UTMACH

UNIDAD ACADÉMICA DE INGENIERÍA CIVIL

CARRERA DE INGENIERÍA DE SISTEMAS

IMPLEMENTACIÓN DE HERRAMIENTAS DE GESTIÓN DE LOGS, INTEGRACIÓN DE DATOS Y SEGURIDAD PARA EL SISTEMA IOTMACH

VALAREZO PAZ KEVIN ADRIAN

MACHALA
2016

UTMACH

UNIDAD ACADÉMICA DE INGENIERÍA CIVIL

CARRERA DE INGENIERÍA DE SISTEMAS

IMPLEMENTACIÓN DE HERRAMIENTAS DE GESTIÓN DE LOGS, INTEGRACIÓN DE DATOS Y SEGURIDAD PARA EL SISTEMA IOTMACH

VALAREZO PAZ KEVIN ADRIAN

MACHALA
2016

UTMACH

UNIDAD ACADÉMICA DE INGENIERÍA CIVIL

CARRERA DE INGENIERÍA DE SISTEMAS

TRABAJO DE TITULACIÓN
PROPUESTAS TECNOLÓGICAS

IMPLEMENTACIÓN DE HERRAMIENTAS DE GESTIÓN DE LOGS, INTEGRACIÓN DE DATOS Y SEGURIDAD PARA EL SISTEMA IOTMACH

VALAREZO PAZ KEVIN ADRIAN
INGENIERO DE SISTEMAS

MAZÓN OLIVO BERTHA EUGENIA

Machala, 18 de octubre de 2016

MACHALA
2016

Nota de aceptación:

Quienes suscriben MAZÓN OLIVO BERTHA EUGENIA, ZEA ORDOÑÉZ MARIUXI PAOLA, REDROVAN CASTILLO FAUSTO FABIAN y JUMBO CASTILLO FREDDY ANIBAL, en nuestra condición de evaluadores del trabajo de titulación denominado IMPLEMENTACIÓN DE HERRAMIENTAS DE GESTIÓN DE LOGS, INTEGRACIÓN DE DATOS Y SEGURIDAD PARA EL SISTEMA IOTMACH, hacemos constar que luego de haber revisado el manuscrito del precitado trabajo, consideramos que reúne las condiciones académicas para continuar con la fase de evaluación correspondiente.

MAZÓN OLIVO BERTHA EUGENIA
0603100512
TUTOR

ZEA ORDOÑÉZ MARIUXI PAOLA
0702801598
ESPECIALISTA 1

REDROVAN CASTILLO FAUSTO FABIAN
0702739228
ESPECIALISTA 2

JUMBO CASTILLO FREDDY ANIBAL
0704167949
ESPECIALISTA 3

Machala, 18 de octubre de 2016

URKUND

Urkund Analysis Result

Analysed Document: VALAREZO PAZ KEVIN ADRIAN.docx (D21636932)
Submitted: 2016-09-07 03:31:00
Submitted By: nivek.adrian123@gmail.com
Significance: 1 %

Sources included in the report:

Tesiscompleta_marloncarangui_3junio.docx (D14715391)
Tesiscompleta_marloncarangui_3junio.docx (D14742358)
Sitio Web para la gestion de citas médicas.docx (D14064997)
Sitio Web para la gestion de citas médicas.docx (D13867341)
Tesis Presentacion.docx (D15483527)
Informe Caso de Estudio Roberto Maldonado.pdf (D14337051)

Instances where selected sources appear:

10

CLÁUSULA DE CESIÓN DE DERECHO DE PUBLICACIÓN EN EL REPOSITORIO DIGITAL INSTITUCIONAL

El que suscribe, VALAREZO PAZ KEVIN ADRIAN, en calidad de autor del siguiente trabajo escrito titulado IMPLEMENTACIÓN DE HERRAMIENTAS DE GESTIÓN DE LOGS, INTEGRACIÓN DE DATOS Y SEGURIDAD PARA EL SISTEMA IOTMACH, otorga a la Universidad Técnica de Machala, de forma gratuita y no exclusiva, los derechos de reproducción, distribución y comunicación pública de la obra, que constituye un trabajo de autoría propia, sobre la cual tiene potestad para otorgar los derechos contenidos en esta licencia.

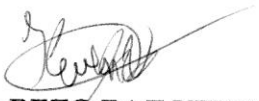
El autor declara que el contenido que se publicará es de carácter académico y se enmarca en las dispociones definidas por la Universidad Técnica de Machala.

Se autoriza a transformar la obra, únicamente cuando sea necesario, y a realizar las adaptaciones pertinentes para permitir su preservación, distribución y publicación en el Repositorio Digital Institucional de la Universidad Técnica de Machala.

El autor como garante de la autoría de la obra y en relación a la misma, declara que la universidad se encuentra libre de todo tipo de responsabilidad sobre el contenido de la obra y que él asume la responsabilidad frente a cualquier reclamo o demanda por parte de terceros de manera exclusiva.

Aceptando esta licencia, se cede a la Universidad Técnica de Machala el derecho exclusivo de archivar, reproducir, convertir, comunicar y/o distribuir la obra mundialmente en formato electrónico y digital a través de su Repositorio Digital Institucional, siempre y cuando no se lo haga para obtener beneficio económico.

Machala, 18 de octubre de 2016



VALAREZO PAZ KEVIN ADRIAN
0704640333

DEDICATORIA

El presente trabajo va dedicado principalmente a mi madre, ya que gracias a su preocupación y perseverancia he llegado donde estoy, impulsándome a seguir adelante para alcanzar los objetivos que me he planteado.

A mis hermanos, que a pesar de todo siempre me brindan su mayor apoyo y por darme la mano en los momentos difíciles vividos hasta la presente fecha.

También el presente va dedicado a mi pequeño sobrino, quien a pesar de su corta edad ha sido pilar fundamental para llegar a cumplir la meta planteada al iniciar mi carrera universitaria.

Para culminar, esto va dedicado también para mis amigos, los pocos con los que cuento, quienes siempre brindan apoyo en los momentos difíciles.

Kevin Adrian

AGRADECIMIENTO

Quiero empezar dándole gracias a Dios, por mantenerme con vida.

A todos los docentes que me han impartido sus cátedras durante esta carrera universitaria, ya que gracias a ellos que me han sabido transmitir sus conocimientos en varias temáticas sobre la carrera.

Un agradecimiento especial es para la Ing. Bertha Mazón y para el Ing. Dixys Hernández quienes además de ser los tutores del presente trabajo, me brindaron la oportunidad de participar en el grupo de investigación denominado IOTMACH.

A la Universidad Técnica de Machala, a la Unidad Académica de Ingeniería Civil, a la Carrera de Ingeniería de Sistemas por haberme acogido durante los últimos años y hacerme crecer tanto personal como profesionalmente.

Kevin Adrian

RESUMEN

El internet de las cosas, mejor conocido como IOT por sus siglas en inglés Internet Of Things, en la actualidad está convirtiéndose en un término común dentro de nuestra localidad, todos los sistemas ya creados y aquellos que recién se están gestando están implementando la ideología del IOT, la misma que está orientada en comunicar dispositivos comunes entre sí a través del internet, por tal motivo el sistema IOTMACH está enfocado en ser uno de los pioneros en implementar una infraestructura que permita alojar aplicaciones IOT conjuntamente con redes de sensores inalámbricos para aplicarlos a la automatización de procesos de varios campos tales como la agricultura, pesca, ganadería, minería entre otros.

Una vez mencionado esto, el presente trabajo de titulación se basa en la implementación de herramientas de gestión de Logs, integración de datos y seguridad para el sistema IOTMACH, en el cual se necesita de la gestión de grandes volúmenes de datos a altas velocidades, la manipulación de múltiples fuentes de información y, controlar el acceso de usuarios hacia la información que maneja el sistema. Por tales motivos se requiere de la implementación de herramientas que gestionen los datos, que brinden la integración de las fuentes donde se tiene almacenada la información y un manejo de los usuarios que tendrán acceso al sistema a través de la asignación de roles con diferentes privilegios.

En cuanto a la organización del proyecto, se empleó la metodología ágil Kanban, en la que no es necesario establecer un tiempo fijo ni reuniones de planificación, enfatizando el cumplimiento de tareas a medida que se vaya desarrollando el proyecto, es decir se van tomando tareas a medida que se vayan cumpliendo tareas previas, permitiendo realizar cualquier cambio de requerimientos sin afectar en el progreso del mismo. Para la elaboración de la propuesta, se realizó la implementación de una herramienta de integración de datos denominada Denodo, y el desarrollo de dos aplicaciones, la primer aplicación consiste en gestionar el tráfico de información que circula por el centro de procesamiento de datos y el almacenamiento del gran volumen de información, y la segunda aplicación tiene como finalidad controlar el acceso de usuarios a las diversas aplicaciones con las que cuenta IOTMACH. Para la creación de las aplicaciones, los recursos utilizados fueron: para la gestión de Logs se efectuó la instalación de servidores de mensajería que funcionan bajo los protocolos MQTT y Kafka, el framework de desarrollo empleado para comunicar los protocolos fue Node.JS; para el almacenamiento de la información se contó con servidores de bases de datos tales como PostgreSQL y MongoDB; para la gestión de usuarios se instaló un servidor de directorios como lo es LDAP manipulados en un sistema de autenticación construido en

Node.JS; y se desarrolló el módulo de seguridad bajo el framework Django, en el cual se implementan los controles necesarios para minimizar las vulnerabilidades.
Al culminar la propuesta, se obtiene que a través de las aplicaciones desarrolladas se puede gestionar el flujo de información que circula por el centro de procesamiento de datos, la creación de vistas distribuidas alimentadas por los servidores de bases de datos, y la autorización a los usuarios para que puedan hacer uso de las aplicaciones que forman parte del sistema IOTMACH.
Palabras clave: IOT, Logs, protocolos, MQTT, Kafka

ABSTRACT

IOT today is becoming a common term in our town, all the already created systems and those just are brewing are implementing the ideology of IOT, the same that is oriented in communicating common devices with each other through the internet, for this reason the IOTMACH system is focused on being one of the pioneers in implementing an infrastructure to accommodate IOT applications together with wireless sensor networks to apply to the process automation of various fields such as agriculture, fishing, livestock, mining and others.
The present work degree is based on the implementation of management tools Logs, data integration and security for the IOTMACH system, which is needed for managing large volumes of data at high speeds, handling multiple information sources and control user access to the information handled by the system. For these reasons it requires the implementation of tools to manage data, which provide the integration of sources where you have stored information and handling of users who have access to the system through the allocation of roles with different privileges.
For the preparation of this work, the implementation of a tool data integration called Denodo, and the development of two applications was made, the first application is to manage data traffic flowing through the data center and storage the large volume of information, and the second application is to control user access to various applications that IOTMACH account. For creating applications, resources used were: to manage logs installing messaging servers that operate under the MQTT and Kafka protocols, for the storage of information took place was featured server database such as PostgreSQL and MongoDB, for managing a directory server users as it is LDAP, and development framework used in both applications is Node.JS, which in turn uses the JavaScript programming language installed.
Key words: IOT, Logs, protocols, MQTT, Kafka

CONTENIDO

Pág.

ÍNDICE DE GRÁFICOS

ÍNDICE DE TABLAS

GLOSARIO

Autenticación: es el proceso de verificación de la identidad digital del remitente de una comunicación como una petición para conectarse a algún sistema o recurso.

Autorización: Proceso por el cual la red de datos autoriza al usuario identificado a acceder a determinados recursos de la misma.

BDD: Base de Datos, llamado así a los bancos de información que contienen datos relativos a diversas temáticas compartiendo entre sí algún tipo de vínculo o relación que busca ordenarlos y clasificarlos en conjunto.

Bróker: intermediario de procesos.

CPD: Centro de Procesamiento de Datos, es una ubicación donde se concentran los recursos necesarios para el procesamiento de la información de una organización.

DTLS: Datagram Transport Layer Security - seguridad de capa de transporte de datagramas, es un protocolo que proporciona privacidad en las comunicaciones para protocolos de datagramas.

Logs: mensaje que genera el programador de un sistema operativo, alguna aplicación o algún proceso, en virtud del cual se muestra un evento del mismo.

IOT: Internet Of Things – Internet de las cosas, es la interconexión digital de objetos cotidianos con el internet.

MQTT: Protocolo que está orientado a la comunicación de sensores, debido a que consume muy poco ancho de banda y puede ser utilizado en la mayoría de los dispositivos empotrados con pocos recursos.

KAFKA: es una plataforma unificada, de alto rendimiento y de baja latencia para la manipulación en tiempo real de fuentes de datos. Puede verse como una cola de mensajes, bajo el patrón publicación-suscripción, masivamente escalable concebida como un registro de transacciones distribuidas.

SQL: Lenguaje de consultas estructuradas. Es un lenguaje declarativo de acceso a las bases de datos relacionales que permite especificar diversos tipos de operaciones en ellas.

RDBMS: Sistema de gestión de bases de datos relacionales. Es un software que te permite crear, actualizar y administrar una base de datos relacional.

INTRODUCCIÓN

El Internet de las Cosas, es un concepto que se refiere a la interconexión de dispositivos electrónicos para establecer comunicación entre sí utilizando el internet. Se pueden realizar varias aplicaciones con el IOT lo cual en su mayoría implica tener sensores y actuadores que estén monitoreando lo que sucede en el entorno en el que se esté trabajando, para ello estos deben enviar información hacia un servidor para respaldar el monitoreo relacionado, haciendo necesario un proceso que manipule los datos generados, establezca comunicación de las aplicaciones con los sensores/actuadores; la información alojada en los servidores, podrían estar en diferentes fuentes de datos lo que complicaría la manipulación de estos en varias aplicaciones, lo cual implica controlar el acceso a los usuarios que manejarán a estas.

El presente trabajo se enfoca en realizar la implementación de herramientas de gestión de logs, integración de datos y de seguridad para el sistema IOTMACH, facilitando la comunicación de los dispositivos de redes de sensores inalámbricos con el centro de procesamiento de datos, gestionando la información que circula desde un adaptador de protocolos hasta las bases de datos.

La gestión de Logs implica en manipular la información que arriba al servidor, guardando dicha información en las bases de datos donde deben ir destinadas. Existen bases de datos cuya estructura varía, es decir, se tiene una base de datos relacional y una base de datos no relacional, por lo que podría complicar el consumo de la información por parte de las aplicaciones alojadas en el CPD, para ello es conveniente la implementación de una herramienta que integre y virtualice estos datos para darle la comodidad a las aplicaciones de que puedan acceder a la información como si fuese un solo conjunto de datos. La información es vital para cualquier sistema, y para ello todo sistema debe contar con un sistema de autenticación y control de seguridad que brinde acceso a usuarios basándose en los roles y privilegios para autorizarles el paso a las aplicaciones y así manipular la información.

La importancia del desarrollo del prototipo para una aplicación IOT es bastante amplia, ya que funciona como un eje principal de distribución de información entre aplicaciones y dispositivos electrónicos, a su vez ofrece la herramienta para integrar la información para tomar como un solo conjunto de datos que provienen de diferentes orígenes, y también controlar el acceso a los usuarios hacia las aplicaciones alojadas en el CPD.

Este documento se encuentra esquematizado de la siguiente manera:

En el **capítulo 1**, se establece el ámbito de la aplicación dando un enfoque general del tema, a su vez se encuentra el establecimiento de los requisitos en la que se describen los problemas y necesidades a solucionar, y también está la justificación en la cual se describe la importancia del tema dentro de IOTMACH, las metodologías y tecnologías empleadas para darle solución a la temática planteada.

Dentro del **capítulo 2**, se orienta a describir la fundamentación teórica y los procesos de desarrollo empleados para elaborar los prototipos que den solución a la temática establecida al inicio del presente documento.

El **capítulo 3**, se enfatiza en establecer el plan de pruebas a las que se someterán los prototipos desarrollados, y a la vez mostrar los resultados que arrojan las pruebas dando a analizar los puntos fuertes y débiles de los prototipos realizados.

1. DIAGNÓSTICO DE NECESIDADES Y REQUERIMIENTOS

1.1 Ámbito de Aplicación

El Internet de las cosas (IOT), es un término al que se refiere como la interconexión de los objetos que normalmente se utiliza en la vida diaria con el internet, haciendo que las actividades que realizan las personas sean mucho más sencillas de ejecutar.

Existen millones de personas en el mundo que utilizan tecnología integrada en un sistema común que proporciona el control e interacción entre dichas tecnologías [1]. El IOT es un término que en la actualidad resulta muy familiar, puesto que, la mayoría de sistemas y dispositivos están interactuando entre sí haciendo que los usuarios se sientan bastante cómodos al manipularlos. El IOT se implementa junto a múltiples objetos inteligentes, tales como celulares, TVs digitales, relojes, entre otros dispositivos con las que se puedan obtener y enviar información.

El presente trabajo de titulación, es un proyecto de IOT enfocado al desarrollo de una plataforma que brinde acceso a varios usuarios hacia múltiples aplicaciones como la agricultura de precisión, hogares inteligentes, inteligencia de negocios, entre otros; siendo que el grupo de investigación IOTMACH está orientado a aplicaciones que automaticen los procesos de agricultura de precisión, es por ello que se necesita del uso de redes sensores inalámbricos que estén realizando las tomas de las variables del ambiente que luego son enviadas al servidor. Una Red de Sensores Inalámbricos (WSN) se compone de pequeños nodos que se despliegan al azar sobre un área con el fin de detectar diversos parámetros relacionados con los fenómenos físicos [2].

Una WSN se emplea para la adquisición de datos específicos tales como la temperatura, humedad, radiación, entre otros; una vez tomados estos datos se requieren tomar acciones en caso de que lleguen a ocurrir eventos que le impida a un cultivo trabajar con normalidad, es aquí donde entran los actuadores, artefactos que necesitan un evento para ejecutarse, los mismos que pueden ser desde un ventilador hasta un pequeño LED.

Los valores obtenidos por una WSN son almacenados en un centro de procesamiento de datos (CPD). Las WSN son emisores constantes de información y a medida que el tiempo avanza, irán aumentando cada vez en mayor número, implicando en el crecimiento constante de un gran volumen de datos llegando al CPD en tiempo real, teniendo el peligro de perdida de información, ya que al no tener un sistema que facilite la gestión de grandes cantidades de datos en diversas velocidades complicaría el rendimiento del sistema. La información que llega al CPD puede ser tomada como logs, y debe ser gestionada como tal.

La Gestión de Logs comprende el tratamiento de grandes volúmenes de mensajes que generan registro de datos [3]. El paso de la información entre la WSN y el CPD, es realizado gracias a la implementación de herramientas de gestión de Logs, las mismas que se encargarán de la manipulación y control del tráfico de los datos; dicha información debe de ser almacenada en los servidores de bases de datos. Cabe indicar que existen otras aplicaciones que hacen uso de la información registrada en los servidores y a su vez esta aplicaciones deben ser manipuladas por múltiples usuarios los cuales deben tener privilegios para acceder a los mismos, es por ello que se debe emplear un mecanismo de autenticación tradicional que consiste en que el usuario proporciona sus credenciales a un servidor para obtener acceso a un recurso o sistema protegido para tener acceso a las aplicaciones del sistema [4].

Estas aplicaciones utilizan las bases de datos del CPD para sus principales funciones, pero al existir diversidad en tipos de bases de datos complica el consumo de la información alojada en ellas, para ello la integración de datos aparece como un proceso de conciliación de datos provenientes de diferentes fuentes y/o bases de datos, es utilizada para unir múltiples fuentes de datos para obtener información fiable y limpia [5]. La implementación de una integración y virtualización de datos permite la agilización de acceso a las distintas bases de datos como si fuera un solo conjunto, debido a que la virtualización facilita el manejo de la información de manera simple e integrada a una velocidad muy alta.

1.2 Establecimiento de requerimientos

El IOT permite la conexión de diferentes dispositivos entre sí para comunicar y/o emitir información utilizando dispositivos WSN, implicando en el crecimiento considerable de los datos generados por dichos dispositivos, haciendo que un sistema convencional al recibir grandes cantidades de información con mayor rapidez colapse en un momento determinado debido a que sus capacidades de procesamiento de datos no están diseñados para trabajar con grandes cantidades de información en tiempo real.

Dentro de IOTMACH existe la principal necesidad de gestionar la transmisión de los datos en altas velocidades generadas por las WSN y que llegan al CPD, ello conlleva a la falta de manejo de la diversidad de la información necesitando así la integración de los datos que se encuentran en los servidores que faciliten el consumo de dicha información en las aplicaciones; a su vez se precisa del control de acceso a los usuarios que utilizarán las aplicaciones, gestionando sus niveles de acceso y manipulación de la información. Es por eso que se requiere la implementación de herramientas que faciliten la gestión de grandes volúmenes de datos en la mayor fluidez de tiempo posible, debido

a que cada cierto tiempo se genera información que necesita ser almacenada en un servidor de base de datos; teniendo en cuenta que la información recibida por las WSN tendrá diversos tipos de datos y se guardan en bases de datos de diferente estructura.
Al existir múltiples bases de datos, habrán aplicaciones que requieran acceder a los diferentes servidores de bases de datos, lo cual en varias ocasiones complicaría los tiempos de respuesta de la información solicitada, para ello se necesita agilizar el acceso a la información contenida en estas fuentes de datos realizando un proceso de integración y virtualización de las mismas; para lograr este proceso se requiere implementar herramientas que brinden el servicio de integración de datos.
La información en todo sistema es de vital importancia, y por ende se requiere la mayor protección posible para la misma. Dentro del CPD se encuentran ubicadas varias aplicaciones, las cuales tienen la facultad de manipular la información almacenada en las bases de datos, los cuales pueden ser objeto de extracción y utilización indebida por personas malintencionadas. Para mitigar los riesgos que corre la información, se requiere de procesos que sirvan como protección a los datos, a su vez se necesita de un control de autenticación de usuarios, los mismos que para acceder a las aplicaciones deben de tener los privilegios necesarios para manipular los datos que se gestionen dentro de cada sistema que haga uso de la información.

1.3 Justificación

Dentro de la Universidad Técnica de Machala, en la Unidad Académica de Ingeniería Civil, el grupo de investigación IOTMACH se encuentra realizando una aplicación basada en la agricultura de precisión, la cual consiste en emitir información de parcelas o zonas de cultivos a través de WSN hacia el CPD, para ello el presente trabajo está enfocado en la implementación de Herramientas de Gestión de Logs e Integración de Datos y brindarle seguridad de acceso a la información para el sistema IOTMACH, para lo cual se desarrollará un prototipo tecnológico que pueda abarcar con las temáticas antes mencionadas.
Una WSN debe tener una conexión con el CPD y para ello deberá existir un Adaptador de Protocolos encargado de receptar la información que emite la WSN y a su vez este adaptador envía la información hacia un gestor de Logs. Para tener mayor fluidez de los datos entre estos componentes, se debe hacer uso de herramientas que manejen el comportamiento de envió/recepción de información en tiempo real, tal es el caso de servidores de mensajería que tienen las características antes mencionadas, para ello se utiliza un bróker de mensajería para el Adaptador de Protocolos y otro para la Gestión de Logs, los mismos que serán Mosquitto y Apache Kafka respectivamente. Al hacer

uso de diferentes servidores de mensajería, se creará un middleware que realice la función de enviar y recibir información entre estos servidores en tiempo real.

Los mensajes que llegan a Kafka deben ser almacenados en una base de datos, en la cual participan middlewares encargados de consumir los datos que van llegando al bróker, realizando funciones específicas en cada middleware consumidor ya sea publicando los mensajes en un adaptador de protocolos, guardando los datos en una BDD relacional o almacenando la información en una BDD no relacional. El desarrollo de los middlewares se realiza con el framework Node.JS cuya estructura se enfoca en procesos en tiempo real; la metodología de desarrollo empleada es Kanban, debido a su flexibilidad en el proceso de cumplimiento de tareas y el enfoque que le da el cumplimiento de las tareas planteadas dando mayor control y planeamiento para culminar ordenada y organizadamente el proyecto.

Al existir diversas bases de datos utilizadas como fuentes de recepción de información, implica la implementación de un proceso de alto nivel de almacenamiento de datos, para lo que se necesita integrar los datos de una manera en la que se pueda tener una gran facilidad de acceso a la ya antes mencionada información, la manera más factible de conseguir un rápido acceso a los datos es implementar una herramienta que permita la integración de datos, ya que al ser una capa de abstracción y una capa de servicios de datos, puede utilizarse de forma complementaria con otras aplicaciones.

La información, al ser lo más importante de todo sistema, debe estar protegida contra usuarios que no tengan permisos para manipular los datos, para ello se debe proporcionar la mayor seguridad posible, es decir, que cada usuario que tenga acceso a los datos, deberá tener los privilegios y tendrán su propio nivel de acceso a los datos del sistema.

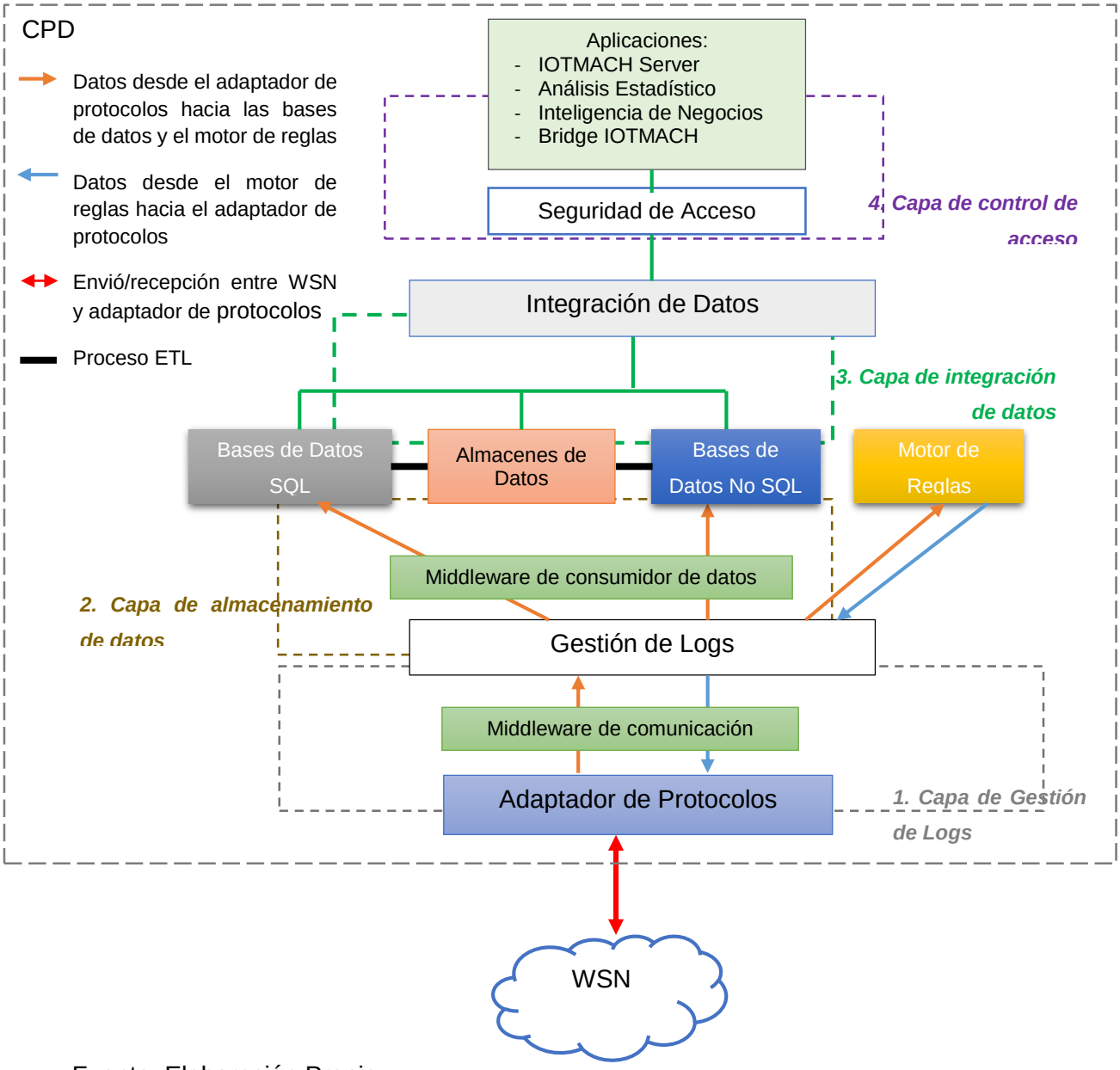
La importancia del prototipo dentro de IOTMACH es muy relevante, ya que funciona como un puente central para la distribución de los datos, siendo la columna principal de comunicación entre la WSN con las bases de datos y el facilitador del acceso de la información con las aplicaciones instaladas en el CPD, ofreciendo facilidad de acceso a las bases de datos que se manejan como si fuesen un solo conjunto de información. Entre los principales beneficios que presenta el desarrollo del prototipo, se tiene la capacidad de comunicar un dispositivo electrónico con un conjunto de servicios y/o aplicaciones, la integración de un conjunto de fuentes de información, y el mitigar las vulnerabilidades de acceso a la información que se pueden presentar. Tanto los servidores de mensajería utilizados para el traslado de la información, la integración de los datos y la implementación de un control de seguridad de acceso a datos resultan bastante beneficioso no solo para IOTMACH, también para que sean utilizados para otro tipo de proyectos que requieran tráfico, integración y seguridad de datos.

2. DESARROLLO DEL PROTOTIPO

2.1 Definición del prototipo tecnológico

El prototipo tecnológico de la propuesta a desarrollar, está conformado por diferentes capas, las mismas que son la capa de gestión de Logs, capa de almacenamiento de datos y la capa de integración de datos y capa de control de acceso; las mismas que cumplen con funciones específicas, graficadas a continuación:

Gráfico 1: Representación de la gestión del prototipo



Fuente: Elaboración Propia

Como está expresado en el **Gráfico 1**, el prototipo se encuentra conformado por tres capas:

La capa 1, se encarga de receptar la información que es emitida por una WSN y por un motor de reglas a través de servidores de mensajería, los mismos que se derivan en un adaptador de protocolos y en un gestor de Logs, en los cuales existe un middleware que trabaja como un puente de información entre los servidores para facilitar el traspaso de la información tanto interna como externamente de la CPD.

La capa 2, es el almacenamiento de los datos que pasan a través de los servidores de mensajería. Esta capa es la que se encarga de guardar la información emitida por la capa 1 en BDD tanto relacional (SQL) como no relacional (No SQL), dicha tarea de acaparamiento es realizada por un middleware cuya función es consumir la información del servidor de mensajería e ir guardando los mensajes en la BDD respectiva.

La capa 3, está conformada con la integración de datos, por lo que en esta capa se implementa una herramienta de integración y virtualización de datos, que facilite el acceso a la información almacenada en las diferentes BDD como si fuera un solo conjunto de información, cuyo acceso a la misma la tienen otras aplicaciones que requieran utilizar dichas fuentes de datos con mayor facilidad y sin tener que estar realizando conexiones individuales a las ya mencionadas BDD.

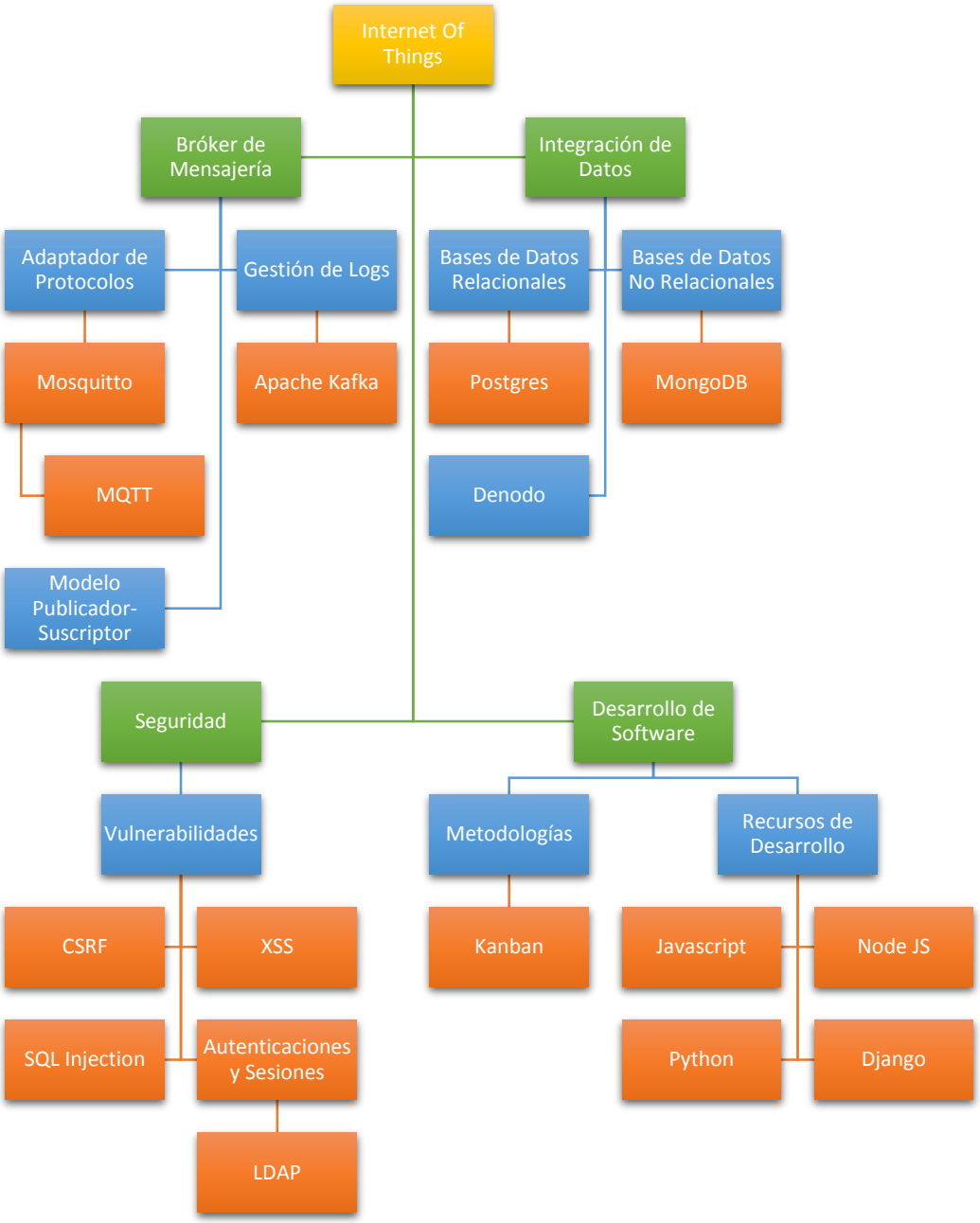
En la capa 4, se encuentra el control de acceso a las aplicaciones, lo cual radica en brindar un control de autenticación para el acceso de los usuarios a las aplicaciones que maneja el CPD de IOTMACH, cada usuario posee sus credenciales correspondientes para su ingreso y a su vez solo pueden realizar acciones dependiendo de los permisos basados en roles que le fueron asignados.

2.2 Fundamentación teórica del prototipo

Dentro del prototipo propuesto, incluye múltiples temáticas para darle solución a los problemas de transporte de información, integración de datos y control de seguridad que presenta IOTMACH.

Para tener un mejor entendimiento sobre cuáles son las temáticas que abarca el desarrollo del prototipo el **Gráfico 2** representa el esquema de los temas empleados.

Gráfico 2: Mapa conceptual de la fundamentación teórica.



Fuente: Elaboración Propia

A continuación se detallaran los aspectos teóricos con los que se desarrolla el prototipo:

2.2.1 *Internet de las Cosas*. El internet de las cosas (IOT) es un término, que se refiere a la manera en cómo están interconectados digitalmente los objetos ordinarios con el internet. El IOT permite a un gran número de dispositivos conectarse inalámbricamente entre sí para un intercambio de datos y la interacción a través de Internet como un medio de comunicación global [6].

Es un concepto que se remonta a finales del siglo pasado, vinculando la conexión de internet con los objetos cotidianos [7], y es que el IOT avanza día a día conforme aparecen nuevas tecnologías que se aplican de a poco en las necesidades del vivir diario, haciendo un punto de inflexión para que los objetos que regularmente se utilizan estén conectados al internet e interactúen entre ellos.

