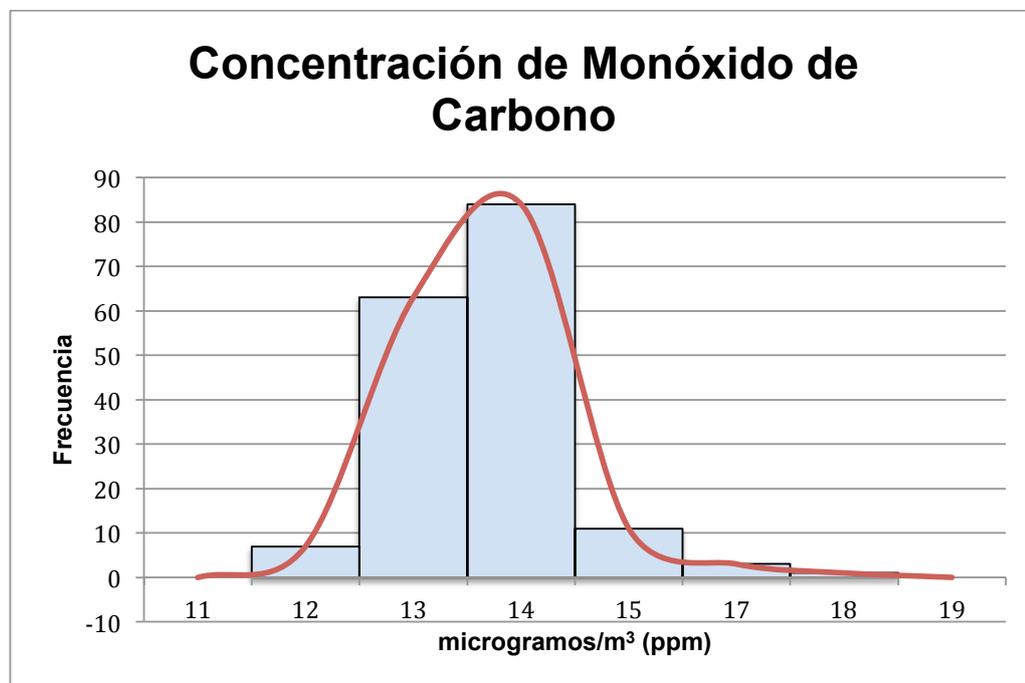


Figura 43 Concentración de Monóxido de Carbono Quito

La Figura 43 muestra el resultado del análisis estadístico de los datos obtenidos de la medición de la concentración de Monóxido de Carbono (CO) en la ciudad de Quito. El conjunto de datos obtenidos suman 169 lecturas cuyo valor mínimo de concentración es 12 ppm y el máximo de 18 ppm. Se puede observar una mayor concentración de lecturas entre los valores de 13 ppm y 14 ppm, lo que se ratifica con una media para esta muestra de 13,68 ppm en una hora de medición. Este promedio está por debajo de los valores permisibles en períodos de exposición de una hora según la USAEPA.

El recuento de las lecturas, del sensor de Densidad de Polvo, hacen a 169 datos cuyo valor mínimo es de 0,00 mg/m³ y su máximo es de 0,23 mg/m³. Se evidencia una mayor concentración de los datos entre los valores de 0,06 mg/m³ y 0,08 mg/m³, obteniéndose de esta muestra representativa una media de 0,075 mg/m³, promedio que se encuentra por debajo del límite

permisible para esta variable según el estándar ASHRAE-62. La Figura 44 ilustra el análisis estadístico aplicado.

