IMPLEMENTACIÓN DE UNA ARQUITECTURA *FOG COMPUTING*

JAVIER PINZÓN CASTELLANOS

UNIVERSIDAD AUTÓNOMA DE BUCARAMANGA – UNAB
FACULTAD DE INGENIERÍA
MAESTRÍA EN GESTIÓN, APLICACIÓN Y DESARROLLO DE SOFTWARE
BUCARAMANGA
2020

IMPLEMENTACIÓN DE UNA ARQUITECTURA *FOG COMPUTING*

JAVIER PINZÓN CASTELLANOS

Proyecto de Grado de la Maestría en Gestión, Aplicación y Desarrollo de Software

Director
Msc. Miguel Cadena Carter

UNIVERSIDAD AUTÓNOMA DE BUCARAMANGA – UNAB
FACULTAD DE INGENIERÍA
MAESTRÍA EN GESTIÓN, APLICACIÓN Y DESARROLLO DE SOFTWARE
BUCARAMANGA
2020

DEDICATORIA

A mis padres por ser mi apoyo fundamental en todo
mi proceso académico y en la vida.
Éste trabajo ha sido posible gracias a ellos,
por confiar en mí.

AGRADECIMIENTOS

El autor desea agradecer la colaboración de todos los socios dentro del proyecto Centro de Excelencia y Apropiación en Internet de las Cosas - CEA-IoT. El autor también desea agradecer a todas las instituciones que apoyaron este trabajo: el Ministerio de Tecnología de la Información y Comunicaciones - MinTIC de Colombia y al Departamento Administrativo de Ciencia, Tecnología e Innovación - Colciencias, a través del Fondo Nacional de Financiamiento para la Ciencia, Tecnología y la Innovación Francisco José de Caldas - ID Proyecto: FP44842-502-2015.

RESUMEN DEL PROYECTO

Fog Computing ó La computación de niebla ha despertado gran interés en la academia, debido a la separación de procesos que se realizan en *Cloud Computing* (la computación en la nube) permitiendo esto el ahorro de costos en las tecnologías de la información y las comunicaciones. Sin embargo, la tendencia del *Cloud Computing* ha ido creciendo en el procesamiento y el almacenamiento por consiguiente los precios también. Por ello se diseña una arquitectura en el cual logra realizar un procesamiento local que ayude a disminuir el consumo de la nube. Con base en lo anterior, se realizó una investigación utilizando la arquitectura de *Fog Computing* que consiste en realizar la implementación de este con *hardware* y *software* libre, identificando los diferentes módulos que se utilizan para comunicarse e integrar módulos de *software.* Es por esto, que se realiza un estado del arte de la arquitectura y un proceso de implementación con metas académicas esto con el fin de entender el comportamiento, su operación y la separación entre las nuevas tendencias de arquitectura como *Internet of Things* (internet de las cosas) y *Dew Computing* (computación de lagrima), dentro de una documentación de la nueva arquitectura tecnológica.

Palabras clave ---- *Fog Computing, Internet of Things, IoT*, *Cloud Computing*.

CONTENIDO

pág.

LISTADO DE FIGURAS

pág.

LISTADO DE TABLAS

pág.

INTRODUCCIÓN

El presente trabajo de investigación es resultado de la labor llevada a cabo como trabajo de grado, donde se desarrolla los aportes potenciales del *Fog Computing* a las diferentes tecnologías, con base en comparaciones e implementaciones realizadas a partir de la revisión bibliográfica y de la realización pruebas en diferentes contextos de aplicación, en el marco del programa de Maestría en Gestión, Aplicación y Desarrollo de *Software* de la Universidad Autónoma de Bucaramanga – UNAB. Éste trabajo hace un aporte a la academia encaminado a la utilización de la arquitectura *Fog Computing*; Como base para futuros proyectos de investigación realizados por el autor, con el propósito de generar interés en este tema y de esta manera favorecer a la comunidad científica en el área.

En la actualidad se están presentando nuevos paradigmas de computación distribuida inteligente, un adjetivo para referirnos a una arquitectura que dispone de un desempeño de apoyo en una labor. También el término inteligente se encuentra con una estrecha vinculación a la manera de realizar bien las actividades, analizar y entender situaciones que requieren.

Para ello se planteó el problema de investigación, propósito y enfoques, resaltando los objetivos, que posteriormente establecen las metas alcanzadas con el desarrollo del proyecto.

En el marco teórico se estudiaron los trabajos de investigación desarrollados sobre la arquitectura de interés. Integrado por el marco conceptual; en el cual se encontraron los fundamentos del proyecto: *Cloud Computing, Fog Computing*, *Dew Computing*, *Internet of Things* (IoT). Seguido de esto se mostró el marco contextual, el cual presenta el enfoque del proyecto y donde se llevaron a cabo los escenarios de desarrollo. El marco legal evidenció los aspectos normativos y los estándares de utilización de la arquitectura, entre otras. Finalmente, el estado del arte recopiló la trayectoria y tendencias sobre el tópico, dando como resultado el marco referencial, relevante para el desarrollo del proyecto.

La descripción del proceso de investigación consistió en el planteamiento de la plataforma, especificaciones, descripción de casos de uso, diagrama de casos, arquitectura general y, por último, la dinámica de peticiones.

Los resultados obtenidos en los objetivos y fases definidas en la investigación, tales como; un estado del arte que contiene la revisión de la literatura sobre la arquitectura, una plataforma específica en cuanto a características de *hardware* y *software,* la configuración de cada uno de los componentes, con sus respectivas instrucciones y herramientas utilizadas, así como las mediciones de tráfico dentro de la arquitectura y las aplicaciones de *Fog Computing* dentro de los diferentes escenarios tecnológicos.

El impacto de la investigación evidenció un estudio donde se presentó una nueva arquitectura diferente a la computación en la nube resaltando su importancia, de manera que se puedan compartir los datos entre las dos arquitecturas y el usuario final compruebe, el aumento de velocidad de los datos y reducción de los tiempos carga y descarga.

El estudio referente a la arquitectura *Fog Computing,* permitió comprender y dar a conocer este paradigma, ubicándolo dentro del contexto computacional logrando un impacto conceptual en la comunidad académica.

Fog Computing ha despertado gran interés en la academia, debido a la separación de procesos que se realizan en *Cloud Computing*; permitiendo el ahorro de costos en las tecnologías de la información y las comunicaciones. Sin embargo, la tendencia de *Cloud Computing* ha ido creciendo en el procesamiento y almacenamiento, a causa de esto, los precios han aumentado, y teniendo en cuenta que el costo es un factor representativo dentro de los procesos, para mantener un sistema de computación escalable. Por ello se presentó una arquitectura en la cual se logró realizar un procesamiento local que ayudó a disminuir el consumo de la nube.

La arquitectura de capas jerárquicas facilita el desarrollo de los sistemas distribuidos y generó nuevas características, entre ellas el rendimiento, el cual optimizó el

procesamiento, logrando así reducir el tiempo de retardo de los datos, y la disponibilidad requerida en el caso de que el sistema falle; Un sistema de recuperación restaurará los datos de manera rápida y oportuna.

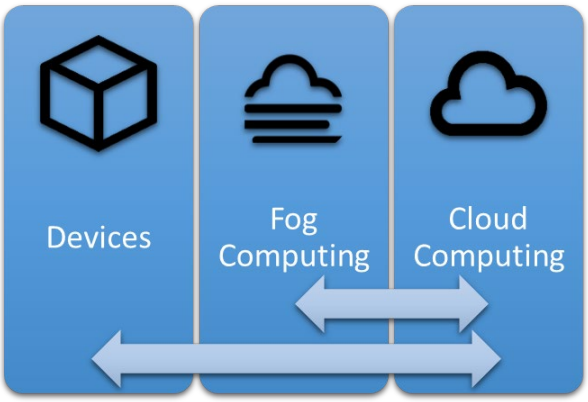
La confiabilidad del sistema requiere fiabilidad tanto en los datos como en la funcionalidad de este.

A partir de los resultados obtenidos se muestran las conclusiones y recomendaciones a las cuales ha llegado el investigador.

1. PROBLEMA DE INVESTIGACIÓN, PROPÓSITO Y ENFOQUES

La investigación plantea que, debido a la alta demanda de servicios de nube, dentro de la que se encuentra *IoT;* se están generando grandes volúmenes de procesamiento, almacenamiento y tráfico de datos; por consiguiente, se requiere un mecanismo que permita la escalabilidad donde existan multiniveles que generen recursos como procesamiento y almacenamiento en cada nivel. Dando lugar a la nueva jerarquía de computación distribuida, la cual está constituida de la siguiente manera: Computación en la nube (*Cloud Computing*), Computación de Niebla (*Fog Computing*) y Computación de Rocío o Lágrima (*Dew Computing*), (organizadas de nivel superior al inferior respectivamente). De esta manera la capa de *Fog Computing* permite disminuir el tráfico entre el usuario y la nube. Por ello se plantea cómo realizar una arquitectura de *Fog Computing*, para disminuir el uso de los servicios de la nube y redireccionarlos a un componente de bajo costo, que esté más cerca al usuario final y pueda accederse de manera eficiente y transparente.

Figura 1. Comparativo entre las capas de computación estructurada



Fuente (Pavel Pohanka, 2015)

El propósito de la investigación es realizar un estudio mostrando una nueva arquitectura diferente a la nube resaltando su importancia, de manera que se puedan compartir los datos entre las dos arquitecturas y el usuario final evidencie, el aumento de velocidad de los datos y reducción de los tiempos carga y descarga. Las tareas que serán realizadas para llegar a la implementación y estudio de la arquitectura, son las siguientes: Elaboración del estado del arte sobre la arquitectura *fog computing* seguido de una realización de la selección de *hardware* y *software*

de la arquitectura: que servirán para realizar una prueba de concepto. Posterior se configura los componentes de *hardware* y *software* para que operen dentro de la arquitectura. De esta manera se implementará un prototipo, donde se creará un protocolo de pruebas con el fin de evidenciar su funcionamiento. Para finalizar se realizará las mediciones pertinentes en cuanto a tráfico de red, que evidencien la ventaja de *Fog Computing* para la disminución de tráfico de red hacia la nube.

2. OBJETIVOS DEL PROYECTO

2.1 OBJETIVO GENERAL

Implementar una arquitectura *Fog Computing* con tecnología libre para fines académicos.

2.2 OBJETIVOS ESPECÍFICOS

- Elaborar un estado del arte referente a *Fog Computing*.
- Seleccionar una herramienta de *Hardware* y *Software* libre mediante un análisis comparativo que permita la implementación de *Fog Computing*.
- Realizar la configuración de los componentes de *Hardware* y *Software* para que operen dentro de una arquitectura *Fog Computing*.
- Realizar las mediciones pertinentes en cuanto a tráfico de red, que evidencien la ventaja de *Fog Computing* para la disminución de tráfico de red hacia la nube.

3. MARCO REFERENCIAL

Es presentado el marco referencial del trabajo de investigación desarrollados en el marco conceptual, teórico, contextual, legal y estado del arte del proyecto.

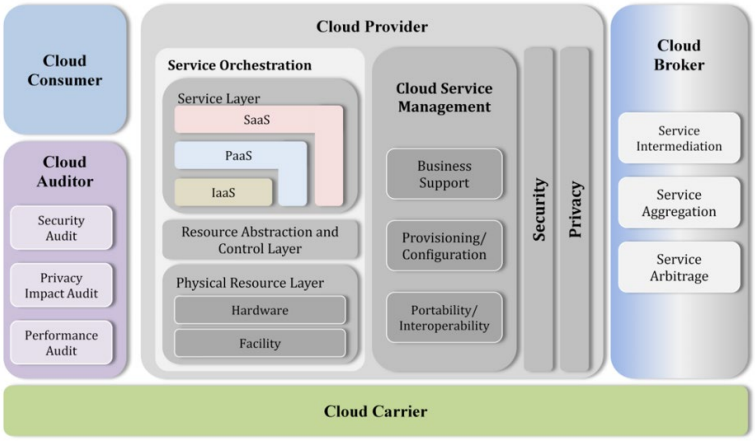
3.1 MARCO CONCEPTUAL

Son presentados los conceptos utilizados para trabajo de investigación.

3.1.1 *Cloud Computing*.

Término en inglés para “computación en la nube”, es un concepto tecnológico utilizado para definir la adopción de la arquitectura computación en la nube, que utiliza diferentes modelos de servicio: *Software* como Servicio (SaaS), Plataforma como Servicio (PaaS) e Infraestructura como Servicio (IaaS). El crecimiento de los modelos de entrega o despliegue de la nube (privada, pública, híbrida y comunitaria) ofrecidos por multitud de proveedores, han hecho habituales la terminología para las estrategias empresariales o centros de investigación. En la figura 2. se muestra la arquitectura de referencia conceptual de *Cloud Computing*, localizando los usuarios auditores de nube, el proveedor, conectores de nube, donde todo en conjunto es un operador de nube.

Figura 2. Arquitectura de referencia conceptual de CC.

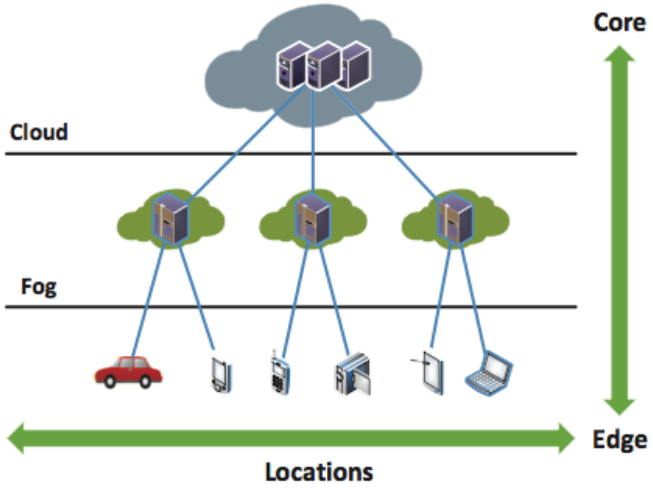


Fuente (Hurwitz, Bloor, Kaufman, & Halper, 2010).

3.1.2 *Fog Computing*.

Término en inglés para “computación de niebla o borde”, es un concepto tecnológico utilizado para definir la adopción de una arquitectura preparada para realizar procesamiento de datos y aplicaciones en el borde de la red, en vez de utilizar la nube. Aunque la conexión con la nube no se pierde, esto facilita el manejo de los componentes de la nube como, almacenamiento y administración de red. Donde los elementos tecnológicos conectados al *Fog Computing* son más rápidos utilizando esta herramienta, reduciendo la carga de los centros de datos y logrando mejores tiempos de respuesta (Skala, Davidovic, Afgan, Sovic, & Sojat, 2015).

Figura 3. Arquitectura *Fog* y *Cloud Computing*.



Fuente: (Stojmenovic & Wen, 2014).

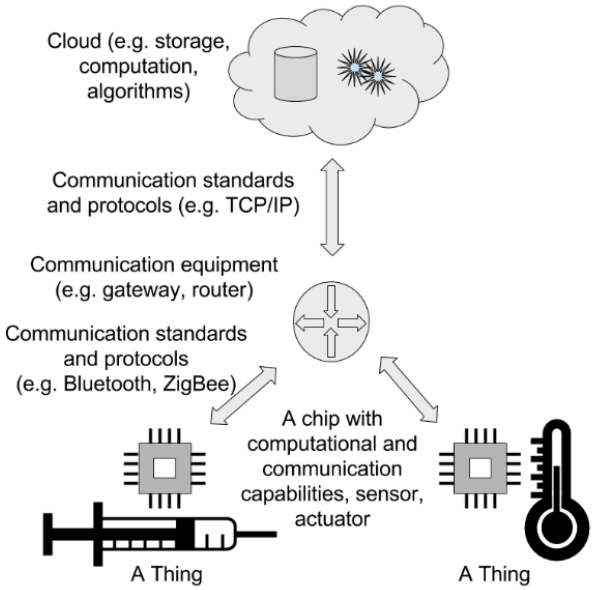
3.1.3 *Dew Computing*.

Término en inglés para “computación de rocío o lagrima”, es un concepto tecnológico utilizado para definir la adopción de la comunicación entre dos dispositivos sin tener acceso a internet, es decir, una conexión punto a punto. La cuál se orienta a la generalización de la terminología, sin importar el protocolo que se utilice entre ellos como: *Bluetooth, RFID, Wifi, NFC*, etc. La utilización del término *Dew Computing* permite la comparación entre las diferentes capas de conectividad y así poder diferenciarlas al momento de identificar una arquitectura. En la figura 5, se puede observar las distintas arquitecturas de computación, en la cual se resalta la diferencia en la conectividad y el funcionamiento de cada una.(Dr. Deepti Sharma, 2015)

3.1.4 *Internet of Things*.

IoT por sus siglas en inglés, término para el “Internet de las Cosas”, es un concepto que se refiere a la interconexión digital de objetos cotidianos con la Internet. El Internet de las cosas se encuentra en el punto en el cual se conectarían a internet más “cosas u objetos” que personas. Si los objetos de la vida cotidiana tuvieran incorporadas etiquetas, podrían ser identificados y gestionados por otros equipos, de la misma manera que si lo fuesen por seres humanos. (Chang, Chen, Chen, & Chao, 2012)

Figura 4. Estructura de *IoT*



Fuente: (Szilagyi & Wira, 2016).

3.2 MARCO TEORICO

El marco teórico de la investigación fue enfocado en la teoría, que explica el concepto de *Fog Computing*.

Las nuevas generaciones de dispositivos inteligentes son capaces de procesar datos localmente en lugar de enviarlos a la nube a cualquier otro sistema computacional para procesamiento, esto permite una nueva área de posibilidades

como El *Fog Computing*, que es una arquitectura de computación que proporciona servicios de datos, computación, almacenamiento y aplicaciones más cerca de los dispositivos del cliente o del usuario. Como enrutadores (*routers*) de red entre otros. Esta arquitectura permite gestionar datos a nivel de red, con dispositivos inteligentes y en el lado del usuario final (por ejemplo, dispositivos móviles), en lugar de enviar datos a una ubicación en la nube para su procesamiento.

El objetivo de la *Fog Computing* es mejorar la eficiencia mediante el procesamiento directo de datos en la red (por ejemplo, en enrutadores (*routers*) de red o dispositivos inteligentes como computadores de placa reducida o única en inglés: *Single Board Computer* o *SBC)*, reduciendo así la cantidad de datos que hay que transferir para el procesamiento, pero también mantener los datos más cercanos al usuario aumentando así la seguridad que es uno de los principales problemas en *Cloud Computing*. Las características distintivas de *Fog Computing* son su proximidad a los usuarios finales, su distribución geográfica (puede ubicarse lo más cerca posible al usuario), la capacidad de reconfigurarse y su apoyo a la movilidad, así como su potencial para mejorar la seguridad y servicios.

El *Fog Computing* se ubica en el borde de la red, lo que reduce el retardo y mejora la calidad del servicio (*QoS*), lo que da como resultado un nivel superior de funcionalidad de computación.

Al ser una nueva solución conceptual para abordar las demandas del creciente número de dispositivos orientados a Internet, denominado Internet de las Cosas (*IoT*). El término cosa se refiere a cualquier objeto que pueda capturar datos y al que se pueda asignarse una dirección de IP, proporcionándole así la capacidad de transferir datos a través de una red a un entorno de computación en la nube; donde hay un gran número de cosas conectadas a internet que son capaces de producir grandes cantidades de datos, para lo cual el *Fog Computing* desempeñará un papel clave en la reducción en el retardo en las aplicaciones dinámicas basadas en la red. al tener una transferencia de una gran cantidad de datos desde sensores a sistemas de procesamiento a distancia pueden ser ineficiente y posiblemente conduce a una sobrecarga de red innecesaria que puede dar lugar a caídas de rendimiento y seguridad. Siendo *Fog Computing* una opción para la analítica en tiempo real y la generación de conocimiento académico.

En contraste con las arquitecturas *Cloud Computing* y *Fog Computing*, se basa en el procesamiento orientado a la información en lugar de orientado a los datos, lo que requiere nuevos protocolos más flexibles y modelos de programación más productivos. En este sentido, los datos (sin procesar) no son información, mientras que la información son datos con adición de metadatos. Los metadatos colocan a los datos en un contenido específico que permite análisis más extenso.

