UNIVERSIDAD CENTRAL DE VENEZUELA
FACULTAD DE CIENCIAS
ESCUELA DE COMPUTACIÓN

Gateway de Internet de las Cosas Optimizado para Computación en la Niebla bajo Ambientes Domoticos

Pedro Luis Boll Lugo
Cédula: 20.173.376

Tutores:
Prof. Jesús Lares
Prof. Antonio Russoniello

Caracas, Octubre 2017

Resumen

Las tecnologías que están siendo desarrolladas basadas en el Internet de las Cosas, cada vez más tienen un poder de cómputo mayor y estarán acompañados con sistemas inteligentes alojados en la nube y que serán alimentados con un volumen de datos enorme. El poder aprovechar tal cantidad de datos para conseguir nuevos conocimientos que no se pueden distinguir trivialmente brinda una serie de oportunidades a organizaciones y particulares de entender de mejor manera lo que los rodea. El hogar es uno de esos ámbitos en donde la próxima generación de dispositivos inteligentes recolectarán información útil para la optimización y automatización de los procesos que son llevados a cabo en las casas.

Para ello es necesario adecuar y/o crear nuevas infraestructuras tecnológicas capaces de poder tratar con todos estos dispositivos conectados, los datos que producen y que generen un valor agregado a la experiencia final en el uso de ellos por parte de los usuarios, por tanto es necesario concentrar esfuerzos en cubrir las necesidades y requerimientos planteados para este fin, además de diseñar y elegir qué tecnologías, herramientas y métodos son los más adecuados en la implementación y despliegue de dichas soluciones tecnológicas.

Palabras Claves: Internet de las Cosas, Big Data, Ciencia de datos, domótica, automatización del hogar, Edge Computing, Apache Hadoop, Apache Spark, Minería de datos, Optimización de procesos, arquitecturas distribuidas.

Índice general

Índice de figuras

Índice de tablas

Capítulo 1

Marco Introductorio

El mundo de la domótica y de la automatización en los hogares es algo que ha estado presente en el mercado desde hace cuatro décadas atrás, pero los costos de implantarlos en los hogares ha sido, en muchos casos exorbitantes, sin embargo, con el paso del tiempo se han vuelto cada vez más asequibles y abarcan más aspectos del hogar.

Por otro lado, la necesidad actual de acceder a la mayor cantidad de conocimientos en los datos que se encuentran disponibles en los sistemas computacionales ha generado un auge por las tecnologías enfocadas en la ciencia de datos y en el uso de grandes volúmenes de datos, con las que se busca aplicar técnicas, herramientas y métodos de disciplinas como la inteligencia artificial, la computación distribuida, la matemáticas y la estadística y así obtener dicho conocimiento no trivial.

Otro termino cuyo impacto en el mediano y largo plazo revolucionará casi todos los aspectos de nuestras vidas es el del Internet de las Cosas, el cual se refiere a dispositivos, artefactos y electrodomésticos con la capacidad de comunicarse entre ellos y otros sistemas y que cuentan con características de cómputo y funcionalidades extendidas que los hacen " inteligentes", como podemos ver en la primera serie de teléfonos, televisores, neveras, lavadoras, relojes y ropa diseñada con tecnologías embebidas. Estos presentan nuevos desafíos, como la integración entre sistemas y dispositivos y cómo manejar toda la información que generan los sensores de los dispositivos que día tras día están más presentes en nuestras vidas.

Cuando unimos estas tres vertientes de desarrollo tecnológico, estamos presenciando uno de los cambios más amplios en la manera en que haremos vida, incluyendo a los espacios de nuestros hogares. Los dispositivos de Internet de las Cosas pueden hacer mediciones y actuar sobre los recursos como el uso de servicios básicos, la seguridad y vigilancia, la accesibilidad y el entretenimiento y otros sistemas, que administrando y procesando la información que recolectan de ellos, pueden optimizar y sugerir mejoras y cambios, atendiendo los gustos personales y reglas predefinidas. Estos sistemas se harán cada vez más comunes, sobre todo contarán con una alta integración con plataformas en la nube.

A través de este documento se podrá observar la investigación realizada, así como la temática requerida para abordar la problemática de construir herramientas dedicadas

a la generación de conocimientos sobre los procesos automatizados de los hogares.

En el capítulo 2 se presenta el marco teórico de esta investigación en las cuales se explica un conjunto de disciplinas, técnicas y tecnologías requeridas para lograr el objetivo de la construcción de sistemas para el análisis de grandes volúmenes de datos, de la minería de datos, de la programación, del Internet de las Cosas y finalmente de la domótica y el automatización del hogar.

A continuación, el capítulo 3 da un resumen de las más importantes metodologías aplicadas en las áreas de la ciencias de datos, de la minería de datos y el desarrollo de proyectos y la creación de productos organizadamente, acortando los tiempos de desarrollo y pasando por pruebas de control de calidad, con los cuales es posible llevar a cabo el objetivo de construir las herramientas necesarias y responder a la necesidades expuestas anteriormente.

Por último el capítulo 4 presenta una propuesta de trabajo especial de grado enmarcado en el problema de investigación, incluyendo la motivación, el alcance y finalmente los objetivos generales y específicos que se desean cumplir con dicha investigación.

Capítulo 2

Marco Teórico

2.1. Ciencia de Datos

2.1.1. Definición

La ciencia de datos, es un área multidisciplinaria que a través del uso de métodos y técnicas estadísticas, matemáticas y computacionales, busca recopilar, almacenar, procesar, analizar, visualizar y gestionar datos de manera eficiente [1]. Las personas que desarrollan las actividades relacionadas con la generación de conocimiento usando la ciencia de datos son llamados científicos de datos.

La ciencia de datos es en la actualidad un área que se encuentra en crecimiento continuo, debido a la necesidad de realizar operaciones sobre datos de manera exhaustiva, más eficiente y eficazmente para obtener resultados acertados velozmente en innumerable cantidad de procesos, muchos de los cuales no eran posibles de analizar por su volumen y complejidad en el pasado. Sin embargo esta disciplina no es nueva e involucra el conocimiento de áreas de las ciencias como la matemática, la estadística, la computación para poder llevar a cabo las tareas necesarias en la obtención de nuevos conocimientos.

2.1.1.1. Dato

Según el diccionario de la Real Academia Española [2], un dato es lo siguiente:
Dato. (Del latín datum 'lo que se da'). Inform. Información dispuesta de manera adecuada para su tratamiento por una computadora.

Así, para la computación un dato se refiere a uno o más símbolos que representan un valor, cuantitativo o cualitativo que puede ser recolectado, analizado y visualizado. Por sí solo, un dato no necesariamente tiene un significado. El tipo de dato más básico que puede procesar un computador es el dato cuyo valor es binario.

2.1.1.2. Información

Cuando se posee un conjunto de datos organizados y procesados, el resultado obtenido es lo que llamamos información. La información puede ser codificada, interpretada

y transmitida además de poseer una serie de características, que sirven para establecer cuál es la calidad de la misma y cuan relevante puede ser para el contexto en donde se está siendo utilizado. Estas características son las siguientes:

- **Significado (semántica)**: Se refiere al significado extraído a través de la interpretación de la información.
- **Disponibilidad**: Es la cualidad en la dificultad de obtener y acceder a la información.
- **Precisión**: La información en un contexto determinado requiere que sea lo suficientemente precisa para el uso al que se va a dar.
- **Confiabilidad**: La confiabilidad se ocupa de la veracidad de la información y de la objetividad con la que se presenta. Sólo se puede utilizar la información con confianza si se está seguro de su fiabilidad y objetividad.
- **Relevancia**: La información debe ser relevante para el propósito para el cual se requiere, por tanto debe ser adecuado. Lo que es relevante para un usuario o proceso puede no ser relevante para otro.
- **Completitud**: La información puede o no contener todos los datos requeridos por el usuario. Mientras estos expliquen de mejor manera el fenómeno del cual proviene, mayor sera la completitud de la información.
- **Nivel de detalle**: La información puede estar en un formato lo suficientemente detallado para permitir su examen y su uso o bien, para poder crear un nivel de abstracción sobre el mismo.
- **Presentación**: La presentación es la cualidad en la que se expresa la información y es un aspecto cuidado de cara al usuario que la requiere o utiliza. La información puede ser más fácilmente asimilada si es estéticamente agradable o su formato es el adecuado.
- **Vigencia**: La información debe ser presentada a tiempo para el propósito para el cual se requiere. La información recibida demasiado tarde puede ser irrelevante.
- **Valor**: La importancia relativa de la información para la toma de decisiones puede aumentar o disminuir su valor para una organización o individuo.
- **Costo**: La información solo puede estar disponible a través de la inversión monetaria o a través de una estructura de costos que pueden variar dependiendo de la situación. Si los costos son demasiado altos para obtener información, una organización puede decidir buscar información ligeramente menos amplia en otra fuente [3].

2.1.1.3. Conocimiento

De manera formal, el diccionario de la Real Academia Española [4] lo define de la siguiente manera:

1. *m. Acción y efecto de conocer.*
2. *m. Entendimiento, inteligencia, razón natural.*
3. *m. Noción, saber o noticia elemental de algo.*

El otras palabras, conocimiento es el entendimiento o la consolidación de información o hechos que podemos comprobar a través de la observación, la experiencia, la reflexión o la introspección.

2.1.2. Perfil de un Científico de los Datos

El científico de datos es aquel profesional que posee experiencia e instrucción requeridos para lograr la extracción de conocimiento a partir de los datos para poder responder a las inquietudes formuladas a través de una situación. Un científico de datos posee conocimientos de matemáticas, estadística, computación, entre otras disciplinas y suele trabajar con profesionales de otras áreas para poder así obtener resultados relevantes.

EL científico de datos debe poseer una serie de habilidades, para poder desarrollar su labor de manera exitosa, que se pueden resumir a través del diagrama de Venn de Drew Conway [5] mostrado en la figura 2.1, realizado para la conferencia Strata 2011 de O'Reilly [6] donde trata de simplificar y agrupar estas habilidades para ser considerado un científico de datos.

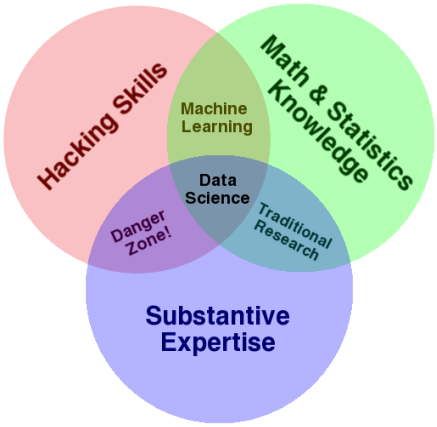

Figura 2.1: Diagrama de Venn de la Ciencia de Datos según Drew Conway

Un científico de datos debe poder aplicar una metodología para poder realizar un buen trabajo, para ello gasta bastante tiempo en el proceso de recolección, limpieza y depuración de los datos. Una vez los datos obtienen un formato adecuado, una parte crucial es el análisis exploratorio de ellos, el cual combina la visualización con el sentido de los datos, buscara los patrones, construirá modelos y algoritmos. Puede diseñar experimentos y formar parte de la toma de decisiones. Se comunicará con los miembros de su equipo, ingenieros y líderes en un lenguaje claro y gracias a las herramientas de visualización de los datos permite a sus colegas no están inmersos en los mismos entender sus implicaciones [7].

2.1.3. Grandes Volúmenes de Datos

2.1.3.1. Definición

Los grandes volúmenes de datos, o como comúnmente es llamado a través del anglicismo "Big data" es el concepto que engloba la capacidad de realizar el almacenamiento, el tratamiento y análisis de una cantidad enorme de datos. Se refiere a conjuntos de datos cuyo tamaño está más allá de la capacidad de las herramientas típicas de software de base de datos para capturarlas, almacenarlas, administrarlas y analizarlas [8].

Los grandes volúmenes de datos han abierto las puertas a un nuevo enfoque de entendimiento y toma de decisiones, la cual es utilizada para describir enormes cantidades de datos. Sin Embargo, Big Data no se refiere a una cantidad en específico, aunque en la actualidad es usualmente utilizado cuando se habla en términos de exabytes, zettabytes y de yottabytes de datos [9].

Generalmente se tienen una serie de atributos que se destacan al momento de enfrentar un problema de grandes volúmenes de datos y generalmente se pueden describir haciendo uso del modelo de las V's (generalización del modelo de las 3 V's de Doug Laney, Analista de Gatner [10] como en la figura 2.2). En la actualidad podemos hablar de un modelo de 7 V's [11]:

- **Volumen**: Se refiere a la vasta cantidad de datos generados cada segundo. Podemos tener estimaciones de cantidad de datos al pensar en todos los correos electrónicos, mensajes de Twitter, fotos, vídeos y datos de sensores que se producen cada segundo. No se habla de terabytes, sino de zettabytes o yottabytes de datos [12].

- **Velocidad**: Se refiere a la velocidad con que los nuevos datos son generados, procesados y a la velocidad de transmisión de los mismos en las redes [12]. Las tecnologías de Big Data ahora permiten analizar los datos mientras son generados sin siquiera haberla almacenado en bases de datos.

- **Variedad**: Se refiere a uno de los retos de los grandes volúmenes de datos. los datos puede estar en formato no estructurado y puede incluir múltiples tipos de datos desde XML a vídeo o SMS. Organizar los datos de una manera significativa no es una tarea sencilla, especialmente cuando los datos mismos cambian rápidamente [11].

- **Variabilidad**: No se debe confundir con la variedad. La variabilidad se refiere a los cambios validos de los datos, es decir, Si su significado es constante o si es cambiante, esto puede tener un gran impacto en la homogenización de los datos.

- **Veracidad**: Se trata de estar seguros que los datos son precisos, lo cual requiere procesos para evitar que datos incorrectos se acumulen en el sistema [11].

- **Visualización**: La visualización es algo critico en la actualidad. Usando gráficos para visualizar gran cantidad de datos complejos es más efectivo en transmitir significados que las hojas de cálculo y los reportes llenos de números y formulas [11].

- **Valor**: Se refiere a la habilidad de convertir los datos en algo valioso. Es importante que las organizaciones entiendan la necesidad de realizar cualquier intento de recopilar y aprovechar los grandes volúmenes de datos. Es fácil caer en la trampa de la emoción y embarcarse en grandes iniciativas de Big Data sin tener una clara comprensión del valor que traerá al negocio [12], lo cual se debe evitar.

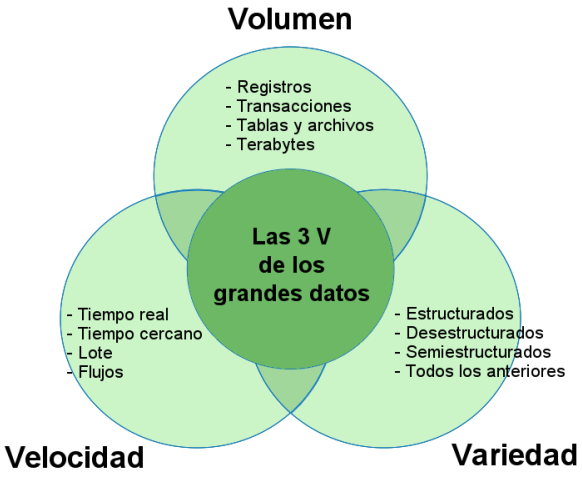

Figura 2.2: Diagrama de Venn del Modelo de las 3 V's de Doug Laney

2.1.3.2. Campos de Aplicación y Uso

Con cada vez más cantidad de datos disponibles en todos los ámbitos, Big Data puede entregar algún valor en casi cualquier área de los negocios o la sociedad [12]. El análisis de los grandes volúmenes de datos está en un auge de rápido crecimiento y adopción. Algunas de las áreas más importantes donde podemos observar este fenómeno son las siguientes:

- **Educación**: Las nuevas tecnologías permiten a las escuelas, instituciones y universidades analizar absolutamente todo lo que sucede en el ámbito del aprendizaje y los procesos académicos, desde el comportamiento del estudiante, resultados de pruebas, el desarrollo profesional de los involucrados, así como las necesidades educativas basadas en sociedades cambiantes. El objetivo general de los grandes volúmenes de datos dentro del sistema educativo debe ser mejorar los resultados de los estudiantes. Mejores estudiantes es mejoría en la sociedad, organizaciones en general, así como las instituciones académicas. Toda la data almacenada puede ayudar a crear un programa personalizado para cada individuo. Los grandes volúmenes de datos permiten la personalización en instituciones de educación superior, incluso si estos tienen una gran cantidad de estudiantes. Esto creará un aprendizaje mixto, una combinación de enseñanza online y offline. Esto dará a los estudiantes la oportunidad de desarrollar su propio programa personalizado de aprendizaje. También el desarrollo y participación en los MOOCs (Massive Online Open Courses) dará a las universidades oportunidad la de buscar la participación de un grupo cada vez mayor de estudiantes, con experiencias de enseñanza en

tiempo real. Finalmente todos estos análisis mejorarán los resultados de los estudiantes y de esta manera, bajar los índices de abandono escolar en todos los niveles de la enseñanza. Haciendo uso de analítica predictiva con todos los datos recolectados se puede dar a los institutos de educación conocimientos del rendimiento escolar. Estas predicciones pueden ser usadas para cambiar un programa si se predicen malos resultados en un programa en particular, antes de que este comience. Universidades y colegios serán más eficientes en el desarrollo de programas que aumentará el desempeño, minimizando de este modo el ensayo y error [13].

- **Banca**: Con una gran cantidad de información de diversas fuentes y de diversos flujos, la banca se enfrenta a la búsqueda de nuevas e innovadoras manera de administrar el gran volumen de datos. Mientras es importante entender a los clientes y aumentar su satisfacción, es igualmente importante minimizar riesgos y fraudes manteniendo el cumplimento de las normativas. Los grandes volúmenes de datos proveen de grandes ideas, pero también requiere que las instituciones financieras estén un paso adelante en el juego usando analítica avanzada. [14]. Otro punto que merece mención es que la ciencia de datos ha permitido elevar los estándares de seguridad en las operaciones, pudiendo inclusive hacer la detección de fraudes en transacciones de manera automatizada.

- **Gobierno**: Cuando los entes gubernamentales son capaces de aprovechar y aplicar análisis a sus grandes volúmenes de datos, ellos ganan terreno a la hora de administrar utilidades, mejor funcionamiento de los entes, tratar con la congestión de tráfico, la prevención del crimen, etc. Estos son algunos de los ejemplos en que la sociedad se beneficiará en el desarrollo de sistemas automatizados que cuiden y administren los recursos de los ciudadanos. Pero mientras todas estas ventajas vienen con el uso de Big Data, los gobiernos también deberán abordar las cuestiones de transparencia y privacidad [14].

- **Cuidado de la Salud**: Historias de pacientes, planificación de los tratamientos, información de prescripciones. Cuando se trata del cuidado de la salud, todo necesita hacerse rápida y acertadamente y en algunos casos, con la suficiente transparencia para satisfacer las rigurosas regulaciones de la industria. Cuando los grandes volúmenes de datos son administrados eficientemente, los proveedores del cuidado de la salud pueden descubrir formas de mejorar el cuidado de los pacientes y a los profesionales del área pueden prescribir tratamientos personalizados.

- **Industria**: Hay numerosas oportunidades de aplicación del Big Data ahora mismo en contextos de la industria. Especialmente en lo relacionado con la optimización de procesos, análisis predictivo para adelantarse a la necesidad del mantenimiento de equipamientos y máquinas, así como para la detección de nuevas oportunidades de eficiencia (y por lo tanto, nuevas fuentes de negocio para las empresas) [15]. Armados con el conocimiento que los grandes volúmenes de datos pueden proporcionar, los fabricantes pueden mejorar la calidad y la producción, minimizando al mismo tiempo los desechos, procesos que son clave en el mercado altamente competitivo de hoy en día. Cada vez más fabricantes están trabajando en una cultura basada en el análisis, lo que significa que pueden resolver problemas más rápido y tomar decisiones empresariales ágiles [14].

- **Logística**: Los grandes volúmenes de datos permite recopilar y procesar la información que se genera durante las etapas de las cadenas de suministro. Con esta información se pueden detectar las tendencias, el comportamiento de los clientes o los errores de operaciones, para poder introducir soluciones a los procesos e incluso detectar nuevos negocios de cara al futuro. Mejora la eficacia de los procesos. Al medir los procesos logísticos, se obtiene un mapa detallado y objetivo de la situación y la calidad de los mismos. Con la introducción de los grandes volúmenes de datos se puede mejorar el control de activos de la empresa: la flota de vehículos, los productos almacenados, las cargas, etc [16]. Si además la empresa integra sus grandes volúmenes de datos con sus sistemas de Customer Relationship Management (CRM), se pueden conocer las necesidades e intereses de sus clientes, además de optimizar los procesos de distribución y permite realizar un seguimiento exhaustivo del estado y de la situación de todos los productos para, entre otras cosas, poder detectar posibles incidentes, fijar precios ajustados a los costos logísticos. Por ultimo permite la optimización de rutas a través del uso de datos en tiempo real de clima y tráfico para entregar los pedidos a los clientes.

- **Deportes**: Para el especialista en rendimiento deportivo, ha habido una convergencia de tecnologías que ha permitido la recolección y análisis de métricas deportivas como nunca antes. La tecnología vestible permite la recolección y transmisión de datos en tiempo real. Fuera en la pista o en el velódromo en una bicicleta y en el taekwondo, gran poder analítico de datos permite que los entrenadores y científicos deportivos realmente puedan entender y mejorar el rendimiento de los atletas [17].

2.1.4. Tipos de Datos

Los datos pueden venir en múltiples formas, incluyendo datos estructurados y no estructurados como datos financieros, archivos de texto, archivos multimedia y asignaciones genéticas. Contrariamente a una gran parte del análisis tradicional de datos realizado por las organizaciones, la mayoría de los grandes volúmenes de datos no está estructurado o semi-estructurado lo que requiere diferentes técnicas y herramientas para procesar y analizar. Los entornos de computación distribuida y las arquitecturas de procesamiento masivo paralelo (MPP) [18].

2.1.4.1. Datos Estructurados

Los datos estructurados son aquellos que contienen un tipo, un formato y una estructura bien definidos; estos pueden ser datos transaccionales, cubos de datos de procesamiento analítico en línea, sistemas manejadores de bases de datos tradicionales, archivos CSV e incluso hojas de cálculo. Este tipo de datos representa la minoría de la información generada por sistemas informáticos.

2.1.4.2. Datos Semi-Estructurados

Los datos semi-estructurados son aquellos datos cuya estructura no encaja en los modelos aceptados por los sistemas de bases de datos relacionales pero que cuentan

con un patrón discernible, es decir, que se autodescriben, tales como archivos de datos XML. Otro ejemplo de este tipo de datos son los JavaScript Object Notation o JSON, que se ha vuelto bastante popular y es utilizado en multitud de aplicaciones dada su simplicidad, consistente en tuplas clave-valor.

2.1.4.3. Datos No Estructurados

Los datos no estructurados son aquellos que no tienen una forma inherente, no están organizados y pueden incluir documentos de texto, PDF, imágenes y vídeo. Representan cerca del 80 % de los datos que son generados.

2.1.5. Bases de Datos

Una base de datos es una colección de datos lógicamente organizados y relacionados. En la actualidad son la base primordial de la computación. Estos conjuntos de datos sirven para modelar los aspectos más relevantes de la realidad. Para poder manipular y acceder a los datos que posee una base de datos, se hace a través del uso de un Sistema Manejador de Base de Datos, es decir un conjunto de programas de propósito general que proporcionan las funcionalidades para facilitar la gestión de la información contenida en una base de datos.

La gestión de los datos implica tanto la definición de estructuras para almacenar la información como la provisión de mecanismos para la manipulación de la información. Además, los sistemas manejadores de bases de datos deben proporcionar la fiabilidad de la información almacenada, a pesar de las caídas del sistema o los intentos de acceso sin autorización. Si los datos van a ser compartidos entre diversos usuarios, el sistema debe evitar posibles resultados anómalos [19].

2.1.5.1. Bases de Datos Relacionales

Una base de datos relacional es una base de datos que se trata como un conjunto de tablas y se manipula de acuerdo con el modelo de datos relacional [20]. Cada fila de una tabla representa una recopilación de valores relacionados de un objeto o entidad. Las tablas se utilizan para guardar información sobre objetos que se van a representar en la base de datos. Cada columna de una tabla guarda un determinado tipo de datos y un campo almacena el valor real de un atributo. La fila en cada tabla se podría identificar con una clave única identificadora (clave primaria) y la fila entre varias tablas se puede relacionar utilizando claves externas (clave foránea). Se puede obtener acceso a estos datos de muchas formas distintas sin reorganizar las propias tablas de la base de datos [21].

Los aspectos más importantes de las bases de datos relacionales son los siguientes:

- SQL: el lenguaje de consulta estructurada o SQL por sus siglas en inglés, es la interfaz principal para comunicarse con base de datos relacionales. SQL se convirtió en un estándar del American National Standards Institute (ANSI) en 1986. Todos los motores de base de datos relacionales populares soportan SQL de la ANSI estándar. Algunos de estos motores también poseen una extensión al

SQL de la ANSI para soportar funcionalidades específicas de este motor. SQL se utiliza para agregar, actualizar o eliminar filas de datos, recuperar subconjuntos de datos para aplicaciones de análisis y procesamiento de transacciones y para administrar todos los aspectos de la base de datos.

- Integridad de Datos: La integridad de los datos es la totalidad, precisión y coherencia general de los datos. Las bases de datos relacionales utilizan un conjunto de restricciones para aplicar la integridad en la base de datos. Esto incluye claves principales, claves externas, restricción "Not Null", restricción "Unique", restricción "Default" y restricción "Check". Estas restricciones de integridad ayudan a aplicar las reglas de negocio en datos de las tablas para garantizar la precisión y fiabilidad de los datos. Ademas de las anteriores, la mayoría de las bases de datos relacionales también permiten la integración de código personalizado en disparadores (Triggers) que se ejecutan en función de una acción en la base de datos.

- Transacciones: Una transacción de base de datos es una o más sentencias SQL que se ejecutan como una secuencia lógica de trabajo. Las transacciones proporcionan una proposición "todo o nada". La transacción completa se debe completar como una sola unidad y se debe escribir en la base de datos, de lo contrario, ninguno de los componentes individuales de la transacción debería ser aceptado. En la terminología de bases de datos relacionales una transacción genera un COMMIT en caso de completarse correctamente o con un ROLLBACK si falla. Cada transacción se trata de forma coherente y fiable independientemente de otras transacciones.

- Conformidad con ACID: Todas las transacciones de base de datos deben ser conformes a ACID (atómicas, coherentes, aisladas y duraderas) para garantizar la integridad de los datos. La atomicidad requiere que la transacción se ejecute correctamente; si una parte de la transacción falla, la transacción completa queda invalidada. La coherencia exige que los datos escritos en la base de datos como parte de la transacción debe cumplir con todas las reglas definidas, así como las restricciones, cascadas y disparadores. El aislamiento es crítico para lograr el control de concurrencia y se asegura que cada transacción sea independiente por si misma. La durabilidad requiere que todos los cambios realizados en la base de datos sean permanentes una vez que la transacción se complete correctamente.

En la tabla 2.1 podemos ver las principales ventajas y desventajas que representa el uso de las base de datos relacionales

2.1.5.2. Bases de Datos No Relacionales

Las bases de datos no relacionales son todas aquellas cuyo enfoque no se rige por el uso de un modelado relacional. Estos no requiere de una estructura rígida para almacenar los datos. En tal sentido estas bases de datos no usan todas las operaciones que tendrían las bases de datos relacionales ni tampoco garantizan el cumplir con los fundamentos de ACID. Estas bases de datos son también llamadas bases de datos NoSQL (Not only SQL) ya que muchas de ellas no usan (o solo parcialmente) SQL para poder operar sobre ellas.

Ventajas	Desventajas
▪ Provee herramientas que garantizan evitar la duplicidad de registros. ▪ Garantiza la integridad referencial, es decir, al eliminar un registro, elimina todos los registros relacionados dependientes. ▪ Favorece la normalización por ser comprensible y fácil de aplicar. ▪ Al tener largo tiempo en el mercado siendo usadas, poseen mayor documentación y soporte que otros tipos de bases de datos.	▪ Presentan deficiencias al tratar con datos gráficos, multimedias, información geográficas, documentos y tipos de datos semi estructurados o no estructurados en general. ▪ La escalabilidad es difícil de implementar dado su esquema es rígido ▪ Al tener que cumplir con los fundamentos de ACID, el rendimiento es impactado de manera negativa.

Tabla 2.1: Ventajas y Desventajas de las Bases de Datos Relacionales

La ventaja de este tipo de base de datos es que puede adaptarse de mejor manera al las reglas de negocio que se requieren modelar, permitiendo almacenar de una manera más fácil y eficientemente datos semi-estructurados o no estructurados.

Una característica importante de las bases de datos no relacionales es que su implementación permite en muchos casos que estas puedan escalar horizontalmente de una manera relativamente sencilla, contrariamente a lo que puede suceder en las bases de datos relacionales, por su rigidez estructural. Las bases de datos no relacionales no son homogéneas en su funcionamiento y son clasificadas bajo el paradigma que estos apliquen al almacenamiento de los datos y sus características son definidas a través del teorema de Brewer, también llamado teorema CAP.

El teorema de Brewer establece que bajo un sistema distribuido que busque proveer consistencia (después de una operación de actualización de algún escritor todos los lectores ven esa actualización de alguna fuente de datos compartida), disponibilidad (cada petición recibida por un nodo activo debe dar por lugar una respuesta. Aunque se produzcan fallos en la red cada solicitud debe terminar) y tolerancia al particionado (se entiende como la capacidad del sistema para continuar la operación en presencia de particiones de red) solo puede obtener dos de estas garantías durante el mismo estado de operación.

Como podemos observar en la figura 2.3 en un determinado momento podemos tener los siguientes estados:

- AP: el sistema siempre responderá aunque se pierda la comunicación entre nodos. Los datos procesados pueden no ser consistentes.
- CA: el sistema siempre responderá y los datos procesados serán consistentes. No

se considera la pérdida de comunicación entre nodos.

- CP: el sistema ejecutara las operaciones de forma consistente, aunque se pierda la comunicación entre nodos, pero no se asegura que el sistema responda [22].

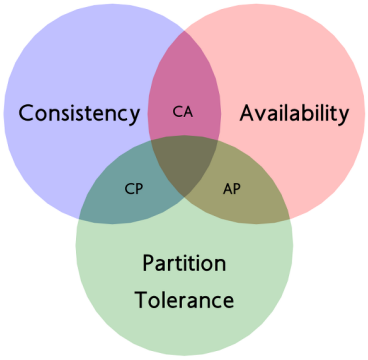

Figura 2.3: Teorema de Brewer o Teorema CAP

En la tabla 2.2 podemos ver las principales ventajas y desventajas que representa el uso de las base de datos relacionales.

Ventajas	Desventajas
▪ Las bases de datos no relacionales tienen la capacidad de escalar de manera horizontal de una manera sencilla y transparente. ▪ El costo de implementar una base de datos no relacional en un conjunto de clusters es menor, debido a que estos no requieren de un poder de cómputo grande. ▪ Su desempeño ante grandes volúmenes de datos es mayor que en las bases de datos relacionales. ▪ Las bases de datos no relacionales son más flexibles en cuanto a los tipos de datos que pueden almacenar.	▪ Al ser de adopción relativamente reciente, las tecnologías de bases de datos no relacionales son menos maduras que se contraparte relacionales. ▪ También este tipo de bases de datos se ven afectadas por la cantidad de personas capacitadas para administrar estos paradigmas. ▪ Como tecnologías de reciente desarrollo y auge están mucho menos documentadas que las bases de datos relacionales.

Tabla 2.2: Ventajas y Desventajas de las Bases de Datos no Relacionales

Existen diversos tipos de bases de datos no relacionales, cada una responde a un tipo de paradigma de almacenamiento distinto. A continuación podemos observar cada una de ellas:

- **Bases de datos clave-valor**: Son bases de datos NoSQL optimizadas para cargas de trabajo de aplicaciones que realizan muchas tareas de lectura (como redes sociales, juegos, uso compartido de archivos multimedia, entre otras) o la ejecución de trabajos de manera intensiva (como motores de recomendación). El almacenamiento en memoria caché mejora el desempeño de las aplicaciones almacenando los datos críticos en memoria para un acceso de baja latencia. A través de la búsqueda por la clave es posible identificar el dato requerido (como en la figura 2.4), lo que hace que el acceso sea veloz. Ejemplo de este tipo de bases de datos: Redis, Riak.

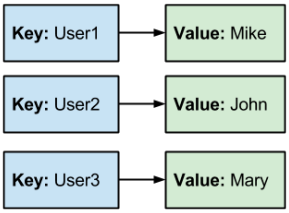

Figura 2.4: Almacenamiento en una Base de Datos Clave-Valor

- **Bases de datos documentales**: Están diseñadas para almacenar datos semi-estructurados como documentos (véase figura 2.5), normalmente en formatos XML o JSON. A diferencia de las bases de datos relacionales tradicionales, el esquema de cada documento, puede variar lo que ofrece más flexibilidad al organizar y almacenar datos de aplicaciones y permite reducir el almacenamiento necesario para los valores opcionales. Ejemplo de este tipo de bases de datos: MongoDB, CouchDB.

```
{id: 1,
 name: Joe,
 url: '...',
 stream:
 [{
   user: 2,
   title: 'today',
   body: 'go fly a kite',
   likes: [
    {user: 3},
    {user: 1},
   ],
 }]
}
```

Figura 2.5: Ejemplo de un Documento JSON

- **Bases de datos orientadas a columnas**: Las bases de datos orientadas a columnas están optimizadas para leer y escribir columnas de datos en lugar de filas. El almacenamiento basado en columnas para tablas de bases de datos es un factor importante en el desempeño de las consultas analíticas, ya que reduce notablemente los requisitos globales de entrada/salida del disco, así como la cantidad de

datos que hay que cargar desde el mismo. Las bases de datos orientadas a columnas forman estructuras que van agrupando filas que a su vez forman familias de columnas. Ejemplo de este tipo de bases de datos: Cassandra, HBase.