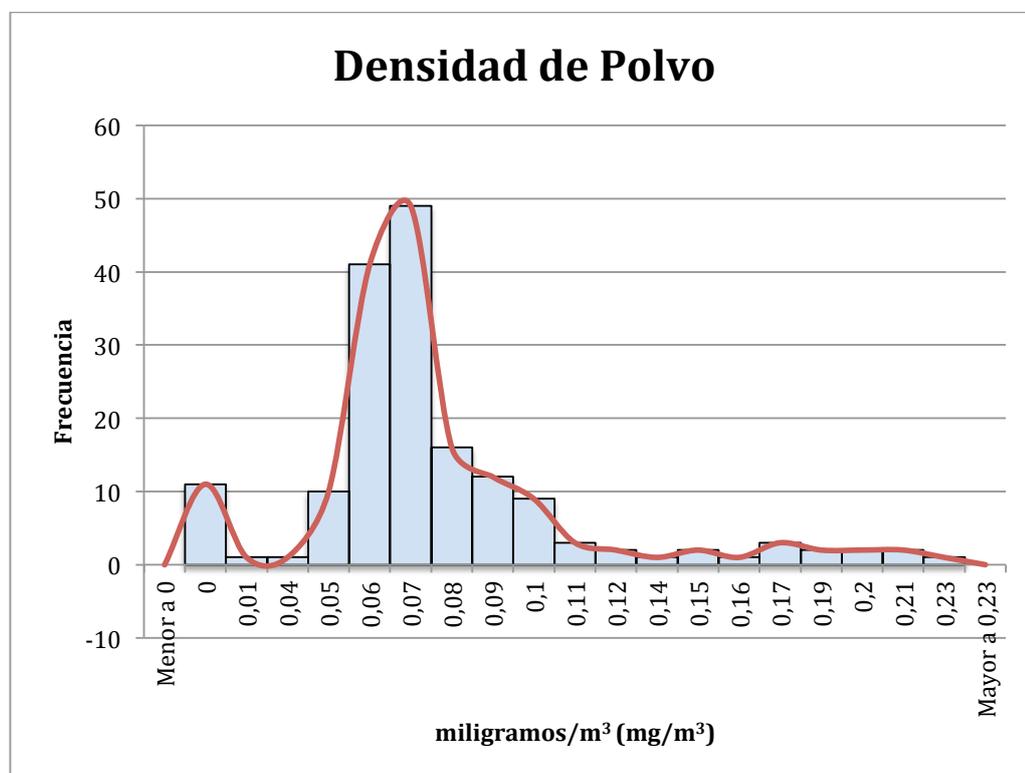


Figura 44 Densidad de Polvo Quito

6.3 Análisis de parámetros en cuevas de investigación, el Tena

Las mediciones en las cuevas de investigación ubicadas en las cercanías de la ciudad del Tena se las realizó bajo la dirección del Dr. Theofilos Toulkeridis, líder del Grupo de Investigación en Geodinámica Interna y Externa, profesor titular y docente investigador a tiempo completo de la Universidad de las Fuerzas Armadas – ESPE.

Se utilizaron los sensores de Dióxido de Carbono (CO₂) y Densidad de Polvo para este ejercicio. Cabe recalcar que la transmisión de los datos se la hizo directamente entre un ordenador y el dispositivo electrónico Arduino por medio del puerto USB, con los siguientes resultados.

La Figura 45 ilustra el análisis estadístico aplicado a la primera cueva visitada, la cual presenta una media de $0,035 \text{ mg/m}^3$ en la variable de Densidad de Polvo de un recuento total de 269 lecturas efectuadas, que van desde $0,0 \text{ mg/m}^3$ hasta $0,10 \text{ mg/m}^3$, donde las mayor concentración se evidencia entre los $0,02 \text{ mg/m}^3$ y $0,05 \text{ mg/m}^3$