Las arquitecturas de computación existentes, operan con enormes cantidades de datos brutos generados por las cosas específicas, a través de servicios predefinidos.

Dado que los datos brutos están fuera de contexto, los servicios deben ser adaptados y específicos de la aplicación, lo que requiere decisiones basadas en datos. Por lo que Fog Computing se encarga de esta decisión de procesar los datos y enviarlos o no. Por otro lado, el *Dew Computing* solo es responsable de recopilar y generar los datos brutos, por lo tanto, es el único componente del ecosistema informático que es completamente ajustado del contexto en el que se generaron los datos. (Skala et al., 2015)

3.3 MARCO CONTEXTUAL

Este trabajo se orientó en el área industrial específicamente para industrias 4.0, ya que estas industrias hacen un fuerte uso de sensores que pueden comprometer aspectos como, el ancho de banda y la seguridad de la empresa. Por ello se implementa una capa de *Fog Computing*, la cual no expone los sensores a internet sino a una red local donde están conectados. Esta información se procesa y puede viajar a la nube de manera segura.

3.4 MARCO LEGAL

La legislación y normatividad colombiana que corresponde al proyecto se enmarca en el cumplimiento de la Ley Nro. 1273 del 2009 – “De la Protección de la Información y de los Datos”. El cual se resalta las siguientes disposiciones específicas:

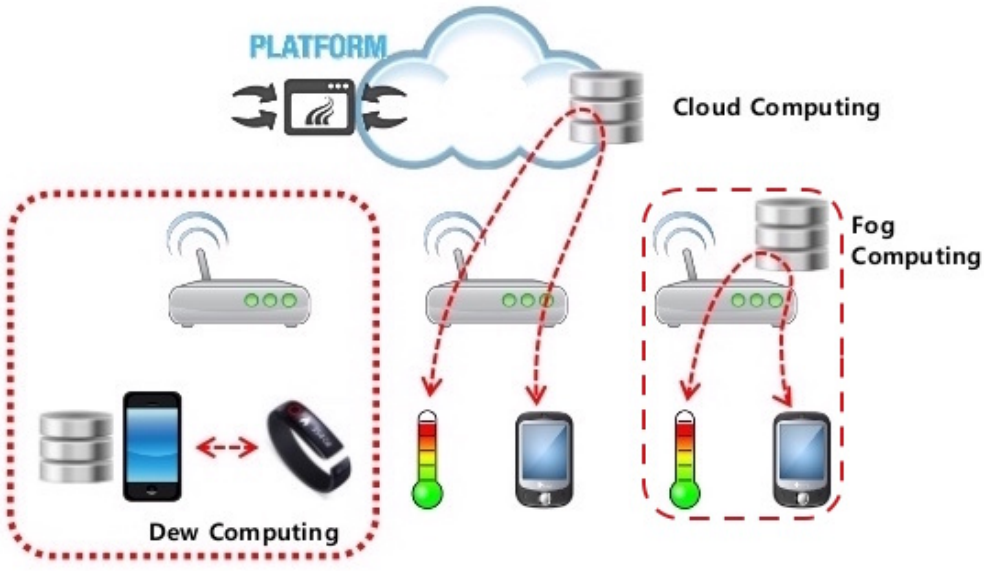
- Acceso ilícito: Artículo 269A, Ley, Protección de la información y de los datos.
- Interferencia en el Sistema: Artículo 269B, Ley, Protección de la información y de los datos.
- Interceptación ilícita: Artículo 269C, Ley, Protección de la información y de los datos.
- Interferencia en los Datos: Artículo 269D, Ley, Protección de la información y de los datos.
- Uso de software malicioso: Artículo 269E, Ley, Protección de la información y de los datos.

La legislación sustancial y procesal de los delitos cibernéticos se basan en la Constitución Nacional de Colombia, en los artículos 15 y 20. Esto con el fin de garantizarle a los usuarios los derechos fundamentales de conocer, actualizar y rectificar la información recolectada y el buen uso que se le dé en el tratamiento de esta. Adicionalmente aplicaría la normativa para la arquitectura de computación en la nube.

3.5 ESTADO DEL ARTE

Para desarrollar el estado del arte, se realizó la revisión bibliográfica académica de los diferentes tipos de arquitecturas en las que se resalta ***Fog Computing***. En la figura 5, se realiza un comparativo de las diferentes arquitecturas con los temas correspondientes a cada arquitectura.

Figura 5. Esquema comparativo de arquitecturas de computación.



Fuente (Hakyong KIM, 2014).

3.5.1 Revisión sistemática de la literatura.

Para ello se realizan las revisiones bibliográficas donde se encuentran con los siguientes criterios de búsqueda como *Fog Computing.*

El resultado de la búsqueda bibliográfica realizada en la base de datos bibliográfica *Scopus*, se presenta con el criterio de búsqueda con el código

*TITLE-ABS-KEY (**"Fog Computing"**)*

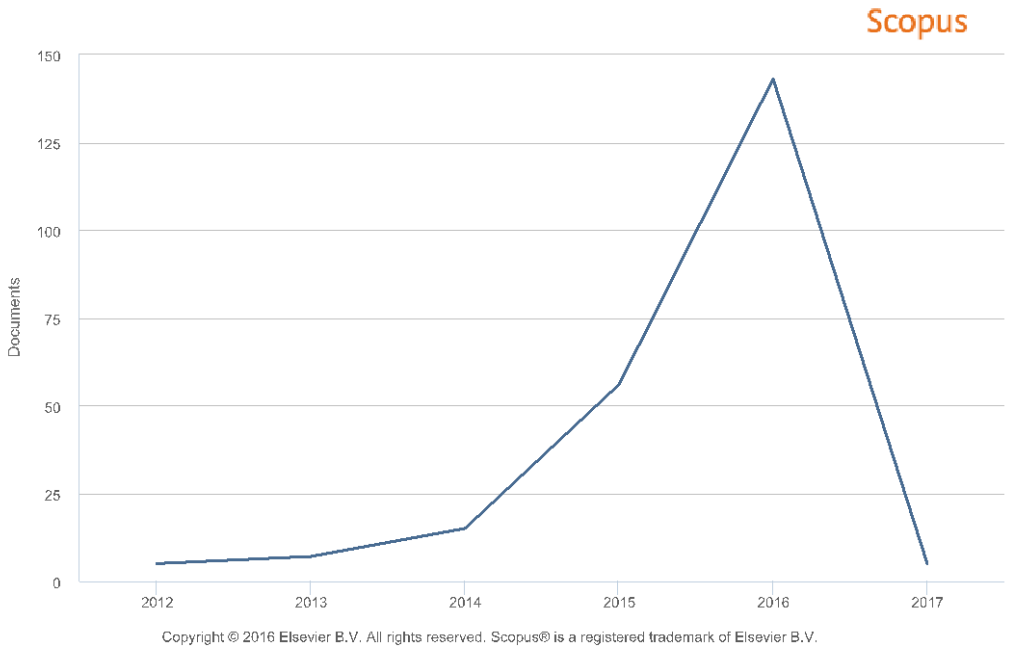
Los resultados, se muestran en la tabla 1 y en la figura 6, se categorizan por años y número de documentos incluidos en la base de datos.

Tabla 1 Resultados de publicaciones por año (Fecha de corte 27/12/2016).

Año	Publicaciones
2017	5
2016	143
2015	56
2014	15
2013	7
2012	5
Total	231

Fuente: *Scopus.*

Figura 6. Gráfica de los datos bibliográficos



Fuente: *Scopus.*

Al realizar este análisis bibliográfico, el proyecto se enmarca en una tendencia donde el *Fog Computing* es un término predilecto para realizar estudios sobre las problemáticas antes mencionadas.

Después de organizar los documentos recuperados según su autor como se muestra en la tabla 2 y figura 7, se entiende que *Mohammad Aazam*, poseen el mayor número de documentos de la búsqueda (8 documentos).

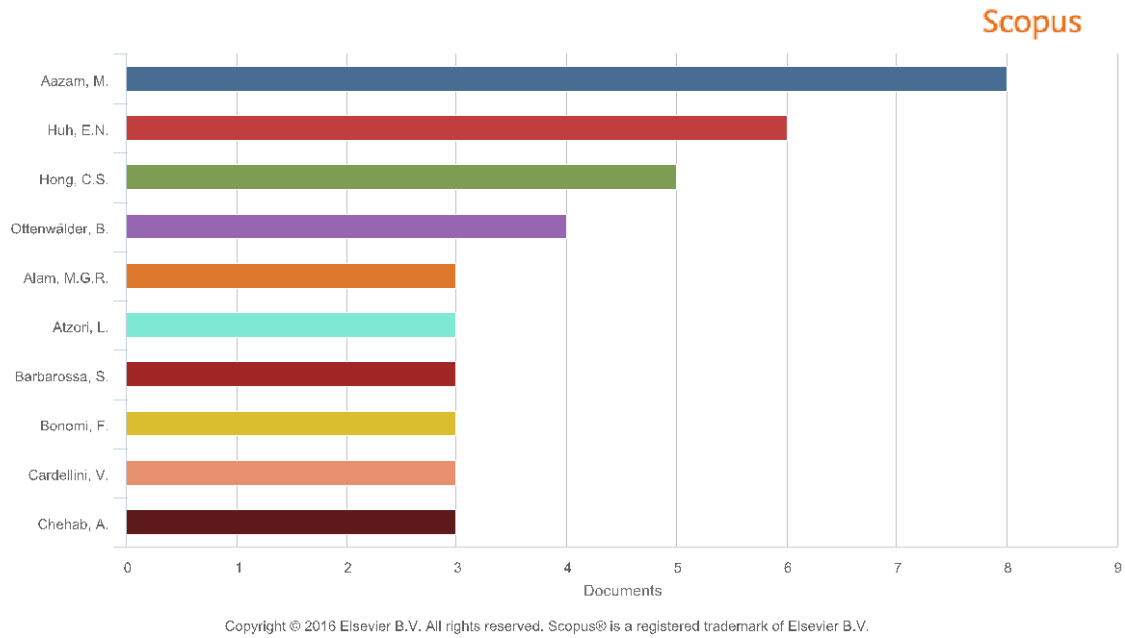
Tabla 2. Resultados bibliográficos por autor.

Autores	Publicaciones
Aazam, M.	8
Huh, E.N.	6
Hong, C.S.	5
Ottenwälder, B.	4
Alam, M.G.R.	3
Atzori, L.	3
Barbarossa, S.	3
Bonomi, F.	3
Cardellini, V.	3
Chehab, A.	3

Fuente: *Scopus.*

En la figura 7 se presenta gráficamente la información de la tabla 2.

Figura 7. Gráfica de los datos bibliográficos organizados por autor.



Fuente: *Scopus.*

Inmediatamente se organizan los documentos según país de origen, es posible observar que Estados Unidos cuenta con el mayor número de documentos producidos con un total de 53, número significativamente superior al de Italia con 24 documentos o Corea del Sur con 22 documentos.

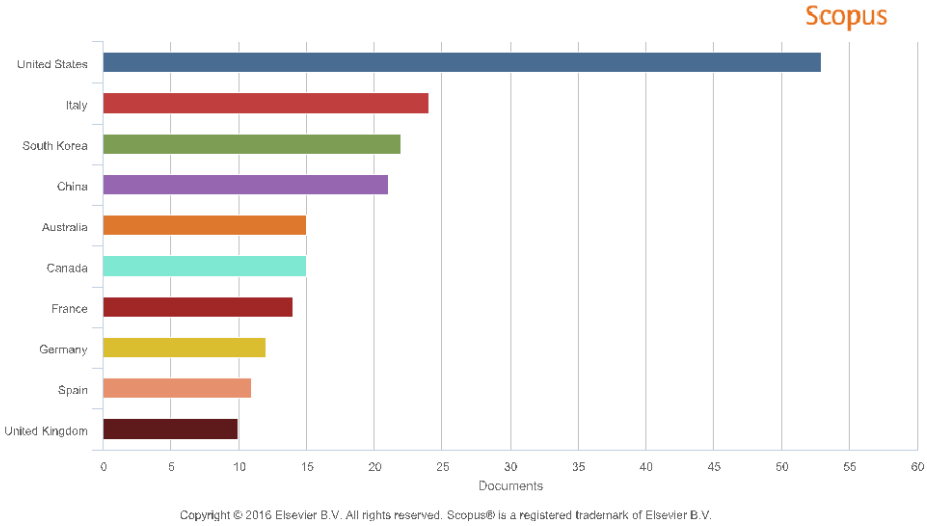
Tabla 3. Resultados bibliográficos por países.

País	Publicaciones
United States	53
Italy	24
South Korea	22
China	21
Australia	15
Canada	15
France	14
Germany	12
Spain	11
United Kingdom	10

Fuente: *Scopus.*

En la figura 8 se presenta gráficamente la información de la tabla 3.

Figura 8. Gráfica de los datos bibliográficos organizados por países.



Fuente: *Scopus.*

En seguida se clasifican los documentos según su tipo. Se tiene que el 65.4% son Artículos de Conferencia, el 20.8% son artículos publicados, el 6.5% son revisiones frente a la temática tratada, el 4.8% son artículos de prensa, el 2.2% son capítulos de libros y el 0.4% editoriales.

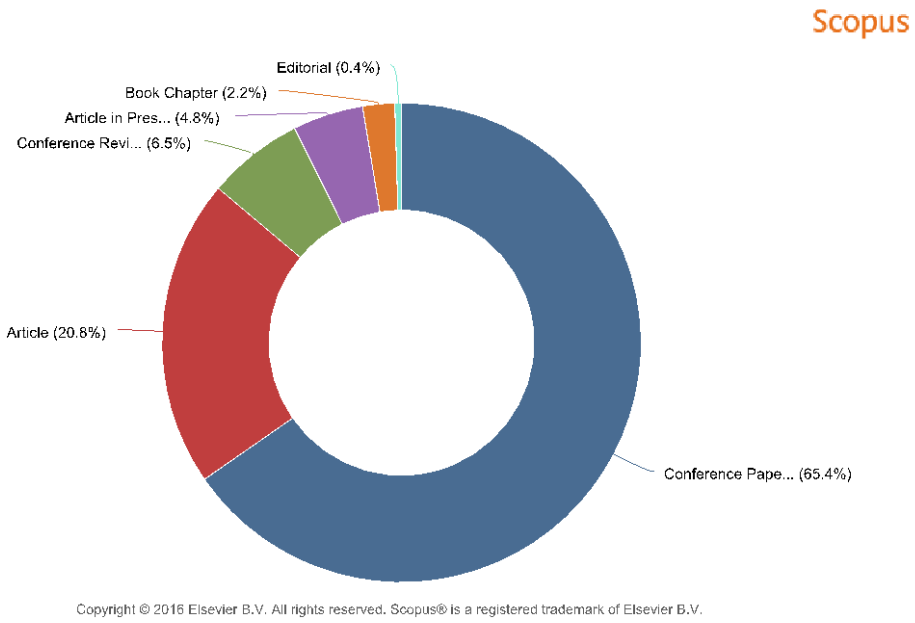
Tabla 4. Resultados de publicaciones por tipo.

Tipo de Publicación	Publicaciones
Conference Paper	151
Article	48
Conference Review	15
Article in Press	11
Book Chapter	5
Editorial	1

Fuente: *Scopus*.

Para la figura 9 se presenta gráficamente la información de la tabla 4.

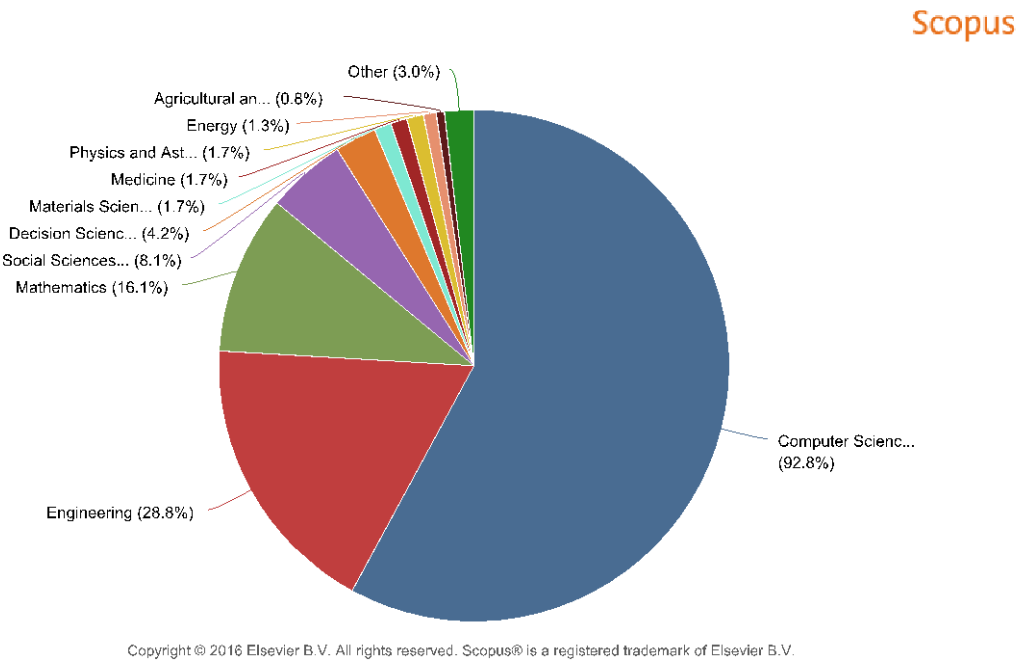
Figura 9. Gráfica de los datos bibliográficos organizados por tipo.



Fuente: *Scopus.*

Por último, se puede observar, en la figura 10, la clasificación de los documentos según el área de estudio. La relevancia del tema radica principalmente en el área de ciencia de computación con 92.8% de documentos, además, también es posible encontrar una cantidad significativa de documentos en las áreas de ingeniería con un 28,8% de documentos. Cabe resaltar que cada documento puede tener varias áreas de estudio.

Figura 10. Gráfica de los datos bibliográficos organizados por área de estudio.



Fuente: Scopus.

3.5.2 Análisis estado del arte.

Los criterios de búsqueda para llevar a cabo esta investigación se muestran en la tabla 5.

Tabla 5. Análisis Estado del Arte.

Palabras claves	*Fog Computing / Internet of Things / Internet of Things (IOT) / Cloud Computing*
Bases de datos consultadas	*ACM* *EBSCOhost* *IEEE* *Scholar* *Science Direct* *Scopus* *Springer Link*
Cantidad de referencias recuperadas	*ACM: 106* *EBSCOhost: 8* *IEEE:37* *Scholar: 686* *Science Direct: 48* *Scopus:172* *Springer Link: 48*
Fecha de búsqueda	28/12/2016
Criterios de búsqueda	Pertinencia con el tema de investigación: • *Fog Computing* • *Internet of Things* • *Cloud Computing*

Fuente: Autor.

Al escoger los diferentes artículos académicos recuperados, los más relevantes para el proyecto se muestran en la tabla 6.

Tabla 6. Artículos académicos recuperados para el proyecto.

Autores	Año	Titulo	Palabras Clave	Base de Datos	Citado
(Aazam & Huh, 2016)	2016	*Fog Computing: The Cloud-IoT / IoE Middleware Paradigm*	*Internet of things, Middleware, Clo ud computing*	IEEE	1

Tabla 6. (continuación)

Autores	Año	Titulo	Palabras Clave	Base de Datos	Citado
(Ashwini & SG, 2015)	2015	*Fog Computing to protect real and sensitivity information in Cloud*	*Cloud Computing, Fog Computing, Data theft*	Scholar	0
(Bonomi, Milito, Zhu, & Addepalli, 2012)	2012	*Fog computing and its role in the internet of things*	*Fog Computing, Cloud Computing, IoT, WSAN, Software Defined Networks, Real Time Systems, Analytics*	Scholar	681
(Celebre, Dubouzet, Medina, Surposa, & Gustilo, 2015)	2015	*Home automation using raspberry Pi through Siri enabled mobile devices*	*Home Automation, Raspberry Pi, Siri, SiriProxy, Speech Recognition.*	IEEE	0
(Dastjerdi, Gupta, Calheiros, Ghosh, & Buyya, 2016)	2016	*Fog Computing: Principals, Architectures, and Applications*	*Internet of Things; IoT; Web of Things; Cloud of Things; Fog Computing; IoT Applications; Edge Computing.*	Scholar	8
(Gia et al., 2015)	2015	*Fog Computing in Healthcare Internet of Things: A Case Study on ECG Feature Extraction*	*Internet of Things, Healthcare, Smart Gateway, Sensor Network, Heart Rate, ECG feature extraction, Fog Computing.*	Scopus	2
(Salman, Elhajj, Kayssi, & Chehab, 2015)	2015	*Edge computing enabling the Internet of Things*	*MEC, SDN, IoT, Fog computing, Cloud computing, NFV.*	IEEE	3
(Skala et al., 2015)	2015	*Scalable distributed computing hierarchy: Cloud, fog and dew computing*	*Distributed computing, grid computing, cloud computing, fog computing, dew computing, high-productivity computing*	Scholar	15
(Stojmenovic & Wen, 2014)	2014	*The fog computing paradigm: Scenarios and security issues*	Fog Computing, Cloud Computing, Internet of Things, Software Defined Networks	Scopus	38
(Szilagyi & Wira, 2016)	2016	*Ontologies and Semantic Web for the Internet of Things - a survey*	semantic web; internet of things; ontology; schema IoT; OWL	IEEE	0

Fuente: Autor.