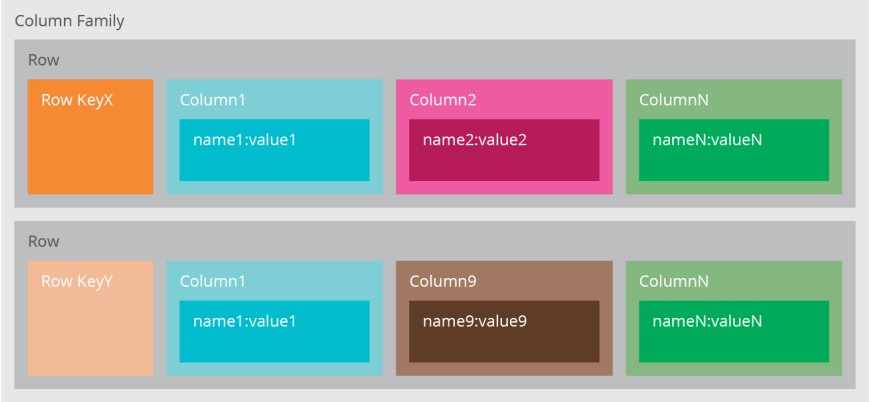

Figura 2.6: Almacenamiento en las Bases de Datos Orientadas a Columnas

- **Bases de datos orientadas a grafos**: Las bases de datos orientadas a grafos almacenan vértices y enlaces dirigidos llamados aristas, las cuales son estructuras donde se puede almacenar datos. Los grafos pueden basarse en base de datos relacionales y no relacionales. Los vértices y aristas pueden tener propiedades asociadas como podemos ver en la figura 2.7. Ejemplo de este tipo de bases de datos: Neo4j, HyperGraphDB.

Figura 2.7: Ejemplo de Base de Datos Orientada a Grafos

- **Bases de datos multi-modelos**: Son bases de datos que pueden dar soporte a múltiples maneras de almacenar los datos, sea como grafos, como documentos,

clave-valor y como objetos. Ejemplo de este tipo de bases de datos: OrientDB, ArangoDB.

2.2. Minería de Datos

2.2.1. Definición

La minería de datos es la práctica de buscar automáticamente en grandes almacenes y bases de datos patrones y tendencias que van más allá del simple análisis [23].

La minería de datos, también referenciada como Descubrimiento del Conocimiento en Bases de Datos (Knowledge Discovery in Databases o KDD), ha sido definida como el proceso de extracción no trivial de información implícita, previamente desconocida y potencialmente útil dentro de un conjunto de datos. La minería de datos incluye áreas del conocimiento tales como la estadística, inteligencia artificial y bases de datos. Se estima que del análisis de esos datos puedan surgir ventajas competitivas o novedosas soluciones a antiguos problemas [24].

Durante la última década, los avances en el poder y velocidad de procesamiento, nos han permitido pasar de prácticas manuales, tediosas y que consumen mucho tiempo a un análisis de datos rápido, fácil y automatizado. Cuanto más complejos sean los conjuntos de datos recolectados, más potencial habrá para descubrir ideas relevantes. Vendedores, la banca, la industria, los proveedores de telecomunicaciones y los aseguradores, entre otros, utilizan la minería de datos para conseguir las relaciones entre todo, desde precios, promociones y la demografía hasta como la economía, los riegos, la competencia y las redes sociales están afectando sus modelos de negocios, ingresos, operaciones y relaciones con los clientes [25].

Las propiedades claves de la minería de datos son:

- Descubrimiento automatizado de patrones.
- Predicción de resultados probables.
- Creación de información accionable.
- Enfocado en amplios conjuntos de datos y bases de datos.

La minería de datos es utilizada en incontable cantidad de áreas, pero entre algunas de sus aplicaciones tenemos:

- Banca y finanzas: Obtención de perfiles de clientes para oferta de productos, asignación de créditos o detectando transacciones fraudulentas.
- Inteligencia de Negocios: Determinando gustos y preferencias de usuarios y clientes para venta y marketing personalizado, predicción de ventas, toma de decisiones en base a patrones de consumo en bienes y servicios.
- Medicina: Búsqueda de relaciones entre enfermedades y condiciones, análisis de historiales médicos, recomendación de tratamientos.
- Educación: Predicción de rendimiento estudiantil y efectividad de programas educativos. Construcción de perfiles de estudiantes y de planes de estudio personalizado.

- Entretenimiento: Sistemas de recomendación de películas, música y videojuegos en base a gustos.
- Redes sociales: Análisis de las interacciones de usuarios, determinación de tendencias.
- Planificación de proyectos: medición y estimación de tiempos, costos y duración.
- Asistencia personalizada: fuentes de información y noticias en base a gustos e intereses, asistentes de voz.

2.2.2. Proceso de Extracción de Conocimiento en Bases de Datos

El proceso de extracción de conocimiento en base de datos es la obtención de conocimiento e información potencialmente útil [26] dentro de un conjunto de datos. Su carácter es en muchos casos iterativo, debido a que el conocimiento generado puede servir para alimentar otro proceso KDD. En general el proceso de extracción de conocimiento se pueden ver como una serie de pasos ordenados en etapas como en la figura 2.8.

Figura 2.8: Proceso de Extracción de Conocimiento

En detalle los pasos del proceso de extracción de conocimiento son los siguientes:

1. Definición del problema: En este paso, se define(n) la(s) pregunta(s) que se desea(n) responder, es decir, se formula los objetivos y requerimientos del proceso, con que herramientas se cuenta y que datos se posee que pueden resultar útiles para el proceso. También se requiere el conocer el estado actual del problema para proveer un contexto a la información y para adquirir conocimiento del dominio.
2. Selección de los datos: Se determinan las fuentes de datos y el tipo de información a utilizar [26]. Es la etapa donde se unifica los datos a utilizar.
3. Pre-procesamiento: Al obtenerse los datos de diferentes fuentes y ser unificados, es necesario realizar una limpieza de los mismos, homogeneizando formatos de datos, quitando valores defectuosos y establecer valores por defecto cuando sea requerido utilizando técnicas estadísticas y uniendo aquellos datos cuyo significado

semántico sea el mismo (como por ejemplo cédula de identidad con número de identificación). Se trata de determinar cuál de las variables obtenidas de las fuentes de datos son necesarias o útiles al momento de realizar el análisis. Este paso tiende a ser el que más tiempo lleva pues se busca que el conjunto de datos a ser minado este en las mejores condiciones para ser procesado y con ello obtener mejores resultados [23] [26].

4. Transformación: Durante este paso se busca generar la aplicación de operaciones de agregación o normalización de datos, creación de nuevas variables a partir de otras existentes o consolidando de manera necesaria la información para producir una vista minable, que no es mas que una estructura de datos para el consumo de la generación de modelos [26].

5. Minería de datos: Durante esta fase, se selecciona y aplican diferentes técnicas, algoritmos y ajustes de parámetros de tal manera que se consiga el modelo que tenga el mejor desempeño en términos de las necesidades y requerimientos deseados para la obtención de patrones, que puedan explicar las relaciones existentes entre los datos [26] [23].

6. Interpretación y evaluación: Se identifican los patrones obtenidos y se determina si se pueden interpretar, si proveen información útil o si son interesantes basándose en métricas como su precisión [26].

7. Despliegue: Una vez validado el modelo, puede integrarse a procesos para la toma de decisiones, ser utilizado en otras aplicaciones o ser aplicado a otros conjuntos de datos [23].

Durante cualquiera de los pasos mencionados anteriormente se puede retomar alguno previo que ayude a mejorar el proceso en el paso siguiente.

2.2.3. Modelos Resultantes de un Proceso K.D.D.

Es de notar que parte importante del proceso de extracción de conocimiento de bases de datos es la obtención de patrones haciendo uso de modelos que expliquen la relación entre los datos que poseen y las variables que intervienen en el mismo. Dadas las características del problema, del conjunto de datos (variables numéricas, nominales o binarios) y del objetivo de la minería de datos podemos aplicar una serie de técnicas y algoritmos que se clasifican en tareas predictivas, que generan modelos predictivos o podemos aplicar tareas descriptivas que generan modelos descriptivos.

2.2.3.1. Modelos Predictivos

Los modelos predictivos buscan predecir el valor de un atributo en particular (variable dependiente u objetivo) a partir los valores de otros atributos (variables independientes o predictivas). Las técnicas que generan modelos predictivos son:

Clasificación: Esta técnica tiene como objetivo predecir una o más variables basada en otros atributos del conjunto de datos [27]. Cada registro del conjunto de datos

contiene los valores de un conjunto de atributos, uno de los cuales es una clase (cada instancia pertenece a una clase), la clasificación encuentra un modelo para el atributo clase en como una función de los valores de los otros atributos. Los algoritmos más utilizados para clasificar son:

- Arboles de de decisión como C4.5 o ID3.
- Clasificadores Bayesianos como el Clasificador Bayesiano Ingenuo o aquellos basados en Redes Bayesianas.
- Clasificadores basada en reglas como clasificadores de reglas "si-entonces", aquellos extraídos por arboles de decisiones o por inducción de reglas usando un algoritmo de cobertura secuencial.
- Redes neuronales como redes neuronales con retropropagación o redes neuronales recursivas.
- Máquinas de Soporte Vectorial.

Regresión: La regresión es una técnica matemática y estadística que relaciona la variable dependiente con k variables explícitas o cualquier transformación de estas que generen un hiperplano que minimiza la distancia de todos los elementos del conjunto. Esta técnica es especialmente efectiva cuando se tiene un conjunto de datos mayoritariamente numérico. El algoritmo clásico utilizado en esta técnica es la regresión lineal.

2.2.3.2. Modelos Descriptivos

Los modelos descriptivos son aquellos que consiguen patrones (correlaciones, grupos, anomalías asociaciones) que sumarizan, explican o describen los datos.

Agrupación (Clustering): Se busca que a partir de un conjunto de elementos, estos se agrupen en clases bajo una determinada característica. Se le llama también clustering debido a que los elementos tienden a ser similares dentro de su partición. Los algoritmos más conocidos para obtener modelos asociativos son:

- K-medias.
- K-mediodes.
- Método del mallado.

Análisis de Asociación: Esta técnica busca establecer asociaciones entre elementos utilizando para ello el análisis de variables cuantitativas como la frecuencia, la varianza, índices de correlación, entre otras. Un ejemplo de algoritmo de análisis de asociación es el algoritmo Apriori.

Detección de Anomalías: Este busca encontrar elementos cuyas características hacen que sea notoriamente distintos al resto dentro de un conjunto, es decir, que no tienen el mismo comportamiento que describen los demás. Los valores bajo esta situación pueden estar representando errores de medición, sin embargo también pueden ser parte de alguna característica de algún hecho y es posible explicarlo

dado el contexto que rodea al fenómeno. Es así como esta técnica es especialmente eficaz cuando se trata de establecer hechos que requieren ser revisados minuciosamente como alertas ante una situación irregular [28]. Entre las maneras de detectar anomalías se destacan las técnicas:

- Basadas en distribuciones estadísticas.
- Basadas en la distancia.
- Basadas en la densidad local.
- Basadas en la desviación del conjunto.

2.2.4. Extensiones Existentes a la Minería de Datos

La minería de datos se ha diversificado de forma que ha especializado soluciones y cubrir campos cuyas dificultades requieren técnicas mejoradas para poder resolver los problemas que se presentan.

2.2.4.1. Minería de Texto

La minería de texto es el proceso de análisis de las colecciones de materiales de texto con el fin de captar los conceptos y temas clave, descubrir relaciones y tendencias ocultas sin necesidad de conocer las palabras precisas o términos que los autores han utilizado para expresar esos conceptos. A pesar de que son bastante diferentes, minería de texto a veces se confunde con la recuperación de información. La minería de texto busca los patrones inmersos en el texto, lo cual implica la extracción y gestión de significados, terminología y las relaciones contenidas dentro de la información como procesos cruciales y críticos.

El principal problema con el manejo de todos estos datos en formato de texto es que no hay reglas estándares para la escritura de texto para que un computador pueda entenderlo. El lenguaje, y por lo tanto el significado, varía para cada documento y cada fragmento de texto. La única manera de recuperar con precisión y organizar dichos datos no estructurados es analizar el lenguaje y así descubrir su significado. Existen varios enfoques automatizados diferentes a la extracción de los conceptos de información no estructurada. Estos enfoques se pueden dividir en dos tipos, lingüísticos y no lingüísticos [29]. El área de la inteligencia artificial que apoya a la extracción de conocimiento a través de la minería de texto es la del procesamiento del lenguaje natural (PLN).

2.2.4.2. Minería Web

La minería web es la aplicación de técnicas de minería de datos para obtener patrones de la Web. Dependiendo del tipo de dato web a procesar, la minería web se divide en tres grandes categorías: contenido, estructura y uso. El análisis de estos datos permite a las instituciones significativas mejoras en la estructura y contenido de los sitios web corporativos, así como la aplicación de complejos sistemas informáticos destinados a personalizar la experiencia del usuario en el sitio que visita [30].

La minería web basada en el contenido trata de descubrir la información existente de textos o el contenido multimedia que se encuentren la web, siendo principalmente usado para la recuperación de información. La minería web enfocada en la estructura busca analizar el nodo y estructura de conexión de un sitio web, con el se puede extraer patrones usando los hipervínculos o bien se puede crear un árbol para analizar las etiquetas de los HTML o XML. Finalmente la minería web del uso trata de la extracción de los registros del servidor, es decir, el historial y comportamiento de los usuarios.

2.3. Lenguajes de Programación

Los lenguajes de programación son lenguajes formales creados para realizar procesos que pueden ser llevados a cabo por autómatas. Con ellos podemos crear programas que controlan el comportamiento físico o lógico de la maquinas, expresados a maneras de algoritmos. Los lenguajes de programación están formados por un conjunto de símbolos y reglas sintácticas y semánticas que dan forma a su estructura y al significado de los elementos y expresiones que se produzcan a partir de estas.

Existen estrategias que permiten ejecutar en una computadora un programa realizado en un lenguaje de programación. Los procesadores del lenguaje son los programas que permiten el tratamiento de la información en forma de los símbolos que se encuentren en el código fuente.

Generalmente un lenguaje de programación responde a uno o más paradigmas, es decir, un enfoque particular o filosofía para el diseño de soluciones. Los paradigmas difieren unos de otros, en los conceptos y a la forma de abstraer los elementos involucrados en el problema, así como en los pasos que integran la solución de las mismas. Los paradigmas de programación más comunes son:

- **Programación imperativa**: Es el paradigma de programación más antiguo y ampliamente utilizado. Se basa en dar instrucciones ordenadas a la maquina en forma de un algoritmo. Ejemplos de lenguajes que usan este paradigma son C, Pascal, BASIC.
- **Programación declarativa**: Se basa en describir el problema declarando propiedades y reglas que deben cumplirse, en vez de instrucciones a ejecutarse. Varios paradigmas se basan en este para sus bases operacionales.
- **Programación orientada a objetos**: Basado en el paradigma imperativo, agrega elementos denominados objetos que forman estructuras que incluyen variables u otros objetos, y métodos que son las serie de acciones asociadas al objeto, cuyo objetivo final es el poder representar una porción de la realidad que se desea modelar. Los lenguajes orientados a objetos más conocidos son Java, C#, Simula, Objective-C entre otros.
- **Programación funcional**: Está basada en la definición de predicados y es de corte matemático, en el marco del concepto de función. Los lenguajes más representativos que usan este paradigma son Haskell y Lisp.
- **Programación lógica**: Este paradigma se basa en la definición de relaciones lógicas. El lenguaje por excelencia de esta categoría es Prolog.
- **Programación multiparadigma**: Estos lenguajes permiten el uso de más de un tipo de paradigma de manera nativa. Python, Javascript, PHP, Visual Basic, son lenguajes que pertenecen a este paradigma.
- **Lenguajes específicos de dominio**: Aquellos lenguajes que son diseñados con el fin de poder resolver un problema en específico, como por ejemplo: SQL, R o Mathematica.

2.3.1. Tipos de Lenguajes de Programación

Existen dos tipos de implementación de lenguajes de programación: los lenguajes que usan la compilación (lenguajes compilados) y los aquellos que usan un intérprete (lenguajes interpretados).

La compilación es el proceso por el cual un lenguaje de programación es traducido a otro lenguaje, de mas bajo nivel (generalmente en código objeto) y luego es enlazado con aquellos rutinas externas y traducido a lenguaje de maquina y generando un ejecutable.

La interpretación es el proceso por el cual un lenguaje de programación va otorgando significado a las instrucciones de un lenguaje formal. El intérprete elimina la necesidad de realizar una compilación después de cada modificación del programa cuando se quiere agregar funciones o corregir errores. Sin embargo un programa compilado al no tener que hacer una interpretación continua del código, se ejecutará más rápidamente.

2.3.2. Python

Python es un lenguaje de programación de alto nivel, multiplaforma, débilmente tipado, de propósito general, multiparadigma e interpretado [31], creado por Guido Van Rossum a finales de los años 80, siendo en el año 1991 cuando es oficialmente lanzado. Su filosofía se basa en el poder crear código que sea muy legible, con una sintaxis simple, utilizando indentaciones para delimitar bloques de código, más corto y potente que en otros lenguajes de programación. [32]

Se dice que Python es un lenguaje que viene con "pilas puestas" [33], es decir que posee un conjunto de funciones amplio para afrontar cualquier tipo de situación dentro de las librería estándar del lenguaje. También es fácilmente extensible pudiéndose agregar nuevos elementos al lenguaje de una manera sencilla. Otra característica de Python es su alta modularidad, que inclusive algunos de ellos están escritos en C o C++.

Los paradigmas de programación que soporta Python son la programación imperativa, programación orientada a objetos y aspectos de la programación funcional. Este lenguaje destaca en su facilidad para aprender y alta portabilidad y es usado en una gran cantidad de aplicaciones, que van desde el área web, pasando por la ciencia de datos, la inteligencia artificial y la programación de dispositivos.

2.3.2.1. Principales Frameworks y Librerías

Aparte de la extensa librería que posee Python, la comunidad ha elaborado sobre este lenguaje un completo ecosistema de frameworks y librerías para diversos usos. Para el desarrollo web se cuenta con:

- Django: Es un framework web de alto nivel para Python que fomenta un desarrollo rápido y un diseño limpio y pragmático [34]. Es de código abierto y gratuito bajo la licencia BSD.

- Flask: Es un microframework, basado en Werkzeug y Jinja2 [35]. Es distribuido bajo licencia BSD.
- Pyramid: Es un framework que se basa en ofrecer solo las herramientas necesarias para realizar un desarrollo web simple pero lo suficiente para permitir escalar las soluciones [36].

Por otro lado Python cuenta con un buen número de librerías especializadas para el ámbito científico y análisis estadístico que lo convierte en uno de los lenguajes preferidos a la hora de afrontar la solución de problemas de Big Data. Los más utilizados son:

- Pandas: pandas es una librería de código abierto distribuida bajo licencia BSD, que provee estructuras de datos de alto rendimiento, fáciles de usar y herramienta de análisis [37].
- Scikit-Learn: Esta librería provee herramientas simples y eficientes para la minería de datos, el análisis de datos y el aprendizaje automatizado. Es código abierto y se distribuye bajo licencia BSD [38].
- Numpy: es una librería que provee algunas estructuras de datos especializadas e integración con funciones de álgebra lineal, transformadas de Fourier y generadores de números aleatorios [39].
- Jupyter Notebook: es un conjunto de librerías que permiten realizar documentos que pueden contener código en lenguajes de programación (optimizado para uso con Python) y elemento de texto y gráficos, utilizado para generar reportes y documentar análisis [40].
- Scipy: Es un ecosistema de librerías de código abierto para matemáticas, ciencias e ingeniería. Incluye dentro de ellas a Numpy, Pandas, entre otras [41].

Por último, ha habido un auge en librerías para la programación de dispositivos, micro controladores y micro procesadores. Entre las más destacadas están:

- RPi.GPIO: Este paquete provee clases para poder controlar los pines GPIO de un Raspberry Pi o dispositivos basados en el [42].
- GPIO Zero: Esta librería permite controlar sensores y actuadores que se encuentre conectados a pines de entrada/salida digitales de micro controladores y microprocesadores [43].

2.3.3. R

R es un lenguaje y entorno de computación y gráficos estadísticos. R ofrece una amplia variedad de técnicas gráficas y estadísticas como el modelado lineal y no lineal, pruebas de estadística clásica, análisis de series de tiempo, clasificación, agrupamiento, entre otras [44]. Uno de los puntos fuertes de R es la facilidad con la que gráficos bien diseñados con calidad de publicación pueden ser producidos, incluyendo símbolos y fórmulas matemáticas cuando sea necesario.

R viene con un conjunto integrado de servicios de software para manipulación de datos, cálculo y representación gráfica. Incluye:

- Un manejo eficaz de datos y facilidad de almacenamiento.
- Un conjunto de operadores para cálculos en arreglos y en particular, matrices.
- Una gran, coherente e integrada colección de herramientas intermedias para análisis de datos.
- Facilidad para la graficación para el análisis de datos y para mostrar en pantallas o en impresiones.
- Un bien desarrollado, simple y efectivo lenguaje de programación que incluye condicionales, ciclos, funciones recursivas definidas por el usuario y facilidades para la entrada y salida.

R está disponible como software libre bajo los términos de la licencia Free Software Foundation's GNU en forma de código fuente. Se compila y se ejecuta en una amplia variedad de plataformas UNIX y sistemas similares (incluyendo FreeBSD y Linux), Windows y MacOS.

2.3.3.1. Principales Frameworks y Librerías

R posee una comunidad fiel de desarrolladores que ha expandido a través de la creación de paquetes las funcionalidades del lenguaje de programación. Los paquetes más utilizados [45] son los siguientes:

- e1071: Provee funciones para el análisis de clases, trasformada rápida de Fourier, agrupamiento difuso, máquinas de soporte vectorial, cómputo de caminos más cortos, clasificadores Bayesianos, entre otros.
- rpart: Contiene funcionalidades para particionamiento recursivo y para arboles de regresión.
- ROCR: Este paquete es utilizado para generar curvas ROC.
- randomForest: Este paquete genera bosques aleatorios para clasificación y regresión.
- tree: Este paquete está diseñado para brindar funciones para hacer tareas de clasificación y árboles de regresión.
- arules: Es el paquete más utilizado para la generación de reglas de asociación o conseguir conjuntos de elementos frecuentes.
- Shiny: Este paquete permite crear interfaces web interactivas con código en R.

2.3.4. Java

Java es un lenguaje de programación y una plataforma informática comercializada por primera vez en el año 1995 por Sun Microsystems [46]. Java es un lenguaje de propósito general cuyo paradigma es orientado a objetos y fuertemente tipado. Su sintaxis es similar a la de C y C++. Utiliza un máquina virtual llamada Java Virtual Machine (JVM) que es usada para poder correr sus archivos ejecutables, cuyo código se llama Java bytecode y es lo que le brinda portabilidad a los desarrollos realizados bajo este lenguaje de programación.

En Java, el problema fugas de memoria se evita en gran medida gracias a la recolección de basura en memoria por parte del automatic garbage collector. El programador determina cuándo se crean los objetos y el entorno en tiempo de ejecución de Java (Java runtime) es el responsable de gestionar el ciclo de vida de los objetos. El programa, u otros objetos pueden tener localizado un objeto mediante una referencia a éste. Cuando no quedan referencias a un objeto, el recolector de basura de Java borra el objeto, liberando así la memoria que ocupaba previniendo posibles fugas. En definitiva, el recolector de basura de Java permite una fácil creación y eliminación de objetos y mayor seguridad.

El diseño de Java, su robustez, el respaldo de la industria y su fácil portabilidad han hecho de Java uno de los lenguajes con un mayor crecimiento y amplitud de uso en distintos ámbitos:

- En dispositivos móviles: Bajo la especificación J2ME, el cual provee de un entorno de ejecución de java reducido y optimizado, especialmente desarrollado para el mercado de dispositivos móviles y sistemas embebidos.
- Navegación web: Java tiene el potencial desarrollar pequeñas aplicaciones o Applets que pueden ser aplicados a páginas web para que sean ejecutados por una JVM instalada como plug-in. Por el lado de servicios web, Java usa Servlets y JSP para poder crear aplicaciones dinámicas.
- Aplicaciones de Escritorio: El entorno de ejecución de Java (JRE) es un componente habitual en muchas plataformas de cómputo personal, por lo que es simplifica el desarrollo para múltiples sistemas operativos.

2.3.4.1. Principales Frameworks y Librerías

Java ha tenido una larga trayectoria en el área web, gracias a sus robustos frameworks. Algunos de los más importantes son:

- Spring: Es un framework basado en el patrón MVC para la creación de aplicaciones bajo un esquema de reutilización, portabilidad y un API robusto [47]. Es de código abierto y está enfocado su uso bajo J2EE.
- Grails: Es un potente framework web que tiene como objetivo multiplicar la productividad de los desarrolladores gracias a una convención sobre configuración, valores predeterminados razonables y APIs optimistas. Se integra perfectamente

con la JVM, lo que le permite ser productivo, mientras que proporciona características de gran alcance, incluyendo ORM integrado, lenguajes específicos de dominio, tiempo de ejecución y meta-programación en tiempo de compilación y eventos asíncronos [48].

- JavaServer Faces: También llamado JSF, es el framework web desarrollado por Sun Microsystems junto con Java para el desarrollo de aplicaciones web. Posee un conjunto de APIs para representar los componentes de interfaz de usuario y administrar su estado, gestionar eventos y validación de entrada, definir la navegación de páginas y apoyar la internacionalización y la accesibilidad. También tiene una biblioteca de etiquetas personalizadas de JavaServer Pages (JSP) para expresar una interfaz de JavaServer Faces dentro de una página JSP [49].

Para el desarrollo de lógica en dispositivos, micro controladores o micro procesadores, java tiene dos buenas alternativas:

- Device I/O: Es una librería, provista nativamente en la implementación de OpenJDK que provee librerías estándares para acceder a pines de entrada/salida de cualquier plataforma embebida [50]. Es de código abierto y esta optimizada para ser usada con Java 8

- Pi4J: Esta librería nace para poder trabajar en Java sobre Raspberry Pi y arquitecturas derivadas [51].

2.3.5. Scala

Scala es un lenguaje de programación similar a Java que unifica la programación orientada a objetos y la programación funcional. Es un lenguaje orientado a objetos puro en el sentido de que cada valor es un objeto. Tipos y comportamiento de los objetos son descritos por clases. Scala está diseñado para funcionar sin problemas con lenguajes orientados a objetos menos puros, como Java. En Scala las clases pueden llamar a los métodos de Java, crear objetos de Java, heredar de clases de Java e implementar interfaces de Java. Nada de esto requiere definiciones de interfaz o pegado de código [52].

Scala también se puede considerar un lenguaje funcional en el sentido de que cada función es un valor. Scala ha sido desarrollado a partir de 2001 en el laboratorio de métodos de programación de la Escuela Politécnica Federal de Lausana.

2.3.5.1. Principales Frameworks y Librerías

Scala no cuenta con un conjunto de frameworks tan amplia como otros lenguajes de programación, debido a su antigüedad. A pesar de ello cuenta ya con algunas librerías robustas como las siguientes:

- Scalaz: es una librería de Scala para la programación funcional. Proporciona estructuras de datos puramente funcionales para complementar las de la librería estándar de Scala. Define un conjunto de clases de tipo fundacional (por ejemplo, Functor, Monad) e instancias correspondientes para un gran número de estructuras de datos [53].

- ScalaTest: Es la librería de pruebas más popular en el ecosistema de Scala. ScalaTest está diseñado para crecer con las demandas de sus usuarios, es decir, puede ampliar y componer los componentes básicos de ScalaTest fácilmente para atender cualquier requerimiento especial que pueda tener. Como resultado, ScalaTest puede escalar a proyectos de todos los tamaños, desde una persona que explora una nueva idea hasta grandes equipos que colaboran en software de misión crítica [54].

2.3.6. C++

C++ es un lenguaje de programación diseñado a mediados de los años 80 por Bjarne Stroustrup. La intención de su creación fue el extender al lenguaje de programación C mecanismos que permiten la manipulación de objetos. C++ es un lenguaje multiparadigma, fuertemente tipado y compilado. Una de las ventajas de C++ es la capacidad de sobrecarga de operadores, lo que le da un dinamismo al momento de trabajar con operaciones con objetos.

Este lenguaje de programación es altamente utilizado en la programación de dispositivos debido a la capacidad de interactuar con sistemas operativos o controladores de manera muy eficiente.

2.3.6.1. Principales Frameworks y Librerías

Existen infinidad de librerías para C++ para cualquier propósito. Para el uso y control de micro controladores tenemos:

- WiringPi: Es una librería especialmente diseñada para comunicarse con sensores y actuadores a través de pines de entrada/salida de dispositivos basado en la Raspberry Pi o derivados. WiringPi incluye una utilidad de línea de comandos GPIO que puede usarse para programar y configurar los pines GPIO. Puede usar esto para leer y escribir los pines e incluso utilizarlo para controlarlos desde shell scripts [55].
- Librerías de Arduino: Son una serie de librerías para el uso y comunicación en dispositivos de arquitectura Arduino que provee una serie de funciones que extienden C++ para comunicarse con los puertos y pines que poseen [56].

2.4. Hadoop

2.4.1. Definición

Hadoop es un framework de código abierto diseñado para escribir y ejecutar aplicaciones distribuidas que procesan grandes cantidades de datos. La computación distribuida es un campo amplio y variado, sobre todos en entornos de gran cantidad de nodos y constante crecimiento donde este framework da un gran aporte gracias a sus distinciones claves:

- Accesible: Haddop se ejecuta en grandes grupos de máquinas básicas o en servicios de computación en la nube como el Elastic Compute Cloud de Amazon (EC2).
- Robusto: Debido a que está diseñado para funcionar con hardware modesto, Hadoop está diseñado con la suposición de que existirán frecuentes fallos de hardware. Puede manejar ágilmente la mayoría de estos fallos.
- Escalable: Hadoop escala linealmente añadiendo más nodos al clúster para manejar un volumen mayor de datos.
- Simple: Hadoop permite a los usuarios escribir rápidamente código paralelo eficientemente.

Un clúster de Hadoop es un conjunto de máquinas conectadas en una misma en red corriendo Hadoop en alguno de sus roles. El almacenamiento y el procesamiento de datos se producen dentro de este conjunto de máquinas. Diferentes usuarios pueden enviar "trabajos"desde clientes individuales de Hadoop, que pueden ser sus propias máquinas de escritorio en ubicación remotas al clúster de Hadoop [57].

Hadoop fue creado por Doug Cutting, el creador de Apache Lucene, una biblioteca de búsqueda de texto ampliamente utilizada. Hadoop tiene sus orígenes en Apache Nutch, un motor de búsqueda web de código abierto, que forma parte del proyecto Lucene.

En 2006 Doug Cutting se unió a Yahoo!, compañía que proporcionó un equipo dedicado y los recursos para convertir Hadoop en un sistema que funcionara a escala web. Esto fue demostrado en febrero de 2008 cuando Yahoo! anunció que su índice de búsqueda de producción estaba siendo generado por un clúster Hadoop de 10.000 núcleos. Hadoop fue hecho un proyecto de alto nivel en Apache, confirmando su éxito y con una comunidad diversa y activa. En ese momento, Hadoop estaba siendo utilizado por muchas otras compañías además de Yahoo! como Last.fm, Facebook y el New York Times.

El papel de Hadoop como plataforma de almacenamiento y análisis de propósito general para grandes datos ha sido reconocido por la industria, y este hecho se refleja en el número de productos que utilizan o incorporan Hadoop de alguna manera. El soporte comercial de Hadoop está disponible en grandes proveedores de compañías establecidas, incluyendo EMC, IBM, Microsoft y Oracle, así como de compañías especializadas de

Hadoop como Cloudera, Hortonworks y MapR. [58]

2.4.2. Arquitectura de Hadoop

Hadoop fue construido originalmente como la infraestructura para el proyecto de Nutch, que rastrea la red y construye un índice para el motor de búsqueda de las páginas rastreadas. Proporciona un sistema de archivos distribuido (HDFS) que puede almacenar datos a través de miles de servidores, y un medio de ejecutar procesos con MapReduce a los cuales se les llama trabajos (map-reduce jobs) a través de esas máquinas [59].

En el centro de la arquitectura de Hadoop se encuentra YARN que permite múltiples motores de procesamiento de datos como SQL interactivo, streaming en tiempo real, la ciencia de los datos y el procesamiento por lotes para manejar los datos almacenados en una sola plataforma, desbloqueando un enfoque totalmente nuevo a la analítica [60].

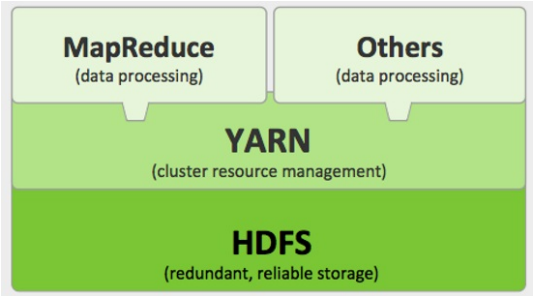

Figura 2.9: Arquitectura de Hadoop

2.4.2.1. Hadoop Common

Hadoop Common es el conjunto de librerías y utilidades necesarias para el resto de módulos que integran Hadoop. También se encarga de la abstracción del sistema operativo y de la reutilización de los componentes de Java a través de los módulos del framework y su ecosistema. Está compuesto por tres elementos:

- Commons Proper: El repositorio de componentes de java reutilizable.
- Commons Sandbox: Un espacio de trabajo para el desarrollo de componentes en Java.
- Commons Dormant: Un repositorio de componentes que se encuentran inactivos en la actualidad.