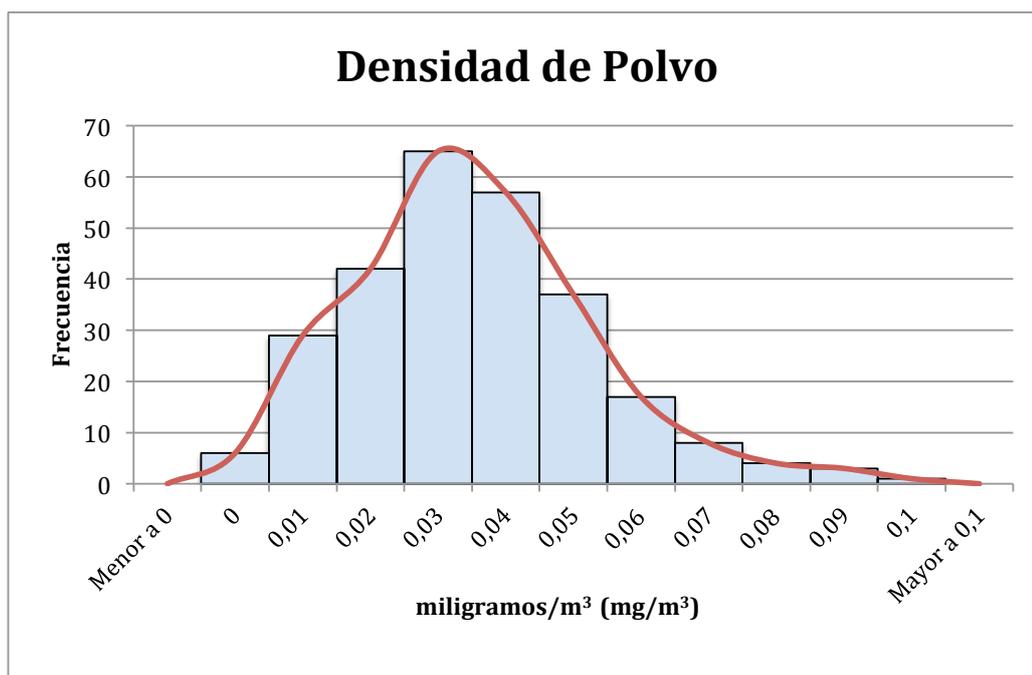


Figura 45 Densidad de Polvo – Cueva Castillo

La Figura 46 muestra el análisis estadístico aplicado para la variable de Dióxido de Carbono (CO₂), con una media de 4.424,24 partes por millón (ppm) en una muestra de 154 lecturas. Se distingue que la mayor concentración de los datos están entre los 2.200,50 ppm y los 6.939,50 ppm.

En la segunda cueva visitada se consiguieron 313 lecturas de la variable de Densidad de Polvo, con una media de 0,03 miligramos por metro cúbico (mg/m^3), ubicando la mayor cantidad de datos entre los $0,02 \text{ mg/m}^3$ y los $0,05 \text{ mg/m}^3$. La Figura 47 ilustra los resultados del análisis estadístico aplicado a esta variable.