El IOT puede ser combinado con el uso potencial de las WSN para facilitar la implementación de múltiples aplicaciones, ya sea desde encender un led hasta la creación de ciudades inteligentes [8], es aquí donde toma punto de partida un proyecto IOT, ya que se debe tener claro el objetivo general y la visión que tendrá el mismo para proceder a desarrollarse y posteriormente implementarse.

La tecnología IOT es adecuada para aplicarse en el monitoreo del medio ambiente jugando un papel clave con la capacidad de sentir y distribuir los fenómenos como datos naturales [9], dichos fenómenos pueden ser temperatura, las precipitaciones, la altura del río, el viento, la fricción entre otros; al desarrollar un proyecto IOT se deben tomar en cuenta estos fenómenos, ya que al saber cuáles son los datos naturales a capturar se emprenderá un proyecto, por ejemplo, los datos naturales de temperatura y humedad dan paso a elaborar un proyecto IOT de agricultura de precisión, pesca de precisión y un sin número de proyectos más.

2.2.2 *Bróker de mensajería.* Un bróker de mensajería es un sistema intermediario utilizado para la validación, la transformación y el ruteo de mensajes entre distintas aplicaciones, puede recibir mensajes de múltiples orígenes, determinar el destino y la ruta correcta del mensaje en el canal correcto [10], permitiendo minimizar el grado de conocimiento mutuo que estas aplicaciones necesitan tener, para poder comunicarse entre sí. Un Bróker de mensajería proporciona un modelo de contexto común en la que las aplicaciones pueden compartir información, dicho modelo funciona no sólo como una interfaz común para el contenido del mundo real, también lo hace para las aplicaciones que comparten las mismas características [11], esto implica que un bróker de mensajería funcione como un lenguaje intercomunicador entre aplicaciones construidas en distintos lenguajes de desarrollo.

A todo esto, existen múltiples bróker de mensajería tales como Apache ActiveMQ [12], WebSphere Message Broker [13], JBoss Messaging [14], Mosquitto [15], Apache Kafka [16], entre otros más. Debido a las características y detalles de implementación, trabaja con Mosquitto y Apache Kafka como brókeres de mensajería, utilizados como un adaptador de protocolos y herramienta de gestión de Logs respectivamente.

2.2.2.1 *Adaptador de protocolos.* Un adaptador de protocolos es una puerta de salida para la comunicación de tecnologías [17]. Es usado como conexión directa entre la WSN y la CPD, y en la que existen diversos protocolos destacándose entre ellos CoAp y MQTT; siendo CoAP un protocolo de comunicación restringida, está diseñado para implementaciones con restricciones, pero a su vez pierde dicha característica al añadir una capa de seguridad DTLS al protocolo, aumentando el consumo de recursos de los dispositivos de una WSN, es por eso que MQTT, otro protocolo de comunicación, es la mejor alternativa en la implementación de una comunicación IOT, por ser un protocolo concebido para ambientes con recursos restringidos, teniendo un sistema de seguridad integrado [18].

- *Modelo Public-Subscribe.* El modelo de publicador-suscriptor es una evolución importante dentro de los sistemas distribuidos, ya que centra la atención en cómo las aplicaciones crean, comparten y manipulan los datos [19]. Los publicadores crean datos y los comparten, en cambio, los suscriptores recuperan dichos datos a través de tópicos, que funcionan como URLS para que las aplicaciones se comuniquen entre sí.

- *Mosquitto.* Mosquitto es un bróker de mensajería de código abierto que implementa el protocolo MQTT [15]. Lo cual implica que el bróker o servidor esté enviando y recibiendo mensajes con el modelo de arquitectura pub/sub (publicador/suscriptor). En pocas palabras, Mosquitto es el encargado de gestionar las peticiones de mensajes que los usuarios realizan [20], publicando los mensajes que a él llegan a los clientes suscritos al bróker.

  - *MQTT.* Se trata de un protocolo de mensajería sencillo y ligero, diseñado para dispositivos limitados, bajo ancho de banda y latencia alta que utilizan el modelo de publicador/suscriptor [21].

El protocolo MQTT crea una conexión con los dispositivos, la mantiene abierta utilizando poca energía, y entrega los mensajes con pequeña sobrecarga de transporte [22]. Esto indica el hecho de que MQTT sea uno de los protocolos de mayor acogida a nivel mundial en la trasferencia de mensajes entre los dispositivos dándole más facilidad para implementar la IOT. En otras palabras es un protocolo de mensajería que facilita la comunicación entre dispositivos o sistemas de diferente lenguaje de desarrollo.

El protocolo MQTT se basa en el modelo de arquitectura de la publicación de mensajes y la suscripción de tópicos, en las cuales múltiples clientes se conectan a un bróker y se suscriben a los tópicos que les interesa [15]. Este modelo hace a MQTT muy versátil,

ya que permite que un cliente pueda publicar mensajes (publicadores) en un tópico donde muchos clientes pueden recibir un mensaje (suscriptores) y viceversa.
Los mensajes en MQTT se publican en los tópicos, no existe la necesidad de crear un tópico basta con publicar sobre el mismo utilizando un slash (/) como separador [15]. Los tópicos se crean simplemente con publicarse en ellos lo que no causa una restricción al generar tópicos para enviar mensajes a distintos clientes que estén suscritos a ellos.

2.2.2.2 *Gestión de Logs.* Empezando desde un sistema operativo hasta la más pequeña aplicación realizada, la gestión de Logs sirve para conocer el estado y el tratamiento brindado a la información. Los Logs son una parte importante de cualquier sistema informático seguro, proporcionan una visión útil del pasado y los actuales estados del sistema [3], ayudan a asegurar que las diversas actividades se registran en los detalles suficientes para un período de tiempo adecuado [23].
La gestión de Logs no es una terminología reciente, ya que ha existido desde siempre en los sistemas de mayores capacidades de procesamiento y que manejan información sensible; se han convertido en elementos básicos para un sistema informático debido a la facilidad de generación de registros de actividad del sistema [24]; los servidores y las aplicaciones pueden generar Logs en una variedad de procesos, desde simplemente capturar el inicio de sesión de un usuario, hasta brindar información de la realización de un proceso más complejo.
Teniendo en cuenta la conceptualización de la gestión de Logs, se puede considerar a los datos que generan tanto la WSN como el Motor de reglas como Logs, ya que al generar estados y valores, cuentan con las características que un Log requiere para ser tomado como tal. Es por ello que es válido mencionar que un bróker de mensajería puede ser considerado como herramienta de gestión de Logs.

- *Apache Kafka.* Kafka es otro bróker de mensajería basado en el modelo de arquitectura publicador/suscriptor, muy similar a MQTT pero con ciertas características que lo diferencia.

Kafka es un sistema de mensajería distribuido de publicación-suscripción puede manejar un gran volumen de datos y permite pasar los mensajes de un punto a otro; los mensajes se conservan en el disco y se replican dentro del clúster para evitar la pérdida de datos [16]. Kafka combina los beneficios del almacenamiento de registro tradicionales y sistemas de mensajería [25].

A diferencia de MQTT, la terminología empleada en publicadores, en Kafka se los denomina productores, a los suscriptores son denominados consumidores, mientras que los tópicos siguen denominándose de igual manera.
Al igual que en MQTT, la transmisión de mensajes se la realiza a través de tópicos, los cuales los tópicos se dividen en particiones, para cada tópico, Kafka mantiene un mínimo de una partición [16]. Esto hace que los mensajes se almacenen en archivos de Logs y conforme están particionados tendrán un tiempo de vida dentro del bróker de Kafka, y los cuales hacen que los nuevos consumidores suscritos a Kafka reciban los mensajes producidos.

2.2.3 *Integración de datos*. La integración de datos consiste en la combinación de distintas fuentes de información. La complejidad que tiene una integración de datos hace hincapié en los niveles de almacenamiento, la estructura y los niveles en que los datos se pueden integrar y operar como una sola entidad [26]. Tanto así que resulta más fácil acceder a la información de diferentes BDD si estuviesen integradas en un solo conjunto de datos.
La integración de datos es requerida más frecuentemente por aplicaciones que necesitan de la información almacenada en las BDD, debido a que transforma las transferencias de datos entre diferentes esquemas antes de que esté listo para ser recuperado por los usuarios finales [5].

2.2.3.1 *Bases de datos relacionales*. Las bases de datos relacionales, cumplen con el modelo relacional que consiste en tener relaciones entre diferentes tablas de registros de datos, cuyos registros contienen atributos de diferente tipo de información. El lenguaje utilizado para realizar consultas a las BDD es el SQL y en cuanto a los RDBMS este tipo de BDD existen muchos, entre los más destacados están Oracle, MySQL, SQLITE3, SQL Server, PostgreSQL, entre otros.

- *PostgreSQL.* Es un sistema de gestión de bases de datos objeto-relacional siendo el sistema de gestión de BDD de código abierto más potente del mercado, utiliza un modelo cliente/servidor y usa multiprocesos en vez de multihilos para garantizar la estabilidad del sistema [27]. Un fallo en uno de los procesos no afectará el resto y el sistema continuará funcionando.
Esto hace que Postgres sea una buena alternativa para la gestión de datos estructurados, ya que su robustez hace que sea uno de los gestores de BDD más utilizados en la actualidad. Al ser de código libre, PostgreSQL es desarrollado no por

una empresa o una persona específica, sino que es desarrollado por un grupo de desarrolladores que trabajan sin fines de lucro y a su vez son apoyados por organizaciones comerciales.

2.2.3.2 *Bases de datos no relacionales*. Un modelo de BDD no relacional, rompe con el paradigma de una BDD relacional, principalmente porque ya no existen relaciones entre registros. Al no utilizar SQL, NoSQL dan mayor flexibilidad y rapidez en obtención de los datos al no requerir estructuras fijas en el almacenaje de la información. Entre los gestores de BDD NoSQL se destacan CouchDB, RavenDB, MongoDB, entre otras más.

• *MongoDB.* Es un sistema de BDD NoSQL orientado a documentos, en lugar de guardar los datos en tablas como se hace en las base de datos relacionales.
MongoDB almacena los datos utilizando un modelo datos similar a JSON, los documentos contienen uno o más campos, incluyendo matrices, datos binarios y subdocumentos, y pueden variar de un documento a otro. Esta flexibilidad permite a los equipos de desarrollo modificar el modelo de datos con rapidez a medida que cambian sus requisitos de aplicación [28].

2.2.3.3 *Denodo*. Es una herramienta que permite integrar y virtualizar datos desde distintas fuentes de información. La virtualización de datos es sinónimo de agilidad en el acceso a la información según necesiten las aplicaciones consumidoras ya que integra datos de fuentes dispersas, en distintas localizaciones y formatos, sin aplicar replicación, el resultado es un acceso más rápido a todos los datos, menores costes asociados y una mayor agilidad frente al cambio [29]. Denodo proporciona un rendimiento impresionante en grandes volúmenes de información, almacenes de datos lógicos y escenarios operativos; se acelera la adopción de la virtualización de datos en la nube; y se agiliza el uso de los datos por los usuarios de negocios, con el descubrimiento de datos de autoservicio y de búsqueda [30].
Es una herramienta potente, con rápido crecimiento puesto a su versatilidad de acceso a distintas fuentes de datos y la integración que le da a dichas fuentes. Denodo ofrece la conectividad con grandes fuentes de datos, ya sean base de datos relacionales como Oracle, Apache Hive, Postgres e inclusive bases de datos no relacionales como MongoDB; además ofrece la creación de consultas distribuidas, es decir, integrar los datos provenientes de diferentes fuentes en un solo conjunto de información.

2.2.4 *Seguridad de la información.* En la actualidad, la información es uno de los activos más importantes en la mayoría de las organizaciones, siendo de vital importancia para el bienestar y el éxito de las mismas, es imprescindible que la práctica de la seguridad de la información esté debidamente jerarquizada desde la más altos a los más bajos niveles de la organización [31]. Es imprescindible contar con un proceso de control de seguridad de la información, más aun clasificar los niveles de seguridad a partir de los roles que se manejan en una organización determinada.

Al implementar seguridad en la información, también hay que hablar de la seguridad de las aplicaciones. Los sistemas más vulnerables son las aplicaciones web, ya que existe mayor registro de problemas presentes dentro de este tipo de aplicaciones.

2.2.4.1 *Vulnerabilidades en aplicaciones*. Todas las aplicaciones son vulnerables a cualquier ataque cibernético, debido a que por más minúsculo que sea una vulnerabilidad un usuario mal intencionado puede explotar la misma. Una vulnerabilidad se refiere a una debilidad en el sistema de requisitos de seguridad, diseño, codificación u operación que podría ocurrir ocasionando fallas en la seguridad del sistema [32].

Existen múltiples vulnerabilidades que causan mayor problema a las aplicaciones web a nivel mundial [33], las cuales son Cross-Site Request Forgery, SQL Injection, Cross Site Scripting, Autenticación y Sesiones, que son las más destacadas.

- *Cross-Site Request Forgery (CSRF).* Es una de las vulnerabilidades más explotadas en los sistemas web, ya que se encuentran en las operaciones sensibles que realizan a través de solicitudes de páginas, desde el cambio de un perfil de usuario hasta la realización de transacciones financieras no autorizadas [34]. Este tipo de vulnerabilidades son mayormente explotadas cuando se maneja información confidencial como las contraseñas de alguna red social o en transacciones financieras hechas desde un sitio web.

Un enlace a una página web se puede colocar dentro de un mensaje de correo electrónico o publicado en un perfil de red social, si el sitio web de destino es vulnerable a XSS, el código de ataque se puede colocar en el mismo dominio, confundiendo aún más a la víctima del ataque y pasando por alto las medidas de seguridad que se basan en CSRF Tokens [35], estos tokens actualmente son manejados por los middlewares de desarrollo para mitigar los ataques en CSRF. Un ataque CSRF normalmente sucede con un usuario víctima que tiene una sesión activa con un sitio de confianza y al mismo tiempo que también visita a un sitio malicioso, este a su vez inyecta una solicitud HTTP para el sitio de confianza en la sesión del usuario víctima comprometer su integridad

[36]. Los atacantes pueden aprovechar esta situación y engañar a la víctima para hacerlo ejecutar código malicioso y obtener información que el usuario víctima le pueda suministrar.

• *SQL Injection.* Este tipo de vulnerabilidades es otra variante de las más explotadas en los sitios web, pero es una de las más peligrosas debido a que es la base de datos quien sufre el ataque.

Para explotar esta vulnerabilidad un atacante debe tener acceso a un parámetro que pasa a través de la aplicación web a la base de datos. Anexando los comandos SQL maliciosos en el parámetro, el atacante hará que la aplicación web para enviar la consulta malintencionada en el servidor de base de datos y ejecutarlo, en caso de no controlar la entrada de valores, la consulta será vulnerable a los ataques de inyección SQL [37]. Es muy común que este tipo de vulnerabilidades se presenten por desarrolladores inexpertos o por descuidos que se pueden producir al programar un sistema.

La razón principal para que existan estas vulnerabilidades es que, las consultas SQL se ejecutan sin una correcta validación de entradas del usuario, para acceder o alterar datos, en el cual, un usuario malintencionado introduce algunos datos manipulados, y la aplicación utiliza esos datos para construir una instrucción SQL [38]. El no control de los datos que ingresan en las consultas SQL efectuadas en un sistema puede causar un riesgo a la seguridad de la información permitiendo obtener datos confidenciales a usuarios que no deberían tener acceso a los mismos.

Las BDD SQL son atacados por la inserción directa de código malicioso en sus parámetros de entrada, haciendo que el servidor no responda ante este código tomándolo como válido [39]. Este tipo de vulnerabilidades son los principales causantes de fallos de seguridad y filtro de información en las organizaciones, aunque actualmente existen técnicas que mitigan considerablemente los ataques SQL.

• *Cross Site Scripting (XSS).* Cross-site scripting (XSS) es un código de inyección de vulnerabilidad a nivel de aplicación, se produce cada vez que un web utiliza entradas sin restricciones a través de peticiones HTTP, BDD o archivos sin ninguna validación [40]. Este código permite que un usuario malicioso pueda robar información sensible almacenada en cookies y sesiones, realizando operaciones malévolas.

Los ataques XSS producen el daño más directo para la privacidad de un usuario, acceden a la información sensible [41]. Este tipo de ataques resultan muy cotidianos incluso para los sistemas más seguros a nivel mundial, por lo que son imprescindibles

y muy difíciles de controlar cada movimiento que un usuario realice dentro de una aplicación.

Si existe una vulnerabilidad XSS en una página web, estos ataques pueden ser enviados al servidor web, el navegador del usuario carga las páginas de respuesta y ejecuta los scripts maliciosos inyectados por el atacante en el servidor [42]. No cabe duda que resulta prescindible contar con un control para la detección de ataques XSS, aunque existen muchas técnicas y frameworks que mitigan estos ataques hay que prever que no existan fallos de seguridad hacia la información.

- *Autenticación y Sesiones.* Las autenticaciones son primordiales para cualquier sistema, permiten controlar el acceso de los usuarios hacia los datos a los que tengan privilegios de manipulación.

La autenticación permite a la aplicación identificar a un usuario conocido, en lugar de ser anónimo. Una autenticación es el proceso para determinar cómo la aplicación valida la identidad de un usuario [43]. Todos los sistemas actuales cuentan con un reconocimiento de usuarios que acceden a la aplicación y a su vez tienen privilegios para acceder a la información de la BDD. La sesión se asocia a un usuario en una aplicación web, y la información de la sesión se gestiona en el lado del servidor. Esta información, contiene el estado de usuario de inicio de sesión [44]. La aplicación web identifica a los usuarios con las sesiones y valida el acceso a la misma mediante dicha sesión.

Eliminar todas las vulnerabilidades en las aplicaciones web es prácticamente imposible. Los desarrolladores web cometen errores y no están familiarizados con todos los tipos de vulnerabilidades web que existen.

- *LDAP.* El control de usuarios es de vital importancia para cualquier sistema de información. LDAP es un protocolo que permite el manejo de la información de usuarios. Dicha información se proporciona al servidor a través de operaciones de conexión, el servidor de acuerdo con la información de identidad controla la petición de acceso propuesto por cliente [45]. Las aplicaciones que se le pueden dar a LDAP van creciendo considerablemente, convirtiéndose en el servicio de directorio de usuarios preferido a la hora de realizar el almacenamiento de información, gestión y consulta de los usuarios.

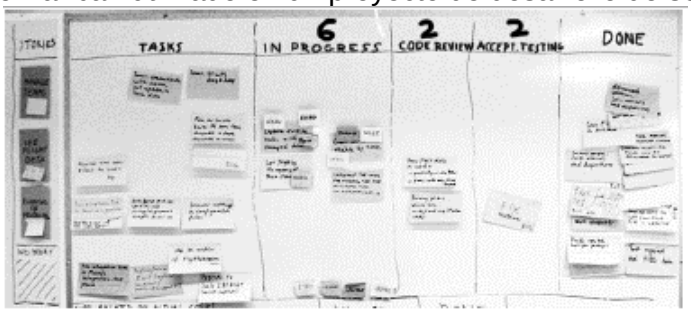
LDAP es un protocolo encargado de la administración de directorios, orientado a utilizarse para la gestión de usuarios de sistemas operativos, pero de a poco se puede ir aplicando su funcionamiento con los sistemas tradicionales.

2.2.5 *Desarrollo de software.* El desarrollo de software puede abordar diversos temas desde la instalación de un servicio hasta un atributo de una BDD. En esta ocasión se trata sobre las metodologías y recursos usados para el desarrollo del prototipo.

2.2.5.1 *Metodologías.* Utilizadas para realizar una estructuración, planificación y clasificación de recursos, una metodología de desarrollo de software ofrece una mayor organización a la hora de embarcarse a un proyecto.
Existen diversas metodologías en el medio, cada cual ofrece características adaptables en cuestión de tiempo y recursos que un grupo desarrollador tiene, entre las cuales se destacan RUP, RAD, XP, SCRUM, y algunas más que resultan ser las más utilizadas en el medio informático. Aunque también existen métodos que son usualmente ocupados en otras áreas distintas a la informática pero que con el paso del tiempo son implementadas en la gestión de proyectos de software, tal es el caso de Kanban el mismo que se describe a continuación:

- *Kanban*. Pertenece al grupo de metodologías agiles de desarrollo cuyo objetivo es obtener respuestas rápidas a las peticiones de los clientes. Kanban utiliza una interfaz basada en columnas como lo hace Scrum para el seguimiento del estado de los procesos sin necesidad de pre-organizar el trabajo ni armar roles de desempeño entre los colaboradores del proyecto.
Kanban visualiza las tareas que se pueden romper en cualquier momento para respetar los límites de trabajo en curso, el objetivo de la práctica de Kanban es visualizar y mejorar el flujo de valor mediante la optimización del tiempo; siendo más apropiado donde existe un alto grado de variabilidad en la prioridad de entrega [46]. Es adaptable en cualquier clase de proyecto, y se puede implementar como una metodología tradicional, emulando las fases en flujos de procesos reflejados en las columnas del tablero.

Gráfico 3: Tablero Kanban utilizado en un proyecto de desarrollo de software.



Fuente: On the impact of Kanban on software project work: An empirical case study investigation [47].

En Kanban se manejan equipos de proyecto para visualizar el flujo de trabajo, limitar el trabajo en curso, y medir los tiempos de los mismos. A pesar de la demanda de visualización del flujo de trabajo, no existen normas especiales relativas cómo poner en práctica el contenido del tablero Kanban [47]. En su forma básica como se muestra en el **Gráfico 3**, las tareas se representan normalmente con los tickets que se mueven dependiendo el proceso realizado a través del tablero.
Kanban limita la obra en curso de acuerdo con la capacidad de los miembros del equipo, equilibra la demanda en contra del rendimiento del trabajo en equipo entregado, ayuda a visualizar los problemas del proceso, disminuir los defectos y mantener un flujo constante de trabajo [48]. Al limitar el trabajo en curso se consigue un ritmo sostenible de desarrollo, produciendo productos de mayor calidad y mayor rendimiento del equipo. La combinación de la mejora del flujo y el software de mayor calidad ayuda a acortar el tiempo de entrega, lo que lleva a las versiones regulares de entrega generando mayor confianza entre los clientes.

2.2.5.2 *Recursos de desarrollo*. Para la elaboración del prototipo se utilizaron los siguientes lenguajes y frameworks para programar.

• *Javascript*. Es un lenguaje orientado a objetos con funciones de primera clase, utilizado como el lenguaje de script para páginas web, pero también usado en entornos sin navegador tales como Node.js o Apache CouchDB. [49]. Es un lenguaje de script que soporta estilos de programación funcional, orientada a objetos e imperativa.

• *Node.JS.* Es una plataforma para la creación de aplicaciones de servidor rápidas y escalables utilizando JavaScript [50]. Al ser un servidor web basado en JavaScript, se manejan eventos haciendo que la programación que se realiza en Node.JS sea

asíncrona. Un proceso asíncrono continúa con su ejecución inmediatamente después de enviar el mensaje al receptor. Es decir, si al existir un proceso A necesita realizar una función B, pues ingresa a realizar la acción de la función B, pero la ejecución continúa con el proceso A [51]. Esto hace que en Node.JS, al existir los eventos asíncronos nunca espera para otro evento para devolver datos o realizar alguna acción, esto hace que los tiempos de respuesta sean más rápidos, ya que al no tener que esperar por una respuesta, agilitan las presentaciones de formularios por citar un ejemplo.

- *Python*. Python es un lenguaje de programación interpretado en el cual no es necesario realizar una compilación [52]. Su sintaxis es sencilla, ya que está orientado a ser un lenguaje legible y entendible para cualquier persona que quiera incursionar en el mundo de la programación.

Un programa de Python se interpreta en forma de código de bytes tiene una sintaxis muy simple que proporcionan una gran comodidad y flexibilidad para los nuevo programadores [53]. Python es muy sencillo de utilizar, una de sus principales características es la reutilización de variables, es decir en un momento determinado una variable “a” puede tener un valor numérico y a línea siguiente tendrá en su contenido una cadena de caracteres. Python se puede emplear para la creación de software tanto para escritorio como para la web, inclusive en el ámbito móvil. Para lo cual existen variados paquetes que son utilizados como librerías para fines específicos. En cuanto al desarrollo de aplicaciones web se destaca Django, que es el framework para Python más empleado.

- *Django.* Es un framework para Python de alto nivel que fomenta el rápido desarrollo de aplicaciones web, ya que se encarga de gran parte de la creación de un servicio Web, para que un programador pueda centrarse en la escritura de su aplicación [54]. Es un framework libre que está teniendo mucha acogida en la actualidad, por su versatilidad y fácil entendimiento a la hora de desarrollar aplicaciones.

Django ofrece múltiples protecciones para posibles vulnerabilidades que puede tener una aplicación web entre las cuales están XSS protection, CSRF protection, SQL injection protection, Clickjacking protection, Host header validation, Session security [55]. Aunque todas estas funciones pueden ser utilizadas en una aplicación, hay que indicar que su implementación no cortará por completo las vulnerabilidades solo las reducirá considerablemente, sin tomar en cuenta los posibles errores de programación que vayan a existir.

2.3 Objetivos del prototipo.

2.3.1 *Objetivo General.*

Implementar herramientas de gestión de Logs, integración de datos y la seguridad de acceso para la distribución, virtualización y control de acceso a los datos de un sistema IOT, utilizando la metodología Kanban para el desarrollo del prototipo y las tecnologías necesarias en el funcionamiento del mismo.

2.3.2 *Objetivos Específicos***.**

- Gestionar el tráfico de datos que se da entre las redes de sensores inalámbricos con el centro de procesamiento de datos.
- Realizar el proceso de envió/recepción de información entre los protocolos de mensajería utilizados como herramientas de gestión de Logs, sirviendo como comunicación para las redes de sensores inalámbricos, los servidores de base de datos y la gestión de eventos.
- Implementar una herramienta que permita la integración de diversas fuentes de datos como un solo conjunto de información.
- Implementar el control de la seguridad de acceso a la información ubicada en las diversas plataformas que forman parte de los servidores de IOTMACH.
- Controlar los privilegios de los usuarios y la mitigación de vulnerabilidades a la aplicación de IOTMACH SERVER, implementando permisos basados en roles.

2.4 Análisis y Diseño del prototipo

El prototipo se divide en cuatro capas para su funcionamiento, tomando en cuenta lo descrito en la **Gráfico 1**: en la primera y segunda capa se abarca sobre la gestión de Logs y el almacenamiento de los datos en servidores de BDD, la tercer capa comprende la integración y virtualización de los datos almacenados en los servidores de BDD, y la cuarta capa se encuentra conformada por la capa de seguridad de acceso a la información. Para ello se trabaja en aplicaciones que cumplan con las necesidades de las capas, dichas aplicaciones son el sistema de puente de protocolos (capa 1 y 2), implementación de Herramienta de integración de datos (capa 3) y el sistema de autenticación (capa 4)

Antes de continuar, hay que destacar que la organización y desarrollo del prototipo será manejada por la metodología Kanban, en el cual en el **Anexo 1** se puede observar la planificación realizada para el desarrollo de las aplicaciones. En dicha planificación se encuentran la organización de las tareas realizadas tanto para la creación del puente de protocolos, el sistema de autenticación y módulo de seguridad, y la implementación de la herramienta de integración de datos; todo esto aplicando los principios del Kanban lo cual implica planificar tareas que cumplan diferentes etapas de desarrollo.

2.4.1 *Sistema de Puente de protocolos***.** El sistema de Puente de protocolos denominado Bridge IOTMACH, abarca el proceso de implementación de un adaptador de protocolos y de herramientas de gestión Logs, para ello se implementan los bróker de mensajería Mosquitto y Kafka respectivamente.

Al utilizar protocolos de mensajería distintos no existe comunicación directa entre ellos por lo que mediante un middleware desarrollado en Node.JS comunica el protocolo MQTT con Kafka. Bridge IOTMACH comprende a su vez, con el proceso de control de los Logs que van llegando al bróker Kafka y a su vez almacenarlos en las BDD relacionales como no relacionales.

2.4.1.1 *Matriz de requisitos*. Aquí se muestran los requisitos para la realización de Bridge IOTMACH.

- *Adaptador de protocolos.* Se implementa un bróker Mosquitto, el mismo que trabaja con protocolos MQTT, para que los dispositivos de la WSN tengan conexión con el CPD.

• *Gestión de Logs.* Se implementa un bróker de Apache Kafka el cual trabaja como un gestor de Logs generados por los dispositivos WSN. Para receptar dichos Logs, deben atravesar por el adaptador de protocolos ya que es la única forma que tienen de conectarse directamente con el CPD.

Tabla 1: Requisitos de servicios de Bridge IOTMACH.

No	Tipo	Descripción	Restricciones	Observaciones	Prioridad
R1	Implementar bróker Mosquitto	El bróker MQTT funcionara como un adaptador de protocolos	Se requiere tener instalado en un servidor el bróker Mosquitto.	Mosquitto funcionara como adaptador de protocolos, lo cual facilita la comunicación entre la WSN y el CPD	Alta
R2	Implementar bróker Kafka	El bróker Kafka funcionara como un gestor de Logs	Se requiere tener instalado en un servidor el bróker Kafka.	El bróker Kafka sirve para gestionar los mensajes generados por el adaptador de protocolos	Alta

Fuente: Elaboración Propia

• *Middleware de comunicación***.** El middleware de comunicación trabaja con el adaptador de protocolos y el gestor de Logs, ya que realiza la comunicación entre los protocolos de mensajería, administra los clientes de los brókeres y gestiona los tópicos en común con los que se van a transmitir los mensajes.

○ *Gestionar Tópicos.* Aquí se gestionaran los tópicos comunes con los que se comunicaran los clientes de los brókeres Mosquitto y Apache Kafka.

Tabla 2: Requisitos de Gestión de Tópicos.

No	Tipo	Descripción	Restricciones	Observaciones	Prioridad
R3	Crear	Se crean los tópicos en el bróker de Kafka.	Almacenan los tópicos dentro de la base de datos del sistema.	Los tópicos se utilizan tanto por el bróker Kafka como en el bróker MQTT	Alta
R4	Listar	Se listan los tópicos.	Se listan todos los tópicos de la BDD	Los tópicos listados se usan para todos los brókeres.	Media

R5	Eliminar	Se elimina el tópico de los de la BDD del sistema.	Deben estar listados los tópicos.	Se desuscribe de todos los clientes.	Media

Fuente: Elaboración Propia

o *Gestionar Clientes Adaptador de Protocolos.* Se administra a los clientes del bróker Mosquitto, ya que se podrá conectar a diversos brókeres de mensajería.

Tabla 3: Requisitos de Gestión de Clientes de adaptador de protocolos

No	Tipo	Descripción	Restricciones	Observaciones	Prioridad
R6	Crear	Conectar un cliente con el bróker del adaptador de protocolos.	Se crea conexión con bróker activo.	El sistema permitirá conectar un cliente MQTT con las respectivas credenciales de conexión.	Alta
R7	Listar	Listar conexiones de los clientes de adaptador de protocolos.	Se listas las conexiones activas e inactivas.	Listar conexiones de los bróker MQTT.	Media
R8	Eliminar	Eliminar las conexiones.	Se elimina la conexión de la BDD y se finaliza la conexión con el bróker.	Se elimina la conexión del sistema.	Media
R9	Suscribir	Suscribir tópico con el cliente	Solo se suscribe a los tópicos que se manejan en la gestión de tópicos.	Se suscribe los tópicos necesarios en cada cliente	Alta
R10	Desuscribir	Desuscribir tópico con el cliente	Se desuscribe de los clientes	Se elimina el registro de la BDD	Alta
R11	Visualizar	Se muestran los mensajes que llegan a los tópicos suscritos por los brókeres	Se mostraran los mensajes que llegan a un solo tópico.	El sistema muestra los mensajes que llegan al tópico suscrito de cada conexión MQTT establecida	Media

Fuente: Elaboración Propia

o *Enviar mensajes desde MQTT hacia Kafka.* Se elabora una función que consuma los de datos que llegan a un bróker Mosquitto y los produce en el bróker de Kafka.

Tabla 4: Requisitos del diseño del suscriptor MQTT productor Kafka

No	Tipo	Descripción	Restricciones	Observaciones	Prioridad
R12	Suscriptor	Se tiene un suscriptor de adaptador de protocolos que produce mensajes en Kafka.	Se tiene suscriptor que esté recibiendo los mensajes del adaptador de protocolos y produzca dicho mensaje en Kafka.	Cada cliente de adaptador de protocolos está suscrito a tópicos y a la vez produciendo el mensaje recibido en Kafka	Alta

Fuente: Elaboración Propia

○ *Enviar mensajes desde Kafka hacia MQTT.* Se elabora una función que consuma los datos que llegan al bróker Kafka y los publica en un bróker Mosquitto.

Tabla 5: Requisitos de Diseño de función del consumidor Kafka publicador MQTT

No	Tipo	Descripción	Restricciones	Observaciones	Prioridad
R13	Consumidor	Se tiene un consumidor de Kafka que publica mensajes en el Adaptador de Protocolos.	Se tiene consumidor que esté recibiendo los mensajes del Kafka y publique en MQTT.	Habrá un consumidor en Kafka encargado de publicar los mensajes en MQTT	Alta

Fuente: Elaboración Propia

• *Middleware consumidor de datos.* El middleware consumidor de datos abarca al consumo de los datos que se publican en el bróker de Kafka, dicho consumo se utilizara para almacenar los datos ya sea en bases de datos relacionales como bases de datos no relacionales.

○ *Consumidores para bases de datos relacionales.* Se diseña un consumidor del bróker de Kafka que se encargue de almacenar los datos consumidos en una base de datos relacional.

Tabla 6: Requisitos de diseño del consumidor para bases de datos relacional

No	Tipo	Descripción	Restricciones	Observaciones	Prioridad
R14	Consumidor	Se tiene consumidores que almacenen los mensajes en las BDD Relacionales	Se tiene un consumidor que recepte mensajes en el bróker Kafka para	Se configuran parámetros de conexión a la BDD y el tópico de donde se consumen los	Alta

			almacenar en una BDD Relacional.	mensajes en el bróker Kafka	

Fuente: Elaboración Propia

o *Consumidores para bases de datos no relacionales.* Se diseña un consumidor del bróker de Kafka que se encargue de almacenar los datos consumidos en una base de datos no relacional

Tabla 7: Requisitos del diseño de consumidor para bases de datos no relacionales

No	Tipo	Descripción	Restricciones	Observaciones	Prioridad
R15	Consumidor	Se tiene consumidores que almacenen los mensajes en las BDD Relacionales	Se tiene un consumidor que recepte mensajes en el bróker Kafka para almacenar en una BDD Relacional.	Se configuran parámetros de conexión a la BDD y el tópico de donde se consumen los mensajes en el bróker Kafka	Alta

Fuente: Elaboración Propia

o *Gestión de conexión de consumidores.* Se administra las conexiones con los consumidores diseñados que permitan darle la versatilidad de necesaria, es decir, le indiquen al consumidor diseñado cual es el tópico que va a tomar la información, cuales son las credenciales de la base de datos a conectar, cuales son los parámetros que se van a almacenar en la base de datos y también que le indique las credenciales del adaptador de protocolos.