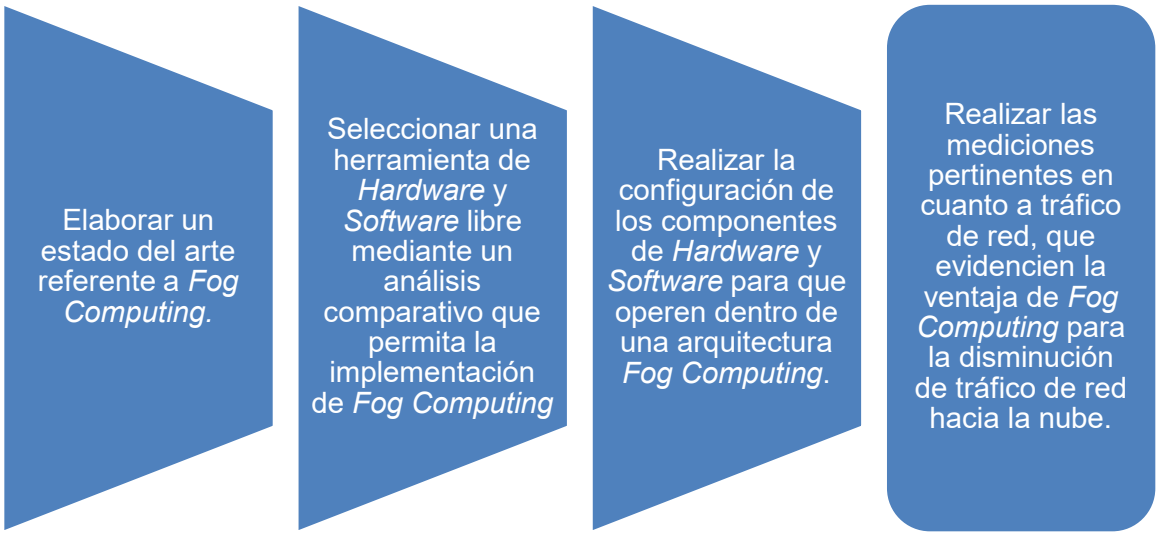
4. PROCESO INVESTIGATIVO

El proceso investigativo es utilizar e instalar una plataforma libre y de fácil disposición puesto que la investigación no profundiza en la construcción de una plataforma sino en tener un sistema de *hardware* y *software* libre para evitar los problemas de licenciamiento.

4.1 DESCRIPCIÓN DEL PROCESO INVESTIGATIVO

A continuación, se presentarán las actividades correspondientes a cada fase.

Figura 11 Representación de las actividades



Fuente: Autor.

Fase 1, Se realizaron las siguientes actividades: (1) Búsqueda y revisión de la literatura sobre uso de *Fog Computing*; (2) Lectura, clasificación y análisis de la información encontrada de acuerdo con la necesidad del proyecto.
Fase 2, Se realizaron las siguientes actividades: (1) Selección de la herramienta de *hardware* mediante un análisis comparativo de las especificaciones necesarias apropiadas para el proyecto; (2) Selección de la herramienta de *software* ajustada las especificaciones necesarias apropiadas para el proyecto.

Fase 3 Se realizaron las siguientes actividades: (1) Implementación de *hardware* adicional como sensores, actuadores e indicadores; (2) Instalación del *software* escogido dentro del hardware seleccionado; (3) Configuración del *hardware* y *software* elegido (4) Puesta en marcha de la arquitectura.
Fase 4, Se realizaron las siguientes actividades: (1) Evaluación del funcionamiento a partir del cumplimiento de tareas mínimas de operación; (2) Realización de pruebas básicas de conectividad; (3) Documentación del proyecto de Investigación.

4.2 PLANTEAMIENTO DE LA PLATAFORMA

Con el propósito de establecer un entorno frente a la plataforma propuesta se desarrolla un diagrama de casos de uso donde se plantea las diferentes peticiones y áreas donde se intervendrán.

4.2.1 Especificación del actor.

El actor simboliza todo usuario que desee interactuar con la plataforma propuesta con la funcionalidad expresada en este caso de uso. Las funcionalidades que la plataforma brinda al usuario son control de acceso, automatización y administración de la información del sistema.

4.2.2 Descripción de casos de uso.

A continuación, se muestra en la figura 11, en detalle los casos de uso.

4.2.2.1 Control de acceso.

El control de acceso consiste en proveer a los usuarios que quieran usar la plataforma una forma para acceder y tener control sobre los mismos dentro del sistema, apoyado en el mecanismo de seguridad de la información donde es almacenado dentro de una base de datos.

4.2.2.2 Administración de la información del sistema.

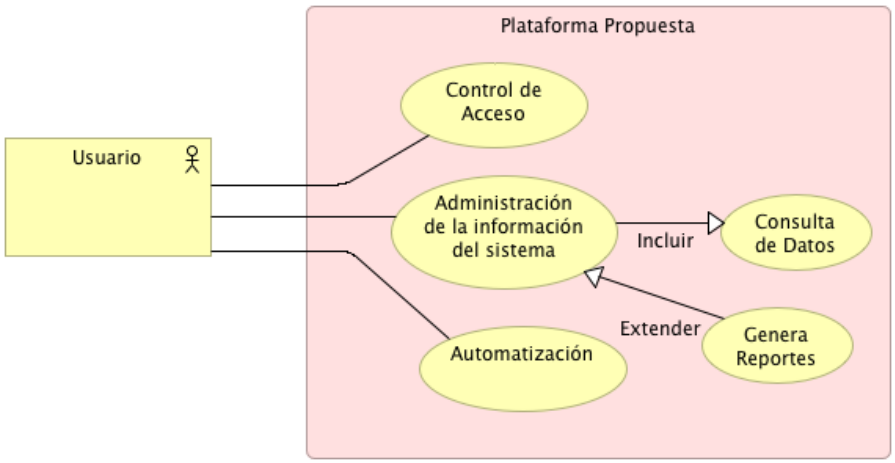
La administración de la información del sistema consiste en, organizar los datos del sistema de manera que se puedan procesar, almacenar, visualizar y exportar hacia otro sistema. En esta se incluye la consulta a los datos almacenados y se extiende la opción de generar reportes esto con el fin de que no es necesario generar el reporte.

4.2.2.3 Automatización.

La automatización consiste en hacer que determinadas acciones se vuelvan automáticas, es decir, que se desarrollen por sí solas y sin la participación directa de un usuario. Es por esto que el concepto suele utilizarse en el ámbito de la industria con referencia a un sistema de automatización, donde los datos son procesados automáticamente y convertidos en información teniendo valor para el actor y así utilizarlo

4.2.3 Diagrama de casos de uso.

Figura 12. Diagrama de Casos de Uso.

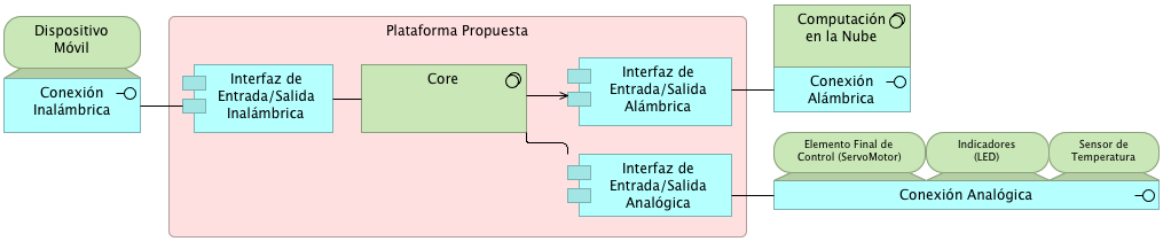


Fuente Autor (Archimate, 2016)

4.2.4 Arquitectura general de la plataforma propuesta.

En la figura 13 se muestra una visión general del sistema, conformado por la plataforma propuesta y los diferentes dispositivos que se relacionan incluyendo, el tipo de conexión e interfaz utilizada.

Figura 13. Visión general del sistema.

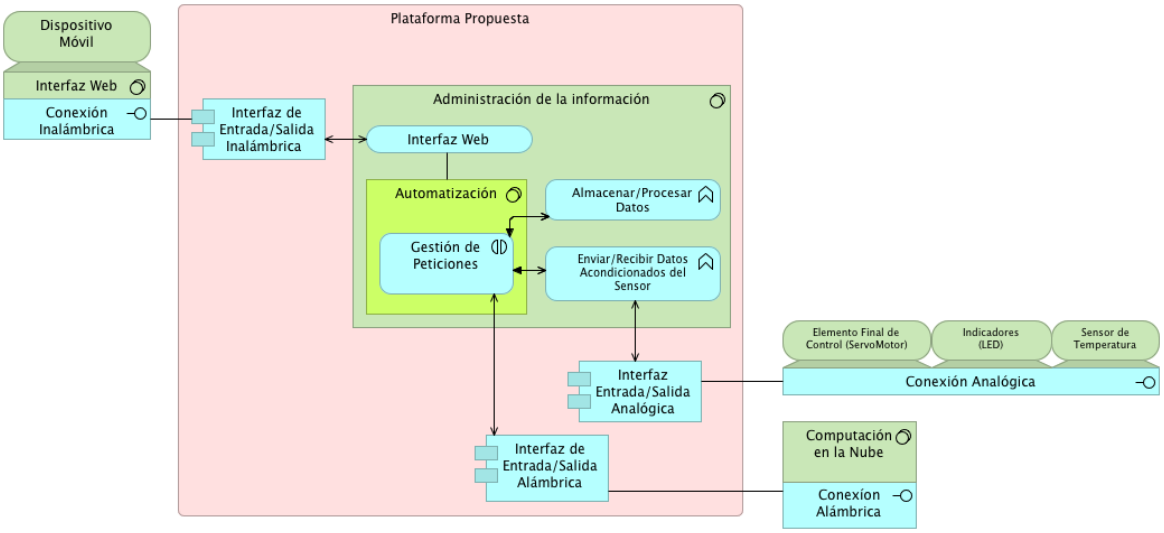


Fuente Autor (Archimate, 2016).

4.2.5 Dinámica de peticiones.

Para la dinámica de peticiones se realiza un esquema detallado de la plataforma propuesta, el cual se muestra en la figura 14, donde se presenta en detalle cada componente que interviene en la construcción de la arquitectura (*hardware* más *software*), esto con el fin de tener claridad sobre las peticiones que se realizan. El módulo para resaltar es el gestor de peticiones ubicado en el módulo de automatización, el cual realiza tareas automáticas como adquirir, almacenar, visualizar y exportar los datos a donde están asignados, siempre y cuando se configure de donde viene la petición.

Figura 14. Esquema general de la plataforma propuesta detallada.



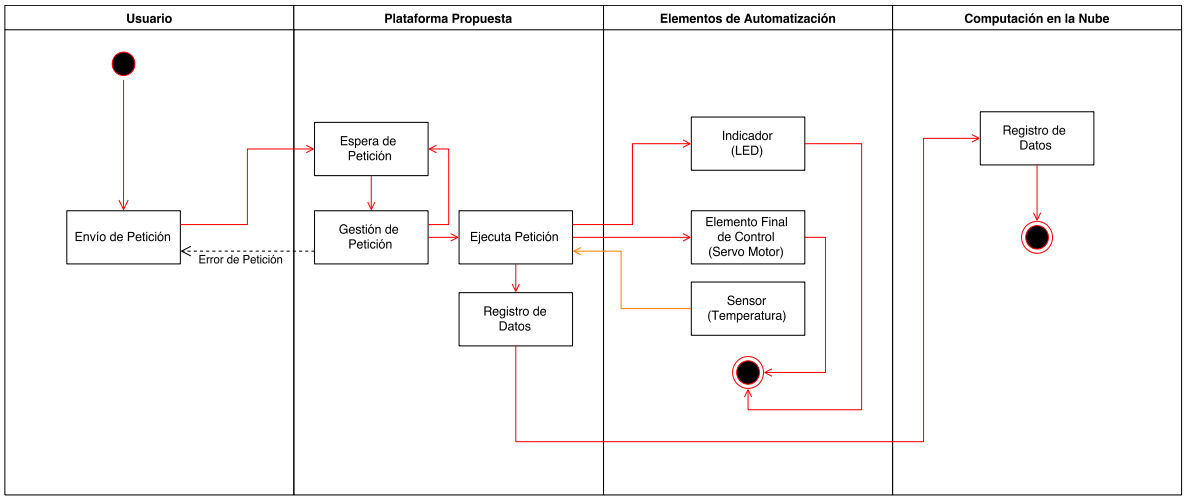
Fuente Autor (Archimate, 2016).

Los siguientes esquemas de peticiones son para el seguimiento de peticiones dentro de la plataforma propuesta y para una plataforma de *IoT*

4.2.5.1 Fog Computing.

Para la figura 15 se muestra un esquema de peticiones (diagrama de estados) en el cual busca orientar al investigador en cómo se debe realizar las diferentes peticiones dentro de los diferentes roles, donde el principal actor es el usuario el cual envía las diferentes peticiones y la plataforma propuesta el principal receptor de las misma. Es por ello que se determina como funciona cada rol anexo, como elementos de automatización y computación en la nube. Ya que ellos son parte de la arquitectura de *fog computing*. Para los elementos de automatización se escogen diferentes elementos en el cual representan, para el caso del sensor de temperatura, los datos son ingresados en el módulo de ejecución de petición debido a que esta se entrega siempre cuando el dato es solicitado (siempre está disponible), es por ello que se denota de diferente manera. Para computación en la nube recibe los datos de la arquitectura de *fog computing* esto es porque el procesamiento se realiza en la capa inferior a ella y recibe los datos ya procesados para registro o almacenaje.

Figura 15. Esquema de Peticiones para *Fog Computing*.

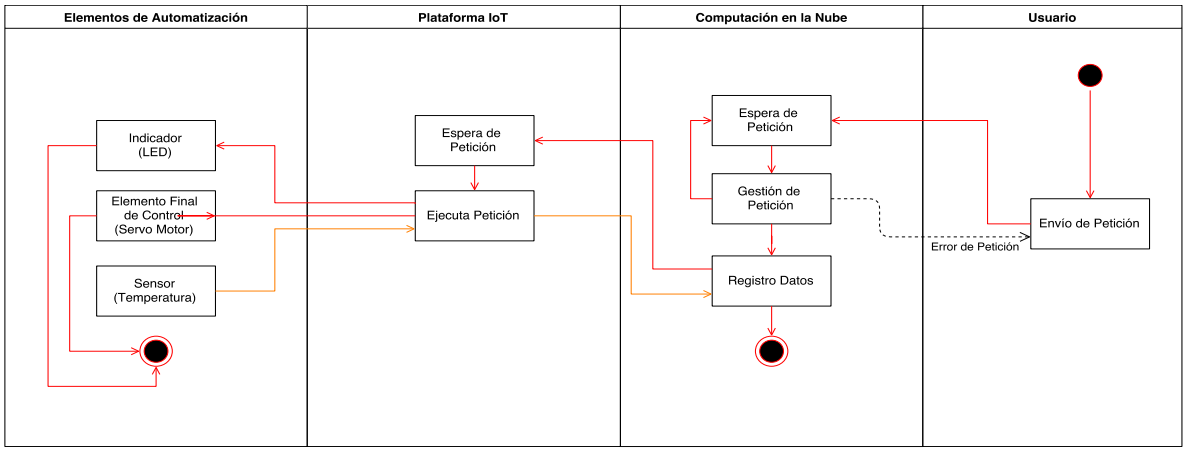


Fuente: Autor (Draw.io, 2017).

4.2.5.2 IoT (Internet of Things).

Para la figura 16 se muestra un esquema de peticiones para *IoT*, (diagrama de estados) en el cual da a conocer al investigador en cómo se hacen las diferentes peticiones dentro de los diferentes roles de *IoT*, donde el principal actor es el usuario el cual envía las diferentes peticiones al rol de computación en la nube donde esta hospedada la arquitectura de *IoT* y donde es el receptor de las peticiones. Es por ello que se determina cómo funciona cada rol anexo, como la conexión a la plataforma de *IoT*. Donde esta recibe la petición enviada y registrada por la computación en la nube donde esta ejecuta la petición hacia los elementos de automatización integrados a la plataforma y son ejecutados por la misma. Para el caso del sensor de temperatura la plataforma adquiere los datos y son enviados para su registro en la nube.

Figura 16. Esquema de Peticiones para IoT.



Fuente: Autor (Draw.io, 2017).

5. RESULTADOS

Son presentados los resultados del trabajo de Investigación por objetivos específicos.

5.1 ESTADO DEL ARTE REFERENTE A *FOG COMPUTING*

Para la realización del estado del arte se realizaron las siguientes actividades dando como resultado una búsqueda y revisión de la literatura sobre uso de *Fog Computing*, una lectura, clasificación y un análisis de la información encontrada de acuerdo con la necesidad del proyecto, logrando así un estado del arte valioso para investigadores. Es por ello que se realizó la selección de los artículos que influyen para el proyecto y que enmarcan su orientación a definir un concepto y construir una prueba de concepto. Por lo anterior se buscaron herramientas desarrolladas, casos de éxito y estudios en base a la necesidad de *fog computing* en el nuevo esquema de computación distribuida.

Por consiguiente, se realiza un análisis del propósito de cada artículo recuperado para que sea base conceptual y de apoyo al proyecto.

Tabla 7. Propósito de los artículos recuperados.

Autores	Año	Título	Propósito
(Aazam & Huh, 2016)	2016	*Fog Computing: The Cloud-IoT / IoE Middleware Paradigm*	El artículo muestra un comparativo entre las diferentes arquitecturas y las diferentes capas del *fog computing*, como transporte, seguridad, almacenamiento, procesamiento, monitoreo y virtualización. Un caso de éxito un *Smart Gateway* con *Fog Computing*.
(Ashwini & SG, 2015)	2015	*Fog Computing to protect real and sensitivity information in Cloud*	El artículo describe un modelo de extensión de computación en la nube como es *fog computing*, por sus cualidades de reducción en tiempo de transporte de la información, procesamiento y almacenamiento *on premise*.