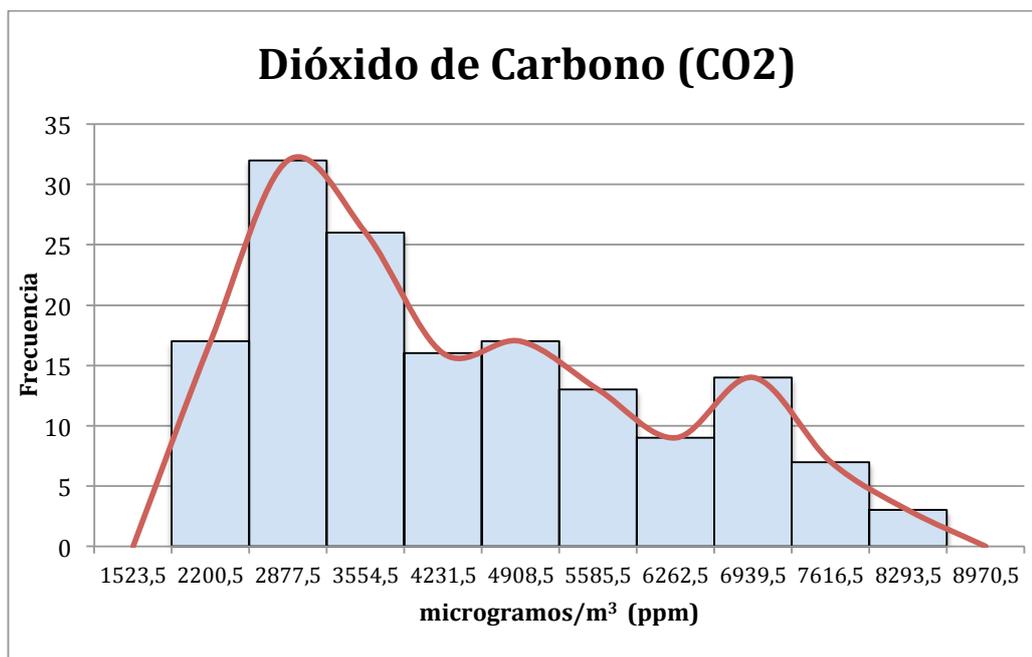


Figura 46 Concentración de Dióxido de Carbono – Cueva Castillo

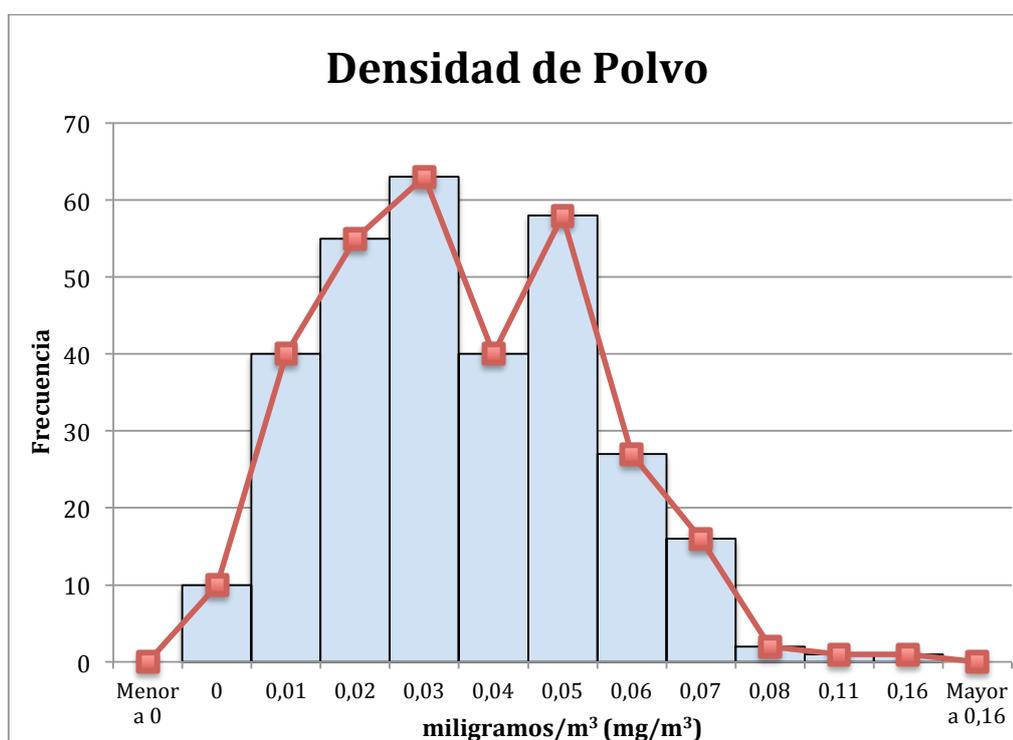


Figura 47 Densidad de Polvo – Cueva Gruta de la Virgen

La concentración de Dióxido de Carbono, ilustrada en la Figura 48, tiene una media de 4.879,81 partes por millón (ppm) en un conjunto de datos de 318 lecturas, cuyo valor mínimo es de 1790,63 ppm y su máximo de 9.033, ppm. La mayor concentración de las lecturas se encuentran entre los 2.877,50 ppm y los 7227,50 ppm.