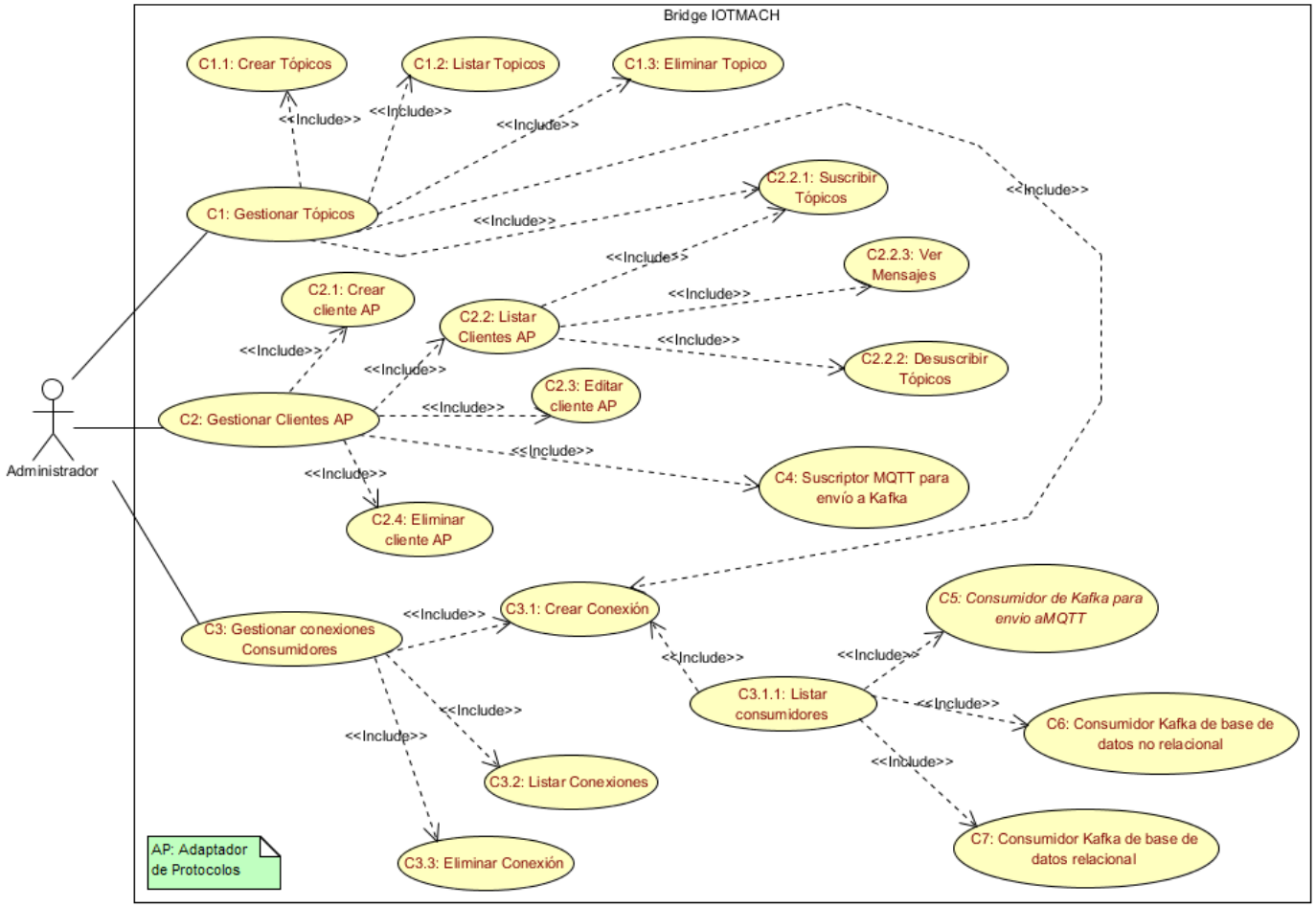
Tabla 8: Requisitos de gestión de conexión de consumidores

No	Tipo	Descripción	Restricciones	Observaciones	Prioridad
R16	Listar	Se listan los consumidores.	Se muestran el nombre de los consumidores.	Se elige un consumidor con cual trabajar	Alta
R17	Crear	Se crea conexión entre la BDD y el consumidor	Se guardan los parámetros de conexión y el tópico.	Se configuran parámetros de conexión con la BDD y el tópico.	Alta
R18	Listar	Se listan los consumidores con parámetros de conexión	Se muestran los nombres de los elementos	Se listaran los consumidores	Media
R19	Eliminar	Se eliminan las configuraciones del consumidor	Se desuscribe el consumidor con el tópico.	Se podrá eliminar las conexiones	Media

Fuente: Elaboración Propia

2.4.1.2 *Modelos de casos de uso*. En el Gráfico 4, se representan el comportamiento del sistema Bridge IOTMACH a través de casos de uso.

Gráfico 4: Diagrama de Casos de Uso del sistema Bridge IOTMACH



Fuente: Elaboración Propia

2.4.1.3 *Modelado de datos***.** Bridge IOTMACH trabaja con un sistema de base de datos no relacional, lo cual implica trabajar con datos de tipo JSON, cuyas colecciones serán las siguientes:

Tabla 9: Modelo de BDD no relacional productores

Colección	Productores
Descripción	Colección encargada de registrar a los clientes de adaptador de protocolos.
Tipo de BD	No Relacional
Parámetros	{ "nombre": String, "tipo": String, "host": String, "usuario": String, "pass": String, "topicos": [], "cliente" : String }

Fuente: Elaboración Propia

Tabla 10: Modelo de BDD no relacional topicos_logs

Colección	topicos_logs
Descripción	Gestiona los tópicos en los clientes de los brókeres de mensajería.
Tipo de BD	No Relacional
Parámetros	{ "nombre_topico": String }

Fuente: Elaboración Propia

Tabla 11: Modelo de BDD no relacional consumers

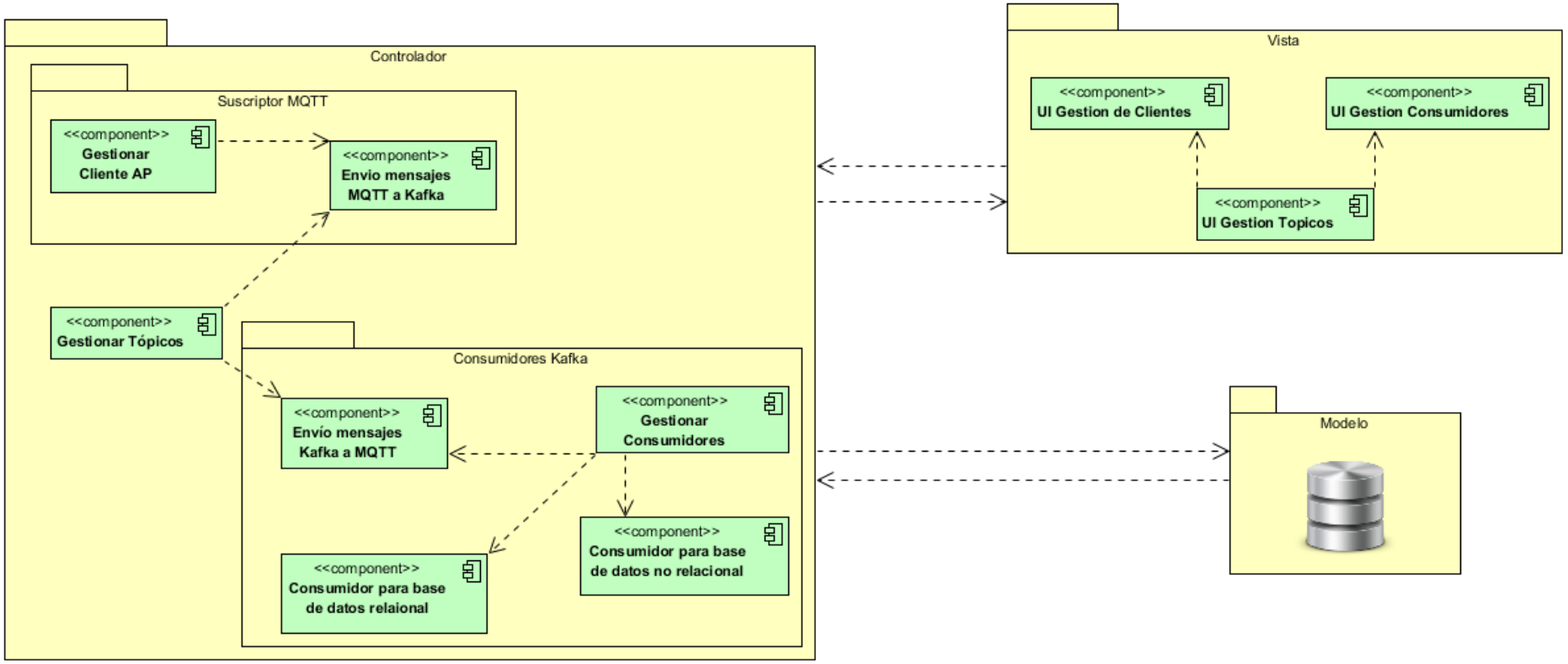
Colección	Colección Consumidores
Descripción	Colección encargada de la gestión de los consumidores de los mensajes del bróker Kafka.
Tipo de BD	No Relacional
Parámetros	{ "funcion": String, "tipo": String, "topico": String }

Fuente: Elaboración Propia

Bridge IOTMACH a través de su sistema los mensajes que circulan manejan los esquemas estipulados en el **ANEXO 2**.

2.4.1.4 *Diagrama de componentes*. En el Gráfico 5 se aprecia cómo interactúan los componentes del Bridge IOTMACH.

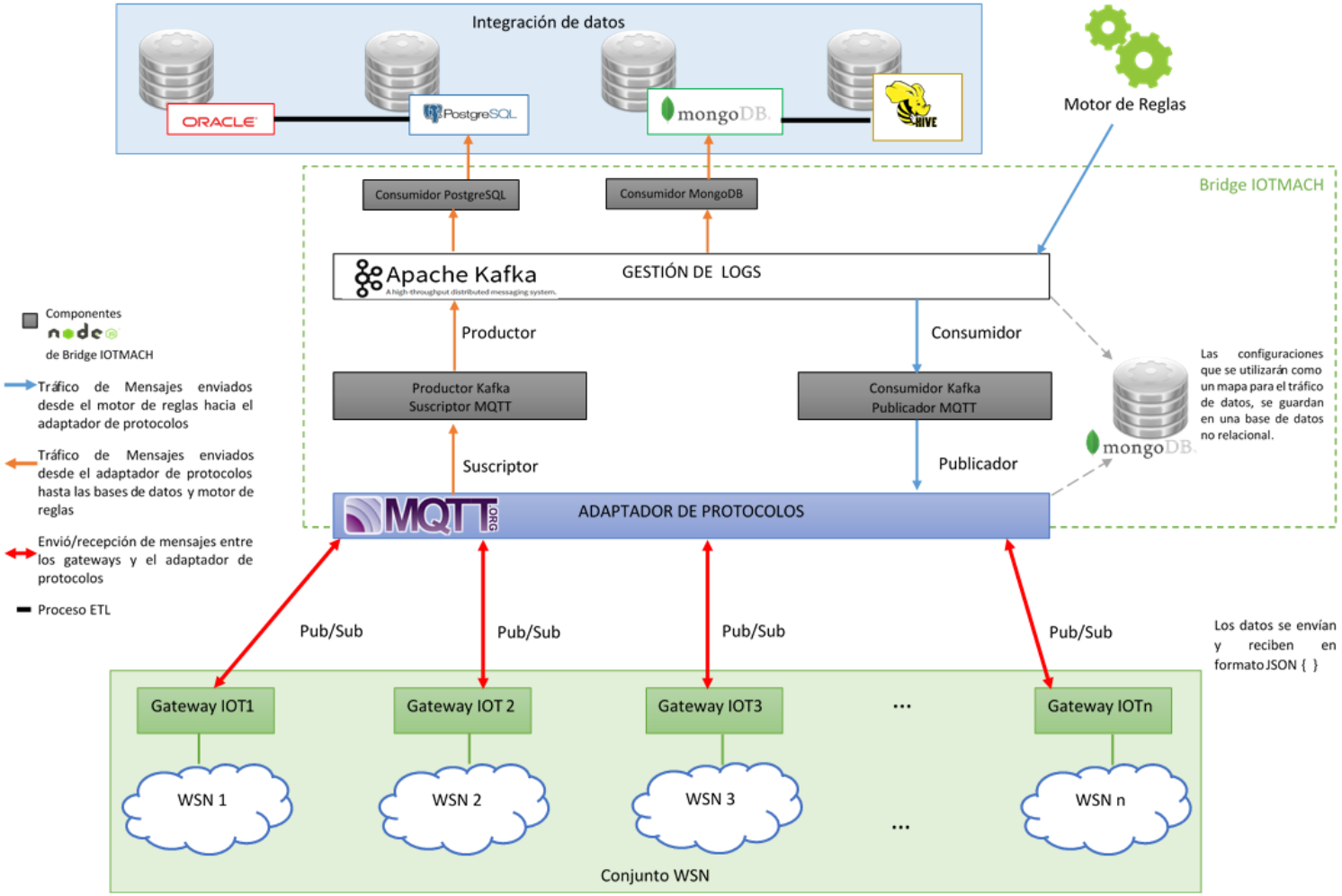
Gráfico 5: Diagrama de componentes – Bridge IOTMACH



Fuente: Elaboración Propia

2.4.1.5 *Diagrama arquitectónico.*

Gráfico 6: Diagrama Arquitectónico - Bridge IOTMACH

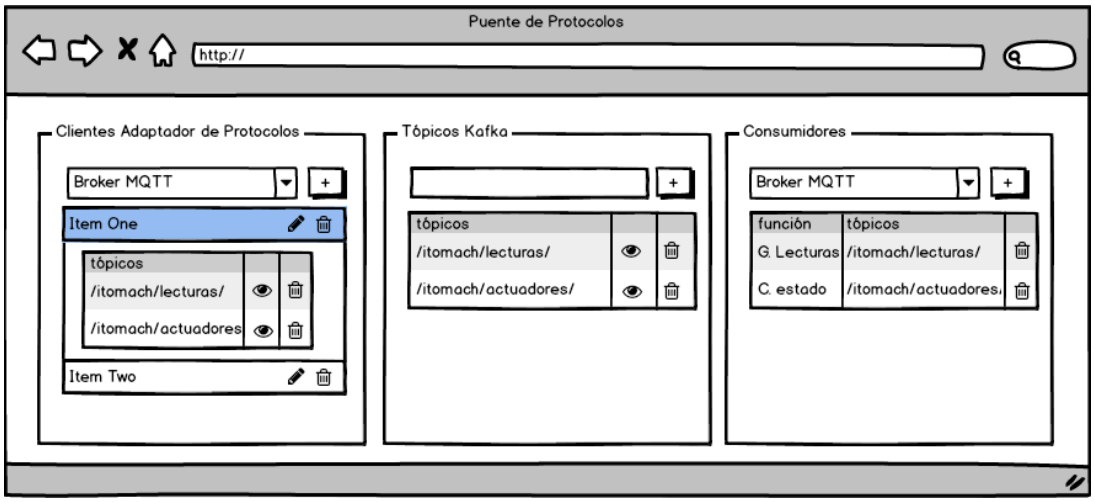


Fuente: Elaboración Propia

2.4.1.6 *Diseño de interfaces*. Se realiza un bosquejo de las interfaces que el usuario va a manipular, las cuales se muestran a continuación:

- Interfaz principal

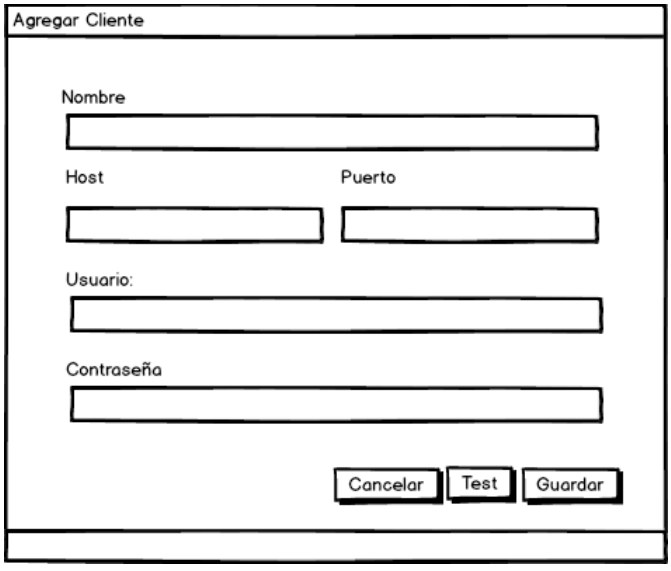
Gráfico 7: Interfaz Principal de Bridge IOTMACH



Fuente: Elaboración Propia

- Interfaz cliente adaptador de protocolos

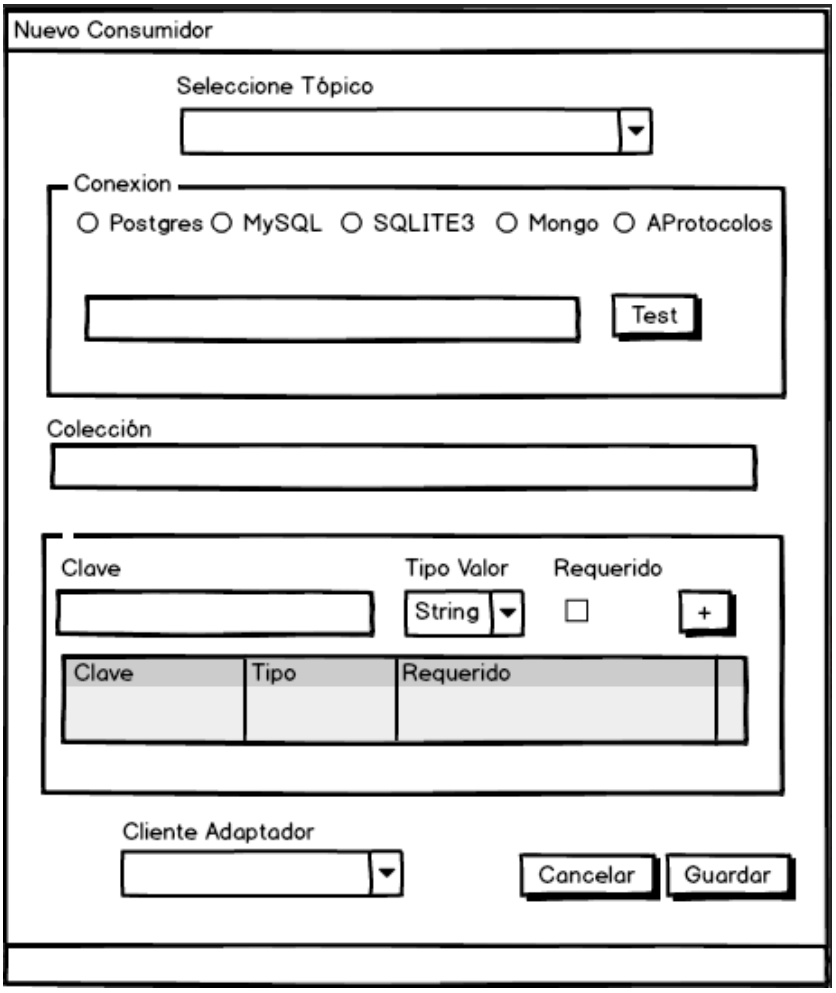
Gráfico 8: Formulario de creación de protocolos



Fuente: Elaboración Propia

• Interfaz de gestión de consumidor

Gráfico 9: Formulario de Gestión de consumidores



Fuente: Elaboración Propia

2.4.2 *Módulo de autenticación y seguridad***.** El módulo de autenticación y seguridad para IOTMACH, contribuye con el control de la seguridad de acceso a la información, ya que proporciona un manejo de usuarios que solo tienen acceso a la información que le compete.

La autenticación permite que varias aplicaciones puedan ser accedidas solo por los usuarios que cuentan con los permisos necesarios para acceder a la información que estas contienen o van a manipular. Cabe recalcar que dentro de cada aplicación existen procesos propios que el framework web utilizado proporciona para tratar las vulnerabilidades que pueden existir.

2.4.2.1 *Matriz de requisitos***.** Los siguientes requisitos cumplirán con el desarrollo del módulo de autenticación y seguridad para IOTMACH.

- *Gestionar empresas*. Registra los datos de las empresas que deseen formar parte de IOTMACH

Tabla 12: Requisitos de Empresa

No	Tipo	Descripción	Restricciones	Observaciones	Prioridad
R1	Inserción	Insertar datos de la empresa en las BDD correspondientes	RUC debe ser único. Registro en LDAP, y base de datos Postgres	Ingreso de datos por formulario	Alta
R2	Edición	Modificar datos de empresa en las BDD	El campo RUC, no será modificable	Solo puede cambiar los datos el administrador	Alta

Fuente: Elaboración Propia

- *Gestionar usuarios*. Se crean los usuarios que tendrán acceso a las aplicaciones de IOTMACH

Tabla 13: Requisitos de Usuarios

No	Tipo	Descripción	Restricciones	Observaciones	Prioridad
R3	Inserción	Insertar datos de los usuarios en las BDD.	Asociar roles de las aplicaciones al usuario.	Se almacenarán los datos según la BDD.	Alta
R4	Edición	Modificar datos de los usuarios en las bases de datos	El administrador edita el rol. Un usuario, cambia sus datos.	Se realizan cambios en todos los datos.	Media
R5	Listado	Listado de datos de los usuarios del gestor de LDAP	Se lista la información de los usuarios, excepto de la contraseña.	Se muestran los datos almacenados en las BDD.	Baja
R6	Eliminación	Eliminar usuario totalmente del sistema.	Solo un usuario administrador podrá eliminar.	Se elimina la información de las BDD.	Baja

Fuente: Elaboración Propia

- *Manejar roles y permisos*. Los roles y permisos se generan a partir de los modelos de datos detallados en IOTMACH Server, ya que se controlan a través del framework Django, el cual brinda funciones para el control de permisos de usuario a través de roles.

Tabla 14: Requisitos Roles y Permisos

No	Tipo	Descripción	Restricciones	Observaciones	Prioridad
R7	Creación	Crear los roles que tendrá el usuario en cada aplicación	Generar los roles por aplicación. Asociar los permisos a los roles.	Se crean los roles de IOTMACH Server	Alta
R8	Muestra	Muestra los roles existentes	Se mostrarán los roles según la aplicación seleccionada. Solo el usuario administrador podrá asignar los roles.	Se listaran los roles en el formulario de creación de usuario	Alta

Fuente: Elaboración Propia

• *Crear sesiones y cookies***.** Manejar las sesiones que genere el sistema dentro de las aplicaciones y controlar mediante cookies detectar el usuario logeado en el navegador

Tabla 15: Requisitos de sesiones y cookies

No	Tipo	Descripción	Restricciones	Observaciones	Prioridad
R9	Creación	Un usuario genera una sesión al iniciar el sistema	Con la sesión iniciada se generan cookies para conocer si el usuario ha iniciado sesión.	Cualquier usuario con credenciales puede iniciar al sistema.	Alta
R10	Generar	Se generan cookies encriptadas según la sesión.	Se generan cookies que se consumen en las aplicaciones.	Las cookies sirven para que las aplicaciones sepan que usuarios han iniciado sesión.	Alta

Fuente: Elaboración Propia

• Crear menús. Se manejan menús en base a los permisos que se generan en los modelos de bases de datos de la aplicación IOTMACH Server.

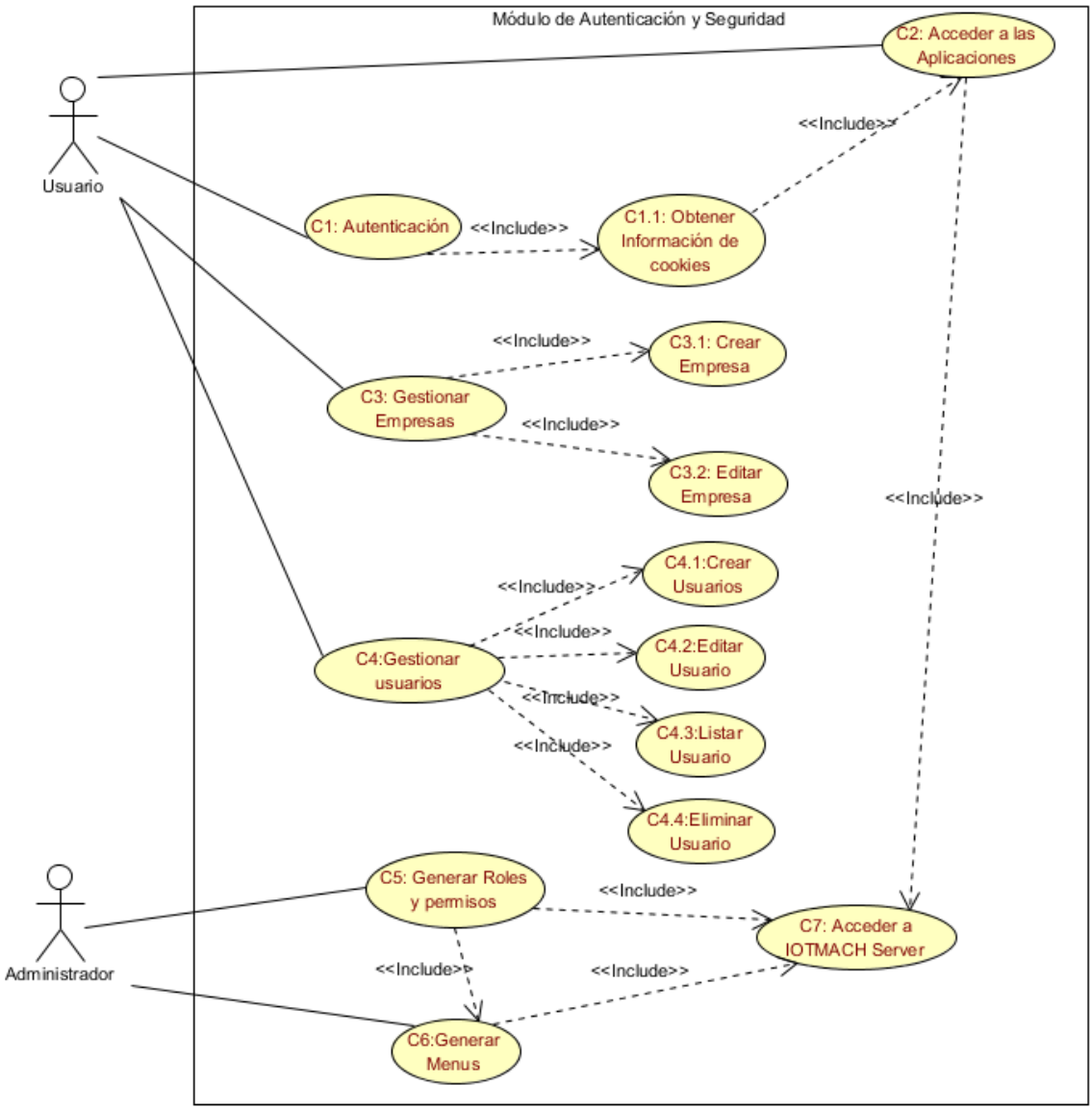
Tabla 16: Requisitos de creación de menús

No	Tipo	Descripción	Restricciones	Observaciones	Prioridad
R11	Creación	Se generan menús basados en los permisos de la aplicación.	Solo un permiso tendrá un menú.	Los menús serán basados en roles.	Alta

Fuente: Elaboración Propia

2.4.2.2 *Modelos de casos de uso***.** En el Gráfico 10, se representan el comportamiento del módulo de autenticación y seguridad a través de casos de uso

Gráfico 10: Diagrama de Casos de uso de módulo de autenticación y seguridad



Fuente: Elaboración Propia

2.4.2.3 *Modelado de datos***.** Los siguientes modelos de datos forman parte del módulo de autenticación y seguridad, los mismos que servirán tanto para la autenticación como para las distintas aplicaciones que hagan consumo de la base de datos relacional.

- Gestión de empresa

Tabla 17: Modelo de Datos de Tabla Empresa

Tabla	Empresa	
Descripción	Datos de la empresa que va a hacer uso de los servicios que la plataforma de IOTMACH ofrece.	
Parámetros	**Nombre**	**Tipo**
	emp_id	Integer
	emp_nom_comercial	String
	emp_nom_juridico	String
	emp_grupo	String
	emp_direccion	String
	emp_email	String
	emp_web	String
	emp_descripcion	String
	emp_dominio	String
	emp_ruc	String

Fuente: Elaboración Propia

- *Gestión de usuarios***.**

Tabla 18: Modelo de Datos de Tabla Usuarios

Tabla	Usuario	
Descripción	Datos del usuario.	
Parámetros	**Nombre**	**Tipo**
	Id	Integer
	first_name	String
	last_name	String
	email	String
	username	String
	password	String
	date_joined	Date
	last_login	Date

Fuente: Elaboración Propia

Las datos de permisos, roles y menús para la aplicación de IOTMACH Server, se detallan a continuación.

➢ *Menús***.**

Tabla 19: Modelo de Datos de Tabla Menús

Tabla	Menú	
Descripción	Datos de los menús que se generaran para la aplicación IOTMACH Server	
Parámetros	**Nombre**	**Tipo**
	men_id	Integer
	men_nombre	String
	men_estado	Boolean
	men_icono	String
	men_ruta	String
	men_tipo	String
	men_id_padre	Integer
	men_estado_url	Boolean
	men_plantilla	String
	men_id_active	Integer
	permisos	Integer

Fuente: Elaboración Propia

➢ *Permisos***.**

Tabla 20: Modelo de Datos de Tabla Permisos

Tabla	Permission	
Descripción	Datos de los permisos que maneja el framework en base a sus modelos predefinidos	
Parámetros	**Nombre**	**Tipo**
	name	Integer
	content_type	Integer
	codename	String
	objects	Object

Fuente: Elaboración Propia

➢ *Roles***.**

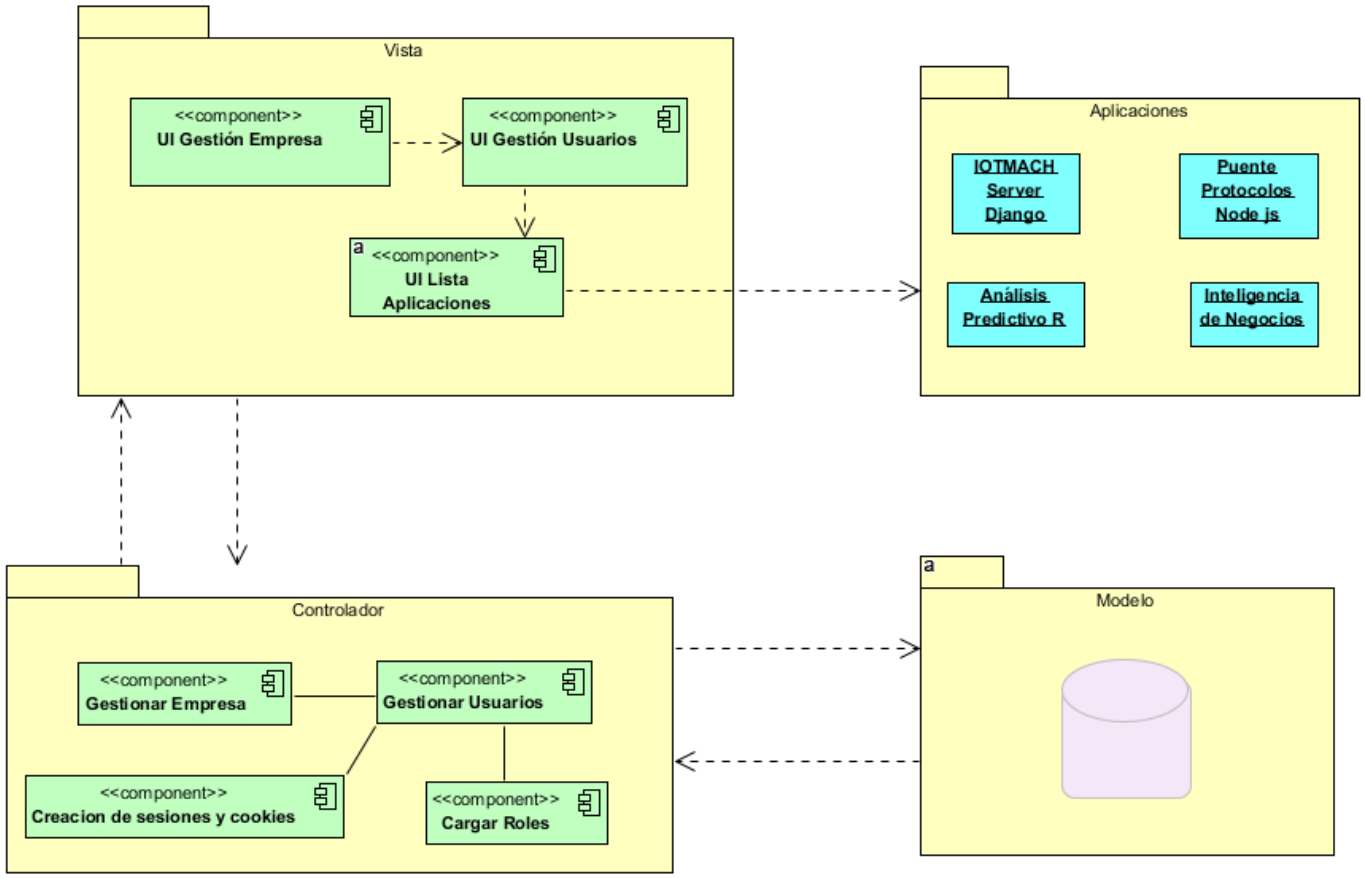
Tabla 21: Modelo de Datos de Tabla Grupos

Tabla	Group	
Descripción	Datos de los grupos que maneja el framework Django en base a sus modelos predefinidos	
Parámetros	**Nombre**	**Tipo**
	name	Integer

Fuente: Elaboración Propia

2.4.2.4 *Diagrama de componentes*.

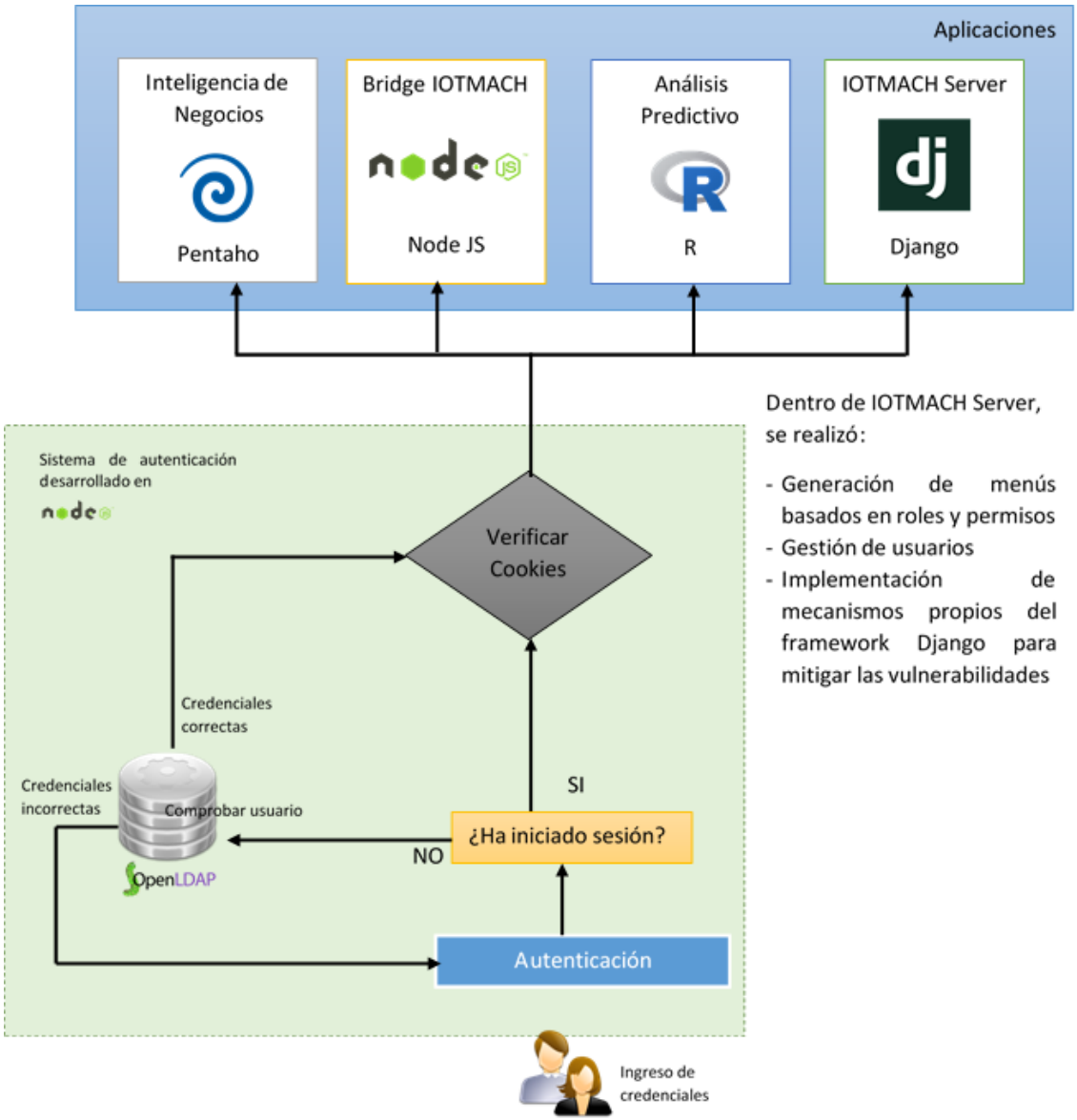
Gráfico 11: Diagrama de Componentes de sistema de autenticación



Fuente: Elaboración Propia

2.4.2.5 *Diagrama arquitectónico*. El diagrama arquitectónico del módulo de autorización y seguridad de IOTMACH, representado en el Gráfico 12, muestra cómo se encuentra organizado sus componentes, y cuales son es su funcionamiento de manera esquematizada.