Tabla 7. (continuación)

Autores	Año	Título	Propósito
(Bonomi et al., 2012)	2012	*Fog computing and its role in the internet of things*	El artículo da muestras de las características fundamentales de la computación de borde, como es la baja latencia, el conocimiento de la localización, y la amplia distribución geográfica, la movilidad, el gran número de nodos, la función predominante del acceso inalámbrico, la fuerte presencia de aplicaciones de *streaming* y de tiempo real y la heterogeneidad con hardware. Siendo este sus puntos críticos para el internet de las cosas.
(Celebre et al., 2015)	2015	*Home automation using raspberry Pi through Siri enabled mobile devices*	El artículo se basa en la prueba de concepto en realizar un sistema de domótica utilizando un hardware de placa única y un comando de voz (Siri). El caso de éxito busca la integración de un sistema de reconocimiento de voz con sensores.
(Dastjerdi et al., 2016)	2016	*Fog Computing: Principals, Architectures, and Applications*	El artículo se basa en que hay aplicaciones como en el sector salud que la respuesta de emergencia requiere una baja latencia y retraso. Por ende, surge fog computing, el concepto, su arquitectura y sus aplicaciones
(Gia et al., 2015)	2015	*Fog Computing in Healthcare Internet of Things: A Case Study on ECG Feature Extraction*	El artículo muestra un caso de éxito con computación de niebla o fog computing y también ofrecer servicios de IoT. El cual enfatizan en técnicas avanzadas de minería de datos incorporada, almacenamiento distribuido y servicios de notificación.
(Salman et al., 2015)	2015	*Edge computing enabling the Internet of Things*	El artículo expone el principal objetivo de MEC que es la exportación de algunas capacidades del Cloud Computing al usuario final, disminuyendo la latencia y aumentando el ancho de banda disponible logrando disminuir la carga en la red central y el mostrando escenario de real IoT, buscando desafíos y modelos.
(Skala et al., 2015)	2015	*Scalable distributed computing hierarchy: Cloud, fog and dew computing*	El artículo considera el enfoque conceptual sobre las jerarquías del Sistema de computación distribuida, caracterizando sus diferentes posiciones y mostrando el nuevo concepto de IoT dentro de este marco jerárquico.
(Stojmenovic & Wen, 2014)	2014	*The fog computing paradigm: Scenarios and security issues*	El artículo extiende el concepto de ampliación de los servicios de computación en la nube a estar más cerca de los usuarios, buscando así las cualidades de la nueva arquitectura y los diferentes escenarios en una prueba de ataque informático.
(Szilagyi & Wira, 2016)	2016	*Ontologies and Semantic Web for the Internet of Things - a survey*	El artículo muestra la realidad del Internet of Things (IoT), con el creciente número de dispositivos y su diversidad, donde desafía los enfoques y las tecnologías actuales para una integración más inteligente de sus datos, aplicaciones y servicios a la web, por ende, el desarrollo de una web inteligente y adaptada a los servicios que el IoT ofrece.

Fuente Autor.

De acuerdo con los artículos recuperados, el desarrollo de la temática *Fog Computing*, aporta y da a conocer ventajas, deficiencias y oportunidades en tópicos que se pueden tratar en trabajos de investigación.

Los investigadores son visionarios en plantear una nueva arquitectura al notar las deficiencias que tiene el internet de las cosas (IoT) y el impacto negativo que tendrá en la capa de computación en la nube. El aporte al trabajo de investigación es el fundamento conceptual y teórico donde se destaca la problemática presente en el transporte de datos y que es necesaria la nueva arquitectura.

El enfoque conceptual de la nueva arquitectura de computación distribuida se presenta de la siguiente manera: *Cloud Computing, Fog Computing* y *Dew Computing*. Las dos últimas son reconocidas como nuevas capas estructurales de la arquitectura, satisfaciendo las necesidades de las demandas informáticas de alto y bajo nivel. La arquitectura de computación distribuida reduce el costo y mejora el rendimiento, especialmente para el internet de las cosas (IoT). (Skala et al., 2015)

Para la investigación se toman conceptos básicos de los sistemas IoT y las tecnologías de la Web. Presentado al internet de las cosas como una evolución de la web donde el objetivo es formalizar conceptualmente y describir el conocimiento de un dominio IoT.

La realidad de *internet of things* (IoT), es su creciente número de dispositivos y su diversidad de aplicativos donde desafía los enfoques y tecnologías actuales para una integración más inteligente de sus datos, desarrollos y servicios. Mientras que la *Web* se ve como una plataforma para integrar las cosas, destacando deficiencias en el desarrollo de una aplicación o servicio de IoT. (Szilagyi & Wira, 2016)

En la investigación, la visión y la definición de las características claves de *Fog Computing* son importantes para entender los nuevos servicios y aplicaciones que se pueden trabajan en el borde de la red, los ejemplos son motivadores de acuerdo con las visiones conceptuales y los prototipos son soluciones muy puntuales para su desarrollo.

El estudio de la arquitectura de esta infraestructura de computación, el almacenamiento y entorno de red son relevantes para la orquestación y manejo de recursos que se encuentran en el borde de la red, donde plantea un análisis donde *Fog Computing* es una plataforma unificadora, para ofrecer una nueva generación de servicios emergentes y permitir el desarrollo de nuevas aplicaciones.

Fog Computing extiende el modelo del *Cloud Computing* al borde de la red, permitiendo así una nueva generación de aplicaciones y servicios. Las características destacadas son: la baja latencia y el conocimiento de la localización, la amplia distribución geográfica, la movilidad, el gran número de nodos, la diversidad en los ambientes desplegados, la función predominante del acceso inalámbrico, la presencia fuerte en aplicaciones de streaming y de tiempo real cercano.

Fog está destinado a ser la plataforma apropiada para un número de servicios y aplicaciones críticas de internet de cosas (IoT), como ejemplo, en vehículos inteligentes, *smart grid* (red eléctrica inteligente), *smart cities* (ciudades inteligentes) y en general, las redes inalámbricas de sensores y actuadores (WSAN). (Bonomi et al., 2012)

Con el rápido aumento de los servicios de IoT, tanto la gestión de la calidad del servicio como la eficiencia y la satisfacción de los usuarios, se han vuelto fundamentales. La nueva propuesta es el CoT donde el IoT está junto al *Cloud Computing* para una mejor administración de recursos, provisión de servicios y escalabilidad.

En situaciones de emergencia, en servicios de salud y servicios sensibles a la latencia, así como la seguridad, requieren que *Fog Computing*, esté presente entre los nodos y la nube distante.

La programación es más eficiente y la gestión de los recursos permiten que *Fog Computing* funcione de acuerdo a la situación y ayuda a la satisfacción del cliente.

El IoT es una revolución tecnológica que representa el futuro de la conectividad y accesibilidad. Los objetos inteligentes son digitales y físicos, que realizan tareas para los seres humanos y/o el medio ambiente. Por esta razón IoT no es sólo un paradigma de hardware y software, sino que también incluye la interacción y los aspectos sociales.

Cuando las personas intervienen y añaden más información, el IoT crece al Internet de todo (IoE). IoE reúne a personas, datos, procesos y cosas; haciendo que las conexiones en red sean más relevantes y valiosas. De esta manera, la información se convierte en acciones, creando nuevas capacidades, experiencias y oportunidades sin precedentes para personas, empresas y países.

Los servicios IoT en situaciones críticas y latencias sensibles requieren una respuesta y procesamiento muy rápidos. En este caso, no es factible comunicarse a través de la nube distante, a través de Internet. Es aquí donde el *Fog Computing* juega un papel muy importante en este sentido donde alude la introducción de recursos cerca de las redes subyacentes, extendiendo el paradigma de la computación en la nube tradicional al borde de la red, permitiendo la creación de aplicaciones o servicios refinados y eficientes.

La investigación sobre *Fog Computing* está en sus primeras etapas; Por lo tanto, no hay una arquitectura estándar disponible sobre la gestión de recursos de *Fog*. Existe una literatura muy limitada, varios autores presentan la arquitectura básica, que no incluye sus implicaciones prácticas y la gestión de recursos para el IoT; de manera similar presentan la protección de datos, pero no en la gestión de los recursos. (Aazam & Huh, 2016)

Los mundos de TI y Telecomunicaciones han experimentado transformaciones conceptuales reales en los últimos años. *Cloud Computing*, *Network Function Virtualization* (NFV) y *Software Defined Network* (SDN) están en la base de estos cambios.

Para el concepto de *Mobile Edge Computing*, tiene como objetivo aplicar *Fog Computing* ó *Cloud at the Edge* al dominio de la red móvil. Además, *Mobile Edge Computing* tendrá un impacto cercano en la forma en que los operadores de telecomunicaciones móviles despliegan nuevos servicios, estos se beneficiarán del NFV y, la combinación de SDN y NFV (SDNv2). De cualquier manera, el IoT, que está altamente asociado con la red móvil, se beneficiará al extender el concepto central de *Mobile Edge Computing* a otras áreas como VANET, WSN, etc.

El trabajo de investigación arroja una luz sobre estas nuevas tendencias y se ha dibujado una nueva visión de internet del futuro, necesitando más implementaciones técnicas para garantizar la eficacia de los esquemas propuestos.

Mobile Edge Computing (MEC) es un nuevo concepto que surgió de la integración de los mundos de TI y telecomunicaciones donde tendrá un gran impacto en la apertura del mercado de telecomunicaciones. Además, la revolución de virtualización que ha permitido el éxito de la computación en la nube, beneficiará al dominio de las telecomunicaciones, que a su vez será capaz de soportar la Infraestructura como servicio (IaaS). El principal objetivo de la solución *Mobile Edge Computing* es la exportación de algunas capacidades de *Cloud Computing* a la proximidad del usuario disminuyendo la latencia, aumentando el ancho de banda disponible y disminuyendo la carga en la red central.

Por otro lado, el IoT, se ha beneficiado de la proliferación en el uso de los teléfonos móviles. Muchas aplicaciones móviles han sido desarrolladas para conectar un mundo de cosas (portátiles, sistemas domóticos, sensores, etiquetas RFID, etc.) a Internet. Incluso si no es una solución completa para una arquitectura de IoT escalable, pero las aplicaciones IoT sensibles al tiempo (salud, seguridad, etc.) se beneficiarán de la arquitectura *Mobile Edge Computing*.

Además, IoT puede extender este paradigma a otras áreas con el uso de la orquestación de Software Defined Network (SDN) para hacer frente a los desafíos que obstaculizan el despliegue real de IoT. (Salman et al., 2015)

Fog computing está surgiendo como solución atractiva al problema del procesamiento de datos en internet de las cosas. Basándose en dispositivos en el borde de la red que tienen más potencia de procesamiento que los dispositivos finales y están más cerca de estos dispositivos que los recursos de nube más potentes, lo que reduce la latencia de las aplicaciones y costos.

En la investigación se introduce una arquitectura de referencia para IoT y se discuten los esfuerzos en curso de la academia y de la industria para definir la visión de *Fog Computing*. Sin embargo, todavía quedan muchos desafíos, con problemas que van desde la seguridad hasta la minimización de los recursos y la utilización de la energía. Los protocolos y las arquitecturas son temas para otras investigaciones que harán que el *Fog Computing* sea más atractivo para los usuarios finales.

La Internet de todo (IoE) atrae todos los objetos a la red, y mantiene el procesamiento de datos en la nube dependiendo del entorno. Esto se debe a que hay aplicaciones como el monitoreo de salud y la respuesta a emergencias que requieren de baja latencia debido a la transferencia de datos a la nube. Para este fin, ha surgido la computación de niebla o *Fog Computing*, donde el *Cloud Computing* se extiende hasta el borde de la red para disminuir la latencia y la congestión de la red.

Fog Computing es un paradigma para administrar un entorno altamente distribuido y posiblemente virtualizado que proporciona servicios de computación y de red entre sensores y centros de datos en la nube.

Los antecedentes, sobre el surgimiento de la computación de niebla y define sus características clave, donde presenta una arquitectura de referencia y se discuten desarrollos y aplicaciones relacionadas recientes. (Dastjerdi et al., 2016)

Con el aumento de los ataques de robo de datos, la seguridad de los datos del usuario se está convirtiendo en un problema serio para los proveedores de servicios en la nube, para los cuales *Fog Computing* es un paradigma que ayuda a monitorear el comportamiento del usuario y proporcionar seguridad a los datos del usuario.

Las plataformas de *Fog Computing* puede conducir a técnicas defensivas mejoradas y de esta forma aumentar el nivel de seguridad de los datos de usuario en la nube.

Lo más importante sin duda es la protección de la autenticación de la información y las firmas digitales, donde existe una necesidad de construir paradigmas seguros y eficientes. Por el momento, sólo existe un número limitado de construcciones probablemente seguras y que son muy ineficientes. Algunos resultados teóricos están disponibles para apoyar construcciones prácticas, pero la mayor parte de nuestro conocimiento sobre sistemas prácticos se origina en procedimientos de ensayo y error. Por lo tanto, es importante que las nuevas propuestas sean evaluadas a fondo por varios investigadores independientes y que no se implementen con demasiada rapidez. Además, las implementaciones deben ser modulares de tal manera que la actualización del algoritmo sea factible. La elección entre diferentes algoritmos también dependerá del *hardware* disponible.

La computación en la nube se puede describir simplemente como la computación basada cerca de Internet. En el pasado, la gente dependía de almacenamiento de equipos o servidores para ejecutar sus programas. Sin embargo, con la introducción del *Cloud Computing*, las personas y las compañías de software y otras agencias están cambiando más hacia el entorno de computación en la nube.

Para lograr una mayor eficiencia operativa en muchas organizaciones y agencias pequeñas o medianas se utiliza el entorno *Cloud* para la gestión de sus datos.

Es una combinación de varias estrategias y conceptos de computación tales como SOA (*Service Oriented Architecture*), virtualización y otras que dependen de Internet.

Cloud Computing proporciona una manera fácil de acceder, gestionar y calcular datos de usuario, pero también tiene algunos riesgos de seguridad elevados. Los riesgos más comunes hoy en día son los ataques de robo de datos. El robo de datos es considerado como una de las principales amenazas a la computación en la nube. Para hacer frente a estos casos y los intrusos maliciosos, hay algunas técnicas que

se utilizan para asegurar datos del usuario. La arquitectura *Fog Computing* está llamando la atención de los usuarios de la nube hoy en día, mejorando la calidad del servicio y también reduciendo la latencia, debido a su amplia distribución geográfica, es muy adecuada para analíticas en tiempo real cercano y de datos grandes. (Ashwini & SG, 2015)

Se Investiga las ventajas de la computación de niebla para servicios en varios dominios, y se provee el análisis del estado de la técnica y seguridad en el paradigma actual. Basado en el trabajo, algunas innovaciones en computación y almacenamiento pueden inspirarse en el futuro para manejar servicios intensivos de datos basados en la interacción entre *Fog* y *Cloud*. Un trabajo futuro se expandirá en el paradigma de *Fog Computing* al *Smart Grid*. En este escenario, se pueden desarrollar modelos de dispositivos *Fog*. Los dispositivos independientes de *Fog Computing* consultan directamente la nube para las actualizaciones periódicas en precio y demandas, mientras que los dispositivos interconectados en *Fog* crean coaliciones para nuevas mejoras.

El control del semáforo también puede ser asistido por el concepto de computación de niebla, la movilidad entre nodos de niebla, y entre niebla y nube, puede ser investigado. A diferencia de los centros de datos tradicionales, los dispositivos *Fog* se distribuyen geográficamente sobre plataformas heterogéneas, esto obliga a que la movilidad del servicio a través de las plataformas debe ser optimizada. Las ventajas de la computación de niebla, motivan trabajos de investigación donde se analizan sus aplicaciones en una serie de escenarios reales. Los puntos de seguridad y privacidad también se revelan de acuerdo con el paradigma actual de *Fog Computing*. Como ejemplo, un ataque típico de hombre en el medio. (Stojmenovic & Wen, 2014)

Al automatizar un hogar con una tarjeta de placa única con 5 electrodomésticos mediante el uso del reconocimiento de voz de Siri, se utilizan la red inalámbrica del dispositivo donde los clientes móviles se conectan y son habilitados para la utilización del asistente digital. El experimento revela que sistema fue capaz de automatizar los cinco equipos dentro de un espacio controlado. Donde se desarrolla las principales funciones como apagar y encender, abrir y cerrar, detectar el aumento y disminución de la temperatura, y controlar algunas funciones básicas del televisor como el volumen y cambiar canales.

El sistema es totalmente funcional y controlado a través del uso de la fidelidad inalámbrica, se recomienda un desarrollo para utilizar un asistente para teléfonos basados en Android y la automatización de equipos adicionales mediante la extensión del número de puertos I/O para utilizar sensores adicionales y otros dispositivos para determinar los estados actuales de los equipos. Ejemplo: Puerta está actualmente abierta.

La domótica es un sistema que permite controlar los dispositivos automáticamente con el fin de cumplir los requisitos de seguridad, comodidad y eficiencia. El asistente digital basado en voz, como el Siri de Apple, proporciona un acceso a internet y a los dispositivos localmente, Este estudio abarca la implementación de un sistema de automatización del hogar a través de la capacidad de reconocimiento de voz de Siri ya través de una placa única como un sistema de control de bajo costo para automatizar dispositivos domésticos como, el aire acondicionado, puertas, luces, la televisión y ventanas. El sistema ha sido probado y verificado a través de pruebas de precisión de reconocimiento de voz, pruebas de latencia de respuesta y pruebas de tasa de éxito. (Celebre et al., 2015)

Fog Computing es una arquitectura que permite mejorar sustancialmente los sistemas de vigilancia de la salud que normalmente se proporcionan mediante IoT. Se mantienen los servicios de *Cloud Computing* pero previamente se procesan los datos en Fog computing permitiendo así mayor velocidad en la captura, análisis y luego en la distribución de la información en los dispositivos pertinentes, ubicados en sitio, sin perder la característica de accesibilidad que permite la computación en la nube, pero si, disminuyendo costos al reducir los tiempos necesarios para acceder la nube, igualmente al disminuir los datos almacenado en ella. Internet de las Cosas proporciona un enfoque competente y estructurado para mejorar la salud y el bienestar de la humanidad. Una de las formas factibles de ofrecer servicios de salud basados en IoT es monitorear la salud de los seres humanos en tiempo real cercano usando sistemas de monitoreo de salud ubicuos que tienen la capacidad de adquirir bio-señales de nodos de sensores y enviar los datos a través de una comunicación inalámbrica, posteriormente en tiempo real cercano se transmiten a un servidor de nube para procesamiento, visualización y diagnóstico.

Los resultados experimentales revelan que, en estos casos, *Fog Computing* permite reducir en más del 90% del ancho de banda y ofrece baja latencia y respuesta en tiempo real cercano en el borde de la red. (Gia et al., 2015)

5.2 HERRAMIENTAS DE *HARDWARE* Y *SOFTWARE* LIBRE QUE PERMITAN UN ANÁLISIS COMPARATIVO EN LA IMPLEMENTACIÓN DE *FOG COMPUTING*

El estudio para configuración, comparación con los diferentes procesos, protocolos de comunicación y calidad de la información medida son actividades que se realizan para implementar la arquitectura de *Fog Computing.* De acuerdo con lo anterior a continuación se realiza la selección de herramientas de *hardware* y *software*.

5.2.1 *Hardware*.

Se presenta una selección de las diferentes herramientas de *hardware* y se escoge las convenientes para la arquitectura *Fog Computing.*

5.2.1.1 Selección de *hardware* de procesamiento.

Para la selección de *hardware* se busca utilizar dispositivos que tengan procesamiento y bajo costo, para ello se toma la decisión de utilizar el recurso de las tarjetas de placa única. Son tarjetas desarrolladas con características especiales similares a un computador funcional, es decir, que contengan un procesador, memoria *RAM*, tarjeta gráfica, conectividad, entradas y salidas analógicas. Esto último es para la utilización de sensores y otros componentes que interactúen con la placa única.

Por lo anterior la selección se realiza de acuerdo a la disponibilidad local de la tarjeta de placa única, donde se realiza un cuadro comparativo entre ellas como se visualiza en la tabla 7, en ella se muestra un comparativo entre los dispositivos y costos para la realización de la implementación.

Dentro de los requisitos necesarios para la selección es deseable que incorpore el módulo de conexión inalámbrica dentro de la placa única, esto debido a que se plantea la conectividad, la reducción de espacio y costo del mismo, para no incurrir en módulos adiciones.

Dentro de las opciones se consideraron dos, *Raspberry Pi 3 model B* y *Banana Pi M2+.* Para escoger la mejor opción se dirige a la característica de procesamiento que para el proyecto es requerido una mayor velocidad de procesamiento, debido a los procesos que se piensan realizar. Al tener esta característica se ubica que *Raspberry Pi 3 model B* en el cual tiene mayor velocidad de desempeño. La siguiente característica es la de arquitectura, la *Banana Pi M2+* cuenta con 32 bits y la *Raspberry Pi 3 model B* con 64 bits esto quiere decir la arquitectura de 64 bits en un procesador permite gestionar mayor cantidad de información que uno de 32 bits.

Por tal motivo se selecciona la *Raspberry Pi 3 model B*, adicionalmente esta tarjeta de placa única presenta una mayor madurez dentro de la comunidad de *hardware* libre, debido a su alto impacto (*IoT Developer Survey*, 2016)

Tabla 8. Comparativa de Tarjetas de Placa Única.