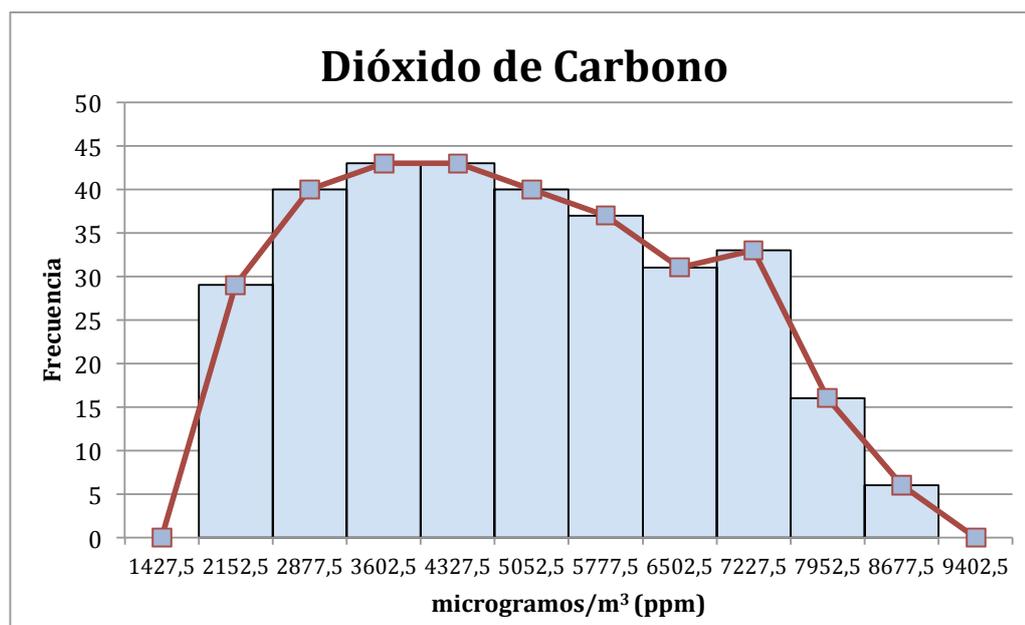


Figura 48 Concentración de Dióxido de Carbono – Cueva Gruta de la Virgen

La Figura 49 ilustra la comparación de la variable de Densidad de Polvo entre las muestras obtenidas en la ciudad de Quito y las que fueron realizadas en la cueva de investigación científica Gruta de la Virgen en la cercanías de la ciudad del Tena. Se observa que la mayor concentración de polvo por metro cúbico tiene la ciudad de Quito, alrededor de los 0,07 mg/m³ mientras que en la cueva se obtuvo una concentración de alrededor de los 0,05 mg/m³.

Los datos de la ciudad de Quito muestran valores, dentro de la muestra obtenida, de 0,23 mg/m³, algo que en la Gruta de la Virgen no se dió, siendo 0,16 mg/m³ el valor máximo alcanzado en la muestra representativa obtenida.