2.4.2.2. Hadoop Distributed File System

El Sistema de Archivos Distribuidos Hadoop (HDFS) es un sistema de archivos distribuido diseñado para ejecutarse en hardware básico. HDFS es altamente tolerante a fallos y está diseñado para ser desplegado en hardware de bajo costo. HDFS proporciona

un alto rendimiento de acceso a los datos de aplicación y es adecuado para aplicaciones que tienen grandes conjuntos de datos. HDFS relaja algunos requisitos POSIX para permitir la transmisión de los datos del sistema de archivos y fue construido originalmente como la infraestructura para el proyecto de motor de búsqueda web Apache Nutch. HDFS es ahora un subproyecto Apache Hadoop [61].

HDFS está optimizado para operaciones de alto rendimiento y de lectura intensiva y es resistente a fallos en el clúster. No evita los fallos, pero es improbable la pérdida de datos, ya que de HDFS realiza de forma predeterminada múltiples copias de cada uno de los bloques de datos. Además, HDFS es un sistema de "write once, read many", es decir una vez se crea el archivo, la API del sistema de archivos sólo le permite anexar al archivo, no sobre escribirlo. Como resultado, HDFS normalmente no es adecuado para las aplicaciones normales de procesamiento de transacciones en línea (OLTP).

La mayoría de los usos de HDFS son para lecturas secuenciales de archivos grandes. Estos archivos se dividen en bloques (por defecto de 64 MB) y estos bloques se distribuyen entre los nodos del servidor. HDFS no asume que los discos subyacentes del host están protegidos por RAID, por lo que de forma predeterminada se realizan tres copias de cada bloque y se colocan en diferentes nodos del clúster [62], como se puede notar en la figura 2.10.

HDFS utiliza una arquitectura maestro-esclavo. Un clúster HDFS consta de un único NameNode, un servidor maestro que gestiona el espacio de nombres del sistema de archivos y regula el acceso a los archivos por parte de los clientes. Además, hay un número de DataNodes, por lo general uno por nodo en el clúster, que gestionan el almacenamiento adjunto a los nodos en los que se ejecutan. HDFS expone un espacio de nombres de sistema de archivos y permite almacenar datos de usuario en archivos. Internamente, un archivo se divide en uno o más bloques y estos bloques se almacenan en un conjunto de DataNodes. NameNode ejecuta las operaciones del espacio de nombres del sistema de archivos como abrir, cerrar y cambiar el nombre de archivos y directorios. También determina la asignación de bloques a DataNodes. Los DataNodes son responsables de servir las peticiones de lectura y escritura de los clientes del sistema de archivos. Los DataNodes también realizan la creación, eliminación y replicación de bloques a partir de la instrucción del NameNode.

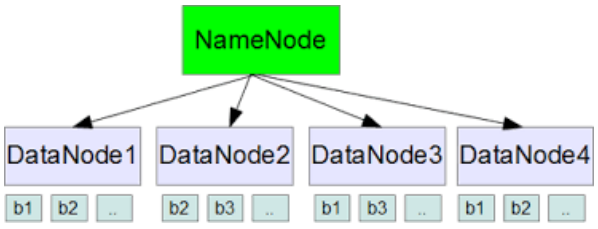

Figura 2.10: Estructura de HDFS

El NameNode y DataNode son dos software diseñados para funcionar en máquinas modestas. Estas máquinas suelen ejecutar un sistema operativo GNU/Linux. HDFS

se construye utilizando el lenguaje Java; Cualquier máquina que admita Java puede ejecutar el NameNode o el software DataNode. El uso del lenguaje Java significa que HDFS se puede desplegar en una amplia gama de máquinas. Un despliegue típico tiene una máquina dedicada que ejecuta sólo el software NameNode. Cada una de las otras máquinas del clúster ejecuta una instancia del software DataNode.

La existencia de un único NameNode en un clúster simplifica enormemente la arquitectura del sistema. NameNode es el árbitro y el repositorio para todos los metadatos HDFS. El sistema está diseñado de tal manera que los datos de usuario nunca fluyen a través del NameNode. Cualquier cambio en el espacio de nombres del sistema de archivos o sus propiedades es registrado por el NameNode. Una aplicación puede especificar el número de réplicas de un archivo que debe ser mantenido por HDFS. El número de copias de un archivo se denomina factor de replicación de ese archivo y la información es almacenada por el NameNode.

La colocación de réplicas es fundamental para la fiabilidad y el rendimiento de HDFS. La optimización de la colocación de réplicas distingue a HDFS de la mayoría de los sistemas de archivos distribuidos. Para minimizar el consumo global de ancho de banda y la latencia de lectura, HDFS intenta satisfacer una solicitud de lectura de una réplica que esté más cerca del lector. Si existe una réplica en el mismo bastidor que el nodo del lector, entonces se prefiere esa réplica para satisfacer la petición de lectura. Si el clúster HDFS abarca varios centros de datos, se prefiere una réplica residente en el centro de datos local en lugar de cualquier réplica remota [61].

2.4.2.3. Hadoop YARN

YARN (acrónimo de *Yet Another Resource Negotiator*, Otro negociador de recursos) es el manejador de recursos que utiliza Hadoop para sus operaciones. La idea fundamental de YARN es dividir las funcionalidades de la administración de recursos y la programación y monitoreo de trabajos en demonios separados. La idea es tener un ResourceManager global (RM) y por un ApplicationMaster (AM). Una aplicación es un trabajo único o un grafo acíclico de trabajos directos [63].

Los ResourceManager y NodeManager forman el framework para el cálculo de datos y su flujo de actividades se puede observar en la figura 2.11. El ResourceManager es la autoridad que arbitra los recursos entre todas las aplicaciones del sistema. El NodeManager es el agente del framework que es responsable de los contenedores, supervisando su uso de recursos (CPU, memoria, disco, red) en cada máquina y reporta lo mismo al ResourceManager. El ApplicationMaster es una biblioteca específica de framework y tiene la tarea de negociar recursos desde el ResourceManager y trabajar con el NodeManager para ejecutar y monitorear las tareas.

El ResourceManager tiene dos componentes principales: El Planificador y ApplicationsManager. El Planificador es responsable de asignar recursos a las diversas aplicaciones en ejecución sujetas a restricciones familiares de capacidades, colas, etc. El Planificador no lleva a cabo la supervisión o seguimiento del estado de la aplicación, además, no ofrece garantías sobre el reinicio de tareas fallidas ya sea debido a fallos de

la aplicación o fallos de hardware. El Planificador realiza su función basado en los requerimientos de recursos de las aplicaciones y lo hace utilizando la noción abstracta de un recurso Container que incorpora elementos como memoria, CPU, disco, red, etc [63].

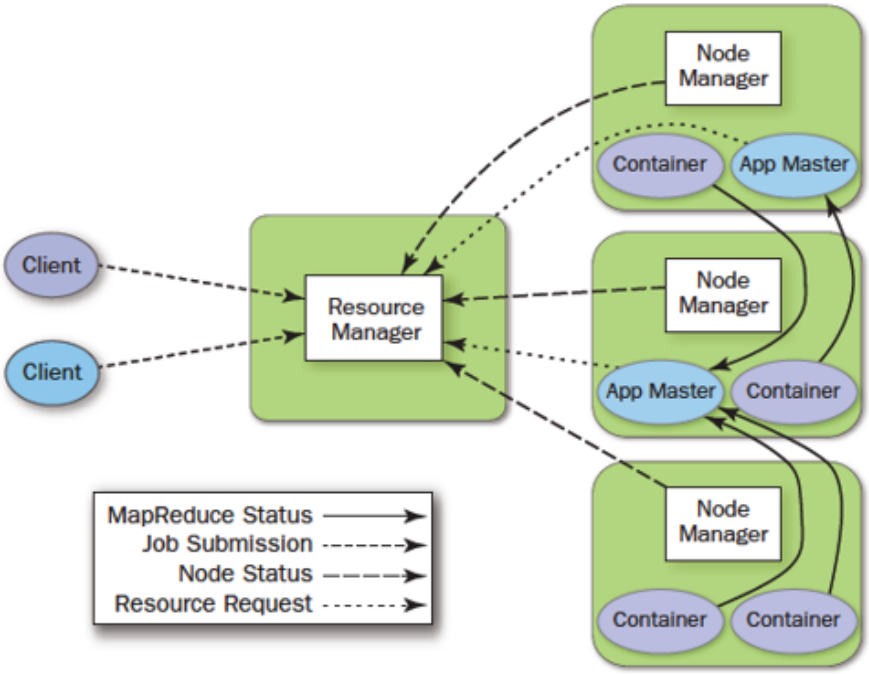

Figura 2.11: Ciclo de Vida de las Tareas de YARN

2.4.2.4. Hadoop MapReduce

MapReduce fue el primero y el principal elemento del framework de programación para el desarrollo de aplicaciones distribuidas en Hadoop. Los jobs de MapReduce consisten en programas Java llamados Map y Reduce. Orquestados por el software de Hadoop, cada uno de los Mappers (interfaces del Map) se les entrega un trozo de datos para ejecutar [62].

Hadoop toma el resultado del jobs de los Map y lo ordena. Para cada map, se crea un valor de hash para asignarlo a un reduce en un paso llamado shuffle. El Reducer (Interfaz para el Reduce) sumaría todos los maps para cada tarea en su flujo de entrada y produciría una lista ordenada de resultados. Se puede pensar en los Mappers como programas que extraen datos de archivos HDFS en Maps y los Reducers como programas que toman la salida de los Mappers y agregan los resultados (véase figura 2.12).

Cabe destacar que la forma en que se ejecuta un job de MapReduce es a través de la petición de un cliente, es decir, el envío de una petición a Hadoop. Junto con el job se debe especificar la ubicación de los archivos de la entrada y de la salida en el sistema de archivos distribuido, los formatos de entrada y salida, y las clases que contienen las funciones Map y Reduce. Luego entra en juego el JobTracker, un programa que coordina y administra los trabajos, acepta la petición y monitorea, controla y administra la

distribución de las tareas en un job para los nodos TaskTrackers.

Generalmente hay un solo JobTracker por clúster. Un TaskTracker administra las tareas en un proceso, como un Map o un Reduce. Puede existir uno o más procesos TaskTracker por nodo en el clúster. Finalmente los resultados son escritos en el sistema de archivos distribuidos. [64]

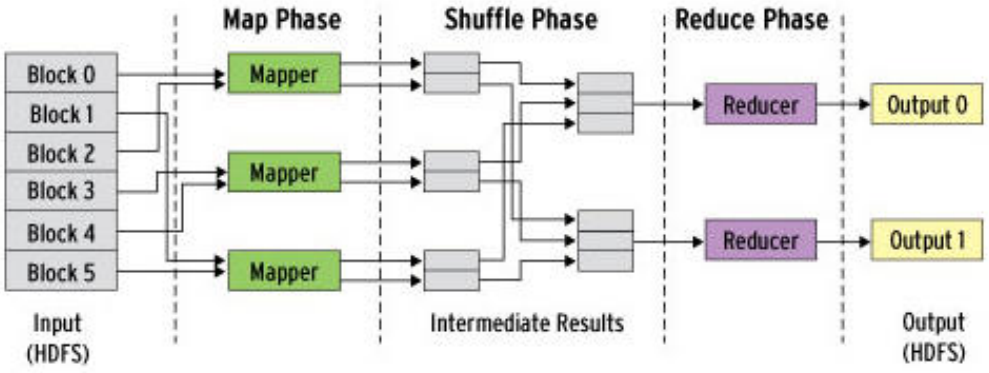

Figura 2.12: Flujo de un Proceso de MapReduce

Gran parte del trabajo duro, es decir, dividir los conjuntos de datos de entrada, la asignación de los Mappers y Reducers a los nodos, barajando los datos de los Mappers a los Reducers y la escritura de los resultados finales a la HDFS se gestiona por Hadoop mismo. Los programadores simplemente tienen que escribir las funciones Map y Reduce. Los Map y los Reduce se escriben generalmente en Java y escribir el código de MapReduce no es trivial para los principiantes. Para ello, se han desarrollado plantillas de alto nivel [62].

2.4.3. Ecosistema Hadoop

Hadoop ha crecido hasta constituir una gran familia de soluciones para el almacenamiento, gestión, interacción y análisis de grandes volúmenes de datos, integradas en un rico ecosistema de código abierto creado por la comunidad de la Apache Software Foundation. Su crecimiento es imparable, sobre todo, gracias a la febril actividad de su incubadora, que actualmente desarrolla decenas de proyectos sin restricciones, que pueden funcionar de forma independiente o llegar a superponerse en funcionalidad [65].

Como podemos ver en la figura 2.13, muchas de las herramientas no solo utilizan los componentes básicos de Hadoop sino que también se ubican transversalmente sobre el framework para aprovechar de mejor manera las capacidades que provee. A continuación se mencionaran un conjunto de herramientas que proveen características mejoradas o agregadas a Hadoop, desde el punto de vista de almacenamiento, procesamiento, integración y operación.

2.4.3.1. Apache Accumulo

Apache Accumulo es una herramienta de almacenamiento clave-valor basada en el diseño de BigTable de Google. Accumulo almacena sus datos en HDFS de Apache

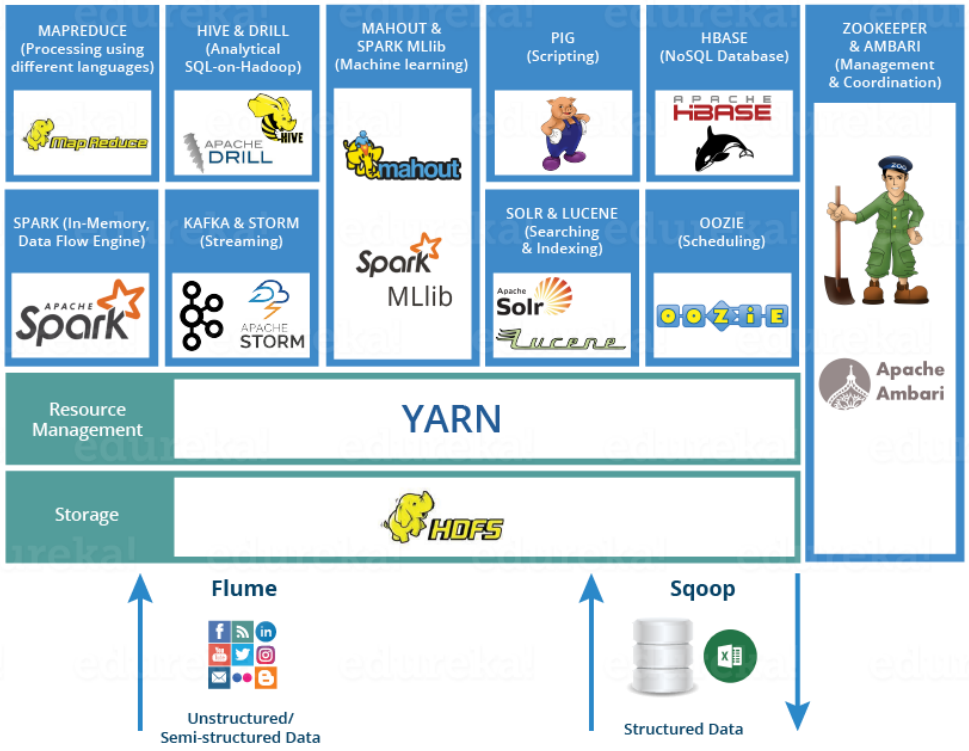

Figura 2.13: Ecosistema de Hadoop

Hadoop y utiliza Apache Zookeeper para el consenso. Mientras que muchos usuarios interactúan directamente con Accumulo, varios proyectos de código abierto utilizan Accumulo como su almacenamiento subyacente [66].

Sus características principales son:

- Programación del lado del servidor: Accumulo tiene un mecanismo de programación (llamado Iterators) que puede modificar pares clave-valor en varios puntos en el proceso de gestión de datos.
- Control de acceso basado en celdas: Cada par clave-valor en Accumulo tiene su propia etiqueta de seguridad que limita los resultados de la consulta basándose en autorización de usuario.
- Diseñado para escalar: Accumulo se ejecuta en un clúster utilizando una o más instancias de HDFS. Los nodos se pueden agregar o quitar como la cantidad de datos almacenados en Accumulo cambie.
- Estable: Accumulo tiene un API para los clientes estable que sigue el versionamiento semántico.

Por razones de seguridad interna, la Agencia de Seguridad Nacional de los Estados Unidos (NSA, por sus siglas en inglés) desarrolló Accumulo y luego donó el código a la fundación Apache. Existen muchas similitudes entre HBase y Accumulo, ya que ambos sistemas se basan en el BigTable de Google. Accumulo mejora ese modelo con su enfoque en seguridad y control de acceso basado en celdas. Cada usuario tiene un

conjunto de etiquetas de seguridad, cadenas de texto simples. [62] El control de acceso a nivel de celda es importante para organizaciones con políticas complejas que rigen a quién se le permite ver datos. Permite la mezcla de diferentes conjuntos de datos con políticas de control de acceso para que van desde el grano fino a los conjuntos de datos que tienen algunos elementos sensibles. Aquellos con permiso para ver datos sensibles pueden trabajar junto a compañeros de trabajo sin esos privilegios. Ambos usuarios pueden acceder a los datos de acuerdo con sus permisos [67].

Sin Accumulo, esas políticas son difíciles de aplicar sistemáticamente, pero Accumulo codifica esas reglas para cada caso. Pero Accumulo agrega algo más que seguridad, también se puede ejecutar a gran escala, con muchos petabytes de datos y con cientos de miles de operaciones de ingestión y recuperación por segundo.

2.4.3.2. Apache HBase

Apache HBase es una base de datos NoSQL de código abierto que proporciona acceso en tiempo real de lectura y escritura a grandes conjuntos de datos y que se ejecuta en la parte superior de HDFS. HBase escala horizontalmente para manejar enormes conjuntos de datos con miles de millones de filas y millones de columnas y combina fácilmente fuentes de datos que utilizan una amplia variedad de estructuras y esquemas diferentes. HBase se integra de forma nativa con Hadoop y funciona perfectamente junto con otros motores de acceso a datos a través de YARN.

Apache HBase proporciona acceso aleatorio y en tiempo real a sus datos en Hadoop. Fue creado para alojar tablas muy grandes, haciéndole una gran opción para almacenar datos estructurados, semi estructurados y no estructurados, sean abundantes o escasos. Los usuarios pueden consultar HBase para un punto en particular en el tiempo, haciendo consultas "flashback" posibles. [68]

Sus características [69] son:

- Escalabilidad horizontal y enfoque modular.
- Consistencia estricta en lectura y escritura.
- Configuración y Sharding automático de tablas.
- Soporte automático de conmutación por error entre RegionServers.
- Convenientes clases base para respaldar trabajos Hadoop MapReduce con tablas de Apache HBase.
- API Java fácil de usar para el acceso del cliente.
- Bloquea caché y filtros Bloom para consultas en tiempo real.
- Gsteway para terceros y un servicio Web REST-ful que soporta XML, Protobuf y opciones de codificación de datos binarios.
- Shell extensible basado en jruby (JIRB).

- Soporte para exportar métricas a través del subsistema de métricas Hadoop a archivos.

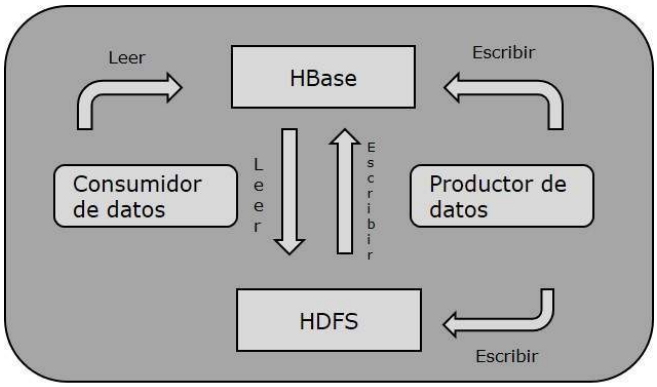

Figura 2.14: Flujo de Trabajo de HBase

HBase se utiliza a menudo para aplicaciones que pueden requerir filas escasas. Es decir, cada fila puede utilizar sólo algunas de las columnas definidas. Es rápido (como lo hace Hadoop) cuando el acceso a los elementos se realiza a través de la clave primaria o la definición del valor de la clave. Es altamente escalable y razonablemente rápido. A diferencia de las aplicaciones tradicionales HDFS, permite el acceso aleatorio a las filas, en lugar de búsquedas secuenciales.

Aunque sea más rápido que otras alternativas, no se debe usar HBase para cualquier tipo de necesidades transaccionales, ni ningún tipo de análisis relacional. No admite ningún índice secundario, por lo que encontrar todas las filas donde una determinada columna tiene un valor específico es tedioso y debe realizarse en el nivel de la aplicación. HBase no tiene una operación JOIN. También se debe proporcionar seguridad en el nivel de aplicación [62].

2.4.3.3. Apache Hive

El software de almacenamiento de datos Apache Hive facilita la lectura, escritura y administración de conjuntos de datos grandes que residen en el almacenamiento distribuido y se consultan con la sintaxis SQL [70].

Hive proporciona funcionalidad estándar de SQL, incluyendo muchas de las características posteriores de SQL 2003 y SQL 2011 para análisis. El SQL de Hive también puede extenderse con código propio a través de funciones definidas por el usuario (UDF), agregados definidos por el usuario (UDAF) y funciones de tabla definidas por el usuario (UDTF). No hay un solo "Formato de Hive" en el que se deben almacenar los datos. Hive viene con conectores integrados para archivos de texto con coma y separadores (CSV, TSV) [71]. Hive define un lenguaje de consulta similar a SQL, llamado HQL, que permite a los usuarios familiarizados con SQL consultar los datos. Las consultas escritas en HQL son convertidas en código MapReduce por Hive y ejecutadas por Hadoop [62].

Hive está diseñado para maximizar la escalabilidad (escala con más máquinas agregadas dinámicamente al clúster Hadoop), rendimiento, extensibilidad, tolerancia a fallos

y acoplamiento suelto con sus formatos de entrada.

Hive ofrece las siguientes características:

- Herramientas para facilitar el acceso a datos a través de SQL, permitiendo así tareas de almacenamiento de datos como extraer, transformar, cargar, reportar y analizar datos.
- Un mecanismo para imponer la estructura en una variedad de formatos de datos.
- Acceso a archivos almacenados directamente en HDFS o en otros sistemas de almacenamiento de datos como HBase.
- Ejecución de consultas a través de Apache Tez, Apache Spark o MapReduce.
- Lenguaje de procedimiento con HPL-SQL.
- recuperación de consultas a través de Hive LLAP, Apache YARN y Apache Slider.

Hive no está diseñado para cargas de trabajo de procesamiento de transacciones en línea (OLTP). Se utiliza mejor para tareas tradicionales de almacenamiento de datos [71].

2.4.3.4. Apache Pig

Apache Pig es una plataforma para el análisis de grandes conjuntos de datos que consta de un lenguaje de alto nivel para expresar programas de análisis, junto con la infraestructura para la evaluación de los mismos. La característica sobresaliente de los programas de Pig es que su estructura es susceptible a la paralelización, lo que a su vez le permite manejar enormes cantidades de información [72].

La capa de infraestructura de Pig se compone de un compilador que produce secuencias MapReduce, lo que permite que los usuarios de Hadoop se enfoquen más en analizar los datos y dedicar menos tiempo en desarrollar aplicaciones MapReduce. El lenguaje de programación que utiliza Pig, Pig Latin, crea estructuras tipo SQL, de manera que, en lugar de escribir aplicaciones separadas de MapReduce, se pueda crear un script de Pig Latin el cual es automáticamente paralelizado y distribuido a través de un clúster [73]. Pig Latin tiene como propiedades claves las siguientes:

- Facilidad de programación: Es trivial lograr la ejecución paralela de tareas de análisis de datos. Las tareas complejas compuestas de múltiples transformaciones de datos interrelacionadas se codifican explícitamente como secuencias de flujo de datos, haciéndolas fáciles de escribir, comprender y mantener.
- Oportunidades de optimización: La forma en que se codifican las tareas permite al sistema optimizar su ejecución automáticamente, permitiendo al usuario centrarse en la semántica en lugar de la eficiencia.
- Extensibilidad: Los usuarios pueden crear sus propias funciones para hacer un procesamiento de propósito especial.

Para lograr un mejor entendimiento sobre el objetivo de la creación de Pig, el equipo de desarrollo decidió definir una serie de enunciados que resumen el proyecto, mediante una similitud con el nombre:

- Pigs eat anything (Los cerdos comen de todo): Al igual que cualquier cerdo que come cualquier cosa, Pig puede operar con cualquier tipo de dato, sea éste estructurado, semi-estructurado o no estructurado.
- Pigs live anywhere (Los cerdos viven en cualquier lado): A pesar de que Pig fue inicialmente implementado en Hadoop, no está orientado solamente a esta plataforma. Su propósito es ser un lenguaje de procesamiento paralelo.
- Pigs are domestic animals (Los cerdos son animales domésticos): Pig está diseñado para ser controlado y modificado fácilmente por sus usuarios. Pig puede enriquecerse a través de funciones definidas por el usuario (UDF). Con el uso de UDFs se puede extender Pig para un procesamiento personalizado.
- Pigs Fly (Los cerdos vuelan): Pig procesa datos rápidamente. La intención es mejorar el rendimiento y no las características, lo que evita que demasiada funcionalidad le impida "volar".

Pig proporciona una abstracción útil sobre MapReduce, es decir, permite escribir sentencias de manipulación de datos y consultas en un lenguaje de alto nivel y gracias a su modelo subyacente es capaz de paralelizar automáticamente y escalar las operaciones realizadas proporcionando un fuerte apoyo para el trabajo con conjuntos de datos muy grandes [73].

2.4.3.5. Apache Mahout

Apache Mahout es un nuevo proyecto de código abierto de la Apache Software Foundation con el objetivo principal de crear algoritmos escalables de aprendizaje automático que son gratuitos para usar bajo la licencia Apache. El proyecto está entrando en su segundo año, con un lanzamiento público en su haber. Mahout contiene implementaciones para agrupamiento, categorización y programación evolutiva. Además, cuando es prudente, utiliza las librerías Apache Hadoop para habilitar a Mahout el poder escalar con eficacia en la nube [74].

El software Apache Mahout ofrece tres características principales:

- Un entorno de programación simple y extensible y un framework para la construcción de algoritmos escalables.
- Una amplia variedad de algoritmos hechos previamente para Scala + Apache Spark, H2O, Apache Flink, entre otros.
- Samsara, es un entorno de experimentación matemática vectorial con sintaxis similar a R que funciona a escala [75].

Mahout cuenta con varios algoritmos de clasificación, la mayoría de los cuales (con una excepción notable, del algoritmo del gradiente estocástico descendiente) están escritos para ejecutarse en Hadoop. Para que los algoritmos de clasificación de Mahout funcionen, se debe entrenar un modelo para que represente los patrones que serán identificados, y luego probados de nuevo contra un subconjunto de los datos. En la mayoría de los problemas de clasificación, una o más personas deben pasar y anotar manualmente un subconjunto de los datos que serán usados en el entrenamiento. No obstante y afortunadamente, en este caso el conjunto de datos ya está separado por proyecto, por lo que no hay necesidad de anotación manual [76].

2.4.3.6. Apache Flume

Flume es un servicio distribuido, confiable y disponible para recopilar, agregar y mover eficientemente grandes cantidades de datos en registros. Tiene una arquitectura simple y flexible basada en flujos de datos en streaming. Es robusto y tolerante a fallos con mecanismos de confiabilidad sintonizables y muchos mecanismos de recuperación y conmutación por error. Utiliza un modelo de datos extensible simple que permite la aplicación de analítica en línea [77].

En Flume, las entidades con las que trabaja se llaman fuentes, decoradores y sumideros. Una fuente puede ser cualquier fuente de datos, y Flume tiene muchos adaptadores de fuentes predefinidas. Un sumidero es el objetivo de una operación específica (y en Flume, entre otros paradigmas que usan este término, el sumidero de una operación puede ser la fuente para la siguiente operación descendente). Un decorador es una operación en el flujo que puede transformar el flujo de alguna manera, que podría ser para comprimir o descomprimir datos, modificar datos mediante la adición o eliminación de piezas de información y mucho más [78].

Flume soporta complejos flujos multi-usuario y fan-in y fan-out. Los eventos se organizan por canales y agentes que llevan el flujo por toda la cadena y que finalmente se eliminan una vez que llegan al siguiente agente o a HDFS, es decir, un sumidero final (véase figura 2.15) Un proceso de Flume tiene un archivo de configuración que lista las fuentes, sumideros y canales para el flujo de datos. Casos de uso típicos incluyen cargar datos de registro en Hadoop. [62] YARN coordina la ingesta de datos de Apache Flume y otros servicios que proporcionan datos sin procesar en un clúster Hadoop [79].

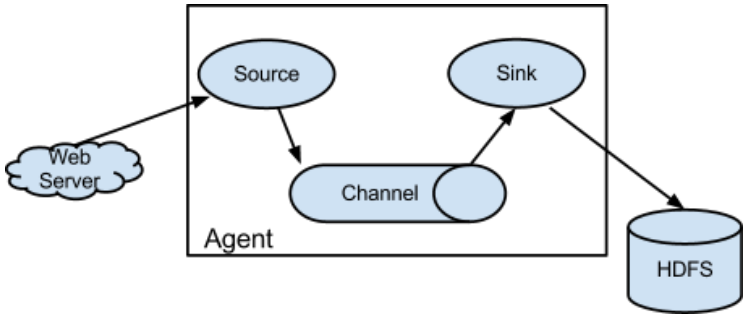

Figura 2.15: Flujo de Datos en Apache Flume

2.4.3.7. Apache Spark

Apache Spark es un sistema de computación de clúster rápido y de propósito general. Proporciona APIs de alto nivel en Java, Scala, Python y R y un motor optimizado que soporta gráficos de ejecución general. También soporta un amplio conjunto de herramientas de alto nivel, como Spark SQL para el uso del estándar SQL y procesamiento de datos estructurado, MLlib para el aprendizaje automatizado, GraphX para procesamiento de gráficos y Spark Streaming [80]. Se puede apreciar su arquitectura en la figura 2.16.

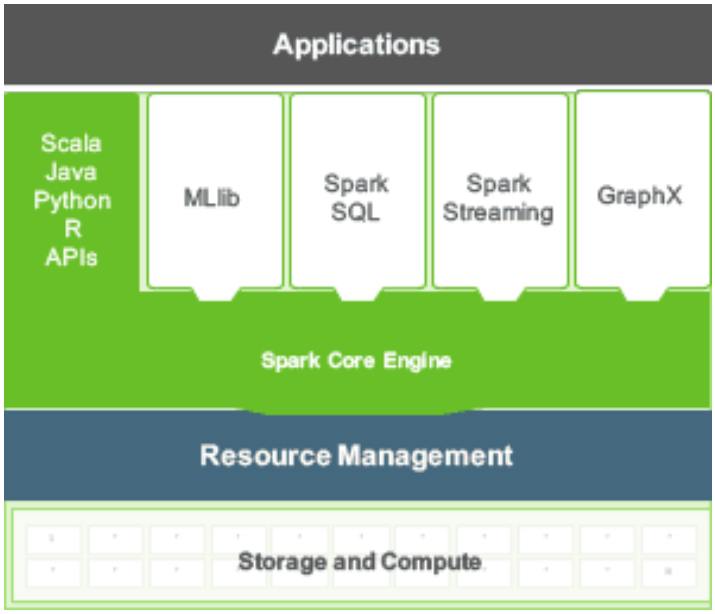

Figura 2.16: Arquitectura de Apache Spark

Spark está diseñado para proporcionar un modelo más flexible que de soporte a muchas de las aplicaciones multipass que vacilan en MapReduce. Se logra este objetivo aprovechando la memoria principal siempre que sea posible a fin de reducir la cantidad de datos que se escriben y leen desde el disco. A diferencia de Pig y Hive, Spark no es una herramienta para hacer MapReduce más fácil de usar. Es un reemplazo completo de MapReduce que incluye su propio motor de ejecución de trabajo [62]. Spark opera con tres ideas principales:

- Resilient Distributed Dataset (RDD): Los RDD (conjuntos de datos elásticos distribuidos) contienen datos que se desea transformar o analizar. Pueden ser leídos de una fuente externa, como un archivo o una base de datos, o pueden ser creados por una transformación.
- Transformación: Una transformación modifica un RDD existente para crear un nuevo RDD. Por ejemplo, un filtro que extrae mensajes ERROR de un archivo de registros sería una transformación.
- Acción: Una acción analiza un RDD y devuelve un solo resultado. Por ejemplo, una acción contaría el número de resultados identificados por un filtro ERROR.

El lenguaje de programación más usado con Spark es Scala. Esto no quiere decir que Scala sea la única manera de trabajar con Spark. El proyecto también tiene un fuerte soporte para Java y Python, pero cuando se agregan nuevas API o características, aparecen primero en Scala.

Las bibliotecas adicionales, construidas encima del núcleo, permiten diversas cargas de trabajo para streaming, SQL y aprendizaje automatizado. Spark está pensado para ser usado en la ciencia de los datos y su abstracción la facilita. Los científicos de datos suelen utilizar el aprendizaje automatizado, un conjunto de técnicas y algoritmos que pueden aprender de los datos. Estos algoritmos son a menudo iterativos, y la capacidad de Spark para almacenar en caché el conjunto de datos en la memoria acelera de manera importante tal procesamiento de datos iterativo, haciendo de Spark un motor de procesamiento ideal para implementar tales algoritmos.

Spark también incluye MLlib, una biblioteca que proporciona un creciente conjunto de algoritmos de aprendizaje de máquina para técnicas comunes como: clasificación, regresión, filtrado colaborativo, agrupamiento y reducción de dimensionalidad. El API de Pipelines de aprendizaje automatizado de Spark es una abstracción de alto nivel para modelar un flujo de trabajo completo de ciencias de datos. Proporciona abstracciones como Transformer, Estimator, Pipeline y Parameters que conforman una capa de abstracción que hace que los científicos de datos sean más productivos [81].

