UNIVERSIDAD TÉCNICA DEL NORTE

FACULTAD DE INGENIERÍA EN CIENCIAS APLICADAS

CARRERA DE INGENIERÍA EN ELECTRÓNICA Y REDES DE COMUNICACIÓN

"SISTEMA EXPERTO EN EL TRATAMIENTO DE INFORMACIÓN DE SENSORES PARA VISUALIZACIÓN DE ALERTAS."

TRABAJO DE GRADO PREVIO A LA OBTENCIÓN DEL TÍTULO DE INGENIERÍA EN ELECTRÓNICA Y REDES DE COMUNICACIÓN

AUTOR: JEFFERSON VINICIO RUIZ FUEL

DIRECTOR: MSC. JAIME ROBERTO MICHILENA CALDERÓN

Ibarra-Ecuador

2017

AUTORIZACIÓN DE USO Y PUBLICACIÓN A FAVOR DE LA UNIVERSIDAD TÉCNICA DEL NORTE.

1. IDENTIFICACIÓN DE LA OBRA.

La Universidad Técnica del Norte dentro del proyecto Repositorio Digital Institucional, determinó la necesidad de disponer de textos completos en formato digital con la finalidad de apoyar los procesos de investigación, docencia y extensión de la Universidad.

Por medio del presente documento dejo sentada mi voluntad de participar en este proyecto, para lo cual pongo a disposición la siguiente información.

DATOS DEL CONTACTO	
Cédula de identidad	0401556113
Apellidos y Nombres	Ruiz Fuel Jefferson Vinicio
Dirección	Av. 17 de julio y panamericana norte
E-mail	jruizf@utn.edu.ec
Teléfono móvil	0986637033
DATOS DE LA OBRA	
Título	SISTEMA EXPERTO EN EL TRATAMIENTO DE INFORMACIÓN DE SENSORES PARA VISUALIZACIÓN DE ALERTAS.
Autor	Ruiz Fuel Jefferson Vinicio
Fecha	2017
Programa	Pregrado
Título	Ingeniero en Electrónica y Redes de Comunicación
Director	Ing. Jaime Michilena Calderón, MSc.

2. AUTORIZACIÓN DE USO A FAVOR DE LA UNIVERSIDAD.

Yo, Ruiz Fuel Jefferson Vinicio, con cédula de identidad Nro. 0401556113, en calidad de autor y titular de los derechos patrimoniales de la obra o trabajo de grado descrito anteriormente, hago entrega del ejemplar respectivo en forma digital y autorizo a la Universidad Técnica del Norte, la publicación de la obra en el Repositorio Digital Institucional y uso del archivo digital en la Biblioteca de la Universidad con fines académicos, para ampliar la disponibilidad de material y como apoyo a la educación, investigación y extensión, en concordancia con la ley de Educación Superior Artículo 144.

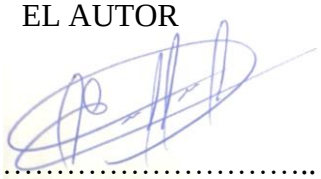
3. CONSTANCIAS.

Yo, RUIZ FUEL JEFFERSON VINICIO declaró bajo juramento que el trabajo aquí escrito es de mi autoría; y que este no ha sido previamente presentado para ningún grado o calificación profesional y que he consultado las referencias bibliográficas que se presentan en este documento.

A través de la presente declaración cedo mis derechos de propiedad intelectual correspondiente a este trabajo, a la Universidad Técnica del Norte, según lo establecido por las leyes de propiedad intelectual, reglamentos y normatividad vigente de la Universidad Técnica del Norte.

En la ciudad de Ibarra, 2017.

EL AUTOR



……..
Ruiz Fuel Jefferson Vinicio
CI: 0401556113

UNIVERSIDAD TÉCNICA DEL NORTE

FACULTAD DE INGENIERÍA EN CIENCIAS APLICADAS

CERTIFICACIÓN.

MAGISTER JAIME MICHILENA, DIRECTOR DEL PRESENTE TRABAJO DE TITULACIÓN CERTIFICA:

Que, el presente trabajo de Titulación "SISTEMA EXPERTO EN EL TRATAMIENTO DE INFORMACIÓN DE SENSORES PARA VISUALIZACIÓN DE ALERTAS." Ha sido desarrollado por el señor Ruiz Fuel Jefferson Vinicio bajo mi supervisión.

Es todo en cuanto puedo certificar en honor de la verdad.

1002198438

DIRECTOR

UNIVERSIDAD TÉCNICA DEL NORTE

FACULTAD DE INGENIERÍA EN CIENCIAS APLICADAS

CESIÓN DE DERECHOS DE AUTOR DEL TRABAJO DE GRADO A FAVOR DE LA UNIVERSIDAD TÉCNICA DEL NORTE.

Yo, Ruiz Fuel Jefferson Vinicio, con cédula de identidad Nro. 0401556113, manifiesto ni voluntad de ceder a la Universidad Técnica del Norte los derechos patrimoniales consagrados en la Ley de Propiedad Intelectual del Ecuador artículos 4, 5 y 6, en calidad de autor del trabajo de grado con el tema: SISTEMA EXPERTO EN EL TRATAMIENTO DE INFORMACIÓN DE SENSORES PARA VISUALIZACIÓN DE ALERTAS. Que ha sido desarrollado con propósito de obtener el título de Ingeniero en Electrónica y Redes de Comunicación de la Universidad Técnica del Norte, quedando la Universidad facultada para ejercer plenamente los derechos cedidos anteriormente.

En mi condición de autor me reservo los derechos morales de la obra antes citada. En concordancia suscribo en el momento que hago entrega del trabajo final en formato impreso y digital a la Biblioteca de la Universidad Técnica del Norte.

…….

Ruiz Fuel Jefferson Vinicio

0401556113

Ibarra, 2017

Agradecimiento.

A la Universidad Técnica del Norte y sus autoridades, por acogerme en su casona durante este proceso de estudios. A la facultad de ingeniería en ciencias aplicadas y a la carrera de ingeniería en electrónica y redes de comunicación, así como también a todos mis profesores quienes se encargaron de guiarme durante el proceso de formación académica.

Especial agradecimiento a mi director de tesis el Ing. Jaime Michilena Calderón MSc. Quien fue parte primordial en la realización de este proyecto de investigación, ya que con sus sabios consejos supo guiarme al término de este trabajo de grado.

A mi madre por confiar en mí y brindarme la mejor educación, por enseñarme que con esfuerzo, dedicación y constancia todo es posible.

A mi familia y amigos quienes se encargaron de motivarme cada día, para culminar mi meta.

Dedicatoria.

Principalmente dedico este trabajo a Dios por ser la guía en mi camino, y darme fuerza para luchar cada día. A mi madre Fabiola, por ser el pilar primordial en mi vida, por sus consejos sabios, su amor incondicional, su cariño admirable y su dedicación infinita hacia mí y por ser quien me encamino y me acompañó en este hermoso trayecto. A mis hermanos Wendy y Joel porque los amo infinitamente. A Edgar por su esfuerzo y dedicación hacia nosotros. A Gabriela por acompañarme y motivarme cada día. Soy la consecuencia del enorme esfuerzo que realizan cada día.

Jefferson Vinicio Ruiz Fuel

Índice.

Índice de Figuras.

Índice de Tablas.

Resumen.

El presente proyecto consiste en el desarrollo de un sistema experto en el tratamiento de información de sensores para visualización de alertas, el cual será aplicado para combatir incendios forestales dentro del bosque protector Guayabillas de la ciudad de Ibarra, con el propósito de servir como sistema de alerta temprana y evitar que afecte a la flora y fauna existente en la zona intervenida, así como también la propagación hacia zonas urbanas.

Para la realización de este proyecto se tomaron en cuenta diferentes estudios de los cuales se obtuvo rangos de tolerancia y factores de sensibilidad de sensores para realizar una toma de datos efectiva, y poder analizar el entorno en rangos de tiempo específico de 30 segundos, y generar alertas al realizar una comprobación de 3 periodos equivalente a 1 minuto y 30 segundos, estos datos son almacenados en tablas las cuales pueden consultarse y organizarse dependiendo de las necesidades existentes para la realización de un informe forestal especializado.

Los datos provenientes del bosque protector serán almacenados en una base de datos MySQL, desde donde el sistema experto se encarga de verificar, almacenar y comparar esta información y presentarla en una interfaz amigable generando alertas visuales y auditivas para evitar inconvenientes, de interpretación.

Posterior a la notificación de alerta generada por el sistema, el encargado de monitoreo toma comunicación con el Sistema Integrado de Seguridad (ECU 911), desde donde se gestiona el correcto despliegue de los organismos de socorro, sea el caso preventivo o accionario, dependiendo del estado en el cual se encuentre sometida la zona.

Abstract.

The present project consists in the development of an expert system in the treatment of sensor information for displaying alerts, which will be applied to combat forest fires inside the protective forest Guayabillas of the city of Ibarra, with the purpose of serving as an early warning system and preventing it from affecting the flora and fauna in the affected area, as well as the urban

In order to carry out this project, different studies were taken into account. tolerance ranges and sensor sensitivity factors were obtained to perform a effective data collection, and to be able to analyze the environment in specific time thirty, seconds and generate alerts when performing a three-period test equivalent to 1 minute and 30 seconds, these data are store in tables the which can be consulted and organized depending on the needs the implementation of a specialized forest report.

Data from the protective forest will be store in a database MySQL, where the expert system is responsible for verifying, storing and comparing this information and present it in a friendly interface by generating visual and hearing aids to avoid drawbacks, of interpretation.

After the alert notification generated by the system, the monitoring officer will communicate with the Integrated Security System (ECU 911), from where the correct deployment of relief agencies, whether depending on the state in which the area is subject.

CAPÍTULO 1. ANTECEDENTES.

En este capítulo se encuentra de manera detallada la información necesaria para el desarrollo del presente trabajo de titulación, se muestra las bases necesarias para expresar la necesidad del estudio de un sistema experto para visualización de alertas, tomando como base información de sensores, se encuentra el tema, la problemática, los objetivos, el alcance y la justificación con el fin de sustentar la investigación.

1.1. Tema.

SISTEMA EXPERTO EN EL TRATAMIENTO DE INFORMACIÓN DE SENSORES PARA VISUALIZACIÓN DE ALERTAS.

1.2. Problema.

En la actualidad contamos con diferentes tipos de redes de sensores y plataformas con las cuales podemos visualizar los datos recolectados de las mismas, pero estas no se encuentran adaptadas al medio donde van a ser empleadas. Las plataformas en el mercado son limitadas en cuanto a sus funcionalidades y medios disponibles, y aquellas que tratan de cumplir con estas características tienen un costo elevado en cuanto a licencias, lo que conlleva a generar grandes costos de implementación en el caso de requerir la utilización de más sensores de los permitidos. (CIIFEN, 2014)

1.3. Objetivos.

1.3.1. Objetivo General.

Implementar un sistema experto, que mediante el procesamiento de información se encargue de la lectura de datos en redes WSN, para permitir a los usuarios visualizar estos valores de acuerdo a las necesidades del entorno a ser utilizada y generar alertas.

1.3.2. Objetivos Específicos

Analizar la bibliografía referente a sistemas expertos para la adquisición de datos, por medio de investigación y visualización de sus principales recursos, para toma de decisiones en sistemas de alerta temprana.

Determinar la normalización de base de datos, mediante el análisis de necesidades en el entorno de incendios forestales, para evitar la redundancia de datos y proteger la integridad de los mismos.

Identificar la lógica de negocio mediante el procesamiento de datos recabados y almacenados, para determinar cómo esta información puede ser creada, analizada y demostrada

Diseñar una plataforma web para redes WSN, que permita la adaptación del sistema a las necesidades del entorno a ser analizado, identificando el lenguaje de programación a utilizar para el desarrollo, con la ayuda del estándar IEEE 29148 según los requisitos establecidos.

1.4. Alcance.

El proyecto pretende brindar la normalización de base de datos, el procesamiento, y la visualización de información obtenida por una red WSN. Para lo cual se revisará la bibliografía existente referente a sistemas expertos para adquisición de datos, y su correcta toma de decisiones en sistemas de alerta temprana.

Realizando un estudio del estado del arte determinar cuáles son las características básicas que los sistemas expertos brindan al usuario, para poder tomar como punto de partida estos servicios y mejorarlos, haciendo que estos se adapten a nuestro caso de estudio, y sirvan de forma efectiva para sistemas de alerta temprana.

Mediante la normalización de base de datos, identificar las variables necesarias a ser aplicadas para evitar la redundancia de datos y proteger la integridad de los mismos. Ubicándolos de manera efectiva en una base de datos donde serán almacenados previo a su procesamiento por el sistema experto, mediante la lógica de negocio.

Por medio de la lógica de negocio generar el procesamiento de datos recabados y almacenados, para determinar cómo esta información puede ser creada, analizada y mostrada a los usuarios, aplicando codificación de las reglas para generar alertas, procesarlas y enviarlas al administrador del sistema para su correcta utilización. Al ser un sistema de alerta temprana las alertas deberán ser generadas al detectar cambios bruscos en los valores normales de funcionamiento, generando con esto un umbral de tolerancia.

Mediante la aplicación del estándar IEEE 29148, se utilizará la metodología para la determinación del software más idóneo para la plataforma. El estándar contiene disposiciones para

los procesos y los productos relacionados con la ingeniería, así como los requisitos para los sistemas, productos de software y servicios (IEEE, 2011).

Se evaluará su funcionamiento, en cuanto a tiempos de envió de alerta y saturación del sistema, cumpliendo con requisitos mínimos para la correcta aplicabilidad tomando en cuenta que será aplicado a sistemas de alerta temprana donde los tiempos de reacción son de suma importancia.

El envió de alertas de manera apropiada es de mucha importancia para los organismos de socorro. Ya que si estos organismos son informados a tiempo pueden generar de manera oportuna maniobras operativas para mitigar los daños que podrían ocasionarse.

Se analizará la factibilidad tecnológica presente en el proyecto basado en software libre, para saber si cumple con los requerimientos de procesamiento de gran cantidad de información, brindando las garantías necesarias para su utilización.

Serán evidenciadas las debidas conclusiones y recomendaciones, encontradas en la realización del sistema. Para poder con esto comprobar su aplicabilidad en el sistema de alerta temprana para el bosque protector Guayabillas.

1.5. Justificación

Contribuyendo con el Objetivo 7 del Plan Nacional del Buen vivir que dicta “La revolución Ecológica, apuesta por la transformación productiva bajo un modelo ecoeficiente con mayor valor económico, social y ambiental. En este sentido, plantean como prioridades la conservación del patrimonio y recursos naturales, así como también la inserción de tecnologías ambientalmente limpias” (Plan Nacional Del Buen Vivir, 2013-2017).

La visualización de datos en redes de sensores, de forma temprana es de gran ayuda para mitigación de incendios forestales, por lo que con la implementación de una plataforma que permita al usuario visualizar lo que está ocurriendo este podría tomar acciones oportunas, contribuyendo con la preservación del bosque protector Guayabillas.

Las plataformas existentes para redes WSN con las cuales se puede contar actualmente en el mercado limitan las funciones del servicio y las capacidades potenciales que brindan, por lo que se complica su utilización si es necesario un procesamiento de datos, almacenamiento de información o mostrar valores históricos. Estas capacidades se encuentran ampliadas en versiones pagadas que resultan en costos elevados a la hora de la aplicación del proyecto.

Existen plataformas donde se permite mostrar datos de redes WSN, pero estos datos proporcionados por el usuario a este tipo de plataformas, no cuentan con una garantía de si los mismos son públicos o privados, lo que lleva a la necesidad de contar con una plataforma propia donde se garantiza que los datos obtenidos no son compartidos a otras personas u organizaciones, si así el usuario lo desea.

La necesidad de implementación de un sistema experto permitirá generar una lógica de negocio la cual será de gran ayuda al usuario para resolver inconvenientes encontrados en las variaciones de los sensores, donde se permita al usuario el manejo de los valores recabados en el bosque protector Guayabillas. Contribuyendo al sistema de alerta temprana y ayudando a generar alertas, para su correcto despliegue de los estamentos de socorro, o instituciones encargadas de la mitigación de incendios forestales.

CAPÍTULO 2. FUNDAMENTOS TEÓRICOS

En este capítulo se realiza la recopilación de información sobre las principales características básicas, de sistemas expertos, así como el estudio del estado del arte, para determinar la lógica del negocio más idónea para nuestro caso de estudio.

2.1. Sistema experto

Sistema experto es un programa informático creado para resolver problemas complejos en un dominio específico, cuyo principal propósito es reproducir las capacidades de razonamiento utilizadas por un especialista en la solución de un problema particular, la idea de creación de estos sistemas es el dominio de información para la toma decisiones en casos concretos, simulando el diagnóstico de un experto en base a su conocimiento, esto ha venido siendo desarrollado desde los años 70, en donde se dio cabida a la Inteligencia artificial (Blázquez, 2001).

La inteligencia artificial está basada en sistemas que convierten el conocimiento de un experto en un tema específico y esto puede traducirse a código. Este código se puede combinar con otros códigos y se utiliza para responder a las preguntas presentadas a través de un ordenador.

Un sistema experto en diagnóstico médico requeriría como datos los síntomas del paciente, los resultados de análisis clínicos y otros hechos relevantes, y, utilizando estos, buscaría en una base de datos la información necesaria para poder identificar la correspondiente enfermedad. Un Sistema Experto de verdad, no solo realiza las funciones tradicionales de manejar grandes cantidades de datos, sino que también manipula esos datos de forma tal que el resultado sea inteligible y tenga significado para responder a preguntas incluso no completamente especificadas (Stevens, 1985).

La toma de decisiones es la base primordial en un sistema experto, cuya función es responde a preguntas y problemas en tiempo real sin que un especialista tenga que ser involucrado, por tal motivo, su implementación es indispensable en sistemas de alerta temprana.

2.1.1. Componentes de sistemas expertos:

Los componentes de un sistema experto se encuentran basados en tres grupos los cuales serán explicados a continuación y mostrados en la Figura 1, (Pérez M. A., 2015):

- **Base de conocimiento:** Está basada en el conocimiento y dominio de un área en específico, Se requiere que los datos a mostrar dependan en gran manera a la recopilación de información exacta y fiable.
- **Máquina de inferencia:** El uso de reglas específicas y eficientes, hacen que el motor de inferencias sea esencial en la resolución de problemas sin defectos. Se encarga de la resolución de normas en conflicto dado cuando diferentes reglas aplican a un caso particular.
- **Interfaz de usuario:** La interfaz de usuario va a interactuar directamente con el encargado de la monitorización y visualización de alertas. El usuario no debe ser necesariamente un experto en inteligencia artificial o en un área específica ya que el motor de inferencias será el encargado de generar los cálculos.

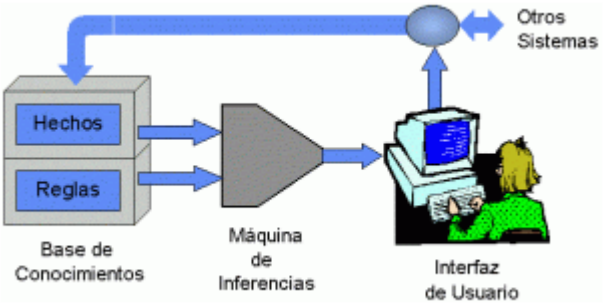


Figura 1 Estructura Sistema Experto
Fuente: (Pérez M. A., 2015)

2.1.2. Características de un sistema experto.

Dado que el sistema debe actuar como un especialista, debe reunir cierto tipo de características que en lo posible se asemejan a las respuestas dadas por un humano, entre las principales se encuentran las mostradas a continuación (Engelmore, 2010):

- Procesamiento de información
- Fiabilidad, es imprescindible que los resultados sean confiables
- Solidez en el dominio de su aplicación
- Envió de alertas y capacidad de resolución de problemas

El cumplimiento de estas características permite un óptimo funcionamiento del sistema, adquiriendo la capacidad de almacenar datos, la facilidad de conseguir conclusiones lógicas y poder tomar decisiones de acuerdo al entorno y sus diferentes variantes, técnicamente el sistema experto tendrá dentro de sus bases, el conocimiento del especialista y un conjunto de cierto tipo de reglas para poder ser aplicado.

2.1.3. Funciones de un sistema experto

Las principales funciones encontradas en un sistema experto están basadas en la capacidad para la cual fue programado dependiendo de cada caso lo que se puede encontrar se detalla a continuación (Pérez M. A., 2015):

- **Monitorización:** Válida la información presente en el sistema realizando varias comparaciones continuas, lo que le permitirá poder anticiparse a diversos tipos de incidentes.
- **Diseño:** El proceso de detallar de forma clara la creación de un dispositivo que satisface a las características planteadas antes del desarrollo.
- **Planificación:** Llevar a cabo de forma ordenada diversos tipos de acciones para cumplir con el objetivo global.
- **Control:** Función central encargada de la realización de diversos tipos de tareas asignadas como la interpretación diagnóstico y visualización de la información válida para el caso de estudio.
- **Simulación:** Se basa en la creación de modelos basados en hechos, afirmaciones e interpretaciones a fin de poder estudiar de mejor manera el entorno.
- **Recuperación de información:** La capacidad para procesar la información almacenada junto con diversos tipos de reglas preestablecidas para poder alcanzar una posible solución y tratamiento de información, así como la recuperación de la información que será llevada para su correcto análisis y puesta a conocimiento de expertos analistas.

2.2. Sistema de alerta temprana.

Sistema encargado del monitoreo y envió de alertas, el mismo por medio de la información recibida de los diferentes instrumentos encargados de la recolección de datos, es capaz de procesar y ofrecer un diagnóstico en tiempo real para evitar posibles daños y efectos que puedan alterar el medio donde se encuentra aplicado, millones de personas alrededor del mundo han logrado sobrevivir y salvar sus recursos materiales, gracias a la respuesta oportuna proveniente de dicho sistema, permitiendo que los organismos encargados pueden actuar con eficacia. (UNESCO, 2011).

Los sistemas de alerta temprana aplicados en riesgos naturales incluyen los diseños para inundaciones, incendios, terremotos, aludes, tsunamis, tornados, deslizamientos de tierra y la sequía. Existen otros sistemas para una variedad de eventos, incluyendo el lanzamiento de misiles, las condiciones del camino y Estrategia.

Las Naciones Unidas, para la reducción de desastres (EIRD) recomienda que los sistemas de alerta temprana deben contar con los siguientes cuatro componentes (EIRD, 2010):

- **Conocimiento del riesgo:** Los datos deben ser recogidos y analizados de forma sistemática y lleva a cabo evaluaciones de riesgo.
- **Vigilancia y alerta del servicio:** Los sistemas deben estar en su lugar para controlar los riesgos y proporcionar servicios de alerta temprana.
- **Difusión y comunicación:** Riesgo información y mensajes de alerta temprana deben ser entregados.
- **Capacidad de respuesta:** Los sistemas deben estar en su lugar para responder a eventos.

En TI (tecnología de la información), sistemas de alerta temprana se utilizan en una variedad de entornos. La Red de Alerta de Salud (HAN sistema de mensajería) utiliza una variedad de herramientas de comunicación, incluyendo correo electrónico, fax de difusión, televisión y llamadas telefónicas, para alertar a los locales, estatales y nacionales y los medios de comunicación acerca de las amenazas urgentes y acciones necesarias. Los sistemas de alerta temprana para los centros de datos pueden ser utilizados para detectar condiciones potencialmente peligrosas en el medio ambiente físico, así como en los sistemas de hardware y software (UNESCO, 2011).

2.3. Estado del arte.

Con el desarrollo de las diferentes tecnologías hoy en día un punto importante es la recolección y análisis de la información que puede ser suministrada por los diferentes medios en desarrollo, por lo que es necesario contar con plataformas de monitoreo que brinden las capacidades suficientes para un correcto estudio y toma de decisiones.

Esta información puede tener varios fines, entre los que se encuentra la representación visual de los datos, la conservación y almacenamiento para un análisis posterior.

Al contar con plataformas multilenguaje, da una pauta para pensar en los aplicativos mirando hacia ciudades inteligentes, y mejorar el estilo de vida de las personas que habitan en este entorno. Estos aplicativos con sensores pueden tomar los datos ambientales y procesarlos para tomar las acciones necesarias.

En la Figura 2 se observa las etapas importantes para el desarrollo de una plataforma enfocada en visualización de datos, iniciando en la parte inferior con la recolección de información de sensores, en la segunda etapa se encuentra con la comunicación entre ellos y finalmente en la

tercera etapa se encuentra el software donde se desarrolla las reglas necesarias para la visualización de los valores obtenidos.

Figura 2 Etapas Sistema Experto
Fuente *(ubidots, 2016)*

2.4. Plataformas tecnológicas IoT.

A continuación, se presenta una recopilación de información de las principales plataformas utilizadas para la visualización y monitoreo de datos obtenidos por sensores ya sean estos multiplataforma o de un fabricante en específico, se tomó en cuenta las plataformas que cuenten con funciones para incorporar la placa arduino a sus sistemas, ya que de acuerdo a sus funciones puede ser utilizada para diferentes fines específicos.

2.4.1. Ubidots

Servicio brindado en la nube que permite a los desarrolladores almacenar e interpretar la información recolectada por sensores en tiempo real, con la posibilidad de crear aplicaciones IoT, como se muestra en la Figura 3 la conexión funciona con la unión de la nube.