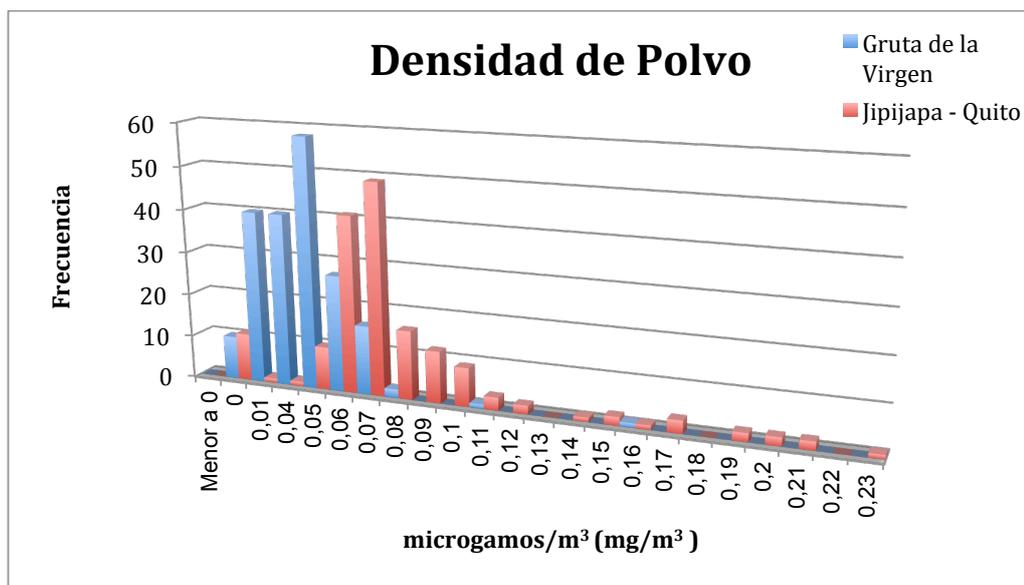


Figura 49 Comparación de la Densidad de Polvo

CAPÍTULO 7

CONCLUSIONES Y RECOMENDACIONES

7.1 Conclusiones

- a) El empleo de la combinación entre las metodologías Scrum y Extreme Programming a lo largo del desarrollo tanto del prototipo Web como del dispositivo electrónico, permitió obtener un producto que cubre todas las necesidades planteadas, dentro de los plazos previstos.
- b) La especificación de requerimientos, los mismos que fueron documentados en base al estándar IEEE-830, sirvió para definir las necesidades básicas de una solución tecnológica capaz de medir los parámetros de calidad del aire y almacenar la información para su posterior análisis.
- c) Las tareas de análisis y diseño, empleando el Lenguaje Unificado de Modelamiento (UML), previas al desarrollo del prototipo de aplicación Web, agilizaron la etapa de iteraciones sucesivas o Sprints, al permitir la definición de una arquitectura de software capaz de adaptarse a los cambios en los requerimientos y facilitando el diseño de las pruebas unitarias aplicadas al código.
- d) El empleo de los distintos elementos que forman parte de la plataforma de desarrollo Java Enterprise Edition (JEE), simplificaron el desarrollo del prototipo de aplicación Web, brindado, adicionalmente, la capacidad de escalar el software en un futuro.
- e) La plataforma Arduino facilitó la construcción de un prototipo electrónico capaz de manejar los sensores de Monóxido de

Carbono (CO), Dióxido de Carbono (CO₂) y Densidad de Polvo. También hizo posible la transmisión, de los datos recolectados, hacia el prototipo de aplicación Web utilizando los protocolos TCP/IP y HTTP cumpliendo, de esta manera, el objetivo de ampliar el ámbito de medición de la red de monitoreo con la que cuenta la ciudad de Quito.

- f)** El uso del servicio Web de tipo “RESTful” permitió la interacción entre el prototipo de aplicación Web y la plataforma electrónica Arduino debido a su baja complejidad, algo que resulta bastante útil en escenarios donde se tiene restricciones en el hardware como es el caso de los microcontroladores Arduino.
- g)** El resultado del análisis estadístico de las mediciones obtenidas en la ciudad de Quito, en el sector de la Jipijapa, indican que la calidad del aire se encuentra bajo los límites permisibles establecidos por las normas internacionales publicadas por US-EPA.
- h)** En el caso de las cuevas de investigación científica, encontramos que la concentración de CO₂ supera el valor referencial de la media atmosférica de 400 partes por millón. La Densidad de Polvo, sin embargo es menor, a la media que presentó la ciudad de Quito.
- i)** El uso de soluciones tecnológicas capaces de medir, transmitir en tiempo real y procesar la información recolectada; haciendo uso de herramientas como el Internet y sistemas de información modernos, brindan un nuevo enfoque en el trabajo de control de la calidad de aire en las ciudades y su difusión.
- j)** Este prototipo está diseñado principalmente para que, los responsables de la gestión de la calidad del aire en la ciudades, tomen mejores decisiones y puedan optimizar los recursos

disponibles. Es importante recalcar que los usuarios finales de la información recogida por el sistema son los ciudadanos.

7.2 Recomendaciones

- a) Ampliar el desarrollo de los dispositivos electrónicos, empleando sensores más precisos y efectuando las calibraciones de manera técnica en un ambiente de laboratorio en conjunto con expertos ambientales y electrónicos.
- b) Incrementar el número de sensores para obtener mejor información de las distintas zonas de la ciudad, donde no tiene cobertura la red de monitoreo de la ciudad de Quito.
- c) Continuar con la investigación sobre el impacto que tienen, en el bienestar y calidad de vida de la población de la ciudad de Quito, este tipo de soluciones tecnológicas, capaces de interactuar con el medio físico que los rodea.

7.3 Trabajo Futuro

Un futuro crecimiento del prototipo incluiría dos iniciativas. La primera, efectuar las mediciones en un período de tiempo mayor y en más puntos de referencia de la ciudad. La segunda se enfoca en la elección de sensores más precisos que puedan medir otros parámetros de la calidad del aire y que permitan comparar los resultados con los datos de medición especializada.

BIBLIOGRAFÍA

Álvarez García, A., de las Heras del Dedo, R., & Lasa Gómez, C. (2012). *Métodos Ágiles y Scrum*. Madrid, España: Anaya Multimedia.