Gráfico 12: Diagrama Arquitectónico - Sistema de Autenticación

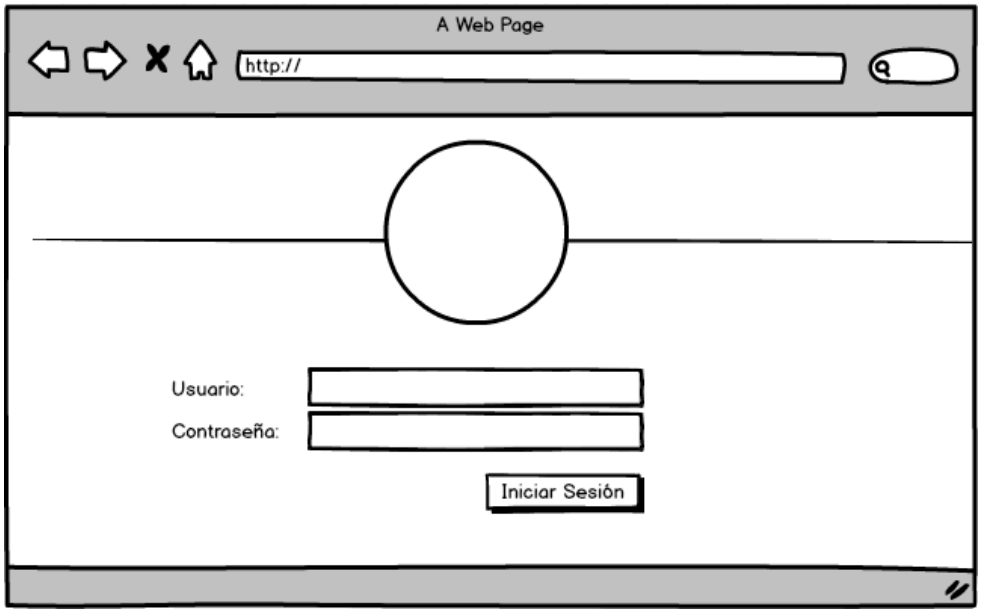


Fuente: Elaboración Propia

2.4.2.6 *Diseño de interfaces***.** Los siguientes bosquejos de las pantallas a utilizarse en el desarrollo en el módulo de autenticación y seguridad son:

➢ Interfaz de Inicio de Sesión

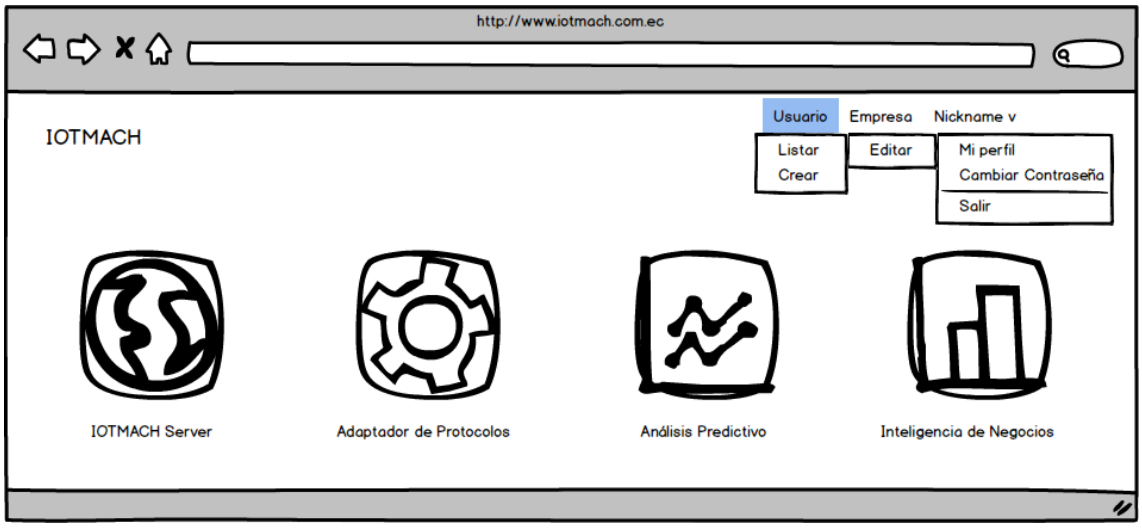
Gráfico 13: Formulario de Inicio de Sesión



Fuente: Elaboración Propia

➢ Interfaz Principal

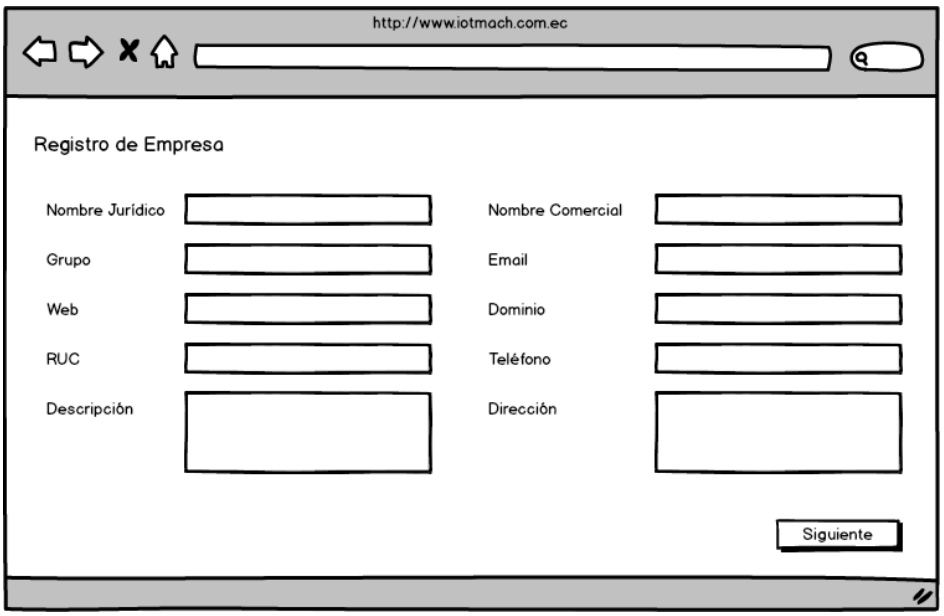
Gráfico 14: Interfaz Principal de Sistema de Autenticación



Fuente: Elaboración Propia

➢ Interfaz Registro de Empresa

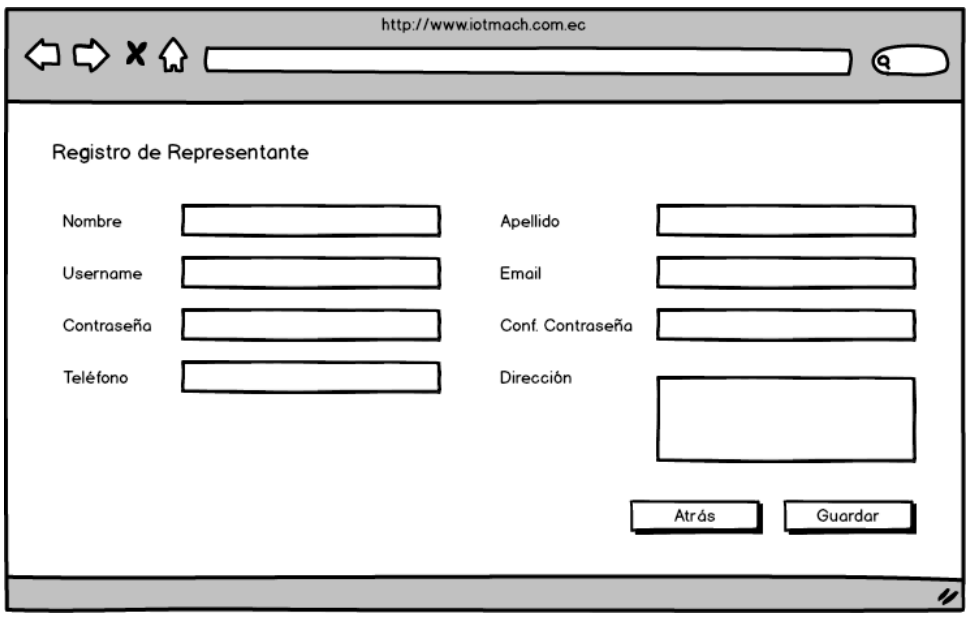
Gráfico 15: Formulario de Empresa



Fuente: Elaboración Propia

➢ Interfaz Registro de Representante

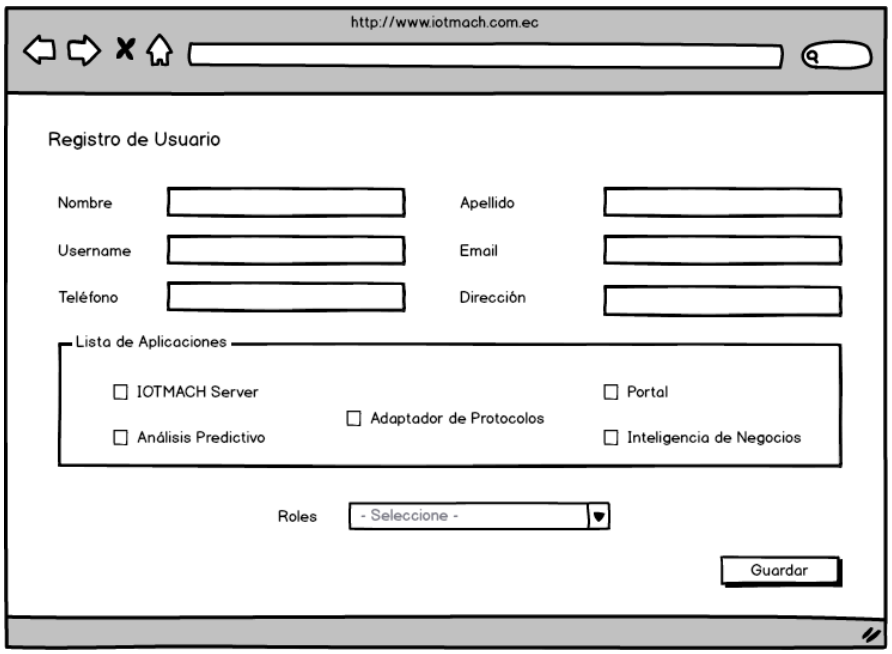
Gráfico 16: Formulario de Representante



Fuente: Elaboración Propia

➢ Interfaz Registro de Usuario

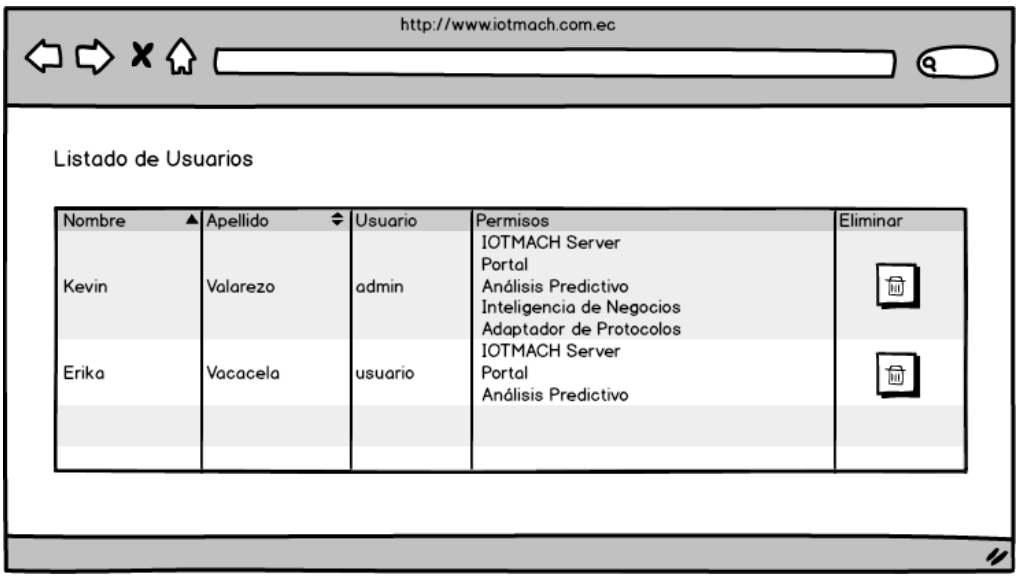
Gráfico 17: Formulario de Usuarios



Fuente: Elaboración Propia

➢ Interfaz de Lista de Usuarios

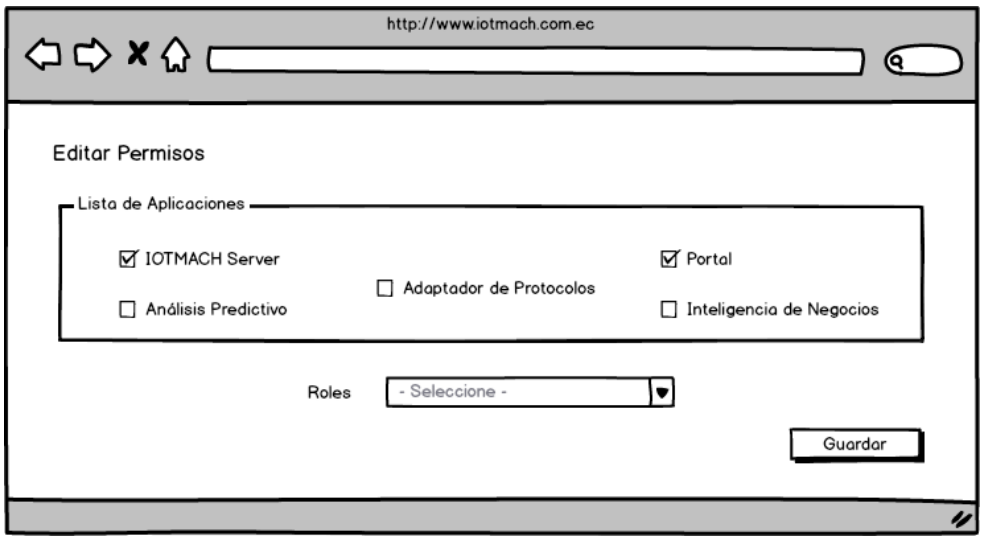
Gráfico 18: Lista de Usuarios



Fuente: Elaboración Propia

➢ Interfaz de Editar Permisos

Gráfico 19: Formulario de edición de Permisos



Fuente: Elaboración Propia

2.4.3 *Implementación de herramienta de integración de datos***.** Para la integración de datos es necesario utilizar una herramienta que permita realizar dicha integración, Denodo Plataform ofrece las características necesarias para integrar y virtualizar datos provenientes de diferentes fuentes de información.

2.4.3.1 *Requisitos Hardware*. Para la instalación de Denodo se requieren tener un equipo con las siguientes características:

Tabla 22: Requisitos Hardware Denodo

IOTMACH	Requisitos del sistema
Procesador	Pentium IV 2,4 GHz
Espacio libre en disco	300 MB
RAM	1 GB

Fuente: Manual de instalación de Denodo

2.4.3.2 *Requisitos Software*. Para instalar Denodo se debe contar con los siguientes requerimientos de software para su correcto funcionamiento:

Tabla 23: Requisitos de Software

IOTMACH	Requisitos del sistema
Sistema Operativo	Windows 2000 o superior
Otras dependencias	Java 2 Runtime Environment Standard Edition (J2SE 1.4, J2SE 1.5 o J2SE 1.6)

Fuente: Manual de instalación de Denodo

2.5 Desarrollo e implementación del prototipo.

Al igual que en el análisis y diseño, el trabajo se encuentra divididos en tres componentes que son el Bridge IOTMACH, el módulo de autenticación y seguridad, y la implementación de la herramienta de integración de datos.

2.5.1 *Sistema de Puente de protocolos*. Para el desarrollo de Bridge IOTMACH, se han requerido principalmente de los bróker de mensajería Mosquitto y Kafka, los cuales servirán como el adaptador de protocolos y el gestor de Logs respectivamente.

2.5.1.1 *Preparación de entorno***.** Para el correcto funcionamiento del middleware de comunicación de protocolos es necesario tener configurado bróker Mosquitto, bróker Kafka, MongoDB y Node.JS.

- *Instalación de Bróker Mosquitto – MQTT.* Es necesario instalar un bróker Mosquitto, que a su vez es un bróker de mensajería que trabaja con el protocolo MQTT, el mismo que será utilizado como adaptador de protocolos y será el comunicador directo con la WSN. Para la instalación se siguieron los pasos registrados en el **ANEXO 3.**

- *Instalación de Apache Kafka***.** Es necesario instalar un bróker Kafka, el mismo que será utilizado como un gestor de Logs de los mensajes que llegan del adaptador de protocolos y del motor de reglas. Para la instalación se siguieron los pasos registrados en el **ANEXO 4.**

- *Instalación de Mongo DB.* Bridge IOTMACH está diseñado para trabajar con una base de datos No Relacional, es ahí la importancia de utilizar Mongo DB. Para la instalación se siguieron los pasos registrados en el **ANEXO 5.**

• *Instalación de Node.JS.* Bridge IOTMACH al estar diseñado para trabajar con brókeres de mensajería, bases de datos no relacionales, pues la solución más concreta a eso es la utilización de Node.JS. Para la instalación se siguieron los pasos registrados en el **ANEXO 6.**

2.5.1.2 *Código Fuente***.** El código fuente es parte vital para cualquier sistema, ya que sin este simplemente no habría sistema. A continuación se coloca el código JavaScript-Node.JS utilizado.

• *Crear tópico*

Tabla 24: Código de función de creación de tópicos

Función	Creación de tópicos
Descripción	Esta función permite crear los tópicos con los que trabajara el bróker Kafka a través de la función `.createTopics()`, a su vez registra los tópicos en la base de datos no relacional, para su utilización en los clientes de adaptador de protocolos y consumidores, con la función `.insertOne()` del driver de Mongo para Node.JS
Lenguaje	JavaScript

Código:

```
function crearTopicos(topico, db, assert){
   var topic=String(topico).replace(/[/]/g,"_");
   console.log(topic);
   var col=db.collection('topicos_logs');
   producer_kafka.createTopics([topic], true, function (err, data) {
      if (err){console.log(err); return}
      else console.log('se creo topico %s', topic);
      col.insertOne({nombre_topico: topico },function(err, result) {
          if(err) {console.log(err); return}
          assert.equal(1, result.insertedCount);
          console.log("Se inserto un nuevo documento en la coleccion
de la gestion de logs");
         db.close();
       });
   });
}
```

Fuente: Elaboración Propia

• *Gestión de Consumidores*

Tabla 25: Código función de creación de consumidores

Función	Crear consumidor
Descripción	Permite crear el registro de conexión que los consumidores utilizaran para gestionar los logs, si es para la función de lecturas se utilizara mongo como base de datos no relacional, si es para guardar la configuración de los motes se utilizara SQL, ya que se tiene una base de datos relacional. Si el consumidor es para ejecutar una acción de los actuadores, se utilizara la opción MQTT puesto que es el único adaptador de protocolos utilizado.
Lenguaje	JavaScript

Código:

```
exports.create = function(data, cb) {
    var consumer = {}
    if(data.option == "Mongo" || data.option == "SQL"){
        consumer = {
            category: data.category,
            function_des: data.function_des,
            topic: data.topic,
            connector: data.connector,
            conexionStr: data.conexionStr
        }
    }
    if(data.option == 'Mongo'){
        var parametros = []
        consumer['collection'] = data.collection
        consumer['parameters'] = JSON.parse(data.parameters);
    }else if(data.option == 'MQTT'){
        consumer = {
            category: data.category,
            function_des: data.function_des,
            topic: data.topic,
            connector: data.connector,
            broker_id: data.broker_id,
            cliente_id: data.cliente_id,
            broker_nombre: data.broker_nombre
        }
    }
    gPMongoClient.connect(url_db, function(err, db){
        var collection = db.collection(COLLECTION);
        collection.insert(consumer, function(err, docs) {
            if (err) return cb(err);
            db.close();
            cb(null, docs.insertedCount);
        })
    });
}
```

Fuente: Elaboración Propia

• *Iniciar clientes de adaptador de protocolos*

Tabla 26: Código Función inicio de clientes MQTT

Función	Iniciar clientes de adaptador de protocolos
Descripción	Esta función permite iniciar el funcionamiento de los clientes MQTT, cuyos registros de conexión están en la base de datos, mediante la función `.on('connect')` se establece conexión con el bróker Mosquitto, la función `.suscribe()` suscribe el cliente con los tópicos y la función `.on('message')` está escuchando el arribo de mensajes nuevos al bróker en base a los tópicos suscritos
Lenguaje	JavaScript

Código:

```
function iniciar_cliente(url_broker, usuario, pass, nombre,
topicos){
   var mqtt_url = url.parse(process.env.CLOUDMQTT_URL ||
'mqtt://'+url_broker);
   var client_mqtt = mqtt.connect('mqtt://'+url_broker, {
       username: usuario,
       password: pass
   });

   client_mqtt.on('connect', function() {
      console.log('se conecto al cliente MQTT: '+nombre);
      clientes_id.push(client_mqtt);

      if(topicos!= undefined && topicos.length!=0){
         var url_topicos=[];
         if(Array.isArray(topicos)){
            for (i = 0; i < topicos.length; i++) {
               url_topicos.push(topicos[i]);
            }
         }else{
            url_topicos.push(topicos);
         }
         client_mqtt.subscribe(url_topicos, function() {
            console.log('suscrito a los topicos del broker:
'+nombre);
         });

      }else{
         console.log('no hay topicos suscritos')
      }
      client_mqtt.on('message', function(topic, message, packet) {
         topicos_logs.producir_mensaje(topic,message);
      });
   });
   return client_mqtt;
}
```

Fuente: Elaboración Propia

• *Consumidor para base de datos relacional.*

Tabla 27: Código de función de consumidor para base de datos relacional

Función	Guardar Configuración
Descripción	Esta función permite receptar mensajes en un tópico definido en el documento de la base de datos de mongo para almacenar información en una base de datos relacional destinada a guardar las configuraciones de los motes y sensores/actuadores, la función `.connect()` establece la conexión con la BDD relacional y a través de la función `.guardar_datos()` guardara la información que es consumida por un cliente de Kafka desde la función `.on('message')`
Lenguaje	JavaScript

Código:

```
function guardarConfiguracion(doc){
    var topics= [{topic: doc.topic.replace(/[/]/g,"_")}];
    var options={groupId: "Kafka_cliente_configuraciones",
                 autoCommit: true,
                 fetchMaxWaitMs: 1000
        }

    var consumer = new HighLevelConsumer(cliente_kafka, topics,
options);

   orm.connect(doc.conexionStr, function (err, db) {
      if (err) return console.log(err);

      var modelos_conf =
require('../../models/modelos_estructurados.js');
      consumer.on('message', function (message) {
         console.log(message.value);
         modelos_conf.guardar_datos(data,db);
      });

      consumer.on('error', function (err) {
          console.log('error consumer fijo de postgres', err);
      });
      console.log('se conectó el postgres');

   });
}
```

Fuente: Elaboración Propia

- *Consumidor para base de datos no relacional*

Tabla 28: Código de función de consumidor para base de datos no relacional

Función	Guardar Lecturas
Descripción	Esta función permite el almacenamiento de datos NOSQL, cuyas credenciales de conexión a la BDD y parámetros de almacenamiento se obtienen de los registros de mongo.
Lenguaje	JavaScript

Código:

```
function guardar_lecturas(doc){
    var mongoose= require('mongoose');
    var Schema = mongoose.Schema;
    mongoose.connect(doc.conexionStr, function (err) {
        if (err) return console.log(err);
        var esquema='';
        try{
            esquema= JSON.parse(doc.parameters);
        }catch (err){
            esquema = doc.parameters;
        }
        var lecturasSchema = new Schema(esquema);
        var Lecturas = mongoose.model(doc.collection,
lecturasSchema);
        var topics= [{topic: doc.topic.replace(/[/]/g,"_")}];
        var options={ groupId: "Kafka_cliente_mongo_lecturas",
autoCommit: true, fetchMaxWaitMs: 1000 }
        var consumer = new HighLevelConsumer(cliente_kafka, topics,
options);
        consumer_lecturas.push(consumer);
        consumer.on('message', function (message) {
            try{
                var data = JSON.parse(message.value);
                var lectura = new Lecturas(data);
                lectura.save(function(err){
                    if(err) return console.log('ocurrio un error en
el registro de lectura: '+err);
                    console.log('se inserto una nueva lectura');
                });
            }catch(err){
                console.log('ocurrio un error al tratar el mensaje
que arribo: '+err);
            }
        });
        consumer.on('error', function (err) {
            console.log('error consumer fijo de mongo', err);
        });
    });
}
```

Fuente: Elaboración Propia

• *Consumidor para enviar mensajes a MQTT.*

Tabla 29: Código de función de envió de mensajes hacia MQTT

Función	Cambiar estado actuador
Descripción	Esta función permite consumir mensajes de un tópico definido en un documento de la base de datos no relacional, y publicarlo en un bróker MQTT específico, para poder cambiar el estado de ejecución de un actuador de algún dispositivo.
Lenguaje	JavaScript

Código:

```
function cambiar_estado(doc){
    var topic= [{topic: doc.topic.replace(/[/]/g,"_")}];
    console.log('se conecto al cliente MQTT:'+doc.broker_nombre);
    var options={ groupId: "Kafka_cliente_mongo_actuadores",
autoCommit: true, fetchMaxWaitMs: 1000 }
    var consumer = new HighLevelConsumer(cliente_kafka, topic,
options);
    consumer.on('message', function (message) {
        try{
            var data = JSON.parse(message.value);
            producir_mensaje.publicar_mensajes(doc.cliente_id,
doc.topic, JSON.stringify(data));
        }catch(err){
            console.log('ocurrio un error al tratar el mensaje que
arribo desde el motor: '+err);
        }
    });

    consumer.on('error', function (err) {
        console.log('error consumer fijo de motor', err);
    });
}
```

Fuente: Elaboración Propia

2.5.2 *Módulo de autenticación y seguridad*. Para el desarrollo del módulo de autenticación y seguridad de IOTMACH se utilizó primordialmente el servidor de directorios de LDAP y como lenguaje de programación se empleó JavaScript para servidor, mejor conocido como Node.JS, a su vez dentro de la aplicación IOTMACH Server el lenguaje utilizado es Python junto con el framework Django.

2.5.2.1 *Preparación de entorno*. Antes de empezar el desarrollo se deben implementar los recursos y servidores a utilizar.

• *Instalación de LDAP*. La autenticación está siendo utilizada bajo los directorios que ofrece LDAP, por lo cual se requiere la instalación de un servidor LDAP. Para la instalación se siguieron los pasos registrados en el **ANEXO 7**.

• *Instalación de Node.JS*. Bridge IOTMACH al estar diseñado para trabajar con brókeres de mensajería, bases de datos no relacionales, pues la solución más concreta a eso es la utilización de Node.JS.

2.5.2.2 *Código Fuente*. A continuación se describe el código fuente más importante empleado para la autenticación y para la gestión del módulo ubicado en IOTMACH Server.

• Crear empresa

Tabla 30: Código función de crear empresa

Función	Crear empresa
Descripción	Esta función permite crear la empresa y el usuario administrador en LDAP, con la función `.bind()` se establece la conexión con el servidor LDAP, la función `.add()` permite crear grupos y usuarios dentro de LDAP. Para crear el mismo registro dentro de la base de datos relacional para que el sistema IOTMACH SERVER logre conocer cuál es la empresa y el usuario, se utiliza la función `.agregar_empresa()`
Lenguaje	Javascript

Código:

```
router.post("/add", function(req,res){
   fs.readFile('./uidGeneral.txt','utf8',function(err,data){
     UidGenerado = parseInt(data)+1;
     fs.writeFile('./uidGeneral.txt', UidGenerado, function(err)
{});
        console.log(UidGenerado);
        var variablesEmpresa ={
            objectClass: ['organizationalUnit','top'],
            description:req.body.txt_nombre_c
        };
        var variablesAdmin = {
            sn: req.body.txt_apellido_p,
            objectClass: ['inetOrgPerson',
'organizationalPerson','person','posixAccount','top'],
            uid: req.body.txt_usuario_p,
            homeDirectory: ('/home/'+req.body.txt_usuario_p),
```

```
            loginshell: '/bin/bash',
            uidNumber: UidGenerado,
            gidNumber: '1001',
            userPassword: req.body.txt_password_v,
            telephoneNumber: req.body.txt_telefono_m,
            mail: req.body.txt_correo_p,
            description: 'server,analisis,pentaho',
            title: 'titulo',
            street: req.body.txt_direccion_p,
            pager: req.body.txt_telefono_f
        };
        var client = ldap.createClient({
            url: 'ldap://192.168.0.42'
        });
client.bind('cn=iotmach,dc=iotmach,dc=com','12345678',function(err){
            if (err) return console.log(err);
client.add('ou='+req.body.txt_ruc+',dc=iotmach,dc=com',variablesEmpr
esa,function(err){
                if (err) return console.log(err);

client.add('cn='+req.body.txt_nombre_p+',ou='+req.body.txt_ruc+',dc=
iotmach,dc=com',variablesAdmin,function(err){
                    if (err) return console.log(err);
db.agregar_empresa(datos_empresa,datos_persona,datos_usuario,
req.body.txt_password_v);
                    client.unbind();
                    res.redirect('/');
                });
            });
        });
    });
});
```

Fuente: Elaboración Propia

- Registrar usuarios – IOTMACH SERVER

Tabla 31: Código Función de registro de usuario parte IOTMACH SERVER

Función	Guardar Usuarios
Descripción	Este método permite guardar el registro de personas y usuarios en la base de datos relacional y a su vez enviar los datos a través de sockets hacia el sistema de autenticación para que se almacene dentro de LDAP utilizando la función `.emit()`
Lenguaje	Python
Código:	

```
def guardarUsuarios(request):
    if request.user.has_perm('auth.add_user'):
        if not
User.objects.filter(username=request.POST['txtUsuario']).exists():
            persona=proy_persona.objects.create(
```

```
                per_nombres = request.POST['txtNombre'],
                per_apellidos = request.POST['txtApellido'],
                per_direccion = request.POST['txtDireccion'],
                per_telef_movil = request.POST['txtTlfono'],
                per_email = request.POST['txtCorreo'],
                per_telef_fijo = "0",
                emp_id = getEmpresa(request))

            vUsuario = User(username=request.POST['txtUsuario'],
                            first_name=request.POST['txtNombre'],
                            last_name=request.POST['txtApellido'],
                            email=request.POST['txtCorreo'],
                            id_persona=persona)

vUsuario.set_password(request.POST['txtUsuario']+'_'+request.POST['t
xtTlfono'])
            vUsuario.save()

vUsuario.groups.add(Group.objects.get(id=request.POST['listRoles']))
            lst_app = 'server'
            for i in request.POST:
                if i == 'pentaho' or i == 'puente' or i ==
'analisis':
                    lst_app+=','+request.POST[i]

            datos = {'cn': vUsuario.first_name,
                     'sn':vUsuario.last_name,
                     'direccion': persona.per_direccion,
                     'telefono':persona.per_telef_movil,
                     'mail':vUsuario.email,
                     'uuid':vUsuario.username,
                     'lst_app':lst_app,
                     'ou': getEmpresa(request).emp_ruc,
                     'uidNumber': vUsuario.pk
                     }

            socketIO = SocketIO('127.0.0.1', 6001)
            vAutenticacion = socketIO.define(autenticacion,
'/autenticacion')
            vAutenticacion.emit('guardar_usuario', datos)
            socketIO.wait(seconds=2)

            acciones("Guarda Usuario "+vUsuario.get_full_name(),
request.user)
            messages.add_message(request, messages.SUCCESS,
'Registro de Usuario Guardado Correctamente')
        else:
            messages.error(request,"Ya esta registrado ese nombre de
Usuario")
        return redirect('/seguridad/usuarios/crear')
    else:
        return
render_to_response("seguridad/errores/error_403.html",context_instan
ce = RequestContext(request))
```

Fuente: Elaboración Propia

• Registrar usuarios – SASIM

Tabla 32: Código Función de registro de usuario parte IOTMACH SASIM

Función	Guardar usuario
Descripción	Este es un socket que permite el almacenamiento de la información de un usuario en LDAP, espera el envío d la información a través de la función `.on('guardar_usuario')` que es enviada desde la aplicación IOTMACH SERVER
Lenguaje	Javascript

Código:

```
io.of('/autenticacion').on('connection', function (socket) {
        console.log('socket conectado...');
        var ldap = require('ldapjs');
        socket.on('guardar_usuario', function (data) {
            fs.readFile('./uidGeneral.txt','utf8',function(err,dato){
                UidGenerado = parseInt(dato)+1;
   fs.writeFile('./uidGeneral.txt', UidGenerado, function(err) {});
                var entry = {
                    sn: data.sn,
                    objectClass: ['inetOrgPerson',
'organizationalPerson','person','posixAccount','top'],
                    uid: data.uuid,
                    homeDirectory: ('/home/'+data.uuid),
                    loginshell: '/bin/bash',
                    uidNumber: UidGenerado,
                    gidNumber: '1001',
                    userPassword: data.uuid+'_'+data.telefono,
                    telephoneNumber: data.telefono,
                    mail: data.mail,
                    description: data.lst_app,
                    title: 'titulo',
                    street: data.direccion,
                    pager: '3154521'
                };
                console.log(entry);
                var client = ldap.createClient({url:
'ldap://192.168.0.42'});
client.bind('cn=iotmach,dc=iotmach,dc=com','12345678',function(err){
                    if (err) return console.log(err);
                    console.log('se conectp al bind');
                    if (err) return console.log(err);

client.add('cn='+data.cn+',ou='+data.ou+',dc=iotmach,dc=com',entry,f
unction(err){
                        if (err) return console.log(err);
                        console.log('agrego usuario');
                        client.unbind();
                    });
                });
            });
        });
    });
```

Fuente: Elaboración Propia

• Inicio Sesión – Sistema de Autenticación

Tabla 33: Código Función de inicio de sesión de Sistema de Autenticación

Función	Login
Descripción	Esta función permite conocer si el usuario que ingreso sus credenciales puede acceder a las aplicaciones de IOTMACH, en la función .authenticate() verifica si se encuentra dicho usuario dentro del servidor LDAP, en la generación de cookies se utiliza la función .encriptar() ya que con eso se protegerá la visibilidad de los datos y el acceso a las aplicaciones en sí, ya que si un usuario que no tiene permiso de acceso a una determinada aplicación no deberá tener creada la cookie en su navegador.
Lenguaje	Javascript

Código:

```
exports.login = function(req, res, next) {
    if (req.session.user) return res.redirect('/')
    var user = req.body.uid;
    var pass = req.body.passwd;

    auth_help.authenticate(user, pass, function(err, sessionData) {
        if (err) {
            res.render('index', {mensaje: "Usuario o contraseña
incorrectos."})
            return console.log("NO encontrado")
        }
        if (sessionData) {
            console.log("creamdo session");
            req.session.user = sessionData;
            var sessionID = logon(sessionData);

            var usuario_en = cifrado.encriptar('2uth1ld2p3');
            var password_en = cifrado.encriptar('9s3r0dAd1s');
            var val_usuario = cifrado.encriptar(user);
            var val_password = cifrado.encriptar(pass);
            res.cookie('sessionID', sessionID, {maxAge:3600000,
path:'/'});
            res.cookie(usuario_en,val_usuario,  {maxAge:3600000,
path:'/'});
            res.cookie(password_en,val_password,  {maxAge:3600000,
path:'/'});
            if(sessionData.ou){
           res.cookie('ruc_empresa',sessionData.ou,
{maxAge:3600000, path:'/'});
            }
            datosAuth = sessionData.apps;
            for(i=0; i<datosAuth.length;i++){
                switch(datosAuth[i]){
                    case 'server':
                        var fie_server =
cifrado.encriptar('10tw4tch53rV3r');
                        res.cookie(fie_server,fie_server,
```

```
{maxAge:3600000, path:'/'});
                        break;
                    case 'analisis': break;
                    case 'pentaho': break;
                    case 'puente':
                        var fie_bridge =
cifrado.encriptar('3r1dg310tw4tch');
                        var val_bridge =
cifrado.encriptar('3r1dg310tw4tch');
                        res.cookie(fie_bridge,val_bridge,
{maxAge:3600000, path:'/'});
                        break;
                }
            }
            res.redirect('/')

        } else {
            res.render('index', {mensaje: "Usuario o contraseña
incorrectos."})
        }
    })
}
```

Fuente: Elaboración Propia

- Inicio sesión – IOTMACH SERVER

Tabla 34: Código Función de inicio de sesión IOTMACH SERVER

Función	Inicio de sesión
Descripción	Este método permite leer los datos de las cookies con `get_credenciales()`, e iniciando la sesión dentro de la aplicación de Django siempre y cuando los datos de las cookies sean los correctos, la función `crearMenus()` permite generar los menús de la aplicación dinámicamente basándose en los roles que el usuario tiene.
Lenguaje	Python

Código:

```
def inicioSesion(request):
    dict = get_credenciales(request)
    user = dict['user']
    passwd = dict['passwd']
    permiso= dict['permiso']
    print(dict)
    if permiso:
        if user != '' and passwd != '':
            if request.user.is_authenticated():
                return redirect('/')
            else:
                cUser=authenticate(username=user, password=passwd)
                if cUser is not None and cUser.is_active:
```

```
                    login(request, cUser)
                    vUser = User.objects.get(username=user)
                    acciones('Inicio de Sesion', vUser)
                    precargar_Permisos()
                    crearMenus()
                    return redirect('/')
                else:
                    messages.add_message(request, messages.ERROR,
'Nombre de usuario o contrasenia incorrectos')
                    return redirect('http://127.0.0.1:6001/')
            else:
                return cerrarSesion(request)
        else:
            return redirect('http://127.0.0.1:6001/')
```

Fuente: Elaboración Propia

- Menús basado en roles

Tabla 35: Código Función de armar menús basados en permisos

Función	Arma Menú
Descripción	Este método permite obtener los menús correspondientes en base a los permisos que posee cada usuario, y los subdivide en tres categorías: menús, submenús, ítems
Lenguaje	Python

Código:

```
def armaMenu(permisos):
    menu = []
    submenu = []
    items=[]
    list_menus=seg_menu.objects.all()
    for j in permisos:
        for i in list_menus:
            if i.permisos != None:
                perm = j.split('.')
                if i.permisos.codename == perm[1]:
                    if i.men_estado:
                        items.append(i)
                        if(i.men_id_padre!=None):
                            submenu.append(i.men_id_padre)
                            if(i.men_id_padre.men_id_padre!=None):

menu.append(i.men_id_padre.men_id_padre)
    items = list(set(items))
    menu = list(set(menu))
    submenu = list(set(submenu))
    return {'menu' : menu,'submenu': submenu,'items': items}
```

Fuente: Elaboración Propia

El framework Django brinda funciones para prevenir vulnerabilidades, entre las cuales se destacan: `@login_required(login_url='/'), @csrf_exempt, csrf_token`

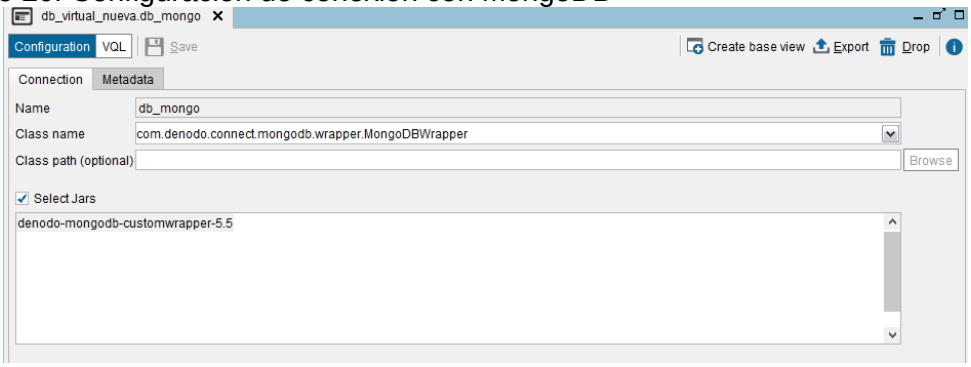
2.5.3 *Implementación de herramienta de integración de datos – Denodo*. Denodo tiene múltiples funcionalidades, y a su vez es de sencilla utilización e instalación.