2.4.3.8. Apache Kafka

Kafka es un sistema de mensajería de publicación-suscripción distribuido que está diseñado para ser rápido, escalable y duradero. Al igual que muchos sistemas de mensajería de publicación-suscripción, Kafka mantiene la alimentación de los mensajes en temas. Los productores pueden escribir datos en los temas y los consumidores pueden leerlos de allí. Al ser Kafka es un sistema distribuido, los temas se reparten y se replican a través de múltiples nodos.

Los mensajes son simplemente las matrices de bytes y los desarrolladores pueden utilizar para almacenar cualquier objeto en cualquier formato. Es posible adjuntar una clave a cada mensaje, en cuyo caso se le garantiza a los productores que todos los mensajes con la misma clave llegarán a la misma partición. Cuando se consume de un tema, es posible configurar un grupo de múltiples consumidores. Cada consumidor en un grupo de consumidores leerá los mensajes de un único subconjunto de particiones en cada tema que se suscriban, por lo que cada mensaje se entrega a un consumidor en el grupo y todos los mensajes con la misma clave llegan al mismo consumidor.

Lo que hace único a Kafka es que trata cada tema particionado como una bitácora (un conjunto ordenado de mensajes). Cada mensaje en una partición se le asigna un desplazamiento único. Kafka no intenta realizar un seguimiento de los mensajes que fueron leídos por cada consumidor y sólo retener los mensajes no leídos, más bien, Kafka se reserva todos los mensajes por una cantidad fija de tiempo y los consumidores son responsables de rastrear su ubicación en cada registro. En consecuencia, Kafka puede soportar un gran número de consumidores y retener grandes cantidades de datos con

muy poca sobrecarga [82].

Kafka tiene cuatro APIs fundamentales:

- La API del productor permite que una aplicación publique un flujo de registros a uno o más temas Kafka.
- La API del consumidor permite que una aplicación pueda suscribirse a uno o más temas y procesar el flujo de los registros producidos con ellos.
- El API de streams permite que una aplicación pueda actuar como un procesador de flujos, el consumo de un flujo de entrada de uno o más temas y producir un flujo de salida a uno o más temas de salida, transformando de manera efectiva el flujo de entrada a un flujo de salida.
- La API de conectores permite la construcción y el funcionamiento de productores o consumidores reutilizables que se conecten con los temas de Kafka a las aplicaciones existentes o sistemas de datos.

Mediante la combinación de las suscripciones de almacenamiento y de baja latencia, las aplicaciones de transmisión de datos pueden procesar datos pasados y futuros de la misma manera. Esto quiere decir que una sola aplicación puede procesar datos históricos almacenados, pero en lugar de terminar cuando llega al último registro, puede mantener el procesamiento de datos que vayan llegando en el futuro. Del mismo modo, para la transmisión de procesos de datos y la combinación de suscripción a eventos en tiempo real hacen posible el uso de Kafka para pipelines de baja latencia, pero la capacidad de almacenar datos de forma fiable hace que sea posible su uso para los datos críticos en los que se debe garantizar la entrega o para la integración con los sistemas fuera de línea que cargan los datos sólo periódicamente o puedan darse de baja durante períodos prolongados de tiempo para el mantenimiento. Las instalaciones de procesamiento de flujo hacen posible la transformación de los datos a medida que llegan [83].

2.4.3.9. Apache Storm

Apache Storm es un sistema de computación en tiempo real distribuido libre y abierto. Storm facilita el procesamiento fiable de flujos de datos sin límites, haciendo para el procesamiento en tiempo real lo que hizo Hadoop para el procesamiento por lotes. Storm es simple, se puede utilizar con cualquier lenguaje de programación [84].

Apache Storm agrega capacidades de procesamiento de datos en tiempo real confiables a Hadoop. Storm sobre YARN es una potente combinación para escenarios que requieren análisis en tiempo real, aprendizaje automatizado y monitoreo continuo de operaciones. Storm se integra con YARN a través de Apache Slider, YARN gestiona Storm mientras también considera los recursos del clúster para los componentes de gobernanza de datos, seguridad y operaciones de una moderna arquitectura de datos. Cinco características hacen que Storm sea ideal para cargas de trabajo de procesamiento de datos en tiempo real, las cuales son:

- Rápido: Su rendimiento ha sido medido con el procesamiento de un millón de mensajes de 100 bytes por segundo por nodo.
- Escalable: Con cálculos paralelos que se ejecutan a través de un grupo de máquinas.
- Tolerante a las fallas: Cuando los demonios mueren, Storm los reiniciará automáticamente. Si un nodo muere, el demonio se reiniciará en otro nodo.
- Confiable: Storm garantiza que cada unidad de datos (tupla) se procesará al menos una vez o exactamente una vez. Los mensajes sólo se reproducen cuando hay fallos.
- Fácil de operar: Las configuraciones estándar son adecuadas para la producción desde el primer día. Una vez desplegado, Storm es fácil de operar.

En Storm, un flujo de trabajo se denomina "topología", con entradas llamadas "picos" y transformaciones llamadas "pernos". Es importante tener en cuenta que las topologías de Storm son muy diferentes de las de MapReduce porque los trabajos tienen un principio y un final mientras que las topologías no. La intención es que una vez que se define una topología, los datos seguirán fluyendo desde su pico y serán procesados a través de una serie de pernos. [62] Las topologías de Storm están dispuestas en forma de grafo dirigido acíclico (DAG). Los flujos de datos entre los componentes es representado como ese grafo. Cada componente consume uno o más flujos de datos, y puede emitir opcionalmente una o más flujos. En la figura 2.17 muestra cómo los flujos de datos entre los componentes en una topología básica para contar el número de palabras.

Los componentes del pico introducen datos en una topología y emiten uno o más flujos en la topología. Los componentes del perno consumen los flujos emitidos por los picos u otros pernos. Los pernos opcionalmente pueden emitir flujos en la topología. Los pernos también son responsables de escribir datos en servicios externos o almacenamiento, como HDFS, Kafka o HBase.

2.4.3.10. Apache Falcon

Apache Falcon es un motor de gestión de datos que define, programa y supervisa las políticas de gestión de datos. Falcon permite a los administradores de Hadoop definir centralmente sus líneas de datos y a continuación, Falcon utiliza esas definiciones para generar automáticamente flujos de trabajo en Apache Oozie. El equipo de InMobi inició el proyecto para satisfacer sus necesidades de políticas para administrar la forma en que fluían esos datos a través de su clúster, específicamente para replicación, gestión del ciclo de vida, procedencia y trazabilidad de datos [85].

Apache Falcon simplifica los flujos de trabajo complicados de gestión de datos en las definiciones de entidades generalizadas, haciendo mucho más fácil:

- Definir pipelines de datos
- Monitorear los pipelines de datos en coordinación con Ambari.

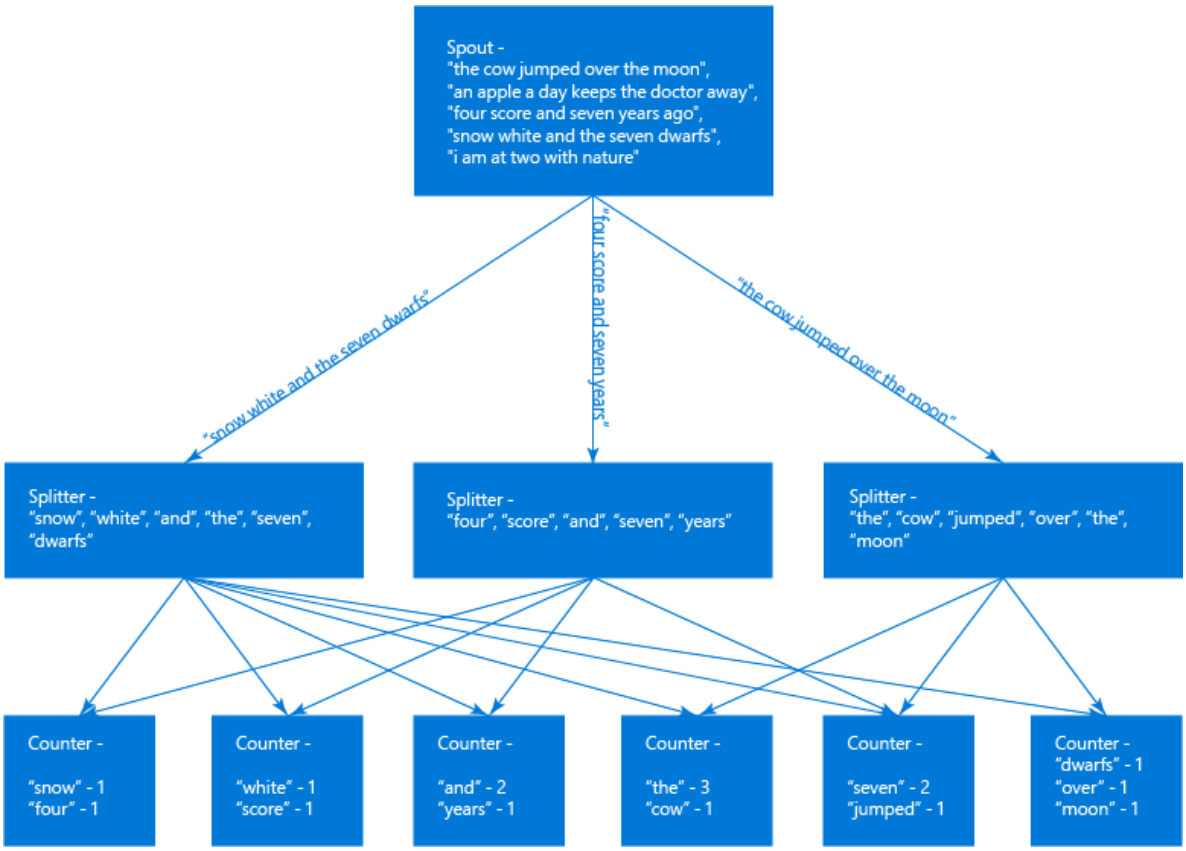

Figura 2.17: Ejemplo de Grafo Dirigido Acíclico en una Topología para Storm

- Trazar pipelines para dependencias, etiquetado, auditorias y procedencia.

Los clústeres Hadoop maduros pueden tener cientos o miles de trabajos para el coordinador de Oozie. A ese nivel de complejidad, resulta difícil administrar tantos conjuntos de datos y definiciones de procesos. Esto resulta en algunos errores comunes. Los procesos pueden usar copias incorrectas de conjuntos de datos. Los conjuntos de datos y los procesos pueden duplicarse y cada vez es más difícil rastrear dónde se originó un conjunto de datos determinado.

Falcon aborda estos desafíos de administración de datos con entidades de alto nivel y reutilizables. Las políticas de gestión de datos se definen en entidades Falcon y se manifiestan como flujos de trabajo de Oozie. Apache Falcon define tres tipos de entidades que se pueden combinar para describir todas las políticas y canalizaciones de administración de datos, las cuales son:

- Clúster.
- Feed (es decir, conjunto de datos).
- Proceso.

2.4.3.11. Apache Sqoop

Apache Sqoop es una herramienta diseñada para transferir de forma eficiente datos en masa entre Apache Hadoop y datos operacionales estructurados como bases de datos relacionales. [86] Sqoop posee conectores por defecto para las bases de datos Teradata, Netezza, Oracle, MySQL, Postgres y HSQLDB, entre otras [87].

Sqoop (que significa SQL a Hadoop) está diseñado para transferir datos entre clústeres Hadoop y bases de datos relacionales. Es un proyecto de alto nivel de Apache desarrollado por Cloudera, ahora en el dominio público. Mientras que Sqoop automatiza gran parte del proceso, se requiere cierto conocimiento SQL para que este trabajo funcione correctamente. El trabajo Sqoop se transforma en un job MapReduce que realiza la(s) tarea(s).

Sqoop comenzará con la importación a Hadoop con una tabla de base de datos que se lea en Hadoop como un archivo de texto o en formato Avro o SequenceFile. También puede exportar un archivo HDFS a un sistema manejador de bases de datos. En este caso, el trabajo de MapReduce lee un conjunto de archivos delimitados por texto dentro de HDFS en paralelo y los convierte en filas en un sistema manejador. Hay opciones para filtrar filas y columnas, alterar delimitadores y más [62].

La importación se realiza en dos pasos como se muestra en la figura 2.18. En el primer paso Sqoop explora la base de datos para reunir los metadatos necesarios para los datos que se importan. El segundo paso es un trabajo Map de Hadoop que Sqoop envía al clúster. Es este trabajo el que realiza la transferencia de datos real utilizando los metadatos capturados en el paso anterior. Los datos importados se guardan en un directorio de HDFS basado en la tabla que se importa. Como es el caso con la mayoría de los aspectos de la operación de Sqoop, el usuario puede especificar cualquier directorio alternativo donde los archivos deban ubicarse .

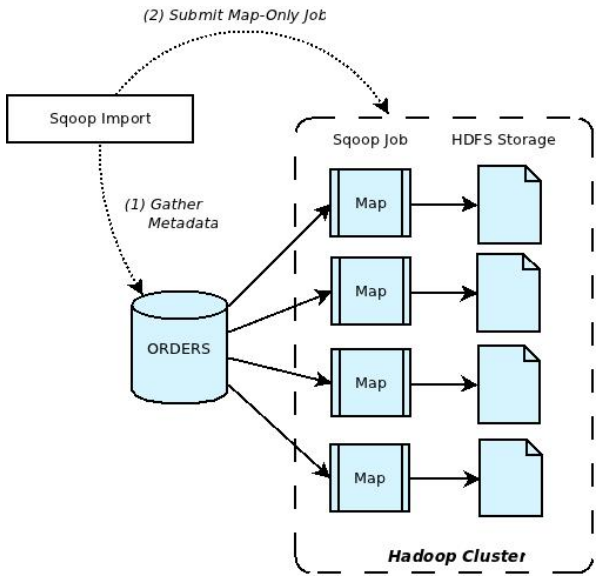

Figura 2.18: Proceso de Importación de Datos Usando Sqoop

En algunos casos, los datos procesados por los pipelines Hadoop pueden ser necesarios en los sistemas de producción para ayudar a ejecutar funciones empresariales críticas adicionales. Sqoop puede utilizarse para exportar dichos datos a almacenes de datos externos según sea necesario. La exportación se realiza en dos pasos, como se muestra en la figura 2.19. El primer paso es explorar la base de datos para los metadatos, seguido de la transferencia de los datos. Sqoop divide el conjunto de datos de entrada en divisiones y luego utiliza tareas Map individuales para empujar las divisiones a la base de datos. Cada tarea Map realiza esta transferencia a través de muchas transacciones con el fin de asegurar un rendimiento óptimo y una utilización mínima de los recursos [88].

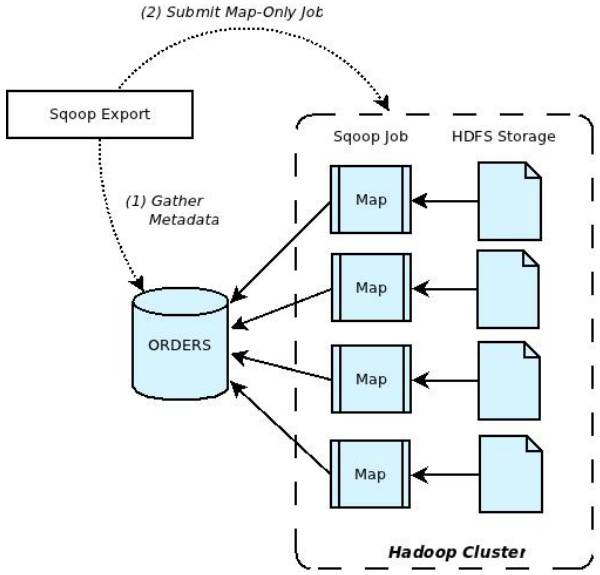

Figura 2.19: Proceso de Exportación de Datos Usando Sqoop

2.4.3.12. Apache Knox

La puerta de enlace Apache Knox es un sistema que proporciona un punto de autenticación y acceso únicos a los servicios de Apache Hadoop en un clúster. La puerta de enlace Knox simplifica la seguridad de Hadoop para los usuarios que acceden a los datos del clúster y ejecuta trabajos y operadores que controlan el acceso y gestionan el clúster. La puerta de enlace se ejecuta como un servidor, o un clúster de servidores, proporcionando acceso centralizado a uno o varios clústeres de Hadoop [89].

La API de Knox está diseñada como un proxy inverso con consideración de la capacidad de conexión en las áreas de cumplimiento de políticas, a través de proveedores y los servicios de back-end para los que propone las solicitudes.

La aplicación de políticas abarca reglas de autenticación, federación, autorización, auditoría, despacho, hostmapping y reescritura de contenido. La política se aplica a través de una cadena de proveedores que se definen en el descriptor de despliegue de topología para cada clúster de Hadoop habilitado por Knox. La definición del clúster también se define dentro del descriptor de despliegue de topología y proporciona la

puerta de enlace de Knox con el diseño del clúster para propósitos de enrutamiento y traducción entre URLs orientadas al usuario y elementos internos del clúster.

Cada clúster de Apache Hadoop que está protegido por Knox tiene su conjunto de APIs REST representadas por una única ruta de contexto de aplicación específica del clúster. Esto permite que la puerta de enlace de Knox proteja a varios clústeres y presente al consumidor de la API REST con un único punto final para el acceso a todos los servicios requeridos, a través de los múltiples clústeres.

Simplemente escribiendo un descriptor de implementación de la topología en el directorio de topologías de la instalación de Knox, se procesa una nueva definición de clúster de Hadoop, se configuran los proveedores de cumplimiento de políticas y se pone a disposición de los usuarios del API la ruta de contexto de la aplicación [90].

2.4.3.13. Apache Ranger

Ranger es un framework para permitir, supervisar y administrar la seguridad integral de datos a través de la plataforma Hadoop. Con la llegada de Apache YARN, la plataforma Hadoop puede ahora soportar una verdadera arquitectura de lago de datos. Las empresas pueden ejecutar varias cargas de trabajo, en un entorno multi-inquilino. La seguridad de los datos dentro de Hadoop evoluciono para soportar múltiples casos de uso para el acceso a los datos, al tiempo que proporciona un marco para la administración central de las políticas de seguridad y el monitoreo del acceso de los usuarios. [91]

Apache Ranger tiene los siguientes objetivos:

- Administración de seguridad centralizada para administrar todas las tareas relacionadas con la seguridad en una interfaz de usuario central o con las API REST.
- Autorización de grano fino para realizar una acción y/u operación específica con el componente/herramienta de Hadoop y gestionarse a través de una herramienta de administración central.
- Estandarizar el método de autorización en todos los componentes de Hadoop.
- Soporte mejorado para diferentes métodos de autorización-control de acceso basado en roles, control de acceso basado en atributo, etc.
- Centralizar la auditoría de acceso de usuarios y acciones administrativas (relacionadas con la seguridad) dentro de todos los componentes de Hadoop.

Con la consola de Ranger, los administradores de seguridad pueden administrar fácilmente las directivas de acceso a archivos, carpetas, bases de datos, tablas o columnas. Estas directivas se pueden establecer para usuarios o grupos individuales y a continuación, se aplican de forma consistente en la pila de aplicaciones de Hadoop.

El servicio de administración de claves de Ranger (Ranger KMS) proporciona un servicio escalable de administración de claves criptográficas para la encriptación HDFS "datos en reposo". Ranger KMS se basa en el Hadoop KMS originalmente desarrollado

por la comunidad Apache y amplía su funcionalidad nativa permitiendo a los administradores del sistema almacenar claves en una base de datos segura [92].

Ranger también proporciona a los administradores de seguridad una visibilidad profunda en su entorno Hadoop a través de una ubicación de auditoría centralizada que realiza el seguimiento de todas las solicitudes de acceso en tiempo real y admite varias fuentes de destino, incluyendo HDFS.

2.4.3.14. Apache Ambari

Apache Ambari tiene como objetivo hacer más sencilla la gestión de Hadoop mediante el desarrollo de software para el aprovisionamiento, gestión y supervisión de clústeres Apache Hadoop. Ambari proporciona una interfaz de usuario intuitiva y fácil de usar para la administración de Hadoop, respaldada por sus APIs RESTful [93].

Con Ambari, los operadores de Hadoop obtienen los siguientes beneficios [94]:

- Instalación, configuración y administración simplificadas: Ambari permite crear, administrar y monitorear clusters a gran escala fácil y eficientemente. Elimina las conjeturas de la configuración con las "Configuraciones Inteligentes" y las "Recomendaciones de Cluster". Permite la creación de clústeres automatizados y repetibles con Ambari Blueprints.
- Configuración de seguridad centralizada: Reduce la complejidad para administrar y configurar la seguridad de clúster en toda la plataforma. Ayuda a automatizar la configuración de capacidades avanzadas de seguridad del clúster como Kerberos y Apache Ranger
- Visibilidad completa en la salud del clúster: Asegura que el clúster está sano y disponible con un enfoque holístico de monitoreo. Configura alertas predefinidas basadas en las mejores prácticas operativas para el monitoreo de clústeres. Captura y visualiza métricas operativas críticas (usando Grafana) para análisis y solución de problemas.
- Altamente extensible y adaptable: Ambari ajusta Hadoop de forma transparente en su entorno empresarial. Altamente extensible con Ambari Stacks para llevar servicios personalizados bajo administración y con Ambari Views para personalizar la interfaz web de Ambari.

2.4.3.15. Apache Oozie

Apache Oozie es una aplicación web Java utilizada para programar los trabajos de Apache Hadoop. Oozie combina múltiples trabajos secuencialmente en una unidad lógica de trabajo. Está integrado con la pila Hadoop, con YARN como su centro de arquitectura y soporta los trabajos de Hadoop para Apache MapReduce, Apache Pig, Apache Hive y Apache Sqoop. Oozie también puede programar trabajos específicos para un sistema, como programas Java o shell scripts [95].

Hay dos tipos básicos de trabajos de Oozie:

- Los Oozie Workflow son grafos dirigidos acíclicos (DAG), que describen qué acciones dependen de las acciones anteriores que terminan exitosamente. Esto se define en un archivo XML grande (en realidad HPDL, Hadoop Process Definition Language) [62].
- Los Oozie Coordinator son trabajos recurrentes de Oozie Workflow que son accionados por la disponibilidad del tiempo y de datos.

Oozie Bundle proporciona una forma de empaquetar múltiples trabajos de coordinador y flujo de trabajo y administrar el ciclo de vida de dichos trabajos.

2.4.3.16. Apache ZooKeeper

ZooKeeper es un servicio centralizado para mantener información de configuración, nombrar, proporcionar sincronización distribuida y proporcionar servicios de grupo. Todos estos tipos de servicios se utilizan de alguna forma u otra mediante aplicaciones distribuidas. Cada vez que se implementan requieren una gran cantidad de trabajo que pasan por la fijación de los errores y las condiciones de carrera que son inevitables. Debido a la dificultad de implementar este tipo de servicios, las aplicaciones suelen inicialmente escatimar en ellos, lo que los hace frágiles ante la presencia de cambios y difíciles de manejar. Incluso cuando se realiza correctamente, las diferentes implementaciones de estos servicios conducen a la complejidad de la administración cuando se implementan las aplicaciones. [96]

Hadoop y HDFS son herramientas eficaces para distribuir trabajo en muchas máquinas, pero a veces es necesario compartir rápidamente pequeños bits de información entre varios procesos simultáneamente en ejecución. ZooKeeper se construye exactamente para este tipo de necesidades: es un mecanismo efectivo para almacenar y compartir pequeñas cantidades de datos de estado y configuración en muchas máquinas.

Mejor aún, el almacenamiento de la información de conexión en ZooKeeper permite a los analistas acceder rápidamente a la información al mismo tiempo que proporciona un mecanismo sencillo para actualizaciones. ZooKeeper no pretende llenar el espacio de HBase o cualquier otra base de datos. De hecho, hay protecciones incorporadas en ZooKeeper para asegurar que la gente no intenta usarlo como un gran almacén de datos. ZooKeeper es, sin embargo, la mejor alternativa para compartir poca información en el entorno [62].

ZooKeeper permite que los procesos distribuidos se coordinen entre sí a través de un espacio de nombres jerárquico compartido de los registros de datos (llamados registros znodes), al igual que un sistema de archivos. A diferencia de los sistemas de archivos normales, ZooKeeper proporciona a sus clientes un alto rendimiento, baja latencia, acceso altamente ordenado y estrictamente accesible a los znodes. Los aspectos de rendimiento de ZooKeeper permiten su uso en grandes sistemas distribuidos. Los aspectos de confiabilidad le impiden convertirse en el único punto de falla en los grandes sistemas. Su ordenamiento estricto permite que las primitivas sofisticadas de sincronización sean implementadas en el cliente [97].

2.4.4. Distribuciones disponibles

Hadoop es distribuido a través de la Apache Software Foundation. Sin embargo muchas organizaciones desean implementar su plataforma tecnológica para tratar con grandes volúmenes de datos, en una cantidad gigantesca de procesos distintos e integrar Hadoop con un conjunto herramientas del ecosistema no siempre es una alternativa válida, por lo que hay compañías que distribuyen soluciones que proveen de Hadoop y un stack completo de herramientas integradas que cumplen con los requerimientos indispensables para llevar a cabo dichos procesos. Las tres principales soluciones (distribuciones) de Hadoop son, Cloudera Distribution Hadoop, Hortonworks Data Platform y MapR Distribution.

2.4.4.1. Cloudera

Cloudera distribuye una plataforma de proyectos de código abierto incluyendo la Distribución de Cloudera Apache Hadoop o CDH. Además, Cloudera ofrece a sus clientes empresariales una familia de productos y servicios que complementan la plataforma de Apache Hadoop. Estos incluyen sesiones de capacitación integral, servicios de arquitectura y soporte técnico para clústeres Hadoop en desarrollo o en producción. Sirven a una amplia gama de clientes incluyendo el comercio minorista, el gobierno, el servicio financiero, la salud, las ciencias biológicas, los medios digitales, la publicidad, las redes y las empresas de telefonía [98].

Para ello Cloudera provee no solo del Hadoop como núcleo de su solución sino un conjunto de herramientas del ecosistema 100 % código abierto que la convierten en la distribución más utilizada de Hadoop. Su Arquitectura se puede apreciar en la figura 2.20.

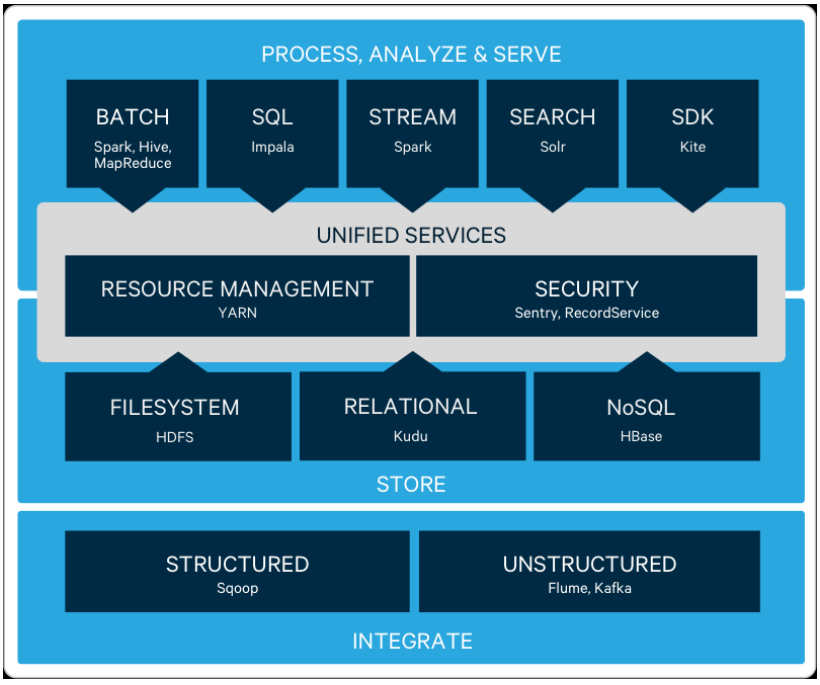

Figura 2.20: Arquitectura de Hadoop bajo la Distribución Cloudera

2.4.4.2. Hortonworks

Como uno de los más grandes distribuidores de Hadoop y de la mano de la comunidad, Hortoworks provee de una de las distribuciones de Hadoop más robustas y fáciles de instalar, administrar, usar, integrar y extender. Su distribución llamada Hortonworks Data Platform (HPD) es también completamente código abierto como su principal competidor, Cloudera. Hortoworks es provisto por soporte técnico, entrenamiento y servicios de terceros involucrados para las organizaciones de usuarios finales y proveedores de tecnología. Su arquitectura se puede observar en la figura 2.21.

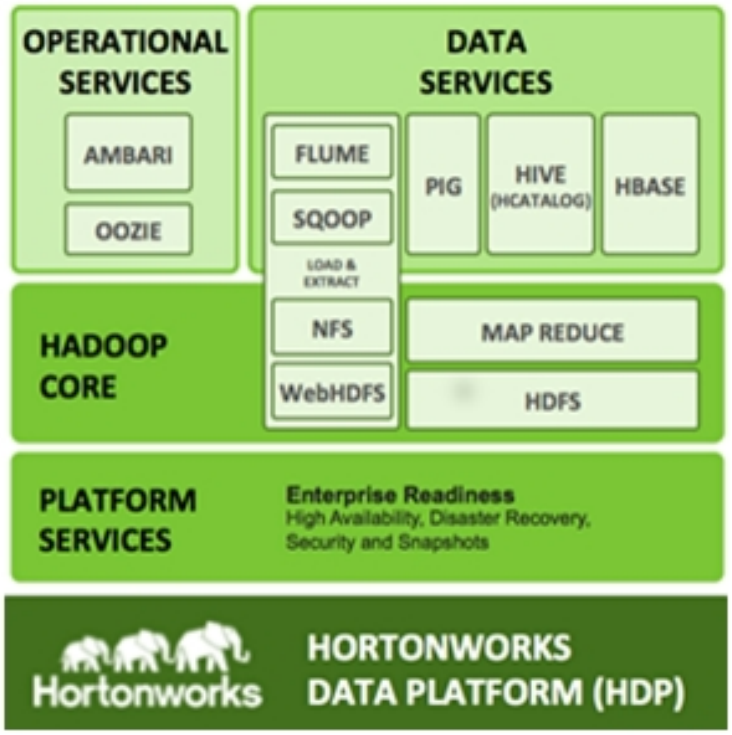

Figura 2.21: Arquitectura de Hadoop bajo la Distribución Hortoworks

2.4.4.3. MapR

MapR Technologies es una empresa de soluciones tecnológicas enfocada en el análisis de datos. Ellos tienen una distribución de Hadoop adaptada a las necesidades de ello, incluyendo herramientas de diseño propio que optimizan las tareas de almacenamiento, flujos de datos y su propia perspectiva de MapReduce. La distribución de MapR se llama MapR Convegered Data Platform.

MapR Converged Data Platform integra Hadoop, Spark y Apache Drill con capacidades de base de datos en tiempo real, streaming de eventos globales y almacenamiento empresarial escalable para alimentar una nueva generación de grandes aplicaciones de datos. Su arquitectura se puede ver en la figura 2.22.

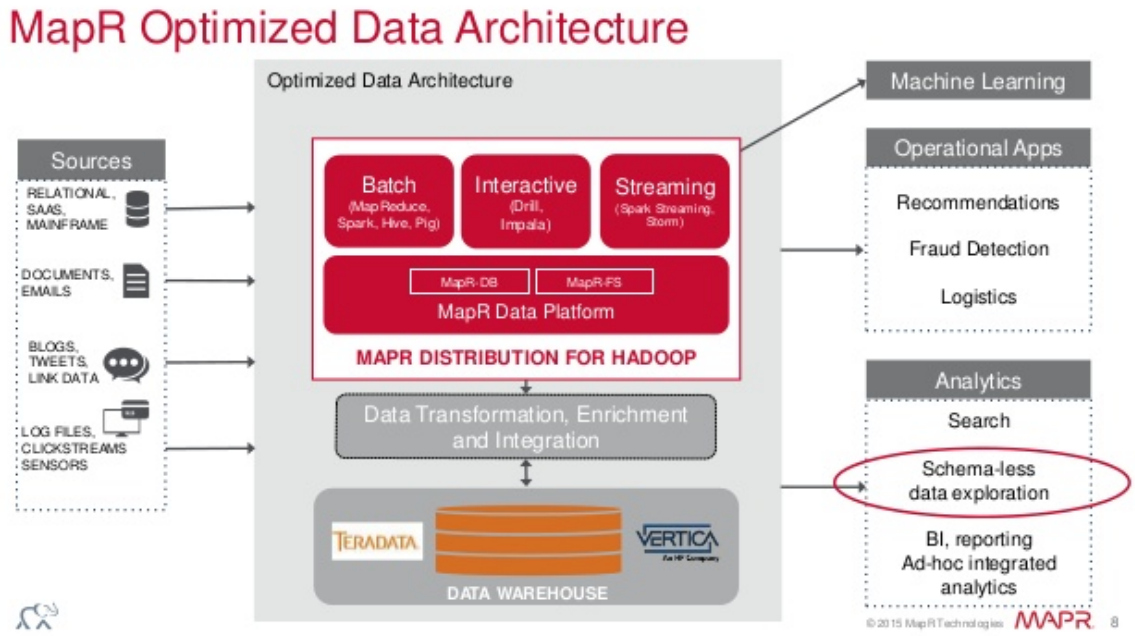

Figura 2.22: Arquitectura de Hadoop bajo la Distribución MapR

2.5. Internet de las Cosas

2.5.1. Definición

El término "Internet de las Cosas"(IoT por sus siglas en inglés) fue usado por primera vez en el año 1999 por el británico Kevin Ashton para describir un sistema en el que los objetos del mundo físico que podrían estar conectados al Internet por medio de sensores. Ashton acuñó el término para ilustrar el poder de conectar Etiquetas de Identificación por Radio Frecuencia (RFID) usadas en las cadenas de suministro corporativas a Internet para contar y rastrear mercancías sin la necesidad de intervención humana [99].