Arduino. (2014). *Arduino Website*. Obtenido de <http://www.arduino.cc>

Arduino Shields. (2014). *Arduino - HomePage*. Obtenido de <http://arduino.cc/en/pmwiki.php?n=Main/ArduinoShields>.

Arduino-Arts. (2014). *Arduino Arts. What is Arduino*. Obtenido de <http://arduinoarts.com/what-is-arduino/>

ArduinoXbeeShield. (2014). *Arduino-ArduinoXbeeShield*. Obtenido de <http://arduino.cc/en/Main/ArduinoXbeeShield>.

Banzi, M. (2011). *Getting Started with Arduino*. Sebastopol: O'Reilly.

Chromatic. (2013). *Extreme Programming Pocket Guide*.

Daum, B. (2004). *Professional Eclipse 3 para desarrolladores Java*. (2004 ed.). Alemania.

DFRobot. (2014). *DFRobot Drive the Future*. Obtenido de http://www.dfrobot.com/wiki/index.php/CO2_Sensor_SKU:SEN0159:www.dfrobot.com

Hansen, G. W. (1997). *Diseño y Administración de Bases de Datos* (II ed.). Prentis Hall.

IBERTMUTUAMUR. (2013). *Calidad de Aire Interior - Normas Ashrae y US EPA*. Obtenido de http://www.seguridadysalud.ibermutuamur.es/IMG/pdf/Calidad_del_aire_interior.pdf

IBM. (2008). *RESTful Web services: The basics*. Obtenido de <http://www.ibm.com/developerworks/library/ws-restful/>

IEEE. (10 de 10 de 2014). *IEEE Standards*. Obtenido de www.ieee.org

Internacional Agency for Reseach on Cancer. (17 de Octubre de 2013). Outdoor air pollution a leading environmental cause of cancer deaths. Ginebra, Suiza, Suiza.

Kimel, P. (2006). *Manual de UML*. México: Mc Graw Hill.

MAE, 2. (2014). *Programa Calidad del Aire Ecuador Fase III*. Obtenido de <http://www.ambiente.gob.ec/programa-calidad-del-aire-fase-iii/>

MySQL. (2014). *MySQL 5.0 Reference Manual*. Obtenido de <http://dev.mysql.com/doc/refman/5.0/en/index.html>

Oracle. (2014). *The Java EE 7 Tutorial*. Obtenido de <https://docs.oracle.com/javaee/7/tutorial/doc/overview003.htm>

Pressman, R. (2010). *Ingeniería del Software*. México, México, México: McGraw-Hill,.

Ricardo, C. (2009). *Bases de Datos*. México, México: Mc Graw Hill.

Scrum-Alliance. (2014). *Scrum Alliance*. Obtenido de <https://www.scrumalliance.org/why-scrum/core-scrum-values-roles>

Secretaría de Ambiente Quito. (Abril de 2014). *Red Metropolitana Ambiental*.
Obtenido de http://190.152.144.74/paginas/web/menu_t1.html

Sparkfun-Sharp. (2006). *Sharp*. Obtenido de
https://www.sparkfun.com/datasheets/Sensors/gp2y1010au_e.pdf.

Sun Microsystems, Oracle. (12 de Septiembre de 1997). *Java Code Conventions*. Obtenido de
<http://www.oracle.com/technetwork/java/codeconventions-150003.pdf>

US-EPA. (20 de 11 de 2014). *Environmental Protection Agency*. Obtenido de
Concentración Global de CO2 en la atmósfera en el tiempo.:
http://www.epa.gov/climatechange/images/indicator_downloads/ghg-concentrations-download1-2014.png

WildFly. (2014). *WildFly Documentation*. Obtenido de
https://docs.jboss.org/author/display/WFLY8/Documentation?_sscc=t

HOJA DE LEGALIZACIÓN DE FIRMAS

ELABORADO POR

Carrera Arízaga Diego Mauricio

**DIRECTOR DE LA CARRERA DE INGENIERÍA EN SISTEMAS EN
SISTEMAS E INFORMÁTICA**

Ing. Mauricio Campaña Ortega MSc.

Sangolquí, Enero de 2015