2.5.3.1 *Preparación de entorno*. Se requiere la instalación de la herramienta de integración y virtualización de datos, para lo cual se siguieron los pasos registrados en el **ANEXO 8**.

2.5.3.2 *Conexión con las bases de datos*. Dentro de Denodo, se pueden crear múltiples conexiones a fuentes de información, ya sean motores de bases de datos distintos o archivos. Para ello se establecerán conexiones con la base de datos de MongoDB y PostgreSQL.

• *Conexión con base de datos Mongo*. Para realizar la conexión a una base de datos mongo, se requiere un módulo adicional. Para adjuntar el módulo de conexión de mongo, desde el menú Archivo, se selecciona Jar management, posteriormente se selecciona en la ventana la opción Create. Se procede a seleccionar el archivo que se encuentra dentro de la carpeta “dist” el archivo denodo-mongodb-customwrapper-5.5-20151207-jar-with-dependencies.jar. Una vez hecho esto se da clic derecho sobre la base de datos virtual → nuevo → Data Source → Custom. Donde se realiza la siguiente configuración.

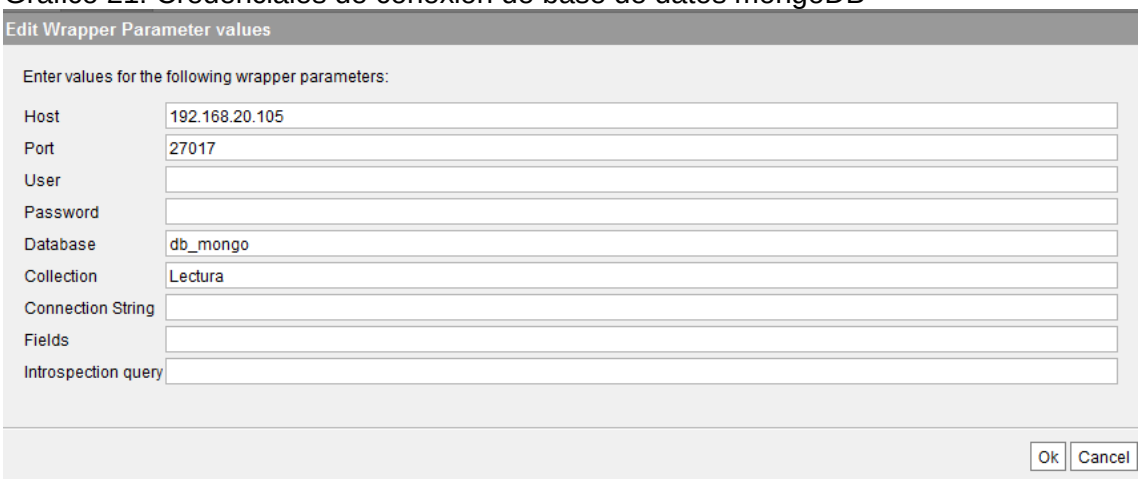
Gráfico 20: Configuración de conexión con MongoDB



Fuente: Denodo Platform

Como se observa en el **Gráfico 20**, se ha ingresado un nombre para la conexión (db_mongo), y se ha seleccionado el jar que ha sido previamente importado, y finalmente Save. Para acceder a una colección, se procede a crear una vista, en la opción Create base view. En donde se establece las opciones de configuración. Con ello se tiene creada la conexión con la base de datos no relacional MongoDB.

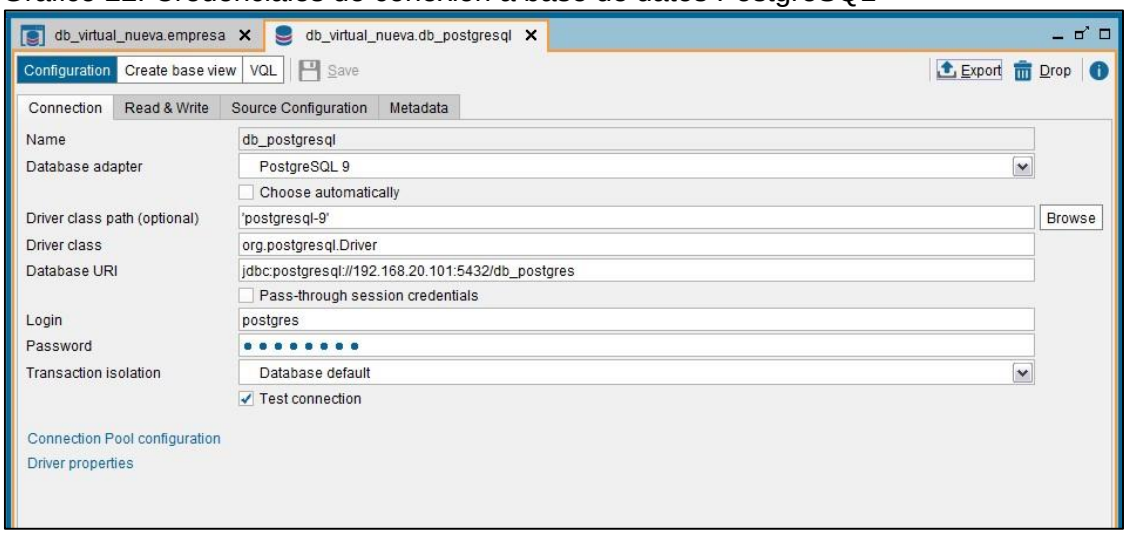
Gráfico 21: Credenciales de conexión de base de datos mongoDB



Fuente: Denodo Platform

• *Conexión con base de datos en PosgreSQL*. Para realizar la conexión a una base de datos mongo, se da clic derecho sobre la base de datos virtual → nuevo → Data Source → JDC. En el que se digitan las conexiones y luego se crea una vista basada en la conexión.

Gráfico 22: Credenciales de conexión a base de datos PostgreSQL



Fuente: Denodo Platform

2.6 Ejecución del prototipo

Como en los puntos anteriores, para el desarrollo del prototipo se ha dividido en tres componentes los cuales son Bridge IOTMACH, sistema de autenticación y la implementación de Denodo. Para ello a continuación se muestra la ejecución y funcionamiento de cada uno de los componentes mencionados

2.6.1 *Sistema puente de protocolos*. Al iniciar la aplicación, se busca a los clientes de adaptador de protocolos si están registrados en la base de datos para iniciarlos, de igual manera con los consumidores; de no ser ese el caso, se procede a ir hacia la página que administra los clientes, en la cual se puede ver lo siguiente:

Gráfico 23: Página principal Bridge IOTMACH

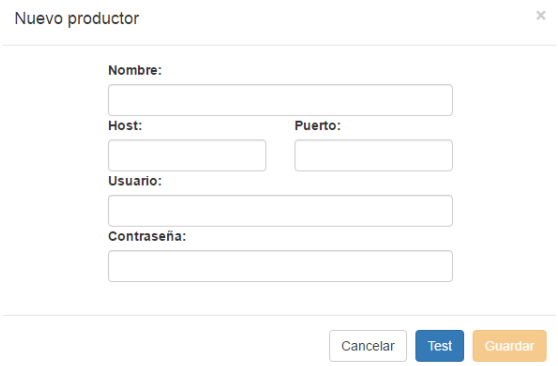
Fuente: Elaboración Propia

Como se puede observar en el Gráfico 23, en la interfaz se tienen tres secciones: la gestión de los clientes de Adaptador de Protocolos (clientes MQTT), la gestión de los Tópicos y la gestión de Consumidores.

En la sección de Clientes Adaptador de Protocolos, existen múltiples opciones en las cuales se describen a continuación:

1) Esta opción sirve para llamar al formulario de agregar un nuevo cliente de adaptador de protocolos. Una vez agregado el cliente. Al seleccionar el cliente de adaptador de protocolos se muestra una ventana emergente en la que se agregan las credenciales de conexión con el bróker seleccionado como se muestra en el Gráfico 24.

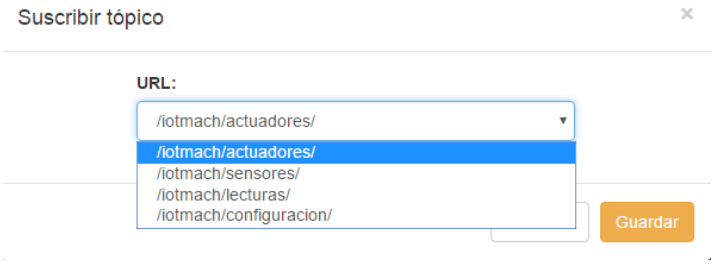
Gráfico 24: Formulario de Agregar Cliente AP



Fuente: Elaboración Propia

2) Una vez agregado el cliente, se lista en la sección de clientes de Adaptador de Protocolos, en la que en esta opción permite la eliminación del cliente.
3) Esta opción permite actualizar los datos de los clientes de Adaptador de Protocolos.
4) Esta opción permite la suscripción del cliente con los tópicos, así como se observa en el Gráfico 25:

Gráfico 25: Suscripción de clientes AP



Fuente: Elaboración Propia

5) Esta opción permite ver los mensajes que cruzan a través del tópico y del bróker.
6) Se desuscribe un tópico del cliente de adaptador de protocolos en esta opción.

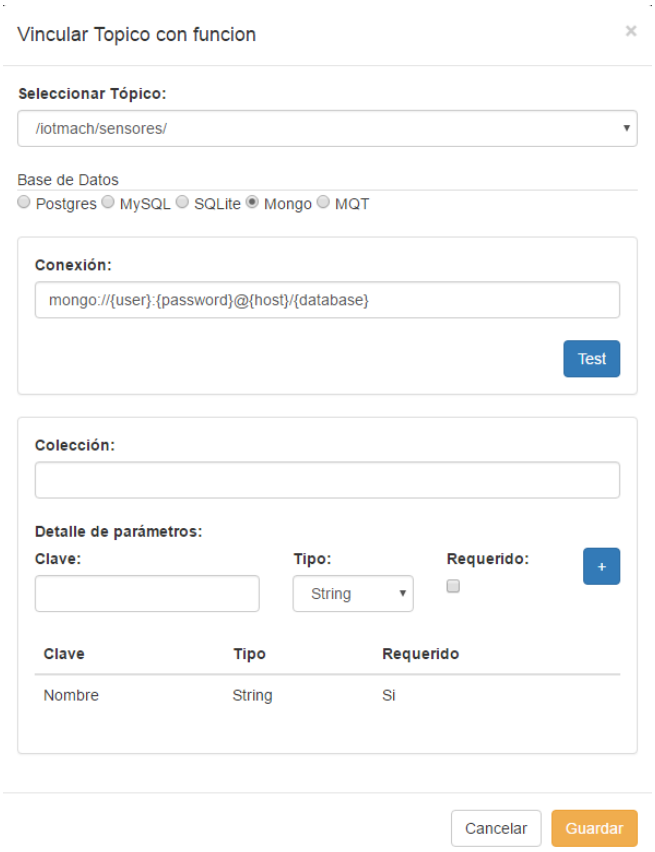
En la sección Tópicos, se gestionan los tópicos tanto para el gestor de Logs (Kafka) como el adaptador de protocolos (MQTT), para ello se tienen las siguientes opciones.

7) A través de esta opción se crearan los tópicos. Cada vez que se van creando tópicos, estos se listan en la sección y se tomaran como referencia tanto para los clientes de Adaptador de protocolos como el de los consumidores.
8) Con esta opción se eliminaran los tópicos, para todo el sistema.

En la sección de los consumidores se listaran los mismos que están esperando ser vinculados con algún tópico para poder recibir los mensajes del bróker de Kafka. A continuación se detallan las opciones:

9) Esta opción permite habilitar el formulario de gestión de consumidores, como se muestra en el Gráfico 26.

Gráfico 26: Formulario suscripción de consumidor



Fuente: Elaboración Propia

10) Permite eliminar el funcionamiento de un consumidor Kafka.

El sistema está activo en background receptando y emitiendo mensajes tanto las WSN como a las BDD, asi como se muestra en el Gráfico 27, por lo que no es necesario entrar a la pagina para que la aplicación funcione.

Gráfico 27: Ejecución en segundo plano de BRIDGE IOTMACH

```
Mongoose: mpromise (mongoose's default promise library) is deprecated, plug
Received a message on '/iotmach/lecturas/' en mqtt'
se inserto una nueva lectura en el dispositivo:F1:E2:D3:C4:B5:A6
Received a message on '/iotmach/lecturas/' en mqtt'
se inserto una nueva lectura en el dispositivo:F0:E1:D2:C3:B4:A5
Received a message on '/iotmach/lecturas/' en mqtt'
se inserto una nueva lectura en el dispositivo:F1:E2:D3:C4:B5:A6
Received a message on '/iotmach/lecturas/' en mqtt'
se inserto una nueva lectura en el dispositivo:F1:E2:D3:C4:B5:A6
Received a message on '/iotmach/lecturas/' en mqtt'
se inserto una nueva lectura en el dispositivo:F0:E1:D2:C3:B4:A5
Received a message on '/iotmach/lecturas/' en mqtt'
se inserto una nueva lectura en el dispositivo:F1:E2:D3:C4:B5:A6
Received a message on '/iotmach/lecturas/' en mqtt'
se inserto una nueva lectura en el dispositivo:F1:E2:D3:C4:B5:A6
```

Fuente: Elaboración Propia

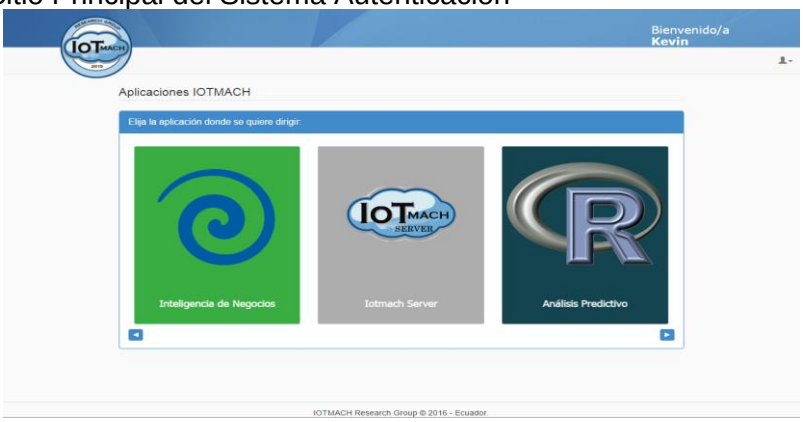
2.6.2 *Módulo de autenticación y seguridad*. Para utilizar las aplicaciones de IOTMACH, se necesita ingresar las credenciales de acceso a las aplicaciones, para lo cual se muestra una pantalla que solicitará al usuario dichas credenciales de acceso a las aplicaciones que forman parte del centro de procesamiento de datos, tales como Bridge IOTMACH, IOTMACH Server, Análisis Predictivo e Inteligencia de Negocios.

Gráfico 28: Formulario de Inicio de Sesión

Fuente: Elaboración Propia

1) Si se cuenta con las credenciales necesarias para acceder a las aplicaciones de IOTMACH, al dar clic en iniciar sesión aparecerá la ventana que muestra el Grafico 29, en el cual se selecciona la aplicación a utilizar.

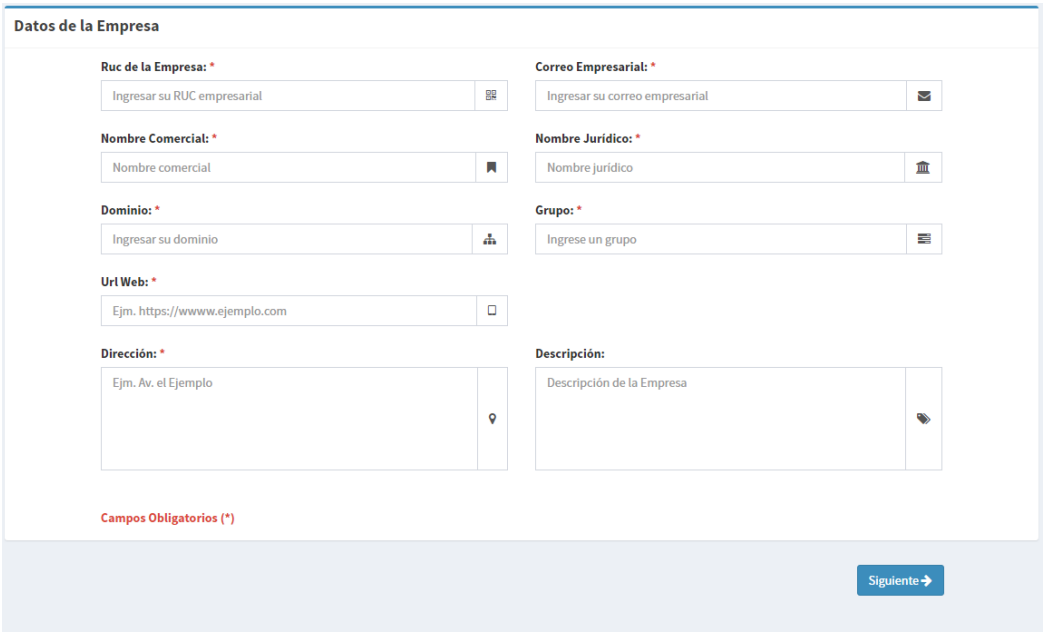
Gráfico 29: Sitio Principal del Sistema Autenticación



Fuente: Elaboración Propia

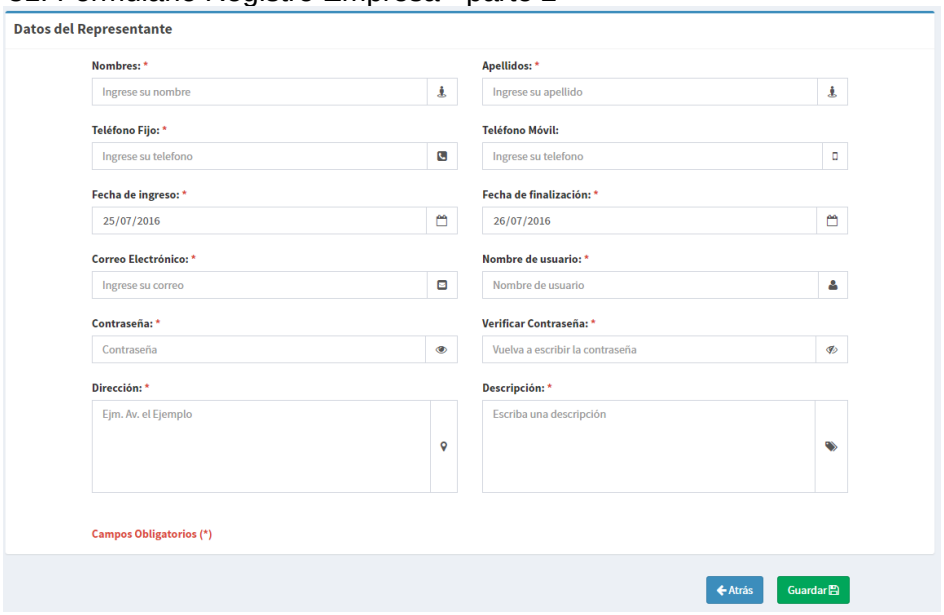
2) En caso de no tener una cuenta dentro de IOTMACH, se puede obtener entrando al enlace proporcionado, el mismo que redirige a los formularios de registro de empresa y de usuario respectivamente, dichos formularios son como se observa en los **Gráficos 30** y **31**.

Gráfico 30: Formulario Registro Empresa - parte 1



Fuente: Elaboración Propia

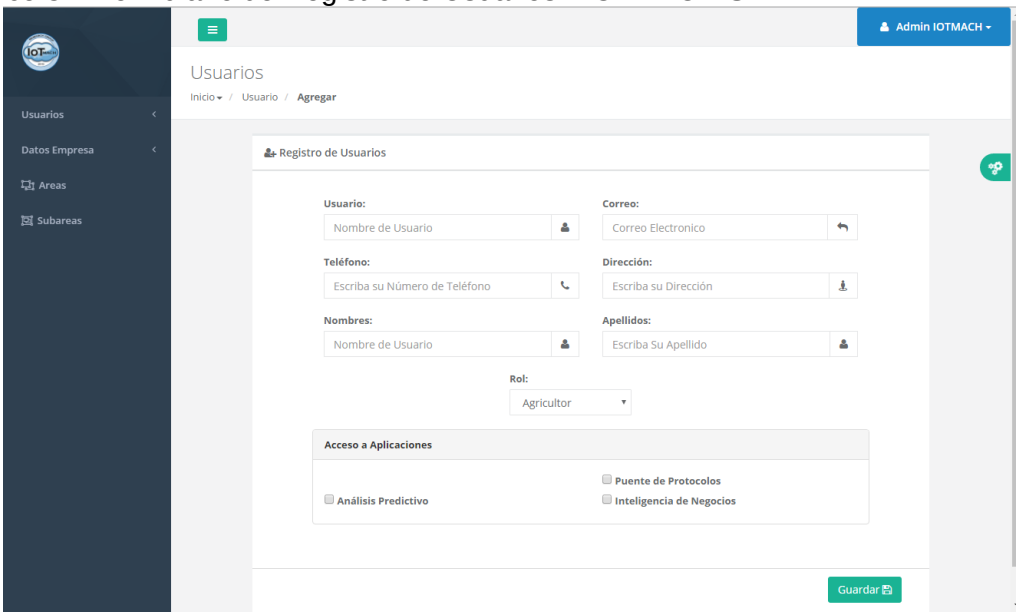
Gráfico 31: Formulario Registro Empresa - parte 2



Fuente: Elaboración Propia

Dentro de IOTMACH Server, se encuentra los módulos de empresa y seguridad, en el cual se tiene control sobre la actualización de los datos de empresa y la administración de los usuarios. Cada administrador de empresa podrá agregar más usuarios para que puedan acceder a la aplicación, en los cuales también se les agregara los roles para IOTMACH SERVER.

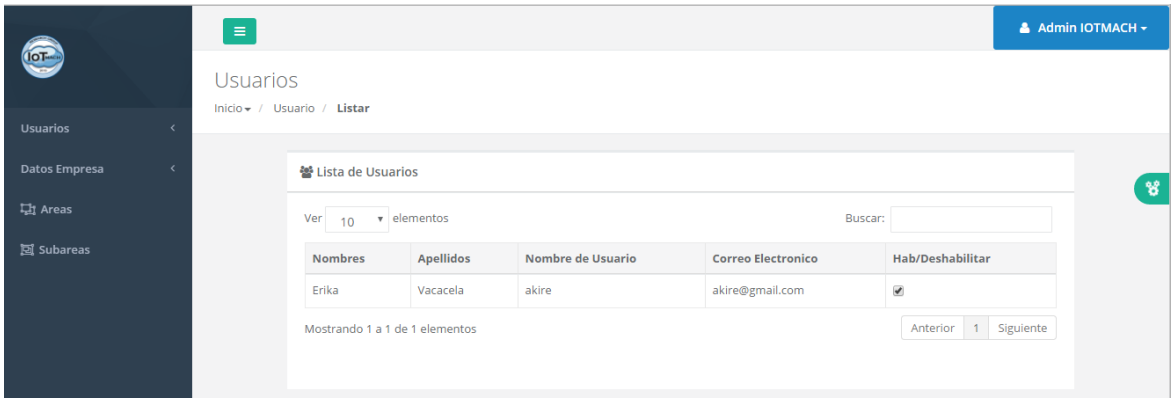
Gráfico 32: Formulario de Registro de Usuarios - IOTMACH SERVER



Fuente: Elaboración Propia

También se podrá listar a los usuarios que forman parte de una empresa.

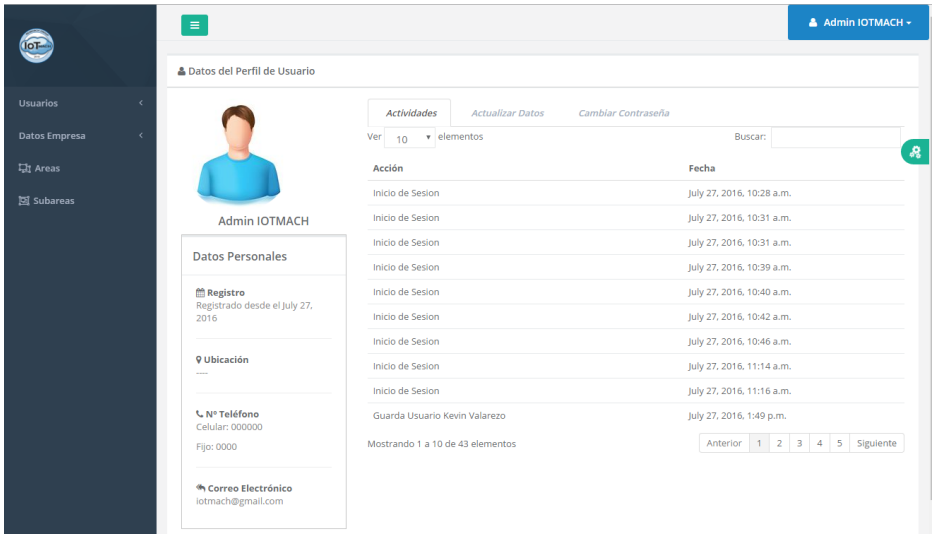
Gráfico 33: Formulario de Lista de Usuarios - IOTMACH SERVER



Fuente: Elaboración Propia

Cada usuario podrá editar su información personal así como su contraseña.

Gráfico 34: Datos del Perfil de Usuario - IOTMACH SERVER



Fuente: Elaboración Propia

Cada usuario que no cuenta con los permisos necesarios para navegar dentro de IOTMACH Server le aparecerá la siguiente pantalla, como se observa en el Gráfico 35.

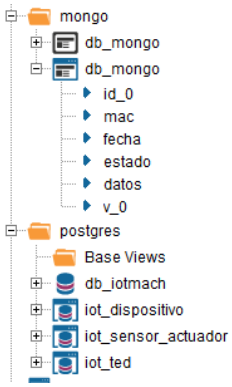
Gráfico 35: Error 403 de autorización

Fuente: Elaboración Propia

2.6.3 *Diseño de las vistas distribuidas en Denodo*. Denodo ofrece la posibilidad de integrar datos almacenados en varias fuentes de información, permitiendo realizar vistas distribuidas, virtualizando la información y creando webservices en base a estas vistas, para que así las aplicaciones puedan acceder a la información con mayor facilidad

Para empezar se tendran que crear las conexiones hacia mongo y postgres respectivamente, luego crear las vistas de los datos en cada BDD.

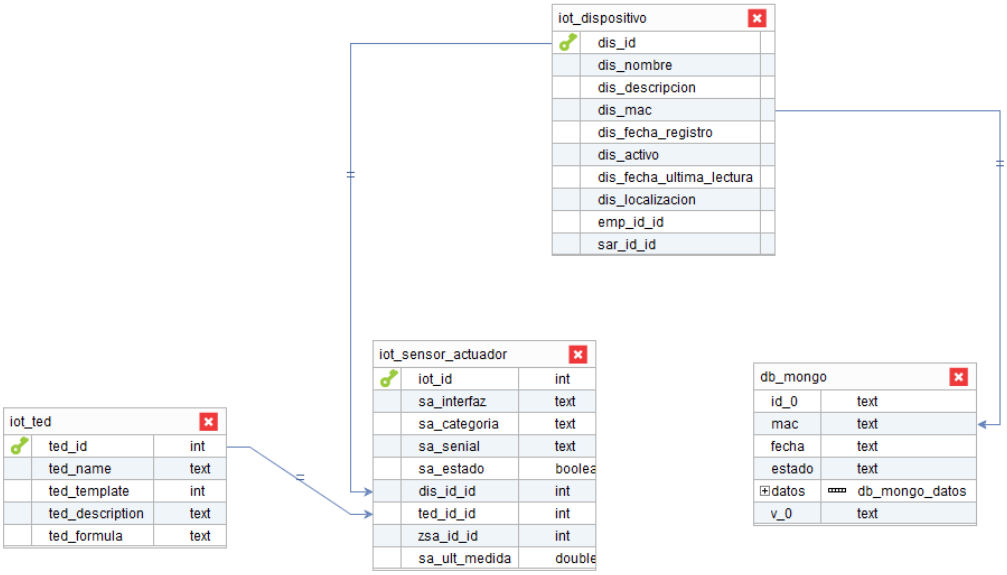
Gráfico 36: Vistas de las BDD generadas en Denodo



Fuente: Denodo Platform

Para realizar una vista distribuida, basta con darle clic derecho sobre una base de datos virtual creada, seleccionar new, luego join. Aparecera una ventana donde se debe arrastarar las tablas que se van a ocupar, y a su vez establecer las relaciones, como si fuese una base de datos relacional.

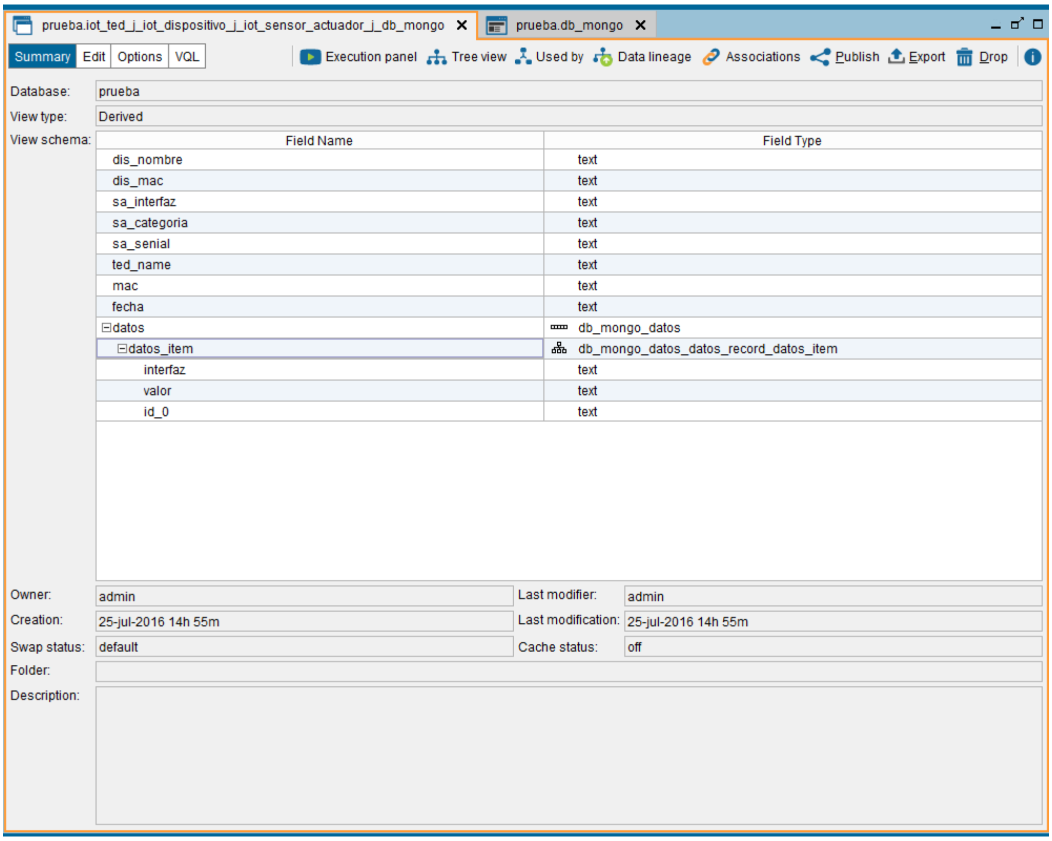
Gráfico 37: Esquema de vista distribuida - Denodo



Fuente: Denodo Platform

Una vez excluidos los datos de las tablas que no son necesarios, se procederá a guardar los datos, lo que Denodo mostrara los datos que la vista manejara.