Hoy en día, a pesar de lo mediático que se ha vuelto el término, el concepto de Internet de las Cosas no es universal y existen diferentes definiciones utilizadas por distintas organizaciones para describir o promover su visión particular de lo que significa el Internet de las Cosas.

Para la Internet Engineering Task Force (IETF), el término "redes de objetos inteligentes"se utiliza comúnmente en referencia al Internet de las Cosas. En este contexto, los "objetos inteligentes"son dispositivos que típicamente tienen restricciones importantes, como energía, memoria y recursos de procesamiento o ancho de banda. El trabajo en el IETF se organiza en torno a requisitos específicos para lograr la interoperabilidad de la red entre varios tipos de objetos inteligentes.

La Recomendación UIT-T Y.2060 [100] de la Unión Internacional de Telecomunicaciones (UIT), *Descripción general de Internet de los objetos* analiza el concepto de interconexión, pero no vincula específicamente la IoT a Internet:

> *3.2.2 Internet de las cosas (IoT): una infraestructura global para la sociedad de la información que permite servicios avanzados interconectando cosas (físicas y virtuales) basadas en tecnologías de información y comunicación interoperables existentes y en evolución.*
>
> *Nota 1 - A través de la explotación de las capacidades de identificación, captura de datos, procesamiento y comunicación, el IoT hace pleno uso de las cosas para ofrecer servicios a todo tipo de aplicaciones, asegurando al mismo tiempo que se cumplan los requisitos de seguridad y privacidad.*
>
> *Nota 2 - Desde una perspectiva más amplia, el IoT puede percibirse como una visión con implicaciones tecnológicas y sociales.* [100]

Esta definición en una convocatoria de artículos para un tema temático de IEEE Communications Magazine vincula el IoT a los servicios en la nube:

> *Internet de las cosas (IoT) es un marco de desarrollo en el que todas las cosas tienen una representación y una presencia en Internet. Más específicamente, Internet de Cosas tiene como objetivo ofrecer nuevas aplicaciones*

y servicios que conecten los mundos físico y virtual, en el que las comunicaciones de Máquina a Máquina (M2M) representan la comunicación base que permite las interacciones entre Cosas y aplicaciones en la nube.

Todas las definiciones describen escenarios en los que la conectividad de red y la capacidad de cálculo se extienden a una constelación de objetos, dispositivos, sensores y elementos cotidianos que normalmente no se consideran computadoras. Esto permite a los dispositivos generar, intercambiar y consumir datos, a menudo con una mínima intervención humana. Las diversas definiciones de IoT no necesariamente están en desacuerdo, sino que enfatizan diferentes aspectos del fenómeno de IoT desde diferentes puntos focales y casos de uso [99].

2.5.2. Modelos de Comunicación

Desde la perspectiva operacional, parte fundamental de cómo opera el Internet de las Cosas depende del modelo utilizado para interconectar los dispositivos con el Internet o con otros dispositivos. En el RFC-7452 [101] se describe un marco de desarrollo con 4 modelos de comunicación y sus características claves.

2.5.2.1. Comunicación Dispositivo a Dispositivo

El modelo de comunicación dispositivo a dispositivo es la manera más simple de establecer pases de mensajes entre emisores y receptores. Un dispositivo se comunica directamente con uno o más dispositivos a través de algún protocolo de red, incluidos el protocolo TCP/IP, o los estándares Bluetooth, Z-Wave o ZigBee, como se muestra en la figura 2.23.

Figura 2.23: Modelo de Comunicación Dispositivo a Dispositivo

Este modelo de comunicación se utiliza comúnmente en aplicaciones como sistemas domóticos, que usualmente utilizan pequeños paquetes de datos de información para comunicarse entre dispositivos con necesidades de velocidad de datos relativamente bajas. Los dispositivos de IoT residenciales como bombillas, interruptores de luz, termostatos y cerraduras de puerta normalmente envían pequeñas cantidades de información entre sí en un escenario de domótica [99].

2.5.2.2. Comunicación Dispositivo a la Nube

En un modelo de comunicación de dispositivo a la nube, el dispositivo IoT se conecta directamente a un servicio en la nube conectándose a Internet con un proveedor de servicios de aplicaciones para intercambiar datos y controlar el tráfico de mensajes (véase figura 2.24). Este enfoque frecuentemente se aprovecha de los mecanismos de comunicación existentes, como las tradicionales conexiones por cable Ethernet o WiFi,

para establecer una conexión entre el dispositivo y la red, que finalmente se conecta al servicio en la nube. [99]

Figura 2.24: Modelo de Comunicación Dispositivo a la Nube

Este modelo también aprovecha la capacidad de conexión a la red para poder realizar tareas de monitoreo, administración y utilización de manera remota del dispositivo. En algunos casos este modelo ofrece un valor agregado al usuario final extendiendo las capacidades del dispositivo, más allá de las que le son nativas.

2.5.2.3. Modelo Dispositivo a Puerta de Enlace

También llamado modelo dispositivo a aplicación de capa de puerta de enlace, este modelo busca que los dispositivos IoT se conecten a los servicios en la nube haciendo uso de una puerta de enlace (como se puede ver en la figura 2.25), que actúa como intermediario y proporciona capacidades uso y traducción de mensajes de protocolos de red y seguridad.

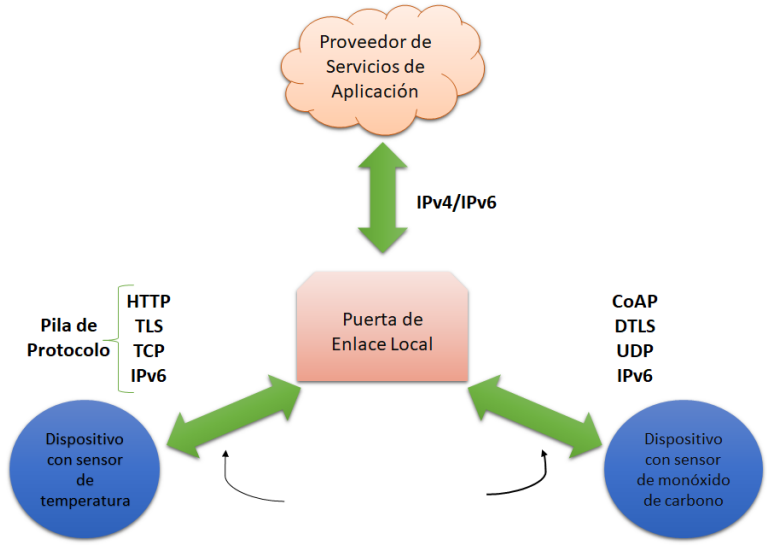

Figura 2.25: Modelo de Comunicación Dispositivo a la Puerta de Enlace Local

Varias formas de este modelo se encuentran en los dispositivos de consumo actuales. En muchos casos, el dispositivo de puerta de enlace local es un teléfono inteligente que ejecuta una aplicación para comunicarse con un dispositivo y retransmitir datos a un servicio en la nube. Éste es a menudo el modelo empleado con artículos populares de

consumo como las pulseras cuantificadoras. Estos dispositivos no tienen la habilidad nativa de conectarse directamente a un servicio en la nube, por lo que frecuentemente recurren a aplicaciones para teléfonos inteligentes o computadores para servir como una puerta de enlace para conectarse a la nube.

La otra forma de este modelo de dispositivo a puerta de enlace es la aparición de dispositivos "hub" en aplicaciones de domótica. Estos son dispositivos que sirven como una puerta de enlace local entre dispositivos IoT individuales y un servicio en la nube, pero también pueden superar la brecha de interoperabilidad entre los dispositivos mismos. Por ejemplo, el concentrador de objetos inteligentes es un dispositivo de puerta de enlace autónomo que tiene transceptores Z-Wave y ZigBee instalados para comunicarse con ambas familias de dispositivos. Un inconveniente de este enfoque es que el desarrollo necesario del software y el sistema de puerta de enlace de capa de aplicación añade complejidad y costo al sistema en general [99].

2.5.2.4. Modelo de Intercambio de Datos en Back-End

El modelo de intercambio de datos en back-end se refiere a una arquitectura de comunicación que permite a los usuarios exportar y analizar datos de objetos inteligentes de un servicio en la nube en combinación con datos de otras fuentes. Esta arquitectura es compatible con "el deseo" (del usuario) de conceder acceso a los datos de los sensores cargados a terceros. Una arquitectura de intercambio de datos en back-end permite que los datos recopilados de flujos desde los dispositivos IoT individuales sean agregados y analizados.

Por ejemplo, un usuario corporativo a cargo de un complejo de oficinas estaría interesado en consolidar y analizar los datos de consumo de energía y servicios públicos producidos por todos los sensores de IoT y sistemas de utilidad habilitados para Internet en las instalaciones. Una arquitectura de intercambio de datos en back-end eficaz permitiría a una organización acceder y analizar fácilmente los datos en la nube producidos por todo el espectro de dispositivos. Además, este tipo de arquitectura facilita las necesidades de portabilidad de datos. Las arquitecturas eficientes de intercambio de datos en back-end permiten a los usuarios mover sus datos cuando cambian entre los servicios de IoT, rompiendo las barreras tradicionales de los almacenes de datos.

El modelo de intercambio de datos en back-end sugiere que se necesita un enfoque de servicios en la nube de forma federada o interfaces de programación de aplicaciones en la nube (API) para lograr la interoperabilidad de los datos de dispositivos inteligentes alojados en la nube. En la figura 2.26 se muestra una representación gráfica de este diseño.

2.5.3. Computación en el Borde y Computación en la Niebla

La computación en el borde, también conocido por el termino en ingles de Edge Computing es un esquema de cómputo distribuido que se enfoca en descentralizar las aplicaciones informáticas, los datos y los servicios hacia el borde de la red, lo que permite que la generación de conocimiento y análisis se produzca más cerca de las fuentes

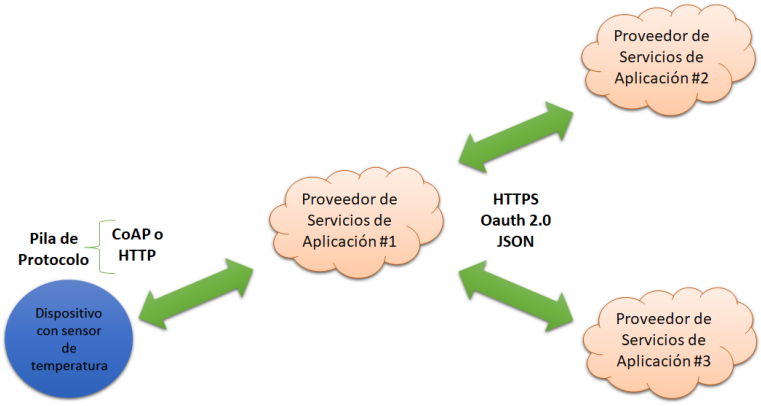

Figura 2.26: Modelo de Comunicación Back-End Compartido

de datos.

La computación en el borde permite aprovechar los recursos que podrían no estar conectados a una red de manera continua o con restricciones de ancho de banda, así como también permite crear interfaces entre nodos, con la idea de realizar tareas de preprocesamiento y almacenamiento antes de ser enviados a nodos que alojen servicios que requieren mayor poder de cómputo como tareas de inteligencia artificial y Big Data. El inconveniente de ésta estrategia reside en que los dispositivos, sensores y actuadores requieren de un poder de cómputo considerablemente mayor, lo que implica costos más altos de fabricación y mantenimiento.

Por otro lado la computación en la niebla (conocido con el anglicismo de Fog Computing) es otro esquema de cómputo distribuido que a diferencia de la computación en el borde trata de llevar el poder de procesamiento y almacenamiento de manera transparente y conveniente tomando en cuenta la características de los nodos a lo largo de toda la topología.

La computación en la niebla trata de aprovechar las verdaderas fortalezas que posea un determinado nodo, optimizando y balanceando la carga de la red, del procesamiento y del almacenamiento para mejores resultados globales. Sin embargo este tiene como desventaja fundamental la posibilidad de que múltiples nodos se conviertan en puntos de falla, afectando el funcionamiento de múltiples procesos delos sistemas involucrados.

2.5.4. Aplicaciones

El Internet de las cosas está llamada a ser la siguiente revolución tecnológica de la humanidad: la capacidad que los objetos cotidianos adquieran características de comunicación y nos permitan obtener y desplegar mayor cantidad de información. A su vez permite optimizar y automatizar procesos a una mayor escala, liberando a las personas de labores repetitivas y/o difíciles. Los campos de aplicación del Internet de las Cosas son el primer paso a una computación ubicua.

Podemos categorizar en términos del campo de acción, los diversos dispositivos existentes y las aplicaciones posibles que tendrán. En la tabla 2.3 veremos un resumen de

Dispositivos de Internet de las Cosas		
Uso	Descripción	Ejemplos
Personas	Dispositivos acarreados o implantados en el cuerpo humano	Dispositivos (tecnologías vestibles o digeribles) para monitorear y mantener la salud y el bienestar. Administración de tratamientos, incremento en desempeño del ejercicio físico
Hogar	Casas y edificios	Control de objetos y recursos del hogar y sistemas de seguridad
Ventas	Tiendas y distribuidores a minoristas	Tiendas, bancos, restaurantes, estadios, cualquier lugar donde un consumidor obtenga un producto o un servicio. Ofertas en tiendas, optimización de inventarios
Oficinas	Espacios de trabajo y de generación de conocimiento	Administración de recursos, seguridad en las oficinas. Productividad mejorada
Fabricas	Entornos de producción estandarizados	Lugares donde tengan rutinas repetitivas de trabajo, incluyendo granjas o lineas de producción. Operaciones eficientes, uso de equipamiento optimizado y de inventarios
Lugares de Construcción y extracción de material	Producción de materias primas	Infraestructuras en Minas, Petroleo y gas. Mantenimiento predictivo; seguridad laboral
Vehículos	Sistemas dentro de vehículos	Vehículos incluyendo carros, camiones, barcos, aviones y trenes. Mantenimiento basado en condiciones del sistema. Diseños basados en uso. Analítica de preventas

Tabla 2.3: Clasificación de Dispositivos de Internet de las Cosas según McKinsey Global Institute

estas configuraciones y algunos ejemplos de que dispositivos interconectados e inteligentes son referenciales a esos procesos [102].

2.5.4.1. Hogares Inteligentes

El hogar, es uno de los lugares donde la tecnología tiene un alto índice de adopción. Es común observar como los objetos en los hogares adquieren capacidades mayores gracias a la inclusión de características de conexión e interacción con las personas de forma bastante innovadoras, llevando a un nuevo nivel, las viviendas, que se les comienza a llamar inteligentes pues presentan la infraestructura tecnológica necesaria para que los dispositivos como electrodomésticos, línea blanca, iluminación, calefacción, centros de entretenimiento y mecanismos de seguridad se comuniquen entre sí, capturando y des-

plegando información, además de permitir en general monitorear lo que sucede en el hogar.

Las viviendas inteligentes permiten a los propietarios personalizar y controlar sus entornos domésticos para una mayor seguridad y un eficiente manejo de los recursos. Ya hay cientos de tecnologías IoT disponibles para monitorear y construir hogares inteligentes. Los fabricantes de productos de consumo como Belkin, Philips, Amazon y Haier se han establecido como empresas prominentes en este mercado [103]. Ya existen multitud de situaciones que aprovechan el poder del Internet de las Cosas en los hogares.

En lo que en la seguridad de los hogares se refiere el fuego es de lejos la principal causa de daños a la propiedad. Durante años, el pitido del detector de incendios ha sido la primera señal de humo en la casa, pero hay varios tipos de contaminantes que pueden amenazar nuestro entorno familiar y la calidad del aire, todo lo que podría conducir a daños materiales y daños a las personas en su interior. Un detector de dióxido de carbono mide los niveles de CO_2 en el aire y advertir a la gente si los niveles son peligrosos. Ya que el CO_2 es inodoro e indetectable sin ayuda, un detector puede ser un protector de la vida, sobre todo cuando se está conectado a un servicio de supervisión de emergencia. Algunos de los nuevos sensores no sólo detectan tanto humo y CO_2, sino también pueden controlar la calidad general del aire en su casa y captar contaminantes como polvo, hollín, polen, gases tóxicos, el estancamiento del aire, etc.

Un sensor de detección de humedad y/o de flujo puede ayudar en caso de una tubería de agua rota o atendiendo fenómenos similares. Estos sensores capaces de alertar sobre posibles fugas o filtraciones en su vivienda ayudando a solucionar el problema de manera temprana y no después de que alcance un nivel de daño mayor. Los sensores se pueden integrar o colocar alrededor de calentadores de agua, lavavajillas, refrigeradores, fregaderos, bombas de suministro y cualquier artefacto con riesgo de fugas. Si algún sensor detecta un flujo de líquido anómalo se envía una notificación al usuario, para que pueda revisar el problema.

Por otro lado, siempre se busca obtener el mayor confort posible en nuestros hogares y automatizando procesos podemos dar un paso más en esa búsqueda. El uso de sensores en puertas y ventanas permite saber cuándo están abiertas o cerradas, cuando la gente está entrando y saliendo de su vivienda e incluso puede encender las luces y apagarlas. Estos sensores sirven también como la primera línea de defensa para el hogar. Algunos sensores detectan incluso cuando una ventana se rompe por un intruso, alertando de una situación de robo. Una vez más, la tecnología inalámbrica le permite recibir notificaciones directamente a su teléfono o tableta y le permite alertar a servicios policiales automáticamente.

Otros dispositivos que permiten ampliar la seguridad son los sensores de movimiento. Estos sensores están de guardia cuando el usuario no está en su vivienda. Pueden avisar la existencia de movimiento dentro de su casa. Los sensores de movimiento se convierten en un par de ojos extra para los dueños de casa, alertando de actividad no deseada en las habitaciones, o si un niño entra en una zona donde el acceso sea

restringido o que requiere de supervisión, tales como un botiquín de primeros auxilios o una piscina. Los sensores de movimiento también se pueden conectar a vídeo, por lo que no sólo se puede conseguir una notificación de que un sensor se ha disparado, sino que también el sensor puede activar la grabación de vídeo para capturar imágenes de la intrusión [104].

Los sensores de movimiento también son excelentes para ayudar a ahorrar energía. Estos sensores pueden estar conectados a la iluminación o el termostato para ayudar a controlar el consumo de energía en una habitación sobre la base de la ocupación de la sala. Por ejemplo, apagar las luces si no hay nadie en la habitación o ajustar a una temperatura eficiente para ahorrar la energía cuando la habitación está desocupada. Con un termostato inteligente no solo se puede mantener las habitaciones frescas, sino que también va a ayudar a ahorrar dinero mediante el control de la temperatura y la humedad dentro su hogar en función de su comportamiento y el uso de la sala y poder por defecto entrar a un modo de ahorro de energía cuando no hay nadie en la habitación. La aplicación de la tecnología cognitiva de estos sensores está allanando el camino para una casa que piensa y conoce sus preferencias.

2.5.4.2. Tecnología Vestible

La tecnología vestible son una de las tendencias que más repercusiones ha tenido en el campo del Internet de las Cosas y una de las más tempranas en adopción. Es un dominio muy grande que consiste en muchos dispositivos, que abarcan el ejercicio, la salud y el entretenimiento. Los prerrequisitos primarios para la tecnología IoT vestible son que los sensores o dispositivos son altamente eficientes en consumo energético, potencia baja y de tamaño pequeño [103].

Los primeros dispositivos de tecnologías vestibles, han sido las pulseras cuantificadoras. Estas permiten recolectar información sobre nuestras rutinas de ejercicio (ritmo cardíaco, calorías consumidas, distancia recorrida), hábitos de sueño, entre otros indicadores propios del área del cuidado de la salud. Otro dispositivo que también ha evolucionado en este sentido han sido los relojes inteligentes, que agregando la capacidades anteriores, también pueden desplegar en pantalla no solo la hora, sino información de aplicaciones de teléfonos inteligentes, notificaciones entrantes, ubicación y también permitir el control de otros dispositivos a través de movimiento o gestos.

Otra área de las tecnologías vestibles son aquellas asociadas a la ropa, agregando sensores de movimiento y que capten la temperatura corporal que permiten cuantificar información para múltiples aplicaciones. En el área deportiva, la ropa puede ayudar seguir los movimientos de los jugadores durante el entrenamiento o el juego ayudando a los entrenadores y a los preparadores físicos a lograr el mejor rendimiento, evitar lesiones y mejorar estrategias deportivas. En la salud estos mismos sensores en la ropa pueden ser utilizados para la medición de constantes vitales en tiempo real de los pacientes.

2.5.4.3. Comercio

Para el comercio, el Internet de las cosas representa una oportunidad de optimizar y mejorar el proceso de ventas. Al permitir el monitoreo y automatización de la cadena de suministros se pueden bajar los costos por un mejor uso de materias primas, así como también automatizar el proceso de pedidos en base a la existencia.

A nivel logístico también el Internet de las Cosas representa un avance importante pues ahora permite el rastreo de bienes, el intercambio de información de inventario en tiempo real entre proveedores y minoristas y capacidades de entrega automatizadas [103] agregando componentes de seguridad y auditorias en el servicio.

2.5.4.4. Ciudad Inteligente

Los servicios de las ciudades son el campo perfecto de aplicación de muchas de las tecnologías más innovadoras del Internet de las Cosas. Las ciudades compiten por los residentes, inversores, turistas, e incluso la financiación del gobierno central. Para que las ciudades sigan siendo relevantes, tienen que ser más inteligentes y más conectadas. El IoT está ayudando a las ciudades más grandes del mundo para hacer esto, a gran escala, y a un ritmo fenomenal. [105] La gestión de recursos críticos como el suministro de agua, electricidad y gas, mejorar el tráfico sugiriendo rutas alternas para el transporte público y privado, las medición de la contaminación son algunos de los campos donde dispositivos pueden ayudar a mejorar la calidad de vida de los ciudadanos.

Para lograr un adecuado uso de los recursos, cada vez más encontraremos que las compañías de servicios contaran con redes inteligentes de medidores que puedan dar información del consumo y el mantenimiento necesario en las infraestructuras, que a su vez permiten la distribución del mismo en la ciudad, pudiendo direccionar automáticamente los mismos donde más se requieren. A nivel ambiental un sistema de recolección de desperdicios, reciclaje y tratamiento de aguas residuales, en conjunto a una red de sensores que midan a calidad de agua y del aire garantizamos un espacio ecológicamente agradable, además de ayudar al gestionar de mejor manera la recolección de residuos y evitando su desbordamiento.

Del punto de vista de seguridad encontraremos un gran conjunto de cámaras de circuito cerrado, de micrófonos ambientales, junto con los sistemas de análisis permitirán crear mejores planes de patrullaje inteligente para evitar el crimen.

Los estudios han demostrado que los sistemas de estacionamiento inteligentes pueden reducir la congestión de estacionamiento en hora pico hasta en un 22 % y pueden reducir el volumen total de tráfico en un 8 %. Con otras tecnologías, los gobiernos locales podrían utilizar dispositivos para recopilar datos en tiempo real para medir el volumen de tráfico, la velocidad y otros parámetros. Esta información podría ser utilizada para mejorar el diseño de políticas y planificación de la ciudad [105].

2.5.4.5. Medicina y Cuidado de la Salud

La supervisión inteligente de signos vitales ayuda a evitar en los pacientes infecciones, el tratamiento temprano de condiciones médicas como diabetes, cánceres, insuficiencias cardíacas, entre otras [103]. Estos dispositivos también nos ayudaran a llevar un historial médico y clínico mas detallado con la información de los sensores de los dispositivos de las tecnologías vestibles o de prótesis.

También podemos ver un avance en la medición de la efectividad de tratamientos médicos en tiempo real a su vez de la tendencia cada vez mayor a tratamientos personalizados. Además la conectividad del sistema de atención de la salud a través de la Internet de las Cosas, hace hincapié en las necesidades del paciente, es decir, los tratamientos proactivos, precisión mejorada cuando se trata de diagnóstico, la intervención oportuna por parte de los médicos y los resultados del tratamiento mejoradas como resultado responsables de la atención que está muy de confianza entre los pacientes [106].

La recopilación precisa de los datos, flujos de trabajo automatizados combinados con toma de decisiones apoyados por los datos son una excelente manera de reducir la pérdida de recursos, lo que reduce los costes del sistema y lo más importante la minimización de errores.

2.5.4.6. Agricultura

La calidad de los productos, mayor productividad de los cultivos, conservación de recursos y el control de costo, son sólo algunas de las formas en que Internet de las Cosas promete transformar la agricultura y producción de alimentos en el futuro, ayudando a aumentar la productividad de los cultivos mediante la gestión y el control de dichas actividades [107] las siguientes maneras:

- Suministro del Agua: El IoT tiene el potencial para ahorrar 50 mil millones de galones de agua dulce en un año. El suministro adecuado de agua es esencial para la agricultura; cultivos pueden ser dañados por cualquiera de exceso de agua o escasez. El IoT en el área agrícola, integrado con Servicios de Mapeo Web (WMS por sus siglas en ingles) y el Servicio de Observación de Sensores (SOS) ofrece una solución para la gestión de las necesidades de agua para el riego o abastecimiento de cultivo. También se puede analizar inteligentemente las necesidades de agua de los cultivos y utiliza la cantidad disponible de suministro para reducir la perdida innecesaria.

- Agricultura de Precisión: La exactitud en la predicción del tiempo y otras entradas de datos dinámicos pueden afectar a la productividad de los cultivos en gran medida. Cuanto mayor sea el nivel de precisión, menores son las posibilidades de que los cultivos se dañen; por lo tanto, las previsiones meteorológicas más precisas pueden dar lugar a mayores niveles de rentabilidad y productividad. Se garantiza una comunicación precisa y eficaz a los agricultores de los datos en tiempo real relacionados con los procesos agrícolas dinámicos (como las previsiones meteorológicas, la siembra, la cosecha, etc), la calidad del suelo, y la disponibilidad y costo de mano de obra.

- Gestión y Control Integrada de Plagas: El trabajo de un agricultor a menudo es destruido por las plagas, causando pérdidas económicas significativas. Para evitar este tipo de situaciones, el Internet de las Cosas puede ser parte de un sistema para monitorear y analizar los parámetros ambientales y crecimiento de las plantas. También hay datos disponibles de los sensores de control de plagas que son capaces de predecir el comportamiento de las mismas.
- La administración del Ganado: Apoyando la salud del ganado y su cuidado con herramientas de monitoreo como etiquetas de oreja para ganado vacuno, capaces de detectar algunas enfermedades. Si se detecta una enfermedad, se envía una alerta para que el animal se puede separar de la manada, y prevenir la propagación de la enfermedad.

2.5.4.7. Transporte

La congestión, impulsado principalmente por los conductores dentro y en los alrededores de las ciudades, es un gran problema económico costando los EE.UU 160 mil millones de dólares por año según investigadores de la Universidad de Texas A & M. Este es un problema mundial, que afecta a menudo a las ciudades en crecimiento en que la mayoría de los casos la infraestructura no puede seguir el ritmo de la urbanización. La urbanización misma es a menudo citada como la causa raíz de estos problemas. Según el Fondo para la Población de las Naciones Unidas, más de la mitad de la población mundial vive en pueblos y ciudades y en 2030 esta cifra aumentará a alrededor de 5 mil millones [108].

Una manera en que se verá afectado el tráfico es gracias al uso de los vehículos conectados y ya es considerado un catalizador para el cambio. Todavía no se han convertido en el elemento básico en la vida de los consumidores, pero los nuevos servicios conectados que nacen todos los días harán más atractivo el uso de estas tecnologías.

Los automóviles tendrán la capacidad de diagnosticarse a sí mismos y saber cuándo necesitan reparaciones y actualizaciones. También se adaptaran a nuestros hábitos de conducción, determinando cómo el conductor se siente y la profundidad de su habilidad general, de modo que puedan ajustarse para mantener a sus ocupantes confortables y más seguros al conducir de manera más eficiente [109].

Para muchos habitantes de las ciudades, los sistemas ferroviarios de los centros urbanos son el principal medio de transporte. Los sistemas de metro dan forma a la estructura misma de las propias ciudades. Mantener un funcionamiento fluido es particularmente difícil para algunos de los sistemas de transporte más antiguos, algunos de los cuales datan de hace cientos de años. El metro de Londres, por ejemplo, se remonta a 1863 y es el sistema de ferrocarril subterráneo más antiguo del mundo. Pero de ninguna manera es anticuado, todo lo contrario. El metro de Londres está utilizando dispositivos IoT para transformar la forma en que opera. El sistema obtiene información en tiempo real de los trenes y estaciones que permite la programación eficiente y el mantenimiento proactivo de la infraestructura. Todos los activos son administrados en un sistema centralizado, incluyendo los trenes, flotas, pistas, señales e incluso insta-

laciones. Esta consolidación de la gestión de activos a mejorará el tiempo de actividad de trenes, estaciones y mejorando la satisfacción de los pasajeros. Esta estrategia está siendo replicada en sistemas ferroviarios al rededor del mundo.

2.5.4.8. Industria

La industria 4.0 tiene que ver con el aumento de la conectividad de los componentes cada vez más pequeños. Utiliza los datos recogidos por las cosas físicas, ya sea incorporado en productos, o en componentes dentro de una máquina en la línea de producción, para impactar cómo estas cosas mismas se fabrican. Cuanta más información que se puede obtener por estos objetos, más eficaz el proceso de fabricación puede llegar a ser.

La idea es que se crea un bucle de retroalimentación entre las cosas físicas y el mundo digital. Las cosas mismas nos pueden decir cómo están siendo utilizados, o cuando están en necesidad de reparación, dando los fabricantes la información que puede ayudar a optimizar todas las etapas del proceso de producción, desde el suministro, el desarrollo y la creación, ahorrando tiempo y dinero [110].

El poder aprovechar la información de ventas al por menor puede dar en tiempo real información sobre qué productos o componentes de productos se venden mejor. Si un artículo se vende con características adicionales opcionales, podemos profundizar en los datos disponibles para saber qué características son las más populares y las que están produciendo mayor nivel de satisfacción en el consumidor en la actualidad por lo que se puede preparar la producción para anticipar la demanda de un producto nuevo con características similares.

Las cadenas de bloque (mejor conocidas por el anglicismo Blockchain), serán claramente una tecnología fundamental para el seguimiento de piezas y procesos a través de la cadena de suministro y la vida útil de artefactos y esto es gracias a la inclusión de sensores y dispositivos IoT que permiten monitorear continuamente el estado actual de la cadena.

2.5.5. Interoperatibilidad entre Infraestructuras y Dispositivos

En el Internet, la interoperabilidad es el valor más básico. El primer requisito de conectividad a Internet es que los sistemas "conectados" puedan "hablar el mismo lenguaje" y esto se logra gracias a la implementación de protocolos y codificaciones estándares. Los denominados "jardines amurallados", en los que se permite a los usuarios interoperar sólo con un subconjunto establecido de sitios y servicios, pueden disminuir sustancialmente los beneficios sociales, políticos y económicos del acceso a Internet en su totalidad [99].

En un entorno completamente interoperable para cualquier dispositivo IoT podría conectarse a cualquier otro dispositivo o sistema e intercambiar información según lo desee. En la práctica, la interoperabilidad es más compleja. La interoperabilidad entre

dispositivos IoT y sistemas ocurre en grados variables en diferentes capas dentro de la pila de protocolos de comunicaciones entre los dispositivos. Además, la plena interoperabilidad entre todos los aspectos de un producto técnico no siempre es factible, necesaria o deseable. La estandarización y adopción de protocolos que especifican estos detalles de comunicación, incluyendo dónde es óptimo tener estándares, están en el corazón de la discusión de interoperabilidad para IoT.

Más allá de los aspectos técnicos, la interoperabilidad tiene una influencia significativa en el impacto económico potencial de IoT. La interoperabilidad de dispositivos bien funcionados y bien definidos puede fomentar la innovación y proporcionar licencias a los fabricantes de dispositivos IoT, aumentando el valor económico general del mercado. Además, la aplicación de las normas existentes y el desarrollo de nuevos estándares abiertos, cuando sea necesario, ayudan a reducir las barreras de entrada, facilitan nuevos modelos de negocio y generan economías de escala.

Además, la interoperabilidad es fundamentalmente valiosa desde la perspectiva tanto del usuario individual y final como a las organizaciones quienes las regulan o las usan. Facilita la posibilidad de elegir dispositivos con las mejores características al mejor precio e integrarlos para que trabajen juntos. Los compradores pueden ser reacios a adquirir productos y servicios IoT si hay inflexibilidad en la integración, alta complejidad de uso, la propiedad final de información, preocupación por el bloqueo del proveedor o temor a la obsolescencia debido al cambio de los estándares [99].

2.5.5.1. Ecosistemas Propietarios y Elección del Consumidor

Algunos fabricantes de dispositivos ven una ventaja en el mercado de crear un ecosistema propietario de productos IoT compatibles, a veces denominados “jardines amurallados”, que limitan la interoperabilidad sólo a aquellos dispositivos y componentes dentro de la línea de productos de la marca. Los partidarios de la interoperabilidad ven estas prácticas como un impedimento para la elección de los usuarios, ya que obstaculiza que los mismos cambien a productos alternativos. También ven esta práctica como una barrera para la innovación y la competencia porque limita la capacidad de los competidores para crear nuevos productos basados en la infraestructura fundamental del ecosistema. Sin embargo, algunos fabricantes de dispositivos ven un enfoque de ecosistema cerrado como un beneficio para los usuarios al proporcionar un protocolo que se puede adaptar más rápida y fácilmente cuando las demandas técnicas o del mercado requieren un cambio.

Las consideraciones de interoperabilidad también se extienden a los datos recogidos y procesados por los servicios de IoT. Uno de los principales atractivos de los dispositivos conectados es la capacidad de transmitir y recibir datos a los servicios en la nube, que a su vez proporciona información valiosa o servicios basados en esos datos. Si bien esto es extremadamente útil, también puede presentar desafíos para un usuario que quiere pasar a un servicio competidor. Incluso si el acceso a los datos generados por los dispositivos se pone a disposición de los usuarios, la obtención de los datos será inútil si están en un formato propietario [99]. Sólo si los datos de origen están disponibles libremente en un formato estándar abierto, los usuarios tendrán libertad para trasladarse a otro

proveedor de servicios o realizar análisis por sí mismos.