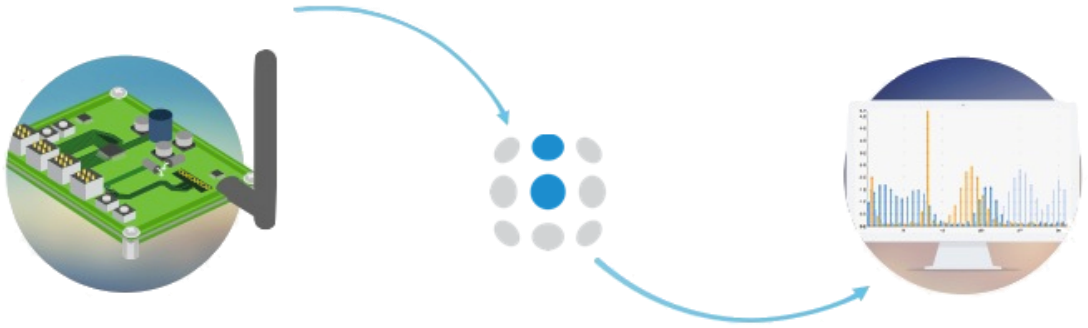


Figura 3 Conexión entre placas y plataforma Ubidots
Fuente: *(ubidots, 2016)*

Con la ayuda de esta herramienta es posible el desarrollo de aplicaciones como telemetría GPS, sistemas de monitoreo y la posibilidad de medir diferentes variables del entorno, y la posibilidad de integrarse con arduino y sus características (ubidots, 2016):

2.4.2.1. Características Ubidots

De la misma manera en que las personas usan un navegador para visitar páginas web a través de URLs, botones y campos de texto, los dispositivos tienen su propia forma de interactuar con los sistemas web. Debido a que los dispositivos no son tan inteligentes como nosotros los humanos, ellos necesitan una "página web especial", con una estructura estandarizada y unos comandos preestablecidos.

"API" significa "Interfaz de Programación de Aplicaciones" (*Application Programming Interface*) y establece cómo diferentes componentes de un sistema de software, deben interactuar entre ellos. En este caso, API específica la interacción entre los dispositivos y Ubidots, a continuación, se expresa su forma de integración y aplicación (ubidots, 2016):

- **Integración:** Un token es una especie de contraseña para que nuestro dispositivo pueda autenticarse de manera segura ante el API de Ubidots. Cuenta con gran infinidad de dispositivos que pueden ser acoplados a la aplicación entre los que se encuentran, Módulos basados en Microchip, Módulos basados en Arduino, Raspberry Pi / Linux Embebido, Electric Imp, Android, Spark.io, Tessel.
- **Aplicación:** Una de las aplicaciones comúnmente más utilizada por su amplio número de dispositivos compatibles con lo cual se pueden generar aplicaciones importantes para desarrollo de proyectos sean estos de aula o profesionales, aplicados a entornos ambientales.

  Una de las plataformas más completas para el desarrollo de entornos IoT, cuenta con la capacidad de poder ser implementada en dispositivos como Arduino o Raspberry Pi, se puede generar diversos tipos de proyectos aplicados en ambientes diversos y complejos, brindando la capacidad de poder visualizar y almacenar esta información de manera dinámica, brindando beneficios al usuario.

2.4.2. Xively

En sus inicios denominada Pachube fue una plataforma con sede en Reino Unido, la se encontraba de forma gratuita basada en web para mostrar los datos de sensores en tiempo real. Actualmente cambió su nombre a Xively la que ha introducido una versión de pago está disponible para empresas que deseen monitorear sus productos conectados a internet.

2.4.2.2. Características Xively

Cuenta con características que le permiten al desarrollador la posibilidad de conectar aplicaciones, dispositivos, datos y lugares todo en una nube, con los beneficios de crear soluciones exponenciales.

La plataforma permite publicar los datos recogidos por distintos sensores (como pueden ser sensores de humedad, temperatura, gases, luminosidad, radiación, etc.) mediante gráficas en tiempo real y widgets. (xively, 2017):

- **Integración:** Es compatible con muchas combinaciones de software y hardware necesarios para crear los productos. Como lo son las bibliotecas oficiales quienes manejan lenguajes y plataformas Java, Ruby, etc.
- **Seguridad:** Proporciona seguridad en toda la plataforma para asegurar la integridad del producto, sofisticadas herramientas de depuración, incluyendo la monitorización en tiempo real de mensajes HTTP.

  Selectivamente y de forma segura el intercambio de datos del dispositivo con otras aplicaciones o dispositivos en sus términos. Una vez publicado, el producto se convierte en parte de la "nube pública".

Xively cuenta con una solución flexible y segura, junto con fuertes socios comerciales, para conectar todo tipo de dispositivos a través de industrias y entornos. Ya sea que su producto es un detector de humo de aplicación casera, o grandes contenedores de transporte. Xively puede ayudarle a navegar el proceso con la consideración para sus necesidades únicas.

2.4.3. Exosite

Exosite (Murano) simplifica la complejidad de la conexión de productos con una plataforma que fue especialmente diseñada por los ingenieros que entienden cómo se desarrollan los productos, fabricados y desplegados. Con las API de fácil uso y herramientas integradas para gestionar la funcionalidad crítica IO, Murano permite conectarse en gran volumen a implementaciones de productos que son seguros y escalables (exosite, 2017):

2.4.3.1. Características Exosite

- **Dispositivo de gestión:** El soporte integrado para gestionar el estado del dispositivo, implementando actualizaciones de firmware, y control de versiones. Las actualizaciones pueden ser simultáneas a todos los dispositivos o dinámicamente a cada uno de ellos.
- **Conectividad de los dispositivos y Seguridad:** Una capa de conectividad de dispositivo que permite a los desarrolladores conectar rápidamente dispositivos y comunicar datos a través de conexiones cifradas usando el protocolo TLS estándar de la industria.
- **Dispositivo de aprovisionamiento:** Cuenta con una interfaz de aprovisionamiento escalable y segura que proporciona a los fabricantes la flexibilidad necesaria para adaptarse a la fabricación, y aprovisionamiento de aplicaciones individuales.
- **API:** Todos los sistemas de producción tienen 24/7 monitoreo desde múltiples ubicaciones, con personal preparado. Servidores y centros de datos los cuales proporcionan una copia de seguridad distribuida en todo el mundo. Cada centro de datos está diseñado para manejar grandes cantidades de tráfico a través de una

agrupación de servidores, proporcionando alta disponibilidad con ningún punto único de fallo.

- **Integración:** Bibliotecas de desarrollo de código abierto para C, C ++, Python, Java, .NET, Node, Go, y más para ayudar al desarrollo de JumpStart. Es una de las compañías mejor posicionadas en desarrollo de software, para recolección de datos de sensores incorporados en objetos físicos.

2.4.4. Thingspeak

ThingSpeak es una plataforma abierta de aplicaciones, diseñada para permitir conectar personas con objetos. Se caracteriza por ser una plataforma Open Source con una API para almacenar y recuperar datos de los objetos usando el protocolo HTTP sobre Internet o vía LAN (Local Area Network) (Thingspeak, 2017):

Se trata de una plataforma basada en Ruby on Rails 3.0 (RoR), este es un framework de aplicaciones web de código abierto basado en Ruby, cuya arquitectura está basada en el Modelo Vista Controlador (MVC). Se caracteriza por su simplicidad a la hora de programar aplicaciones del mundo real, escribiendo menos código y con una configuración mucho más sencilla que otros frameworks. Otra de las características que hacen de RoR un framework perfecto para el desarrollo de aplicaciones es que permite el uso de meta programación, haciendo que su sintaxis sea más legible y llegué a un gran número de usuarios.

La aplicación incluye todo lo necesario para poder empezar a trabajar, desde una aplicación web en la que se puede gestionar usuarios, gestionar claves de API, gestión de canales y cartografía.

2.4.4.1. Características ThingSpeak

Destacan algunos puntos importantes en toda plataforma tales como su API, App (si las tiene), integración, hardware:

Un punto importante a la hora de desarrollar cualquier proyecto es encontrar un API disponible de forma sencilla para que el desarrollador tenga los mecanismos necesarios para el desarrollo de la aplicación.

En este caso, ThingSpeak dispone de una API la misma que se encuentra disponible en GitHub para su descarga en un servidor propio. Es totalmente abierta, por lo que también se puede modificar su código fuente original y así contribuir a la comunidad con nuevas características, un principio básico en toda plataforma Open Source (Thingspeak, 2017):

- **Canales:** La forma que tiene esta plataforma de almacenar y publicar los datos es a través de los "Channels" (Canales). Su creación es muy simple y rellenando una serie de datos está disponible sin mayor complicación.
- **Plugins:** Para extender la funcionalidad del sitio brinda la oportunidad de desarrollar plugins. Estos ofrecen la posibilidad de crear aplicaciones de forma nativa en la plataforma ThingSpeak.

  Soporta HTML, CSS y JavaScript como lenguajes de programación. Al igual que los canales los plugins pueden ser público o privados según sean nuestras necesidades. Ofrece la posibilidad de usar Google Gauge Visualization, facilitando la posibilidad de visualizar los datos de una forma rápida y amigable, con un nivel de personalización muy amplio.

- **Integración:** Uno de los puntos fuertes en cualquier plataforma IoT, es que permite una amplia integración con diversos dispositivos Hardware y software. En este caso ThingSpeak permite la integración de su plataforma con, Arduino, Raspberry Pi, Electric Imp, Móviles / Aplicaciones web, Análisis de datos con MATLAB.

2.4.5. Carriots

Carriots es una plataforma española en la nube, que da un servicio PaaS orientada a proyectos IoT y máquina a máquina (M2M). Al no ser una plataforma libre permite registrar un máximo de 10 dispositivos de forma totalmente gratuita, pero con algunas restricciones de funcionalidad. Para registrar más de 10 dispositivos y hacer uso de todas las funcionalidades que ofrece la plataforma sin ningún tipo de restricción es necesario pagar (Carriots, 2017):

2.4.5.1. Características Carriots

Carriots ofrece una API propia basada en REST, (la cual puede descargarse desde Github) para poder comunicarse con la plataforma y gestionar los datos de una forma sencilla. REST se caracteriza principalmente por la existencia de recursos que pueden ser accedidos utilizando un identificador global. Para la manipulación de estos recursos, la comunicación de red se usa el estándar HTTP.

Respecto a los mecanismos de seguridad de la REST API destaca sobre todo el uso de API Keys se caracteriza por un Token que sustituye al clásico usuario y contraseña, Checksum es un hash de datos más fecha más una clave simétrica y HTTPS (Carriots, 2017):

- **Integración:** El punto importante en este tipo de plataformas es evaluar la integración que ofrece con sistemas externos. En este caso ofrece la integración con sistemas externos, a través de su API REST, el PUSH de datos y con peticiones HTTP/s o sockets. Con estos mecanismos se puede tener integración con bases de datos, ERPs, CRM, data warehouse, etc.
- **Hardware:** Carriots ofrece un amplio catálogo de hardware compatible, completo que soporta cualquier tipo de hardware que haya en el mercado, como los principalmente ocupados como lo son, Arduino, Raspberry Pi, Electric IMP, incluyendo otro Hardware (Siempre y cuando pueda comunicarse con su API REST)

Esta plataforma no está, orientada a prototipos, aunque trae una opción gratuita, sino que claramente su línea de negocio son aplicaciones reales basadas en el IoT en los distintos ámbitos en los que puede aplicarse.

2.5. Tabla resumen plataformas IoT

A continuación, en la Tabla 1 se observa un resumen de las principales características ofertadas por las distintas plataformas investigadas.

Tabla 1 Plataformas IoT

Plataformas IoT	**Interfaz**	**Integración**	**Aplicación**
Ubidots	Web, API REST	Mediante la utilización de un token, se autentica con la aplicación. Cuenta con la posibilidad de acoplarse con una infinidad de dispositivos, entre los más utilizados, arduino, Raspberry Pi, Android.	Ampliamente utilizado en entornos ambientales, disponible para realizar proyectos de aula, reales, o experimentales
Xively	Web, API REST	Compatible con la mayoría de software y hardware necesarios para crear productos. Cuenta con bibliotecas oficiales como (Java, Ruby, etc)	Cuenta con la mayor capacidad de escalabilidad gracias a Xively MQTT,
Exosite	Web, API REST	Cuenta con servidores multi-arrendatario cuenta con un monitoreo de 24/7 con múltiples dispositivos bibliotecas de código abierto para C, C ++, Python, Java, .NET, Node, Go, y más para ayudar al desarrollo de JumpStart.	Cuenta con diferentes proyectos galardonados, Una de las mejores compañías en desarrollo de software para recolección de datos de sensores incorporados en objetos físicos
Thingspeak	Web, API REST, código totalmente abierto	Compatible con gran cantidad de dispositivos tanto en hardware como en software entre los más importantes, Arduino, Raspberry Pi, Redes Sociales, Análisis de datos con MATLAB	Se encuentra en su gran mayoría proyectos profesionales, Por lo que principalmente está orientado al mundo del Smart Home.
Carriots	Web, API REST	Ofrece la integración con sistemas externos, a través de su API REST, el PUSH de datos y con peticiones HTTP/s o sockets. Con estos mecanismos se integra con bases de datos, ERPs, CRM, data warehouse, etc.	Puede ser utilizada en diferentes proyectos smart.

Fuente: Elaboración propia.

2.6. Estándar ISO/IEC/IEEE 29148:2011

La norma no sólo define los procesos de la actividad de ingeniería de requisitos, sino que también fórmula exigencias para la documentación de acuerdo a las necesidades. A los efectos de este documento, especialmente digno de mención que la norma proporciona directrices para la aplicación de los procesos relacionados con los requisitos de las normas 12207 y 15288.

Esta norma podría considerarse la madre de todas las "normas de requisitos", ya que da una descripción más bien extensa del dominio de la ingeniería de requisitos. Todos los otros estándares se pueden utilizar junto con éste. La norma comienza con la definición de los conceptos importantes de la ingeniería de requisitos. Si primero define el término requisitos de ingeniería en sí. Luego se habla de las partes interesadas y de las necesidades a ser transformadas en requisitos. Plantea lo que es un "requisito bien formado", propone 3 plantillas que se pueden utilizar para formular requisitos textuales, y proporciona características de "buenas exigencias" y "buenas especificaciones de requisitos". Aparte de la "dos", también proporciona algunos "no hacer" en relación con el lenguaje de los requisitos (por ejemplo, evitar el lenguaje subjetivo) (ISO, 2011):

Posteriormente, se proponen tipos de requisitos (incluido el tipo de requisitos de calidad de las normas ISO 250xx). Para concluir las secciones de concepto, se destaca la relación entre los procesos de requisitos y los "elementos de información de requerimientos" resultantes, especialmente su alcance. El estándar entonces apunta nuestra atención a los procesos de ingeniería de requisitos. Se observa que los dos referidos a los estándares 15288 y 12207 no están obsoletos y reformulados, sino anotados y detallados. Así que el 29148 puede ser visto como una elaboración de los dos estándares.

2.6.1. Relaciones con otras normas

La norma 29148 es una "norma de armonización" que resulta de la armonización de las normas ISO / IEC / IEEE 12207: 2008, ISO / IEC / IEEE 15288: TR 19759, IEEE Std 830, IEEE Std 1233, IEEE Std 1362, ISO / IEC TR 24748-1 e ISO / IEC / IEEE 24765.

2.6.2. Discusión crítica

Parece que el ISO / IEC / IEEE 29148: 2011 es en realidad el estándar que todos los ingenieros de requisitos deben estar familiarizados. No es sólo una colección de los principios más básicos de la ingeniería de requisitos, sino que también indica cómo el requisito de alta calidad se puede lograr. Por lo tanto, es un buen punto de partida para la adquisición de conocimiento, ya que se relaciona con otras normas que detallan ciertos aspectos de la ingeniería de requisitos (por ejemplo, apunta a la serie 250xx para requisitos de calidad) (ISO, 2011):

Finalmente, los ingenieros de requisitos no necesitan leer las normas 15288 y 12207, ya que todas las tareas relevantes para la ingeniería de requisitos se citan en forma original y luego se elaboran. Sin embargo, el estándar es un poco corto con el lector con respecto a las categorías de requisitos. Como esto todavía parece ser un punto de discusión en la comunidad de investigación, el estándar se habría beneficiado al proporcionar declaraciones más fuertes con respecto a un conjunto básico de categorías de requisitos.

El estándar debilita esta parte diciendo solamente qué ejemplos de tipos de requisitos son ("ejemplos importantes", no obstante, pero sería una afirmación más fuerte decir "estos son los tipos de requisitos que son útiles). Así que el 29148 no aborda completamente a los estándares 15288 y 12207.

CAPÍTULO 3. DESARROLLO EXPERIMENTAL Y ELECCIÓN DEL SOFTWARE

En el presente capítulo se realizará la normalización de la base de datos para la obtención de las variables a ser utilizadas en nuestro caso de estudio, como lo son temperatura, humedad relativa, Dióxido de carbono, radiación, humo, ya que los datos seleccionados servirán para determinar los rangos de tolerancia necesarios a ser mostrados en forma de alertas. Se procederá a la elección del software adecuado para la realización del sistema experto, y desarrollo basado en características cumpliendo con los requerimientos establecidos en la norma IEEE 29148.

Por medio de la lógica de negocio generar el procesamiento de datos recabados y almacenados, para determinar, cómo esta información puede ser creada, analizada y mostrada a los usuarios, y mediante la aplicación de las reglas generar alertas, procesarlas y enviarlas al administrador del sistema para su correcta utilización.

3.1. Metodología

Parte importante en el desarrollo de software es la utilización de un sistema metodológico adecuado, el cual brinda una guía para poder alcanzar los objetivos planteados y la mejora de la investigación, siendo necesario procedimientos apropiados para el seguimiento ordenado y sistemático que ayuda a determinar las necesidades para generar una solución.

El método a ser utilizado por su robustez y claridad es el modelo en V, o modelo de 4 niveles, el mismo que en cada nivel genera pruebas de validación, que brindaran una idea clara del estado del sistema. La metodología se encuentra ligada a la implementación el estándar ISO/IEEE 29148 donde se especifica los requerimientos para ingeniería.

3.2. Modelo en V

El modelo en V es una de las metodologías más utilizadas para el desarrollo de proyectos, el cual consiste en etapas de revisión a 4 niveles, donde serán verificadas de acuerdo a lo planteado en cada fase específica, de tal manera que se gestioné la elaboración y progreso del sistema (Rodríguez, 2008):

El modelo en V, trata de dos fases las mismas se encuentran claramente establecidas, la primera fase se encarga en sí del desarrollo del sistema, sus características, especificaciones y requerimientos, en cambio la segunda fase se encarga de la evaluación y verificación de cada uno de los pasos que se dieron en la construcción del sistema.

A continuación, en la Figura 4 se observa, cada uno de los niveles y su respectiva etapa de verificación las cuales tienen concordancia entre la fase desarrollo y la fase de ratificación.

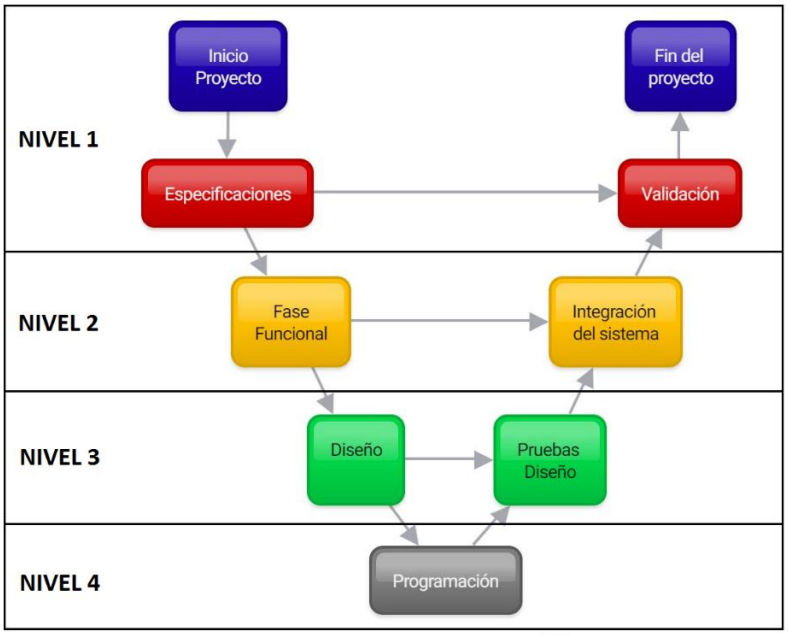


Figura 4 Metodología Modelo en V
Fuente: (Rodríguez, 2008)

A continuación, se especifica las acciones realizadas en cada nivel del modelo.

- Nivel 1 enfocado al usuario, es el punto de partida y fin del proyecto, se especifica los requerimientos que el sistema debe cumplir, para su correcto funcionamiento.
- Nivel 2 en esta parte se determina las especificaciones funcionales que el sistema requiere para implementación y funcionamiento, una vez terminado se evalúa e integra con el sistema inicial.
- Nivel 3 En esta fase se trata lo referente al software y hardware que el sistema requiere para su desarrollo, y sus respectivas pruebas individuales, cumple la función de separar las fases de desarrollo y validación.
- Nivel 4 en esta parte se desarrolla el sistema, es la etapa de programación, que se basa en la parte del diseño, para codificar los requerimientos.

3.3. Análisis

Para determinar las características necesarias a ser implementadas en el sistema experto se realizó un estado del arte en el capítulo 2 en el cual se muestran los principales beneficios que este tipo de plataformas brindan a los usuarios, aunque viéndose limitadas en número de sensores y opciones que se encuentran bloqueadas para versiones pagadas. A partir de ahí se desarrolla un sistema experto que cuente con las opciones para ser aplicado al entorno del caso de estudio, mediante un análisis posterior determinar los requisitos necesarios.

3.4. Situación actual

En la actualidad existen diferentes tipos de redes de sensores, las mismas pueden ser monitoreadas por plataformas especializadas en visualización de datos, los cuales son recolectados por diferentes sensores, el mayor inconveniente en estas plataformas es que no se encuentran adaptadas al medio donde se requiere su monitorización y su correcta representación de la información, en el mercado se encuentra plataformas en la nube que tratan de simular estas características pero tienen un costo comercial elevado por el uso de licencias propietarias

El hecho de que estos sistemas no se encuentren preparados para sistemas de alerta temprana, lleva a que los datos que han sido recabados y almacenados no sean evaluados de manera efectiva, y sea imprescindible la necesidad de un experto que interprete la información mostrada, lo que incurre en costos adicionales a los planteados en el proyecto inicialmente

Los datos obtenidos no solamente deben ser mostrados sino también contar con la capacidad de unir variables para una mayor comprensión por parte del operador y envió de alerta oportuna, cumpliendo con la integración al sistema de alerta temprana, la información recabada será de gran ayuda para llevar a cabo una investigación del medio donde se aplica la red de sensores.

3.5. Propósito del sistema experto

La propuesta presentada se basa en el desarrollo de un sistema experto basado en plataformas de internet de las cosas, con la finalidad de enfocarse al sistema de alerta temprana desplegado en el bosque protector Guayabillas, donde los datos almacenados no cuentan con un sistema de monitoreo y visualización en tiempo real

El sistema experto pretende determinar rangos de valores específicos, los cuales serán mostrados en forma de alertas, asumiendo el rol de un especialista en el área, estos datos recabados serán expuestos gráficamente, a un operador que sin tanto conocimiento podrá interpretar y ayudar en la difusión de los problemas generados.

El estudio se encuentra basado en los requerimientos que el entorno requiere, tomando desde la normalización de una base de datos donde las variables, temperatura, humedad relativa, dióxido de carbono, radiación, humo, son almacenadas de forma tal que no exista una confusión y entre ellas pueden ser mezcladas para generar alertas procesadas.

3.6. Definición de abreviaturas

Para mejor comprensión del texto se utiliza los acrónimos representados en la Tabla 2 los mismos que serán utilizados en la continuación del escrito.

Tabla 2 Definición de Acrónimos y Abreviaturas

Abreviatura	**Descripción**
UTN	Universidad Técnica del Norte
CIERCOM	Carrera de Ingeniería en Electrónica y Redes de Comunicación
FICA	Facultad de Ingeniería en Ciencias Aplicadas
StRU	Documento de especificación de requerimientos del usuario
SyRS	Documento de especificación de requerimientos del sistema
SrSH	Documento de especificación de requerimientos de Software

Fuente: (Barreto, 2014)

3.7. Descripción General

Para su correcta integración los datos son recolectados en un nodo central que se encuentra dentro del bosque protector Guayabillas, desde donde son enviados hacia la Universidad Técnica del Norte, por medio de un radio enlace que conecta Loma de Guayabillas - La Esperanza – Universidad Técnica del Norte.

El punto del enlace en la UTN llega a la Facultad de Ingeniería en Ciencias Aplicadas (FICA), donde se cuenta con un centro de datos ubicado en la planta baja donde será el punto de almacenamiento de los datos que se encuentran siendo recabados por los sensores dentro del entorno estudiado.

3.8. Requerimientos del sistema

Tal como se mencionó en el capítulo 1, se pretende realizar un sistema experto que cuente con las capacidades necesarias para poder mostrar valores de variables como lo son temperatura, humedad relativa, Dióxido de carbono, radiación, humo, así como también diferentes alertas que se presentan con la manipulación de esta información recabada, para una correcta interpretación y pronta respuesta al ser implementada en un sistema de alerta temprana. Con la finalidad de llegar a obtener una plataforma que ayude a mitigar incendios forestales en el bosque protector Guayabillas.

3.8.1. Requerimientos indirectos en el desarrollo del sistema

Se deben tomar ciertas consideraciones previas al momento de la elaboración del sistema experto, respecto a ciertos parámetros importantes para un correcto funcionamiento y utilización por

parte del usuario, en la Tabla 3 a continuación, se muestra los requerimientos necesarios para esta investigación.

Tabla 3 Requerimientos indirectos necesarios para el desarrollo del sistema

Requerimientos Operacionales del sistema				
#	Requerimiento	Prioridad		
		Alta	Media	Baja
SyRS 1	Soporte sistema operativo Open Source	X	-	-
SyRS 2	Conexión a internet	X	-	-
SyRS 3	Capacidad para generar alertas	X	-	-
SyRS 4	Bajo consumo de energía	-	X	-
SyRS 5	Comunicación con otros dispositivos	-	X	-
SyRS 6	Fiabilidad en la lectura de datos	X	-	-
SyRS 7	Eficaz tiempo de respuesta	X	-	-
SyRS 8	Capacidad para funcionar por largos periodos de tiempo	X	-	-

Fuente: Elaboración propia.

Para un correcto funcionamiento e integración del sistema, debe ser cubierto en el desarrollo del sistema experto, cada uno de los requerimientos que han sido planteados, siguiendo sus prioridades de cada uno de ellos.