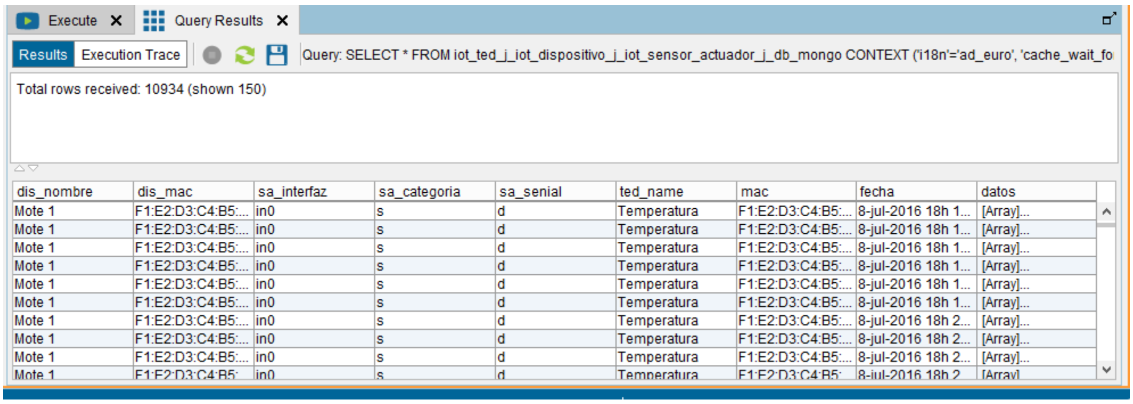
Gráfico 38: Datos de la vista distribuida - Denodo



Fuente: Denodo Platform

Para ver si la consulta distribuida es correcta, se dará clic en “execution panel”, para ver los resultados de la consulta.

Gráfico 39: Resultados de consulta distribuida.



Fuente: Denodo Platform

3. EVALUACIÓN DEL PROTOTIPO

3.1 Plan de evaluación

3.1.1 *Pruebas de usabilidad*. Una prueba de usabilidad es una medida empírica para medir la usabilidad de una herramienta, sitio o aplicación, tomada a partir de la observación sistemática de los usuarios llevando a cabo tareas reales [56]. Se enfoca en determinar el nivel del uso y facilidades de manejo que un sistema o aplicación brinda. La usabilidad debe ser entendida en relación con las características y necesidades propias de los usuarios [57]. Debido a que un usuario en particular manejara un sistema, aplicación o herramienta desarrollada es el mismo el encargado de determinar cuan fácil es el manejo de la plataforma elaborada.

Este tipo de test es una muy buena herramienta para entender cómo los usuarios interactúan con el sistema [58], ya que con estas pruebas se pueden detectar los distintos niveles de manejo y frustración que un usuario puede tener por no entender cómo manejar los datos.

Para ver el plan de pruebas empleado, dirigirse a los **Anexos 9 y 10.**

3.1.2 *Pruebas de stress*. Este tipo de pruebas, se realiza para determinar la solidez de la aplicación en momentos de ejecución a una carga extrema de trabajo, ayudando a determinar si la aplicación rendirá lo suficiente en caso sobrecarga. Son utilizadas con el objetivo de llevar a la aplicación al límite, probando el rendimiento de cada componente que interactúa para verificar si hay un decaimiento o algún comportamiento de alto consumo de recursos [59].

Dentro de las pruebas de stress, se escala la cantidad de carga con el tiempo hasta que se encuentren los límites del sistema, ayuda a los desarrolladores a determinar si la aplicación rendirá lo suficiente en caso de que la carga real supere a la carga esperada.

Para ver el plan de pruebas empleado, dirigirse al **Anexo 11.**

3.1.3 *Pruebas de seguridad***.** Este tipo de pruebas son usadas para mitigar los riesgos a las vulnerabilidades en las aplicaciones y/o sistemas, minimizando la posibilidad de sufrir ataques de usuarios malintencionados que causen daño al sistema y organización.

Dentro de las pruebas de seguridad se encuentran múltiples factores de los cuales se destacan la verificación de la gestión de la información, autenticación, autorización, gestión de sesiones, validación de sesiones entre otros [60].

Para ver el plan de pruebas empleado, dirigirse al **Anexo 12**.

3.1.4 *Pruebas de integración*. La prueba de integración consiste en ir asociando los distintos componentes que forman parte de un sistema, enfocándose en la prueba del funcionamiento correcto entre los elementos que conforman el sistema IOTMACH.
Para ver el plan de pruebas empleado, dirigirse al **Anexo 13**.

3.2 Resultados de la evaluación

Las evaluaciones realizadas a las aplicaciones, se aplicaron para medir el rendimiento y correcto funcionamiento de las mismas, verificando el nivel de errores que existen dentro de ellas, estas pruebas fueron efectuadas por usuarios con conocimientos respecto al tema, ver **Anexo 14**; es decir, son enfocadas para usuarios con roles de administrador quienes darán el criterio específico para solucionar las fallas que se presenten. Entre las pruebas aplicadas se encuentran:

a) Pruebas de usabilidad: en este tipo de pruebas el testeador verifica la facilidad de uso que tienen las aplicaciones, ya que manipula los componentes de las mismas y da su opinión para agilizar la utilización de las mismas.
b) Pruebas de stress: dentro de esta prueba se somete a cargas máximas de información que puede soportar la aplicación de Bridge IOTMACH, para poder determinar cuáles son los límites de rendimiento que éste presenta.
c) Pruebas de seguridad: en estas pruebas, es necesario verificar los niveles de seguridad de la información, se inspecciona los niveles de acceso que los usuarios pueden entrar y direccionarse a las aplicaciones indexadas.
d) Pruebas de integración: dentro de estas pruebas, se verifican el nivel de funcionamiento que tienen los componentes que forman parte de IOTMACH, es decir se determina el nivel de acoplamiento que tienen los elementos entre sí.

3.2.1 *Análisis de Resultados*. Una vez realizadas las diferentes pruebas, se cuenta con los siguientes resultados:

3.2.1.1 *Resultados de Pruebas de usabilidad*. Las pruebas de usabilidad fueron realizadas tanto para el Bridge IOTMACH como para el sistema de autenticación y seguridad.

- *Resultados de la prueba de Usabilidad de Bridge IOTMACH*. El sistema fue manipulado por usuarios que tienen conocimientos sobre brókeres de mensajería, lo cual con los siguientes datos se determina si la aplicación es satisfactoria o poco

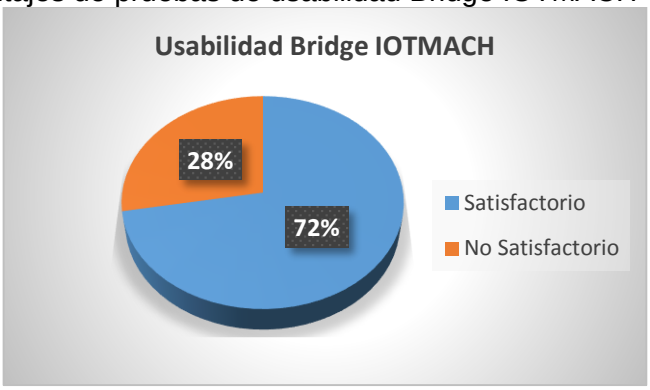
satisfactoria. Como se puede observar en el **Anexo 15,** en el cual se muestran la sumatoria de las respuestas dadas por los testers, dando como resultado lo siguiente:

Tabla 36: Totales de criterios de usabilidad a Bridge IOTMACH

Usabilidad	**Porcentaje**
Satisfactorio	72%
No Satisfactorio	28%
Total	100%

Fuente: Elaboración Propia

Gráfico 40: Porcentajes de pruebas de usabilidad Bridge IOTMACH



Fuente: Elaboración Propia

Interpretación: Como se puede apreciar en el **Gráfico 40**, se indica que la usabilidad del sistema Bridge IOTMACH es satisfactoria con un 72%, mientras que en un 28% no es satisfactorio en su utilización teniendo que mejorar en algunos aspectos.

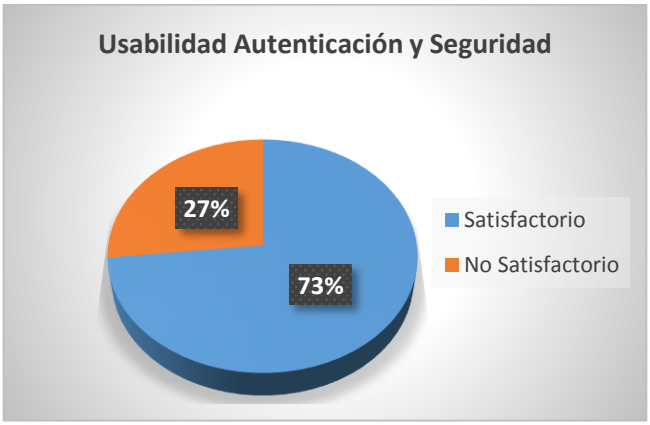
• *Resultados de la prueba de Usabilidad de módulo de Seguridad y autenticación*. El sistema de autenticación y el módulo de seguridad de IOTMACH, fue manipulado por múltiples usuarios que a su criterio determinaron si la aplicación es satisfactoria o poco satisfactoria. Como se puede observar en el **Anexo 16**, en el cual se muestran la sumatoria de las respuestas dadas por los testers, dando como resultado lo siguiente:

Tabla 37: Totales de criterios de usabilidad a módulo de Seguridad y autenticación

Usabilidad	**Porcentaje**
Satisfactorio	73%
No Satisfactorio	27%
Total	100%

Fuente: Elaboración Propia

Gráfico 41: Porcentajes de usabilidad de modulo de Seguridad y Autenticación



Fuente: Elaboración Propia

Interpretación: Al observar el **Gráfico 41**, indica que la usabilidad del sistema de Autenticación y módulo de Seguridad IOTMACH es satisfactoria con un 73%, mientras que en un 27% no es satisfactorio, ya que se tienen que realizar varias mejoras en el funcionamiento.

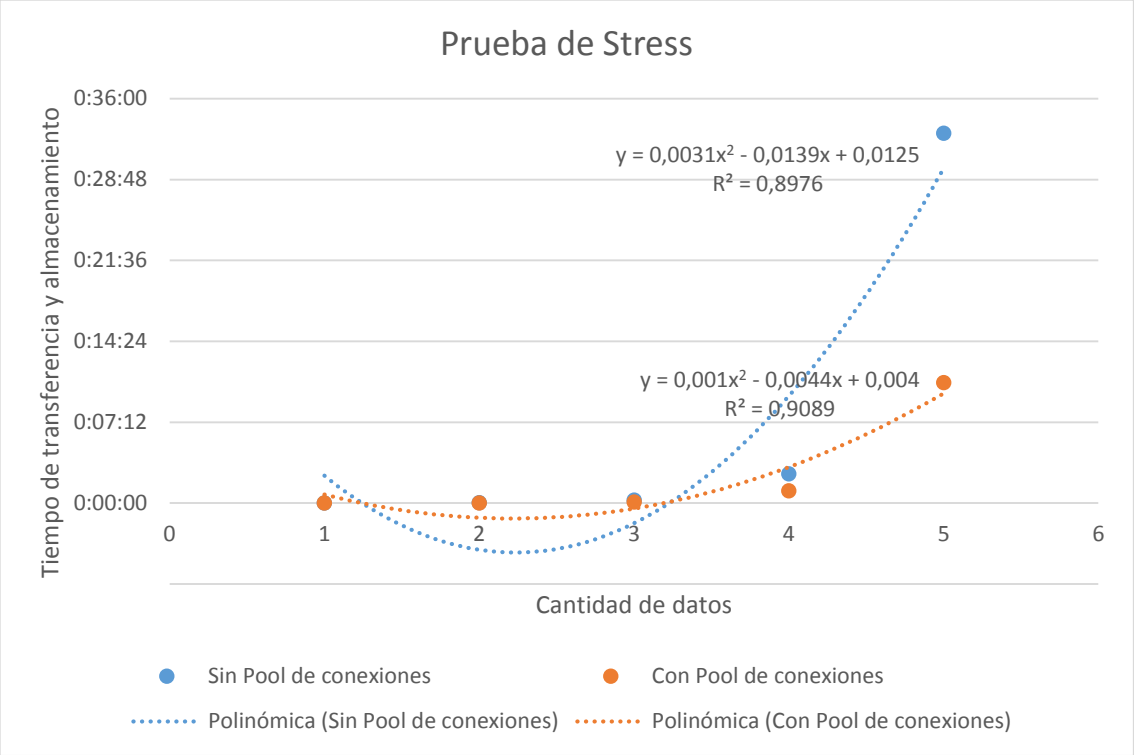
3.2.1.2 *Resultados de Pruebas de stress*. Las pruebas de stress fueron realizadas al sistema Bridge IOTMACH implementado en servidores con sistemas operativos CentOS 6.7, en la que difieren de memoria RAM y almacenamiento, ya que un servidor contaba con 1GB de RAM y 75GB de almacenamiento, mientras que el otro servidor cuenta con 45GB de RAM y 200GB de almacenamiento; teniendo en cuenta que estas pruebas se realizaron al proceso más crítico y a la vez más importante del sistema que es, la recepción de datos desde una WSN y el almacenamiento de dicha información en una base de datos no relacional. Al realizar la prueba en el servidor CentOS 6.7 con 1GB de RAM arrojo los siguientes resultados:

Tabla 38: Prueba de Stress a Bridge IOTMACH con Servidor 1GB de RAM

#	Cantidad Datos	Sin Pool de conexiones	Con Pool de conexiones
1	10	0:00:00	0:00:00
2	100	0:00:02	0:00:01
3	1000	0:00:16	0:00:06
4	10000	0:02:36	0:01:06
5	100000	0:32:55	0:10:43

Fuente: Elaboración Propia

Gráfico 42: Prueba de Stress a Bridge IOTMACH con Servidor 1GB de RAM



Fuente: Elaboración Propia

Interpretación: El sistema tiene tiempos de respuesta aceptables hasta cuando se gestionan 10000 paquetes de datos en un mismo instante, comienza a fallar cuando se tienen 100000 documentos en memoria, la respuesta tardara mucho en procesar la información originada en los brókeres de mensajería hasta almacenarlas en las BDD.

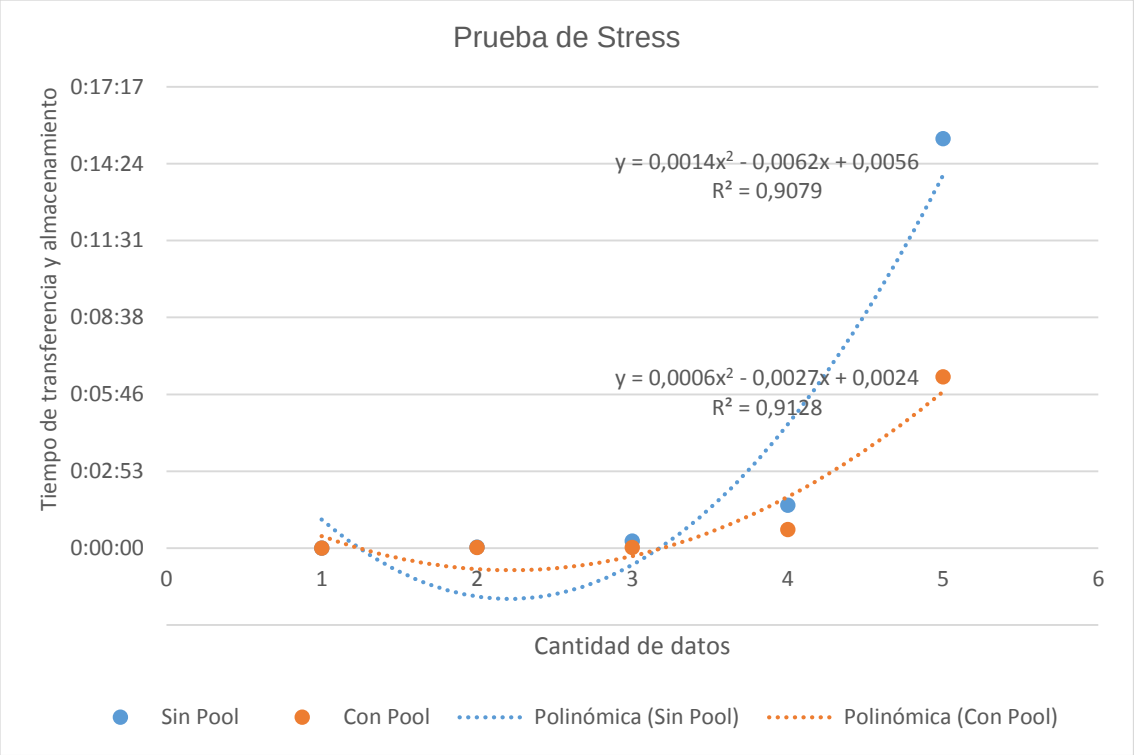
Al realizar la prueba en el servidor CentOS 6.7 con 4GB de RAM arrojo los siguientes resultados:

Tabla 39: Prueba de Stress a Bridge IOTMACH con Servidor 4GB de RAM

#	Cantidad Datos	Sin Pool de conexiones	Con Pool de conexiones
1	10	0:00:00	0:00:00
2	100	0:00:02	0:00:01
3	1000	0:00:16	0:00:01
4	10000	0:01:36	0:00:42
5	100000	0:15:20	0:06:25

Fuente: Elaboración Propia

Gráfico 43: Prueba de Stress a Bridge IOTMACH con Servidor 4GB de RAM



Fuente: Elaboración Propia

Interpretación: El sistema trabaja dentro de un entorno con mayores características, mejora los tiempos de respuesta cuando se gestionan hasta 10000 paquetes de datos en un mismo instante, pero sin embargo comienza a fallar cuando se gestionan 100000 documentos en memoria, la respuesta tardara mucho en procesar la información originada en los brókeres de mensajería hasta almacenarlas en las BDD, más aun cuando no se manejan pool de conexiones.

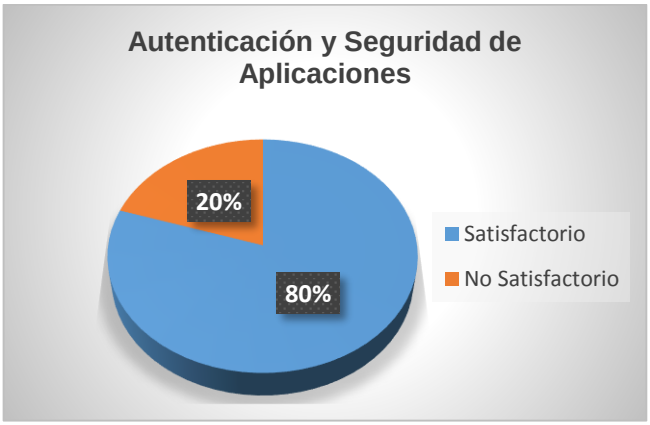
3.2.1.3 *Resultados de Pruebas de seguridad*. Las pruebas de seguridad fueron efectuadas en el sistema de autenticación y módulo de seguridad de IOTMACH, como se puede observar en el **Anexo 17,** en el cual se muestran la sumatoria de las respuestas dadas por los testers, dando como resultado lo siguiente:

Tabla 40: Totales de criterios de prueba de Seguridad.

Usabilidad	**Porcentaje**
Satisfactorio	80%
No Satisfactorio	20%
Total	100%

Fuente: Elaboración Propia

Gráfico 44: Porcentajes pruebas de Seguridad.



Fuente: Elaboración Propia

Interpretación: Al observar el **Gráfico 44**, indican que el sistema de Autenticación y módulo de Seguridad IOTMACH, cumple con un 80% de satisfacción en la seguridad de la información, mientras que se tiene un 20% de no satisfacción, debido a que existen controles que deben ajustarse para la seguridad total del sistema.

3.2.1.4 *Resultados de pruebas de integración*. Las pruebas de integración se realizaron conjunto los responsables de las otras aplicaciones que forman parte del Grupo de Investigación IOTMACH, para poder determinar el acoplamiento de las mismas entre sí, y proceder a realizar las respectivas correcciones.

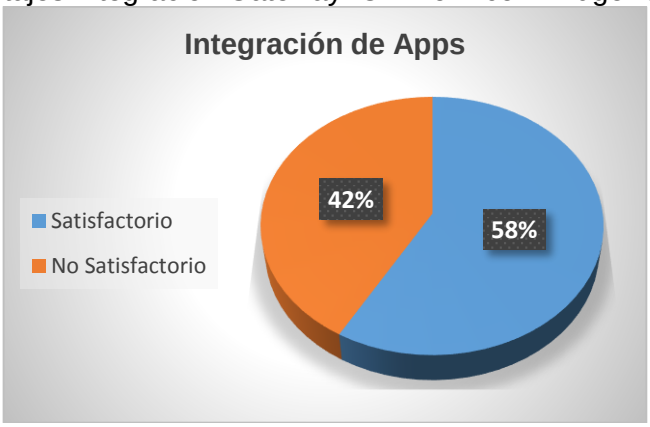
• *Resultados de la integración de Bridge IOTMACH con Gateway IOT Móvil*. La integración del sistema Bridge IOTMACH junto con el Gateway IOT Móvil, como se puede observar en el **Anexo 18**, se dieron los siguientes resultados:

Tabla 41: Criterios de Integración Gateway IOT Móvil con Bridge IOTMACH.

Integración	**Porcentaje**
Satisfactorio	58%
No Satisfactorio	42%
Total	100%

Fuente: Elaboración Propia

Gráfico 45: Porcentajes Integración Gateway IOT Móvil con Bridge IOTMACH.



Fuente: Elaboración Propia

Interpretación: Al observar el **Gráfico 45**, indica que las aplicaciones Gateway IOTMACH con Bridge IOTMACH tienen un 58% de satisfacción en la integración de su funcionamiento, mientras que en un 42% es de no satisfacción, debido a que existen errores que deben ajustarse para el correcto funcionamiento del sistema en conjunto.

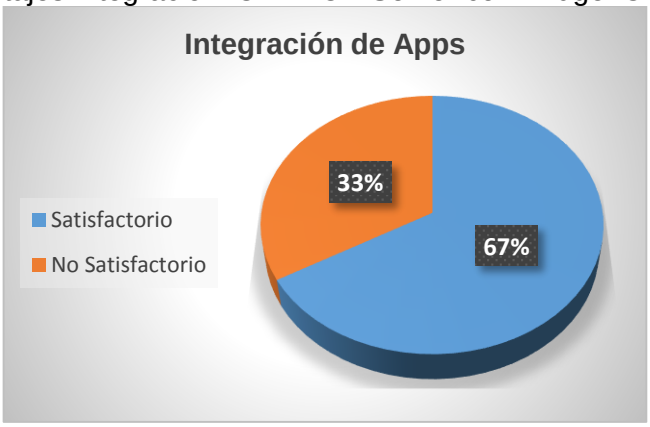
• *Resultados de la integración de Bridge IOTMACH con IOTMACH Server*. La integración del sistema Bridge IOTMACH junto con IOTMACH Server, como se puede observar en el **Anexo 19**, se dieron los siguientes resultados:

Tabla 42: Criterios de Integración IOTMACH Server con Bridge IOTMACH.

Integración	**Porcentaje**
Satisfactorio	67%
No Satisfactorio	33%
Total	100%

Fuente: Elaboración Propia

Gráfico 46: Porcentajes Integración IOTMACH Server con Bridge IOTMACH.



Fuente: Elaboración Propia

Interpretación: Al observar el **Gráfico 46**, indica que las aplicaciones Gateway IOTMACH con Bridge IOTMACH tienen un 67% de satisfacción la integración de su funcionamiento, mientras que se tiene un 33% de no satisfacción, debido a que existen errores que deben ajustarse para el correcto funcionamiento del sistema en conjunto.

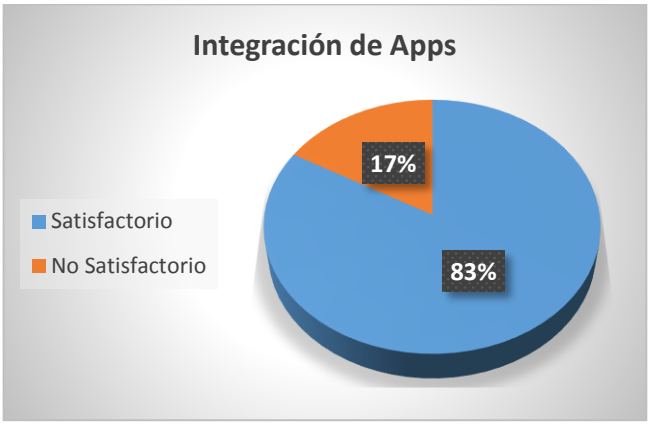
• *Resultados de la integración de Bridge IOTMACH con Gateway IOTMACH*. La integración del sistema Bridge IOTMACH junto con Gateway IOTMACH, como se puede observar en el **Anexo 20**, se dieron los siguientes resultados:

Tabla 43: Criterios de Integración Gateway IOTMACH con Bridge IOTMACH.

Integración	Porcentaje
Satisfactorio	83%
No Satisfactorio	17%
Total	100%

Fuente: Elaboración Propia

Gráfico 47: Porcentajes Integración Gateway IOTMACH con Bridge IOTMACH.



Fuente: Elaboración Propia

Interpretación: Al observar el **Gráfico 47**, indica que las aplicaciones Gateway IOTMACH con Bridge IOTMACH tienen un 83% de satisfacción la integración de su funcionamiento, mientras que se tiene un 17% de no satisfacción, debido a que existen errores que deben ajustarse para el correcto funcionamiento del sistema en conjunto.

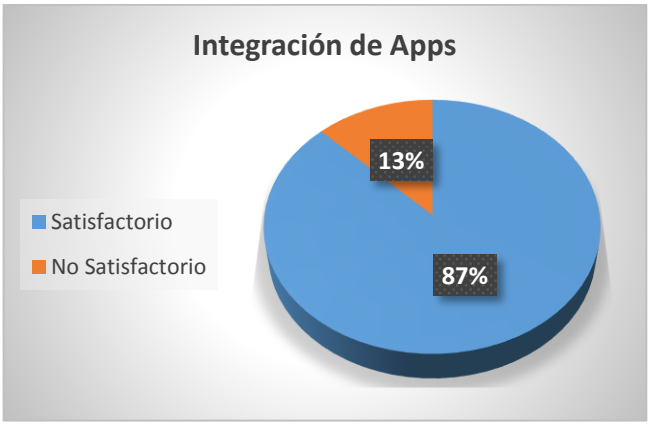
• *Resultados de la integración de Autenticación IOTMACH con IOTMACH Server*. La integración del sistema Autenticación IOTMACH junto con Gateway IOTMACH, como se puede observar en el **Anexo 21**, se dieron los siguientes resultados:

Tabla 44: Criterios de Integración Gateway IOTMACH con Bridge IOTMACH.

Integración	Porcentaje
Satisfactorio	87%
No Satisfactorio	13%
Total	100%

Fuente: Elaboración Propia

Gráfico 48: Porcentajes Integración Gateway IOTMACH con Bridge IOTMACH.



Fuente: Elaboración Propia

Interpretación: Al observar el **Gráfico 48**, indica que las aplicaciones Autenticación IOTMACH con Bridge IOTMACH tienen un 87% de satisfacción en la integración de su funcionamiento, mientras que se tiene un 13% de no satisfacción, debido a que existen errores que deben ajustarse para el correcto funcionamiento del sistema en conjunto.

CONCLUSIONES

- El prototipo desarrollado cumple con la implementación de herramientas de gestión de Logs, de integración de datos y de procesos de seguridad necesarios para el correcto funcionamiento de los sistemas que forman IOTMACH, ya que con este se facilita la manipulación del tráfico, tratamiento y acceso de la información que circula por el centro de procesamiento de datos.

- A través de adaptadores de protocolos, se puede realizar la comunicación entre dispositivos electrónicos con un conjunto de servidores, fortaleciendo el concepto de IOT. Los dispositivos de redes de sensores inalámbricos envían información hacia el centro de procesamiento de datos, por lo que la implementación del bróker de mensajería MQTT facilito el arribo de los datos al servidor, y mediante a la realización del Bridge IOTMACH se obtiene dichas información y se la almacena las respectivas bases de datos.

- Con la llegada de los datos originados en los dispositivos WSN, el sistema Bridge IOTMACH comunica los protocolos de mensajería entre sí, en este caso MQTT y Kafka; dicho sistema funciona como eje central en la comunicación de los componentes, ya sea enviando información hacia las bases de datos o al motor de reglas, o a su vez dando respuesta a los dispositivos de redes de sensores inalámbricos para que generen un evento en un momento determinado.

- Denodo es una plataforma que facilita la obtención de los datos alojados en diferentes fuentes de información, ya que las aplicaciones tardarían mucho tiempo en acceder a estos registros, por lo que Denodo mejora los tiempos de respuesta de las aplicaciones gracias a la integración de los datos que este brinda.

- El grupo de Investigación IOTMACH está conformado por diversos componentes que al unirlos se complementan un solo sistema; la implementación de un sistema de autenticación único entre aplicaciones facilita el control de los niveles de acceso que un usuario puede tener a las mismas.

- IOTMACH Server cuenta con un módulo de seguridad en el que se gestionan la autorización que tienen los usuarios hacia las funciones, a su vez se utilizan funciones y procesos que mitigan las posibles vulnerabilidades que pueda tener la aplicación. Ninguna aplicación es 100% segura pero, con la implementación de los controles adecuados se mitigan las deficiencias que puedan existir.

RECOMENDACIONES

- Buscar otras alternativas que permitan mejorar el funcionamiento del prototipo desarrollado, ya sea en cuanto a servidores de mensajería o en lenguaje de programación que permite optimizar recursos y tiempos de respuesta.

- En el momento que se requiera aplicar escalabilidad al sistema Bridge IOTMACH, uno de los principales motivos de esto es el inevitable crecimiento de los dispositivos WSN que se encontrarán en un número de equipos bastante grande enviando información al centro de procesamiento de datos, por lo que es necesario emplear mecanismos de balanceo de carga que mejoren el rendimiento de los procesos de comunicación de los protocolos de mensajería y el almacenamiento en las diversas bases de datos empleadas en los servidores.

- Optimizar el código fuente escrito en Node.JS que forma parte de Bridge IOTMACH, ya que en un futuro posiblemente existirán mejores soluciones para los clientes MQTT y Kafka, de no ser así, aprender mucho más a fondo el framework Node.JS o algún lenguaje de programación y/o framework de desarrollo que maneje el tiempo real de manera natural.

- En la integración y virtualización de datos, existen muchas más aplicaciones que se pueden investigar e implementar, pero antes de ello es más conveniente investigar métodos de conexión a consultas distribuidas en los lenguajes de programación que se vayan a emplear, ya que dependiendo de dichos métodos un integrador de datos tendrá mucho más utilidad en el funcionamiento de un sistema.

- En cuanto a la autenticación de las aplicaciones, es necesario investigar otras soluciones para la comunicación de información entre aplicaciones, por ejemplo la utilización de tokens; en el caso de seguir utilizando cookies, se deben mejorar mecanismos de encriptación de información, ver la posibilidad de leer y escribir dichas cookies en dominios distintos, e implementar escalabilidad a dicho sistema de autenticación, puesto que IOTMACH está proyectado a seguir desarrollando aplicaciones IOT.

- Siempre que se vaya a comenzar a desarrollar una aplicación o simplemente al comenzar con algún recurso de programación desconocido, es recomendable leer toda la documentación, debido a que mayoritariamente estos poseen mecanismos de seguridad integrados que los desarrolladores promedio desconocen, esto facilita mucho el minimizar el impacto de ataques a las vulnerabilidades del sistema.

BIBLIOGRAFÍA

[1] N. Bari, G. Mani, and S. Berkovich, “Internet of Things as a Methodological Concept,” in *2013 Fourth International Conference on Computing for Geospatial Research and Application*, 2013, pp. 48–55.

[2] R. K. Kodali and S. Soratkal, “Trust model for WSN,” in *2015 International Conference on Applied and Theoretical Computing and Communication Technology (iCATccT)*, 2015, pp. 903–906.

[3] J. R. Rajalakshmi, M. Rathinraj, and M. Braveen, “Anonymizing log management process for secure logging in the cloud,” in *2014 International Conference on Circuits, Power and Computing Technologies [ICCPCT-2014]*, 2014, pp. 1559–1564.

[4] A. Alotaibi and A. Mahmmod, “Enhancing OAuth services security by an authentication service with face recognition,” in *2015 Long Island Systems, Applications and Technology*, 2015, pp. 1–6.

[5] N. A. H. M. Rodzi, M. S. Othman, and L. M. Yusuf, “Significance of data integration and ETL in business intelligence framework for higher education,” in *2015 International Conference on Science in Information Technology (ICSITech)*, 2015, pp. 181–186.

[6] A. Zimmermann, R. Schmidt, K. Sandkuhl, M. Wissotzki, D. Jugel, and M. Mohring, “Digital Enterprise Architecture - Transformation for the Internet of Things,” in *2015 IEEE 19th International Enterprise Distributed Object Computing Workshop*, 2015, pp. 130–138.

[7] BBVA Innovation Center, “Internet de las Cosas, modelo de negocio en auge,” *Innovation Trends*, pp. 6–9, 2015.

[8] A. W. Burange and H. D. Misalkar, “Review of Internet of Things in development of smart cities with data management & privacy,” in *2015 International Conference on Advances in Computer Engineering and Applications*, 2015, pp. 189–195.

[9] M. H. Asghar, A. Negi, and N. Mohammadzadeh, “Principle application and vision in Internet of Things (IoT),” in *International Conference on Computing, Communication & Automation*, 2015, pp. 427–431.

[10] H. Gregor, “Hub and Spoke [or] Zen and the Art of Message Broker Maintenance,” *Enterprise Integration Patterns*, 2003. [Online]. Available: http://www.enterpriseintegrationpatterns.com/ramblings/03_hubandspoke.html.

[11] J. Yamamoto, H. Nakagawa, K. Nakayama, Y. Tahara, and A. Ohsuga, “A Context Sharing Message Broker Architecture to Enhance Interoperability in Changeable

Environments," in *2009 Third International Conference on Mobile Ubiquitous Computing, Systems, Services and Technologies*, 2009, pp. 31–39.

[12] Apache Software Foundation, "Apache ActiveMQ ™," 2011. [Online]. Available: http://activemq.apache.org/.

[13] IBM, "WebSphere Message Broker," 2016. [Online]. Available: http://www.ibm.com/support/knowledgecenter/SSKM8N_8.0.0/com.ibm.etools.mft.doc/bb43020_.htm.

[14] R. Hat, "JBoss Messaging," 2013. [Online]. Available: http://docs.jboss.org/jbossmessaging/docs/guide-1.0.1.SP5/html_single/#about.

[15] Eclipse, "Mosquitto," 2011. [Online]. Available: https://mosquitto.org/.

[16] Apache Software Foundation, "Apache Kafka," 2014. [Online]. Available: http://kafka.apache.org/.

[17] K. Dotchkoff, "Compatibilidad con protocolos adicionales para Centro de IoT," *Microsoft Azure*, 2016. [Online]. Available: https://azure.microsoft.com/es-es/documentation/articles/iot-hub-protocol-gateway/. [Accessed: 22-Jul-2016].

[18] A. Campoverde, D. Hernández, and B. Mazón, "Cloud computing con herramientas open-source para Internet de las cosas," *Maskana*, pp. 173–182, 2015.

[19] J. P. Loyall, M. Gillen, K. Z. Haigh, R. Walsh, C. Partridge, G. Lauer, and T. Strayer, "A concept for publish-subscribe information dissemination and networking," in *2012 IEEE International Conference on Communications (ICC)*, 2012, pp. 5810–5816.

[20] A. Kumar and S. Johari, "Push notification as a business enhancement technique for e-commerce," in *2015 Third International Conference on Image Information Processing (ICIIP)*, 2015, pp. 450–454.

[21] Eclipse, "MQTT," 2014. [Online]. Available: http://mqtt.org/.

[22] H. W. Chen and F. J. Lin, "Converging MQTT Resources in ETSI Standards Based M2M Platform," in *2014 IEEE International Conference on Internet of Things(iThings), and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber, Physical and Social Computing (CPSCom)*, 2014, no. iThings, pp. 292–295.