2.5.5.2. Restricciones Técnicas y de Costos

A medida que los fabricantes desarrollan dispositivos IoT, existen restricciones técnicas inherentes, tiempo a mercado y costos que influyen en la interoperabilidad y el diseño del dispositivo. A corto plazo, puede resultar más costoso diseñar características de interoperabilidad en un producto y comprobar si cumple con una especificación de estándares. En algunos contextos, el camino más barato al mercado puede ser el uso de protocolos y sistemas propietarios. Sin embargo, esto debe compararse con las ganancias del ciclo de vida del producto a largo plazo que ofrece la interoperabilidad.

2.5.5.3. Riesgo Programado

En un mercado globalmente competitivo, a menudo hay una ventaja de primer plano para llevar un producto al mercado rápidamente y establecer cuota de mercado, y esto ciertamente se aplica a los fabricantes de dispositivos IoT. Se plantea un problema para la interoperabilidad del dispositivo IoT cuando el diseño de producto del fabricante del dispositivo supera la disponibilidad de los estándares de interoperabilidad.

2.5.5.4. Riesgo Técnico

Cuando un fabricante o usuario de un dispositivo IoT está planificando el desarrollo de un producto, debe evaluar los riesgos de diseño técnico de los protocolos en el proceso de desarrollo. La incorporación de estándares existentes y probados en diseños de productos o sistemas puede representar un menor riesgo técnico en comparación con el desarrollo y uso de protocolos propios. El uso de estándares genéricos, abiertos y ampliamente disponibles puede traer ciertos beneficios, como el acceso a grupos más grandes de talento técnico, software desarrollado y costos de desarrollo más baratos.

2.5.5.5. Funcionamiento en los Dispositivos

La falta de normas y buenas prácticas documentadas tienen un impacto mayor que limitar el potencial de los dispositivos IoT. De manera pasiva, la ausencia de estos estándares puede permitir el mal comportamiento de los dispositivos. En otras palabras, sin estándares para guiar la manufactura, los desarrolladores de estos dispositivos a veces diseñan productos que operan de manera perjudicial en Internet sin tener en cuenta su impacto real.

2.5.5.6. Sistemas Heredados

La normalización de la interoperabilidad es un reto para los nuevos dispositivos IoT que necesitan interactuar con los sistemas ya implantados y en funcionamiento. Esto es relevante para muchos entornos específicos de la industria y aplicaciones específicas. Los ingenieros que desarrollan dispositivos IoT se enfrentan a los inconvenientes del diseño para mantener la compatibilidad con los sistemas heredados, al mismo tiempo que tratan de lograr una mayor interoperabilidad con otros dispositivos mediante el uso de estándares establecidos y probados.

2.5.5.7. Configuración

Los usuarios se enfrentarán a mayores desafíos en la gestión de un mayor número de dispositivos IoT. Uno de estos desafíos es la necesidad de modificar rápida y fácilmente los ajustes de configuración de muchos dispositivos en una red. Cuando se enfrenta a la perspectiva desalentadora de configurar cientos de dispositivos individuales, será esencial disponer de herramientas de configuración, métodos e interfaces para la tarea.

2.5.5.8. Proliferación de Esfuerzos por Estándares

Muchas nuevas coaliciones industriales han surgido junto con las tradicionales Organizaciones de Desarrollo de Normas para aumentar los esfuerzos para evaluar, desarrollar, modificar o armonizar las normas y protocolos relacionados con el IoT. El tiempo y la inversión requeridos por la industria y otras partes interesadas para participar en la amplia gama de esfuerzos de estandarización probablemente será costoso. Además, es probable que haya superposición e incluso estrategias de estandarización contradictorias entre algunos de los esfuerzos. Además de aumentar los costos de desarrollo de estándares, la ausencia de coordinación entre esfuerzos podría producir protocolos en conflicto, retrasar la implementación del producto y llevar a la fragmentación a través de los productos y servicios verticales de la industria.

2.5.6. Protocolos Utilizados

Para obtener el mayor potencial del Internet de las Cosas deben existir acuerdos en la manera en que estos deben funcionar y poder comunicarse. Varias "cosas"siempre han estado en Internet, y las computadoras de propósito general, centros de datos y hogares suelen ser capaces de usar los protocolos de Internet como se han definido para ellos. Sin embargo, hay un valor considerable en la extensión de Internet a dispositivos más restringidos que a menudo necesitan versiones optimizadas o uso especial de estos protocolos [111].

Siguiendo los modelos de comunicación sugeridos para tratar con dispositivos IoT en el RFC-7452 [101], para el correcto diseño o uso de un protocolo, se debe tener en cuenta en que escenario y datos podrán transmitirse. En general para el Internet de las Cosas los dispositivos deben comunicarse entre sí (Dispositivo a Dispositivo o D2D). Los datos del dispositivo a continuación, deben ser recogidos y enviados a la infraestructura de servidores (ya sea a través del modelo Dispositivo a la Nube o D2C o través del modelo Dispositivo a Puerta de enlace o D2G). Esa infraestructura de servidores también puede compartir los datos del dispositivo (Modelo de Intercambio de datos en Back-End), posiblemente proporcionándolos a otros dispositivos, a los programas de análisis, o para las personas [112].

A continuación se presentan algunos de los protocolos más importantes empleados por los dispositivos del Internet de las Cosas.

2.5.6.1. Protocolo HTTP

El protocolo de transferencia de hipertexto o mejor conocido como HTTP, es un protocolo de la capa de aplicación para la trasmisión de documentos hipermedia, como HTML. Fue diseñado para la comunicación entre los navegadores y servidores web, aunque puede ser utilizado para otros propósitos también. Se trata de un protocolo sin estado, lo que significa que el servidor no guarda ningún dato (estado) entre dos peticiones. Aunque la mayoría de los casos se basa en una conexión del tipo TCP /IP, pude usarse con cualquier otro protocolo del nivel de la capa de transporte orientado a conexión [113].

Las principales características clave del protocolo son las siguientes:

- Sencillez: HTTP está pensado y desarrollado para ser leído y fácilmente interpretado por las personas, haciendo de esta manera más fácil la depuración de errores, y reduciendo la curva de aprendizaje para las personan que empieza a trabajar con él.
- Extensible: Presentadas en la versión HTTP/1.0, las cabeceras de HTTP, han hecho que este protocolo sea fácil de ampliar y de experimentar con él. Funcionalidades nuevas pueden desarrollarse, sin más que un cliente y su servidor, comprendan la misma semántica sobre las cabeceras de HTTP.
- Basado en sesiones, sin estados: HTTP es un protocolo sin estado, es decir, no guarda ningún dato entre dos peticiones en la misma sesión. Esto plante la problemática, en caso de que los usuarios requieran interactuar con determinadas páginas Web, de forma ordenada y coherente. Pero mientras HTTP ciertamente es un protocolo sin estado, el uso de HTTP cookies, si permite guardar datos con respecto a la sesión de comunicación. Usando la capacidad de ampliación del protocolo HTTP, las cookies permiten crear un contexto común para cada sesión de comunicación.
- Orientado a Conexión: Una conexión se gestiona al nivel de la capa de trasporte, y por tanto queda fuera del alcance del protocolo HTTP. Aún con este factor, HTTP no necesita que el protocolo que lo sustenta mantenga una conexión continua entre los participantes en la comunicación, solamente necesita que sea un protocolo fiable, o que no pierda mensajes (como mínimo, en todo caso, un protocolo que sea capaz de detectar que se ha pedido un mensaje y reporte un error).

HTTP es un protocolo que funciona en dos fases fundamentales, una fase de petición y una fase de respuesta. Esto se hace de la siguiente manera:

- En la fase de petición:

    1. Se abre una conexión: Se utiliza un protocolo de conexión, generalmente TCP, que se usará para hacer una o una serie de peticiones y recibir una respuesta. El cliente puede abrir una conexión nueva, reusar una existente, o abrir varias a la vez hacia el servidor.

2. Se hace una petición HTTP: Los mensajes HTTP (previos a HTTP/2) son legibles en texto plano. A partir de la versión del protocolo HTTP/2, los mensajes se encapsulan en franjas, haciendo que no sean directamente interpretables, aunque el principio de operación es el mismo.

- En la fase de respuesta:

1. Se recibe y lee la respuesta enviada por el servidor.
2. Se cierra o recicla la conexión para futuras peticiones.

Existen dos tipos de mensajes HTTP: peticiones y respuestas, cada uno sigue su propio formato.

- Petición HTTP: Indican una acción a realizarse sobre un recurso determinado. Hace uso de un verbo o método (GET, POST, PUT, DELETE o HEAD) que define la operación que el cliente desea realizar, la dirección del recurso pedido en formato de URL, la versión del protocolo y el cuerpo del mensaje. Opcionalmente puede llevar cabeceras que pueden aportar mas información.

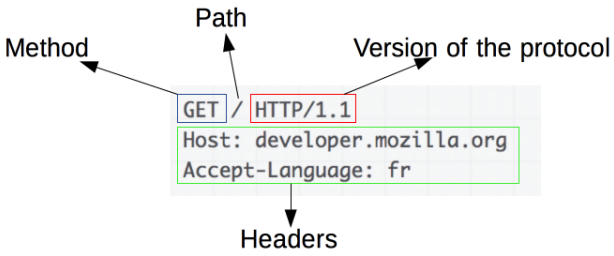

Figura 2.27: Formato de un Mensaje de Petición HTTP

- Respuesta HTTP: Formados por los campos de versión del protocolo, el código del estado indicando si la petición fue exitosa o no y el porqué, una descripción del estado, cabeceras HTTP y por último y dependiendo de lo requerido, un recurso.

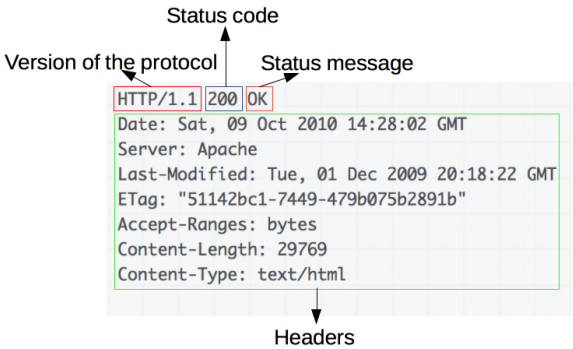

Figura 2.28: Formato de un Mensaje de Respuesta HTTP

2.5.6.2. Protocolo CoAP

El protocolo CoAP es una iniciativa de la Internet Engineering Task Force (IETF) para el creciente aumento de dispositivos de Internet de las Cosas. CoAP (siglas de Constrained Application Protocol) es un protocolo a nivel de capa de aplicación, que provee un modelo de solicitud/respuesta uno a uno con capacidad para el envío multicast (aun en etapa de estandarización) y nace del RFC 7252 [114] y cuya cabecera (figura 2.29) es mas pequeña que la utilizada por el protocolo HTTP

Este protocolo está pensado para el uso de dispositivos de baja potencia, a través de un modelo simple, con una cabecera pequeña. Posee la versatilidad de trabajar junto con el protocolo HTTP y con APIs REST. Las características principales del protocolo son:

- Soporte nativo para UDP: CoAP puede utilizar UDP en aplicaciones donde la información puede mostrar pocos cambios entre los paquetes emitidos. Esto también ayuda a que sea menor el consumo que posee el dispositivo al no tener que retransmitir los paquetes y además genera menos overhead dentro de la conexión.
- Soporte para envío de paquetes multicast: CoAP trabaja con un modelo dispositivo a dispositivo, sin embargo también permite el envío multicast de paquetes. Esta característica aun sigue en desarrollo y se apoya en el uso de IPv6. Es importante destacar que el envío de paquetes multicast puede repercutir de manera negativa en el consumo energético de los dispositivos.
- Seguridad: CoAP utiliza el protocolo DTLS para proporcionar seguridad sobre la capa de transporte, independientemente si se usa TCP o UDP.
- Recursos y Servicios independientes: CoAP también hace uso de URIs para la interacción entre los nodos de la red. Esto permite un alto grado de autonomía en los paquetes enviados al dispositivo. Las URI pueden definirse de forma que un recurso o servicio represente un sensor o un actuador y como tal, adaptar el envío y recepción de paquetes de forma que se haga un consumo eficiente tanto de la energía utilizada en el proceso como en la comunicación.
- Comunicación asíncrona: Los paquetes en el protocolo son enviados bajo un esquema de petición respuesta en el protocolo. Sin embargo CoAP también permite un mecanismo simplificado de "observación"similar al paradigma de publicación-suscripción de MQTT que permite a los nodos observar publicaciones en otros nodos, sin participar activamente en la comunicación.

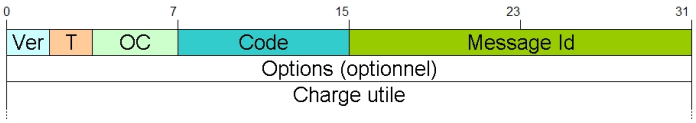

Figura 2.29: Formato de Mensaje en CoAP

CoAP también presenta una serie importante de retos para su adopción masiva. El principal reto es que CoAP es un estándar reciente y aun en evolución, que como tal debe

pelear con otras alternativas ya establecidas como MQTT por los nichos de mercado. También esta evolución continua representa una dificultad a nivel de interoperabilidad, sobre todo, al momento de trabajar con sistemas ya implantados. Por otro lado, otro punto débil de CoAP es la fiabilidad de la comunicación, pues provee de una manera muy simple de confirmación o no de los paquetes, quienes solo reciben acuses de recibo, pero sin que este confirme que el contenido este correcto o no este corrupto [115].

2.5.6.3. Protocolo MQTT

MQTT son las siglas de Message Queue Telemetry Transport. Como su nombre indica, su propósito principal es la telemetría, o el monitoreo remoto. Su objetivo es recopilar datos de muchos dispositivos y transportarlos a la infraestructura de se servicios en red. Se dirige a grandes redes de pequeños dispositivos que necesitan ser monitoreados o controlados desde la nube [112].

IBM creo el protocolo MQTT para poder establecer comunicación vía satélite con sensores en campos petroleros. Fue diseñado para que las comunicaciones a través de este protocolo fuesen confiables y utilizando la menor cantidad de energía posible. MQTT utiliza un modelo de publicación-suscripción y requiere un corredor MQTT central (Broker) para administrar y direccionar mensajes entre los nodos de una red MQTT (figura 2.30.). Utiliza TCP en la capa transporte, que se caracteriza por ser confiable, ordenado y con comprobación de errores [115] y su cabecera se puede ver en la figura 2.31.

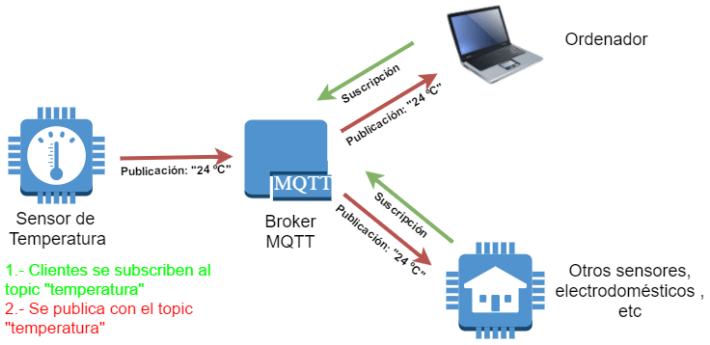

Figura 2.30: Diagrama de Funcionamiento del Protocolo MQTT

El protocolo MQTT tiene como fortalezas:

- Modelo de publicación-suscripción: El modelo de publicación-suscripción es un modelo que no tiene problemas al escalar la cantidad de nodos en la red y además hace de un uso eficiente de la energía. Los Brokers y los nodos publican información y otros se suscriben según el contenido, el tipo o el tema del mensaje. Generalmente el Broker se suscribe a todos los mensajes y luego administra el flujo de información a sus nodos.

- Seguridad: Al utilizar TCP como protocolo en la capa de transporte, Los paquetes MQTT no están encriptados por defecto. Pero puede utilizar la seguridad de Internet TLS/SSL. TLS es un método muy seguro para cifrar el tráfico, pero

también es un recurso intensivo para clientes ligeros debido a su apretón de manos requerido y el aumento del overhead de los paquetes. Para las redes donde la energía es una prioridad muy alta y la seguridad mucho menos, cifrar sólo la carga útil del paquete puede ser suficiente.

- Soporte para Calidad de Servicio (QoS): En MQTT, los niveles QoS 0, 1 y 2 describen niveles crecientes de entrega garantizada de mensajes. EL nivel 0 no repite paquetes, es decir, es un nivel de "apunta y dispara". El nivel 1 intenta garantizar que un mensaje sea recibido al menos una vez por el destinatario previsto. Una vez que un mensaje publicado es recibido y comprendido por el destinatario deseado, reconoce el mensaje con un mensaje de acuse de recibo (PUBACK) dirigido al nodo de publicación. Por último el nivel 2 intenta garantizar que el mensaje sea recibido y decodificado por el destinatario deseado. Este es el nivel de QoS MQTT más seguro y confiable.

- Testamento y última voluntad (LWT): MQTT proporciona un mensaje de "último deseo y testamento"que se puede almacenar en el Broker MQTT en caso de que un nodo se desconecte inesperadamente de la red. Este LWT conserva el estado y propósito del nodo, incluyendo los tipos de comandos que publicó y sus suscripciones. Si el nodo desaparece, el intermediario notifica a todos los suscriptores del LWT del nodo. Y si el nodo vuelve, el corredor lo notifica de su estado anterior. Esta característica acomoda redes con alto porcentaje o probabilidades de pérdidas.

- Suscripciones flexibles de temas: Un nodo MQTT puede suscribirse a todos los mensajes dentro de una funcionalidad determinada. Por ejemplo, en un cocina un nodo horno puede suscribirse a todos los mensajes de "cocina/horno/+", con el "+" como comodín. Esto permite una cantidad mínima de código. Hay otros comodines MQTT igualmente útiles para reducir la cantidad del código utilizado y por lo tanto el tamaño y el coste de la memoria.

Por otro lado MQTT tiene tres desventajas evidentes:

- El Broker: Como intermediario central puede ser un inconveniente para los sistemas IoT distribuidos. En entornos donde la existe poca presencia de sensores y/o actuadores o donde no crece la cantidad de los dispositivos, la presencia de un Broker genera latencia dentro de la red. Por otro lado el Broker puede convertirse en un único punto de fallo para la red completa.

- El uso de TCP: TCP fue diseñado originalmente para dispositivos con más memoria y recursos de procesamiento que los que pueden estar disponibles en una red de dispositivos IoT. Por ejemplo, el protocolo TCP requiere que las conexiones se establezcan en un proceso de apretón de manos de varios pasos antes de intercambiar cualquier mensaje. Esto aumenta los tiempos de activación y de comunicación y reduce la duración de la batería a largo plazo.

- Retrasos en la transmisión: Una vez más, el uso de TCP sin persistencia de sesión puede requerir un tiempo de transmisión incremental para el establecimiento de la conexión. Para nodos con tráfico periódico y repetitivo, esto puede afectar el consumo energético.

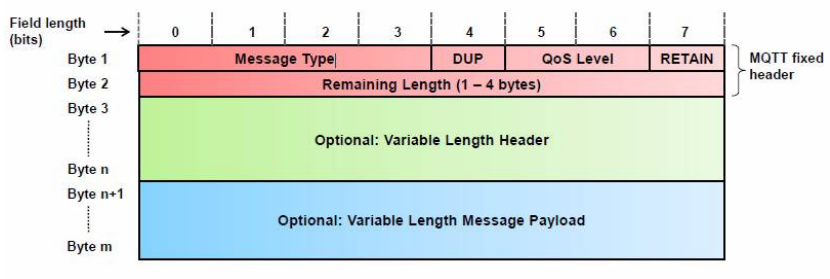

Figura 2.31: Formato de Mensaje del Protocolo MQTT

2.5.6.4. Protocolo Mosquitto

Mosquito es una implementación de código abierto del protocolo MQTT. Como tal, Mosquitto está destinado para su uso en todas las situaciones donde hay una necesidad de mensajería ligera, en particular en los dispositivos con restricciones con recursos limitados [116]. El proyecto Mosquitto está auspiciado por la Fundación Eclipse.

Esta implementación está divida en tres partes:

1. EL servidor de Mosquitto.
2. Las utilidades del cliente mosquitto_pub y mosquitto_sub que son los métodos de comunicación con un servidor MQTT.
3. Una librería para clientes MQTT escrita en C, con un contenedor C++.

Mosquitto se utiliza en otros proyectos de código abierto como el proyecto de automatización de hogares openHAB y OwnTracks, el proyecto de seguimiento de ubicación personal y se ha integrado en otros productos comerciales.

2.5.6.5. Protocolo XMPP

XMPP fue originalmente llamado “Jabber”. Fue desarrollado para mensajería instantánea (IM) para conectar a personas con otras personas a través de mensajes de texto. XMPP significa Protocolo de presencia y mensajería extensible [112]. La idea de XMPP comenzó alrededor de 2002 cuando había un número creciente de clientes de mensajería instantánea que no podían hablar entre sí. Necesitabas muchas cuentas y siempre había diferencias de funcionalidad [117].

En el contexto del IoT, XMPP ofrece una manera fácil de abordar un dispositivo. Esto es especialmente útil si los datos van entre puntos lejanos, en su mayoría no relacionados, al igual que el caso que de persona a persona. No está diseñado para ser rápido, de hecho, la mayoría de las implementaciones usan el sondeo o revisan las actualizaciones sólo cuando se solicitan. XMPP utiliza el formato de texto XML como su tipo de dato nativo, haciendo que las comunicaciones de dispositivo a dispositivo sean naturales. Al igual que MQTT, se ejecuta sobre TCP y su diagrama de conexión se puede observar en la figura 2.32.

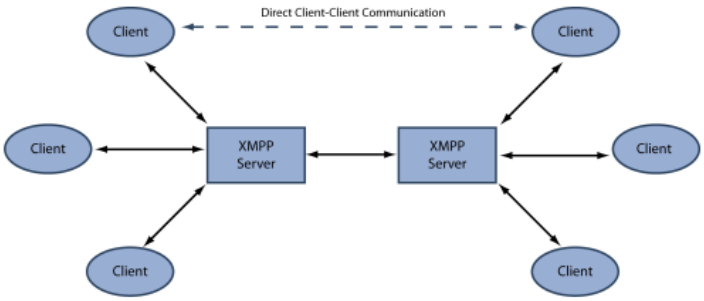

Figura 2.32: Diagrama de Conexión bajo XMPP

Los XMPP Extension Protocols añaden funcionalidades al protocolo y lo hacen óptimo para su uso en el desarrollo de IoT. Las ventajas de XMPP son las siguientes:

- Esquemas de comunicación flexibles: XMPP provee de un estándar de sockets y streams a través de HTTP síncrono (protocolo BOSH) ambos bidireccionales. También cuenta con una extensión que comprime los XML de forma que estos sean más eficientes durante el envío. Otra característica de XMPP es que da soporte a varios paradigma de comunicación como el de petición-respuesta, de publicación-suscripción o el de mensajería asíncrono.

- Autonomía: Los dispositivos IoT en general que tienen recursos limitados, tanto en procesamiento, memoria y consumo energético. El protocolo provee extensiones para delegar estas tareas en aquellos dispositivos que poseen mejores características para ello haciendo labores de servidor (XMPP Server) el cual decidirá qué dispositivos pueden comunicarse entre sí, qué dispositivos pueden leer qué tipo de datos de otro dispositivo y qué dispositivos pueden controlar otro dispositivo y qué pueden hacer.

- Escalabilidad: XMPP proporciona "identidades globales", que ayuda a identificar un número ilimitado de dispositivos por sus direcciones XMPP únicas. XMPP utiliza direcciones únicas a nivel mundial, basadas en el Sistema de Nombres de Dominio, para direccionar y entregar mensajes a través de la red. Todas las entidades XMPP son direccionables en la red, especialmente clientes y servidores, pero también varios servicios adicionales a los que pueden acceder clientes y servidores.

- Interoperatibilidad: El protocolo XMPP tiene un gran número de implementaciones de cliente, servidores y librerías que se distribuyen como software libre y de código abierto. XMPP también proporciona un conjunto de extensiones para promover la interoperabilidad entre dispositivos IoT, la realización de lecturas de sensores, el control de actuadores, federación de servicios.

- Modularidad: XMPP consiste en diferentes componentes y módulos que los desarrolladores pueden activar durante el tiempo de compilación en función de los escenarios específicos. Adicionalmente, el protocolo se puede extender a través del uso de los XEP y que implementen características novedosas o necesarias para la correcta comunicación de la red.

- Seguridad: El protocolo también proporciona una base sólida y flexible para las funciones de seguridad. XMPP facilita la administración de identidades, la autenticación, la autorización, la OTR (Off-the-Record Messaging) y el cifrado, incluido el cifrado de extremo a extremo. [118] Por defecto XMPP usa TLS como seguridad en la capa de transporte y SASL para autenticación.

La principal desventaja que presenta el protocolo XMPP es que produce un alto overhead en la red además de un largo proceso de handshake, lo cual causa que en los dispositivos donde se implemente, tengan un consumo energetico mayor al que con otros protocolos.

2.5.6.6. Protocolo AMQP

El protocolo de colas de mensajes avanzada (AMQP por sus siglas en inglés) es un protocolo que nace de la necesidad de un protocolo de comunicación que ante todo sea fiable y se enfoca en no perder mensajes utilizando el modelo de publicación-suscripción. Las comunicaciones de los publicadores a los Brokers y de las colas a los suscriptores utilizan TCP, que proporciona una conexión punto a punto estrictamente fiable. Además, los puntos finales deben reconocer la aceptación de cada mensaje. El estándar también describe un modo de transacción opcional con una secuencia de confirmación formal [112].

AMQP se utiliza principalmente en mensajería comercial. Normalmente define "dispositivos" como teléfonos móviles que se comunican con centros de datos. En el contexto IoT, AMQP es un protocolo apropiado para el control o funciones de análisis basadas en servidor. [119] Entre sus ventajas destaca:

- Eficiente en la comunicación: AMQP es un protocolo orientado a conexión que utiliza una codificación binaria para las instrucciones del protocolo y los mensajes comerciales transferidos sobre él. Incorpora sofisticados esquemas de control de flujo para maximizar la utilización de la red y los componentes conectados.
- Confiable: El protocolo permite intercambiar mensajes con una gama de garantías de fiabilidad y con niveles de QoS, desde el nivel 0 de "dispara y olvido" y el hasta el nivel 2 con entrega confiable, exactamente una vez reconocida.
- Flexibilidad: AMQP es un protocolo flexible que se puede utilizar para soportar diferentes modelos de comunicación, es decir, el mismo protocolo se puede utilizar para las comunicaciones de dispositivo a dispositivo, de dispositivo a puerta de enlace y de intercambio de datos en back-end.

Las desventajas que posee el protocolo AMQP son las mismas que las presentadas por el protocolo XMPP, su consideración de tener comunicaciones fiables y buscando la menor cantidad de perdidas posible lo hacen un protocolo que tiene un alto overhead y consumo energetico que no todos los dispositivos orientados al Internet de las Cosas pueden asumir.

2.5.6.7. DDS

El Servicio de Distribución de Datos para sistemas en tiempo real (Denominado de forma abreviada también como DDS) es la especificación para un middleware de tipo publicación-suscripción en sistemas distribuidos. DDS ha sido creado en el año 2011, en respuesta a las necesidades de la industria de estandarizar sistemas centrados en datos [120].

DDS implementa un espacio virtual de datos globales (figura 2.33) donde las aplicaciones pueden compartir información con la simple lectura y escritura de datos u objetos abordados por medio de un nombre definido por la aplicación (Topic) y una clave. Los topics pueden ser definidos en tiempo de ejecución y declarados no sólo mediante el lenguaje de especificación IDL (Lenguaje de Definición de Interfaces), sino también utilizando XML (Lenguaje de Marcas Generalizado) o XSD (Lenguaje de Esquema XML).

En la actualidad el protocolo está extendiéndose sobre la integración de DDS con diferentes tecnologías web, tales como HTTP, clientes SOAP (Protocolo de Acceso de Objeto Simple) o REST (Transferencia de Estado Representacional), así como la definición de un marco de seguridad. DDS cuenta con un control preciso y extenso de los parámetros de calidad de servicio, incluyendo la fiabilidad, ancho de banda, los plazos de entrega, y los límites de recursos. Los pilares de DDS son:

- Centrado en los datos: DDS es exclusivamente centrada en los datos, que es ideal para la Internet de las cosas. El que este centrado en los datos garantiza que todos los mensajes incluyan la información contextual que una aplicación necesita para comprender los datos que recibe.
- Espacios Globales de Datos: DDS conceptualmente ve un almacén local de datos llamado “espacio global de datos”. Para una aplicación, el espacio global de datos se parece a la memoria nativa a la que se accede a través de una API. Se escribe a lo que parece su almacenamiento local pero en realidad, DDS envía mensajes para actualizar los almacenes apropiados en nodos remotos. El espacio de datos global es un concepto virtual que es realmente sólo una colección de almacenes locales, así el dispositivo lee de lo que parece una almacén de datos local. En conjunto, los almacenes locales dan a las aplicaciones la ilusión de tener acceso a todo el espacio de datos global. Esto es sólo una abstracción pues no hay un lugar donde viven todos los datos. Cada aplicación almacena locamente sólo lo que necesita y sólo durante el tiempo que lo necesite. DDS se ocupa de los datos en comunicación. Cada aplicación, en casi cualquier lenguaje, que se ejecuta en cualquier sistema, ve memoria local en su formato nativo óptimo. El espacio de datos global comparte datos entre aplicaciones embebidas, móviles y de nube en cualquier transporte, independientemente del lenguaje o sistema, y con una latencia extremadamente baja.
- Niveles de QoS: Los datos también pueden compartirse con especificaciones de calidad de servicio (QoS) flexibles, incluyendo fiabilidad, salud del sistema e incluso seguridad. En un sistema real, no todos los puntos finales necesitan cada elemento de su almacén local. DDS es inteligente acerca de enviar lo que necesita.

Si los mensajes no siempre alcanzan sus destinos, el protocolo implementa la confiabilidad donde sea necesario. Cuando los sistemas cambian, el protocolo calcula dinámicamente hacia dónde enviar los datos e informa a los participantes de los cambios. Si el tamaño total de los datos es enorme, DDS inteligentemente filtra y envía sólo los datos que cada punto final realmente necesita. Cuando las actualizaciones deben ser rápidas, DDS envía mensajes de multidifusión para actualizar muchas aplicaciones remotas a la vez. A medida que evolucionan los formatos de datos, DDS realiza un seguimiento de las versiones utilizadas por varias partes del sistema y traduce automáticamente. Para aplicaciones de seguridad crítica, DDS controla el acceso, impone rutas de flujo de datos y cifra los datos al vuelo.

- Descubrimiento dinámico: DDS proporciona Descubrimiento dinámico de editores y suscriptores. El descubrimiento dinámico va incluso más allá del descubrimiento de puntos finales. DDS descubrirá si el punto final está publicando datos, suscribiéndose a datos o ambos. Descubrirá el tipo de datos que están siendo publicados o suscritos. También descubrirá las características de comunicación ofrecidas por el publicador y las características de comunicaciones solicitadas por el suscriptor. Todos estos atributos se tienen en cuenta durante el descubrimiento dinámico y la coincidencia de participantes de DDS. Los participantes de DDS pueden estar en la misma máquina o en una red: la aplicación utiliza la misma API DDS para las comunicaciones. Dado que no es necesario conocer o configurar direcciones IP, ni tener en cuenta las diferencias en las arquitecturas de máquinas, añadir un participante de comunicación adicional en cualquier sistema operativo o plataforma de hardware se convierte en una tarea fácil.

- Arquitectura Escalable: La arquitectura del protocolo DDS está diseñada para ser escalable desde pequeños dispositivos a la nube y para sistemas muy grandes. DDS permite al Internet de las Cosas escalar a miles o millones de participantes, entregando datos a muy alta velocidad, manejando miles de objetos de datos y proporcionando disponibilidad y seguridad en ambos extremos. DDS simplifica el desarrollo de sistemas distribuidos absorbiendo gran parte de la complejidad en una única capa de comunicaciones estándar.

La desventaja que presenta el protocolo DDS es que requiere que los dispositivos tengan unas capacidades no tan restringidas de cómputo y memoria con respecto a otros protocolos pues este se enfoca más en cómo evitar la comunicación excesiva de datos repetidos en la red. Estas características de cómputo y memoria por sobre la comunicación hacen que sea tenga un consumo promedio mayor por dispositivo, por lo que se debe tomar en cuenta para su implementación en una red de sensores actuadores y dispositivos en en general.

2.5.6.8. Protocolos IPv4 e IPv6

Internet Protocol o IP es un protocolo de capa de red, no orientado a conexión, con una política de mejor esfuerzo en la entrega de los paquetes y cuya principal tarea es permitir la comunicación bidireccional en la red haciendo uso de direcciones a las cuales se pueden enrrutar convenientemente. IP no provee ningún mecanismo para determinar

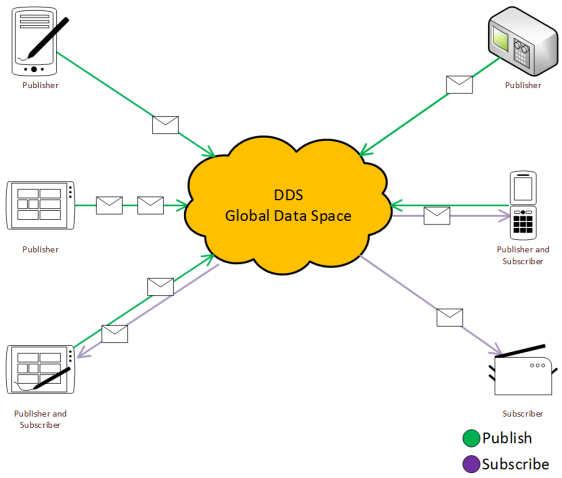

Figura 2.33: Modelo de Comunicación de DDS

si un paquete alcanza o no su destino y únicamente proporciona seguridad (mediante checksums o sumas de comprobación) de sus cabeceras y no de los datos transmitidos.