3.8.2. Requerimientos iniciales del sistema

En la elaboración del sistema experto propuesto, se plantea los requerimientos con los cuales da inicio a las funcionalidades, planteando el comportamiento junto a sus propiedades en la siguiente Tabla 4 se definirá cada una de las funcionalidades necesarias que el sistema debe proporcionar, así

como también las limitaciones existentes de acuerdo con lo especificado en su entorno a ser utilizado.

Tabla 4 Requerimientos iniciales del sistema

Requerimientos Funcionales				
#	Requerimiento	Prioridad		
		Alta	Media	Baja
StRU 1	Monitorización de datos	X	-	-
StRU 2	Interfaz amigable	X	-	-
StRU 3	Manual de configuración	X	-	-
StRU 4	Precisión en la lectura de datos	X	-	-
StRU 5	Estabilidad	X	-	-
StRU 6	Comunicación con otros dispositivos	X	-	-
StRU 7	Almacenamiento necesario	X	-	-
StRU 8	Procesamiento y visualización de alertas	X	-	-

Fuente: Elaboración propia.

Cada requerimiento planteado es parte importante en el desarrollo del sistema experto, cada uno debe ser tomado en cuenta y manejado de forma correcta para su correcta ejecución.

3.8.3. Requerimientos iniciales de Arquitectura

Para el correcto funcionamiento del sistema experto planteado se describe el hardware y software necesario para un adecuado proceso y puesta a punto del proyecto, en la Tabla 5 que se expone a continuación se muestran las características que posee tanto en hardware como en software a ser utilizado.

Tabla 5 Requerimientos de Hardware y software

Requerimientos Funcionales				
#	Requerimiento	Prioridad		
		Alta	Media	Baja
SrSH 1	Comunicación y optimización del acceso a base datos MYSQL	X	-	-
SrSH 2	Conexión de dashboard	X	-	-
SrSH 3	Velocidad de Procesamiento mínimo 2 GHz	-	X	-
SrSH 4	Capacidad de almacenamiento de datos	X	-	-
SrSH 5	Capacidad de programación de alertas	X	-	-
SrSH 6	Proporcionar seguridad ante las vulnerabilidades	X	-	-
SrSH 7	Soporte a largo plazo de sistema operativo	X	-	-

Fuente: Elaboración propia.

Para la elección del software es necesario tomar en cuenta cada uno de los siguientes puntos ya que estos servirán para una correcta elección de los componentes a ser utilizados en la investigación y su correcto desarrollo.

3.9. Elección del Software y Hardware para el desarrollo del sistema

Una vez terminado el análisis de los requerimientos necesarios, se plantea los diferentes componentes y programas disponibles que cumplirían con los establecido y por ende ayudar en el desarrollo de la investigación planteada. Posteriormente se evalúa cada uno de ellos mediante una comparación hecha a medida de las opciones planteadas para calificar cuál de ellos cumple con todos los requerimientos o la mayor parte de ellos.

3.9.1. Elección del hardware.

Para la implementación del sistema experto es necesario un equipo que por sus características brinde la capacidad de procesamiento requerido para que no existan interrupciones en cuanto a los datos recabados, y estos puedan ser procesados de forma efectiva. En la Tabla 6 se efectúa una valoración en cuanto a las particularidades proporcionadas anteriormente, de tal forma que el hardware mejor valorado será el elegido para el caso de estudio.

Tabla 6 Elección de Hardware para el procesamiento de datos

Hardware	**Requerimientos**						**Valoración Total**
	SyRS 1	SrSH 2	SrSH 3	SrSH 4	SrSH 5	SrSH 6	
Intel Galileo	X	X	-	-	-	-	2
Raspberry Pi	X	X	X	-	-	X	4
Computador	X	X	X	X	X	X	6
	Elección: Computador						
	X Cumple				- No cumple		

Fuente: Elaboración propia.

3.9.2. Elección del software.

Para la elección del software adecuado, que se adapte a nuestras necesidades, se toma en cuenta los criterios que fueron especificados en los requerimientos, para tomar la mejor elección desde el punto de vista funcional y operativo. Como se observa en la Tabla 7 mostrada a continuación.

Tabla 7 Elección sistema operativo

Sistema Operativo	**Requerimientos**						**Valoración Total**
	SyRS 1	SyRS 8	SyRU 2	StSH 1	SrSH 2	SrSH 7	
Ubuntu 16.04	X	X	X	X	X	X	6
Centos 7.0	X	X	-	X	X	X	5
Debian 8.6	X	X	-	X	-	-	3
	Elección: Ubuntu 16.04						
	X Cumple				- No cumple		

Fuente: Elaboración propia.

De acuerdo a la tabla mostrada anteriormente, la elección de software para el sistema operativo es linux en su versión Ubuntu, la cual se encuentra en su versión estable y con soporte, los requerimientos mínimos para su instalación son: procesador de 1GHz. 1.5 GB de RAM. 8 GB de espacio disponible en disco duro, acceso a internet para su actualización.

Tabla 8 Elección de software

Lenguaje de Programación	**Requerimientos**						**Valoración Total**
	SyRS 3	SyRS 5	SyRS 8	StRU 8	SrSH 1	SrSH 6	
PYTHON	X	X	X	X	-	-	4
PHP	X	X	X	X	-	-	4
JAVA	X	X	X	X	X	X	6
	Elección: JAVA						
	X Cumple				- No cumple		

Fuente: Elaboración propia.

En la tabla 8 se muestra la elección de lenguaje de programación el cual será encargado de generar las condiciones y procesamiento de información.

3.10. Normalización de la base de datos

La normalización de base de datos hace referencia a la organización de los datos obtenidos para evitar que existan redundancias, se incluye también la creación de tablas adecuadas a las necesidades para que tengan una mejor interpretación, así como también se pueda generar una relación entre ellas, diseñadas específicamente para el caso de estudio.

La redundancia puede incurrir en el excesivo uso del espacio de almacenamiento, y crear cierto tipo de problemas, tanto de mantenimiento como de interpretación, existen reglas que deben ser cumplidas en la normalización de la base de datos, cada regla es denominada una forma normal, si se cumple con lo dispuesto en la primera regla se dice que la base de datos se encuentra en este momento en la primera forma normal, al encontrarse en la tercera regla o tercera forma normal se considera que la base de datos se encuentra en el nivel más alto necesario (Microsoft, 2013):

3.10.1. Primera forma normal

- Eliminar los grupos de repetidos de las tablas individuales.
- Crear una tabla independiente para cada conjunto de datos relacionados.
- Identifiqué cada conjunto de datos relacionados con una clave principal.

3.10.2. Segunda forma normal

- Crear tablas independientes para conjuntos de valores que se apliquen a varios registros.
- Relacione estas tablas con una clave externa.

3.10.3. Tercera forma normal

- Eliminar los campos que no dependan de la clave.

3.10.4. Estructura de la base de datos

Como se observa en la Tabla 9, se considera los campos involucrados donde no existe redundancia de los datos y el sistema puede generar alertas de acuerdo a sus respectivos valores obtenidos en cada uno de sus sensores.

Tabla 9 Base de datos normalizada

Nodo	**1**	**2**	**3**	**4**	**5**
	Temperatura	Humedad Relativa	Dióxido de carbono	Radiación	Presencia de Humo
Nodo 1	28°C	40%	360 ppm	No	No

Fuente: Elaboración propia.

3.11. Sistemas e ingeniería de software - especificaciones de requerimientos del software basado en el estándar ISO/IEC/IEEE 29148-2011

La norma internacional ISO/IEC/IEEE 29148-2011 es la encargada de realizar la organización de procesos unificados que intervienen en el desarrollo de un sistema de ingeniería de acuerdo a su ciclo de vida y elección de software basado en requerimientos específicos (ISO, 2011):

3.11.1. Diseño del sistema

Se pretende desarrollar el sistema experto que satisfaga las necesidades existentes en el medio a ser aplicado por ende brindar al usuario de forma gráfica, el comportamiento de los factores ambientales existentes en el bosque protector Guayabillas, en tiempo real.

3.11.2. Características del usuario

Para poder ingresar al sistema el usuario necesita ser autenticado por medio de credenciales como lo son usuario y contraseña, las mismas que serán entregadas al encargado. En la Tabla 10 se encuentra las características y limitaciones.

Tabla 10 Características y limitaciones de usuario

Usuario	Contraseña	Descripción	Limitaciones
Admin	SE.UTN.2017	Personal encargado del monitoreo de bosque protector.	El personal podrá verificar los valores de forma visual, y en tiempo real, que han sido registrados. Además, tendrá el acceso a verificar los valores almacenados para un estudio posterior.

Fuente: Elaboración propia.

3.12. Requerimientos específicos

Dentro de los requerimientos específicos se genera las pautas del funcionamiento del sistema, y una descripción de cada uno de sus componentes para su correcta interpretación, así como flujogramas donde se muestra sus pasos a seguir.

3.12.1. Interfaz

El encargado podrá acceder al sistema mediante un navegador el cual tendrá un acceso local si se encuentra dentro de la red o de forma externa mediante la internet, dentro de la plataforma se encuentra con una interfaz amigable, que ayuda con la experiencia de uso.

De acuerdo con la teoría del color la elección del mismo en un diseño web, es una de las partes principales ya que es donde el usuario permanecerá la mayor parte del tiempo, por tal motivo se busca que sea simple y no resalten colores fuertes que permitan la desconcentración dependiendo del caso donde se encuentre expresado (Musumeci, 2016).

La utilización del color blanco como fondo de la pantalla denota luz, tecnología, pureza y verdad, también considerado como el color de la perfección ya que su utilización brinda al usuario la sensación de encontrarse en un entorno amigable y sobrio, permitiendo que la concentración sea positiva y no resulte cansado para la vista de quien se encuentra operando el sistema, buscando también la perfecta fusión entre los colores que formen parte de la misma paleta para las funciones que acompañan al sistema dentro de su página, siendo estos los colores azules claros, los mismos que acompañados del blanco fijan la mirada en la parte principal del sistema (González, 2013).

3.13. Funciones

A continuación, se despliega de forma independiente cada uno de los procesos realizados en cada función, indicando de manera clara el proceso que cumple cada uno de ellos para llegar a su adecuado despliegue.

3.13.1. RF-01 Inicio de sesión

La función inicio de sesión es la encargada de permitir la conexión entre el usuario y la plataforma web, realizando la autenticación del usuario por una clave previamente compartida, la misma se encuentra almacenada en el sistema, en la Tabla 11 se encuentra descrito cada una de las funciones y su descripción, que serán llevadas a cabo e implementadas en el sistema.

Tabla 11 Tabla caso de uso iniciar sesión

Función	**Descripción**
Caso de usos asociados	No aplica
Actores	Usuario
Descripción	Permite iniciar sesión
Flujo principal	Ingresar el usuario Ingresar la contraseña Validación de datos. Finaliza la acción.
Flujo de alternativo	El sistema genera un mensaje de autenticación erróneo, si este fuese el caso.
Flujo de excepciones	El usuario no recibe la autenticación.
Precondiciones	El usuario y contraseña deben encontrarse registrados en el sistema para obtener el acceso e iniciar sesión.

Postcondiciones	El usuario obtiene el acceso correcto.
Puntos de extensión	El usuario no logra obtener el acceso, pese a tener el usuario y contraseña correctos.

Fuente: Elaboración propia.

La Figura 5 muestra el proceso realizado por el sistema cuando el usuario ingresa sus credenciales como lo son el nombre de usuario y contraseña, siendo estas correctas válida la información y permite la autenticación en el sistema, paso previo antes de realizar la visualización de información.

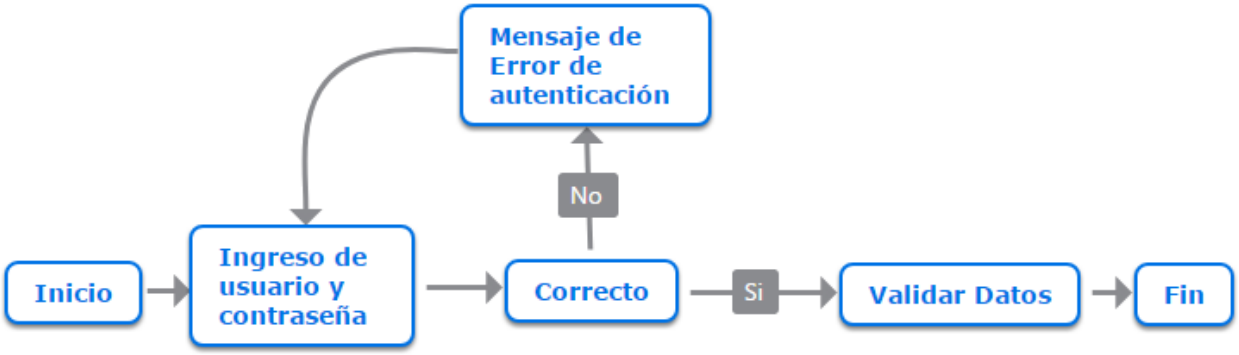


Figura 5 Flujograma inicio de sesión
Fuente: Elaboración propia.

3.13.2. RF-02 Visualización de datos

La función visualización de datos se encarga de mostrar al usuario previamente autenticado en el sistema, los valores que se encuentran recolectados y almacenados, de acuerdo a cada uno de los nodos presentes en el sistema de alerta temprana, en la Tabla 12 se muestra las funciones y su descripción, para una mejor apreciación del proceso realizado.

Tabla 12 Tabla caso de uso visualización de datos

Función	**Descripción**
Caso de usos asociados	Iniciar sesión.
Actores	Usuario.
Descripción	Permite la visualización de los datos recolectados, que han sido registrados.
Flujo principal	El usuario visualiza el despliegue del sistema de monitoreo de forma gráfica donde serán observados los valores almacenados del nodo que el usuario decida monitorizar con la posibilidad de migrar a cualquier nodo de forma indistinta. Finaliza el proceso de uso.
Flujo de alternativo	El sistema mantiene continúo el flujo de datos provenientes de la WSN, para poder observarlos en tiempo real.
Flujo de excepciones	Falla del sistema regresa a su paso inicial y procede a iniciar sesión nuevamente.
Precondiciones	Previamente debe iniciarse sesión en el sistema.
Postcondiciones	No aplica.
Puntos de extensión	No aplica.

Fuente: Elaboración propia.

La Figura 6, muestra el flujograma del proceso realizado posterior a la autenticación del usuario, se muestra los gráficos de monitoreo de acuerdo a cada uno de los nodos, permitiendo seleccionar el nodo que se desea monitorizar.

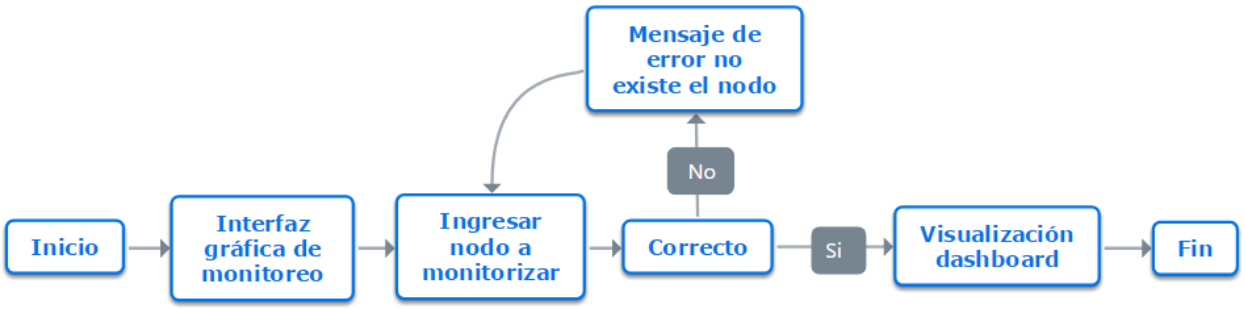


Figura 6 Flujograma visualización de datos.
Fuente: Elaboración propia.

3.13.3. RF-03 Adquisición del historial de datos

La función adquisición del historial de datos, es la encargada de permitir que el usuario pueda visualizar los datos que se encuentran almacenados en el sistema experto, los datos del historial tanto de datos y alertas, se van a mostrar en tablas distintas para su correcta visualización. En la Tabla 13 se encuentran las funciones y descripción que se deben cumplir en este punto.

Tabla 13 Tabla caso de uso adquisición del historial de datos

Función	**Descripción**
Caso de usos asociados	Iniciar sesión.
Actores	Usuario.
Descripción	Permite la visualización del historial de datos almacenados en el sistema.
Flujo principal	El usuario selecciona la pestaña historial. El sistema se encarga del despliegue de los gráficos históricos generados por cada uno de los nodos de acuerdo a las fechas establecidas. Se despliega el historial de forma gráfica. Finaliza el caso de uso.
Flujo de alternativo	El sistema mantiene continúo el flujo de datos provenientes de la WSN, para poder ser almacenados en la base de datos.

Flujo de excepciones	Falla del sistema regresa a su paso inicial y procede a iniciar sesión nuevamente.
Precondiciones	Previamente debe iniciarse sesión en el sistema.
Postcondiciones	No aplica.
Puntos de extensión	No aplica.

Fuente: Elaboración propia.

En la Figura 7, se aprecia el flujograma en el que se puede identificar cada paso del proceso llevado a cabo por el sistema para la correcta visualización del historial. Para que el usuario pueda acceder a estos datos debe encontrarse autenticado previamente.

Figura 7 Flujograma adquisición del historial de datos
Fuente: Elaboración propia.

3.13.4. RF-04 Generación de alertas.

La función generación de alertas, es una de las principales funciones del sistema ya que será la encargada de realizar el proceso de aviso de las anomalías encontradas en el entorno aplicado, en la Tabla 14 se muestra las funciones y su descripción, este sistema se encarga de la verificación de los datos ingresados para no incurrir en notificación de alertas erróneas.

Tabla 14 Tabla caso de uso generación de alertas

Función	**Descripción**
Caso de usos asociados	Iniciar sesión.
Actores	Usuario.
Descripción	El sistema antes de notificar con una alerta realiza la verificación de valores ingresados comparando 3 datos almacenados en la base de datos anteriores, si los datos ameritan generar la alerta esa se procede de acuerdo a los límites preestablecidos.
Flujo principal	El sistema experto se encuentra receptando datos de forma continua. El dato que se encuentre dentro de cada límite genera una alerta en forma de mensaje. Se muestra en la pantalla el nodo y el tipo de alerta generada. Finaliza el caso de uso.
Flujo de alternativo	El sistema mantiene continúo el flujo de datos provenientes de la WSN, para poder ser almacenados en la base de datos.
Flujo de excepciones	Falla del sistema regresa a su paso inicial y procede a iniciar sesión nuevamente.
Precondiciones	Previamente debe iniciarse sesión en el sistema.
Postcondiciones	No aplica.
Puntos de extensión	No aplica.

Fuente: Elaboración propia.

En la Figura 8 se observa el flujograma del proceso realizado por el sistema para la generación de las alertas, estas alertas serán identificadas por rangos de tolerancia según sea el caso de estudio y el entorno aplicado.

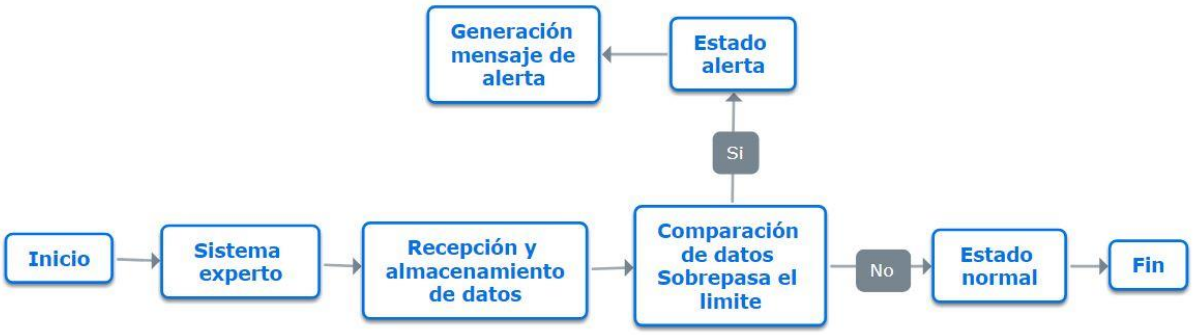


Figura 8 Flujograma generación de alertas
Fuente: Elaboración propia.

3.14. Requerimientos de usabilidad

Garantiza al usuario que el sistema experto será capaz de brindar requerimientos como lo son fiabilidad, precisión, seguridad y facilidad de acceso a los datos recabados por medio del uso de la plataforma de monitoreo, de la misma forma garantizar el acceso en tiempo real a los valores almacenados en la base de datos de cada uno de los nodos existentes.

3.14.1. Requerimientos de rendimiento

El rendimiento y la disponibilidad del sistema serán permanentes en el tiempo, con la facilidad de brindar escalabilidad en cuanto a nodos implementados según las necesidades existentes en el bosque protector Guayabillas.

3.14.2. Requerimientos de requisitos de base de datos lógica

Optimizar el diseño implementado en la base de datos, con normalización para el correcto manejo de los datos almacenados, y garantizar que ante varias consultas simultáneas no se vea afectada en cuanto a su rendimiento.

3.14.3. Restricciones de diseño

Para su verificación de las diferentes alertas el sistema debe encontrarse encendido y la página de monitoreo abierta. El correcto funcionamiento del sistema depende en su totalidad de los datos ingresados por la WSN implementada en el bosque protector Guayabillas, y su enlace inalámbrico desplegado para su recepción en la Universidad Técnica del Norte.

Los datos que ingresan al sistema experto provenientes de la WSN deben llegar en conjunto de acuerdo a su nodo, ya que la base de datos registra el nombre del nodo, la fecha y la hora además de cada una de las variables como lo son temperatura, humedad relativa, dióxido de carbono, humo y radiación. Caso contrario el sistema no podrá reconocer los valores en el mismo tiempo y generar problemas en cuanto a la generación de las diferentes alertas y diagramas históricos.

3.14.4. Restricción notificación.

Dentro de las restricciones de diseño, se encuentra que el despliegue de los organismos de socorro, como lo son bomberos, policía nacional, cruz roja, no pueden ser notificados por ningún medio escrito, ya sea este mensaje de texto, correo electrónico, o llamada pre grabada.

Por tal motivo para este tipo de desarrollos externos con gran impacto existen dos formas de realizar la notificación, siendo totalmente válidas y los organismos de socorro podrían desplegarse inmediatamente, a continuación, se especifica cada una de las formas de notificación.

- El sistema integrado de seguridad ECU-911 realiza la instalación de botones de pánico los mismos que deben ubicarse en sitios estratégicos, para el presente desarrollo el botón se encontraría dentro del centro de monitoreo y al alcance del

personal encargado, este sistema funciona mediante la tecnología GPRS, la cual permite sectorizar la alerta e identificar de manera inmediata el lugar donde se suscita la emergencia, por tal motivo los operadores del ECU-911 procederán a realizar una llamada para validar la información.

- Para evitar incurrir en errores, se realiza por medio de una llamada al sistema integrado de seguridad ECU-911 con su línea única 911 donde se debe proporcionar cierta información personal para la verificación de identidad de la persona que se encuentra realizando la llamada, y posteriormente dar los datos para el despliegue de los organismos.

Según lo manifestado por el Tcnl. Carlos Cadena, Coordinador técnico zonal del Sistema Integrado de Seguridad (SIS ECU 911), Cuya oficina se encuentra ubicada en la ciudad de Ibarra, en las instalaciones del ECU 911. El SIS cuenta con la implementación de la norma de calidad EENA, la misma es una norma europea, que índica lineamientos a seguir por cada zona, y precautelar la seguridad de la información que se encuentra circulando por su red, así como también, el centro integrado de seguridad no puede realizar modificaciones en cuanto a su protocolo e integración de una plataforma externa a sus sistemas (Cadena, 2017):

En caso de que se requiera la implementación de esta plataforma de carácter urgente, deberá pasar diferentes tipos de pruebas y asegurar la legitimidad de los datos, evaluada en la ciudad de Quito en las instalaciones ubicadas en la calle Julio Endara s/n sector parque Itchimbía, por los encargados del departamento de proyectos e innovación, a cargo del ingeniero Wilmer López quienes serán los encargados de dar el permiso para la programación en su centro de datos y generación de protocolo frente a una emergencia de este tipo.

Según lo manifestado por el ingeniero Wilmer López, las pruebas que deben realizarse a la plataforma, sería la evaluación del código, y verificar si la programación puede ser adaptada a su sistema, tiempos de respuesta, notificación de alertas, y el radio enlace debería llegar directamente hacia las instalaciones del ECU-911 ubicado en la ciudad de Ibarra parque ciudad blanca (López, 2017).

3.14.5. Atributos del sistema de software

El sistema implementa una interfaz amigable con el usuario y de fácil uso, limitando la complejidad y suministrando al usuario su correcto despliegue de información necesaria para el análisis del entorno en el medio implementado. Así como también brinda la seguridad necesaria ya que cuenta con sistema de autenticación por medio de usuario y contraseña. para su funcionamiento.

3.14.6. Información de apoyo

No aplica.

3.15. Verificación

La verificación del sistema se lleva a cabo mediante la realización de pruebas de funcionamiento y diseño, donde se incluirá:

- Análisis del interfaz y diseño.
- Validación de acceso.
- Verificación de consultas simultáneas
- Validación de datos

3.16. Apéndices

3.16.1. Suposiciones y dependencias

Se supone que el servidor donde se encuentra alojado el sistema experto tiene una disponibilidad operativa de 7 días a la semana y 24 horas al día.

Se supone que el enlace inalámbrico se encuentra operativo y sin restricciones para su correcto transporte de datos sin incurrir en saturación del canal de trasmisión.

Depende de los organismos encargados el correcto despliegue del personal ante cualquier tipo de emergencia suscitada.

3.16.2. Acrónimos y abreviaturas

- WSN: Wireless Sensor Network (Red Inalámbrica de Sensores).
- UML: Lenguaje Unificado de Modelado.
- SIS: Sistema Integrado de Seguridad
- Tcnl: Teniente Coronel.

3.17. Sistemas e ingeniería de software - descripción de la arquitectura del software basado en el estándar ISO/IEC/IEEE 42010.2011.