	Intel Galileo 2	Raspberry Pi 2 Model B	Raspberry Pi 3 Model B	Raspberry Pi Zero	Banana Pi M2+
Tamaño de la Board	12.3cm x 7.2cm	8.56cm x 5.6 cm	8.56cm x 5.6 cm	6.0cm x 3.0cm	9.2cm x 6.0cm
Procesador	Intel Quark X1000 - Single Core	Broadcom BCM2836 (ARMv7-A) ARM Cortex-A7 - Quad Core	Broadcom BCM2837 - ARMv8-A - ARM Cortex-A53 - Quad Core	Broadcom BCM2835 (ARMv7-A) ARM Cortex-A7 - Single-Core	Allwinner A31S - (ARMv7-A) - ARM Cortex-A7 - Quad-core
Arquitectura	32bit	32bit	64bit	32bit	32bit
Velocidad	400Mhz	900 Mhz	1.2Ghz	1Ghz	1Ghz
Memoria Cache	16kb L1 cache	12Kb L1 cache 1MB L2 cache	12Kb L1 cache 1MB L2 cache	2Kb L1 cache 1MB L2 cache	256Kb L1 cache 1MB L2 cache
Memoria RAM	256MB DDR3	1GB DDR2	1GB DDR2	512MB	1GB DDR3
GPU	NO	Videocore IV	Videocore IV	Videocore IV	PowerVR SGX54MP2 Comply with OpenGL ES 2.0 OpenCL 1x DX9_3
Almacenamiento Externo	MicroSD	MicroSD	MicroSD	MicroSD	MicroSD
Video	NO	HDMI	HDMI	HDMI	HDMI
Audio	NO	3.5 mm Jack y HDMI	3.5 mm Jack y HDMI	3.5 mm Jack y HDMI	3.5 mm Jack y HDMI
Conector JTAG	SI	NO	NO	NO	SI
Proposito General - GPIO	12 Pin	40 Pin	40 Pin	40 Pin	40 Pin
USB	2x USB 2.0	4x USB 2.0	4x USB 2.0	2x USB 2.0	4x USB 2.0
ETHERNET	10/100 Ethernet RJ45	10/100 Ethernet RJ45	10/100 Ethernet RJ45	10/100 Ethernet RJ45	10/100/1000 Ethernet RJ45
WI-FI	NO	NO	WiFi 802.11b/g/n	NO	WiFi 802.11b/g/n
PCIe	SI	NO	NO	NO	NO
ALIMENTACIÓN	5V DC @ 3A	5V DC @ 1.8A	5V DC @ 2.5A	5V DC @ 1.6A	5V DC @ 2A
PRECIOS (USD)	$60,90	$32,00	$38,00	$5,00	$35,00

Fuente autor.

5.2.1.2 Selección de *hardware* de automatización.

Se utilizan tres tipos de elementos de automatización, los cuales son Elemento Final de Control (Servo Motor), Indicadores (LED) y Sensor de Temperatura.

5.2.1.2.1 Elemento final de control (Servo Motor).

El elemento final de control (Servo Motor) seleccionado es el HK15138 de *HobbyKing* que cuenta con las especificaciones requeridas en la siguiente tabla.

Figura 17. Fotografía del Servo Motor *HobbyKing HK15138*

Fuente Servo Motor *HobbyKing HK15138* Recuperado de *https://hobbyking.com/en_us/hobbykingtm-hk15138-standard-analog-servo-4-3kg-0-17sec-38g.html*

Tabla 9. Datos de operación del servo motor HobbyKing HK15138.

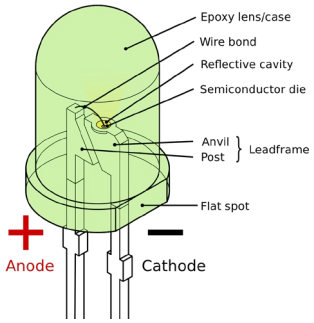
HK15138 Standard Analog Servo
Spec.
Torque: **3.8kg @ 4.8v, 4.3kg @ 6v**
Weight: **38g**
Speed: **0.21 / 60deg @4.8v, 0.17 / 60deg @ 6v**
Voltage:**4.8v~6v**
Plug: **JR Style**

Fuente Servo Motor *HobbyKing HK15138* Recuperado de *https://hobbyking.com/en_us/hobbykingtm-hk15138-standard-analog-servo-4-3kg-0-17sec-38g.html*

5.2.1.2.2 Indicadores (LED).

El indicador seleccionado es un LED (*light emitting diode*) de 5mm, sus componentes y terminales para realizar la conexión se muestra en la figura 13 y la tabla 10 muestra los datos de operación

Figura 18. Fotografía del LED 5mm.



Fuente: *Wikimedia Commons user: Inductiveload.* Recuperado de *https://commons.wikimedia.org/wiki/File:LED,_5mm,_green_(en).svg*

Tabla 10. Datos de operación del LED 5mm

Fordward drop	1.8-2.2VDC
Max Current	20mA
Suggested Current	16-18mA
Luminous Intensity	150-200mcd

Fuente: *Sparkfun Electronics.* Recuperado de *https://www.sparkfun.com/datasheets/Components/LED/COM-09590-YSL-R531R3D-D2.pdf*

5.2.1.2.3 Sensor de temperatura.

El sensor de temperatura para este caso sería el de temperatura DHT11 que cuenta con las especificaciones en la siguiente tabla 8. y contiene compatibilidad completa con la *Raspberry Pi 3 model B,* esto debido a que tiene los controladores afines.

El sensor cuenta con pines los cuales están distribuidos como +VCC(3.3V), GND (Tierra) y S (Señal).

Figura 19. Fotografía del DHT 11

Fuente DHT11 *Humidity & Temperature Sensor* Recuperado de *http://www.micropik.com/PDF/dht11.pdf*

Tabla 11. Datos de operación del DHT11

Item	Measurement Range	Humidity Accuracy	Temperature Accuracy	Resolution	Package
DHT11	20-90%RH 0-50 ℃	±5%RH	±2℃	1	4 Pin Single Row

Parameters	Conditions	Minimum	Typical	Maximum
Humidity				
Resolution		1%RH	1%RH	1%RH
			8 Bit	
Repeatability			±1%RH	
Accuracy	25℃		±4%RH	
	0-50℃			±5%RH
Interchangeability	Fully Interchangeable			
Measurement Range	0℃	30%RH		90%RH
	25℃	20%RH		90%RH
	50℃	20%RH		80%RH
Response Time (Seconds)	1/e(63%)25℃, 1m/s Air	6 S	10 S	15 S
Hysteresis			±1%RH	
Long-Term Stability	Typical		±1%RH/year	
Temperature				
Resolution		1℃	1℃	1℃
		8 Bit	8 Bit	8 Bit
Repeatability			±1℃	
Accuracy		±1℃		±2℃
Measurement Range		0℃		50℃
Response Time (Seconds)	1/e(63%)	6 S		30 S

Fuente DHT11 *Humidity & Temperature Sensor* Recuperado de *http://www.micropik.com/PDF/dht11.pdf*

5.2.2 *Software.*

Se presenta una selección de las diferentes herramientas de *hardware* y se escoge las convenientes para la arquitectura *Fog Computing.*

5.2.2.1 Sistema operativo.

Para ello se realiza una selección de sistema operativo el cual se configura el dispositivo como se muestra en la tabla 11. Donde se escoge *Raspbian* de base *Debían* debido a que cuenta con las librerías, controladores y módulos que necesita la *Raspberry Pi* para funcionar correctamente. Los demás sistemas operativos tienen diferentes propósitos, pero queda la posibilidad de realizar proyectos con ellos.

Tabla 12. Comparativo entre sistemas operativos y bases.

Nombre	***Raspbian***	*Ubuntu Mate*	*Snappy Ubuntu*	*Risc OS*	*Pidora*	*Kali Linux*	*Windows 10 IoT*
BASE	***DEBIAN***	*Ubuntu Core*	*Ubuntu Core*	*RISC*	*FEDORA*	*ARCH LINUX*	*Windows Embedded*

Fuente Autor.

5.2.2.2 *Software* complementario.

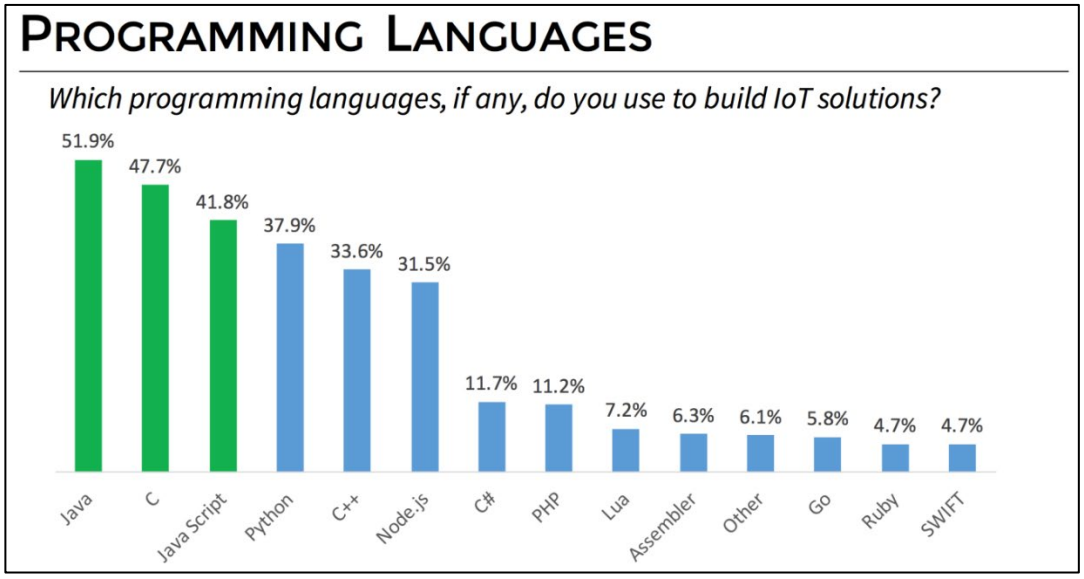
Al terminar la escogencia del sistema operativo se realiza un comparativo de *software* complementario, para ello se enfatiza en la escogencia del lenguaje de programación que se relaciona con los sensores y sus controladores, debido a que se busca un acople sin problemas, actualizada y la lectura del código sea sencilla.

Por lo anterior se realiza la búsqueda de programas de automatización usando como características la conexión y el lenguaje de programación como se encuentra en la tabla 12.

Al terminar la búsqueda se realiza una selección de los programas de automatización que utilicen el lenguaje de programación ***Python*** como opción deseada.

Esto con el fin de cumplir con los requisitos anteriormente mencionados y comprobando que es el lenguaje más utilizado para automatización, industrial e *IoT*. (*IoT Developer Survey*, 2016)

Figura 20. Lenguajes de Programación utilizados para *IoT*.



Fuente (*IoT Developer Survey*, 2016)

Tabla 13. Comparativo de programas de automatización.

Programas Seleccionados para Fog Computing		
Programas de Automatización		
Nombre	**Conexión**	**Lenguaje**
Ago Control	(SERIAL/ETHERNET)	[C++/PYTHON]
Calaos	(SERIAL/ETHERNET)	[C++]
DomotiGa	(SERIAL/ETHERNET/MQTT)	[GAMBAS]
Domoticz	(SERIAL/ETHERNET/MQTT)	[C++/PYTHON]
EasyIoT	(SERIAL/ETHERNET)	[C++]
FHEM	(SERIAL/ETHERNET)	[PERL]
Freedomotic	(SERIAL/MQTT)	[JAVA]
Heimcontrol.js	(SERIAL/ETHERNET)	[NODE.JS]
Home Assistant	(SERIAL/ETHERNET)	[PYTHON]
Home Genie	(MQTT)	[C++]
HouseMon	(MQTT)	[GO]
ioBroker	(MQTT)	[NODE.JS]
Jeedom	(SERIAL/ETHERNET)	[NODE.JS]
MajorDoMo	(MQTT)	[PHP]
MisterHouse	(SERIAL/ETHERNET)	[PERL]
MyController.org	(SERIAL/ETHERNET/MQTT)	[JAVA]
OpenRemote	(SERIAL/ETHERNET/MQTT)	[JAVA]
PiDome	(SERIAL/MQTT/GPIO)	[JAVA]
Pimatic	(SERIAL/GPIO)	[NODE.JS]
Pytomation	(SERIAL/ETHERNET)	[PYTHON]
The Thing System	(MQTT)	[NODE.JS]
homA	(MQTT)	[NODE.JS]
home.pi	(MQTT)	[NODE.JS]
openHAB	(MQTT)	[JAVA]
openLuup	(ETHERNET)	[LUA]

Fuente Autor.

Al escoger los cuatro con características deseables (que utilizan *Python*), se realizó la prueba de cada uno de ellos donde se escoge ***Domoticz*** debido a que contiene un módulo de servicios *web* interno (dentro de la red local) y externo (internet), compatibilidad con los controladores (*drivers*) y librerías de los diferentes sensores que se van a utilizar para la implementación. Sumado a lo anterior, se incluye en la escogencia del *software Domoticz* por la sección de complementos (*plugins*) en el cual se encuentra *Homebridge,* que proporciona un puente entre la *API* de *Domoticz* y la *API HomeKit* de *iOS* (*Homekit Siri - Domoticz*, 2016). Dentro de *HomeKit* se realiza la conexión con *Siri*, el asistente personal controlado por voz de *Apple* ("What is Siri?", 2015).

Figura 21. Diagrama de conexión entre *frameworks*

Fuente Autor.

5.3 CONFIGURACIÓN DE LOS COMPONENTES DE *HARDWARE* Y *SOFTWARE* PARA QUE OPEREN DENTRO DE UNA ARQUITECTURA *FOG COMPUTING*

A continuación, se presenta la configuración de los componentes de *hardware* y *software* acorde a la operación de la arquitectura de *Fog Computing*.

5.3.1 Configuración del *hardware* y *software.*

La configuración se realiza con la *Raspberry Pi* 3 con el sistema operativo base es *Raspbian Jessie with Pixel versión September 2016*, que se inserta en una *microSD* de 32*Gb*, esto con el fin de iniciar la implementación de la arquitectura.

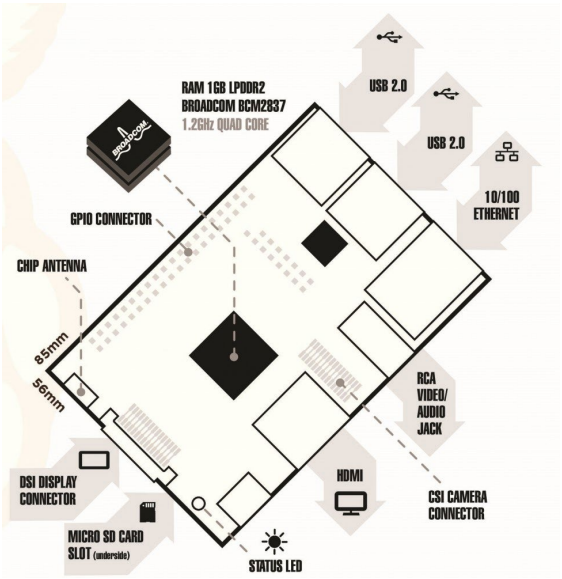
5.3.1.1 *Hardware.*

Luego de realizar la configuración del sistema operativo se procede a revisar las conexiones analogicas respectivas, como se muestra en la figura 15, la cual muestra los diferentes componentes del dispositivo, para el proyecto no se utiliza los conectores *CSI Camera Connector* y *DSI Display Connector*. Para el proyecto se utiliza el puerto de conexión GPIO como se muestra acontinuación.

5.3.1.1.1 *Raspberry Pi.*

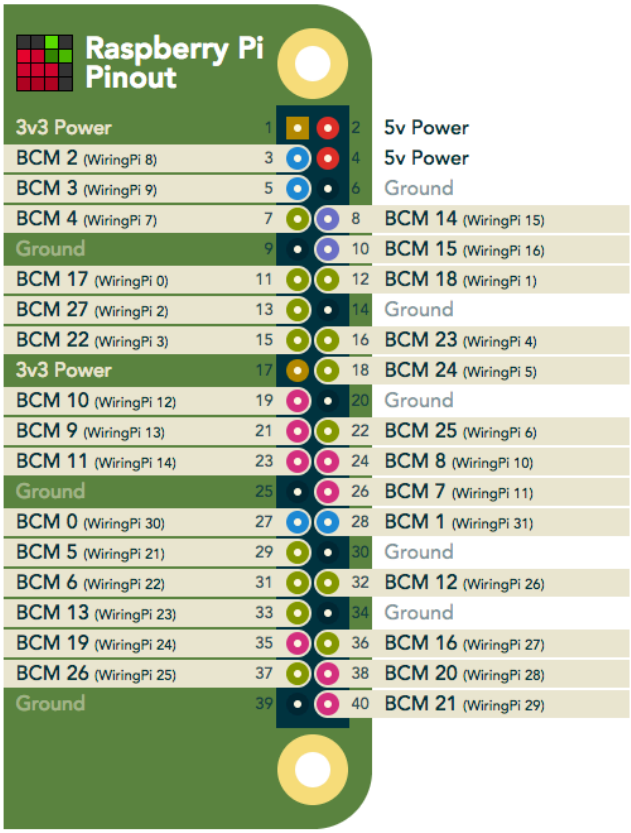
Raspberry Pi cuenta con una GPIO (*general purpose input/output*) donde se conectan los dispositivos analógicos y digitales. Para ello se ubica dentro de la hoja de datos de la misma el diagrama esquemático de conexión, para ello se recurre a un repositorio donde se muestra tres tipos de numeración como son; *Board*, BCM y *WiringPi*, para el proyecto se utiliza los pines BCM, debido a la compatibilidad con el lenguaje de programación utilizado. Esto quiere decir que al realizar la configuración de los sensores se debe tener en cuenta la numeración BCM como se muestra en la figura 16. Cabe resaltar que GPIO pueden ser programadas como entradas y salidas, estas últimas tienen como valor de voltaje 3.3v, esto para el caso que se necesite enlazar más sensores.

Figura 22. Diagrama detallado de *Raspberry Pi* 3



Fuente *Pi-Supply* recuperada de https://www.pi-supply.com/wp-content/uploads/2016/02/Raspberry-Pi-3-1.jpg

Figura 23. Diagrama de conexiones de GPIO.

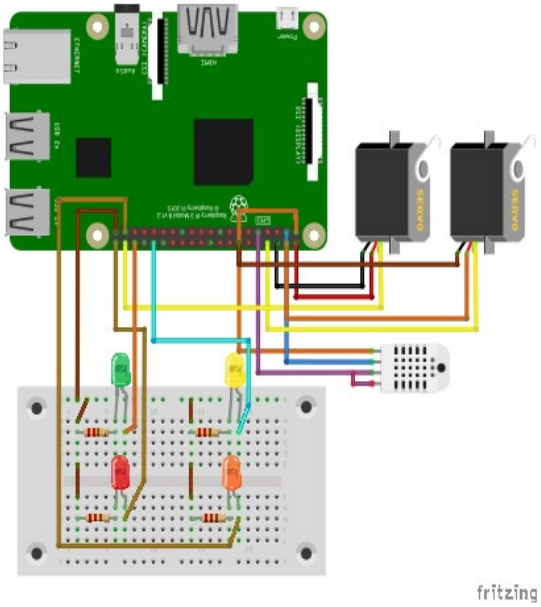


Fuente *Pinout* recuperado de https://pinout.xyz/pinout/wiringpi

5.3.1.1.2 Elementos de automatización.

Al determinar los pines de conexión se realiza la implementación de los sensores, determinando los VCC, GND y Señal. Dando importancia a la capacidad de proveer energia, es por ello por lo que la *Raspberry Pi* cuenta con dos salidas de voltaje VCC de 5v y una de 3.3v. Por lo anterior se distribuye entre ellas los sensores, llenando la capacidad. Como se muestra en la figura 16. Para los sensores indicadores es posible compartir la conexión de GND.

Figura 24. Diseño esquemático detallado del *hardware*.



Fuente Autor.

5.3.1.2 *Software.*

Componente donde se procede a instalar y utilizar los diferentes servicios que integran el proyecto de arquitectura de *fog computing*. Para los siguientes modulos necesitan de las credenciales de *root,* que se utilizan dentro del terminal. Esto con el fin de tener todos los permisos para realizar todas las operaciones sin problemas.