Este protocolo tiene como pilares fundamentales:

- El direccionamiento: Se refiere a la capacidad de poder asignar una dirección lógica a una interfaz de red y a su vez conocer la manera en que se divide la red a la que pertenece. Para el direccionamiento efectivo, el protocolo hace uso de direcciones IP, un sistema de identificación lógica y jerarquicamente una interfaz de red.
- El redireccionamiento: El redireccionamiento permite redistribuir de manera eficiente el tráfico de las comunicaciones entre nodos de manera que estos puedan ir de su origen a su destino.

El protocolo IP ha ido evolucionando y en la actualidad conviven dos versiones del protocolo, IPv4 establecido en el RFC-791 [121] e IPv6 establecida en el RFC-2460 [122], ambos basados en los mismos principios pero con diferencias en la implementación, siendo IPv6 la versión que remplazará a IPv4.

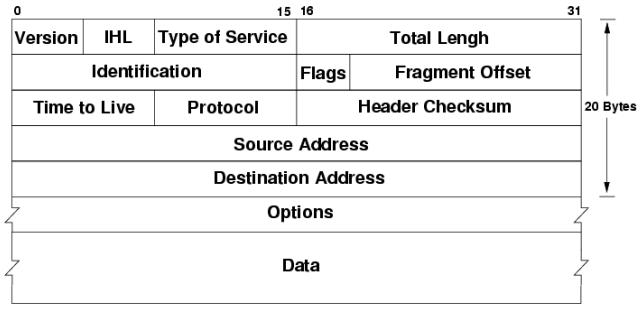

Figura 2.34: Formato de Mensaje IPv4

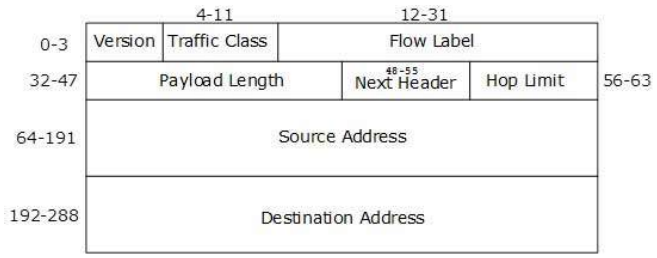

Figura 2.35: Formato de Mensaje IPv6

2.5.7. Estándares comunes

Los protocolos antes observados son parte del conjunto de opciones disponibles pensados para dar solución para entornos de alta cantidad de dispositivos en comunicación, cada uno de ellos con características que están diseñadas para dar una ventaja competitiva en este ámbito.

Sin embargo la adopción de cada protocolo dependerá no solo de las ventajas que representa su uso, sino también del apoyo que este tenga de la industria, de las organizaciones certificadoras y de los entes reguladores de forma que estos puedan pasar a una etapa de implementación.

2.5.7.1. Estándar Bluetooth

Bluetooth es uno de los primeros estándares de comunicación inalámbrica en dispositivos móviles. Desarrollado por la compañía sueca Ericsson estableció un protocolo de comunicación para redes de áreas personal (PAN), con una tasa de transmisión inicial de 1 Mbps, que luego paso a ser parte del estándar IEEE 802.15.1 [123]. Los dispositivos Bluetooth utilizan señales de radio en la frecuencia de 2.4 GHz para establecer dicha conexión inalambrica. Con este estándar se busca que los dispositivos puedan:

- Establecer conexiones con poco gasto de energía.
- Establecer enlaces por lo general de corta duración.
- Otorgar seguridad mediante diversas maneras de cifrado de datos, además de exigir el uso de un PIN para establecer conexiones entre equipos.
- Soportar tráfico de voz y datos.
- Tener un bajo costo de producción e implementación.

En la actualidad el estándar Bluetooth se compone de un núcleo o también llamado Bluetooth Core que contiene las directivas de comunicaciones generales e inherentes al estándar y de dos extensiones llamadas Bluetooth BR/EDR (Basic Rate/Enhanced Data Rate) optimizado para hacer uso de intercambio de streams de datos como audio y el Bluetooth Low Energy diseñado para hacer un uso muy eficiente de la energía utilizada en la comunicación, en casos donde el intercambio de datos es pequeño y que ha sido implementado desde la versión 4.0 del estándar. Los dispositivos tiene la posibilidad de

implementar ya sea Bluetooth BR/EDR o Bluetooth Low Energy o ambas de forma mixta.

La gran ventaja de este estándar es que gran cantidad de dispositivos cuentan con la capacidad de comunicarse a través de este estándar, además de que permite una comunicación con un modelo dispositivo a dispositivo fácil de establecer para el usuario final. Sin embargo estas comunicaciones inalámbricas son de corto alcance cosa que puede afectar el desempeño entre dispositivos. El Bluetooth Low Energy abre las puertas a dispositivos que requieren manejar de manera eficiente el consumo energético de las comunicaciones pero no es aconsejado uso para el intercambio exhaustivo de información.

Por otro lado, es recomendable utilizar el Bluetooth BR/EDR cuando la aplicación que se usa requiere enviar una cantidad de información importante o continuamente, pero tiene la desventaja que el consumo energético es mayor afectando negativamente la duración de los dispositivos si estos no operan con una fuente estable de energía.

2.5.7.2. Estándar Zigbee

Zigbee es otro estándar de comunicación inalámbrica que agrupa un conjunto de protocolos de comunicación basado en el estándar IEEE 802.15.4 [124] y que está diseñado para ser usado en la banda de 2.4 GHz en intercambios de paquetes poco frecuentes. Opera en un rango no mayor a 100 metros.

Zigbee tiene como características un funcionamiento con operación en bajo consumo energético, ofreciendo seguridad, robustez y escalabilidad de altas cantidades de nodos y dispositivos conectados y está pensado para ser utilizado para el intercambio de paquetes bajo un modelo dispositivo a dispositivo [125].

Una de las principales ventajas de Zigbee es lo sencillo y el bajo coste que supone para la empresa producir dispositivos con esta tecnología de comunicación ya que es mucho más sencillo que Bluetooth por ejemplo [126]. Existen tres distintos roles que puede cumplir un nodo o dispositivo que hace uso del estándar Zigbee:

- El Coordinador Zigbee: es el nodo más completo y se encarga de controlar toda la red y los caminos para su comunicación [126]. Es de carácter obligatorio para la comunicación de los dispositivos.
- El Router Zigbee: interconecta los nodos para poder ejecutar código del usuario, es decir, ofrece un nivel de aplicación dentro del stack de protocolos [126]. Puede ser el mismo nodo que es Coordinador.
- El Dispositivo: Los nodos finales de ZigBee sólo reciben información y se comunican únicamente con el nodo padre (coordinador y/o router) [126].

Zigbee es altamente utilizado en entornos industriales, sin embargo no es tan ampliamente utilizado como su rival más cercano el estándar Bluetooth, que posee más compañías asociadas brindándole apoyo.

2.5.7.3. Estándar Z-Wave

El estándar Z-Wave es una tecnología de comunicación inalámbrica de baja potencia diseñada principalmente para la automatización del hogar en productos como lámparas, sensores y muchos otros. Esta optimizado para ser estable, de baja latencia y con velocidad de datos de hasta 100 kbps [125].

A diferencia de otros estándares , Z-Wave no utiliza la banda de los 2.4 GHz, sino que utiliza la banda de los 900 MHz, pues con ello proporciona un rendimiento superior por dos motivos: menos interferencias (por funcionar a baja frecuencia) con otros dispositivos y mayor penetración de las ondas en paredes, pisos y muebles (al tener mayor longitud de onda) [127].

Una característica importante de este estándar es que implementa redes malladas (Mesh) para la comunicación entre nodos, permitiendo una cantidad alta de dispositivos interactuando en la red y posibilitando un amplio rango en el despliegue de la misma. Los tres principales problemas que posee el estándar son:

- Al utilizar una red mallada, la comunicación entre dispositivos se maneja en cada nodo, por lo que este requiere tener un poder de procesamiento acorde para ello, además del gasto de energía que pueda suponer el comunicarse con múltiples puntos de la red.
- Los chips avalados para trabajar con este estándar son fabricados por una sola compañía (Sigma Designs), lo cual afecta la producción de dispositivos que manejen el estándar por la distribución de los chips.
- Al usar la banda de los 900 MHz dependiendo de la región en donde se planee utilizar los dispositivos, hace falta en muchos casos homologarlos pues la banda puede estar siendo utilizada para otros fines.

2.5.7.4. Estándar 6LowPAN

El estándar 6LowPan (IPv6 over Low Power Wireless Personal Area Networks) es una tecnología basada en IP (Internet Protocol) versión 6. En lugar de ser un estándar de protocolos de aplicaciones IoT como Bluetooth o ZigBee, 6LowPAN define mecanismos para la mejor utilización disponible de las opciones del protocolo IPv6 para la comunicación entre nodos. El estándar tiene la libertad de banda de frecuencia y capa física y también se puede utilizar a través de múltiples plataformas de comunicaciones, incluyendo Ethernet, WiFi, 802.15.4 y sub-1GHz ISM [125].

La especificación base para el desarrollo de 6LowPAN es tan enmarcados en los RFC-4944 [128], RFC-6282 [129] y el RFC-6775 [130]. El principal problema del estándar es que deja demasiada libertad en cuanto al resto de la pila de protocolos y configuraciones en las comunicaciones posible, haciendo que sea más complicado diseñar alternativas que usen este estándar por sobre otros.

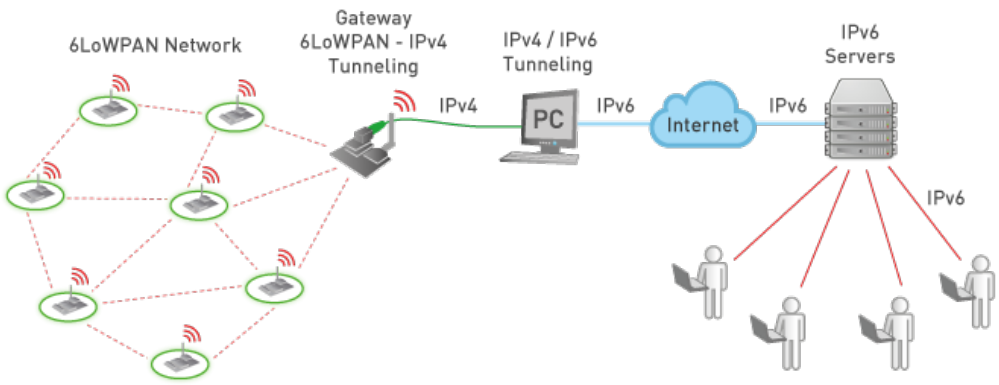

Figura 2.36: Red de una Conexión del Estándar 6LowPan

2.5.7.5. Estándar WiFI (802.11)

WiFi es el estándar de comunicación inalámbrico más utilizado en el mundo después de las redes celulares y ha sido una selección obvia al momento de integrar comunicación de dispositivos con una red. El estándar WiFi está basado en el estándar de la IEEE 802.11 [131] y ha atravesado una serie de versiones que han permitido mejorar su cualidades de despliegue en termino de distancia, velocidad de transmisión, seguridad e interoperatibilidad.

El estándar 802.11 consta de una serie de técnicas de modulación half duplex inalámbrica que utilizan el mismo protocolo básico. Las versiones 802.11b y 802.11g utilizan la banda ISM de 2.4 GHz. La dificultad que representa usar esta banda es que los dispositivos pueden sufrir interferencias con electrodomésticos tan comunes como el microondas o el horno o con dispositivos Bluetooth. Es por eso que deben controlar dicha susceptibilidad a las interferencias mediante métodos de señalización de espectro ensanchado por secuencia directa (DSSS) y de multiplexación por división de frecuencia ortogonal (OFDM), respectivamente.

Por otro lado la versión 802.11a y la versión 802.11ac utilizan la banda de 5 GHz. También hay versiones que pueden usar ambas bandas, como 802.11n pueden utilizar las bandas de 2,4 GHz o la de 5 GHz. El alcance de este estandar depende de la potencia del dispositivo, de la versión de WiFi y de las bandas utilizadas, al igual que la velocidad de transmisión pudiendo pasar de 54 Mbps a 1 Gbps.

A pesar de ser uno de los estándares más utilizados, WiFi posee como inconvenientes la complejidad del hardware y software requerido así como también la cantidad de energía requerida para establecer y mantener la comunicación, convirtiéndose en prohibitivo para dispositivos con capacidades limitadas. En las revisiones más nuevas del estándar se está buscando la manera de que se mantengan las ventajas de transmisión y de espectros utilizados, sin que esto signifique un impacto severo en el consumo energético de los dispositivos.

2.5.7.6. Estándares de red celular 3G y 4G

Los estándares celulares son las más utilizadas en cuanto a redes inalámbricas. Extendidas alrededor del mundo proveen a los dispositivos una manera sencilla de poder conectarse a servicios de voz y datos.

Las tecnologías de 3G son la respuesta a la especificación IMT-2000 de la Unión Internacional de Telecomunicaciones [132]. En Europa y Japón se seleccionó el estándar UMTS (Universal Mobile Telecommunication System) [133], basado en la tecnología W-CDMA [134]. UMTS está gestionado por la organización 3GPP, también responsable de GSM, GPRS y EDGE. Para el continente americano, para África y para el resto del mundo se establecieron los estándares High-Speed Packet Access (HSPA) [135] es una fusión de dos protocolos móviles, High Speed Downlink Packet Access (HSDPA) [135] y High Speed Uplink Packet Access (HSUPA) [135] que extiende y mejora el rendimiento de las redes de telecomunicaciones móviles de tercera generación (3G), como son el 3.5G o HSDPA y 3.5G Plus, 3.75G o HSUPA existentes utilizando los protocolos WCDMA.

A finales de 2008 se lanzó un estándar 3GPP aún más mejorado, Evolved High Speed Packet Access (también conocido como HSPA+) [135], posteriormente adoptado a nivel mundial a partir de 2010. Este nuevo estándar permitía llegar a velocidades de datos tan altas como 337 Mbps en el enlace descendente y 34 Mbps en el enlace ascendente.

La cuarta generación de tecnología en telecomunicaciones móviles, comúnmente conocido como 4G, es el estándar aplicado al mercado móvil de la actualidad. Para que una tecnología pueda ser considerada como parte del 4G, esta debe de cumplir con ciertos requisitos dentro del estándar. Entre las tecnologías que se encuentran dentro de dicho estándar están: LTE-Advanced [136], WiMAX móvil (IEEE 802.16e) [137] y WiMAX Release 2 (IEEE 802.16m) [137].

Los requerimientos para que una red sea considerada 4G están expuestos en la IMT-advanced de la ITU y son:

- Alto grado de uniformidad de funciones en todo el mundo, manteniendo al mismo tiempo la flexibilidad de admitir una amplia gama de servicios y aplicaciones rentables.
- Compatibilidad de servicios con las IMT y las redes fijas.
- Interoperatibilidad con otros sistemas de acceso radioeléctrico.
- Servicios móviles con soporte para calidad de servicio.
- Capacidad de itinerancia mundial.
- velocidades de 100 Mbits para una movilidad alta y de 1 Gbits para una movilidad baja.

Plenamente diseñados para hacer su uso en dispositivos de carácter móvil y aprovechando la infraestructura existente, hacen de los estándares de comunicación celular

una de las opciones más fáciles y convenientes de implementar en soluciones de comunicación inalámbrica, sobre todo cuando no se puede implementar una red dedicada para el tráfico de los nodos. Hacen un uso eficiente de la energía del dispositivo, pues es el recurso crítico para las operaciones.

2.5.7.7. Estándar NFC

La Comunicación de Campo Cercano o NFC por sus siglas en inglés, es un estándar que permite transferencia bidireccional de datos entre dispositivos. Su rango de transmisión es corto de máximo 10 cm de distancia, con tasas de transferencia máxima de 420kbps. El conjunto de tecnologías que permiten está comunicación están enmarcados en el estándar ISO/IEC 18000-3 [138].

NFC puede funcionar en dos modos:

- Activo, en el que ambos equipos con chip NFC generan un campo electromagnético e intercambian datos.
- Pasivo, en el que solo hay un dispositivo activo y el otro aprovecha ese campo para intercambiar la información.

En cuanto a la seguridad, NFC utiliza su corta distancia de transmisión como un mecanismo de protección pues cualquier ataque deberá realizarse cerca de los dispositivos que realizan la transmisión.

2.5.8. Seguridad

La seguridad de los dispositivos diseñados para formar parte del Internet de las Cosas, deben estar centrados no solamente en la recolección de datos, el despliegue de información o acciones sobre el ambiente sino en todos los eslabones que hagan parte del conjunto de tecnologías utilizadas para tales fines.

Los aspectos fundamentales de seguridad deben ser de carácter transversal a toda la infraestructura de IoT. También debe ser abordada durante todo el ciclo de vida de los dispositivos, desde el diseño inicial, pasando por las pruebas requeridas para comprobar su correcto funcionamiento, hasta la implantación en el entorno deseado y que incluye todo lo concerniente a la comunicación, al acceso y administración de los dispositivos, equipos y servicios de la arquitectura utilizada, a los datos que pueden proveer, o todo recurso que los haga un blanco ideal para los piratas informáticos.

La seguridad debe ser considerada en todos los elementos de cualquier plataforma o servicio que haga uso de algún aspecto del IoT [139]. Así se pueden separar en categorías las diferentes medidas aplicables para establecer la seguridad de la siguiente forma:

- Dispositivos o Equipos: Para todos los equipos utilizados en los procesos en los que interviene el IoT, es importante que estos tengan durante su ciclo de vida, actualizaciones regulares que corrijan errores y malos funcionamientos, tanto a nivel del sistema operativo, firmware de dispositivos y funcionalidades previstas

a nivel de software y hardware. También es importante controlar el acceso a estos dispositivos y equipos por parte de otros equipos o servicios utilizando servicios de listas blancas y listas negras.

- Protocolos de Seguridad en Internet: La seguridad que pueden brindar los protocolos de red es parte importante de las consideraciones en el diseño y funcionamiento de los dispositivos. Se establece haciendo uso de las características de encriptado, del uso de certificados de confianza y aprovechando las ventajas que pueda proveer cada protocolo en particular.
- Redes y Puertas de Enlaces: Desde el punto de vista de las redes, los dispositivos ruteadores y gateways, la seguridad viene de la mano de asegurar los mecanismos provistos para las comunicaciones, haciendo uso de cortafuegos, logs y previniendo el acceso no autorizado a los equipos de red de manera física o remota y siempre utilizando métodos estrictos de autenticación de usuarios.
- Protocolos de Comunicación: Los protocolos de comunicación son claves para mantener a salvo la información generada y en transmisión. La clave radica en que además de cumplirse con los estándares de calidad y seguridad establecidos por normativas o regulaciones, también se establezcan políticas de seguridad extras con el fin de mantener a los nodos seguros y así evitar las intrusiones de terceros.
- Consumidores y Aplicaciones: Los dispositivos y equipos deben ser capaces de darle la opción a los usuarios de definir el nivel de seguridad requerido en los niveles de toda la infraestructura tecnológica utilizada, haciendo uso de interfaces de control o monitoreo. La educación de los usuarios, en cuanto a las mejores prácticas y el manejo adecuado de los dispositivos, plataformas y la información generada o almacenada, es también parte fundamental para mantener la seguridad en todo momento.
- Arquitecturas: Las soluciones tecnológicas deben contemplar las diversas amenazas y vulnerabilidades que puedan presentar el software, el hardware y el middleware en cada elemento de la arquitectura del Internet de las cosas y que deben de ser contemplados desde la etapa de diseño. La evaluación de riesgos debe ser parte fundamental en cada escenario de uso y en cada momento de la puesta a punto de los infraestructura tecnológica, y validando los resultados a través del uso de pruebas, adquisición de certificados de entes competentes y permitiendo auditorías periódicas de los resultados obtenidos.
- Identificación y Privacidad: Los diversos usuarios y servicios (incluyendo las terceras partes) que pueden interactuar con el Internet de las Cosas, tanto para el despliegue como para la recolección de datos e información, necesitan regirse según modelos de gestión, administración, identificación y permisología adecuados, de manera que la privacidad, los privilegios, la integración y de todas las acciones que ocurren en los sistemas, estén siendo manejados de manera responsable, cumpliendo con los estándares se seguridad requeridos según el contexto.

La acelerada proliferación y adopción de dispositivos conectados ha despertado la preocupación general por las consecuencias que podrían llegar a tener los fallos de seguridad de los mismos. Ya se han dado oportunidades donde brechas de seguridad en

acceso a los dispositivos, han terminado formando parte de botnets que han causado ataques de Denegación de Servicio Distribuido (DDoS) a servicios web de grandes compañías y redes sociales mundiales. [140] Para evitar que estos ataques se produzcan en el futuro hace falta realizar una serie de esfuerzos [141] que ayuden a mitigar las amenazas posibles. Estos esfuerzos son:

- Coordinar a través de los departamentos y agencias gubernamentales para involucrarse con las partes interesadas de IoT y explorar conjuntamente formas de mitigar los riesgos planteados por IoT.
- Crear conciencia de los riesgos asociados con el IoT entre los grupos de interés. Es importante que las partes interesadas sean conscientes de los riesgos de IoT para que puedan posicionarse para abordarlos.
- Identificar y promover incentivos para incorporar la seguridad en el IoT. Quienes formulan políticas, los legisladores y las partes interesadas deben considerar formas de incentivar los esfuerzos para mejorar la seguridad de la Internet de las Cosas. En el entorno actual, a menudo no está claro quién tiene la responsabilidad de la seguridad de un producto o sistema dado. Además, los costos de la escasa seguridad a menudo no son soportados por los que están mejor posicionados para aumentar la seguridad. Los interesados deben considerar cómo la responsabilidad civil, la legislación, la reglamentación, la gestión voluntaria de la certificación, las iniciativas de establecimiento de normas y las iniciativas voluntarias al nivel de la industria y otros mecanismos podrían mejorar la seguridad y al mismo tiempo fomentar la actividad económica y la innovación.
- Contribuir a los procesos de desarrollo de estándares internacionales para IoT. Al ser parte de un ecosistema global, las organizaciones están comenzando a evaluar cuáles son las consideraciones de seguridad a nivel general que debe poseer este conjunto de tecnologías. Es importante que las actividades relacionadas con el IoT no se dividan en conjuntos inconsistentes de normas o reglas.

2.6. Automatización del Hogar (Domótica)

2.6.1. Definición

La automatización del hogar, también conocida como domótica (portmanteau entre *domus* que en latín significa casa y el sufijo "tica" en referencia a la informática). Esta área del desarrollo tecnológico busca la automatización de actividades domésticas diversas con el fin de aumentar la comodidad, el confort, la eficiencia energética y/o la seguridad. Para ello, una serie de servicios en red están vinculados a diferentes elementos dentro del hogar, tales como la iluminación, sistemas HVAC (siglas en inglés de Calefacción, Ventilación y Aire Acondicionado), electrodomésticos, dispositivos de entretenimiento y otros sistemas [142].

El objetivo es disminuir la interacción humana tanto como sea técnicamente posible y deseable en varios procesos domésticos, y reemplazándolos con sistemas electrónicos programados esencialmente para cumplir las tareas domésticas.

El concepto de automatización del hogar no es nuevo. Desde la revolución industrial, cada vez mas se hacía posible remplazar a las personas en tareas repetitivas con la idea de que se pudiera tener mayor tiempo libre y con la aparición de las primeros artefactos automáticos para el hogar y la masificación de la electricidad potenciaron esta tendencia para facilitar las labores de diarias.

Electrodomésticos y artefactos de ocio se han hecho de un espacio cada vez mayor en los hogares y estos adquieren capacidades de comunicación y recolección de datos, el despliegue de información y la actuación programada de tareas, por lo que se surge la necesidad de crear nuevas formas de controlar, de monitorear y de utilizar los objetos que nos rodearan y las diversas aplicaciones que tendrán en nuestra vida diaria.

Inicialmente, la única manera de construir una instalación domótica era con el uso de sensores y actuadores que se unían, con una arquitectura centralizada, a un autómata o controlador que tenía embebida toda la inteligencia que se exigía a la vivienda. Casi siempre eran sistemas propietarios, muy poco flexibles, difíciles y costosos [143].

A pesar de la aparición de estándares y tecnologías que han abaratado y reducido la complejidad de las instalaciones domóticas, hasta la fecha esta industria no ha tenido la difusión ni los índices de adopción esperados. Pero ahora, gracias a Internet, estamos viendo como están apareciendo multitud de fabricantes y proveedores de servicios que están desarrollando nuevos productos que conjugan lo mejor de Internet con tecnologías de redes de datos, con controles automatizados, asequibles y estandarizados.

La capacidad de integrar varios sistemas y estar continuamente aprendiendo y adaptándose a través del análisis de datos complejos es la siguiente etapa en el desarrollo de los hogares inteligentes. Este proceso de aprendizaje impulsa un sistema centralizado que gestione los principales subsistemas de su hogar: iluminación, calefacción, seguridad, audio, persianas, etc. En este escenario, una casa inteligente es un ecosistema, supervisado por un "cerebro" central y controlado a través de un teléfono inteligente,

tableta, computador o de un asistente virtual. Como valor agregado final se busca el detectar rápidamente patrones de uso y sus preferencias personales para predecir lo que mejor le conviene y desea el usuario [144].

2.6.1.1. Arquitecturas

Para desarrollar aplicaciones de automatización de hogares, se debe tener en cuenta que la manera en la que los artefactos, dispositivos, electrodomésticos y demás objetos se comunicaran, las diversidad de los mismos y donde residirá la lógica programada de los elementos. De esta forma podemos pensar en cuatro distintas arquitecturas dentro de entornos domóticos:

- Arquitectura centralizada: Esta arquitectura está organizada de forma que un nodo sea el controlador o el "eje central" del sistema, recibiendo la información de los sensores, analizándola, y enviando una orden a los actuadores, según la configuración, o la información que reciba por parte del usuario.
- Arquitectura distribuida: Este tipo de arquitectura se diferencia por tener sensores y actuadores que son a su vez sus propios controladores, es decir son capaces de analizar la información, y están conectados a través de un bus central (medio de comunicación) para compartir datos e instrucciones.
- Arquitectura semi-distribuida: En una arquitectura semi-distribuidas existen varios controladores, conectados a sensores y actuadores, quienes a su vez están interconectados a uno o más hubs.
- Arquitectura mixta: Esta arquitectura se combinan las ventajas de los sistemas centralizados, distribuidos y/o semi-distribuidos. Por lo que puede disponer de un controlador central o varios controladores distribuidos. Los dispositivos pueden ser interfaces, sensores y actuadores y a su vez ser controladores y procesar la información (que captan ellos mismos o de otro sensor) tomando la iniciativa de que hacer para adaptarse a las necesidades, reglas o requerimientos del usuario.

2.6.2. Protocolos y Estándares Utilizados

Una vez establecidos las diversas arquitecturas existentes para desplegar una solución de domótica, se han implementado diversos protocolos que aprovechan las ventajas de dichas arquitecturas. Algunos protocolos requieren del uso de materiales y dispositivos especiales para poder ser instalados en el hogar, mientras que otros aprovechan la infraestructura existente para poder llevar a cabo todos los procesos necesarios por los nodos, indistintamente de si son sensores, actuadores, de control o con múltiples roles.

2.6.2.1. X10

El protocolo X10 es un estándar de comunicación para transmitir señales de control entre equipos de automatización del hogar a través de la red eléctrica (220V o 110V). X10 es uno de los protocolos más antiguos y aun en uso en aplicaciones domóticas. Fue diseñado en Escocia entre los años 1976 y 1978 con el objetivo de transmitir datos por

las líneas de baja tensión a muy baja velocidad (60 bps en EEUU y 50 bps en Europa) y muy bajo costo. Al usar las líneas eléctricas de la vivienda, no es necesario tender nuevos cables para conectar dispositivos [145].

Gracias a su madurez y a las tecnologías empleadas, los productos bajo el estándar X10 tienen un precio muy competitivo, de forma que han sido líderes en el mercado residencial y de pequeñas empresas. En lo que a instalaciones se refiere, estas pueden ser realizadas por los usuarios finales o electricistas sin conocimientos de automatización.

El protocolo X10, en sí, no es propietario, es decir, cualquier fabricante puede producir dispositivos X10 y ofrecerlos en su catálogo, sin embargo si está obligado a usar los circuitos del fabricante Pico Electronics of Glenrothes que diseñó esta tecnología.

2.6.2.2. KNX/EIB

El Bus de Instalación Europeo (EIB o EIBus) actualmente llamado KNX es un sistema de domótica basado en un bus de datos. A diferencia de X10, que utiliza la red eléctrica, y otros sistemas actuales por radio frecuencia, el KNX utiliza su propio cableado, con lo cual se ha de proceder a instalar la canalización adecuada en el hogar y para el sistema. El KNX, a través de puertas de enlace puede ser utilizado en sistemas inalámbricos como los infrarrojos, radiofrecuencia o incluso empaquetado para enviar información por Internet u otra red TCP/IP [146].

KNX nace de la unión de las iniciativas de tres asociaciones europeas unidas para crear un único estándar europeo para la automatización de las viviendas y oficinas [145]:

- EIBA, (European Installation Bus Association).
- BatiBUS Club International.
- EHSA (European Home System Association).

Este protocolo posee 3 modos de funcionamiento:

- S.mode (System mode): En esta configuración el sistema usa la misma filosofía del EIB actual, es decir, los diversos dispositivos o nodos de la red son instalados y configurados por profesionales con ayuda del software de aplicación especialmente diseñada para este propósito. Está especialmente pensado para su uso en instalaciones como oficinas, industrias, hoteles, etc. Sólo los instaladores profesionales tendrán acceso a este tipo de material y a las herramientas de desarrollo. Los dispositivos S.mode sólo pueden ser comprados a través de distribuidores eléctricos especializados.
- E.mode (Easy mode): En esta configuración sencilla los dispositivos son programados en fábrica para realizar una función concreta. Aún así deben ser configurados algunos detalles en la instalación, ya sea con el uso de un controlador central (como una puerta de enlace residencial o similar) o mediante unos microinterruptores alojados en el mismo dispositivo. Cualquier electricista sin formación en manejo de herramientas informáticas o cualquier usuario final autodidacta, puede

conseguir dispositivos E.mode en ferreterías o almacenes de productos eléctricos. Aunque la funcionalidad de estos productos está limitada (viene establecida de fábrica), la ventaja de este modo es que se configuran en un instante seleccionando, en unos microinterruptores, las opciones ofrecidas con una pequeña guía de usuario.

- A.mode (Automatic mode): En esta configuración automática, con una filosofía Plug&Play, ni el instalador, ni el usuario final tienen que configurar el dispositivo. Este modo está especialmente indicado para ser usado en electrodomésticos, equipos de entretenimiento (consolas, set-top boxes, HiFi, etc) y proveedores de servicios. Es el objetivo al que tienden muchos productos informáticos y de uso cotidiano. Con la filosofía Plug&Play, el usuario final no tiene que preocuparse de leer complicados manuales de instalación o perderse en un mar de referencias o especificaciones. Tan pronto como conecte un dispositivo A.mode a la red este se registrará en las bases de datos de todos los dispositivos activos en ese momento en la instalación o vivienda y pondrá a disposición de los demás sus recursos (procesador, memoria, entradas/salidas, etc). Son los fabricantes de electrodomésticos y de pasarelas residenciales, así como los proveedores de servicios (empresas de telecomunicaciones, eléctricas, ISPs), los más interesados en este tipo de productos ya que permitirán ofrecer nuevos servicios a sus clientes de forma rápida y sin necesidad de complicadas instalaciones.

2.6.2.3. OSGI

La asociación Open Services Gateway Initiative (OSGI) fue creada en marzo de 1999 con el objetivo de crear una especificación software abierto y libre de regalías, que permita diseñar y construir plataformas compatibles que sean capaces de proporcionar múltiples servicios en el mercado residencial y automovilístico.

En el ámbito de la gestión técnica de las edificaciones, el OSGI pretende ofrecer una arquitectura completa de extremo a extremo, que cubra todas las necesidades del proveedor de servicios, del cliente y de cualquier dispositivo instalado en las viviendas [147].

Las áreas en que se vuelcan todos los esfuerzos del OSGI son:

- Servicios: Se busca la creación de plataformas que sean capaces de procesar y tratar de forma correcta toda la información necesaria para proporcionar servicios de comunicaciones, de entretenimiento, de control y de seguridad. Por lo tanto, la especificación OSGI debe tener los interfaces adecuadas para soportar todos estos servicios sin incompatibilidades además de permitir gestionarlos de forma adecuada.

- Métodos de acceso: La idea es que el gateway para OSGI sea capaz de acceder al mundo exterior (redes de datos, Internet) usando cualquiera de las tecnologías disponibles actualmente.

- Redes de datos y control de las viviendas: Teniendo en cuenta la variedad de hogares y edificios en donde este tipo de pasarelas deben ser instaladas, esta iniciativa

no escoge una única tecnología de conexión en red de múltiples dispositivos de las viviendas. Su objetivo es definir un interfaz común para todas ellas, dejando la responsabilidad a los fabricantes de construir los controladores adecuados para cada una de ellas. Con esto en mente, los gateways OSGI podrán usar tecnologías de conexión inalámbricas (IrDa, IEEE 802.11x, Bluetooth), sobre cables telefónicos (HomePNA), sobre la red de baja tensión (HomePlug, LonWorks, EIB/KNX, etc), sobre conexiones como Ethernet o USB. Por lo tanto, la especificación OSGI será el "gateway" que transforme los paquetes de información procedentes del mundo exterior a un paquete de datos de cualquiera de estas tecnologías y viceversa.