3.17.1. Introducción

Encargado de la definición de requisitos para la designación de las arquitecturas de un sistema, del software. Su objetivo es estandarizar la práctica de la descripción de la arquitectura,

mediante la definición de términos estándar, presentando una base conceptual para la expresión, la comunicación y la revisión de la misma.

3.17.2. Propósito

La arquitectura del software tiene como su principal propósito permitir al usuario la visualización de los datos recabado en tiempo real de la WSN ubicada en el bosque protector Guayabillas, mediante una interfaz web adaptada al medio que cumple con las condiciones establecidas y especificaciones de arquitectura propuestas por el estándar.

3.17.3. Alcance

La arquitectura del software precisa una interfaz web intuitiva, que permite la recolección de datos almacenamiento, procesamiento, y generación de alarmas, se encuentra alojado en un servidor y su visualización será de forma local.

3.17.4. Usuarios interesados

Los usuarios interesados en el desarrollo del sistema experto aplicado a un entorno especificó, son los encargados de llevar el registro y el manejo de la información recabada para un análisis posterior y su correcto manejo de los organismos de socorro.

3.18. Framework conceptual

3.18.1. Descripción de la arquitectura en concepto

El software implementado para el sistema experto consta de un servidor web apache y una base de datos MySQL alojados en un servidor de software libre, que permite al usuario acceder de forma segura mediante la autenticación de usuario y contraseña, para visualizar en tiempo real las diferentes variables implementadas, el acceso se realiza mediante un navegador web.

Los datos recabados por cada uno de los nodos en la WSN son receptados en un nodo central ubicado en el bosque protector Guayabillas, de donde mediante un enlace inalámbrico serán enviados a la Universidad Técnica del Norte, donde serán almacenados en una base de datos normalizada en el servidor, los mismos son procesados e interpretados mediante condiciones establecidas, que garantizan el correcto manejo de alertas contribuyendo con el sistema de alerta temprana, en la Figura 9 se muestra la petición de acceso y respuesta desde el usuario hacía el servidor.

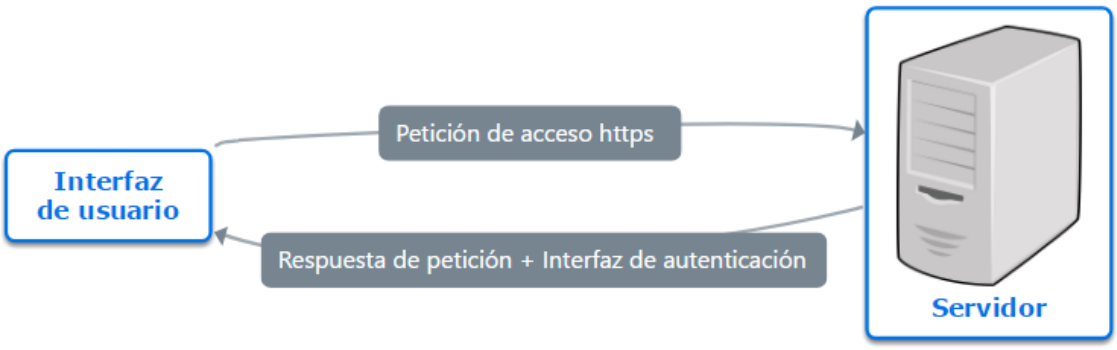


Figura 9 Petición de acceso al servidor
Fuente: Elaboración propia.

3.18.2. Stakeholders y sus roles

Se refiere a los encargados del sistema quienes se encuentran principalmente trabajando con la visualización de las alertas, para su interpretación y toma de decisiones ante cambios bruscos de las variables implementadas.

El encargado debe realizar las siguientes acciones detalladas a continuación para el despliegue de los organismos de socorro.

- Reconocer el nodo exacto de la alerta
- Llamar al ECU-911
- Brindar el soporte básico

Estos pasos serán realizados por el encargado del sistema siempre que se presente una emergencia comprobada ya que de lo contrario se podría incurrir en sanciones legales. Desde el centro integrado de seguridad se encargan de gestionar a los organismos de socorro, y la verificación de su pronta respuesta ante la emergencia, sea esta preventiva o accionaria.

3.18.3. Actividades de arquitectura en el ciclo de vida

Las actividades encargadas en el ciclo de vida es recolectar la información proporcionado por la WSN y almacenarla para su correcto despliegue en forma de valores históricos los cuales serán de gran importancia para futuros informes forestales y toma de decisiones.

3.19. Descripciones prácticas de arquitectura

3.19.1. Documentación de la arquitectura

"La documentación debe ser clara y organizada, por tal motivo la información proporcionada cumple con el modelo de arquitectura" (Kruchten, 1995), este tipo de sistema es implementado para obtener una visualización del sistema desde diferentes puntos de vista.

- Vista lógica
- Vista de procesos
- Vista de desarrollo
- Vista física
- Vista de escenario

3.19.2. Identificación de los Stakeholders y sus responsabilidades

El involucrado en el caso de estudio es el administrador del sistema quien de forma efectiva puede interpretar la información proporcionada por cada uno de los nodos implementados, las responsabilidades se basan en las 5 vistas de arquitectura

- Vista lógica: Modelo de objetos, clases, entidad – relación.
- Vista de procesos: Modelo de concurrencia y sincronización.
- Vista de desarrollo: Organización estática del software en su entorno de desarrollo.
- Vista física: Modelo de correspondencia software – hardware.
- Vista de escenario: Caso de uso.

3.20. Selección de los puntos de vista de la arquitectura.

Desde el punto de vista de arquitectura de software, se muestra a continuación los diferentes componentes encargados de realizar la comunicación y procesamiento de datos dentro del sistema experto. En la Tabla 15, se aprecia cada una de las vistas y su lenguaje unificado de modelado.

Tabla 15 Tabla vista de arquitectura

Vistas	**UML**
Lógica	Entidad – Relación
Procesos	Secuencia
Desarrollo	Componentes
Física	Despliegue
Escenario	Caso de uso

Fuente: Elaboración propia.

3.20.1. Vistas de arquitectura.

Las vistas de la arquitectura lógica, representar una vista parcial de las propiedades específicas que se encuentran en el sistema, contiene de manera simplificada una muestra de las características implementadas y sus respectivas conexiones de acuerdo al tipo de información que se encuentra almacenada.

3.20.1.1. Vista lógica – clases.

Como puede apreciarse en la Figura 10, el diagrama lógico de clases involucradas en el desarrollo del sistema experto, cada una de las clases cuenta con las variables que van a involucrarse y la comunicación bidireccional entre las mismas.

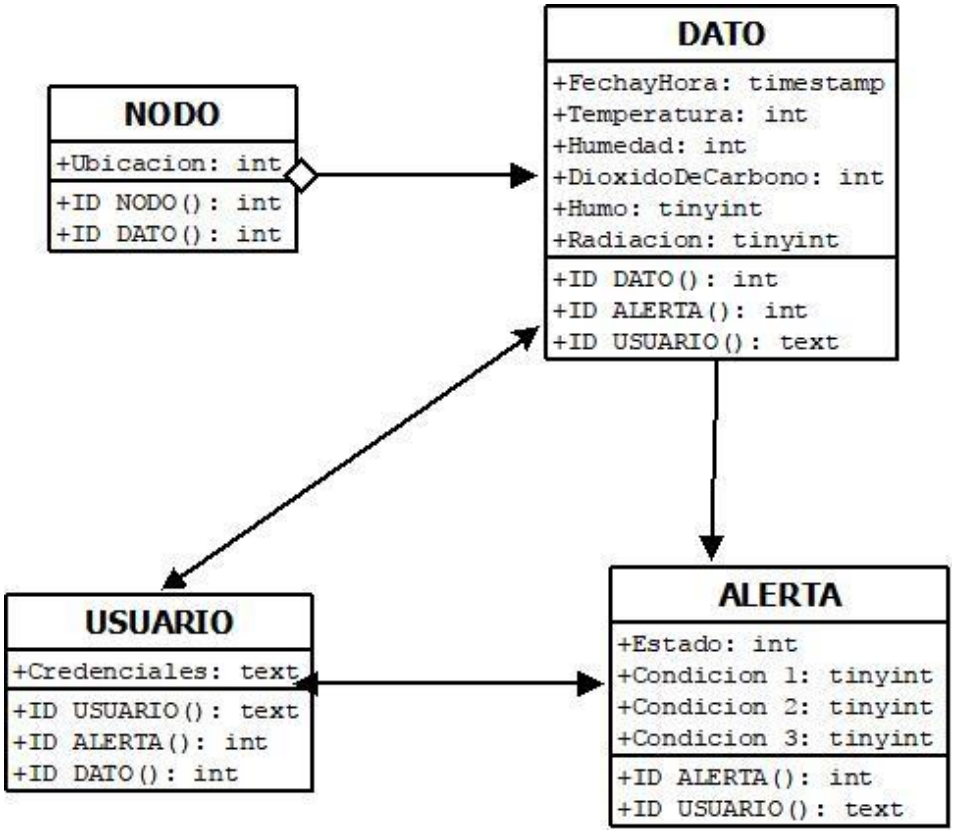


Figura 10 Diagrama de clases
Fuente: Elaboración propia.

3.20.2. Vista procesos

Para el caso de la vista de procesos, se encuentra diferentes funciones realizadas por cada uno de los componentes involucrados en el sistema, así como su comunicación y cada una de las funciones que van a ser realizadas de forma independiente según sus características.

3.20.2.1. Diagrama secuencia – Monitoreo local

Como se aprecia en la Figura 11, el diagrama de secuencia para el monitoreo se lleva a cabo desde la petición del usuario hacia la base de datos, solicitando se le muestre los datos almacenados provenientes de la WSN y su nodo central.

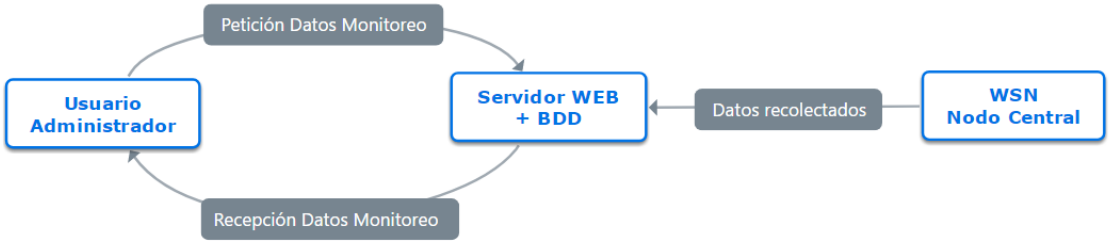


Figura 11 Diagrama Monitoreo
Fuente: Elaboración propia.

3.20.2.2. Vista desarrollo – Componentes

La vista de desarrollo y sus componentes muestra la comunicación existente para su correcto despliegue de cada uno de los datos, los componentes involucrados se encuentran conectados y funcionando de forma activa en todo el proceso del correcto desarrollo del sistema, como se puede apreciar en la Figura 12.

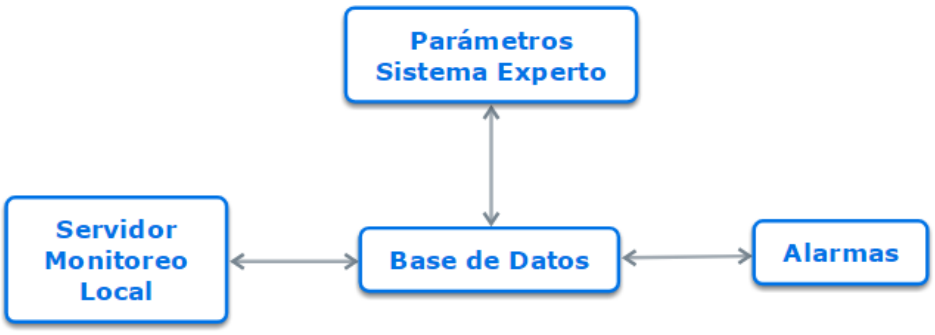


Figura 12 Diagrama componentes
Fuente: Elaboración propia.

3.20.2.3. Vista física – Despliegue

Como puede ser apreciado en la Figura 13, se muestra la vista física del sistema donde cada uno de los componentes interactúa de forma única y cumple sus funciones para el despliegue de la información solicitada por el usuario.

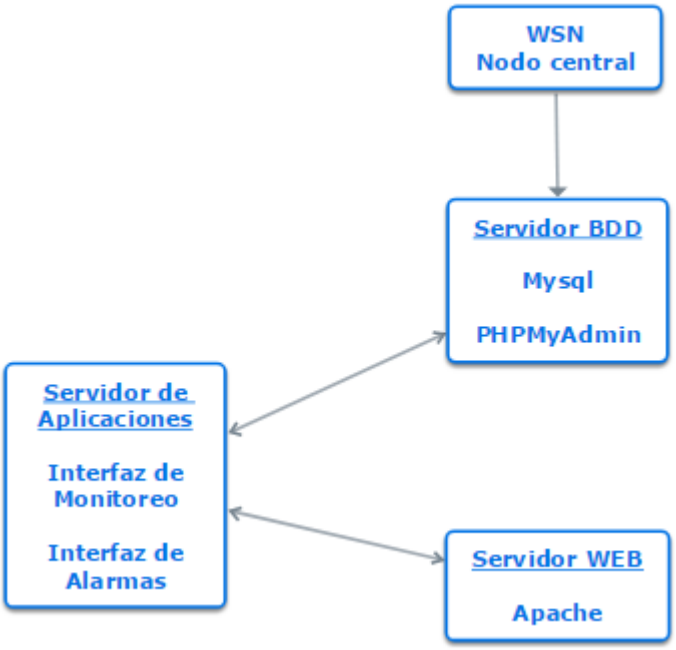


Figura 13 Diagrama despliegue
Fuente: Elaboración propia.

3.20.3. Vista escenario – Casos de uso

Diagrama caso de uso, especifica cada uno de los procesos de petición realizada por el usuario de forma lógica ante el sistema, para recibir la información solicitada, de forma clara y cumpliendo los procesos específicos.

3.20.3.1. Diagrama caso de uso – Acceso al sistema.

Como se aprecia en la Figura 14, para el caso de uso acceso al sistema el usuario debe autenticarse por tal motivo, sigue un proceso lógico que permitirá el correcto ingreso si es que cumple con las credenciales adecuadas.

Figura 14 Diagrama acceso al sistema
Fuente: Elaboración propia.

3.20.3.2. Diagrama caso de uso – Sistema de monitoreo.

Para el diagrama de caso de uso sistema de monitoreo, el usuario realiza la petición al servidor web, el cual de forma automática sigue los procesos y consulta en los componentes con los que se encuentra conectado, para obtener la información almacenada, y poder responder con los gráficos y la información proveniente de la WSN, como se puede observar en la Figura 15.

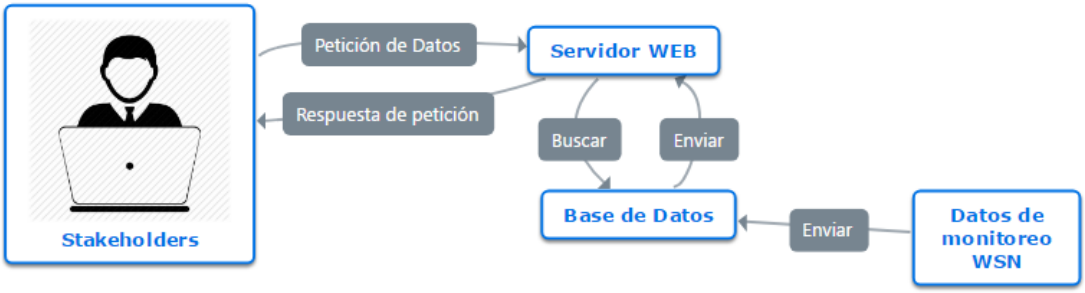


Figura 15 Diagrama sistema de monitoreo
Fuente: Elaboración propia.

3.20.4. Consistencia en la cantidad de vistas de la arquitectura

Cada componente integrado en el sistema genera una vista en la arquitectura implementada, cada una de estas será representada con la descripción de cada uno de los mismos dando así una consistencia en información que va a ser entregada al usuario.

3.20.4.1. Descripción de componentes

En la Tabla 16, puede ser apreciada de forma descriptiva cada uno de los componentes que se encuentran siendo aplicados en el desarrollo del sistema, si existen componentes a los que se encuentra ligado o relacionado.

Tabla 16 Tabla de componentes

Nombre Del Componente	Descripción	Componentes Relacionados
Servidor de monitoreo	Componente encargado de permitir la interacción entre el usuario y el sistema experto	Servidor de monitoreo Base de datos
Alarmas	Componente encargado de generar alertas que permitirán al usuario una lectura de los datos fuera de los límites preestablecidos.	Alarmas Base de datos
Base de datos	Componente encargado del almacenamiento de datos provenientes de la WSN.	Servidor de monitoreo Alarmas Parámetros de monitoreo Base de datos
Parámetros de monitoreo	Componente encargado de la gestión y procesamiento de datos.	Parámetros de monitoreo Base de datos

Fuente: Elaboración propia.

3.21. Arquitectura lógica

Representa la forma en la que los componentes lógicos, van a interactuar según las capacidades y sus necesidades del sistema experto, formando un proceso organizado que brinde una solución específica adaptada al medio en el que se encuentra involucrado.

3.21.1. Performance

La presente arquitectura se basa en los requerimientos establecidos para proporcionar un registro de datos provenientes de la WSN ubicada en el bosque protector Guayabillas, así como también la consulta y notificación de los mismos.

- El sistema experto podrá soportar usuarios simultáneos
- El sistema se encarga de la presentación de datos en un tiempo de 30 segundos, con lo que se busca un monitoreo en tiempo real que sirva para la generación de alertas, y contribuir con sistema de alerta temprana.
- La plataforma brinda seguridad en cada una de sus etapas protegida por medio de usuario y contraseña.

3.21.2. Calidad de los datos

La arquitectura de software presente, basa su calidad en una interfaz amigable con el usuario y su presentación en colores claros como el blanco que hacen que el encargado del monitoreo tenga mayor facilidad de ubicación y concentración en cuanto a la visualización de las alertas.

3.22. Detalles de la implementación

En los detalles de la implementación, se encuentran especificados cada uno de los puntos que salen a relucir en el desarrollo de la plataforma, así como el lenguaje seleccionado por el desarrollador de acuerdo a los requerimientos y entorno en el cual va a ser aplicado el sistema experto así como su rango de tolerancia , y el despliegue de los respectivos informes forestales, que serán de gran ayuda a las personas especializadas en temas afines a la ingeniería forestal para su correcta interpretación.

3.22.1. Lenguajes y plataformas

La lógica de diseño aplicada en el sistema experto se encuentra basada en un lenguaje de programación orientado a objetos, sin encontrarse inmerso en condiciones que apliquen restricciones, se diseñó un sistema adaptable al caso de estudio, sistema de alerta temprana contra incendios forestales, que brinde la capacidad de crecimiento de acuerdo a las necesidades del entorno fuera del alcance de este proyecto.

El entorno de desarrollo integrado y libre para este proyecto fue en base a la utilización de un Framework que simplifica la implementación de la aplicación, de acuerdo a los requerimientos planteados anteriormente en la elección del software, se optó por NetBeans y su lenguaje de programación Java el cual, mediante la integración de módulos, brinda al usuario diferentes características que pueden actualizarse dentro de la misma interfaz de la aplicación.

Así como también la utilización de jsf que permite el diseño de sistemas web, que brinden seguridad, eficiencia, y fluido despliegue de funciones. Ya que su codificación resulta en gran medida robusta en el back end. Y genera que tola la lógica del negocio sea implementada en Java.

3.22.2. Informe de situación

Según lo establecido por la secretaría de gestión de riesgos de Ecuador los informes son realizados de acuerdo a cada incidente forestal de manera diaria, y a partir de esto se presenta un informe anual donde se expresa de manera escrita los percances presentados a lo largo del año en las diferentes regiones del país.

En estos informes se especifica la provincia, cantón, número de hectáreas comprometidas, número de organismos desplegados, así como también en número de personal tanto en mujeres como hombres asignados, unidades móviles, tipo emergencia, presencia de equipo aéreo, y herramientas disponibles.

Las acciones de respuesta frente a estos eventos de catástrofe son articuladas por las organizaciones quienes cuentan con protocolos en casos de emergencia, estas instituciones son: las coordinaciones zonales (CZ), la secretaría de gestión de riesgos (SGR), y el sistema nacional descentralizado de gestión de riesgo (SNDGR), así como también el principal organismo como lo es el sistema integrado de seguridad (SIS) ECU-911.

3.22.3. Rango de tolerancia.

En la Tabla 17 se observa los valores que fueron establecidos en el sistema experto de tal modo se garantiza que pueda ser implementado en un entorno específico de alerta temprana para incendios forestales.

Tabla 17 Rangos de Tolerancia

Variable ambiental	Valores normales	Valores alerta	Valores emergencia
Temperatura	0 - 30	31 - 50	51 – adelante
Humedad relativa	30 - adelante	21 - 30	0 – 20
Dióxido de carbono	0 - 999	1000 – 1500	1501 – adelante
Humo	NO	-	SI
Radiación	NO	-	SI

- No existe valor

Fuente: (Ortiz, 2017)

3.22.4. Condiciones extras.

Las condiciones ayudan al encargado del sistema en la manipulación de las variables obtenidas por la WSN dichas condiciones deben cumplir características para que su funcionalidad entre en acción. En la Tabla 18 se observa la alerta emitida por cada condición

Tabla 18 Rangos Condicionales

Condiciones	Valores normales	Valores alerta	Valores emergencia
Condición 1	NO	-	SI
Condición 2	NO	-	SI
Condición 3	NO	-	SI

- No existe valor

Fuente: Elaboración propia.

Los valores establecidos para que cada una de las condiciones genere su estado de emergencia se reflejan en la Tabla 19, proporcionando un mayor estado de alerta cuando las condiciones se vuelven extremas.

Tabla 19 Rango de activación de condiciones

Alerta condicionada	**Temperatura > 50°**	**Humedad < 20%**	**Dióxido De carbono >1501 ppm**	**Humo = Si**	**Radiación = Si**
Condición 1	X	X	X	X	X
Condición 2	X	X	X	-	-
Condición 3	X	-	-	X	-

X Cumple - No cumple

Fuente: Elaboración propia.

3.22.5. Elección del Hardware

Para realizar la elección del hardware se revisa, las características que deben ser cubiertas, para solventar el software a ser implementado en el desarrollo del sistema experto, como equipo según lo establecido en los requerimientos iniciales Tabla 6, se opta por la utilización de un computador que debe cumplir con lo siguiente:

- CPU AMD o Intel con procesamiento 2 GHz
- Memoria ram 4 GB, 64 bits.
- Almacenamiento 500 GB
- Tarjeta de sonido y parlantes
- Monitor 15 pulgadas
- Tarjeta de red.

A partir de lo anteriormente nombrado el sistema brinda la adaptación a sistemas con mejores características y prestaciones, que en su gran mayoría no incurrirán en fallas ya que el sistema no requiere mayor procesamiento, tomando en cuenta las necesidades del software antes indicado

CAPÍTULO 4. PRUEBAS Y FACTIBILIDAD TECNOLÓGICA

El presente capítulo tiene como finalidad presentar los resultados que fueron evidenciados en el desarrollo del sistema experto, así como verificar que el alcance y los objetivos planteados en un principio se cumplen correctamente, además se realiza un análisis de factibilidad tecnológica en la implementación del proyecto.

4.1. Pruebas y resultados

Las pruebas a realizarse para el sistema experto, serán ejecutadas de manera controlada en forma de ensayo de laboratorio, debido a que de esta manera se establece un escenario optimo con la finalidad de conocer la forma en la que reacciona el sistema frente a cambio en cada una de las variables.

Se debe aclarar que cada una de las variables a ser ingresadas en el sistema, será de forma manual, directamente a la base de datos para tener la capacidad de realizar cambios bruscos y poder demostrar el correcto funcionamiento de todo el sistema frente a cada rango de tolerancia.

4.2. Almacenamiento de datos

Para el almacenamiento de datos se utiliza una base MySQL a la que se ingresa mediante el administrador phpmyadmin para su respectivo manejo, configuración y asignación de usuarios, esta base de datos tiene la capacidad de permitir el ingresar de valores de forma manual para su

demostración de laboratorio como puede ser apreciado en la Figura 16, el ingreso se lo realiza por medio del navegador web donde se digita la siguiente dirección: http://127.0.0.1/phpmyadmin/

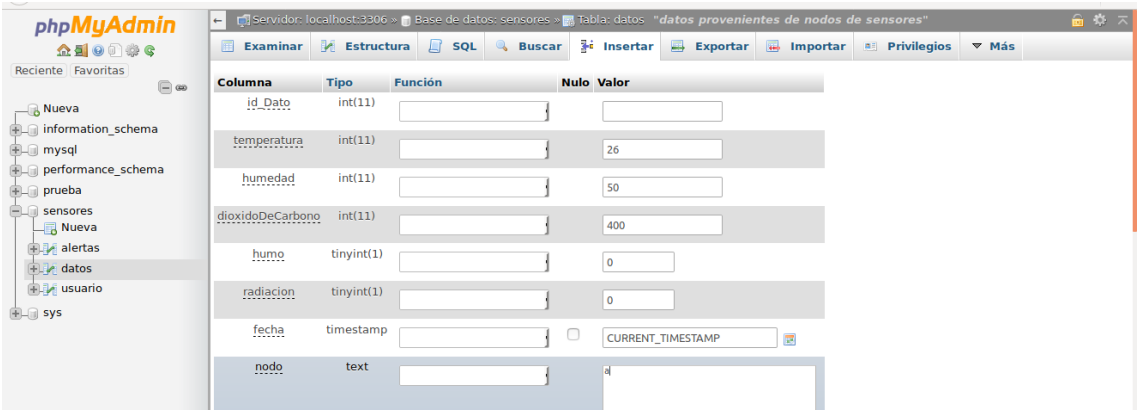


Figura 16 Vista principal base de datos
Fuente: Elaboración propia.

.