[23] P. Murugesan and I. Ray, "Audit Log Management in MongoDB," in *2014 IEEE World Congress on Services*, 2014, pp. 53–57.

[24] Rapid7, "Intro to Log Management," *https://logentries.com/doc/log-management/*, 2015. [Online]. Available: https://logentries.com/doc/log-management/.

[25] J. Kreps and L. Corp, "Kafka : a Distributed Messaging System for Log Processing," *ACM SIGMOD Work. Netw. Meets Databases*, p. 6, 2011.

[26] A. Kadadi, R. Agrawal, C. Nyamful, and R. Atiq, “Challenges of data integration and interoperability in big data,” in *2014 IEEE International Conference on Big Data (Big Data)*, 2014, pp. 38–40.

[27] Rafaelma, “Sobre PostgreSQL,” *postgresql.org.es*, 2010. [Online]. Available: http://www.postgresql.org.es/sobre_postgresql.

[28] MongoDB, “What Is MongoDB?,” 2016. .

[29] Denodo, “¿Qué es la virtualización de datos?,” 2016. .

[30] Denodo, “Why the Denodo Platform?,” 2016. [Online]. Available: http://www.denodo.com/en/denodo-platform/why-denodo-platform.

[31] R. Von Solms, K.-L. Thomson, and P. M. Maninjwa, “Information Security Governance control through comprehensive policy architectures,” in *2011 Information Security for South Africa*, 2011, pp. 1–6.

[32] S. Rafique, M. Humayun, B. Hamid, A. Abbas, M. Akhtar, and K. Iqbal, “Web application security vulnerabilities detection approaches: A systematic mapping study,” in *2015 IEEE/ACIS 16th International Conference on Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing (SNPD)*, 2015, pp. 1–6.

[33] OWASP, “Top 10 Web applications vulnerabilities,” 2016. .

[34] H. Shahriar and M. Zulkernine, “Client-Side Detection of Cross-Site Request Forgery Attacks,” in *2010 IEEE 21st International Symposium on Software Reliability Engineering*, 2010, pp. 358–367.

[35] T. Alexenko, M. Jenne, S. D. Roy, and W. Zeng, “Cross-Site Request Forgery: Attack and Defense,” in *2010 7th IEEE Consumer Communications and Networking Conference*, 2010, pp. 1–2.

[36] M. S. Siddiqui and D. Verma, “Cross site request forgery: A common web application weakness,” in *2011 IEEE 3rd International Conference on Communication Software and Networks*, 2011, pp. 538–543.

[37] A. Sadeghian, M. Zamani, and S. Ibrahim, “SQL Injection Is Still Alive: A Study on SQL Injection Signature Evasion Techniques,” in *2013 International Conference on Informatics and Creative Multimedia*, 2013, pp. 265–268.

[38] M. K. Gupta, M. C. Govil, and G. Singh, “Static analysis approaches to detect SQL injection and cross site scripting vulnerabilities in web applications: A survey,” in *International Conference on Recent Advances and Innovations in Engineering (ICRAIE-2014)*, 2014, pp. 1–5.

[39] Li Qian, Zhenyuan Zhu, Jun Hu, and Shuying Liu, “Research of SQL injection attack and prevention technology,” in *2015 International Conference on Estimation, Detection and Information Fusion (ICEDIF)*, 2015, no. 123456, pp.

303–306.

[40] M. K. Gupta, M. C. Govil, and G. Singh, “Predicting Cross-Site Scripting (XSS) security vulnerabilities in web applications,” in *2015 12th International Joint Conference on Computer Science and Software Engineering (JCSSE)*, 2015, pp. 162–167.

[41] Y. Sun and D. He, “Model Checking for the Defense against Cross-Site Scripting Attacks,” in *2012 International Conference on Computer Science and Service System*, 2012, pp. 2161–2164.

[42] X. Guo, S. Jin, and Y. Zhang, “XSS Vulnerability Detection Using Optimized Attack Vector Repertory,” in *2015 International Conference on Cyber-Enabled Distributed Computing and Knowledge Discovery*, 2015, pp. 29–36.

[43] R. Kumar, “Mitigating the authentication vulnerabilities in Web applications through security requirements,” in *2011 World Congress on Information and Communication Technologies*, 2011, vol. 58, no. 12, pp. 1294–1298.

[44] Y. Takamatsu, Y. Kosuga, and K. Kono, “Automated detection of session management vulnerabilities in web applications,” in *2012 Tenth Annual International Conference on Privacy, Security and Trust*, 2012, pp. 112–119.

[45] L. Lian and C. Song, “Research of RBAC access control model based on LDAP tree storage,” in *Proceedings of 2011 International Conference on Electronic & Mechanical Engineering and Information Technology*, 2011, vol. 3, pp. 1363–1365.

[46] M. O. Ahmad, P. Kuvaja, M. Oivo, and J. Markkula, “Transition of Software Maintenance Teams from Scrum to Kanban,” in *2016 49th Hawaii International Conference on System Sciences (HICSS)*, 2016, no. i, pp. 5427–5436.

[47] M. Ikonen, E. Pirinen, F. Fagerholm, P. Kettunen, and P. Abrahamsson, “On the Impact of Kanban on Software Project Work: An Empirical Case Study Investigation,” in *2011 16th IEEE International Conference on Engineering of Complex Computer Systems*, 2011, pp. 305–314.

[48] M. O. Ahmad, J. Markkula, and M. Oivo, “Kanban in software development: A systematic literature review,” in *2013 39th Euromicro Conference on Software Engineering and Advanced Applications*, 2013, no. September 2013, pp. 9–16.

[49] Development, “JavaScript,” *Mozilla Developer Network*, 2016. [Online]. Available: https://developer.mozilla.org/es/docs/Web/JavaScript.

[50] Microsoft, “Node.js Applications with VS Code,” 2016. [Online]. Available: https://code.visualstudio.com/Docs/runtimes/nodejs.

[51] F. Plaza, “Comunicación Síncrona vs Comunicación Asíncrona,” 2009. [Online]. Available: http://www.fernandoplaza.com/2009/03/comunicacion-sincrona-vs-

comunicacion-asincrona.asp.

[52] P. S. Foundation, “Documentación de Python en Español,” 2009. [Online]. Available: http://docs.python.org.ar/tutorial/2/appetite.html.

[53] Z. Chen, L. Chen, Y. Zhou, Z. Xu, W. C. Chu, and B. Xu, “Dynamic Slicing of Python Programs,” in *2014 IEEE 38th Annual Computer Software and Applications Conference*, 2014, pp. 219–228.

[54] D. S. Foundation, “Django makes it easier to build better Web apps more quickly and with less code,” 2016. [Online]. Available: https://www.djangoproject.com/.

[55] Django Software Foundation, “Security in django,” 2016. [Online]. Available: https://docs.djangoproject.com/en/1.9/topics/security/.

[56] M. Mañas, “USABILIDAD Pruebas/test/encuestas.” Universidad Politecnica de Valencia, p. 23.

[57] Y. Hassan, F. J. Martín, and I. Ghzala, “Diseño Web Centrado en el Usuario: Usabilidad y Arquitectura de la Información,” *Hipertext.net*, 2004. .

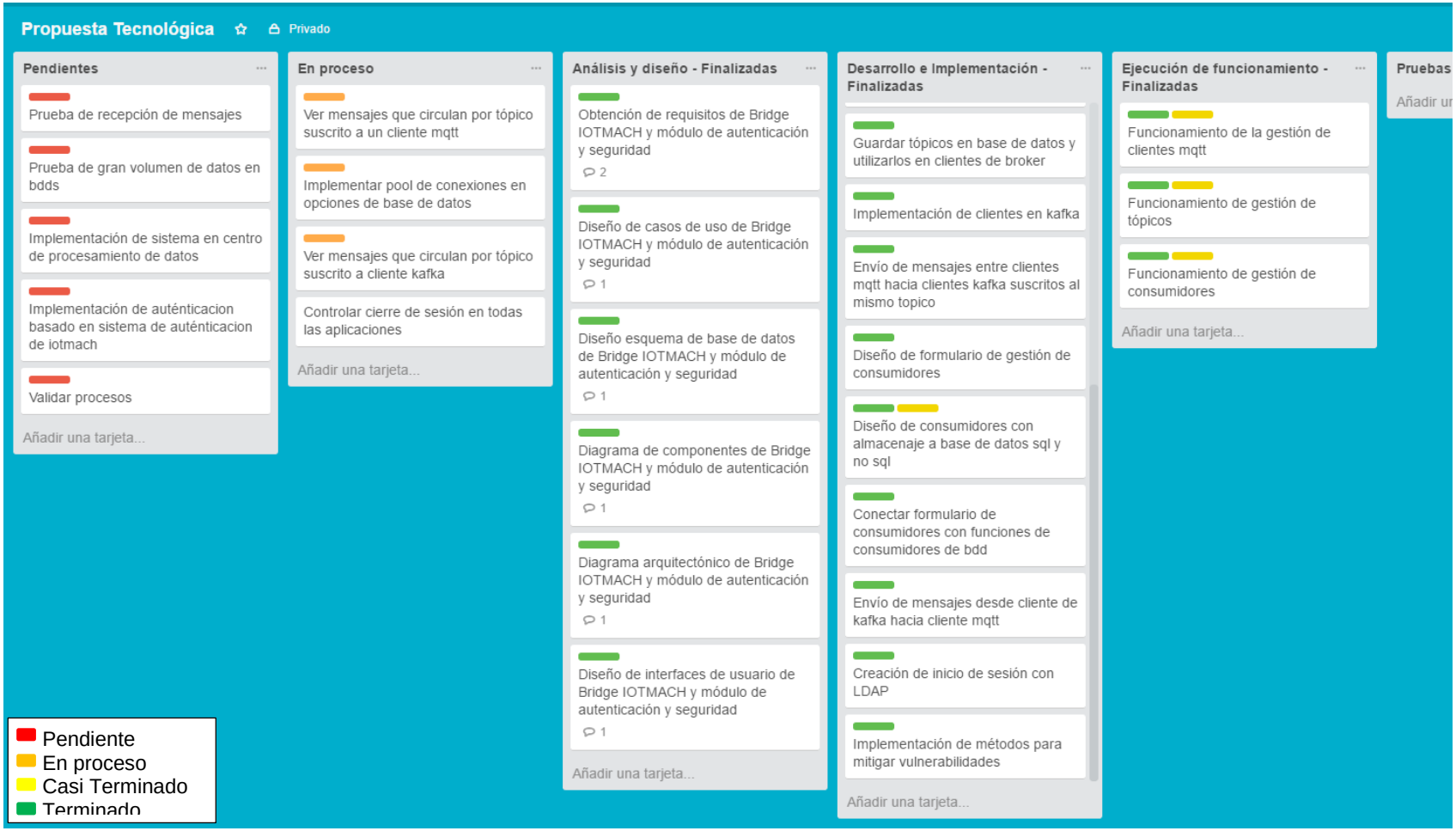
[58] G. González, “¿Tests de usabilidad? Cuando tus visitas pueden darte la clave del diseño de tu web,” *es.jimbo*, 2015. [Online]. Available: http://es.jimdo.com/2015/02/03/tests-de-usabilidad-cuando-tus-visitas-pueden-darte-la-clave-del-dise%C3%B1o-de-tu-web/.

[59] GALATEA, “PRUEBAS DE ESTRÉS DE SOFTWARE,” 2016. [Online]. Available: https://www.galatea-it.com/servicios/calidad-de-software/prueba-estres#page.

[60] OWASP, “Web Application Security Testing Cheat Sheet,” 2015. [Online]. Available: https://www.owasp.org/index.php/Web_Application_Security_Testing_Cheat_Sheet.

ANEXOS

Anexo 1. Planificación del desarrollo del prototipo



Anexo 2. Datos procesados por puente de protocolos

El sistema de puente de protocolos manipula datos adicionales a su sistema, es decir, los consumidores que se encargan de receptar los mensajes de Kafka y almacenar dichos mensajes a las bases de datos relacionales y no relacionales, cuyo formato de datos es documentos JSON, y a continuación se detallan los parámetros que manejan los documentos:

Tópico: /iotmach/actuadores/

DBD	Actuadores	
Descripción	Documento emitido por el motor de reglas.	
Tipo de BD	Documento JSON	
Parámetros	**Clave**	**Tipo de dato**
	paquete_id	String
	Mac	String
	Datos	Array
	Fecha	Date
	interfaz	String
	Valor	String

Esquema de datos JSON para colección actuadores

```
{
        paquete_id:      Number,
        mac:             String,
        datos: [{
                interfaz:        String,
                valor:           Number
                },]
        fecha:           Date
}
```

Tópico: /iotmach/lecturas/

DBD	Lecturas	
Descripción	Documento emitido por la WSN.	
Tipo de BD	Documento JSON	
Parámetros	**Clave**	**Tipo de dato**
	paquete_id	String
	Mac	String
	Datos	Array
	Fecha	String

Esquema de datos JSON para colección lecturas

```
{
        paquete_id:     Number,
        mac:            String,
        bateria:        Number,
        datos: [{
                interfaz:       String,
                valor:          Number
                },]
        fecha:          Date
}
```

Tópico: /iotmach/configuracion/

DBD	Configuración	
Descripción	Documento emitido por la WSN.	
Tipo de BD	Documento JSON	
Parámetros	**Clave**	**Tipo de dato**
	nombre	String
	Mac	String
	bateria	Number
	datos	Array
	localizacion	String
	descripcion	String
	categoria	String
	senial	String
	ted_id	String
	interfaz	String

Esquema de datos JSON para los documentos de configuración

```
Configuración = {
        nombre:  String,
        mac:            String,
        empresa:        String,
        localizacion:   String,
        descripcion:    String,
        datos: [
                {
                categoria:      String,
                senial:  String,
                ted_id:  String,
                interfaz:       String
                },
        ]
}
```

Anexo 3. Instalación del bróker Mosquitto en CentOS 6.7

Preparando Entorno del S.O.

Abrir una terminal, logearse con una cuenta de privilegios de super usuario, posteriormente introducir las siguientes instrucciones.

```
$su –
# yum groupinstall "Development Tools"
# yum install wget mercurial cmake openssl-devel c-ares-devel libuuid-devel
# cd /home/servidor1/Descargas/
#wget https://github.com/warmcat/libwebsockets/archive/v1.4-chrome43-
firefox-36.tar.gz
# tar xf v1.4-chrome43-firefox-36.tar.gz
# cd libwebsockets-1.4-chrome43-firefox-36
# mkdir build; cd build
# cmake .. -DLIB_SUFFIX=64
# make install
# echo "/usr/local/lib64" | sudo tee -a
/etc/ld.so.conf.d/libwebsockets.conf
# ldconfig
# ldconfig -p | grep libwebsockets
# cd /home/servidor1/Descargas/
# wget http://mosquitto.org/files/source/mosquitto-1.4.tar.gz
# tar xvzf mosquitto-1.4.tar.gz
# cd mosquitto-1.4
```

Editando Archivo de compilación

```
# vim config.mk
```

Una vez abierto el archivo desplazarse a través del mismo hasta la línea WITH_WEBSOCKETS:= no y modificar por la siguiente:

```
WITH_WEBSOCKETS:=yes
```

Guardar y Cerrar (vim Esc :wq Enter), Continuando con la instalación:

```
# make binary
# make install
```

Ahora crearemos un nuevo usuario al sistema, denominado mosquitto:

```
#useradd -r -m -d /var/lib/mosquitto -s /usr/sbin/nologin -g nogroup
```

Si al ejecutar el comando mosquitto_sub se presenta el error

```
# mosquitto_sub
  mosquitto_sub: error while loading shared libraries:
libmosquitto.so.1: cannot open shared object file: No such file or
directory mosquito
```

Se deberá agregar el siguiente enlace simbólico:

```
# ln -s /usr/local/lib64/libwebsockets.so.4.0.0
```

Luego editar el archivo ldconfg (Centos 6.7)

```
#vi /etc/ld.so.conf
  include ld.so.conf.d/*.conf
  include /usr/local/lib
  /usr/lib
  /usr/local/lib
```

Guardar y Cerrar.Posteriormente Recargar el archivo ldconfig en el sistema.

```
#/sbin/ldconfig
```

Para lograr ejecutar mosquitto, deberemos hacerlo a partir de un archivo de configuración, para ello crearemos uno nuevo en caso de que no exista ninguno.

```
#cd /etc/mosquitto/

#vim mosquito.conf mosquito
```

Colocaremos dentro del archivo la siguiente configuración.

```
listener 1883
protocol websockets
password_file
/etc/mosquitto/mosquitto.pwd
allow_anonymous false
use_identity_as_username false
```

Las líneas anteriores se encuentran haciendo referencia a un archivo que actualmente no existe en el directorio, como lo es password_file que almacena las contraseñas de los usuarios para el servidor mosquitto, para crear este archivo será necesario emplear la siguiente instrucción, en donde especificaremos el usuario:

```
# mosquitto_passwd -c mosquitto.pwd user1
```

Para saber si lo que hemos configurado está bien, a través de la terminal ejecutar:

```
#/usr/local/sbin/mosquitto -c
```

El resultado debería ser similar a:

```
1458086744: mosquitto version 1.4 (build date 2016-03-15 18:05:28-0500)
starting
1458086744: Config loaded from mosquitto.conf.
1458086744: Opening ipv4 listen socket on port 1883.
1458086744: Opening ipv6 listen socket on port 1883.
```

Para mayor información visitar la siguiente página:

https://www.justinribeiro.com/chronicle/2014/10/22/mosquitto-libwebsockets-google-compute-engine-setup/

Y desde la página oficial de Mosquitto:

http://mosquitto.org/man/

Anexo 4. Instalación del bróker Kafka en CentOS 6.7

Preparando los archivos necesarios

Se descarga el paquete desde:

https://www.apache.org/dyn/closer.cgi?path=/kafka/0.9.0.0/kafka_2.11-0.9.0.0.tgz

Una vez descargado, abrimos la terminal desde la carpeta donde está alojado el archivo descargado:

```
$ su
# tar -xzf kafka_2.11-0.9.0.0.tgz
# cd kafka_2.11-0.9.0.0
```

Iniciando el servidor

Kafka utiliza ZooKeeper, por lo que es necesario iniciar un servidor ZooKeeper puesto que ya viene uno preestablecido con los archivos descargados. Se puede utilizar el script que viene empaquetado con Kafka para obtener una instancia de un nodo ZooKeeper.

```
#bin/zookeeper-server-start.sh config/zookeeper.properties
```

Y a su vez levantamos el servidor de Kafka

```
#bin/kafka-server-start.sh config/server.properties
```

Creación de Tópicos

Para probar el funcionamiento de kafka se creara un topico denominado prueba_1

```
#   bin/kafka-topics.sh   --create   --zookeeper   localhost:2181   --
replication-factor 1 --partitions 1 --topic prueba 1
```

Para ver si se ha creado satisfactoriamente el tópico ejecutamos la siguiente línea de comandos:

```
# bin/kafka-topics.sh --list --zookeeper localhost:2181
```

Iniciar el productor

Ejecutar el productor y se escribe algunos mensajes en la consola para enviar al servidor:

```
#bin/kafka-console-producer.sh --broker-list localhost:9092 --topic prueba_1
hola que tal
prueba de kafka
```

Iniciar el consumidor

Se inicia el consumidor

```
#bin/kafka-console-consumer.sh --zookeeper localhost:2181 --topic prueba_1 --from-beginning
hola que tal
prueba de kafka
```

Si se tiene cada uno de los comandos anteriores se ejecutando en un terminal diferente, entonces debería ser capaz de escribir mensajes en el terminal productor y verlos aparecen en el terminal del consumidor.

Para mayor información acerca de la configuración del servidor Apache Kafka, visitar el siguiente enlace:

http://kafka.apache.org/documentation.html

Anexo 5. Instalación de MongoDB en CentOS 6.7

Para instalar el servidor de MongoDB en CentOS 6.7, se realizan los siguientes pasos:

Crear un archivo /etc/yum.repos.d/mongodb-org-3.2.repo para que se pueda instalar directamente usando el yum. Dentro del archivo se colocan las siguientes líneas:

```
[mongodb-org-3.2]
name=MongoDB Repository
baseurl=https://repo.mongodb.org/yum/redhat/$releasever/mongodb-
org/3.2/x86_64/
gpgcheck=1
enabled=1
gpgkey=https://www.mongodb.org/static/pgp/server-3.2.asc
```

Una vez que se agregaron los repositorios al yum, se procede la instalación del mongoDB:

```
$sudo yum install -y mongodb-org
```

Ya que se haya terminado la instalación, se procede a configurar el SELinux ubicado en /etc/selinux/config.

Para iniciar el servicio y también para que inicie cada vez que se enciende el equipo:

```
$sudo service mongod start
$sudo chkconfig mongod on
```

Para mayor información sobre las configuraciones al servidor visitar el siguiente enlace:

https://docs.mongodb.com/manual/tutorial/install-mongodb-on-red-hat/

Anexo 6. Instalación de Node.JS en CentOS 6.7

Para instalar Node JS en CentOS, se debe hacer lo siguiente:

Iniciar como root

```
$ su -
```

Para la versión 4.03 se usa:

```
#curl --silent --location
https://rpm.nodesource.com/setup_4.x | bash -
```

Para la version 0.10, que es la que se necesita, se tiene:

```
#curl --silent --location
https://rpm.nodesource.com/setup | bash -
```

Luego se realiza la instalación con:

```
#yum -y install nodejs npm
```

Y con eso se tiene ya corriendo Node dentro del servidor, para mayor información visitar el siguiente enlace:

https://nodejs.org/en/download/package-manager/

Anexo 7. Instalación de LDAP

Para la instalación del protocolo LDAP en CentOS 6.7 hay que seguir los siguientes pasos:

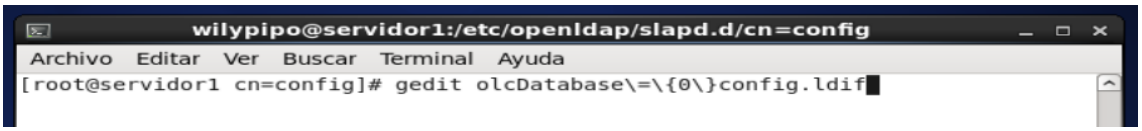
1. Se comienza abriendo una terminal y entramos como “root” con el comando “su” luego digitamos la clave que le asignamos a nuestro “root”.

2. Se instala los paquetes de “OpenLDAP” con el comando “yum –y install openldap openldap-clientes openldap-servers”, asi se descargara los paquetes tanto para cliente y servidor LDAP.

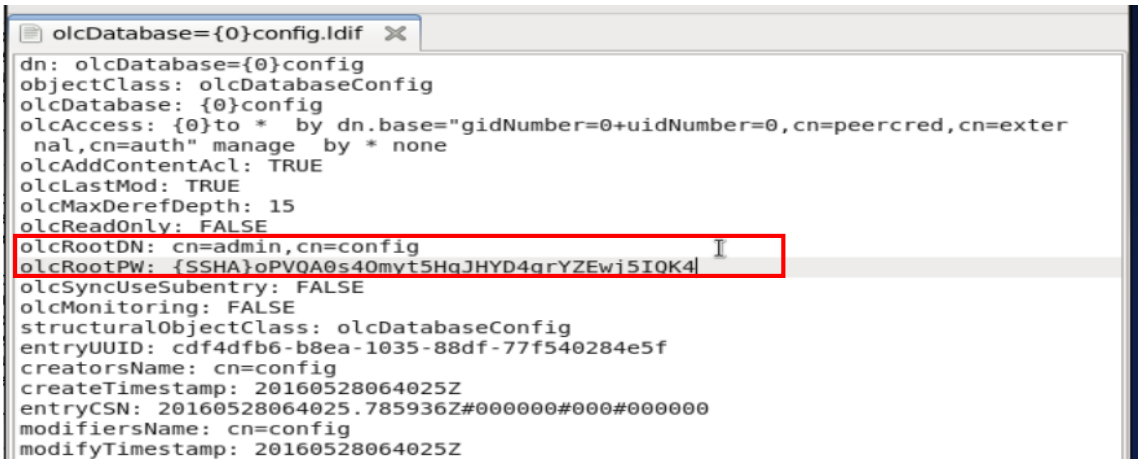
3. Crear una contraseña encriptada para nuestro “admin” de todo el directorio de LDAP, para esto usamos el comando “slappasswd”, digitamos nuestra clave y la clave que nos muestre la copiamos en un archivo de texto y la guardamos ya que se la usara después.

4. Nos dirigimos a la siguiente ruta “cd /etc/openldap/slapd.d/cn=config”.

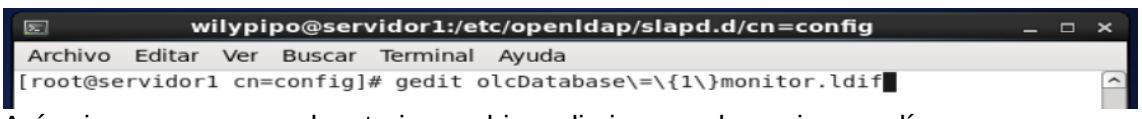
5. En esta ruta con el comando “gedit” modificaremos el siguiente archivo:



   - Al abrir el archivo eliminamos las primeras líneas que aparecen comentadas con “#” luego agregar las líneas que estar en el cuadro rojo, “cn=admin” va hacer el administrador de todo el árbol, en ves de “admin” se le pude poner otro “username” y olcRootPW es la clave del administrador en este caso “admin”, la clave es la que se creó anteriormente con el comando“slappasswd”.



6. En la misma ruta con el comando “gedit” modificaremos el siguiente archivo:

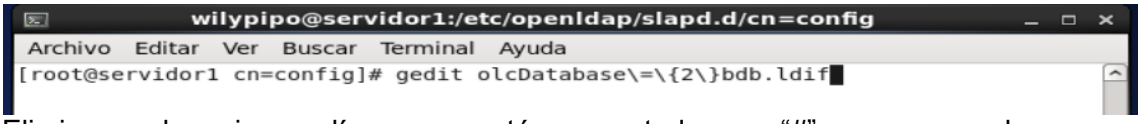


   - Así mismo como en el anterior archivo eliminamos las primeras líneas que aparecen comentadas con “#”.
   - Luego creamos la ruta base de nuestro administrador en el árbol “cn=admin,dc=abc,dc=local” significa que el “admin” pertenece a “abc.local”.

olcDatabase={1}monitor.ldif

```
dn: olcDatabase={1}monitor
objectClass: olcDatabaseConfig
olcDatabase: {1}monitor
olcAccess: {0}to *  by dn.base="gidNumber=0+uidNumber=0,cn=peercred,cn=exter
 nal,cn=auth" read  by dn.base="cn=admin,dc=abc,dc=local" read  by * n
 one
olcAddContentAcl: FALSE
olcLastMod: TRUE
olcMaxDerefDepth: 15
olcReadOnly: FALSE
olcSyncUseSubentry: FALSE
olcMonitoring: FALSE
structuralObjectClass: olcDatabaseConfig
entryUUID: cdf4e25e-b8ea-1035-88e0-77f540284e5f
creatorsName: cn=config
createTimestamp: 20160528064025Z
entryCSN: 20160528064025.785936Z#000000#000#000000
modifiersName: cn=config
modifyTimestamp: 20160528064025Z
```

7. En la misma ruta con el comando “gedit” modificaremos el siguiente archivo:



   - Eliminamos las primeras líneas que estén comentadas con “#” y agregamos las siguientes líneas que están en el cuadro rojo, “oclSuffix” nos permitirá conectarnos con el servidor ya que es la raíz de ldap y es usado en los lenguajes de programación como Node.js, Java, Django, C#, etc, asi mismo agregamos “DN y PW” que viene hacer la ruta del administrador del LDAP y su clave encriptada.

Abrir Guardar Deshacer

olcDatabase={2}bdb.ldif

```
dn: olcDatabase={2}bdb
objectClass: olcDatabaseConfig
objectClass: olcBdbConfig
olcDatabase: {2}bdb
olcSuffix: dc=abc,dc=local
olcAddContentAcl: FALSE
olcLastMod: TRUE
olcMaxDerefDepth: 15
olcReadOnly: FALSE
olcRootDN: cn=admin,dc=abc,dc=local
olcRootPW: {SSHA}oPVQA0s4Omyt5HqJHYD4grYZEwj5IQK4
olcSyncUseSubentry: FALSE
olcMonitoring: TRUE
olcDbDirectory: /var/lib/ldap
olcDbCacheSize: 1000
olcDbCheckpoint: 1024 15
```

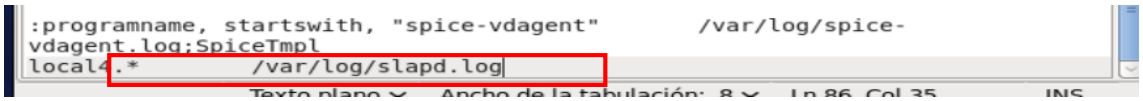
8. Nos dirigimos a la ruta “cd /etc/openldap” y modificamos el archivo con gedit “ldap.conf”

9. “BASE” nos permite establecer la raíz, y “URI” permite establecer una conexión por donde los clientes podrán acceder con el servidor LDAP, también se puede colocar el DNS, agregamos lo que esta con marco rojo, para habilitar lo anterior dicho y el “loglever 128” es para ver donde se generan los problemas con LDAP ya sea cliente o servidor.

ldap.conf

```
#
# LDAP Defaults
#

# See ldap.conf(5) for details
# This file should be world readable but not world writable.

BASE dc=abc,dc=local
URI ldap://192.168.45.1

#SIZELIMIT      12
#TIMELIMIT      15
#DEREF          never

TLS_CACERTDIR /etc/openldap/cacerts
loglevel 128
```

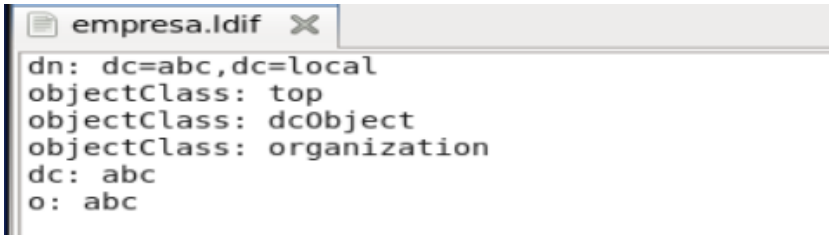
10. Nos dirigimos a “cd /etc” y modificamos el archivo “rsyslog.conf” nos ubicamos al final del documento y agregamos la línea que esta con marco rojo.



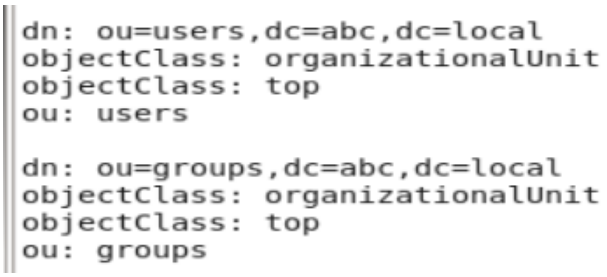
11. Reiniciamos los servicios con “service rsyslog restart” y “service sldap restart” y deberá salir “OK, OK” si sale “FALLO” reinicien los servicios de nuevo.

12. Creamos nuestro archivo con extensión “.ldif” este archivo será nuestro árbol base donde agregaremos grupos y usuarios necesarios para el árbol. En la ruta “cd /etc/openldap” con el comando “gedit” crearemos el archivo “miarchivo.ldif”

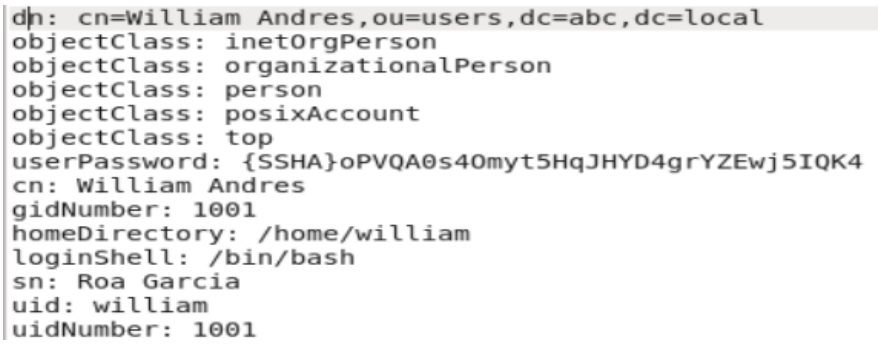
13. Primeramente debemos crear nuestra raíz para que sea reconocida en el árbol y a partir de esta raíz se agregaran todo los grupos y usuarios que vallamos creando.



14. Luego creamos nuestros grupos, un grupo es una unidad organizativa(ou) donde puede tener más subgrupos y usuarios:



15. Luego los usuarios “cn” viene hacer el nombre común de la persona seguido del grupo donde será ingresado



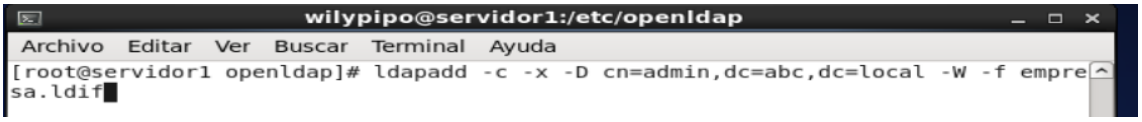
```
dn: cn=Ana Betsabeth,ou=users,dc=abc,dc=local
objectClass: inetOrgPerson
objectClass: organizationalPerson
objectClass: person
objectClass: posixAccount
objectClass: top
userPassword: {SSHA}oPVQA0s4Omyt5HqJHYD4grYZEwj5IQK4
cn: Ana Betsabeth
gidNumber: 1001
homeDirectory: /home/ana
loginShell: /bin/bash
sn: Andrade Carrion
uid: ana
uidNumber: 1002
```

a. Se puede crear un archivo de grupos y otro archivo de usuarios para una mayor administración.