5.3.1.2.1 Red inalambrica.

La red inalambrica se le asigna una dirección IP a cada cliente ó dispositivo que ingresa, se configura como se muestra en el siguiente código.

```
nano /etc/network/interfaces
```

```
auto wlan0
allow-hotplug wlan0
iface wlan0 inet static
address 192.168.10.1
netmask 255.255.255.0
network 192.168.10.0
post-up echo 1 > /proc/sys/net/ipv4/ip_forward
```

Para la red inalámbrica se asigna el intervalo de direcciones IP 192.168.10.0 a 192.168.10.254 para los clientes. La dirección 192.168.10.1 se asigna a *Raspberry Pi.* Para activar el redireccionamiento de IP de la red inalámbrica a la red alámbrica se necesita activar un parámetro en el cual se encuentra de la siguiente manera en el código. Se ingresa y se cambia el parámetro;

```
nano /etc/sysctl.conf
```

```
net.ip4.ip_forward=1
```

Comando para reiniciar los parámetros de red.

```
/etc/init.d/networking restart
```

5.3.1.2.2 *Hostapd.*

Hostapd es un complemento de la configuración de la red inalámbrica, donde el punto de acceso Wi-Fi sea visible. Con el siguiente comando se procede a la instalación.

```
apt-get install –y hostapd
```

luego se edita el archivo /etc/default/hostapd y agrega al final el siguiente parámetro:

```
DAEMON_CONF = "/etc/hostapd/hostapd.conf"
```

Luego se edita el archivo /etc/hostapd/hostapd.conf y se copian las siguientes líneas:

```
#interface de la wlan Wi-Fi
interface=wlan0
# Nl80211 con todos los drivers de Linux mac80211
driver=nl80211
#nombre de la red Wi-Fi
ssid=IoT-Wi-Fi
#modo Wi-Fi (a = IEEE 802.11 a, b = IEEE 802.11 b, g = IEEE 802.1 g)
hw_mode =g
#canal de frecuencia de Wi-Fi(1-14)
channel=6
```

```
# Wi-Fi abierta
auth_algs=1
# Beacon interval (1.024 ms)
beacon_int=100
# DTIM(delivery traffic information message)
dtim_period=2
# Maximum number of stations allowed in station table max_num_sta = 254
# RTS / CTS threshold; 2347 = disabled(default)
rts_threshold=2347
#Fragmentation threshold; 2346 = disabled(default)
fragm_threshold=2346
```

Luego se inicia el modulo con el siguiente comando.

```
service hostapd start
```

5.3.1.2.3 FreeRadius.

Para la base de datos se utiliza *MySQL* donde la contraseña que asignamos es raspbian. *FreeRadius* es una base de datos dirigida a *hotspots* con el fin de tener datos de usuarios los cuales pueden acceder a la red interna.

Acontinuación se configura la base de datos con un usuario *root* y la contraseña *raspbian*, la instalación de freeradius como se muestra en el siguiente codigo.

```
apt-get install –y debconf-utils debconf-set-selections <<< 'mysql-server mysql-server/root_password password raspbian'
debconf-set-selections <<< 'mysql-server mysql-server/root_password_again password raspbian'
apt-get install –y debhelper libssl-dev libcurl4-gnutls-dev mysql-server freeradius freeradius-mysql gcc make libnl1 libnl-dev pkg-config iptables
```

Para la configuración de *FreeRadius* se debe crear una base de datos dedicada. Con el siguiente código en el terminal del rasbian.

```
echo "create database radius;" | mysql –u root –p raspbian
```

Luego, se crea el esquema de la base de datos.

mysql –u root –p raspbian radius < /etc/freeradius/sql/mysql/schema.sql
Inmediatamente, se crea un perfil administrativo para la base de datos.

mysql –u root –p raspbian radius < /etc/freeradius/sql/mysql/admin.sql

Adicionalmente, se crea las tablas de almacenamiento *NAS*

mysql –u root –p raspbian radius < /etc/freeradius/sql/mysql/nas.sql

Luego en la línea 700 se debe remover el comentario del archivo de configuración se */etc/freeradius/radiusd.conf*, debido a que se debe iniciar el módulo de SQL.

Para habilitar la autenticación de la base de datos se debe remover el comentario de las líneas 177, 406 y 454.

Para iniciar y probar la configuración realizada se debe detener el modulo con el siguiente comando.

service freeradius stop

y luego reiniciar el módulo FreeRadius en modo *debug*.

freeradius –X

Para realizar una conexión de prueba el cual se crea un usuario prueba “usertest” y con contraseña “passwd”

echo "insert into radcheck (username, attribute, op, value) values ('usertest', 'Cleartext-Password', ':=', 'passwd');" | mysql –u root –p raspbian radius

El código para realizar la prueba de usuario y contraseña es

radtest usertest passwd localhost 0 testing123

El valor testing123 viene de los parámetros de configuración del archivo /etc/freeradius/clients.conf el cual consiste en una palabra clave, para realizar la conexión entre el *FreeRadius* y el NAS. Al terminar debe estar la base de datos de usuarios lista y con un usuario administrativo para ingresar.

5.3.1.2.4 *daloRADIUS.*

Es un gestor de usuarios, donde se pueden gestionar usuarios, de manera gráfica, con atributos y parámetros de ingreso. Este módulo posee tipos de protocolos que realizan tres funciones: autenticación, autorización y contabilización (en inglés, *Authentication, Authorization and Accounting,* AAA.) Esta administración de AAA se procesa en un servidor *RADIUS* dentro de este módulo.

Para realizar la instalación de *daloRADIUS* se necesitan los siguientes paquetes de instalación.

```
apt-get install-y php5–mysql php–pear php5-gd php-db php5-fpm libgd2–xpm–dev
libpcrecpp0 libxpm4 php5-xcache
```

Luego se descarga el *daloRADIUS*, y se instala en la carpeta de apache www/html

```
git clone https://github.com/lirantal/daloradius.git
```

Se agrega la información utilizada por *daloRADIUS* en la base de datos de *FreeRadius*

```
mysql –u root –p raspbian radius </www/daloradius/contrib/db/fr2-mysql-
daloradius-and-freeradius.sql
mysql -u root -p raspbian
GRANT ALL ON radius.* to 'radius'@'localhost';
GRANT ALL ON radius.* to 'radius'@'127.0.0.1';
exit;
```

En el archivo www/daloradius/library/daloradius.conf.php se define los permisos de acceso a la base de datos

```
$configValues['CONFIG_DB_USER']='radius';
$configValues['CONFIG_DB_PASS']='radpass';
$configValues['CONFIG_DB_NAME']='radius';
```

se verifica el soporte PHP con el servidor apache después de dar inicio al servicio.

```
service apache2 restart
```

Luego se puede ingresar al gestor de usuarios en la dirección web http://192.168.10.1/daloradius con el usuario: administrator y contraseña: radius

5.3.1.2.5 *CoovaChilli.*

Es un módulo de control de acceso a una red local o pública. Esto se utiliza para acceso a la red inalámbrica antes configurada, permitiendo un ingreso controlado a la red.

Para el proceso de instalación se realiza de la siguiente manera, en el terminal se ubica una carpeta especial para *CoovaChilli* y se procede a descargar el archivo de instalación.

```
git clone https: //github.com/coova/coova-chilli.git
```

Luego de obtener el repositorio se procede a instalar con los siguientes comandos:

```
sh bootstrap
./configure
make
```

Al finalizar se realiza un cambio en los parámetros de *CoovaChilli*, en el cual se inserta la siguiente línea de código, que permite el acceso de la red *Wi-Fi* a *Ethernet*. Se inserta al final del archivo ubicado en /etc/chilli/up.sh.

```
iptables –I POSTROUTING –t nat –o $HS_WANIF –j MASQUERADE
```

Para hacer que el *CoovaChilli* se inicie cuando arranca el sistema operativo se cambia un parámetro como se muestra a continuación.

En el archivo /etc/default/chilli se reemplaza,

START_CHILLI=0

Por

START_CHILLI=1

En el archive /etc/chilli/config se realiza la configuración principal de los parámetros a definir, como es, la interfaz ethernet de salida a internet y la interfaz de red local para ello se configura de la siguiente manera.

Configuración detallada de los parámetros.

HS_WANIF: es la interfaz de salida a internet.
HS_LANIF: es la interfaz de red Wi-Fi.
HS_NETWORK: la red asignada para Wi-Fi.
HS_UAMLISTEN: la puerta de enlace de la red Wi-Fi.
HS_UAMALLOW IP: direcciones permitidas para conectarse a la red local.
HS_SSID: el nombre de la red Wi-Fi visible.

```
HS_WANIF=eth0
HS_LANIF=wlan0
HS_NETWORK=192.168.10.0
HS_UAMLISTEN=192.168.10.1
HS_UAMALLOW=192.168.10.0/24
HS_SSID=IoT-WiFi
```

Para preparar el inicio el servicio se asigna el siguiente código.

```
update-rc.d chilli start 99 2 3 4 5.stop 20 0 1 6 .
```

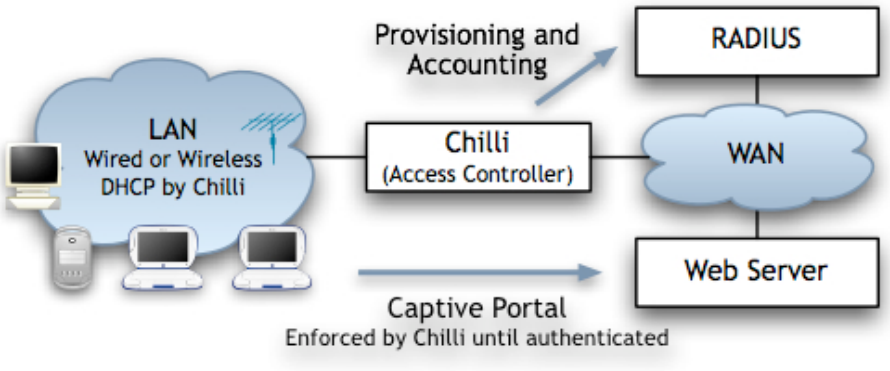
Inicio del servicio de *CoovaChilli*

```
service chilli start
```

Con el comando ifconfig se puede ver una interfaz tun0 confirmando que está bien ejecutado el *CoovaChilli*, al finalizar el diagrama de funcionamiento debe ser de la

siguiente manera ya que se utiliza el control de acceso con una base datos y una salida a internet. Como se muestra en la figura 15.

Figura 25. Diagrama de funcionamiento de *CoovaChilli*



Fuente recuperado de https://coova.github.io/img/Chilli_2.jpg

5.3.1.2.6 Python 2.7.

Anteriormente mensionado, *python* versión 2.7, es el lenguaje de programación escogido para realizar la conexión con los sensores. Para el siguiente modulo se insertan los programas utilizados interactuar con los sensores, cabe resaltar que raspbian traer por defecto el compilador de python 2.7.

5.3.1.2.6.1 LED.

El LED cuenta con dos estados, encendido (*ON*) y apagado (*OFF*) los cuales al ejecutar cada programa por separado envia la orden para cambiar el estado y no producir daños en el dispositivo.

5.3.1.2.6.1.1 *On.*

```
#!/usr/bin/python
# -*- coding: utf-8 -*-
import RPi.GPIO as GPIO
import time
import sys
GPIO.setmode(GPIO.BCM) # Ponemos la Raspberry en modo BCM
```

```
GPIO.setwarnings(False)
LED=7
GPIO.setup(LED,GPIO.OUT)  #se asigna el pin 7 como salida.
GPIO.output(LED,GPIO.HIGH) #se asigna la salida como estado alto.
```

5.3.1.2.6.1.2 *Off*.

```
#!/usr/bin/python
# -*- coding: utf-8 -*-

import RPi.GPIO as GPIO
import time
import sys
GPIO.setmode(GPIO.BCM)
GPIO.setwarnings(False)
LED=7
GPIO.setup(LED,GPIO.OUT) #se asigna el pin 7 como salida.
GPIO.output(LED,GPIO.LOW) # se asigna la salida como estado bajo.
GPIO.cleanup()  #limpiar puertos.
```

5.3.1.2.6.2 Servo motor.

El servo motor cuenta con dos estados, encendido (ON) y apagado (OFF) los cuales al ejecutar cada programa por separado envia la orden para cambiar el estado y no producir daños en el dispositivo.

5.3.1.2.6.2.1 *On*.

```
#!/usr/bin/python
# -*- coding: utf-8 -*-

import RPi.GPIO as GPIO    #Importamos la libreria RPi.GPIO
import time                #Importamos time para poder usar time.sleep
import sys
print len(sys.argv)
```

```
if len(sys.argv)==1:

    GPIO.setmode(GPIO.BCM) #Ponemos la Raspberry en modo BCM
    GPIO.setwarnings(False)
    GPIO.setup(20,GPIO.OUT) #Ponemos el pin 20 como salida
    p = GPIO.PWM(20,50) #Ponemos el pin 20 en modo PWM y enviamos 50 pulso
s por segundo
    p.start(7.5)              #Enviamos un pulso del 7.5% para centrar el servo
    try:
        while True:      #iniciamos un loop infinito

            p.ChangeDutyCycle(12.5)   #Enviamos un pulso del 12.5% para girar el ser
vo hacia la derecha, el valor del porcentaje puede cambiar.
            time.sleep(0.5)          #pausa de medio segundo
            p.ChangeDutyCycle(0.5)    #Enviamos un pulso del 0.5% para centrar el se
rvo de nuevo, el valor del porcentaje puede cambiar.
            time.sleep(0.5)          #pausa de medio segundo


    except KeyboardInterrupt:  #Si el usuario pulsa CONTROL+C
        p.stop()   #Detenemos el servo
        GPIO.cleanup() #Limpiamos los pines GPIO de la Raspberry y cerramos el sc
ript
if len(sys.argv)==2:
    GPIO.setmode(GPIO.BCM)   #Ponemos la Raspberry en modo BCM
    GPIO.setwarnings(False)
    GPIO.setup(20,GPIO.OUT)    #Ponemos el pin 20 como salida
    p = GPIO.PWM(20,50)       #Ponemos el pin 20 en modo PWM y enviamos 50 p
ulsos por segundo
    p.stop()                  #Detenemos el servo
GPIO.cleanup()# Limpiamos los pines GPIO de la Raspberry y cerramos el script
```

5.3.1.2.6.2.2 *Off*.

```
#!/usr/bin/python
# -*- coding: utf-8 -*-

import RPi.GPIO as GPIO    #Importamos la libreria RPi.GPIO
import time                #Importamos time para poder usar time.sleep
import sys
GPIO.setmode(GPIO.BCM)   #Ponemos la Raspberry en modo BOARD
GPIO.setwarnings(False)
GPIO.setup(20,GPIO.OUT)    #Ponemos el pin 20 como salida
GPIO.cleanup()             #Limpiamos los pines GPIO de la Raspberry y cerra%co
de%nbsp;
sys.exit()
```

5.3.1.2.6.3 Temperatura.

El sensor de temperatura DHT 11, esta conectado al PIN 4 de GPIO BCM, para ello en el programa se asigna los parametros y se utiliza la librería de *Adafruit_DHT* diseñada para este sensor. Cabe resaltar que la medicion de la temperatura se realiza con unidades de grados Celsius y Para humedad en porcentaje %

```
#!/usr/bin/python
import sys
import Adafruit_DHT
while True:
humidity, temperature = Adafruit_DHT.read_retry(11, 4)
print 'Temp: {0:0.1f} C Humidity: {1:0.1f} %'.format(temperature, humidity)
```

5.3.1.2.7 *Domoticz.*

El software domoticz es un sistema de domótica para controlar varios dispositivos, recibiendo, analizando y mostrando información de diferentes sensores. Este sistema se puede utilizar con sensores como: Interruptores de luces, Sensores de puertas, Timbres, Dispositivos de seguridad, Sensores de tiempo: UV/sensores de lluvia/anemómetros, Sensores de temperatura, Medidores de pulsos y Medidores de Voltaje / AD. (“Manual de Domoticz en Español”, s/f)

Para este sistema se utiliza el servicio *Apache/Web* y los puertos analógicos para los sensores como temperatura, humedad, indicadores (leds) y Fans (Servo Motores). Esto con el fin de manejar, recolectar y visualizar la información los sensores.

Para instalar *domoticz* se ejecuta el siguiente comando.

sudo curl –L –k install.domoticz.com | sudo bash

Adicionalmente se ejecuta el siguiente comando, donde son complementos necesarios para *domoticz*.

sudo apt-get install cmake make gcc g++libssl-dev git curl libcurl4-openssl-dev libusb-dev wiringpi

Otra manera de instalar es.

git clone https://github.com/domoticz/domoticz.git
carpeta dev-domoticz

5.3.1.2.8 Homebridge.

Homebridge es un módulo de software que proporcionan un puente de red a la API *HomeKit* de iOS.

Para este módulo se necesita instalar los siguientes componentes para que pueda funcionar correctamente.

apt-get install g++
apt-get install –y nodejs
apt-get install libavahi-compat-libdnssd-dev

Para instalar *homebridge* con sus complementos se ejecutan los siguientes comandos.

sudo npm install –g --unsafe-perm homebridge hap-nodejs node-gyp
cd /usr/local/lib/node_modules/homebridge/

```
sudo npm install--unsafe-perm bignum cd/usr/local/lib/node_modules/hap-nodejs/node_modules/mdns
sudo node-gyp BUILDTYPE=Release rebuild
```

Para la configuración del módulo se realiza los siguientes pasos, *Homebridge* se ubica en un directorio llamado /home/pi/homebridge y su configuración se ubica en ~/.homebridge# en el archivo config.json, en el cual se muestra los parámetros utilizados para conectar *domoticz*.

El siguiente código es el utilizado para la arquitectura *fog computing*.

```
{
    "bridge": {
        "name": "Homebridge",
        "username": "CC:22:3D:E3:CE:30",
        "port": 51826,
        "pin": "031-45-154"
    },
    "description": "Configuration file for Domoticz platform.",
    "platforms": [{
        "platform": "Domoticz",
        "name": "Domoticz",
        "server": "192.168.10.1",
        "port": "8080",
        "roomid": 0,
        "loadscenes": 2,
        "mqttenable": 1,
        "mqttserver": "192.168.10.1",
        "mqttport": "1883",
        "mqttauth": 0,
        "mqttuser": "",
        "mqttpass": ""
    }],
    "accessories": [{
        "accessory": "Dht",
        "name": "CPU",
        "service": "Temperature"
```

```
    }, {
        "accessory": "Dht",
        "name": "Temp/Humidity Sensor",
        "service": "dht22"
    }]
}
```

A continuación, se habilita y ejecuta el servicio (primera vez) con los siguientes comandos:

```
systemctl daemon-reload
systemctl enable homebridge
systemctl start homebridge
```

Para revisar el estado del servicio con el siguiente comando.

```
systemctl status homebridge
```

5.3.1.2.9 *HomeKit.*

HomeKit es un *framework* de iOS que permite a los usuarios de *iPhone/iPad*, comunicarse, controlar y configurar los electrodomésticos inteligentes. Mediante el uso de una aplicación llamada Home, se puede diseñar espacios, alertas y acciones que se integran a los sensores.

Los usuarios también pueden activar las acciones automáticas a través de un comando de voz utilizando Siri. *HomeKit* pueden habilitarse para su uso a través de un *hotspot*, como un concentrador que conecta entre estos dispositivos y el servicio *HomeKit*.

5.3.1.2.10 *Siri.*

Siri es un asistente personal inteligente, que forma parte de los sistemas operativos *iOS, watchOS, macOS y tvOS* de *Apple Inc*. La característica principal es que utiliza una interfaz de usuario de lenguaje natural para responder preguntas, hacer recomendaciones y realizar acciones delegando solicitudes a un conjunto de servicios *web*.

El software como tal, es una función de *iOS*, se adapta al uso del idioma del usuario y las búsquedas de este con el uso continuo, y devuelve los resultados que se individualizan.

Figura 26. Módulos de *software* utilizados para la arquitectura.

Fuente Autor.

5.4 MEDICIONES PERTINENTES EN CUANTO A TRÁFICO DE RED, QUE EVIDENCIEN LA VENTAJA DE *FOG COMPUTING* PARA LA DISMINUCIÓN DE TRÁFICO DE RED HACIA LA NUBE.