La base de datos permanecerá siempre encendida siempre y cuando el equipo donde se encuentra almacenada permanezca activo, garantizando así que los datos provenientes del nodo ubicado en el bosque protector Guayabillas están siendo almacenados de manera de forma continua.

En el análisis realizado por (Ortiz, 2017): *"Benchmark de selección de sensores para una WSN de recolección de datos para un sistema de alerta temprana de incendios forestales"*, los tiempos de respuesta de cada uno de los sensores varía en un rango que comprende los 10 - 30 segundos, donde cada sensor toma sus lecturas de forma independiente.

Para el caso de recepción de esta información se toma el valor más alto en el tiempo que sería de 30 segundos, para que así el nodo tenga la posibilidad de realizar ciertas medidas antes de su envío al sistema y se proceda a su almacenamiento.

Para cubrir el caso en donde exista presencia de viento, los datos pueden tardar hasta 57 segundos para conseguir ser detectados de manera correcta y mostrar valores de alerta, por tal motivo

se realiza una comprobación de datos ingresados antes de la notificación de alerta, para no incurrir en alarmas erróneas.

La recepción de los datos se realiza de forma secuencial por un puerto definido previamente, lo que permite que en el sistema no exista saturación en cuanto a datos ingresados por la WSN, también cuenta con la propiedad de almacenar los datos de forma que sea identificado el nodo al que pertenecen, facilitando así el registro y la correcta ubicación en la tabla cada uno de los valores

Dado el caso hipotético que llegase información de 2 nodos o más al mismo tiempo a la base de datos, esta información será almacenada en el búfer de la memoria ram y posteriormente integrada al azar en el campo correspondiente (ROSENBERG, 2016).

4.3. Capacidad de almacenamiento y saturación.

Dentro de las capacidades del sistema es indispensable conocer cuál será el almacenamiento necesario, para poder dimensionar y brindar una solución adaptable a cambios, sin incurrir en costos adicionales a futuro.

4.3.1. Almacenamiento

Tabla 20 Capacidad de almacenamiento

Columna	**Tamaño en bytes**	**Número de campos**	**Bytes de sobrecarga**	**Tamaño en bytes**
Cabecera para fila	29	1	1	29
Byte por campo	2	6	1	12

Nvarchar	25	6	1	150
Sobrecarga	-	-	6	-
TOTAL			9	191

Fuente: (Microsoft, 2017)

El tamaño total de cada fila en la tabla se realiza con la suma de los bytes totales y los bytes de sobrecarga (191 + 9) de acuerdo con los datos de la Tabla 20, dando un valor por cada fila de 200 bytes, si los datos van a ser almacenados cada 30 segundos, en 1 año se tiene un número de filas 1051200 como se observa en la Tabla 21 (Microsoft, 2017).

Tabla 21 Aproximado de filas almacenadas y su tamaño en bytes

Tiempo	**Número de filas**	**Tamaño en bytes**
30 seg.	1	200
1 min	2	400
1 hora	120	24000
1 día	2880	576000
1 mes	86400	17280000
1 año	1051200	210240000
2 años	2102400	420480000
5 años	5256000	1051200000

Fuente: Elaboración propia.

En este punto se realiza la multiplicación del tamaño de cada fila por el número de filas en 1 año (200 * 1051200) dando como resultado 210240000 bytes. La conversión a megabytes (MB),

tomando en cuenta los datos mostrados en la Tabla 22. dando un valor de base de datos 210.24 MB aproximadamente, por cada tabla.

Tabla 22 Representación de las unidades de medida en bytes

Unidad	**Tamaño**	**Tamaño en bytes**
1 byte	8 bits	-
1 kilobyte (Kb)	1024 bytes	-
1 megabyte (Mb)	1024 kb	1 048 576 bytes.
1 gigabyte (Gb)	1024 Mb	1 073 741 824 bytes.

Fuente: (Fuentes, 2004)

Esta información es utilizada para realizar la estimación del tamaño de la base de datos de acuerdo a cada uno de los datos ingresados, esta estimación sirve para poder planificar el almacenamiento asignado de acuerdo a los requerimientos y sus respectivas funciones.

4.3.2. Saturación.

Como se encuentra planteado los datos vendrán desde la WSN ubicada en el bosque protector Guayabillas, la llegada de estos datos puede incurrir en saturación del sistema cuando varios de estos valores lleguen en el mismo instante de tiempo, parte de la forma de evitar este problema viene desde el establecimiento de un protocolo que actúa para enviar cada cierto instante de tiempo los datos recabado por cada nodo, pero desde el punto de vista de la plataforma se verifica los valores de acuerdo a los datos mostrados a continuación.

Las tramas enviadas por el enlace inalámbrico cuentan con un formato que consta de 20 paquetes TCP de 480 bits cada uno y un paquete HTTP de 1936 bits, dando un total de 11536 bits lo que es equivalente a 1,442 kbytes, por cada trama, considerando que el tiempo de duración es de 1 segundo y se envía las 5 variables planteadas temperatura, humedad, dióxido de carbono, humo y radiación (Pule, 2017):

Tomando en cuenta cada uno de estos datos, la trama que llega al sistema y va a ser almacenada de forma temporal en la memoria RAM antes de ingresar al sistema es de 1,442 kbytes por este motivo la cantidad de conexiones será limitada por la memoria, encontrando almacenadas en esta misma distintos buffers, pools, caches, el código de ejecución y el sistema operativo.

Aproximadamente ocupa 2 GB de la memoria RAM, sumando cada uno de los componentes mencionados, por lo que se cuenta con 2 GB para las conexiones dando un límite de conexiones posible de 1386963 conexiones soportadas por la memoria (Pagani, 2012).

4.4. Sistema de monitoreo

El sistema está montado en un servidor GNU/Linux sistema operativo de software libre, que se encuentra continuamente operativo, a la espera de la llegada de los datos provenientes de la WSN ubicada en el bosque protector Guayabillas, para realizar su posterior tratamiento y procesamiento de dicha información, antes de hacerla visible en forma de una interfaz web amigable e interactiva.

4.4.1. Inicialización del sistema

En esta parte para iniciar por primera vez el sistema, se ingresa el código especificado a continuación, para arrancar el software de monitoreo, el cual necesita ser iniciado solo una vez al momento de arrancar el sistema desde la consola de comandos como se muestra en la Figura 17.

- Ingresamos a la carpeta: cd glassfish-4.1.1/glassfish/bin
- Código inicialización: ./asadmin start-domain domain1

Figura 17 Terminal linux-Ubuntu inicio de aplicativo
Fuente: Elaboración propia.

4.4.2. Acceso al sistema

Previo al ingreso al sistema experto el usuario debe contar con los siguientes requisitos indispensables que garantizan la seguridad en cuanto a la visualización de la información.

- Contar con acceso a un dispositivo Smart (Smartphone, Tablet, PC, Smart Tv etc.) Que posea la capacidad de abrir un navegador web sea este (Firefox. Chrome. Opera, Edge, etc.), Para el correcto acceso el usuario debe encontrarse dentro de la red local o en su caso a través de internet para una visualización remota del sistema experto.
- Tener actualizados los complementos Java, Adobe flash player de acuerdo al navegador a utilizar, y los diferentes complementos gráficos necesarios dependiendo del dispositivo en el que vaya a ser visualizada la interfaz.

- Ingresar mediante el navegador web la siguiente dirección URL, donde se direcciona a la plataforma del sistema. Http://sistemaexpertoutn:8080/index.xhtml

4.4.3. Autenticación

Se realiza la autenticación por medio del ingreso de un usuario y contraseña previamente asignados, Usuario: admin Contraseña: info.2017, a continuación, dar clic en el botón de inicio como se aprecia en la Figura 18.

Figura 18 Pantalla principal autenticación
Fuente: Elaboración propia.

4.4.4. Interfaz de monitoreo

Una vez realizada la autenticación se permite el ingreso a la página principal donde se encuentran mostrados los valores del último nodo que ingreso al sistema, la visualización es gráfica

y se realizó mediante el desarrollo de plugins representado así cada uno de los valores obtenidos y procesados por el sistema, así como se observa en la Figura 19.

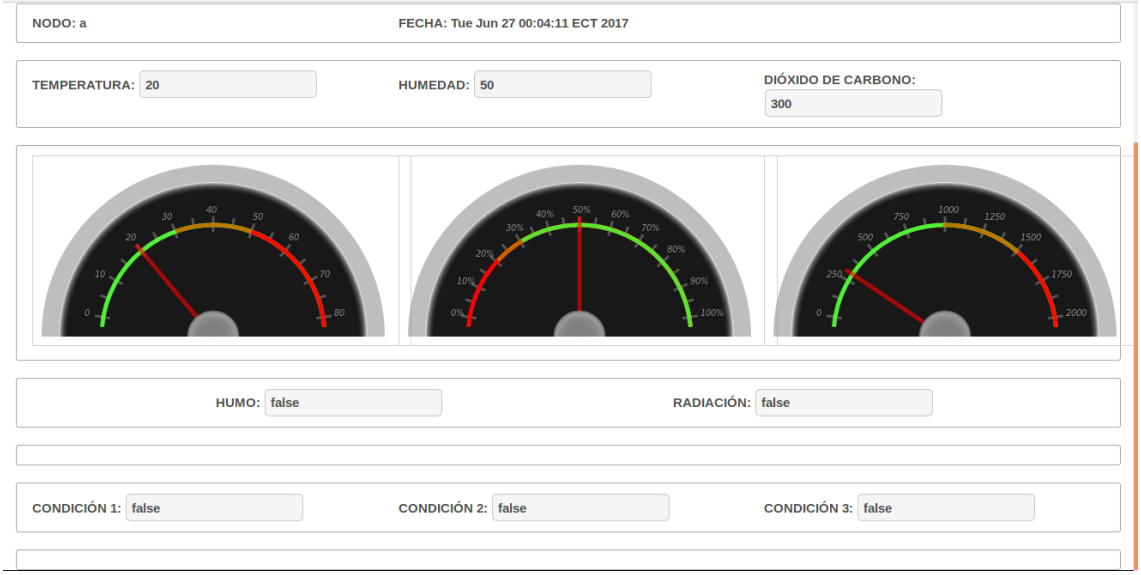


Figura 19 Pantalla principal sistema monitoreo
Fuente: Elaboración propia.

4.4.5. Alarmas

Las alarmas es la parte primordial en el sistema ya que serán las encargadas de notificar al administrador la existencia de cambios de brusco en el entorno aplicado, sobrepasando los niveles establecidos, con la capacidad de identificar el tipo de alarma que se ha producido ya que se muestran los valores de forma textual y gráfica, como se aprecia en la Figura 20, para su correcta interpretación.

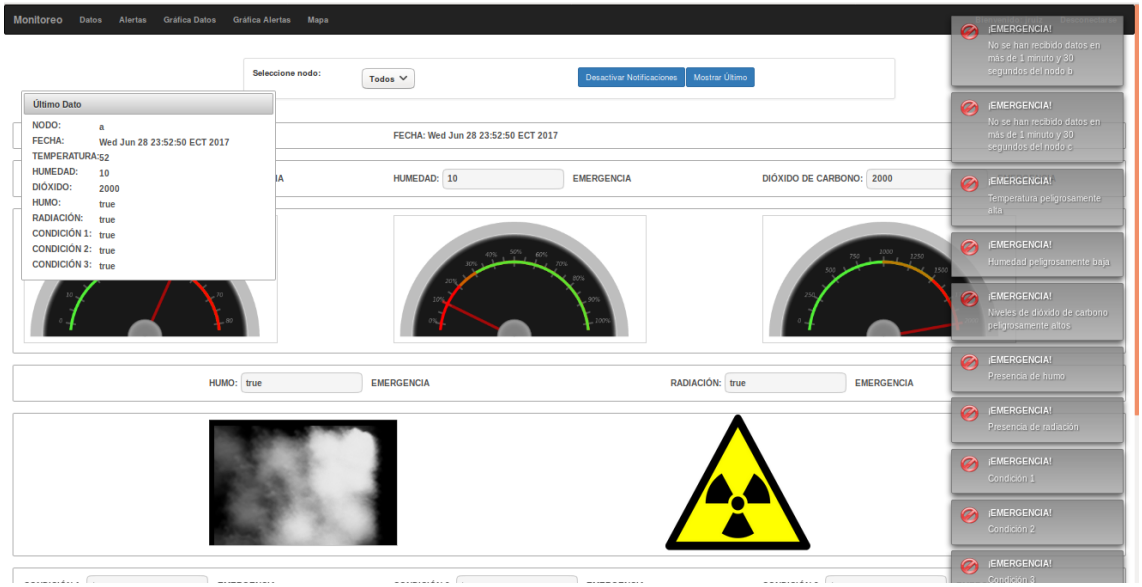


Figura 20 Visualización de alarmas
Fuente: Elaboración propia.

De acuerdo con lo establecido y el correcto despliegue de los organismos encargados de socorro en este punto el encargado realiza la llamada al centro integrado de seguridad ECU-911, quien se encarga del correcto manejo y disponibilidad operativa de su personal encargado Figura 21, se aprecia al personal encargado de la recepción de las llamadas. (capítulo 3).

Figura 21 SIS ECU-911
Fuente: (ECU-911, 2017)

4.4.6. Consulta historial datos recabados

El sistema se encarga de realizar el almacenamiento de los diferentes valores que son ingresados a la base de datos y poderlos presentar de forma clara para realizar informes de acuerdo a los datos relevantes y distintas mediciones, para su interpretación técnica, estos valores sirven para la toma de decisiones en cuanto a acciones de prevención y futuro plan de contingencia, el sistema permite filtrar cada uno de los datos de acuerdo a las necesidades del encargado en su revisión como se muestra en la Figura 22 mostrada a continuación.

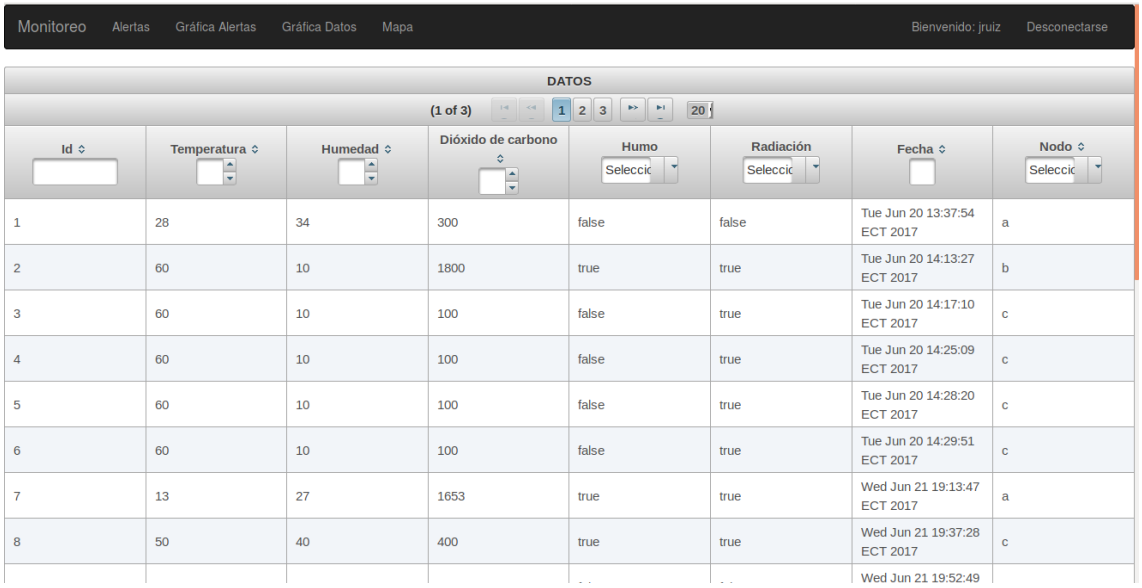


Figura 22 Historial Datos
Fuente: Elaboración propia.

4.4.7. Gráfica datos recabados

Para una comprobación gráfica de los valores que se encuentran almacenados, el sistema cuenta con la parte de visualización representada en diagrama de barras, como se muestra en la Figura 23, esto servirá para una interpretación mucho más clara del comportamiento del entorno en el que se encuentra aplicado el sistema.

Los datos almacenados para la temperatura con la siguiente estructura, el identificador de valor con el cual se guardó el dato, la fecha, el dato marcado por el sensor, el identificador del nodo, esta información se muestra de color verde para diferenciar del resto de datos, la gráfica se encuentra en grados centígrados.

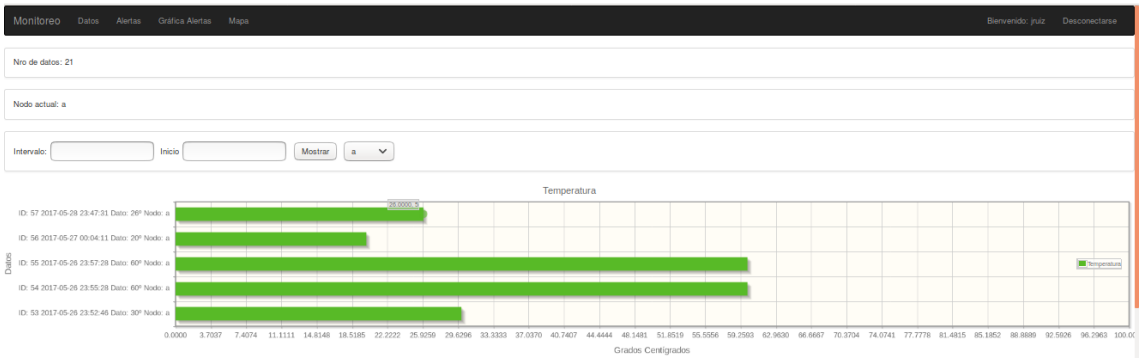


Figura 23 Temperatura
Fuente: Elaboración propia.

A continuación, se muestra la Figura 24, donde se aprecia de color azul a los datos recabados de la humedad relativa presente en el entorno aplicada. La escala se encuentra en porcentaje (%).

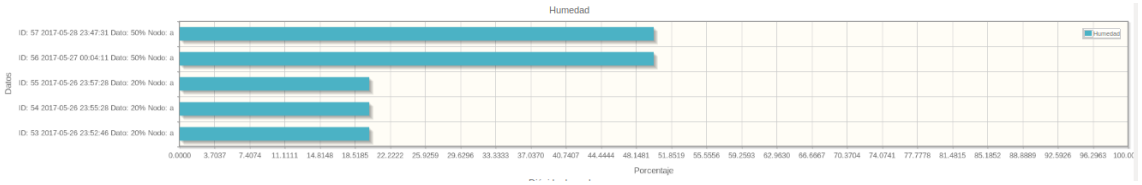


Figura 24 Humedad Relativa
Fuente: Elaboración propia.

En la Figura 25, que se aprecia a continuación se encuentra representada de color naranja los datos almacenados referentes al Dióxido de carbono, estos datos se encuentran en partículas por millón(ppm).

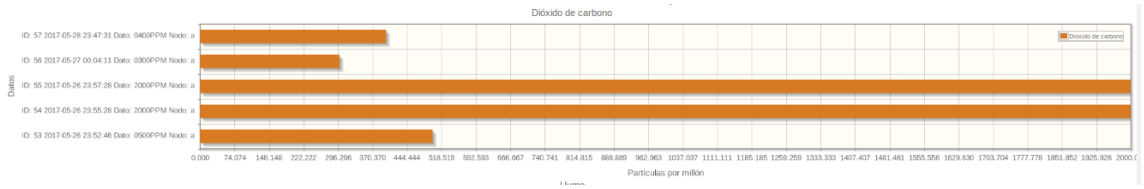


Figura 25 Dióxido de Carbono
Fuente: Elaboración propia.

Seguido se muestra la Figura 26, donde se encuentra representado por el color morado si existe la presencia de humo, este dato recabado por el sensor tiene solo 2 estados, si existe o no existe presencia de humo.

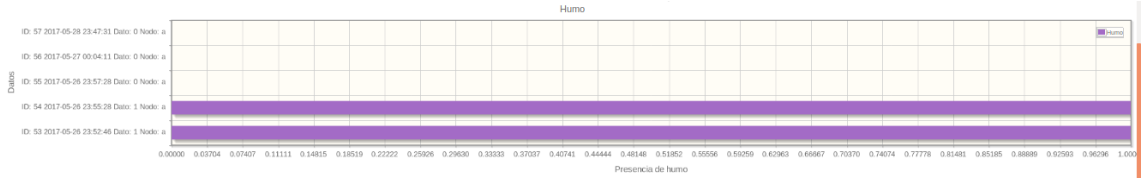


Figura 26 Humo
Fuente: Elaboración propia.

En la Figura 27, se observa de color rosa si existe la presencia de Radiación, este sensor tiene solo 2 estados, si existe o no existe, el dato representado de color quiere decir que existe presencia de radiación.

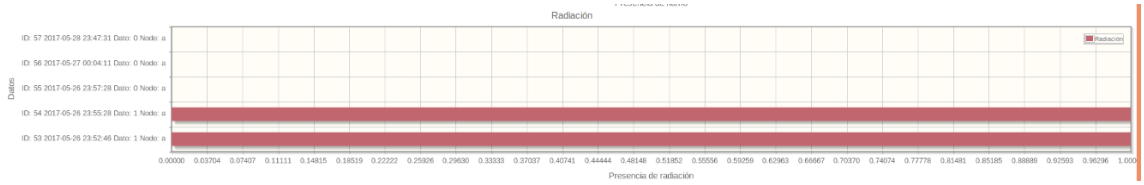


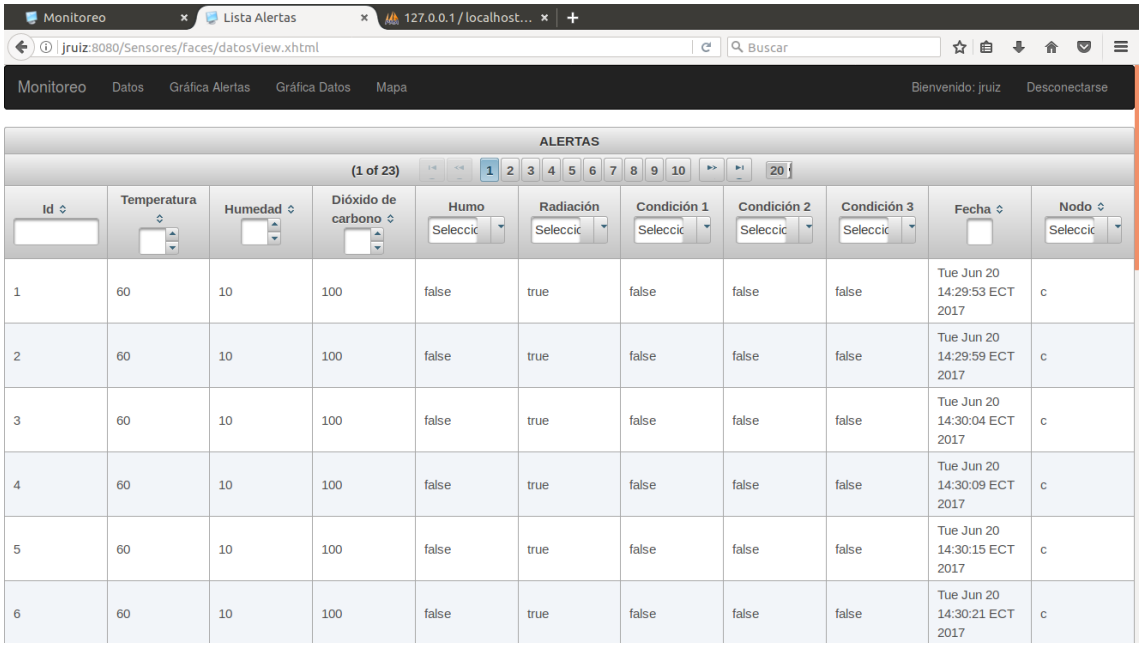
Figura 27 Gráfica Radiación
Fuente: Elaboración propia.

4.4.8. Consulta historial alarmas

El sistema cuenta con la capacidad de comprobación de cada uno de los valores ingresados para evitar problemas en cuanto a la generación de las llamadas erróneas y posibles complicaciones legales. Por tal motivo el sistema informa de manera gráfica y auditiva, que llegó un valor que posiblemente genera emergencia, a esto el sistema espera los siguientes 2 datos para comprobar si la alerta es verdadera, en el transcurso de 1 minuto y 30 segundos al comprobarse que los 3 valores marca emergencia se guarda la alarma en la base de datos.

Esta validación de información se genera ya que como se especificó anteriormente, si existe presencia de vientos en la zona los datos pueden variar abruptamente, y tomar un tiempo prudencial de 57 segundos para la toma correcta de valores, lo que incurre en error si los datos son enviados cada 30 segundos. Así como también por tratarse de un medio inalámbrico el utilizado para el envió de información hacia la UTN cabe la posibilidad de que exista interferencia electromagnética la cual podría generar algún tipo de alteración a los datos recabados.

Para visualización y comprobación se muestra en forma de historial cada alarma real generada, en la Figura 28, se muestran datos de alarmas reales y su hora exacta en la que se registró, cuenta con la capacidad de filtrar la información almacenada de acuerdo a las necesidades del encargado.



Id	Temperatura	Humedad	Dióxido de carbono	Humo	Radiación	Condición 1	Condición 2	Condición 3	Fecha	Nodo
1	60	10	100	false	true	false	false	false	Tue Jun 20 14:29:53 ECT 2017	c
2	60	10	100	false	true	false	false	false	Tue Jun 20 14:29:59 ECT 2017	c
3	60	10	100	false	true	false	false	false	Tue Jun 20 14:30:04 ECT 2017	c
4	60	10	100	false	true	false	false	false	Tue Jun 20 14:30:09 ECT 2017	c
5	60	10	100	false	true	false	false	false	Tue Jun 20 14:30:15 ECT 2017	c
6	60	10	100	false	true	false	false	false	Tue Jun 20 14:30:21 ECT 2017	c

Figura 28 Historial Alertas
Fuente: Elaboración propia.