16. Creamos 2 subgrupos que pueden servir para permitir o denegar el acceso.

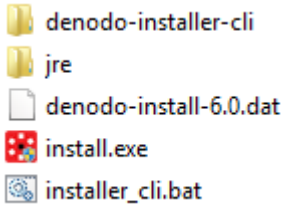
```
dn: cn=hombres,ou=groups,dc=abc,dc=local
objectClass: posixGroup
objectClass: top
cn: hombres
gidNumber: 1001
memberUid: william
memberUid: miguel

dn: cn=mujeres,ou=groups,dc=abc,dc=local
objectClass: posixGroup
objectClass: top
cn: mujeres
gidNumber: 1002
memberUid: jessica
memberUid:ana
```

17. Una vez que tengamos nuestro archivo listo tenemos que agregar esa información a la base de datos de LDAP con el siguiente comando:



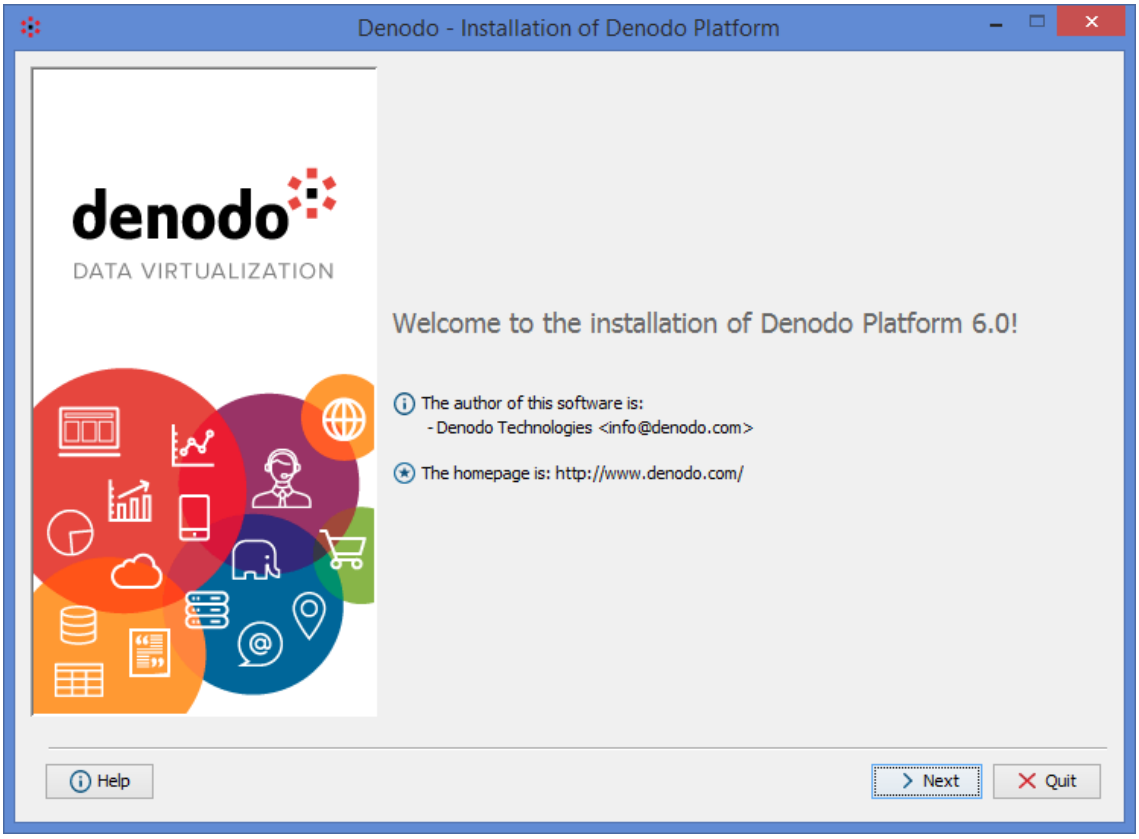
Anexo 8. Instalación de Denodo

El paquete de instalación es un archivo .zip. Después de descomprimir el paquete, verá los archivos que se muestran en la siguiente imagen:



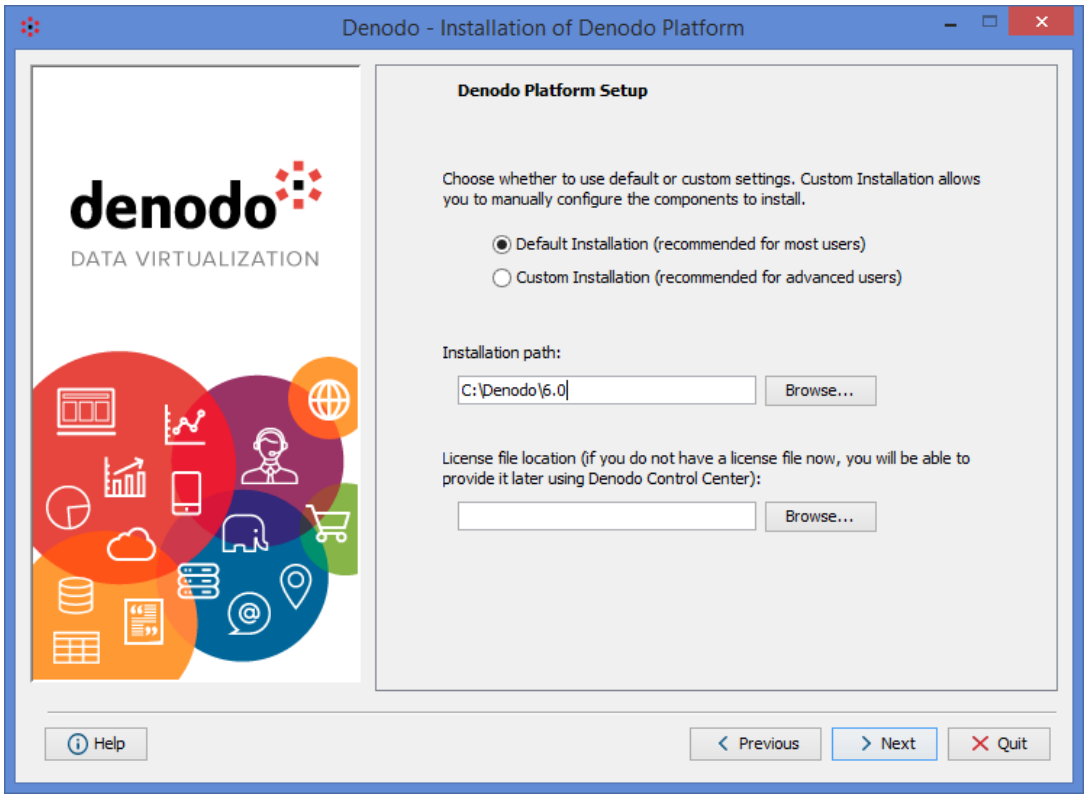
Haga clic en el install.exe archivo y seleccione Ejecutar como administrador.

El programa de instalación se mostrará.

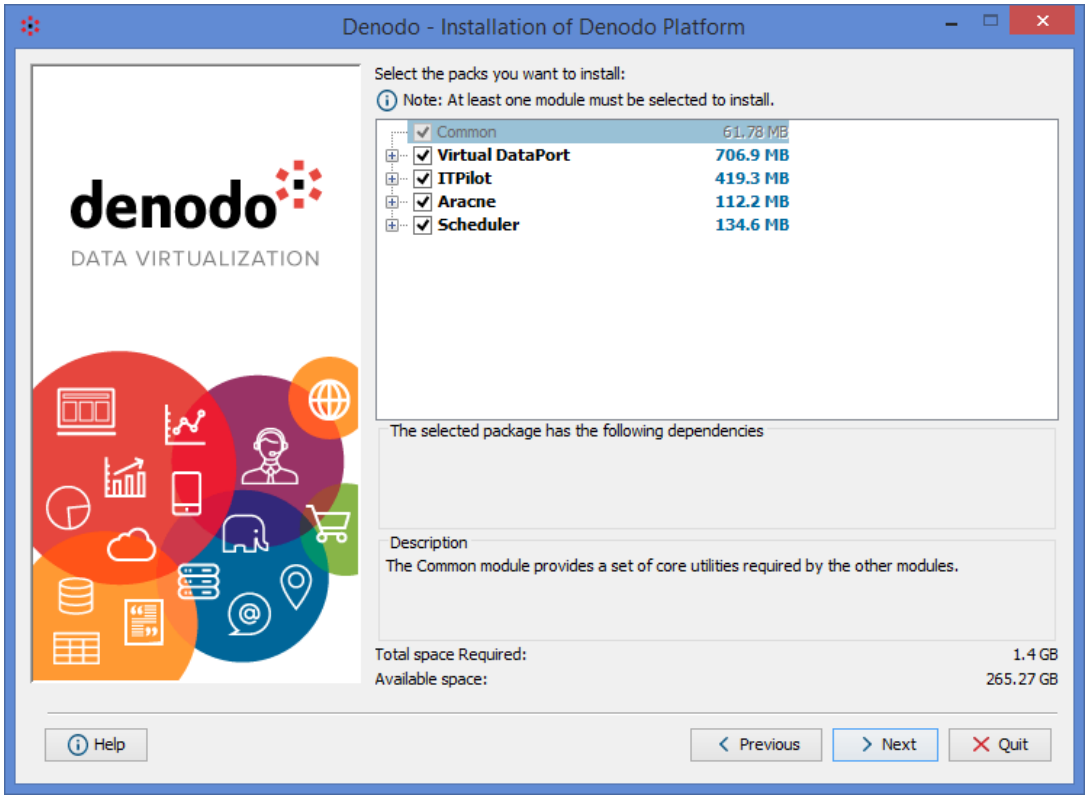


Después de aceptar los términos de la licencia, usted tiene que seleccionar el directorio de instalación (por ejemplo: C: /Denodo/6.0 o /opt/denodo/6.0).

Si ya tiene un archivo de licencia de Denodo, puede seleccionarlo haciendo clic en el botón "Examinar". De lo contrario, puede instalar la licencia posteriormente desde el Centro de Control de Denodo



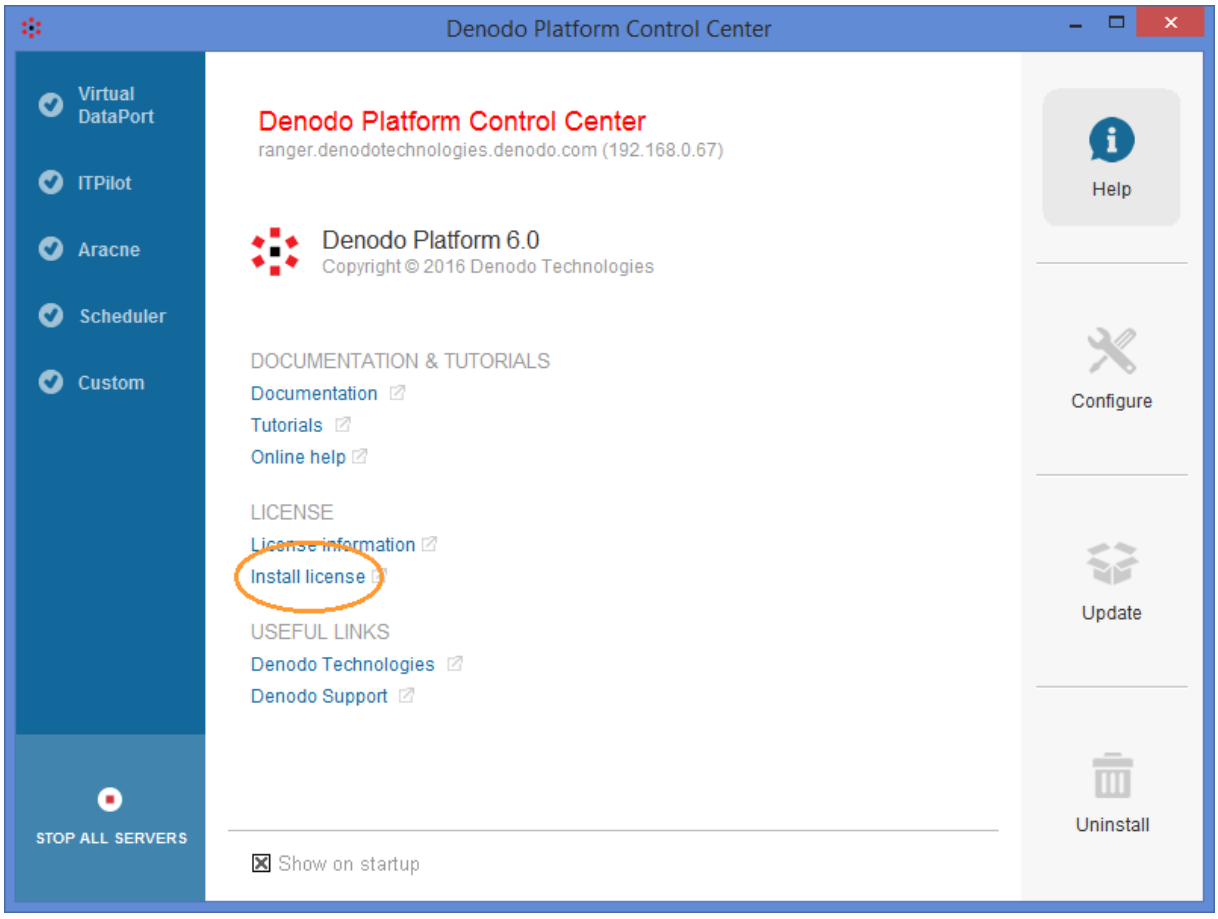
En el siguiente paso tiene que seleccionar los módulos que se instalen. Este tutorial cubre cada módulo de manera que todos ellos se aconseja la instalación, pero se necesita solamente DataPort virtual para empezar.



Puede dejar el resto de las opciones con sus valores por defecto y completar la instalación.

Una vez completada la instalación, puede elegir crear un acceso directo de escritorio que se puede utilizar para iniciar el Centro de control de Denodo.

Si no ha seleccionado ninguna licencia Denodo durante el proceso de instalación, puede hacerlo desde el Centro de Control de Denodo cuando empiece el programa. (Haciendo clic primero en la Ayuda de botón y luego instalar la licencia, se abrirá un nuevo cuadro de diálogo para seleccionar el archivo de licencia y confirme su instalación.



Anexo 9. Plantilla pruebas de usabilidad Bridge IOTMACH

PLAN DE PRUEBAS USABILIDAD

SISTEMA/APLICACIÓN: Bridge IOTMACH

Pruebas (Unitarias/Funcionalidad):	**Fecha**	
Ciclo o Procesos:	Usabilidad del sistema	
Objetivo:	Conocer si el sistema es de fácil entendimiento y uso para un usuario administrador	
Módulos o Programa:	Bridge IOTMACH – Parte administrativa	

Roles que intervienen en la prueba	**Responsable**
Proveedor	Kevin Valarezo
Testeador	

Requerimiento previos a la prueba

Tener un bróker MQTT y Kafka activos.
Tener la dirección URL para acceder al sistema

Descripción de la prueba

Se verificara que tan complejo se hace el uso del sistema para un usuario de tipo administrador

Plan de Pruebas

SI = Satisfactorio
NO = No Satisfactorio

Nº	**Criterios de Prueba**	**Cumple**		**Observación**
		Si	**No**	
1	Es posible observar de forma global lo que abarca el contenido del sistema			
2	La información es suministrada en niveles progresivamente detallados.			
3	¿Los términos usados en el sistema para describir funciones o secciones, indican de forma clara lo que representan?			
4	¿La disposición y localización de los diferentes elementos son mantenidos de forma consistente en todo el sistema?			
5	¿Las acciones a realizar son claramente intuitivas?			
6	¿El sistema está proyectado de forma a minimizar la ocurrencia de errores?			
7	¿El texto de los mensajes de error es significativo e identifica el tipo de problema ocurrido?			
8	¿Tiene el sitio una dirección URL correcta, clara y fácil de recordar?			
9	¿Está claramente indicado el nombre de la página que se está navegando?			

10	¿Los íconos se interpretan su propósito con facilidad?			
11	¿Tiene página de ayuda? ¿Está colocada en una zona visible?			
12	¿Se conoce de forma precisa y completa qué contenidos o servicios ofrece realmente el sistema?			
13	¿La información está ordenada lógicamente?			
14	¿La información mostrada es relevante para el sistema?			
15	¿La información está libre de errores gramaticales y ortográficos?			
16	¿Emplea un lenguaje claro y conciso?			
17	¿Es adecuado el tamaño de la letra utilizada?			
18	¿La imagen o color de fondo ofrece un buen contraste con el tipo de letra?			

Observaciones al Plan de Prueba

Aceptación de la Prueba

Rol	Responsable	Firmas
Proveedor	Sr. Kevin Adrian Valarezo Paz	
Testeador		

Anexo 10. Plantilla pruebas de usabilidad Autenticación IOTMACH

PLAN DE PRUEBAS USABILIDAD

SISTEMA/APLICACIÓN: Autenticación IOTMACH

Pruebas (Unitarias/Funcionalidad):	**Fecha**	
Ciclo o Procesos:	Usabilidad del sistema	
Objetivo:	Conocer si el sistema es de fácil entendimiento y uso para el usuario.	
Módulos o Programa:	Sistema de autenticación – Módulo de Empresa de IOTMACH SERVER	

Roles que intervienen en la prueba	**Responsable**
Proveedor	Kevin Valarezo
Testeador	

Requerimiento previos a la prueba

Tener un servidor LDAP activo.
Tener las aplicaciones a enlazarse activas
Tener usuarios con distintos roles listos para acceder a las aplicaciones

Descripción de la prueba

Se verificara que tan complejo se hace el uso del sistema para cualquier tipo de usuario.

Plan de Pruebas

SI = Satisfactorio
NO = No Satisfactorio

Nº	**Criterios de Prueba**	**Cumple**		**Observación**
		Si	**No**	
1	Es posible observar de forma global lo que abarca el contenido del sistema			
2	¿Los términos usados en el sistema para describir funciones o secciones, indican de forma clara lo que representan?			
3	¿La disposición y localización de los diferentes elementos son mantenidos de forma consistente en todo el sistema?			
4	¿Las acciones a realizar son claramente intuitivas?			
5	¿El sistema está proyectado de forma a minimizar la ocurrencia de errores?			
6	¿El texto de los mensajes de error es significativo e identifica el tipo de problema ocurrido?			
7	¿Se incluye un mapa del sitio?			
8	¿Se puede identificar con rapidez la página que se quiere visitar y llegar fácil y directamente a ella?			
9	¿Tiene el sitio una URL correcta, clara y fácil de recordar?			

10	¿Está claramente indicado el nombre de la página que se está navegando?			
11	¿Los enlaces son fáciles de identificar?			
12	¿Los íconos se interpretan su propósito con facilidad?			
13	¿Tiene página de ayuda? ¿Está colocada en una zona visible?			
14	¿Se informa constantemente al usuario acerca de lo que está pasando?			
15	¿Se conoce de forma precisa y completa qué contenidos o servicios ofrece realmente el sistema?			
16	¿La información está ordenada lógicamente?			
17	¿La información mostrada es relevante para el sistema?			
18	¿La información está libre de errores gramaticales y ortográficos?			
19	¿Emplea un lenguaje claro y conciso?			
20	¿Es adecuado el tamaño de la letra utilizada?			
21	¿La imagen o color de fondo ofrece un buen contraste con el tipo de letra?			

Observaciones al Plan de Prueba

Aceptación de la Prueba

Rol	Responsable	Firmas
Proveedor	Sr. Kevin Adrian Valarezo Paz	
Testeador		

Anexo 11. Plantilla pruebas de Stress

PLAN DE PRUEBAS STRESS

SISTEMA/APLICACIÓN: Bridge IOTMACH

Pruebas de Funcionalidad:	**Fecha**	
Ciclo o Procesos:	Rendimiento del sistema	
Objetivo:	Comprobar el nivel máximo de funcionamiento del sistema	
Módulos o Programa:	Bridge IOTMACH – Núcleo del sistema	

Roles que intervienen en la prueba	**Responsable**
Testeador	Kevin Valarezo

Requerimiento previos a la prueba

Tener un bróker MQTT activo.
Tener un bróker Kafka activo.
Tener emuladores de dispositivos WSN emitiendo mensajes al bróker MQTT.

Descripción de la prueba

Se comprobará el funcionamiento del sistema, a través de situaciones exigentes para la aplicación

Plan de Pruebas

Nº	**Criterios de Prueba**	**Tiempo de Respuesta**	**Observación**
1	Recepción de mensajes en el bróker MQTT y los envía a Kafka por un tiempo determinado.		
2	Recepción de mensajes en el bróker Kafka y los envía a MQTT por un tiempo determinado.		
3	Almacenamiento de 10 documentos en MongoDB sin pool de conexiones.		
4	Almacenamiento de 100 documentos en MongoDB sin pool de conexiones.		
5	Almacenamiento de 1000 documentos en MongoDB sin pool de conexiones.		
6	Almacenamiento de 10000 documentos en MongoDB sin pool de conexiones.		
7	Almacenamiento de 10 documentos en MongoDB con pool de conexiones.		

8	Almacenamiento de 100 documentos en MongoDB con pool de conexiones.		
9	Almacenamiento de 1000 documentos en MongoDB con pool de conexiones.		
10	Almacenamiento de 10000 documentos en MongoDB con pool de conexiones.		
11	Almacenamiento de 100000 documentos en MongoDB con pool de conexiones.		

Observaciones al Plan de Prueba

Aceptación de la Prueba

Rol	Responsable	Firmas
Testeador	Sr. Kevin Adrian Valarezo Paz	

Anexo 12. Plantilla pruebas de seguridad

PLAN DE PRUEBAS SEGURIDAD

SISTEMA/APLICACIÓN: Autenticación IOTMACH

Pruebas (Unitarias/Funcionalidad):	**Fecha**	
Ciclo o Procesos:	Rendimiento del sistema	
Objetivo:	Comprobar el control de acceso de los usuarios a las aplicaciones de IOTMACH	
Módulos o Programa:	Autenticación IOTMACH	

Roles que intervienen en la prueba	**Responsable**
Proveedor	Kevin Valarezo
Testeador	

Requerimiento previos a la prueba

Tener un servidor LDAP activo.
Tener las aplicaciones a enlazarse activas
Tener usuarios con distintos roles listos para acceder a las aplicaciones

Descripción de la prueba

Se verificara si existe el control de acceso correcto de los usuarios que manipularan las aplicaciones de IOTMACH, y a su vez que no exista información vulnerable para usuarios que no tienen privilegios.

Plan de Pruebas

SI = Satisfactorio
NO = No Satisfactorio

Nº	**Criterios de Prueba**	**Cumple**		**Observación**
		Si	**No**	
1	Funciona correctamente el inicio de sesión con las credenciales de usuario y contraseña			
2	Muestra información relacionada con el usuario que inicio sesión			
3	La información vulnerable de la empresa a usuarios sin privilegios esta oculta correctamente.			
4	Los usuarios solo pueden acceder a las funciones que le fueron designadas.			
5	Identifica al usuario con sus respectivos privilegios de acceso a las aplicaciones.			
6	El usuario no puede acceder a una aplicación a la cual no tenga permiso.			
7	Un usuario no puede ver la información de otro usuario.			

8	Las cookies del navegador están protegidas.			
9	Dentro de la dirección URL no se muestra información considerada como vulnerabilidad			
10	Dentro de IOTMACH Server, el usuario solo puede realizar las funciones asignadas por sus respectivos roles.			
11	¿Existe niveles de contraseña segura?			

Observaciones al Plan de Prueba

Aceptación de la Prueba

Rol	Responsable	Firmas
Proveedor	Sr. Kevin Adrian Valarezo Paz	
Testeador		

Anexo 13. Plantilla pruebas de integración

Responsables		**Fecha**	

Requisitos Previos	

Escenario	

Caso de Prueba	

		Componentes				
Criterios	**Descripción**					**Observación**
		Satisfactorio	**No Satisfactorio**	**Satisfactorio**	**No Satisfactorio**	
Comunicación	Existe comunicación entre las dos aplicaciones.					
Integridad de la Información	Los paquetes fueron enviados y recibidos completamente					
Tiempo de Respuesta	Tiempo de envío/recepción de los datos es aceptable					
Autenticación de Conexión	Existe un proceso de autenticación en la base de datos.					
Información Correcta	Los datos contienen la información correcta					
Base de datos	No existen problemas con la conexión a la base de datos.					

Anexo 14. Información de personas que realizaron las pruebas

Las siguientes personas realizaron las pruebas efectuadas a las aplicaciones desarrolladas, ya que tienen el conocimiento necesario para dar su criterio respecto a los procesos que contiene el prototipo.

#	Nombres y Apellidos	Teléfono
1	Danilo Antonio Sánchez Chuico	0987176673
2	Juan Francisco Morocho Ajila	0986411943
3	Gustavo Eduardo Belduma Vacacela	0982344652
4	Jennifer Estefanía Honores Cún	0990171512
5	María Fernanda Landín Pacheco	0988674566

Anexo 15. Condensado de pruebas de usabilidad de Bridge IOTMACH.

		Testers					Conteo	
#	Descripción	1	2	3	4	5	SI	NO
1	Es posible observar de forma global lo que abarca el contenido del sistema	SI	SI	SI	SI	SI	5	0
2	La información es suministrada en niveles progresivamente detallados.	SI	SI	NO	NO	SI	3	2
3	¿Los términos usados en el sistema para describir funciones o secciones, indican de forma clara lo que representan?	NO	NO	NO	SI	NO	1	4
4	¿La disposición y localización de los diferentes elementos son mantenidos de forma consistente en todo el sistema?	SI	SI	SI	SI	SI	5	0
5	¿Las acciones a realizar son claramente intuitivas?	SI	SI	SI	SI	SI	5	0
6	¿El sistema está proyectado de forma a minimizar la ocurrencia de errores?	SI	SI	SI	SI	SI	5	0
7	¿El texto de los mensajes de error es significativo e identifica el tipo de problema ocurrido?	SI	SI	SI	SI	SI	5	0
8	¿Tiene el sitio una dirección URL correcta, clara y fácil de recordar?	NO	SI	NO	NO	NO	1	4
9	¿Está claramente indicado el nombre de la página que se está navegando?	SI	SI	SI	NO	SI	4	1
10	¿Los íconos se interpretan su propósito con facilidad?	SI	SI	SI	SI	SI	5	0
11	¿Tiene página de ayuda? ¿Está colocada en una zona visible?	NO	NO	NO	NO	NO	0	5
12	¿Se conoce de forma precisa y completa qué contenidos o servicios ofrece realmente el sistema?	SI	SI	SI	NO	NO	3	2
13	¿La información está ordenada lógicamente?	SI	SI	NO	SI	SI	4	1
14	¿La información mostrada es relevante para el sistema?	SI	SI	SI	SI	SI	5	0
15	¿La información está libre de errores gramaticales y ortográficos?	NO	NO	NO	NO	NO	0	5
16	¿Emplea un lenguaje claro y conciso?	SI	SI	SI	SI	SI	5	0
17	¿Es adecuado el tamaño de la letra utilizada?	SI	SI	SI	SI	SI	5	0
18	¿La imagen o color de fondo ofrece un buen contraste con el tipo de letra?	SI	SI	SI	NO	SI	4	1
		Total					65	25

Si = Satisfactorio

No = No Satisfactorio

DATOS	
(a) Número de Testers	5
(b) Número de Criterios	18
Total Respuestas (a*b)	90

Usabilidad	N° Respuestas	Porcentaje
Satisfactorio	65	72%
No Satisfactorio	25	28%
Total	90	100%

Anexo 16. Condensado de pruebas de usabilidad de seguridad y autenticación de IOTMACH.

#	Descripción	Testers					Conteo	
		1	2	3	4	5	SI	NO
1	Es posible observar de forma global lo que abarca el contenido del sistema	SI	SI	SI	SI	SI	5	0
2	¿Los términos usados en el sistema para describir funciones o secciones, indican de forma clara lo que representan?	SI	SI	SI	SI	NO	4	1
3	¿La disposición y localización de los diferentes elementos son mantenidos de forma consistente en todo el sistema?	SI	SI	SI	SI	SI	5	0
4	¿Las acciones a realizar son claramente intuitivas?	SI	SI	SI	SI	NO	4	1
5	¿El sistema está proyectado de forma a minimizar la ocurrencia de errores?	SI	SI	SI	SI	SI	5	0
6	¿El texto de los mensajes de error es significativo e identifica el tipo de problema ocurrido?	SI	NO	SI	NO	SI	3	2
7	¿Se incluye un mapa del sitio?	NO	SI	SI	SI	NO	3	2
8	¿Se puede identificar con rapidez la página que se quiere visitar y llegar fácil y directamente a ella?	SI	SI	SI	SI	SI	5	0
9	¿Tiene el sitio una URL correcta, clara y fácil de recordar?	NO	NO	NO	NO	NO	0	5
10	¿Está claramente indicado el nombre de la página que se está navegando?	SI	SI	NO	SI	SI	4	1
11	¿Los enlaces son fáciles de identificar?	NO	SI	SI	SI	NO	3	2
12	¿Los íconos se interpretan su propósito con facilidad?	SI	SI	SI	SI	SI	5	0
13	¿Tiene página de ayuda? ¿Está colocada en una zona visible?	NO	NO	NO	NO	NO	0	5
14	¿Se informa constantemente al usuario acerca de lo que está pasando?	SI	SI	NO	SI	SI	4	1
15	¿Se conoce de forma precisa y completa qué contenidos o servicios ofrece realmente el sistema?	SI	SI	SI	SI	SI	5	0
16	¿La información está ordenada lógicamente?	NO	SI	SI	SI	NO	3	2
17	¿La información mostrada es relevante para el sistema?	SI	SI	SI	SI	SI	5	0
18	¿La información está libre de errores gramaticales y ortográficos?	NO	NO	NO	NO	NO	0	5
19	¿Emplea un lenguaje claro y conciso?	SI	SI	SI	SI	NO	4	1
20	¿Es adecuado el tamaño de la letra utilizada?	SI	SI	SI	SI	SI	5	0
21	¿La imagen o color de fondo ofrece un buen contraste con el tipo de letra?	SI	SI	SI	SI	SI	5	0
		Total					77	28

Si = Satisfactorio No = No Satisfactorio

DATOS	
(a) Número de Testers	5
(b) Número de Criterios	21
Total Respuestas (a*b)	105

Usabilidad	N° Respuestas	Porcentaje
Satisfactorio	77	73%
No Satisfactorio	28	27%
Total	105	100%

Anexo 17. Condensado de pruebas de seguridad de Autenticación IOTMACH.

#	Descripción	Testers					Conteo	
		1	2	3	4	5	SI	NO
1	Funciona correctamente el inicio de sesión con las credenciales de usuario y contraseña	SI	SI	SI	SI	SI	5	0
2	Muestra información relacionada con el usuario que inicio sesión	SI	SI	SI	SI	SI	5	0
3	La información vulnerable de la empresa a usuarios sin privilegios esta oculta correctamente.	SI	SI	NO	SI	SI	4	1
4	Los usuarios solo pueden acceder a las funciones que le fueron designadas.	SI	SI	SI	SI	SI	5	0
5	Identifica al usuario con sus respectivos privilegios de acceso a las aplicaciones.	SI	SI	SI	SI	SI	5	0
6	El usuario no puede acceder a una aplicación a la cual no tenga permiso.	SI	SI	NO	NO	NO	2	3
7	Un usuario no puede ver la información de otro usuario.	SI	NO	SI	SI	SI	4	1
8	Las cookies del navegador están protegidas.	SI	SI	SI	SI	SI	5	0
9	Dentro de la dirección URL no se muestra información considerada como vulnerabilidad	SI	SI	NO	SI	SI	4	1
10	Dentro de IOTMACH Server, el usuario solo puede realizar las funciones asignadas por sus respectivos roles.	SI	SI	SI	SI	SI	5	0
11	¿Existe niveles de contraseña segura?	NO	NO	NO	NO	NO	0	5
		Total					44	11

Si = Satisfactorio No = No Satisfactorio

DATOS	
(a) Número de Testers	5
(b) Número de Criterios	11
Total Respuestas (a*b)	55

Seguridad	N° Respuestas	Porcentaje
Satisfactorio	44	80%
No Satisfactorio	11	20%
Total	55	100%

Anexo 18. Resultados Pruebas de Integración con Gateway IOTMACH Móvil.

Responsables	- Gustavo Belduma - Kevin Valarezo	**Fecha**	28/07/2016

Requisitos Previos	Estar conectados a una red común Manejar tópicos comunes entre las aplicaciones

Escenario	Conexión entre la aplicación Gateway IOT Móvil con Bridge IOTMACH, enviando los datos emitidos por la WSN hacia el Centro de Procesamiento de Datos. En el cuál el gateway utiliza el protocolo Eddystone para la recolección de los datos enviados por los motes y los envía hacia el Bridge IOTMACH.

Caso de Prueba	Conectividad e integración entre Gateway IOT Móvil y Bridge IOTMACH

Criterios	**Descripción**	**Componentes**				**Observación**
		Gateway IOT - Android		**Bridge IOTMACH**		
		Satisfactorio	**No Satisfactorio**	**Satisfactorio**	**No Satisfactorio**	
Comunicación	Existe comunicación entre las dos aplicaciones	X		X		
Integridad de la Información	Los paquetes fueron enviados y recibidos completamente	X		X		
Tiempo de Respuesta	Tiempo de envío/recepción de los datos es aceptable	X		X		
Autenticación de Conexión	Existe un proceso de autenticación para comunicar el envío/recepción de datos		X	X		Cargar configuración en la aplicación Android para implementar autenticación.
Información Correcta	Los datos contienen la información correcta		X		X	Corregir parámetros de la información.
Base de datos	Se presentaron problemas con la conexión a la base de datos.		X		X	

Datos	
(a) # de Apps	2
(b) # de Criterios	6
Total (a) * (b)	12

Integración	**Conteo**	**Porcentaje**
Satisfactorio	7	58%
No Satisfactorio	5	42%
Total	12	100%

Anexo 19. Resultados Pruebas de Integración con IOTMACH Server

Responsables	- Danilo Sánchez - Kevin Valarezo	**Fecha**	28/07/2016

Requisitos Previos	Estar conectados a una red común Tener conexión con una base de datos no relacional en común.

Escenario	Conexión entre Bridge IOTMACH con IOTMACH Server a través de MongoDB, en el que Bridge IOTMACH recibe datos y los almacena en la base de datos, mientras tanto IOTMACH Server accede a dicha base de datos para ir mostrando la información para posteriormente irla monitoreando.

Caso de Prueba	Conectividad e integración entre Bridge IOTMACH e IOTMACH Server

Criterios	**Descripción**	**Componentes**				**Observación**
		IOTMACH Server		**Bridge IOTMACH**		
		Satisfactorio	**No Satisfactorio**	**Satisfactorio**	**No Satisfactorio**	
Comunicación	Existe comunicación entre las dos aplicaciones.		X		X	No debería existir comunicación directa entre aplicaciones, sin embargo su comunicación es a través de MongoDB.
Integridad de la Información	Los paquetes fueron enviados y recibidos completamente	X		X		
Tiempo de Respuesta	Tiempo de envío/recepción de los datos es aceptable	X		X		
Autenticación de Conexión	Existe un proceso de autenticación en la base de datos.		X		X	Crear conexiones con la base de datos utilizando credenciales de usuarios.
Información Correcta	Los datos contienen la información correcta	X		X		
Base de datos	No existen problemas con la conexión a la base de datos.	X		X		Debería contar con credenciales de acceso

Datos	
(a) # de Apps	2
(b) # de Criterios	6
Total (a) * (b)	12

Integración	**Conteo**	**Porcentaje**
Satisfactorio	8	67%
No Satisfactorio	4	33%
Total	12	100%

Anexo 20. Resultados Pruebas de Integración con Gateway IOTMACH

Responsables	- Jorge Prado - Kevin Valarezo	**Fecha**	10/08/2016

Requisitos Previos	Estar conectados a una red común Manejar tópicos comunes entre las aplicaciones

Escenario	Conexión entre la aplicación Gateway IOT con Brdige IOTMACH, enviando los datos emitidos por la WSN hacia el Centro de Procesamiento de Datos. En el cuál el gateway utiliza el protocolo Eddystone para la recolección de los datos enviados por los motes y los envía hacia el Bridge IOTMACH.

Caso de Prueba	Conectividad e integración entre Gateway IOT Móvil y Bridge IOTMACH

Criterios	**Descripción**	**Componentes**				**Observación**
		Gateway IOT		**Bridge IOTMACH**		
		Satisfactorio	**No Satisfactorio**	**Satisfactorio**	**No Satisfactorio**	
Comunicación	Existe comunicación entre las dos aplicaciones	X		X		
Integridad de la Información	Los paquetes fueron enviados y recibidos completamente	X		X		
Tiempo de Respuesta	Tiempo de envío/recepción de los datos es aceptable	X		X		
Autenticación de Conexión	Existe un proceso de autenticación para comunicar el envío/recepción de datos	X		X		Cargar configuración en la aplicación Android para implementar autenticación.
Información Correcta	Los datos contienen la información correcta	X		X		Corregir parámetros de la información.
Base de datos	Se presentaron problemas con la conexión a la base de datos.		X		X	

Datos	
(a) # de Apps	2
(b) # de Criterios	6
Total (a) * (b)	12

Integración	**Conteo**	**Porcentaje**
Satisfactorio	10	83%
No Satisfactorio	2	17%
Total	12	100%

Anexo 21. Resultados Pruebas de Integración con IOTMACH Server

Responsables	- Danilo Sánchez - Kevin Valarezo	**Fecha**	10/08/2016

Requisitos Previos	Estar conectados a un mismo dominio Tener servicios necesarios levantados.

Escenario	Acceder a través del sistema de autenticación hacia la aplicación IOTMACH Server.

Caso de Prueba	Integración entre Autenticación IOTMACH e IOTMACH Server

Criterios	**Descripción**	**Componentes**				**Observación**
		IOTMACH Server		**Autenticación IOTMACH**		
		Satisfactorio	**No Satisfactorio**	**Satisfactorio**	**No Satisfactorio**	
Comunicación	Existe comunicación entre las aplicaciones.	X		X		
Integridad de la Información	La información no se encuentra expuesta para ningún usuario sin credenciales.	X			X	Controlar el cierre de sesión, ya que no se están eliminando correctamente las cookies.
Autenticación	Existe autenticación única entre las aplicaciones.	X		X		
Base de datos	No existen problemas con la conexión a la base de datos.	X		X		

Datos	
(a) # de Apps	2
(b) # de Criterios	4
Total (a) * (b)	12

Integración	**Conteo**	**Porcentaje**
Satisfactorio	7	88%
No Satisfactorio	1	13%
Total	8	100%