Se presenta un análisis del tráfico de la red, donde se enmarca dentro la arquitectura, Presentación de resultados.

5.4.1 Funcionamiento de la arquitectura *fog computing*.

Las pruebas de funcionamiento se pueden ver en un dispositivo movil, tablet y *desktop* (computador de escritorio) via inalámbrica, permitiendo mayor movilidad y ahorro de conexiones fisicas innecesarias.

Al realizar la configuración de *CoovaChilli* se puede acceder a la red inalambrica y acceder a los servicios de *domoticz* y *hombridge*.

La configuración inicial de *homebridge* se realiza con los parametros de inicializar el servicio de exploración del puerto 51826, tambien los servicios de *MQTT* y *HTTP*, el cual conecta a la aplicación *Home* de *iOS* 10.

En la figura 20, se muestra el panel de sensores conectados y el estado el cual están (encendido o apagado)

Figura 27. *Dashboard web* de los sensores

Fuente Autor.

Las figuras 27, 28, 29, 30 (fuente autor) muestran las pruebas realizadas con un dispositivo móvil conectado vía *Wi-Fi* donde ingresa al portal web donde se encuentran los sensores y los registros de ellos, además se ingresa a la aplicación *Home* del *framework* Home Kit de *Apple* y se comprueba la interacción, posteriormente se realiza la prueba con el sistema de reconocimiento de voz, *Siri*. El cual se envía y recibe órdenes establecidas.

5.4.2 Escenarios propuestos.

Se presenta los resultados en base a los diferentes escenarios propuestos para evaluar.

5.4.2.1 Escenario: Desconexión total.

En el escenario de desconexión total se puede evidenciar automáticamente se puede inferir que la disminución del consumo de envío de datos hacia la nube, ya que no es obligatoria.

Una vez descrito lo anterior, se realizan las pruebas para determinar la indicación, la manipulación y el almancenaje de un registro de datos adquiridos por la plataforma *domoticz*. Como se muestran en la figura 23, con ello se evidencia un reporte de los datos de temperatura.

Por lo anterior genera una disminución de la utilización de la red y disminución de utilización la nube.

Figura 27. App Home Kit con los sensores activos/inactivos

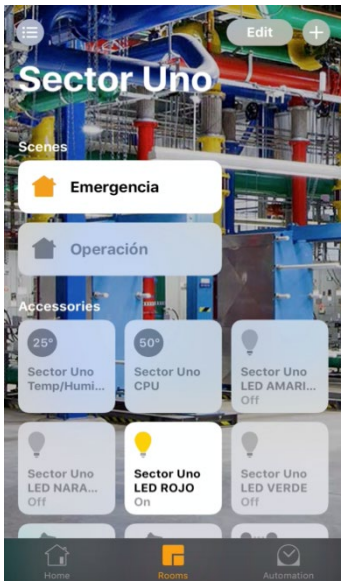


Figura 28 Escenarios donde se puede asignar

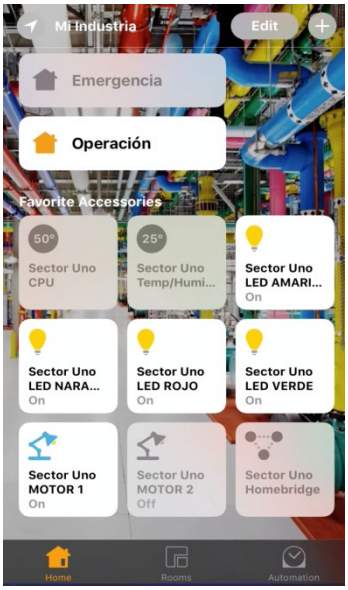


Figura 29 Presentación Web del registro del sensor de temperatura y humedad.

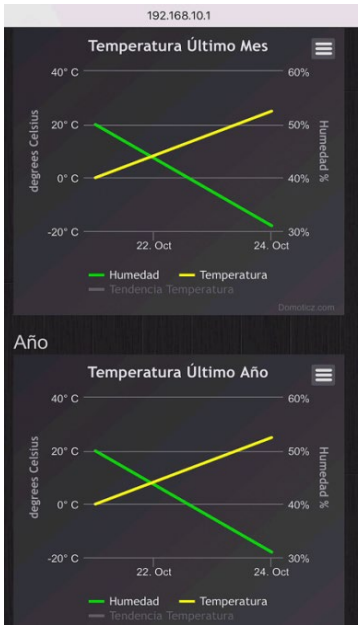


Figura 30 Demostración con Siri y la interacción con los sensores.

5.4.2.2 Escenario: Comparativo entre *IoT* y *Fog Computing.*

Para la comparación la arquitectura ambas arquitecturas se seleccionó el *framework* de *IoT* llamado *cayenne* de la compañía *myDevices*. Al realizar esta comparación por las características semejantes al *framework domoticz*. (“myDevices”, 2017)

Diferentes sensores indicadores son conectados a *cayenne* como se muestra en la figura 25, para realizar la medición se utiliza un *framework* de medición por DNS llamado *OpenDNS* como se muestra en la figura 26, en el cual mide las peticiones recibidas y enviadas. Esto con el fin de realizar la medición de peticiones requeridas hacia la nube, en este caso la nube es la dirección update.mydevices.com.

A continuación, se toma un parámetro de 5 días para realizar la medición y saber cuánto fue el número de peticiones exigido por el dispositivo *Raspberry Pi* hacia la nube. Esto no incluye tamaño de los paquetes ni tiempo que tarda en llegar a su destino, esto debido al enfoque que exige el proyecto.

Por lo anterior, se muestra los resultados de acuerdo con las mediciones tomadas.

Tabla 14. Resultados de Peticiones durante 5 días.

Hour	Date	Requests
0:00:00	1/01/17	0
1:00:00	1/01/17	0
2:00:00	1/01/17	0
3:00:00	1/01/17	0
4:00:00	1/01/17	0
5:00:00	1/01/17	0
6:00:00	1/01/17	0
7:00:00	1/01/17	0
8:00:00	1/01/17	0
9:00:00	1/01/17	0
10:00:00	1/01/17	0
11:00:00	1/01/17	0
12:00:00	1/01/17	0
13:00:00	1/01/17	0
14:00:00	1/01/17	0
15:00:00	1/01/17	0
16:00:00	1/01/17	0
17:00:00	1/01/17	0
18:00:00	1/01/17	677
19:00:00	1/01/17	541
20:00:00	1/01/17	448
21:00:00	1/01/17	241
22:00:00	1/01/17	217
23:00:00	1/01/17	147
0:00:00	2/01/17	171
1:00:00	2/01/17	275
2:00:00	2/01/17	453
3:00:00	2/01/17	831
4:00:00	2/01/17	459
5:00:00	2/01/17	1160
6:00:00	2/01/17	1045
7:00:00	2/01/17	806
8:00:00	2/01/17	297
9:00:00	2/01/17	268
10:00:00	2/01/17	700
11:00:00	2/01/17	356
12:00:00	2/01/17	233
13:00:00	2/01/17	730
14:00:00	2/01/17	779
15:00:00	2/01/17	604
16:00:00	2/01/17	731
17:00:00	2/01/17	554
18:00:00	2/01/17	895
19:00:00	2/01/17	1023
20:00:00	2/01/17	1026

Tabla 14. (continuación)

Hour	Date	Requests
21:00:00	2/01/17	422
22:00:00	2/01/17	225
23:00:00	2/01/17	265
0:00:00	3/01/17	259
1:00:00	3/01/17	252
2:00:00	3/01/17	342
3:00:00	3/01/17	384
4:00:00	3/01/17	505
5:00:00	3/01/17	1181
6:00:00	3/01/17	1002
7:00:00	3/01/17	522
8:00:00	3/01/17	256
9:00:00	3/01/17	333
10:00:00	3/01/17	1020
11:00:00	3/01/17	736
12:00:00	3/01/17	219
13:00:00	3/01/17	143
14:00:00	3/01/17	202
15:00:00	3/01/17	266
16:00:00	3/01/17	454
17:00:00	3/01/17	608
18:00:00	3/01/17	759
19:00:00	3/01/17	676
20:00:00	3/01/17	287
21:00:00	3/01/17	230
22:00:00	3/01/17	209

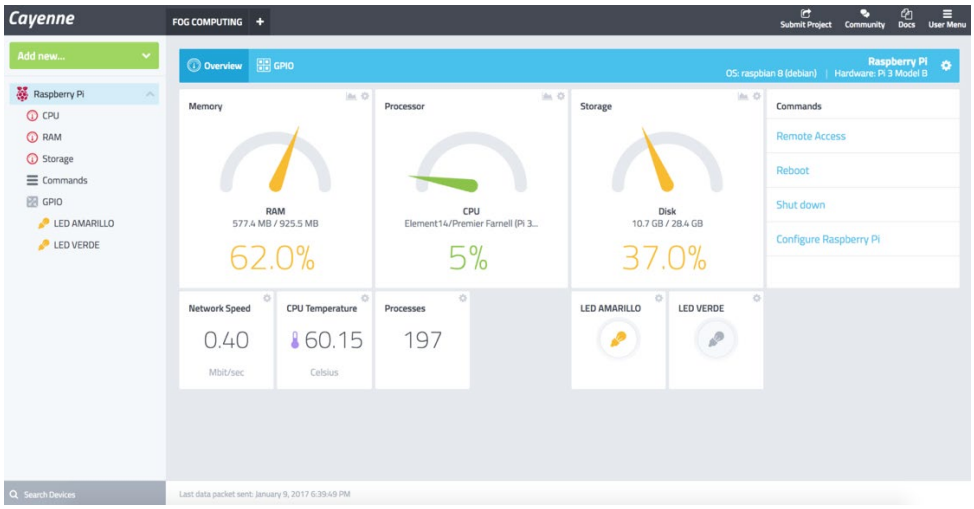
Hour	Date	Requests
23:00:00	3/01/17	209
0:00:00	4/01/17	232
1:00:00	4/01/17	153
2:00:00	4/01/17	357
3:00:00	4/01/17	330
4:00:00	4/01/17	232
5:00:00	4/01/17	695
6:00:00	4/01/17	1066
7:00:00	4/01/17	821
8:00:00	4/01/17	750
9:00:00	4/01/17	624
10:00:00	4/01/17	1011
11:00:00	4/01/17	607
12:00:00	4/01/17	347
13:00:00	4/01/17	277
14:00:00	4/01/17	278
15:00:00	4/01/17	224
16:00:00	4/01/17	362
17:00:00	4/01/17	299
18:00:00	4/01/17	881
19:00:00	4/01/17	1282
20:00:00	4/01/17	536
21:00:00	4/01/17	354
22:00:00	4/01/17	347
23:00:00	4/01/17	324
0:00:00	5/01/17	189

Hour	Date	Requests
1:00:00	5/01/17	215
2:00:00	5/01/17	319
3:00:00	5/01/17	573
4:00:00	5/01/17	398
5:00:00	5/01/17	439
6:00:00	5/01/17	545
7:00:00	5/01/17	341
8:00:00	5/01/17	254
9:00:00	5/01/17	294
10:00:00	5/01/17	937
11:00:00	5/01/17	0
12:00:00	5/01/17	0
13:00:00	5/01/17	0
14:00:00	5/01/17	5
15:00:00	5/01/17	217
16:00:00	5/01/17	368
17:00:00	5/01/17	0
18:00:00	5/01/17	0
19:00:00	5/01/17	0
20:00:00	5/01/17	0
21:00:00	5/01/17	0
22:00:00	5/01/17	0
23:00:00	5/01/17	0

Fuente: Autor

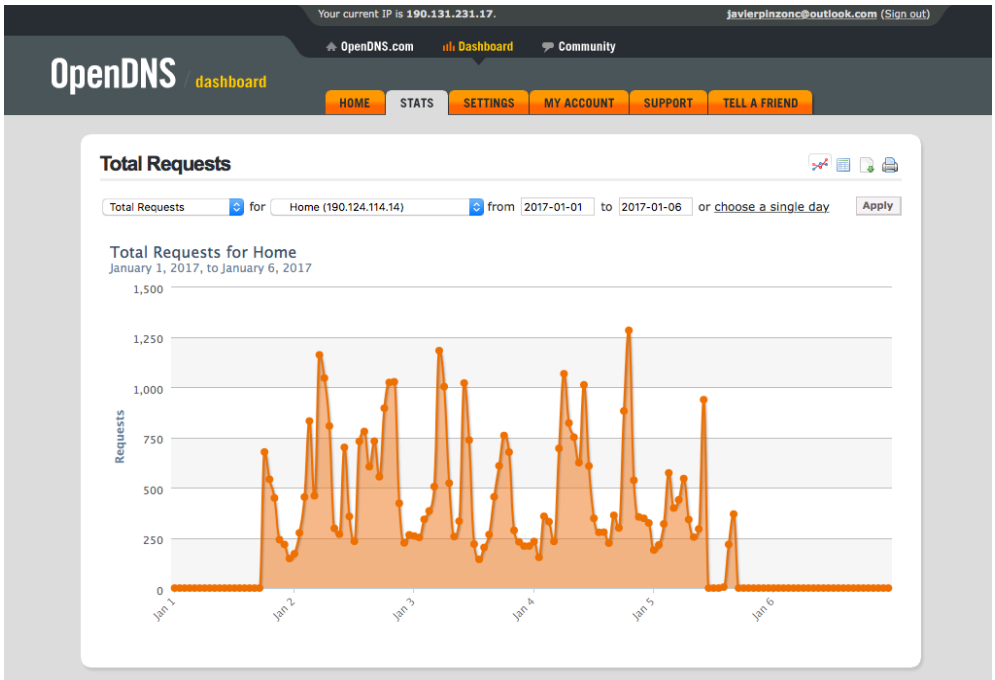
Al obtener resultados se grafican los datos de las peticiones durante los 5 días, donde se muestra la tendencia realizada durante la medición. Donde se muestran las diferentes tendencias de peticiones por hora y por día. Por lo que marcan un comportamiento no constante sino por demanda de petición, es decir, si es requerido el servicio es registrado y enviado a cumplir su tarea.

Figura 28 Dashboard Cayenne.



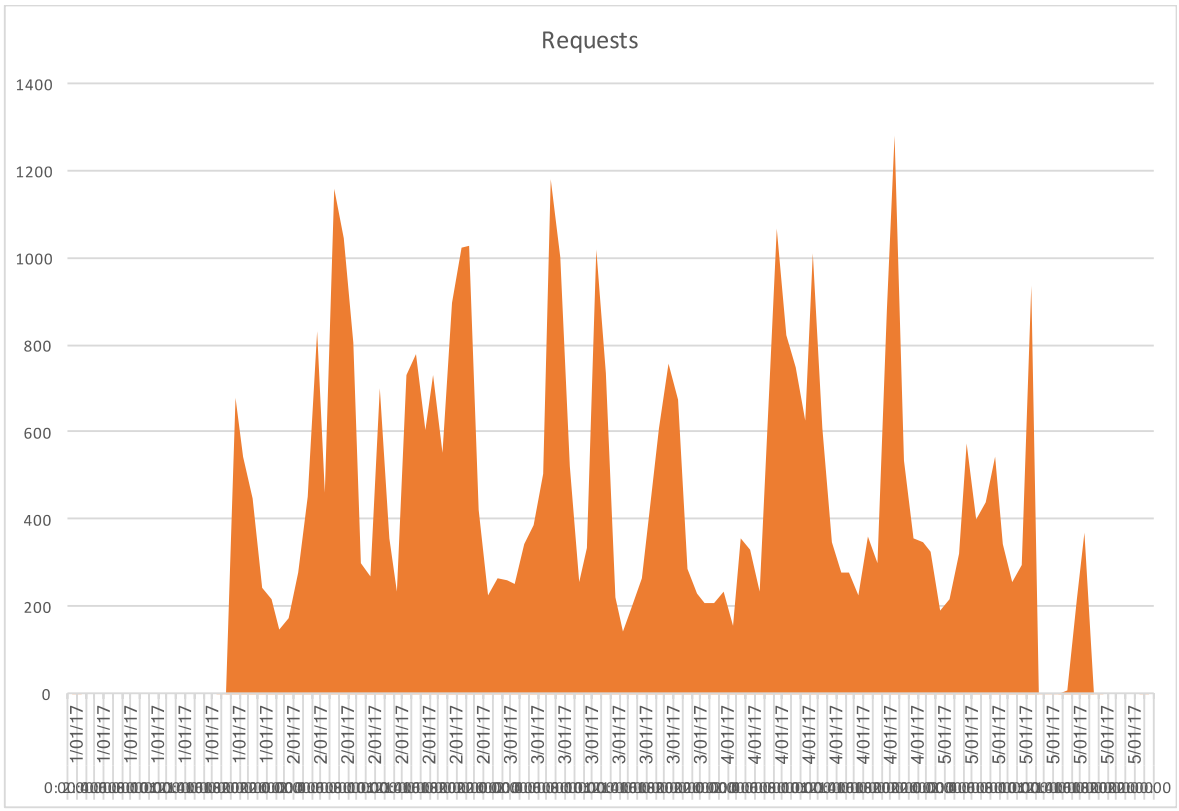
Fuente: myDevices, recuperado de http://cayenne.mydevices.com

Figura 29 Dashboard *OpenDNS*.



Fuente *OpenDNS*, recuperado de http://dashboard.opendns.com.

Figura 30 Gráfica de peticiones durante 5 días.



Fuente Autor.

La siguiente tabla se muestra los diferentes dominios y la cantidad de veces requeridos durante el periodo de tiempo establecido.

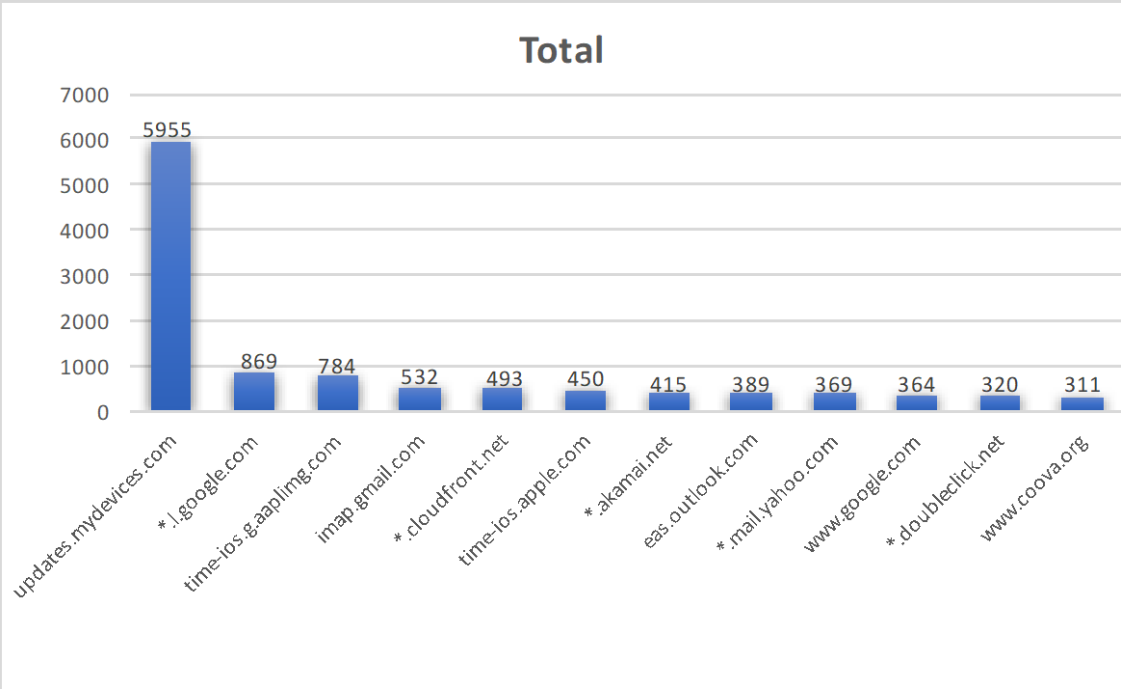
Tabla 15. Total de Peticiones por Dominio

Dominios	**Total**
updates.mydevices.com	5955
*.l.google.com	869
time-ios.g.aaplimg.com	784
imap.gmail.com	532
*.cloudfront.net	493
time-ios.apple.com	450
*.akamai.net	415

Dominios	Total
eas.outlook.com	389
*.mail.yahoo.com	369
www.google.com	364
*.doubleclick.net	320
www.coova.org	311

Fuente Autor.