4.4.9. Gráfico de alertas históricas

Para una comprobación gráfica de las emergencias generadas, que se encuentran almacenadas, el sistema cuenta con la parte de visualización representada en diagrama de barras, como se muestra en las siguientes ilustraciones, esto servirá para una interpretación mucho más clara del comportamiento del entorno en el que se encuentra aplicado el sistema,

Como se observa en la Figura 29, muestra los datos almacenados si existo la comprobación de 3 valores, sobre los rangos de tolerancia, para la temperatura con la siguiente estructura, el identificador de valor con el que se guardó el dato, la fecha, el dato marcado por el sensor, el identificador del nodo, esta información se muestra de color verde para diferenciar del resto de datos, la gráfica se encuentra en grados centígrados.

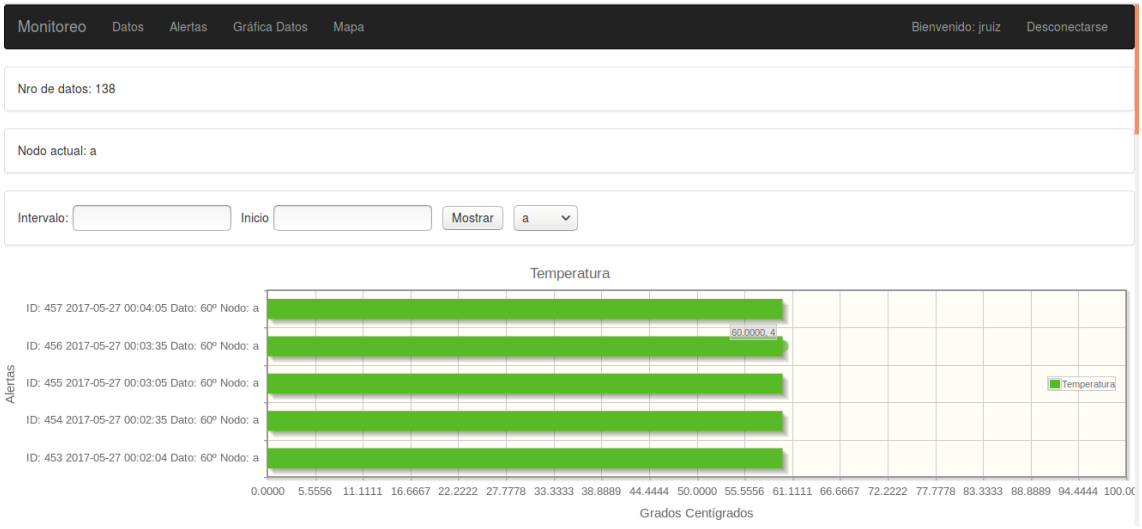


Figura 29 Gráfica Alertas Temperatura
Fuente: Elaboración propia.

A continuación, se muestra la Figura 30, donde se aprecia de color azul los datos recabados de humedad relativa presente en el entorno aplicado. La escala se encuentra en porcentaje (%).

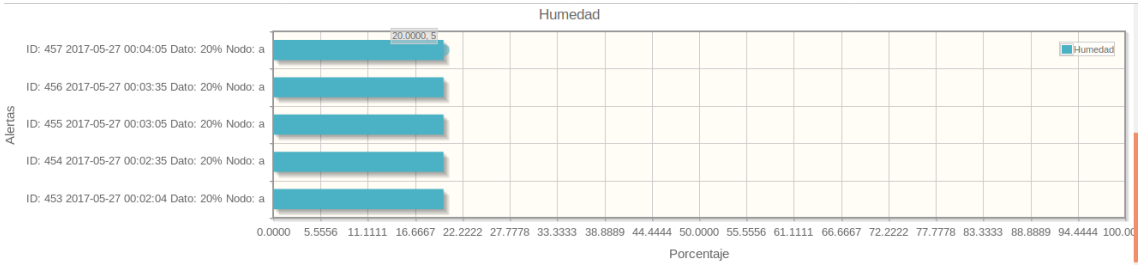


Figura 30 Gráfica Alertas Humedad
Fuente: Elaboración propia.

En la Figura 31, que se aprecia a continuación se encuentra representada de color naranja los datos almacenados referentes al Dióxido de carbono, estos datos se encuentran en partículas por millón(ppm).

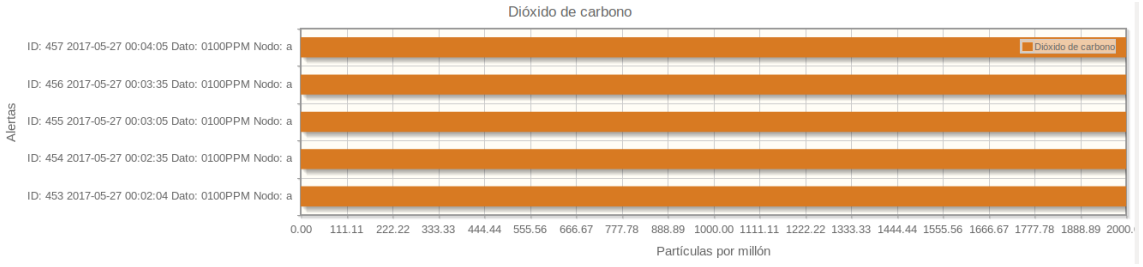


Figura 31 Gráfica Alertas Dióxido de Carbono
Fuente: Elaboración propia.

Seguido se muestra la Figura 32, donde se encuentra representado por el color morado si existe la presencia de humo, este dato recabado por el sensor tiene solo 2 estados, si existe o no existe presencia de humo.

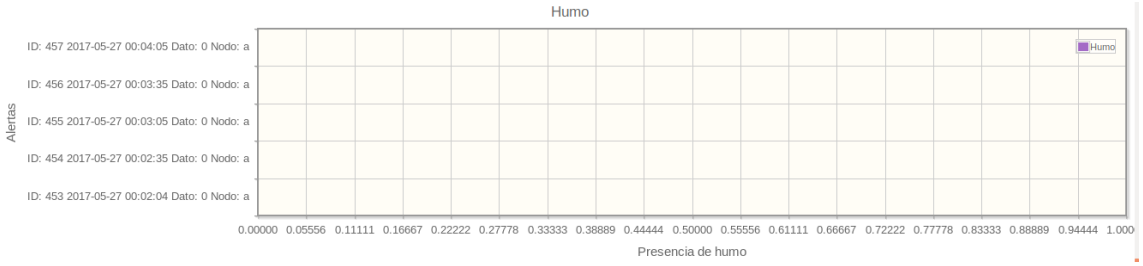


Figura 32 Gráfica Alertas Presencia de Humo
Fuente: Elaboración propia.

En la Figura 33, se observa de color rosa si existe la presencia de Radiación, este sensor tiene solo 2 estados, si existe o no existe, el dato representado de color quiere decir que existe presencia de radiación.

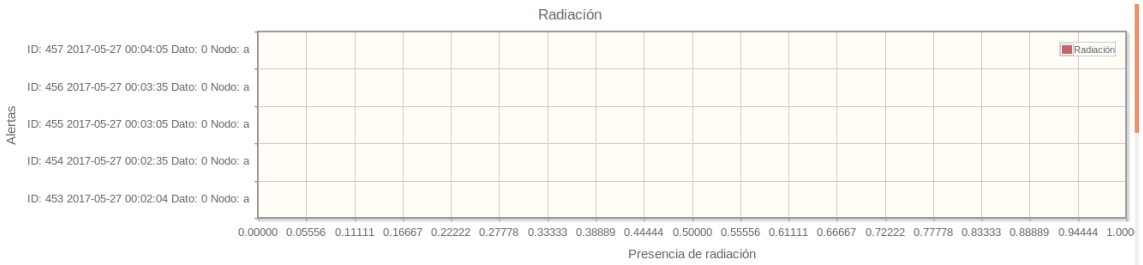


Figura 33 Gráfica Alertas Presencia Radiación
Fuente: Elaboración propia.

En la Figura 34, se encuentra los datos almacenados cuando los valores sobrepasan los rangos de tolerancia y exista la comprobación de los 3 datos antes de guardar la alerta, y de acuerdo a la condición 1, envía un mensaje más de alerta indicando que todos los sensores entraron en estado de emergencia.

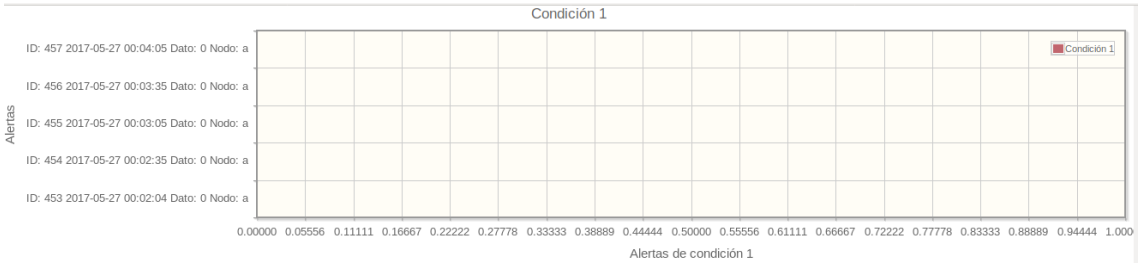


Figura 34 Gráfica Alertas Condición 1
Fuente: Elaboración propia.

En la Figura 35, se muestra los datos almacenados cuando los valores sobrepasan los rangos de tolerancia y exista la comprobación de los 3 datos antes de guardar la alerta, y de acuerdo a la condición 2, envía un mensaje más de alerta indicando que la temperatura, la humedad y el Dióxido de carbono entraron en estado de emergencia.

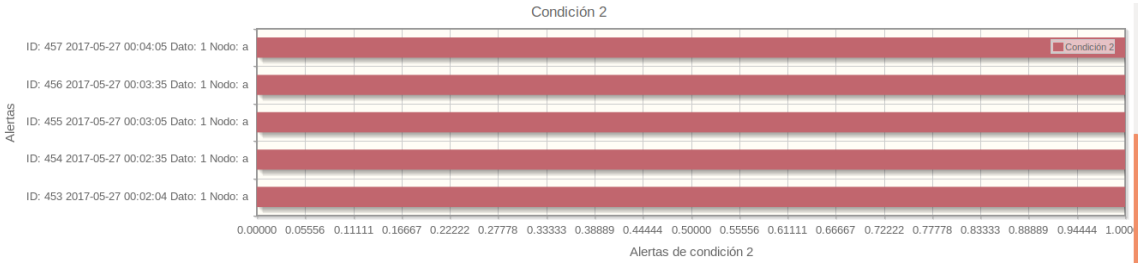


Figura 35 Gráfica Alertas Condición 2
Fuente: Elaboración propia.

En la Figura 36, se muestra los datos almacenados cuando los valores sobrepasan los rangos de tolerancia y exista la comprobación de los 3 datos antes de guardar la alerta, y de acuerdo a la condición 1, envía un mensaje más de alerta indicando que los sensores de temperatura y presencia de Humo entraron en estado de emergencia.

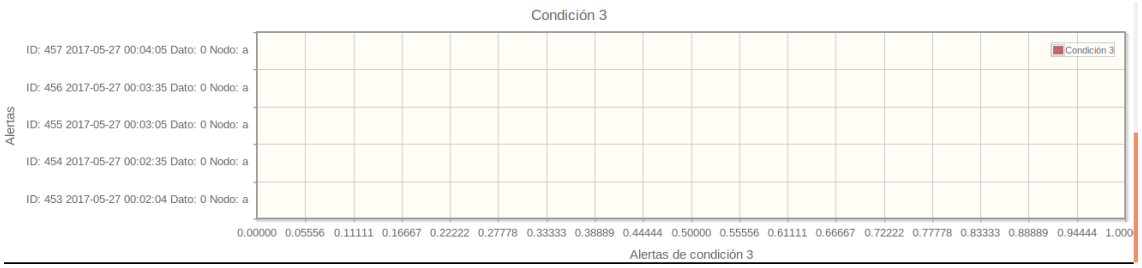


Figura 36 Gráfica Alertas Condición 3
Fuente: Elaboración propia.

4.5. Informe de factibilidad

En el presente informe se determina la viabilidad del proyecto realizado tomando en cuenta los puntos de vista técnicos, económicos y operativos, que permitirán determinar si el desarrollo del mismo se encuentra apto para su implementación y sus futuras modificaciones fuera del alcance de este proyecto de grado.

4.5.1. Factibilidad técnica o tecnológica

A continuación, se describe de forma breve los componentes involucrados en el desarrollo del sistema tanto en software como en hardware, tomando en cuenta cada uno de los requisitos anteriormente descritos.

4.5.1.1. Software y herramientas a utilizar

El sistema experto cumpliendo con el apoyo al sistema de alerta temprana contra incendios forestales, ubicado en el bosque protector Guayabillas, se encuentra basada en software libre mediante la utilización de la distribución Linux en su versión Ubuntu 16,04, sobre el que será desarrollado en el lenguaje de programación java en la plataforma NetBeans, así mismo una base de datos desarrollada en MySQL con el administrador Phpmyadmin.

El software utilizado se basa en plataformas de libre desarrollo con el cual se puede implementar diferentes tipos de aplicaciones ya sea de escritorio o a su vez en entornos web, de acuerdo a las necesidades del entorno, estas plataformas pueden encontrarse en internet con la facilidad de realizar la descarga de forma gratuita.

Al ser Java considerada una de las plataformas más seguras brinda al desarrollador la capacidad de utilidad de procesos ordenados en el desarrollo y posterior publicación (Ortega, 2014).

4.5.1.2. Hardware

El presente desarrollo será llevado a cabo en una computadora portátil HP Pavilion 15 AMD A10-5745M con 8,0 GB de memoria ram, y con la disponibilidad de 100 GB para almacenamiento de datos, cumpliendo con las capacidades necesarias para la implementación y su correcta ejecución del aplicativo.

Técnica y tecnológicamente el desarrollo del sistema es factible, ya que en sus requerimientos e implementación cumple con el desarrollo en software libre, lo que conlleva a no incurrir en gastos excesivos en la adquisición de productos de desarrollo, así como también se dispone del conocimiento necesario para el correcto desarrollo de este tipo de sistemas.

4.5.2. Factibilidad económica

En el presente informe serán tomados en cuenta cada uno de los recursos económicos necesarios para el correcto despliegue del sistema experto, tomando en cuenta que cada uno de los

ítems especificados a continuación forman parte del informe y esto incurre en costos tanto en hardware, software, y recursos humanos

- Costos de Hardware.

El costo de hardware que se especifica a continuación está dado según lo establecido en el informe de factibilidad técnica, y el equipo que se adapte a los requerimientos mínimos.

Costo Computador $730 (Novicompu, 2017).

- Costos en software y licencias.

Software utilizado es la Plataforma NetBeans con el lenguaje de programación java, base de datos Mysql con su administrador Phpmyadmin. Distribuciones libres que no necesitan licencias.

Costo $0

- Costo de recursos Humanos

De acuerdo a la consulta de personas especializadas en el desarrollo de software los honorarios asignados al desarrollador son los mostrados en la Tabla 23, los costos asignados por concepto del análisis, desarrollo, implementación y documentación del sistema experto, en este caso se cuenta con valores que pueden variar dependiendo de la dificultad y medios donde pueden ser implementados.

Tabla 23 Costos Recursos Humanos

Recursos	**Cantidad**	**Precio $**	**Subtotal $**
Análisis	1	500	500
Desarrollo	1	1000	1000
Implementación	1	500	500
Documentación	1	500	500
Total			2500

Fuente: Elaboración propia.

4.5.2.1. Ventajas económicas

Al ser un sistema desarrollado en plataformas libres y locales se controla el crecimiento considerando las necesidades del entorno sin verse limitado en número de nodos, generando así la principal ventaja frente a plataformas IoT desarrolladas y puestas en la nube en las cuales el crecimiento es muy costoso. Y no cumplen con la adaptación necesaria al medio donde se encuentra aplicado el sistema. Con la mayor venta de reducción de costos de implementación ya que los valores con costo serían por desarrolló y no por tema de capacitaciones y pago de licencias, en cierto caso excesivas, como se aprecia en la Tabla 24.

Tabla 24 Costos de inversión

Detalle	**Subtotal**
Equipo	$730
Licencias	$0
Recursos Humanos	$2500
Total	$3230

Fuente: Elaboración propia.

Es conveniente la implementación del sistema ya que, al tratarse de un problema social, este debe encontrarse totalmente adaptado y generar crecimiento para futuras expansiones, así como también contar con los equipos que puedan sustentar el despliegue del sistema.

4.5.3. Factibilidad operativa

El sistema va dirigido al encargado de monitoreo ya que será quien se encuentre en la capacidad de interpretación de las diferentes alertas generadas, para su correcta utilización, los conocimientos son básicos de linux ya que la mayor parte será generada dentro de la plataforma la misma es intuitiva y amigable con el usuario.

Desde el punto de vista operativo el usuario encargado no va a encontrarse con inconvenientes, por tal motivo es totalmente factible su aplicación, ya que antes de generar y almacenar la alerta se encarga de validar la información 3 veces, en intervalo de 1 minuto y 30 segundos, permitiendo así evitar problemas legales de falsas alarmas.

4.5.4. Resultados de factibilidad

Como puede especificarse en cada uno de los ítems anteriores, tanto la factibilidad técnica, económica y operativa resultaron favorables en cuanto a su correcto despliegue del sistema. El resultado final brinda las capacidades inicialmente planteadas y su utilidad será de mucha importancia para la mitigación de incendios forestales a gran escala, lo que será de ayuda para los

organismos de socorro encargados, con gran contribución ambiental. Así como también cumple con las funciones planteadas inicialmente en cada uno de los objetivos.

De acuerdo a datos obtenidos del incendio forestal generado en agosto del 2014 el bosque protector guayabillas sufrió la afectación de aproximadamente 30 hectáreas de las 54 que lo conforman, el fuego fue controlado pasadas las 2 horas de trabajo arduo tanto de bomberos militares, personas cercanas al lugar, y con la ayuda de un helicóptero policial (Rosero, 2014).

Para evitar que estos hechos se produzcan nuevamente, el desarrollo del sistema experto genera una alerta temprana antes de que el incendio se genere, si es de forma natural o de pronta respuesta si es producido por personas inescrupulosas, evitando así que gran parte del bosque se vea afectado y reduciendo al mínimo el impacto del fuego.

De acuerdo a datos del Ministerio del Ambiente, controlar un incendio puede demandar entre USD 5 000 y 20 000, por hora. Y puede llegar hasta los USD 70 000 si usan helicópteros con bolsas de agua. La recuperación tarda décadas. Los costos no solo son económicos, lo más grave es el daño a la naturaleza (Paucar, 2017).

4.6. Funciones del sistema experto

- **Monitorización:** La plataforma realiza varias comparaciones y validación de información antes de almacenar una alerta, para evitar que se genere llamadas erróneas y un despliegue innecesario de organismos emergentes. Lo que permite anticipar un incendio forestal generado por causas naturales.

- **Diseño:** El diseño de la plataforma fue enfocada a satisfacer las necesidades existentes en el sistema de alerta temprana en el bosque protector guayabillas, el cual satisface las necesidades planteadas por los requisitos planteados en sus objetivos.
- **Planificación:** La creación del sistema se encuentra especificado en el capítulo 3, que va desde los requisitos, la elección de hardware y software, normalización de base de datos, aplicación del estándar 29148 y 42010.
- **Control:** La recepción, procesamiento, generación de alerta, y visualización de información son llevadas a cabo por medio del código generado en Java el cual permite al usuario tomar acciones correctivas frente a una emergencia englobada en los rangos de tolerancia configurados.
- **Simulación:** Para fines de estudio del entorno, los datos que serán ingresados de forma manual, con datos aleatorios los cuales permiten un análisis de tiempos de respuesta y toma de decisiones oportunas por el operador del sistema.
- **Recuperación de información:** Cuenta con la capacidad de mostrar en forma de historial las diferentes variantes generadas en la zona de cobertura de los sensores. Lo que sirve para generar un informe forestal especializado dependiendo del enfoque y la afectación presente en el medio.

CAPÍTULO 5. CONCLUSIONES Y RECOMENDACIONES

5.1. Conclusiones

Se realizó la implementación de un sistema experto, el cual se encarga del procesamiento, almacenamiento y visualización de los datos provenientes de la WSN, ubicada en el bosque protector Guayabillas, permitiendo al usuario la facilidad de manipulación de los datos y generación de informes ambientales de ser estos necesarios.

Luego de la investigación bibliográfica realizada a los sistemas expertos existentes y a diferentes tipos de plataformas de adquisición de datos, el sistema de alerta temprana tendrá una validación de información, al ingresar los datos cada 30 segundos serán almacenados en la tabla de datos, posterior la plataforma tarda 1 minuto y 30 segundos para guardar la alerta y de ahí poder desplegar los organismos de socorro y la secretaría de gestión de riesgos si fuese necesario.

Para evitar que exista redundancia de datos y esto complique el procesamiento de la información se realizó la normalización de la base de datos de acuerdo a las necesidades ambientales, protegiendo así la integridad de los datos y su generación real de alarmas.

Se identificó la lógica del negocio y se procedió a la codificación de todas las funciones y rangos de tolerancia que el sistema soporta antes de enviar una alerta, la información creada por los sensores va a ser almacenada en la base de datos y posteriormente mostrada en la plataforma con sus rangos específicos de tolerancia.

El diseño del sistema experto cumple con las características necesarias para poder ser implementado en sistema de alerta temprana aplicado al bosque protector Guayabillas ya que como se demostró en el capítulo 4 los datos cumplen las características y funciones que son necesarias.

El principal propósito de creación del sistema experto fue cumplido, ya que brinda capacidades de adaptación y crecimiento sin incurrir en costos excesivos de implementación, así como también tener la seguridad de que los datos que se encuentran almacenados están siendo protegidos dentro de la red local.

5.2. Recomendaciones

Al ser un desarrollo externo al sistema integrado de seguridad ECU-911, las llamadas retrasan de cierta forma el despliegue de los organismos, para lo cual se recomienda la asesoría de la secretaría de gestión de riesgos para una implementación de esta plataforma en diferentes entornos donde puede ayudar a la mitigación de incendios forestales.

La plataforma debe brindar seguridad frente a diferentes tipos de anomalías que pueden presentarse, por tal motivo se recomienda la posterior validación por medio del proyecto OWASP (Open Web Application Security Project), el cual se enfoca en la mejora de seguridad de software.

Una manera de optimizar el proyecto es la implementación de la variable viento, ya que es de mucha importancia cuando el incendio forestal se encuentra en expansión, y el viento juega un papel clave en la dirección del fuego.

Mantener el tiempo de validación como parte fundamental del proyecto, ya que, por normativa vigente en Ecuador, al incurrir en llamadas erróneas, la institución encargada del monitoreo puede encontrarse en faltas legales graves.

REFERENCIAS

Barboza, P. (31 de Marzo de 2014). *Secretaria de alerta temprana.* Obtenido de http://www.who.int/ihr/publications/WHO_HSE_GCR_LYO_2014.4es.pdf

Barreto, F. (13 de Mayo de 2014). *IEEE 29148*. Obtenido de https://prezi.com/8mki1yjihkdp/ieee-29148/

Blázquez, F. (2001). *Sociedad de la información y educación* . Obtenido de http://www.ub.edu/prometheus21/articulos/obsciberprome/blanquez.pdf

Cadena, T. C. (29 de Junio de 2017). Recepcion de emergencia por incendios forestales. (J. Ruiz, Entrevistador)

Carriots. (2017). *Carriots.* Obtenido de https://www.carriots.com/

CIIFEN. (2014). Metodología para la Estimación de Vulnerabilidad en Ecuador, Perú y. págs. 1 - 45.

ECU-911. (2017). *Sistema integrado de seguridad ECU-911.* Obtenido de http://www.ecu911.gob.ec/

EIRD. (2010). *EIRD.* Obtenido de http://www.eird.org/index-esp2.html

Engelmore, R. S. (2010). *EXPERT SYSTEMS AND ARTIFICIAL INTELLIGENCE.* Obtenido de http://www.wtec.org/loyola/kb/c1_s1.htm

exosite. (2017). *exosite murano*. Obtenido de https://exosite.com/

Fuentes, I. (12 de Abril de 2004). *Publicaciones I&IMS.* Obtenido de http://bertoki.com.ar/portal_educativo/gestion_recursos_tecnologicos/documentos/digitalizacion_de_documentos.pdf

González, Y. (15 de Mayo de 2013). *ICEMD.* Obtenido de http://blogs.icemd.com/blog-estrategias-de-marketing-percepcion-o-realidad-/psicologia-del-color-en-el-logo-de-una-marca/

IEEE. (2011). *IEEE-SA*. Obtenido de https://standards.ieee.org/findstds/standard/29148-2011.html

ISO. (12 de 2011). *International Organization for Standardization.* Obtenido de https://www.iso.org/standard/45171.html

Kruchten, P. (Noviembre de 1995). *Architectural Blueprints—The "4+1" View.* Obtenido de http://www.win.tue.nl/~wstomv/edu/2ip30/references/Kruchten-4+1-view.pdf

López, I. W. (30 de Junio de 2017). Despliegue de organismos de socorro ECU-911. (J. Ruiz, Entrevistador)

Microsoft. (12 de Julio de 2013). *Prueba Microsoft Edge*. Obtenido de https://support.microsoft.com/es-ec/help/283878/description-of-the-database-normalization-basics

Microsoft. (17 de Enero de 2017). *Microsoft.* Obtenido de https://support.microsoft.com/es-ec/help/827968/how-to-estimate-the-size-of-a-sql-server-ce-or-sql-server-2005-mobile

Microsoft. (17 de Enero de 2017). *Microsoft.* Obtenido de https://support.microsoft.com/es-ec/help/827968/how-to-estimate-the-size-of-a-sql-server-ce-or-sql-server-2005-mobile

Musumeci, V. (11 de Abril de 2016). *Valentina Musumeci Blog.* Obtenido de http://valentinamusumeci.com/blog/teoria-psicologia-color/

Novicompu. (Junio de 2017). *Novicompu*. Obtenido de https://www.novicompu.com/laptops/3264/laptop-hp-15-ay191-i3-7100-1tb-8gb-156-touch-dvd-rw-win10.html?search_query=hp+pavilion+15&results=334

Ortega, M. (5 de Febrero de 2014). *Betabeers.* Obtenido de https://betabeers.com/forum/que-lenguaje-backend-tiene-mas-futuro-572/

Ortiz, J. L. (31 de Marzo de 2017). *BENCHMARK DE SELECCIÓN DE SENSORES PARA UNA WSN*. Obtenido de http://repositorio.utn.edu.ec/handle/123456789/6527

Pagani, J. B. (17 de Febrero de 2012). *Investigación en tecnologias de la información.* Obtenido de http://investigacionit.com.ar/es/optimizacion-de-bases-de-datos-mysql/

Paucar, E. (21 de Julio de 2017). 100 000 hectáreas afectadas en cinco años por incendios forestales. Quito, Pichincha, Ecuador.

Pérez, M. A. (18 de Junio de 2015). *Sistemas expertos optimizados.* Obtenido de https://sistemasexpertostsu.wordpress.com/2015/06/18/estructura-basica-de-un-sistema-experto-2/

Pérez, M. A. (19 de Junio de 2015). *Sistemas Expertos Optimizados.* Obtenido de https://sistemasexpertostsu.wordpress.com/2015/06/18/funciones-del-sistema-experto/

(2013-2017). *Plan Nacional Del Buen Vivir.* QUITO.

Pule, D. (14 de Junio de 2017). *Repositorio UTN.* Obtenido de http://repositorio.utn.edu.ec/handle/123456789/6896

Rodríguez, J. (28 de Septiembre de 2008). *Unidad de Desarrollo Tecnológico.* Obtenido de http://www.iiia.csic.es/udt/es/blog/jrodriguez/2008/metodologia-desarrollo-sotware-modelo-en-v-o-cuatro-niveles

ROSENBERG, D. A. (Abril de 2016). *DISEÑO DE UNA INTERFAZ DE MONITOREO PARA SERVICIOS.* Obtenido de http://repositorio.ug.edu.ec/bitstream/redug/11963/1/B-CINT-PTG-N.44%20ROSENBERG%20MIRANDA%20DENISSE%20ALEXANDRA%20%20Y%20%20O%C3%91ATE%20CERVANTES%20ERICK%20TRAJANO.pdf

Rosero, A. (25 de Agosto de 2014). Incendio consume bosque protector de Guayabillas. Ibarra, Imbabura, Ecuador.

Stevens, L. (1985). *Artificial intelligence.* Hasbrouck Heights, New Jersey.

Thingspeak. (2017). *Thingspeak.* Obtenido de https://thingspeak.com/

ubidots. (2016). *ubidots*. Obtenido de https://ubidots.com/

UNESCO. (2011). *UNESCO*. Obtenido de http://www.unesco.org/new/es/natural-sciences/special-themes/disaster-risk-reduction/geohazard-risk-reduction/early-warning-systems/

xively. (2017). *xively*. Obtenido de https://www.xively.com/

GLOSARIO DE TÉRMINOS Y ACRÓNIMOS

TCP. (Transmission Control Protocol) El protocolo de control de trasmisión, esencial para las comunicaciones en internet

HTTP. (Protocolo de transferencia de hipertexto) Permite la transferencia de información en la internet

RAM. (Radom Access Memory), memoria de acceso aleatorio, donde se carga las instrucciones que se encuentran en ejecución por el procesador, tiene la capacidad de leer y escribir en cualquier posición sin ser secuencial.

WSN (Wireless Sensor Network) Red de sensores inalámbricos, utilizados en su mayoría para toma de valores ambientales o físicos dentro de medios controlados.

Nodo Lugar donde se encuentran concentrados varios sensores

Linux. Sistema operativo de licencia libre o código abierto, utiliza kernel GNU el cual le permite ser modificado.

PNBV. (Plan Nacional del Buen Vivir)

PPM. (Partes por millón) unidad de medida utilizada para el dióxido de carbono.

WEB. Página electrónica que contiene información variada y su acceso es por medio de un navegador.

IoT. (Internet of Things), Internet de las cosas

IEEE. Instituto de ingenieros eléctricos y electrónicos, encargado de la normalización y estructura de estándares especializados en tecnología de la comunicación.

UML. Lenguaje Unificado de Modelado.

SIS. Sistema Integrado de Seguridad.

Tcnl. Teniente coronel.

UTN. Universidad técnica del norte

CIERCOM. Carrera de ingeniería en electrónica y redes de comunicación

FICA. Facultad de ingeniería en ciencias aplicadas

StRU. Documento de especificación de requerimientos del usuario

SyRS. Documento de especificación de requerimientos del sistema

SrSH. Documento de especificación de requerimientos de Software

JSF. Java Server Faces, es un framework de desarrollo basado en el patrón MVC (Modelo Vista Controlador).

ANEXOS

Anexo 1. Entrevista SIS ECU-911

Entrevista personal en el centro integrado de seguridad ECU-911 de la ciudad de Ibarra sector, Parque “Ciudad Blanca”. Personas en la fotografía de derecha a izquierda. Coordinador zonal 1 Tcnl. Carlos Cadena. Ing Wilmer López encargado de la dirección de proyectos e innovación de dicho centro. Sr Jefferson Ruiz estudiante encargado de la entrevista.

Anexo 2. Implementación control de ingreso

LoginDAO

```
package jsf.dao;

import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;
import jsf.util.DataConnect;
public class LoginDAO {
public static boolean validate(String user, String password) {
Connection con = null;
PreparedStatement ps = null;
try {
con = DriverManager.getConnection("jdbc:mysql://localhost/sensores","root","VinicioRuiz");

ps = con.prepareStatement("Select id_Usuario, contrasenia from usuario where id_Usuario = ? and contrasenia
= ?");
ps.setString(1, user);
ps.setString(2, password);
ResultSet rs = ps.executeQuery();
if (rs.next()) {
//result found, means valid inputs
return true;}
} catch (SQLException ex) {
System.out.println("Login error -->" + ex.getMessage());
return false;
} finally {
DataConnect.close(con);}
return false;}}
```

Anexo 3. Controlador de datos y generador de alertas

datosController

```
/*
 * To change this license header, choose License Headers in Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */
package beans.sessions;
import java.io.Serializable;
import java.util.ArrayList;
import java.util.Date;
import java.util.List;
import java.util.Locale;
import javax.annotation.PostConstruct;
import javax.ejb.EJB;
```

```
import javax.faces.application.FacesMessage;
import javax.faces.bean.ManagedBean;
import javax.faces.bean.SessionScoped;
import javax.faces.context.FacesContext;
import org.primefaces.context.RequestContext;
import sensores.model.entities.Datos;
import sensores.model.manager.ManagerDatos;

/**
 *
 * @author jruiz
 */
@ManagedBean
@SessionScoped
public class datosController implements Serializable {

    private int idDatos = 0;
    private int temperatura;
    private int humedad;
    private int dioxido;
    private boolean humo;
    private boolean radiacion;
    private String nodo;
    private Date fecha;

    private boolean condicion1;
    private boolean condicion2;
    private boolean condicion3;
```

Anexo 4. Variables gráficas

```
    private int idDatosAux = 0;
    private int temperaturaAux;
    private int humedadAux;
    private int dioxidoAux;
    private boolean humoAux;
    private boolean radiacionAux;
    private String nodoAux;
    private Date fechaAux;

    private boolean condicion1Aux;
    private boolean condicion2Aux;
    private boolean condicion3Aux;

    private List<Datos> listaAux;

    private boolean alertaTemperaturaAux;
    private boolean alertaHumedadAux;
    private boolean alertaDioxidoAux;

    private boolean emergenciaTemperaturaAux;
    private boolean emergenciaHumedadAux;
```

```
private boolean emergenciaDioxidoAux;
```

Anexo 5. Variables alertas.

```
private boolean alertaTemperatura;
private boolean alertaHumedad;
private boolean alertaDioxido;

private boolean emergenciaTemperatura;
private boolean emergenciaHumedad;
private boolean emergenciaDioxido;

private List<Datos> lista;
private List<String> nodos;

int contador;
int idActual;
List<Integer> contadores;
List<Integer> idActuales;

boolean datosAusentes;
List<Boolean> listaDatosAusentes;

@EJB
private ManagerDatos itemService;

@PostConstruct
public void init() {
   lista = itemService.findDatos();
   nodos = cargarnodos();
   idActuales = new ArrayList<>();
   contadores = new ArrayList<>();
   listaDatosAusentes = new ArrayList<>();
   actionListenerCargarUltimoPorNodo("todos");
   actionListenerCargarUltimo();
}
public void actionListenerCargarUltimoPorNodo(String nodo) {
   lista = itemService.findDatos();
   if (nodo.equals("todos")) {
      listaAux = lista;
   } else {
      listaAux = itemService.findNodosByNodo(nodo);
   }
   Datos dato = null;
   if (listaAux.size() > 0) {
      dato = listaAux.get(listaAux.size() - 1);
   } else {
      return;
   }

   idDatosAux = dato.getIdDato();
   temperaturaAux = dato.getTemperatura();
   humedadAux = dato.getHumedad();
```

```
                dioxidoAux = dato.getDioxidoDeCarbono();
                humoAux = dato.getHumo();
                radiacionAux = dato.getRadiacion();
                nodoAux = dato.getNodo();
                fechaAux = dato.getFecha();
                condicion1Aux = false;
                condicion2Aux = false;
                condicion3Aux = false;
                monitorTemperaturaAux();
                monitorHumedadAux();
                monitorDioxidoAux();
                monitorHumoAux();
                monitorradiacionAux();

        if (monitorTemperaturaAux() == 2 && monitorHumedadAux() == 2 && monitorDioxidoAux() == 2 &&
monitorHumoAux() && monitorradiacionAux()) {
                    condicion1Aux = true; }
        if (monitorTemperaturaAux() == 2 && monitorHumedadAux() == 2 && monitorDioxidoAux() == 2) {
                    condicion2Aux = true;}
        if (monitorTemperaturaAux() == 2 && monitorHumoAux()) {
                    condicion3Aux = true;}}
           public void onNodoChange() {
              System.out.println("cambio de nodo");}
           public void actionListenerCargarUltimo() {
              lista = itemService.findDatos();
              Datos dato = null;
              if (lista.size() > 0) {
                 dato = lista.get(lista.size() - 1);
              } else {
                 return;}
              idDatos = dato.getIdDato();
              temperatura = dato.getTemperatura();
              humedad = dato.getHumedad();
              dioxido = dato.getDioxidoDeCarbono();
              humo = dato.getHumo();
              radiacion = dato.getRadiacion();
              nodo = dato.getNodo();
              fecha = dato.getFecha();
              condicion1 = false;
              condicion2 = false;
              condicion3 = false;

        if (monitorTemperatura() == 2 && monitorHumedad() == 2 && monitorDioxido() == 2 && monitorHumo()
&& monitorradiacion()) {
                    condicion1 = true;}
        if (monitorTemperatura() == 2 && monitorHumedad() == 2 && monitorDioxido() == 2) {
                    condicion2 = true;}
        if (monitorTemperatura() == 2 && monitorHumo()) {
                    condicion3 = true;}
        if (condicion1 || condicion2 || condicion3 || monitorTemperatura() > 0 || monitorHumedad() > 0 ||
monitorDioxido() > 0 || monitorHumo() || monitorradiacion()) {
                    List<Datos> listaNodos = itemService.findNodosByNodo(nodo);
                    int contTemperatura = 0;
                    int contHumedad = 0;
                    int contDioxido = 0;
                    int contHumo = 0;
                    int contRad = 0;
```

```
                int limiteSuperior = listaNodos.size() - 4;
            if (limiteSuperior < 0) {
            limiteSuperior = 0;
                }
            for (int i = listaNodos.size() - 1; i > limiteSuperior; i--) {
            if (listaNodos.get(i).getTemperatura() > 30) {
            contTemperatura++;}}
            for (int i = listaNodos.size() - 1; i > limiteSuperior; i--) {
            if (listaNodos.get(i).getHumedad() < 31) {
            contHumedad++;}}
            for (int i = listaNodos.size() - 1; i > limiteSuperior; i--) {

            if (listaNodos.get(i).getDioxidoDeCarbono() > 999) {
            contDioxido++;}}
            for (int i = listaNodos.size() - 1; i > limiteSuperior; i--) {
            if (listaNodos.get(i).getHumo() == true) {
            contHumo++;}}
            for (int i = listaNodos.size() - 1; i > limiteSuperior; i--) {
            if (listaNodos.get(i).getRadiacion() == true) {
            contRad++;}}
            if (contTemperatura == 3 || contHumedad == 3 || contDioxido == 3 || contHumo == 3 || contRad == 3) {
                    try {
                        itemService.agregarRegistro(temperatura, humedad, dioxido, humo, radiacion, condicion1,
condicion2, condicion3, nodo);
                    } catch (Exception e) {
                        e.getStackTrace();}}}}
                public int monitorTemperatura() {
                    if (temperatura > 30 && temperatura <= 50) {
                        alertaTemperatura = true;
                        emergenciaTemperatura = false;
                        return 1;
                    } else if (temperatura > 50) {
                        alertaTemperatura = false;
                        emergenciaTemperatura = true;
                        return 2;
                    } else {
                        alertaTemperatura = false;
                        emergenciaTemperatura = false;
                        return 0;}}
                public int monitorHumedad() {
                    if (humedad <= 30 && humedad >= 21) {
                        alertaHumedad = true;
                        emergenciaHumedad = false;
                        return 1;
                    } else if (humedad < 21) {
                        alertaHumedad = false;
                        emergenciaHumedad = true;
                        return 2;
                    } else {
                        alertaHumedad = false;
                        emergenciaHumedad = false;
                        return 0;}}
                public int monitorDioxido() {
                    if (dioxido > 1000 && dioxido <= 1500) {
                        alertaDioxido = true;
                        emergenciaDioxido = false;
                        return 1;
```

```
    } else if (dioxido > 1500) {
        alertaDioxido = false;
        emergenciaDioxido = true;
        return 2;
    } else {
        alertaDioxido = false;
        emergenciaDioxido = false;
        return 0; }}
public boolean monitorHumo() {
    if (humo) {
        return true;
    } else {
        return false; }}
public boolean monitorradiacion() {
    if (radiacion) {
        return true;
    } else {
        return false;}}
public int monitorTemperaturaAux() {
    if (temperaturaAux > 30 && temperaturaAux <= 50) {
        alertaTemperaturaAux = true;
        emergenciaTemperaturaAux = false;
        return 1;
    } else if (temperaturaAux > 50) {
        alertaTemperaturaAux = false;
        emergenciaTemperaturaAux = true;
        return 2;
    } else {
        alertaTemperaturaAux = false;
        emergenciaTemperaturaAux = false;
        return 0;}}
public int monitorHumedadAux() {
    if (humedadAux <= 30 && humedadAux >= 21) {
        alertaHumedadAux = true;
        emergenciaHumedadAux = false;
        return 1;
    } else if (humedadAux < 21) {
        alertaHumedadAux = false;
        emergenciaHumedadAux = true;
        return 2;
    } else {
        alertaHumedadAux = false;
        emergenciaHumedadAux = false;
        return 0;}}
public int monitorDioxidoAux() {
    if (dioxidoAux > 1000 && dioxidoAux <= 1500) {
        alertaDioxidoAux = true;
        emergenciaDioxidoAux = false;
        return 1;
    } else if (dioxidoAux > 1500) {
        alertaDioxidoAux = false;
        emergenciaDioxidoAux = true;
        return 2;
    } else {
        alertaDioxidoAux = false;
        emergenciaDioxidoAux = false;
        return 0;}}
```

```
public boolean monitorHumoAux() {
    if (humoAux) {
        return true;
    } else {
        return false;}}
public boolean monitorradiacionAux() {
    if (radiacionAux) {
        return true;
    } else {
        return false;}}
public void actionListenerCargar(Datos dato) {
    idDatos = dato.getIdDato();
    temperatura = dato.getTemperatura();
    humedad = dato.getHumedad();
    dioxido = dato.getDioxidoDeCarbono();
    humo = dato.getHumo();
    radiacion = dato.getRadiacion();
    nodo = dato.getNodo();
    fecha = dato.getFecha();
}
public int getContador() {
    return contador;
}
public void setContador(int contador) {
    this.contador = contador;
}
public boolean isDatosAusentes() {
    return datosAusentes;
}
public void setDatosAusentes(boolean datosAusentes) {
    this.datosAusentes = datosAusentes;
}
public int getIdDatos() {
    return idDatos;
}
public void setIdDatos(int idDatos) {
    this.idDatos = idDatos;
}
public int getTemperatura() {
    return temperatura;
}
public void setTemperatura(int temperatura) {
    this.temperatura = temperatura;
}
public int getHumedad() {
    return humedad;
}
public void setHumedad(int humedad) {
    this.humedad = humedad;
}
public int getDioxido() {
    return dioxido;
}
public void setDioxido(int dioxido) {
    this.dioxido = dioxido;
}
public boolean isHumo() {
```

```
    return humo;
}
public void setHumo(boolean humo) {
    this.humo = humo;
}
public boolean isRadiacion() {
    return radiacion;
}
public void setRadiacion(boolean radiacion) {
    this.radiacion = radiacion;
}
public String getNodo() {
    return nodo;
}
public void setNodo(String nodo) {
    this.nodo = nodo;
}
public Date getFecha() {
    return fecha;
}
public void setFecha(Date fecha) {
    this.fecha = fecha;
}
public List<Datos> getLista() {
    return lista;
}
public void setLista(List<Datos> lista) {
    this.lista = lista;
}
public boolean isCondicion1() {
    return condicion1;
}
public void setCondicion1(boolean condicion1) {
    this.condicion1 = condicion1;
}
public boolean isCondicion2() {
    return condicion2;
}
public void setCondicion2(boolean condicion2) {
    this.condicion2 = condicion2;
}
public boolean isCondicion3() {
    return condicion3;
}
public void setCondicion3(boolean condicion3) {
    this.condicion3 = condicion3;
}
public boolean isAlertaTemperatura() {
    return alertaTemperatura;
}
public void setAlertaTemperatura(boolean alertaTemperatura) {
    this.alertaTemperatura = alertaTemperatura;
}

public boolean isAlertaHumedad() {
    return alertaHumedad;
}
```

```
public void setAlertaHumedad(boolean alertaHumedad) {
   this.alertaHumedad = alertaHumedad;
}
public boolean isAlertaDioxido() {
   return alertaDioxido;
}
public void setAlertaDioxido(boolean alertaDioxido) {
   this.alertaDioxido = alertaDioxido;
}
public boolean isEmergenciaTemperatura() {
   return emergenciaTemperatura;
}
public void setEmergenciaTemperatura(boolean emergenciaTemperatura) {
   this.emergenciaTemperatura = emergenciaTemperatura;
}
public boolean isEmergenciaHumedad() {
   return emergenciaHumedad;
}
public void setEmergenciaHumedad(boolean emergenciaHumedad) {
   this.emergenciaHumedad = emergenciaHumedad;
}
public boolean isEmergenciaDioxido() {
   return emergenciaDioxido;
}
public void setEmergenciaDioxido(boolean emergenciaDioxido) {
   this.emergenciaDioxido = emergenciaDioxido;
}
public int getIdDatosAux() {
   return idDatosAux;
}
public void setIdDatosAux(int idDatosAux) {
   this.idDatosAux = idDatosAux;
}
public int getTemperaturaAux() {
   return temperaturaAux;
}
public void setTemperaturaAux(int temperaturaAux) {
   this.temperaturaAux = temperaturaAux;
}
public int getHumedadAux() {
   return humedadAux;
}
public void setHumedadAux(int humedadAux) {
   this.humedadAux = humedadAux;
}
public int getDioxidoAux() {
   return dioxidoAux;
}
public void setDioxidoAux(int dioxidoAux) {
   this.dioxidoAux = dioxidoAux;
}
public boolean isHumoAux() {
   return humoAux;
}
public void setHumoAux(boolean humoAux) {
   this.humoAux = humoAux;
}
```

```
public boolean isRadiacionAux() {
   return radiacionAux;
}
public void setRadiacionAux(boolean radiacionAux) {
   this.radiacionAux = radiacionAux;
}
public String getNodoAux() {
   return nodoAux;
}
public void setNodoAux(String nodoAux) {
   this.nodoAux = nodoAux;
}
public Date getFechaAux() {
   return fechaAux;
}
public void setFechaAux(Date fechaAux) {
   this.fechaAux = fechaAux;
}
public boolean isCondicion1Aux() {
   return condicion1Aux;
}
public void setCondicion1Aux(boolean condicion1Aux) {
   this.condicion1Aux = condicion1Aux;
}
public boolean isCondicion2Aux() {
   return condicion2Aux;
}
public void setCondicion2Aux(boolean condicion2Aux) {
   this.condicion2Aux = condicion2Aux;
}
public boolean isCondicion3Aux() {
   return condicion3Aux;
}
public void setCondicion3Aux(boolean condicion3Aux) {
   this.condicion3Aux = condicion3Aux;
}
public boolean isAlertaTemperaturaAux() {
   return alertaTemperaturaAux;
}
public void setAlertaTemperaturaAux(boolean alertaTemperaturaAux) {
   this.alertaTemperaturaAux = alertaTemperaturaAux;
}
public boolean isAlertaHumedadAux() {
   return alertaHumedadAux;
}
public void setAlertaHumedadAux(boolean alertaHumedadAux) {
   this.alertaHumedadAux = alertaHumedadAux;
}
public boolean isAlertaDioxidoAux() {
   return alertaDioxidoAux;
}
public void setAlertaDioxidoAux(boolean alertaDioxidoAux) {
   this.alertaDioxidoAux = alertaDioxidoAux;
}
public boolean isEmergenciaTemperaturaAux() {
   return emergenciaTemperaturaAux;
}
```

```
    public void setEmergenciaTemperaturaAux(boolean emergenciaTemperaturaAux) {
        this.emergenciaTemperaturaAux = emergenciaTemperaturaAux;
    }
    public boolean isEmergenciaHumedadAux() {
        return emergenciaHumedadAux;
    }
    public void setEmergenciaHumedadAux(boolean emergenciaHumedadAux) {
        this.emergenciaHumedadAux = emergenciaHumedadAux;
    }
    public boolean isEmergenciaDioxidoAux() {
        return emergenciaDioxidoAux;
    }
    public void setEmergenciaDioxidoAux(boolean emergenciaDioxidoAux) {
        this.emergenciaDioxidoAux = emergenciaDioxidoAux;
    }

    public List<String> getNodos() {
        return nodos;
    }

    public void setNodos(List<String> nodos) {
        this.nodos = nodos;
    }
    private String message;
    public String getMessage() {
        return message;
    }
    public void setMessage(String message) {
        this.message = message;
    }
    public void saveMessage() {
        RequestContext request = RequestContext.getCurrentInstance();
        if (listaDatosAusentes == null) {
            listaDatosAusentes = new ArrayList<>();
        }
        if (!listaDatosAusentes.isEmpty()) {
            for (int i = 0; i < listaDatosAusentes.size(); i++) {
                if (listaDatosAusentes.get(i)) {
    FacesContext.getCurrentInstance().addMessage(
                        null,
                        new FacesMessage(FacesMessage.SEVERITY_FATAL,
                            "¡EMERGENCIA!",
                            "No se han recibido datos en más de 1 minuto y 30 segundos del nodo " + nodos.get(i)));
                }}}
        if (emergenciaDioxido || emergenciaHumedad || emergenciaTemperatura || humo || radiacion || condicion1
|| condicion2 || condicion3) {
            request.execute("playAudio()");
        } else {
          request.execute("pausar()");
        }
        if (alertaDioxido) {
      FacesContext.getCurrentInstance().addMessage(
                null,
                new FacesMessage(FacesMessage.SEVERITY_WARN,
                    "¡ALERTA!",
                    "Niveles elevados de dióxido de carbono"));
        }
```

```
        if (alertaTemperatura) {
FacesContext.getCurrentInstance().addMessage(
                null,
                new FacesMessage(FacesMessage.SEVERITY_WARN,
                    "¡ALERTA!",
                    "Temperatura elevada"));
        }
        if (alertaHumedad) {
FacesContext.getCurrentInstance().addMessage(
                null,
                new FacesMessage(FacesMessage.SEVERITY_WARN,
                    "¡ALERTA!",
                    "Nivel de humedad bajo"));
        }
        if (emergenciaTemperatura) {
FacesContext.getCurrentInstance().addMessage(
                null,
                new FacesMessage(FacesMessage.SEVERITY_FATAL,
                    "¡EMERGENCIA!",
                    "Temperatura peligrosamente alta"));
        }
        if (emergenciaHumedad) {
FacesContext.getCurrentInstance().addMessage(
                null,
                new FacesMessage(FacesMessage.SEVERITY_FATAL,
                    "¡EMERGENCIA!",
                    "Humedad peligrosamente baja"));
        }
        if (emergenciaDioxido) {
FacesContext.getCurrentInstance().addMessage(
                null,
                new FacesMessage(FacesMessage.SEVERITY_FATAL,
                    "¡EMERGENCIA!",
                    "Niveles de dióxido de carbono peligrosamente altos"));
        }
        if (humo) {
FacesContext.getCurrentInstance().addMessage(
                null,
                new FacesMessage(FacesMessage.SEVERITY_FATAL,
                    "¡EMERGENCIA!",
                    "Presencia de humo"));
        }
        if (radiacion) {
FacesContext.getCurrentInstance().addMessage(
                null,
                new FacesMessage(FacesMessage.SEVERITY_FATAL,
                    "¡EMERGENCIA!",
                    "Presencia de radiación"));
        }
        if (condicion1) {
FacesContext.getCurrentInstance().addMessage(
                null,
                new FacesMessage(FacesMessage.SEVERITY_FATAL,
                    "¡EMERGENCIA!",
                    "Condición 1"));
        }
        if (condicion2) {
```

```
            FacesContext.getCurrentInstance().addMessage(
                        null,
                        new FacesMessage(FacesMessage.SEVERITY_FATAL,
                            "¡EMERGENCIA!",
                            "Condición 2"));
                }
                if (condicion3) {
            FacesContext.getCurrentInstance().addMessage(
                        null,
                        new FacesMessage(FacesMessage.SEVERITY_FATAL,
                            "¡EMERGENCIA!",
                            "Condición 3"));
                }
                System.out.println("Mostrando mensajes");
            }
            public List<String> cargarnodos() {
                return itemService.obtenerNodos();
            }
            public void verificar() {
                System.out.println("verificando");
                while (nodos.size() > idActuales.size()) {
                    System.out.println("igualar");
                    idActuales.add(0);
                    contadores.add(0);
                    listaDatosAusentes.add(false);
                }
                for (int i = 0; i < idActuales.size(); i++) {
                    List<Datos> listaux = itemService.findNodosByNodo(nodos.get(i));
                    int aux = idActuales.get(i);
                    idActuales.set(i, listaux.get(listaux.size() - 1).getIdDato());
                    if (aux == idActuales.get(i)) {
                        contadores.set(i, contadores.get(i) + 5);
                    } else {
                        contadores.set(i, 0);
                        listaDatosAusentes.set(i, false);
                    }
                    if (contadores.get(i) >= 90) {
                        listaDatosAusentes.set(i, true);
                    }}}
            public boolean filterByDate(Object value, Object filter, Locale locale) {
                Date a = (Date) filter;
                Date b = (Date) value;
                if (a == null || a.equals("")) {
                    return true;
                }
                if (value == null) {
                    return false;
                }
                boolean result;
                if (a.getMonth() == b.getMonth() && a.getYear() == b.getYear() && a.getDay() == b.getDay()) {
                    if (a.getHours() > 0) {
                        if (a.getMinutes() > 0) {
                            if (a.getSeconds() > 0) {
                                result = a.getHours() == b.getHours() && a.getMinutes() == b.getMinutes() && a.getSeconds()
== b.getSeconds();
                            } else {
                                result = a.getHours() == b.getHours() && a.getMinutes() == b.getMinutes();
```

```
                }
            } else {
                result = a.getHours() == b.getHours();
            }
        } else {
            return true;
        }
    } else {
        result = false;
    }
    return result;
}}
```

Anexo 6. Conexión de base de datos.

dataConnect

```
package jsf.util;
import java.sql.Connection;
import java.sql.DriverManager;
public class DataConnect {
public static Connection getConnection() {
try {
Class.forName("com.mysql.jdbc.Driver");
Connection con = DriverManager.getConnection(
"jdbc:mysql://localhost:3306/sensores", "root", "");
return con;
} catch (Exception ex) {
System.out.println("Database.getConnection() Error -->"
+ ex.getMessage());
return null;
}}
public static void close(Connection con) {
try {
con.close();
} catch (Exception ex) {
}}}
```

Anexo 7. Verificación de usuario y contraseña.