Figura 31. Gráfico de los resultados de las peticiones por dominio.



Fuente Autor.

Los resultados muestran un alto nivel de petición hacia el dominio update.mydevices.com, esto con relación a la confirmación de la gran demanda de acceso a la nube, por su enfoque de *IoT.* Por lo anterior las mediciones pertinentes en cuanto a tráfico de red, que evidencien la ventaja de *Fog Computing* quedan comprobadas de acuerdo con la implementación de la arquitectura. Adicionalmente se confirma con los estudios realizados por los autores (Sarkar & Misra, 2016) en la reducción de procesamiento, servicios y comunicación en la nube.

Por lo anterior los resultados demuestran que la arquitectura de *fog computing* es un complemento de la computación en la nube y un nuevo enfoque al internet del todo.

5.5 APLICACIONES DE *FOG COMPUTING*

Para la arquitectura de *fog computing* existen diversas áreas donde se pueden aplicar, para el proyecto se enfatizaron en cuatro de gran importancia y que están soportadas por trabajos de implementación e investigación, como se muestra a continuación.

5.5.1 *Fog computing* en domótica.

En el área de domótica, el proyecto desarrollado se orienta a este tema, buscando la utilización de los diferentes elementos de hardware de automatización para cumplir una tarea en el ámbito de hogar y edificios inteligentes. Es por ello que la propuesta es basada en el proyecto implementado llamado *Home automation using raspberry Pi through Siri enabled mobile devices* de los autores (Celebre et al., 2015). En el cual desarrollan una propuesta de domótica utilizando *fog computing* y la aplicación de tecnologías como *Raspberry Pi* y el asistente *Siri* dentro de un ambiente industrial.

En la figura 31 se muestra cómo se realiza un escenario de *fog computing* en domótica*,* donde se muestra la plataforma de *fog computing* conectada a elementos cotidianos e industriales y la utilización de computación en la nube para diversas peticiones.

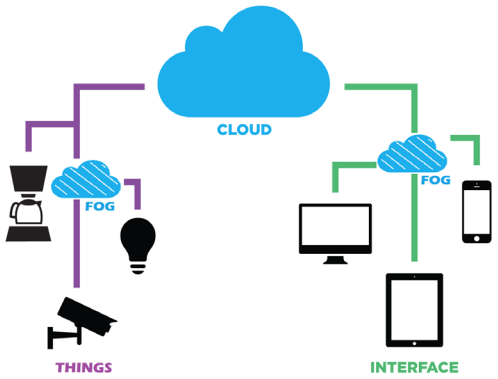
El proyecto de investigación desarrolla elementos de software a la medida, acondicionados y utilización de *frameworks* en función de formar una arquitectura de *fog computing* es por ello que el autor logra una interconexión entre herramientas de trabajo libres, para cumplir los módulos requeridos para completar la arquitectura. Es por esto que el autor recomienda seguir con el proceso de interconexión de *frameworks* entre ello la reutilización y el desarrollo de parámetros e instrucciones donde sea visible una metodología ágil para impulsar la arquitectura.

Los inconvenientes presentados durante el proyecto son las limitaciones de procesamiento del hardware ya que existe un nivel máximo para procesar los datos de los sensores, es por ello que se recomienda realizar una interconexión entre arquitecturas *fog computing.* La utilización de diferentes *frameworks* puede presentarse como benéfico o perjudicial, ya que no se tiene control sobre cada módulo y solo se limita al resultado que arroje el componente. Esto puede elevar el nivel de procesamiento dedicado para cada módulo y retardar la entrega de resultados consolidados para enviar a la siguiente arquitectura de nube.

Para *cloud computing* se utilizan nubes públicas como *Amazon Web Services, Azure* de *Microsoft, Cayenne* entre otras, con el fin de programar, insertar algoritmos y diferentes procesos que logren que la información recopilada por la arquitectura *Fog* sea almacenada, generen alertas, activen un registro entre otras variables que puedan hospedarse en la nube y puedan ser consultada después.

En la figura 32 se muestra la arquitectura de *fog computing* dentro de un ámbito de hogar o edificio inteligente, donde se presenta diferentes conexiones como sensores de temperatura y sensores de movimiento, estado de los electrodomésticos, encendido y apagado, sistema de video vigilancia, acceso a puertas donde puedo obtener estadísticas del accionamiento de cada elemento y la información generada puedo realizar análisis, alertas entre otros para optimizar mi consumo dentro del hogar

Figura 32. Estructural de *fog computing* en domótica.



Fuente: Tomada de *http://iot-labs.com.my/wp-content/uploads/2016/03/cloud-fog-and-things-300x225.png.*

Figura 33. Arquitectura *Fog Computing* en Domótica

Fuente: modificada por el autor, tomada de *http://www.cds.net/blog/2015/10/iot-testing-big-data-collection-storage-limits/.*

5.5.2 *Fog computing* en la industria.

En el área de industria se busca la centralización de la información logrando así tener los indicadores más importantes de los equipos como son temperatura, humedad, encendido/apagado entre otros, y sean visibles desde los equipos como moviles, tablets, entre otros. Los datos no son enviados directamente a internet sino se quedan en la plataforma de *fog computing*, el punto donde son procesados y si es necesario se envían a la computación en la nube para realizar análisis y registros.

El uso es netamente industrial, ya que el dispositivo se conecta a una máquina o un espacio industrial, para realizar las medidas con los diferentes sensores e indicar, registrar y analizar los datos desde una plataforma *web* dentro y/o fuera de la empresa, como se muestra en la figura 35. Las aplicaciones de análisis, estadísticas industriales y comportamiento de los equipos dentro de una zona industrial son el toque diferenciador de la arquitectura ya que integra el ámbito industrial con tecnologías cotidianas, como el uso de la web, redes inalámbricas y acceso a la nube.

Figura 34. Esquema de la arquitectura *fog computing* dentro de una industria.

Fuente: *LocalGrid Fog Computing http://www.localgridtech.com/wp-content/uploads/2014/03/smart_mfg_platter-e1421883782596.jpg*

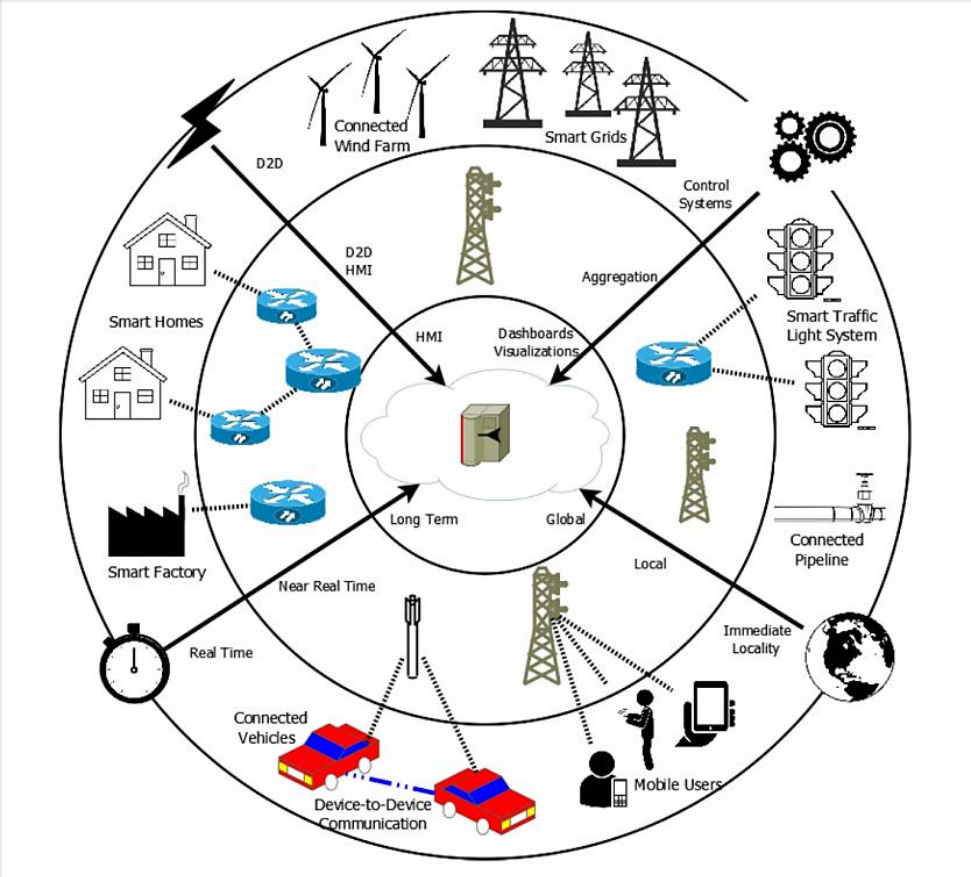
5.5.3 *Fog Computing* en las ciudades inteligentes.

Para el área de ciudades inteligentes, se enmarca en un ecosistema donde tenemos casas inteligentes, fabricas, vehículos, semáforos, tuberías, postes de luz, generadores de energía hasta los mismos ciudadanos con dispositivos inteligentes. Es por ello que dentro de la investigación se basa en un trabajo realizado por los autores (Dastjerdi et al., 2016) en el cual muestran los beneficios y explican cada capa de desde el dispositivo, sensores y elementos finales, hasta la capa de recopilación de información en la nube.

En la figura 35, se muestra cada capa y área donde se utilizan los diferentes tipos de comunicación, ya sea móvil o cableada, también los diferentes protocolos que se utilizan como *Device to Device* (D2D), Interfaz Maquina Humano (HMI) y la posición geográfica de los elementos del *fog computing* (inmediatamente local, local y global), esto relacionado a la toma de datos en tiempo real, cerca del tiempo real y largo plazo.

Todo lo anterior relacionado a la muestra de una ciudad inteligente, donde hay diferentes tipos de escenarios, por ejemplo; los datos de un conjunto de casas son enviados a la arquitectura de *fog computing* y estos a su vez son transformados en información y son enviados a la nube donde puede verse desde cualquier dispositivo móvil los reportes y estadísticas requeridas para este caso. Para la ciudad, los semáforos son parte importante de la movilidad, es por ello que se conectan a una capa de *fog computing* y el consolidado de la información es enviado a los sistemas en la nube para su posterior análisis y almacenaje estadístico.

Figura 35. Beneficios de *fog computing* dentro de ciudades inteligentes.



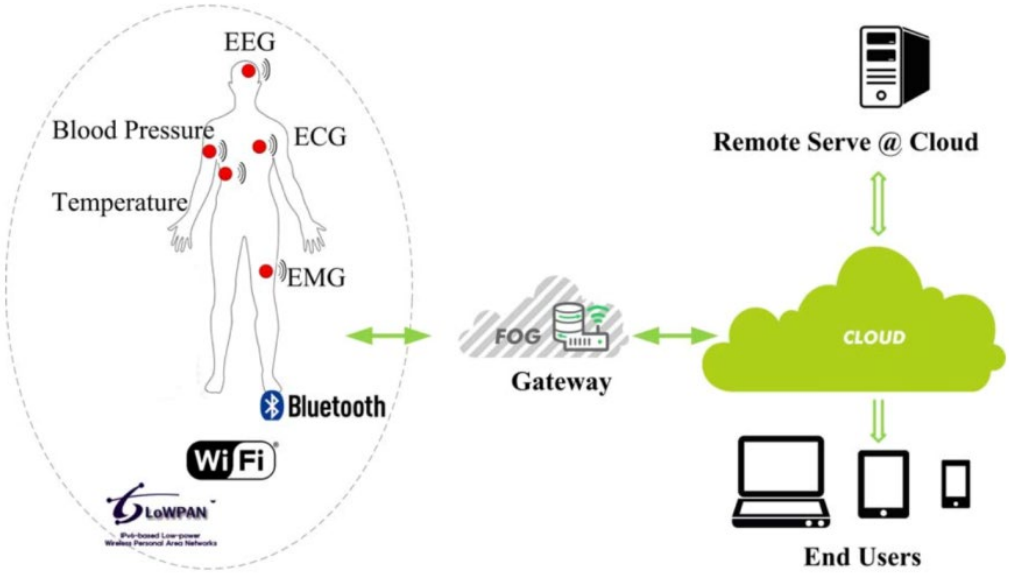
Fuente: Figura 4.4 del trabajo *Fog Computing: Principles, Architectures, and Applications* de (Dastjerdi et al., 2016)

5.5.4 *Fog computing* en medicina.

Las aplicaciones de *fog computing* para el área de la salud existe un caso de éxito que está muy cercano a la estructura de este proyecto y es un trabajo con título "*Fog Computing in Healthcare Internet of Things: A Case Study on ECG Feature Extraction*". De los autores (Gia et al., 2015) donde muestran cómo trabajan con diversos sensores conectados a un paciente y los datos recopilados son enviados hacia la capa de *fog computing,* donde son procesados y posteriormente enviados hacia la capa de *cloud computing* y son almacenados en un servidor para ser visualizados por los usuarios finales.

Los sensores utilizados son de: Electroencefalografía (EEG), Electrocardiografía (ECG), Electromiografía (EMG), Presión sanguínea y Temperatura. Los cuales son utilizados para supervisar el estado del paciente y se utilizaron dichas mediciones para enviar los datos hacia la computación de niebla. Esta realiza un filtrado de la información por medio de cálculos estadísticos, con el fin de reducir el tráfico de datos hacia la nube, ya que lo que se transfiere es la información relevante para realizar un registro de la información y almacenamiento para posteriores análisis.

Figura 36. Sistema de monitoreo médico utilizando *Fog Computing*.



Fuente: (Gia et al., 2015)

6. CONCLUSIONES Y RECOMENDACIONES

Al terminar el proyecto, se logra la implementación de la arquitectura de *fog computing*.

El presente documento establece un marco de referencia para construir e implementar la arquitectura de *fog computing*, con las plataformas de *hardware* y *software* con administración *web* y recursos abstractos e intuitivos que hacen más fácil el entendimiento logrando así nuevas ideas y recursos intelectuales que generen alternativas tanto *hardware* como en *software*, para implementar esta nueva tendencia en la academia.

Existen dos limitantes en la implementación de este tipo de tecnología, En primer lugar, se requiere una plataforma de procesamiento robusta, para operar con mayor fluidez los diferentes sistemas de almacenamiento, estructura de archivos y comunicaciones. La segunda limitante está ligada al uso de los sistemas operativos basados en *linux*.

Se realizó un análisis de tráfico con el fin de entender qué comunicaciones se utilizan en la arquitectura, donde se concluye que las comunicaciones *HTTP* y *MTTQ* están disponibles para esta arquitectura.

Todos los módulos utilizados para este propósito cumplen la meta propuesta para el proyecto. El resultado se obtuvo, tal como se había encontrado en la revisión de la literatura, el desarrollo de *software* a la medida para trabajar en conjunto con el hardware determinado para él proyecto, la utilización de protocolos de comunicación recomendados para compartir datos dentro de la arquitectura. En general se observó el procesamiento de los datos de los sensores antes que fueran transportados hacia la arquitectura de nube y la independencia de la misma.

Recomendaciones para el proyecto, es fundamental la escogencia del *hardware* ya que con ella se puede interactuar los diferentes módulos de *software*. Para el caso se sugiere continuar los pasos de la investigación donde se integren los diferentes

tipos de módulos, tales como comunicaciones, adquisición, procesamiento y visualización de datos, manejo de usuarios y ampliación de la capa de seguridad de la información.

Este proyecto es un paso importante para el desarrollo e implementación de las herramientas de computación de borde (*fog computing*), ya que es un beneficio para la comunidad académica tener estos resultados y recursos generados, debido a que se establece un marco de referencia para una solución escalable y adaptable a los requerimientos.

7. REFERENCIAS

- Aazam, M., & Huh, E.-N. (2016). Fog Computing: The Cloud-IoT\/IoE Middleware Paradigm. *IEEE Potentials*, *35*(3), 40–44. https://doi.org/10.1109/MPOT.2015.2456213
- Ashwini, T., & SG, M. A. (2015). Fog Computing to protect real and sensitivity information in Cloud. Recuperado a partir de http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.675.846&rep=rep1&type=pdf
- Bonomi, F., Milito, R., Zhu, J., & Addepalli, S. (2012). Fog computing and its role in the internet of things. En *Proceedings of the first edition of the MCC workshop on Mobile cloud computing* (pp. 13–16). ACM.
- Celebre, A. M. D., Dubouzet, A. Z. D., Medina, I. B. A., Surposa, A. N. M., & Gustilo, R. C. (2015). Home automation using raspberry Pi through Siri enabled mobile devices. En *2015 International Conference on Humanoid, Nanotechnology, Information Technology,Communication and Control, Environment and Management (HNICEM)* (pp. 1–6). https://doi.org/10.1109/HNICEM.2015.7393270
- Chang, K.-D., Chen, J.-L., Chen, C.-Y., & Chao, H.-C. (2012). IoT operations management and traffic analysis for Future Internet. En *Computing, Communications and Applications Conference (ComComAp), 2012* (pp. 138–142). IEEE.
- Dastjerdi, A. V., Gupta, H., Calheiros, R. N., Ghosh, S. K., & Buyya, R. (2016). Fog Computing: Principals, Architectures, and Applications. *arXiv preprint arXiv:1601.02752*.
- Dr. Deepti Sharma, P. K. (2015). A Detail Review on Cloud, Fog and Dew Computing. *International Journal of Science, Engineering and Technology Research (IJSETR)*, *5*(5), 9.
- Gia, T. N., Jiang, M., Rahmani, A.-M., Westerlund, T., Liljeberg, P., & Tenhunen, H. (2015). Fog Computing in Healthcare Internet of Things: A Case Study on ECG Feature Extraction (pp. 356–363). IEEE. https://doi.org/10.1109/CIT/IUCC/DASC/PICOM.2015.51
- Hakyong KIM. (2014, julio 11). Comparison among Cloud Computing, Fog Computing, and Dew Computing. Recuperado a partir de http://hakyongkim.net/Sharing/style/computing.png
- Homekit Siri - Domoticz. (2016). Recuperado el 2 de enero de 2017, a partir de https://www.domoticz.com/wiki/Homekit_Siri

- Hurwitz, J., Bloor, R., Kaufman, M., & Halper, F. (2010). *Cloud Computing For Dummies.* Indianapolis, Indiana.: Wiley Publishing, Inc,.
- *IoT Developer Survey*. (2016). Recuperado a partir de http://iot.ieee.org/images/files/pdf/iot-developer-survey-2016-report-final.pdf
- Manual de Domoticz en Español. (s/f). Recuperado a partir de http://www.domoticz.com/DomoticzManual_es.pdf
- myDevices. (2017). Recuperado el 10 de enero de 2017, a partir de http://mydevices.com/about/
- Pavel Pohanka. (2015). Internet of Things. Recuperado el 4 de agosto de 2016, a partir de http://i2ot.eu/en/internet-of-things/
- Salman, O., Elhajj, I., Kayssi, A., & Chehab, A. (2015). Edge computing enabling the Internet of Things. En *Internet of Things (WF-IoT), 2015 IEEE 2nd World Forum on* (pp. 603–608). IEEE.
- Sarkar, S., & Misra, S. (2016). Theoretical modelling of fog computing: a green computing paradigm to support IoT applications. *IET Networks*, *5*(2), 23–29.
- Skala, K., Davidovic, D., Afgan, E., Sovic, I., & Sojat, Z. (2015). Scalable distributed computing hierarchy: Cloud, fog and dew computing. *Open Journal of Cloud Computing (OJCC)*, *2*(1), 16–24.
- Stojmenovic, I., & Wen, S. (2014). The fog computing paradigm: Scenarios and security issues. En *Computer Science and Information Systems (FedCSIS), 2014 Federated Conference on* (pp. 1–8). IEEE. Recuperado a partir de http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=6932989
- Szilagyi, I., & Wira, P. (2016). Ontologies and Semantic Web for the Internet of Things - a survey. En *IECON 2016 - 42nd Annual Conference of the IEEE Industrial Electronics Society* (pp. 6949–6954). https://doi.org/10.1109/IECON.2016.7793744
- What is Siri? Apple's personal voice assistant explained. (2015, octubre 12). Recuperado el 4 de enero de 2017, a partir de http://www.pocket-lint.com/news/112346-what-is-siri-apple-s-personal-voice-assistant-explained.