Login

```
package jsf.beans;
import java.io.Serializable;
import javax.faces.application.FacesMessage;
import javax.faces.bean.ManagedBean;
import javax.faces.bean.SessionScoped;
import javax.faces.context.FacesContext;
import javax.servlet.http.HttpSession;
```

```
import jsf.dao.LoginDAO;
import jsf.beans.SessionUtils;
@ManagedBean
@SessionScoped
public class Login implements Serializable {
private static final long serialVersionUID = 1094801825228386363L;
private String pwd;
private String msg;
private String user;
	private int contador;

public String getPwd() {
return pwd;
}
	public int getContador() {
return contador;
}
public void setPwd(String pwd) {
this.pwd = pwd;
}
public String getMsg() {
return msg;
}
public void setMsg(String msg) {
this.msg = msg;
}
public String getUser() {
return user;
}
public void setUser(String user) {
this.user = user;
}
//validate login

public String validateUsernamePassword() {
boolean valid = LoginDAO.validate(user, pwd);
if (valid) {
HttpSession session = SessionUtils.getSession();
session.setAttribute("username", user);
return "index";
} else {
		System.out.println("Error de autenticacion");
FacesContext.getCurrentInstance().addMessage(
null,
new FacesMessage(FacesMessage.SEVERITY_WARN,
"Incorrect Username and Passowrd",
"Please enter correct username and Password"));
return "login";
}}
//logout event, invalidate session
public String logout() {
HttpSession session = SessionUtils.getSession();
session.invalidate();
	System.out.println("Desconectado");
return "login";
}}
```

Anexo 8. Clase controladora de sesiones.

SessionUtils

```
package jsf.beans;

import javax.faces.context.FacesContext;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpSession;

public class SessionUtils {
public static HttpSession getSession() {
return (HttpSession) FacesContext.getCurrentInstance()
.getExternalContext().getSession(false);
}
public static HttpServletRequest getRequest() {
return (HttpServletRequest) FacesContext.getCurrentInstance()
.getExternalContext().getRequest();
}
public static String getUserName() {
HttpSession session = (HttpSession) FacesContext.getCurrentInstance()
.getExternalContext().getSession(false);
return session.getAttribute("username").toString();
}
public static String getUserId() {
HttpSession session = getSession();
if (session != null)
return (String) session.getAttribute("userid");
else
return null;
}}
```

Anexo 9. Clase controladora de alertas.

alertasController

```
/*
 * To change this license header, choose License Headers in Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */
package jsf.beans;
import java.util.Date;
import java.util.List;
import java.util.Locale;
import javax.annotation.PostConstruct;
import javax.ejb.EJB;
import javax.faces.bean.ManagedBean;
import javax.faces.bean.RequestScoped;
```

```
import sensores.model.entities.Alertas;
import sensores.model.manager.ManagerDatos;
/**
 *
 * @author jruiz
 */
@ManagedBean
@RequestScoped
public class alertasController {
    /**
     * Creates a new instance of alertasController
     */
    private int idDatos = 0;
    private int temperatura;
    private int humedad;
    private int dioxido;
    private boolean humo;
    private boolean radiacion;
    private String nodo;
    private Date fecha;

    private boolean condicion1;
    private boolean condicion2;
    private boolean condicion3;

    private List<Alertas> lista;
    private List<Alertas> listafiltrada;
    private List<String> nodos;

    @EJB
    private ManagerDatos manager;

    @PostConstruct
    public void init() {
        lista = manager.findAlertas();
        nodos=cargarnodos();
    }
    public void actionListenerCargar(Alertas alerta) {
        idDatos = alerta.getIdAlerta();
        temperatura = alerta.getTemperatura();
        humedad = alerta.getHumedad();
        dioxido = alerta.getDioxidoDeCarbono();
        humo = alerta.getHumo();
        radiacion = alerta.getRadiacion();
        nodo = alerta.getNodo();
        fecha = alerta.getFecha();
        condicion1=alerta.getCondicion1();
        condicion2=alerta.getCondicion2();
        condicion3=alerta.getCondicion3();
    }

    public List<Alertas> getListafiltrada() {
        return listafiltrada;
    }
    public List<String> cargarnodos(){
    return manager.obtenerNodos();
    }
```

```
public void setListafiltrada(List<Alertas> listafiltrada) {
    this.listafiltrada = listafiltrada;
}
public List<String> getNodos() {
    return nodos;
}
public void setNodos(List<String> nodos) {
    this.nodos = nodos;
}
public int getIdDatos() {
    return idDatos;
}
public void setIdDatos(int idDatos) {
    this.idDatos = idDatos;
}
public int getTemperatura() {
    return temperatura;
}
public void setTemperatura(int temperatura) {
    this.temperatura = temperatura;
}
public int getHumedad() {
    return humedad;
}
public void setHumedad(int humedad) {
    this.humedad = humedad;
}
public int getDioxido() {
    return dioxido;
}
public void setDioxido(int dioxido) {
    this.dioxido = dioxido;
}
public boolean isHumo() {
    return humo;
}
public void setHumo(boolean humo) {
    this.humo = humo;
}
public boolean isRadiacion() {
    return radiacion;
}
public void setRadiacion(boolean radiacion) {
    this.radiacion = radiacion;
}
public String getNodo() {
    return nodo;
}
public void setNodo(String nodo) {
    this.nodo = nodo;
}
public Date getFecha() {
    return fecha;
}
public void setFecha(Date fecha) {
    this.fecha = fecha;
}
```

```
    public boolean isCondicion1() {
        return condicion1;
    }
    public void setCondicion1(boolean condicion1) {
        this.condicion1 = condicion1;
    }
    public boolean isCondicion2() {
        return condicion2;
    }
    public void setCondicion2(boolean condicion2) {
        this.condicion2 = condicion2;
    }
    public boolean isCondicion3() {
        return condicion3;
    }
    public void setCondicion3(boolean condicion3) {
        this.condicion3 = condicion3;
    }
    public List<Alertas> getLista() {
        return lista;
    }
    public void setLista(List<Alertas> lista) {
        this.lista = lista;
    }
    public boolean filterByDate(Object value, Object filter, Locale locale) {
        Date a = (Date) filter;
        Date b = (Date) value;
        if (a == null || a.equals("")) {
            return true;
        }
        if (value == null) {
            return false;
        }
        boolean result;

        if (a.getMonth() == b.getMonth() && a.getYear() == b.getYear() && a.getDay() == b.getDay()) {
            if (a.getHours() > 0) {
                if (a.getMinutes() > 0) {
                    if (a.getSeconds() > 0) {
                        result = a.getHours() == b.getHours() && a.getMinutes() == b.getMinutes() && a.getSeconds()
== b.getSeconds();
                    } else {
                        result = a.getHours() == b.getHours() && a.getMinutes() == b.getMinutes();
                    }
                } else {
                    result = a.getHours() == b.getHours();
                }
            }else return true;
        } else {
            result = false;
        }
        return result;
    }}
```

Anexo 10. Clase de tabla y búsqueda.

chartView

```
package jsf.beans;
import java.io.Serializable;
import java.util.List;
import javax.annotation.PostConstruct;
import javax.ejb.EJB;
import javax.faces.application.FacesMessage;
import javax.faces.bean.ManagedBean;
import javax.faces.context.FacesContext;

import org.primefaces.model.chart.Axis;
import org.primefaces.model.chart.AxisType;
import org.primefaces.model.chart.BarChartModel;
import org.primefaces.model.chart.ChartSeries;
import org.primefaces.model.chart.HorizontalBarChartModel;
import sensores.model.entities.Datos;
import sensores.model.manager.ManagerDatos;

@ManagedBean
public class ChartView implements Serializable {

    private HorizontalBarChartModel barModel;
    private HorizontalBarChartModel humedadBarModel;
    private HorizontalBarChartModel dioxidoBarModel;
    private HorizontalBarChartModel humoBarModel;
    private HorizontalBarChartModel radiacionBarModel;
    private List<Datos> lista;
    private List<Datos> listaPorNodo;
    private List<String> nodos;
    private String nodo;
    private int cantidadAnalizable;
    private int cantidadEnLista;
    private int inicio;

    @EJB
    private ManagerDatos managerDatos;

    @PostConstruct
    public void init() {
        lista = managerDatos.findDatos();
        cantidadEnLista = lista.size();
        cantidadAnalizable = 5;
        nodo = "todos";
        ObtenerNodos();
        createBarModels(0);
    }
    public void ObtenerNodos() {
        nodos = managerDatos.obtenerNodos();
    }
    public void ObtenerNodosEspecificos(String nodoAct,int inicioActual) {
        if (nodoAct.equals("todos")) {
            listaPorNodo = lista;
```

```
            inicio = listaPorNodo.size();
        } else {
            listaPorNodo = managerDatos.findNodosByNodo(nodoAct);
            inicio = listaPorNodo.size();
            cantidadEnLista = listaPorNodo.size();
        }
        if (inicioActual<=0||inicioActual>listaPorNodo.size()) {
        }else inicio=inicioActual;
    }
    public int getInicio() {
        return inicio;
    }
    public void setInicio(int inicio) {
        this.inicio = inicio;
    }
    public int getCantidadEnLista() {
        return cantidadEnLista;
    }
    public void setCantidadEnLista(int cantidadEnLista) {
        this.cantidadEnLista = cantidadEnLista;
    }
    public int getCantidadAnalizable() {
        return cantidadAnalizable;
    }
    public void setCantidadAnalizable(int cantidadAnalizable) {
        this.cantidadAnalizable = cantidadAnalizable;
    }
    public String getNodo() {
        return nodo;
    }
    public void setNodo(String nodo) {
        this.nodo = nodo;
    }
    public List<String> getNodos() {
        return nodos;
    }
    public BarChartModel getBarModel() {
        return barModel;
    }
    public BarChartModel getHumedadBarModel() {
        return humedadBarModel;
    }
    public BarChartModel getDioxidoBarModel() {
        return dioxidoBarModel;
    }
    public BarChartModel getHumoBarModel() {
        return humoBarModel;
    }
    public BarChartModel getRadiacionBarModel() {
        return radiacionBarModel;
    }
    private String formatearDato(int val) {
        String dato = "";
        if (val < 10) {
            dato = "0" + val;
        } else {
            dato = val + "";
```

```
            }
            return dato;
        }
        private String formatearDioxido(int val) {
            String dato = "";
            if (val < 100) {
                dato = "00" + val;
            } else if (val < 1000) {
                dato = "0" + val;
            } else {
                dato = val + "";
            }
            return dato;
        }

        public void actualizar() {
            System.out.println("Cantidad actual: " + cantidadAnalizable);
        }

        private HorizontalBarChartModel initBarModel() {
            int limiteInferior;
            HorizontalBarChartModel model = new HorizontalBarChartModel();

            ChartSeries temperatura = new ChartSeries();
            temperatura.setLabel("Temperatura");

            if (inicio >= cantidadAnalizable) {
                limiteInferior = inicio - cantidadAnalizable;
            } else {
                limiteInferior = 0;
            }
            for (int i = limiteInferior; i < inicio; i++) {

                temperatura.set("ID: " + listaPorNodo.get(i).getIdDato() + " " +
(listaPorNodo.get(i).getFecha().getYear() + 1900) + "-" + formatearDato(listaPorNodo.get(i).getFecha().getMonth()) +
"-" + formatearDato(listaPorNodo.get(i).getFecha().getDate()) + " " +
formatearDato(listaPorNodo.get(i).getFecha().getHours()) + ":" +
formatearDato(listaPorNodo.get(i).getFecha().getMinutes()) + ":" +
formatearDato(listaPorNodo.get(i).getFecha().getSeconds()) + " Dato: " +
formatearDato(listaPorNodo.get(i).getTemperatura()) + "ºC" + " Nodo: " + listaPorNodo.get(i).getNodo(),
listaPorNodo.get(i).getTemperatura());
            }
            model.addSeries(temperatura);
            return model;
        }
        private HorizontalBarChartModel initHumedadBarModel() {
            int limiteInferior;
            HorizontalBarChartModel model = new HorizontalBarChartModel();
            ChartSeries humedad = new ChartSeries();
            humedad.setLabel("Humedad");
            if (inicio >= cantidadAnalizable) {
                limiteInferior = inicio - cantidadAnalizable;
            } else {
                limiteInferior = 0;
            }
            for (int i = limiteInferior; i < inicio; i++) {
```

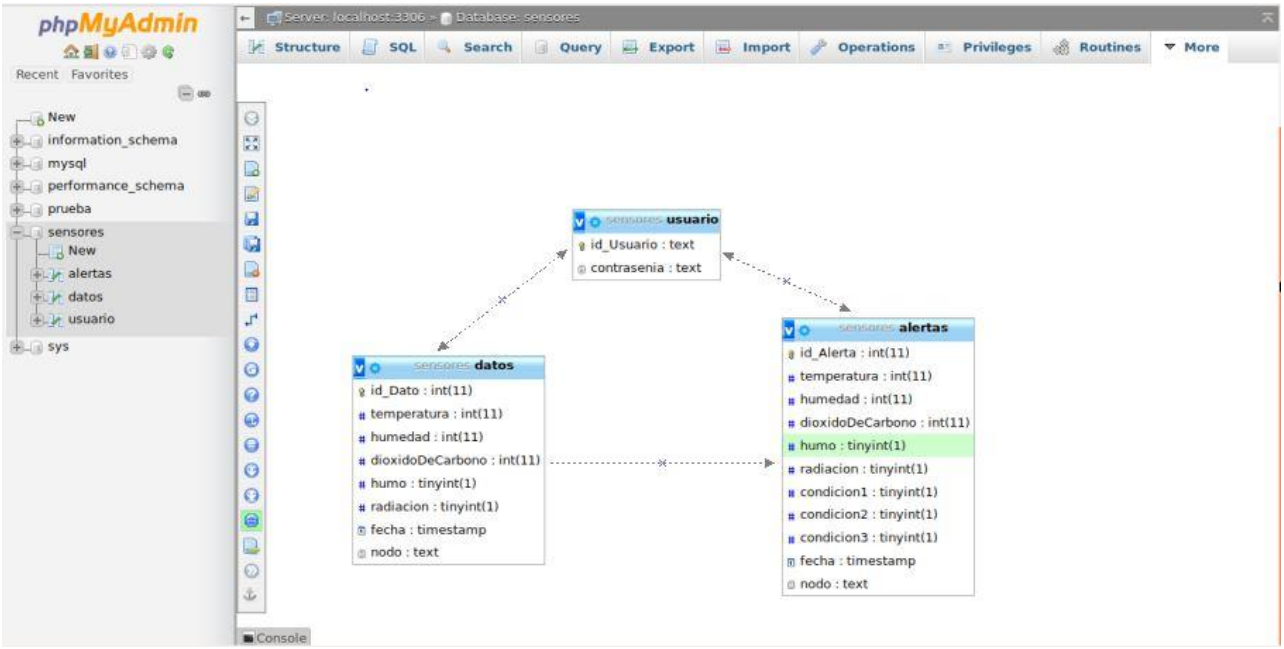
```
                humedad.set("ID: " + listaPorNodo.get(i).getIdDato() + " " + (listaPorNodo.get(i).getFecha().getYear()
+ 1900) + "-" + formatearDato(listaPorNodo.get(i).getFecha().getMonth()) + "-" +
formatearDato(listaPorNodo.get(i).getFecha().getDate()) + " " +
formatearDato(listaPorNodo.get(i).getFecha().getHours()) + ":" +
formatearDato(listaPorNodo.get(i).getFecha().getMinutes()) + ":" +
formatearDato(listaPorNodo.get(i).getFecha().getSeconds()) + " Dato: " +
formatearDato(listaPorNodo.get(i).getHumedad()) + "%" + " Nodo: " + listaPorNodo.get(i).getNodo(),
listaPorNodo.get(i).getHumedad());
                }
                model.addSeries(humedad);
                return model;
            }
            private HorizontalBarChartModel initDioxidoBarModel() {
                int limiteInferior;
                HorizontalBarChartModel model = new HorizontalBarChartModel();
                ChartSeries dioxido = new ChartSeries();
                dioxido.setLabel("Dióxido de carbono");
                if (inicio >= cantidadAnalizable) {
                    limiteInferior = inicio - cantidadAnalizable;
                } else {
                    limiteInferior = 0;
                }
                for (int i = limiteInferior; i < inicio; i++) {
                    dioxido.set("ID: " + listaPorNodo.get(i).getIdDato() + " " + (listaPorNodo.get(i).getFecha().getYear()
+ 1900) + "-" + formatearDato(listaPorNodo.get(i).getFecha().getMonth()) + "-" +
formatearDato(listaPorNodo.get(i).getFecha().getDate()) + " " +
formatearDato(listaPorNodo.get(i).getFecha().getHours()) + ":" +
formatearDato(listaPorNodo.get(i).getFecha().getMinutes()) + ":" +
formatearDato(listaPorNodo.get(i).getFecha().getSeconds()) + " Dato: " +
formatearDioxido(listaPorNodo.get(i).getDioxidoDeCarbono()) + "PPM" + " Nodo: " + listaPorNodo.get(i).getNodo(),
listaPorNodo.get(i).getDioxidoDeCarbono());
                }
                model.addSeries(dioxido);
                return model;
            }
            private HorizontalBarChartModel initHumoBarModel() {
                int limiteInferior;
                HorizontalBarChartModel model = new HorizontalBarChartModel();
                ChartSeries humo = new ChartSeries();
                humo.setLabel("Humo");

                if (inicio >= cantidadAnalizable) {
                    limiteInferior = inicio - cantidadAnalizable;
                } else {
                    limiteInferior = 0;
                }
                for (int i = limiteInferior; i < inicio; i++) {
                    int val = listaPorNodo.get(i).getHumo() ? 1 : 0;
                    humo.set("ID: " + listaPorNodo.get(i).getIdDato() + " " + (listaPorNodo.get(i).getFecha().getYear() +
1900) + "-" + formatearDato(listaPorNodo.get(i).getFecha().getMonth()) + "-" +
formatearDato(listaPorNodo.get(i).getFecha().getDate()) + " " +
formatearDato(listaPorNodo.get(i).getFecha().getHours()) + ":" +
formatearDato(listaPorNodo.get(i).getFecha().getMinutes()) + ":" +
formatearDato(listaPorNodo.get(i).getFecha().getSeconds()) + " Dato: " + val + " Nodo: " +
listaPorNodo.get(i).getNodo(), val);
                }
                model.addSeries(humo);
```

```
        return model;
    }
    private HorizontalBarChartModel initRadiacionBarModel() {
        int limiteInferior;
        HorizontalBarChartModel model = new HorizontalBarChartModel();
        ChartSeries radiacion = new ChartSeries();
        radiacion.setLabel("Radiación");

        if (inicio >= cantidadAnalizable) {
            limiteInferior = inicio - cantidadAnalizable;
        } else {
            limiteInferior = 0;
        }
        for (int i = limiteInferior; i < inicio; i++) {
            int val = listaPorNodo.get(i).getRadiacion() ? 1 : 0;
            radiacion.set("ID: " + listaPorNodo.get(i).getIdDato() + " " + (listaPorNodo.get(i).getFecha().getYear()
+ 1900) + "-" + formatearDato(listaPorNodo.get(i).getFecha().getMonth()) + "-" +
formatearDato(listaPorNodo.get(i).getFecha().getDate()) + " " +
formatearDato(listaPorNodo.get(i).getFecha().getHours()) + ":" +
formatearDato(listaPorNodo.get(i).getFecha().getMinutes()) + ":" +
formatearDato(listaPorNodo.get(i).getFecha().getSeconds()) + " Dato: " + val + " Nodo: " +
listaPorNodo.get(i).getNodo(), val);
        }
        model.addSeries(radiacion);
        return model;
    }
    private void createBarModels(int inicioActual) {
        ObtenerNodosEspecificos(nodo,inicioActual);
        createBarModel();
        createHumedadBarModel();
        createDioxidoBarModel();
        createHumoBarModel();
        createRadiacionBarModel();
    }

    private void createBarModel() {
        barModel = initBarModel();

        barModel.setTitle("Temperatura");
        barModel.setLegendPosition("e");
        barModel.setSeriesColors("58BA27,FFCC33,F74A4A,F52F2F,A30303");

        Axis xAxis = barModel.getAxis(AxisType.X);
        xAxis.setLabel("Grados Centígrados");
        xAxis.setMin(0);
        xAxis.setMax(100);

        Axis yAxis = barModel.getAxis(AxisType.Y);
        yAxis.setLabel("Datos");

    }

    private void createHumedadBarModel() {
        humedadBarModel = initHumedadBarModel();

        humedadBarModel.setTitle("Humedad");
        humedadBarModel.setLegendPosition("ne");
```

```
    Axis yAxis = humedadBarModel.getAxis(AxisType.Y);
    yAxis.setLabel("Datos");

    Axis xAxis = humedadBarModel.getAxis(AxisType.X);
    xAxis.setLabel("Porcentaje");
    xAxis.setMin(0);
    xAxis.setMax(100);
}

private void createDioxidoBarModel() {
    dioxidoBarModel = initDioxidoBarModel();
    dioxidoBarModel.setTitle("Dióxido de carbono");
    dioxidoBarModel.setLegendPosition("ne");
    dioxidoBarModel.setSeriesColors("d87a22,d87a22,d87a22,d87a22,d87a22");

    Axis yAxis = dioxidoBarModel.getAxis(AxisType.Y);
    yAxis.setLabel("Datos");

    Axis xAxis = dioxidoBarModel.getAxis(AxisType.X);
    xAxis.setLabel("Partículas por millón");
    xAxis.setMin(0);
    xAxis.setMax(2000);
}
private void createHumoBarModel() {
    humoBarModel = initHumoBarModel();

    humoBarModel.setTitle("Humo");
    humoBarModel.setLegendPosition("ne");
    humoBarModel.setSeriesColors("a36bc6,a36bc6,a36bc6,a36bc6,a36bc6");

    Axis yAxis = humoBarModel.getAxis(AxisType.Y);
    yAxis.setLabel("Datos");

    Axis xAxis = humoBarModel.getAxis(AxisType.X);
    xAxis.setLabel("Presencia de humo");
    xAxis.setMin(0);
    xAxis.setMax(1);
}

private void createRadiacionBarModel() {
    radiacionBarModel = initRadiacionBarModel();

    radiacionBarModel.setTitle("Radiación");
    radiacionBarModel.setLegendPosition("ne");
    radiacionBarModel.setSeriesColors("c1666e,c1666e,c1666e,c1666e,c1666e");

    Axis yAxis = radiacionBarModel.getAxis(AxisType.Y);
    yAxis.setLabel("Datos");

    Axis xAxis = radiacionBarModel.getAxis(AxisType.X);
    xAxis.setLabel("Presencia de radiación");
    xAxis.setMin(0);
    xAxis.setMax(1);
}

public void onNodoChange(String valorAct,String inicioAct) {
```

```
        try {
            int inicioNuevo;
            int valor;
            if (valorAct.isEmpty()) {
                valor = 5;
            } else {
                valor = Integer.parseInt(valorAct);
            }
            if (inicioAct.isEmpty()) {
                inicioNuevo=0;
            } else {
                inicioNuevo = Integer.parseInt(inicioAct);
            }
            if (valor < 0) {
                valor = 1;
            } else if (valor > 15) {
                valor = 15;
            }

            cantidadAnalizable = valor;
            createBarModels(inicioNuevo);
        } catch (Exception e) {
            e.printStackTrace();
            saveMessage();
        }}

    public void saveMessage() {
        FacesContext.getCurrentInstance().addMessage(
                null,
                new FacesMessage(FacesMessage.SEVERITY_WARN,
                        "Formato incorrecto",
                        "Por favor ingrese solo números"));
    }
}
```

Anexo 11. Entidad – Relación.



El modelo entidad relacion muestra las entidades que describen los objetos mostrados los campos involucrados y su respectiva ubicacion dentro de la base de datos Mysql.