2.6.2.4. LonWorks

LonWorks es una plataforma de control creada por la compañía norteamericana Echelon [146]. Presentado en el año 1992 es un protocolo diseñado para cubrir los requisitos de la mayoría de las aplicaciones de control: edificios de oficinas, hoteles, transporte, industrias, monitorización de contadores de energía, viviendas, etc.

El protocolo LonWorks se encuentra homologado por las distintas normas Europeas (EN-14908), de Estados Unidos (EIA-709-1) y Chinas (GB/Z20177-2006) así como por el estándar europeo de electrodomésticos CEDEC AIS. Su arquitectura es un sistema abierto a cualquier fabricante que quiera usar esta tecnología, sin depender de sistemas propietarios, permitiendo reducir los costes y aumentar la flexibilidad de la aplicación.

Cualquier dispositivo LonWorks, o nodo, está basado en un microcontrolador llamado Neuron Chip. El diseño inicial del Neuron Chip y el protocolo LonTalk fueron desarrollados por Echelon en el año 1990. Actualmente toda la información para implementar LonWorks en otro chip está publicada en medios oficiales pero al estar la familia Neuron Chips adaptada y dimensionada exclusivamente para este objetivo los fabricantes que eligen otras opciones son muy escasos.

Para el funcionamiento de LonWorks en los dispositivos, estos deben implementar lo siguiente:

- Identificador único: El Neuron ID, que permite direccionar cualquier nodo de forma unívoca dentro de una red de control LonWorks. Este identificador, con 48 bits de ancho, se graba en la memoria EEPROM durante la fabricación del circuito.
- Modelo de comunicaciones independiente del medio físico: Los datos pueden transmitirse sobre cables de par trenzado, ondas portadoras, fibra óptica, radiofrecuencia y cable coaxial, entre otros.
- El firmware que implementa el protocolo LonTalk, proporciona servicios de transporte y routing punto a punto: Está incluido un sistema operativo que ejecuta y planifica la aplicación distribuida y que maneja las estructuras de datos que intercambian los nodos.

Los datos pueden enviarse en dos formatos, un mensaje explícito o una variable de red. Los mensajes explícitos son la forma más sencilla de intercambiar datos entre dos aplicaciones residentes en dos Neuron Chips del mismo segmento LonWorks. Por el contrario, las variables de red proporcionan un modelo estructurado para el intercambio automático de datos distribuidos en un segmento LonWorks. Aunque son menos flexibles que los mensajes explícitos, las variables de red evitan que el programador de la aplicación distribuida esté pendiente de los detalles de las comunicaciones [145].

2.6.2.5. Universal Plug and Play (UPnP)

Universal Plug and Play (UPnP) es una estándar de software abierto y distribuido que permite a las aplicaciones de los dispositivos conectados a una red intercambien información y datos de forma sencilla y transparente para el usuario final, sin necesidad de que este tenga que ser un experto en la configuración de redes, dispositivos o sistemas operativos. Esta arquitectura está por encima de protocolos como TCP, UDP, IP, entre otros y es independiente de éstos.

Este protocolo es capaz de detectar cuando se conecta un nuevo equipo o dispositivo a la red, asignándole una dirección IP, un nombre lógico, informando a los demás de sus funciones y capacidad de procesamiento, e informarle, a su vez, de las funciones y prestaciones de los demás. De esta forma, el usuario no tiene que preocuparse de configurar la red ni de perder el tiempo instalando drivers o controladores de dispositivos [147].

2.6.2.6. CEBus

En 1984 varios miembros de la Electronics Industry Association (EIA) llegaron a la conclusión de la necesidad de un bus domótico que aportara más funciones que las que aportaban sistemas de aquella época (ON, OFF, DIMMER xx, ALL OFF, etc). Especificaron y desarrollaron un estándar al cual llamaron CEBus (Consumer Electronic Bus). En 1992 fue presentada la primera especificación.

Se trata de un protocolo, para entornos distribuidos de control, que está definido en un conjunto de documentos del estándar. Como es una especificación abierta cualquier empresa puede conseguir estos documentos y fabricar productos que implementen este estándar. En Europa, una iniciativa similar en prestaciones, teniendo en cuenta el mercado al que va dirigido, es el protocolo EHS (European Home System).

Los desarrollos en curso son conducidos por un grupo conocido como el CIC (CEBus Industry Council). El CIC es una organización sin fines de lucro compuesta por representantes de muchas empresas internacionales de electrónica como Microsoft, IBM, Compaq Computer Corp, AT&T Bell Labs, Honeywell, Panasonic, Sony, entre otros.

Aunque no hay ninguna restricción para cualquiera que use el estándar CEBus, el CIC está desarrollando un laboratorio de pruebas sin fines de lucro que será financiado por cargos de certificación. Se alienta a los fabricantes a utilizar el laboratorio de pruebas para verificar la conformidad de su producto y su rendimiento en un entorno de red

doméstica. Cuando el rendimiento está certificado, el fabricante paga una cuota de certificación y tiene licencia para incluir el logotipo de CEBus en su producto.

2.6.2.7. BACnet

BACnet es un protocolo de comunicación de datos diseñado para comunicar entre sí a los diferentes aparatos electrónicos presentes en los edificios actuales. Originalmente diseñado por la ASHRAE actualmente es también un estándar de la ISO y ANSI [146].

El principal objetivo, a finales de los años ochenta, era la de crear un protocolo abierto que permitiera interconectar los sistemas de aire acondicionado y calefacción de las viviendas y edificios con el único propósito de realizar una gestión energética inteligente de la vivienda. Se definió un protocolo que implementaba el estándar OSI y se decidió empezar usando, como soporte de nivel físico, la tecnología RS-485 [145].

La ventaja de este protocolo es el esfuerzo que han realizado para definir un conjunto de reglas de hardware y software que permiten comunicar dos dispositivos, independientemente si estos usan protocolos como el EIB, el EHS, el LonTalk, TCP/IP, etc. El BACnet no busca cerrarse a un nivel físico o a un protocolo de nivel de capa de transporte concreto, realmente lo que pretende definir es la forma en que se representan las funciones que puede hacer cada dispositivo llamadas "objetos", cada una con sus propiedades concretas. Existen objetos con entradas/salidas analógicas, digitales, bucles de control entre otros.

2.6.2.8. EHS

El estándar European Home System (EHS) fue otro de los intentos que la industria europea (año 1984) procuró, auspiciada por la Comisión Europea, buscando crear una tecnología que permitiera la implantación de la domótica en el mercado residencial de forma masiva. El resultado fue la especificación del EHS en el año 1992. Estuvo basada en una topología de red basada en el modelo OSI, y se especificaron los niveles: físico, de enlace de datos, de red y de aplicación.

Desde su inicio estuvieron involucrados los fabricantes europeos más importantes de electrodomésticos de línea marrón y blanca, las empresas eléctricas, las operadoras de telecomunicaciones y los fabricantes de equipamiento eléctrico. La idea era crear un protocolo abierto que permitiera cubrir las necesidades de interconexión de los productos de todos estos fabricantes y proveedores de servicios.

Tal y como fue pensado, el objetivo de la EHS fue cubrir las necesidades de automatización de la mayoría de las viviendas europeas cuyos propietarios no podían permitirse el lujo de usar sistemas más potentes pero también más caros como LonWorks, EIB o BatiBUS, debido fundamentalmente a la mano de obra especializada que exigía su instalación. El EHS viene a cubrir, por prestaciones y objetivos, el espacio que tienen el CEBus norteamericano y el HBS japonés y rebasa las prestaciones del X10 [145].

2.6.3. Procesos de Automatización de Hogares

En los hogares existen una cantidad de tareas, procesos y de quehaceres que se deben atender, muchas de ellas de carácter repetitivo y tedioso. Estos cubren aspectos del uso y administración de los recursos que se tienen en los hogares con el caso de la energía, el agua, los alimentos, entre otros, que presentan una gran oportunidad para poder automatizar en un mayor grado al actual.

Aspectos como el ocio y el confort también requieren de algunas rutinas manuales que pueden manejarse de mejor manera si estos son abordados desde la perspectiva de nuevas formas de interacción y de personalización de experiencias gracias a la recolección de información relacionada a las personas que estén en el hogar.

Por último los aspectos de seguridad y accesibilidad son altamente potenciados gracias al uso de la tecnología. Un conjunto de dispositivos y sistemas que automáticamente perciban situaciones de peligro y vigilancia para los residentes de un hogar y que pueda tomar las previsiones necesarias y generar los avisos de alerta, harán que los peligros se reduzcan aún más. Las personas con algún tipo de dificultad o discapacidad podrán aprovechar las posibilidades que les brinda un ambiente más amigable y adecuado, gracias a dispositivos que los ayuden a llevar una vida lo más normal posible.

2.6.3.1. Administración y Monitoreo de Recursos

La gestión de los recursos básicos es una tarea primordial dentro de los hogares. El consumo y el monitoreo energético, del agua y de alimentos no solo se puede cuantificar sino que también representa parte fundamental de la administración general de las familias. El conocer cómo se usan dichos elementos abre la puerta a mejorar el consumo o administración de los mismos, optimizando los procesos involucrados. Si agregamos una capa tecnológica, podemos utilizar la información generada para mejorar y conocer los patrones de consumo y optimizar cada elemento posible de esta cadena de manera automatizada.

De esta manera podemos hablar de las diversas estrategias que se pueden llevar a cabo, según el recurso involucrado:

- Gestión en el consumo energético: La electricidad y el gas son elementos de suma importancia en el hogar actual. En los hogares casi todos los artefactos requieren de una fuente eléctrica, con lo que es uno de los recursos más utilizados. El poder conocer de manera concienzuda cuales artefactos o habitaciones realizan el mayor consumo eléctrico, detallando los horarios de uso e incluyendo el poder reconocer patrones, permite utilizar y establecer de manera inteligente, políticas automatizadas para la regulación en el consumo de este recurso. Del mismo modo, el registrar información de uso del gas en cocinas y sistemas de calefacción permite establecer dichos patrones y así no solo regular automáticamente el uso, sino establecer niveles de eficiencia energética deseables y tomando en cuenta la inclusión de perfiles según gustos y necesidades de usuarios. El poder agregar sensores de medición en las infraestructura de los hogares, en los dispositivos y la comunicación entre ellos y sistemas de análisis en tiempo real también puede

usarse para ayudar a monitorear de manera remota cual es el estado actual de los artefactos de nuestro hogar, permitiendo incluso desactivarlos o activarlos sin requerir de presencia de personas.

- Manejo de recursos hídricos: El agua es considerado un recurso crítico, no solo en los hogares, sino en donde sea que este es utilizado. Este valioso recurso es utilizado ampliamente en la alimentación, en la higiene personal y de la vivienda, entre otros y muchas veces es subestimado su acceso y consumo, pero las crecientes dificultades en el suministro confiable, la preocupación por la contaminación del vital líquido entre otras razones hace cada vez más necesario la implantación de sensores que permiten conocer en todo momento los niveles existentes, los patrones de uso y la calidad del mismo.

- Alimentos y desperdicios: Los adquisición de alimento o reposición de los mismos es una de las formas en las que al introducir artefactos con características de comunicación y procesamiento de información ayudan a sumar esfuerzos en pro de la automatización de los hogares. Neveras con capacidad de reconocer cuando se han acabado la o las existencias de uno o más productos o si un determinado alimento ha expirado, permite generar listas de compras o realizar los pedidos correspondientes, teniendo en cuenta los perfiles de gustos de las personas e incluso de su dieta. Por otro lado la gestión correcta de los desechos da como resultado ser conscientes con el ambiente y también ayuda a clasificar los elementos reciclables automáticamente.

2.6.3.2. Entretenimiento y Confort

Los sistemas de entretenimiento han sido desde el principio parte del desarrollo de la domótica. Muchos elementos han ido promocionando la integración de televisiones, reproductores de música y otros con controles universales o con teléfonos inteligentes, tabletas o computadoras. Cada vez más, los artefactos de entretenimiento se pueden conectar a Internet y sincronizarse con servicios de suscripción de música, de películas y series, con nuestros archivos multimedia en las computadoras casi sin requerir configuración alguna de parte de los usuarios.

Por otro lado, los sistemas de iluminación, HVAC, entre otros también agregan una capa de confort a los hogares automatizados y que en la actualidad se hace más común que el resto de elementos del hogar inteligente, ofreciendo capas de personalización importante a aspectos tradicionales de los hogares.

2.6.3.3. Seguridad

Otro aspecto fundamental en la incorporación de tecnología en el hogar ha sido el uso de sensores, cámaras de circuito cerrado y alarmas de seguridad. En cuanto a la domótica se refiere, estas tecnologías han ido evolucionando en capacidad y en utilidad, contando ahora con características de comunicación tanto con el usuario, como con servicios policiales y/o emergencia y también con otros dispositivos y artefactos.

Con la aparición de sistemas de cámaras de circuito cerrado que pueden estar funcionando de manera autónoma o activarse bajo eventos y con los cuales se pueden observar no solo desde un dispositivo sino desde cualquier lugar, utilizando Internet para ello. Permite elevar los niveles de seguridad y confianza en los usuarios, pero esta no es la última línea de defensa contra delincuentes o vandalismo: redes de sensores de movimiento de infrarrojo o ultra sonido ubicados estratégicamente al rededor y dentro el hogar permite conocer, incluso de manera remota si existe algún peligro de intrusión no deseado, pudiendo activar las alarmas o alertar a las autoridades competentes de manera automática.

El uso de timbres y sistemas intercomunicadores que alertan a los usuarios donde quieran que estén de las personas que se encuentran en la puerta del hogar hacen más fácil la tarea de reconocer posibles amenazas de personas que no se conozcan, o por el contrario, permitir que personas autorizadas puedan entrar sin la necesidad de recibirlas presencialmente.

Pero también existen otros peligros en los hogares que no necesariamente vienen dados por riesgo de robos, intrusiones, también los sistemas de seguridad han cambiado para agregar maneras inteligentes de reconocer otros riesgos potenciales para las familias. Las alarmas de seguridad ahora no solo se activan bajo entradas no autorizadas sino también cuando los detectores de humo y fuego indican la presencia de un incendio y notificar a los servicios de urgencias del riesgo potencial; también es posible detectar cuando la calidad del aire dentro de una o más habitaciones no es la adecuada.

En el caso de familias que poseen niños pequeños también la seguridad a nivel de procesos de hogares automatizados ha mejorado, con la introducción de monitores y cunas de bebes inteligentes, las cuales permiten monitorear en tiempo real y generar estadísticas sobre posibles factores de riesgo en los infantes. También esto es aplicable a otros aspectos del hogar para evitar accidentes, como es el caso de notificaciones de acceso a áreas de los hogares sin la supervisión adulta en piscinas, desvanes, sótanos, jardines, e incluso de artefactos, permitiendo a los padres y responsables tomar medidas tempranas antes de que pudiese ocurrir un accidente.

2.6.3.4. Accesibilidad

La accesibilidad dentro y fuera de los hogares es un aspecto que ha estado en desarrollo desde el principio en la domótica. Las personas que poseen algún tipo de discapacidad muchas veces enfrentan grandes retos en el uso de los espacios y recursos dentro de sus viviendas y una manera ideal de minimizar estas dificultades es automatizando gran parte de las tareas que se puedan realizar, desde la limpieza realizada por robots autónomos o con la creación de nuevas maneras de interactuar (interfaces) que permitan controlar los artefactos dentro del hogar.

Personas que poseen alguna discapacidad motriz pueden valerse de asistentes de voz y de sistemas robóticos permiten al usuario encargase de las tareas de limpieza y organización. Del mismo modo las personas con discapacidad visual pueden aprovechar las cualidades cognitivas de inteligencia artificial en los artefactos para realizar las labores,

utilizando por ejemplo, comandos de voz.

Otro sector de la población que se ve beneficiado de la introducción de más características inteligentes en los hogares son las personas de la tercera edad, pues cada vez es más común el uso de múltiples interfaces por las cuales controlar y realizar tareas del hogar, gracias a artefactos inteligentes que rompen con el esquema de una curva de aprendizaje elevada.

2.6.4. Tecnologías de Automatización

Existen una gran cantidad formas y presentaciones por las cuales la tecnología se hace presente en los hogares automatizados y en el área de la domótica. Se puede abordar desde el punto de vista de artefactos tecnológicos que han adquirido capacidades de comunicación o de cognición o desde el punto del software, hardware e infraestructuras utilizadas para obtener un hogar con aplicaciones automáticas.

2.6.4.1. OpenHAB

El open Home Automation Bus (openHAB) [148] es una plataforma de automatización domótica agnóstica de código abierto que funciona como el centro de una casa inteligente [149]. El objetivo principal que busca cubrir openHAB es el integrar diferentes sistemas y tecnologías de domótica en una única solución que permite reglas de automatización globales y que ofrece interfaces de usuario uniformes.

Como software que gobierna otros sistemas openHAB está capacitado con las siguientes características:

1. Está diseñado para ser absolutamente neutral ante los fabricantes, el hardware o los protocolos implementados.
2. Puede ejecutarse en cualquier dispositivo que sea capaz de usar una Java Virtual Machine (JVM) incluyendo sistemas operativos Linux, Mac, Windows.
3. Le permite integrar una gran cantidad de tecnologías domóticas diferentes en una sola.
4. Tiene un potente motor de reglas para satisfacer todas las necesidades de automatización requeridas.
5. Viene con diferentes interfaces de usuario basadas en la Web, así como interfaces de usuario nativas para iOS y Android.
6. Es totalmente código abierto.
7. Es mantenido por una comunidad apasionada y creciente.
8. Es fácilmente extensible para integrarse con nuevos sistemas y dispositivos.
9. Proporciona APIs para integrarse con otros sistemas.

OpenHAB no busca reemplazar las soluciones domóticas existentes, por lo que puede considerarse como un sistema de sistemas. Por lo tanto, asume que los subsistemas se configuran y existen de manera independiente a openHAB, ya que esto es a menudo un asunto muy específico y complejo (incluyendo procesos de "emparejamiento", enlaces de dispositivos directos, etc). En cambio, openHAB se centra en el "uso diario"de las cosas y los resúmenes de los propios dispositivos.

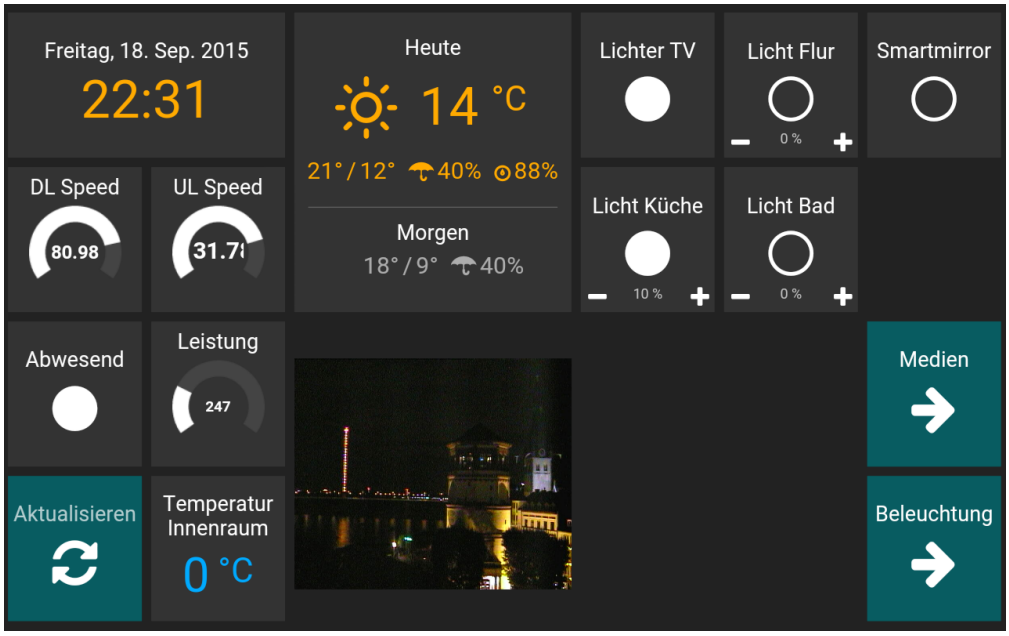

Figura 2.37: Dashboard de OpenHab

Un concepto central para openHAB es la noción de un "item". Un item es un bloque de construcción atómico funcional centrado en los datos. OpenHAB no le importa si un item (por ejemplo, un valor de temperatura) está relacionado con un dispositivo físico o alguna fuente como un servicio web o como un resultado de cálculo. Todas las características ofrecidas por openHAB utilizan esta abstracción de item, lo que significa que no encontrará ninguna referencia a las cosas específicas del dispositivo (como direcciones IP, ID, etc) en las reglas de automatización, las definiciones de la interfaz de usuario y así sucesivamente. Esto hace que sea perfectamente fácil reemplazar una tecnología por otra sin hacer ningún cambio en las reglas e interfaces de usuario.

Un aspecto muy importante de la arquitectura de openHAB es su diseño modular. Es muy fácil agregar nuevas características (como la integración con otro sistema a través de una "vinculación") y puede agregar y eliminar tales características en tiempo de ejecución. Este enfoque modular ha sido un aliciente enorme para atraer a una comunidad activa alrededor de openHAB con muchos colaboradores comprometidos.

OpenHAB es altamente flexible y personalizable, pero esto trae como contra que se tiene que invertir tiempo para aprender sus conceptos y para establecer un sistema individual adaptado a las necesidades propias. No es un producto comercial que se conecta y está listo para usar y muchas partes del software requieren configuración y

el acceso potencial a los archivos de registro. Por lo tanto, la configuración de openHAB es principalmente un trabajo para personas con conocimientos en las tecnologías implicadas [150].

2.6.4.2. Home Assistant

Home Assistant es una plataforma de domótica de código abierto que escrita en Python 3. Su objetivo es el poder otorgar la capacidades de rastrear y controlar todos los dispositivos en el hogar y poder establecer control automatizado [151]. Home Assistant se basa en tres pilares fundamentales:

- Observación: Home Assistant rastreará el estado de todos los dispositivos en el hogar, por lo que no es necesario intervención humana alguna.
- Control: El objetivo es controlar todos tus dispositivos desde una interfaz única, compatible con dispositivos móviles. Home Assistant no requiere almacenar ninguno de los datos en la nube, buscando mantener su privacidad.
- Automatización: El usuario establece las reglas, incluso de carácter avanzado para controlar dispositivos y mantener su hogar activo.

Al ser escrito en Python 3, este software es altamente portable y es capaz de ser ejecutado en computadores de capacidades modestas o incluso distribuir el sistema entre varios nodos. También al utilizar APIs, este puede ser extendido de la manera que se requiera, agregando componentes modulares a su arquitectura. La manera en que Home Assistant muestra la información de los diferentes dispositivos es través de una interfaz web, lo que lo hace fácil de acceder desde cualquier dispositivo que este dentro de la misma red.

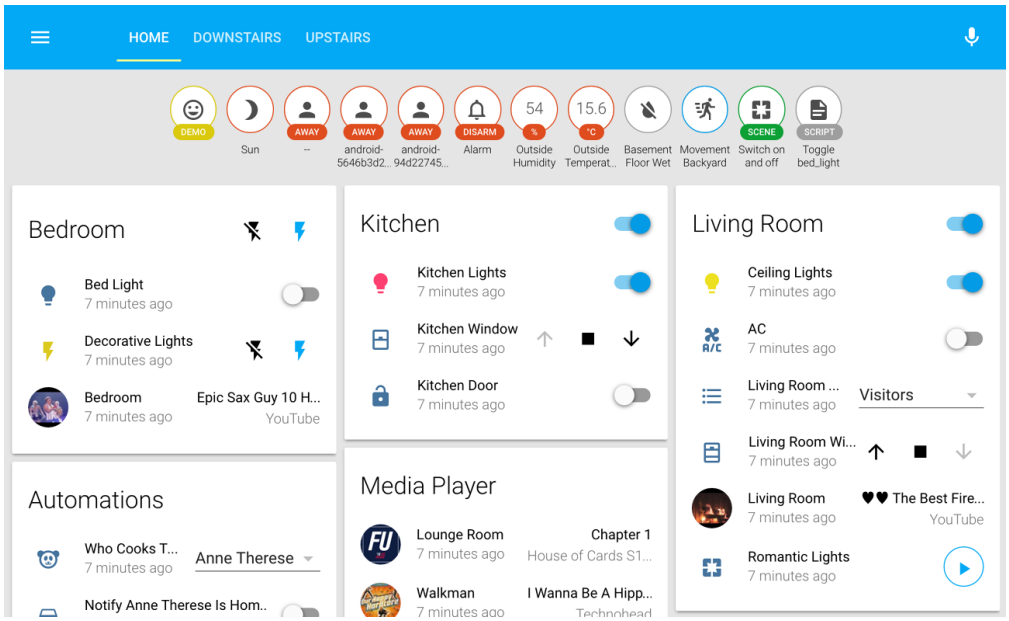

Figura 2.38: Dashboard de Home Assistant

Al igual que openHAB posee una alta integración con dispositivos de más de 700 marcas [152]. También comparte su principal inconveniente que es su difícil configuración, la cual requiere de conocimientos medios o avanzados sobre las tecnologías implicadas o de los dispositivos que se desean utilizar a través de este software.

2.6.4.3. Domoticz

Domoticz es un sistema de domótica muy ligero que le permite monitorear y configurar diversos dispositivos, incluyendo luces, interruptores, varios sensores como temperatura, lluvia, viento, radiación ultravioleta (UV), uso o producción de electricidad, consumo de gas, consumo de agua, entre otros. Las notificaciones y alertas se pueden enviar a cualquier dispositivo móvil [153].

Domoticz está escrito en C ++ y fue lanzado al mercado en el año 2012. El sistema está diseñado para operar en varios sistemas operativos (Linux, Windows, Mac, dispositivos programados). La interfaz de usuario es un front-end web HTML5 escalable y se adapta automáticamente para dispositivos de escritorio y móviles [154]. Utiliza su propio servidor web integrado, escrito también en C ++, para una ejecución eficiente y evitar dependencias.

La lógica de manejo de eventos de los dispositivos es programable por el usuario, ya sea mediante Blockly (para codificación visual usando gráficos de interconexión) o Lua (un lenguaje de programación de guiones, ideal para soluciones integradas).

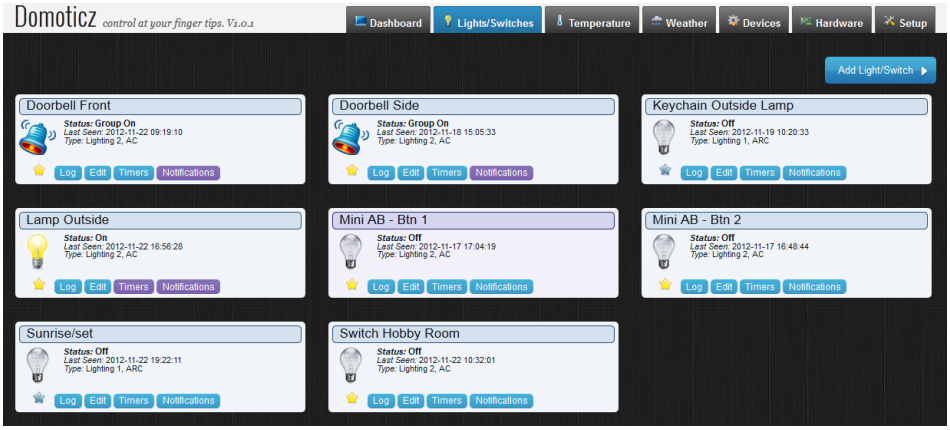

Figura 2.39: Dashboard de Domoticz

La principal desventaja de Domoticz es que es una solución de domótica comparativamente más difícil de configurar y extender que sus contrapartes openHAB y Home Assistant, aunque su base de datos de sensores, actuadores y dispositivos es mayor.

2.6.4.4. Android Things

Android Things es un sistema operativo para dispositivos embebidos y para micro computadores basado en el sistema operativo Android y parte del ecosistema de soluciones tecnológicas de Google. Nace del proyecto Brillo y evoluciona adecuándose

con otros proyectos de la compañía como Android Auto, Android Wear, Android TV o Chrome OS bajo un kit de desarrollo estándar (SDK) [155] que provee APIs y librerías que faciliten la creación de aplicaciones interoperables.

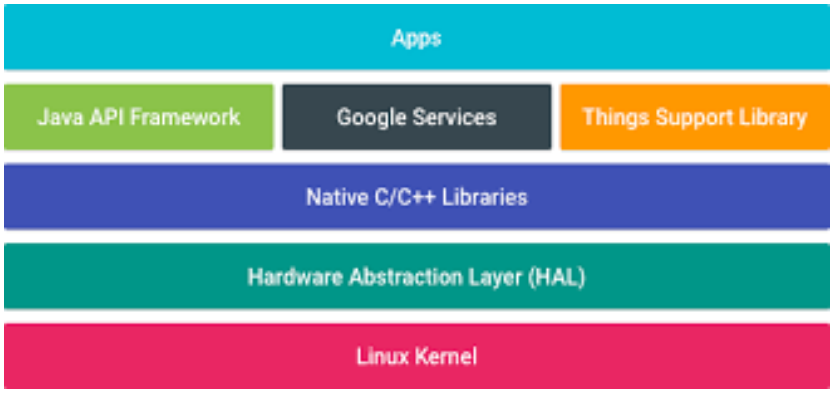

Figura 2.40: Arquitectura de Android Things

Android Things está pensado para usar pocos recursos comparados con otros sistemas operativos e implementa un protocolo propio llamado Weave con el cual se busca realizar la comunicación entre dispositivos bajo el sistema operativo Android de manera transparente y automática.

La principal ventaja de Android Things es su robusto SDK para el desarrollo de aplicaciones de forma que los programadores y desarrolladores pueden crear aplicaciones fácilmente, especialmente si se posee alguna experiencia desarrollando aplicaciones para algún producto del ecosistema Android además de la integración que poseen las aplicaciones que se se hayan desarrollado, permitiendo que nuevos productos sean creados y probados muy rápidamente.

La principal desventaja que posee el sistema operativo de Google es que su esquema de desarrollo es rígido comparado con otros sistemas operativos para dispositivos embebidos o enfocados en el Internet de las Cosas, dejando de lado otros protocolos y estándares de la industria lo cual podría causar problemas con sistemas y con dispositivos legados o antiguos.

2.6.4.5. Amazon Echo

Amazon Echo es un altavoz manos libres controlado con la voz, desarrollado por la empresa norteamericana Amazon, como una manera de llevar a los hogares su inteligencia artificial Amazon Alexa, dando la posibilidad a los usuarios de reproducir música desde servicios en la nube, realizar y recibir llamadas y mensajes, proporcionar información, noticias, resultados deportivos, el clima y más, al instante tan solo preguntando o usando comandos de voz.

Alexa es el asistente de voz construido por la compañía, basada en inteligencia artificial alojada en la nube por lo que siempre se hace más inteligente. Cuanto más usa Echo, más se adapta a sus patrones de habla, vocabulario y preferencias personales.

Figura 2.41: Amazon Echo y Amazon Echo Dot

Como Echo está diseñado para estar siempre conectado, las actualizaciones se entregan automáticamente, agregando de manera constante nuevos servicios y opciones de interacción con otros servicios o con otros dispositivos.

En cuanto al aspecto de conectividad, Amazon Echo cuenta con conexión WiFi de doble banda y doble antena (MIMO) para una transmisión más rápida. Soporta redes WiFi 802.11a/b/g/n. Además también posee conectividad Bluetooth con soporte de perfil de distribución de audio avanzado (A2DP) para la transmisión de audio desde dispositivos móviles a Amazon Echo y AVRCP (Audio/Video Remote Control Profile) para control de voz de dispositivos móviles conectados [156].

Figura 2.42: Amazon Echo View

2.6.4.6. Google Home

Google Home es un altavoz activado por voz fabricada por Google para competir en el ramo de los asistentes de voz en los hogares. Utiliza la inteligencia artificial Google Assistant para poder responder a una gran cantidad de comandos [157]. La bocina inteligente puede determinar por la voz, cual fue el usuario que emitió algún comando u orden, permitiendo así, relacionar la información con los perfiles de usuario con sus cuentas de Google.

La característica principal del Google Home es la alta integración con los servicios de Google como el correo, servicios de música, mapas y tráfico, gustos, vídeos, siendo capaz de enviar la información a los dispositivos de manera conveniente. Como su competencia Amazon Echo, Google Home también provee servicio de llamadas y de mensajería a listas de contactos registrados y ya una gran cantidad de compañías ya

Figura 2.43: Google Home

han anunciado que proveerán de servicios y de dispositivos que podrán ser controlados a través del altavoz.

Google Assistant es la asistente virtual que evoluciono del uso del primer asistente creado por la compañía llamado Google Now y del agregar nuevas capacidades a la inteligencia artificial desarrollada por ellos. Este asistente puede acceder de a la información que se tenga disponible de las cuentas de Google con las cuales se sincroniza dando a los usuarios una experiencia bastante personalizada.

Google Home utiliza conexión WiFi con soporte para 802.11b/g/n/ac bajo 2.4GHz/5Ghz para transmisión y recepción al mismo tiempo en sus antenas (MIMO), pero a diferencia de Amazon Alexa, no cuenta con conectividad Bluetooth.

2.6.4.7. Apple HomePod

El Apple HomePod es la respuesta de la compañía Apple a los altavoces Google Home y Amazon Echo. Apple HomePod está especialmente diseñado para dos tareas, la de poder reproducir música con una gran calidad de audio y el poder integrarse con los servicios de la compañía, haciendo uso de su asistente virtual, Siri.

Figura 2.44: Apple HomePod

Este es el primer producto de Apple que se afirma en la nueva plataforma HomeKit para productos en el hogar, pero desde ya este ofrece una gran una gran cantidad de empresas y marcas cuyos productos podrán ser controlados haciendo uso de este dispositivo. El poder de la inteligencia artificial del asistente virtual Siri, brinda una

importante parte de la experiencia del usuario, pues este podrá reconocer más comandos de voz aun que los previstos en los dispositivos iOS o con MacOS.

Apple HomePod atiene una fecha estimada de salida al mercado para diciembre de 2017, sin embargo se conocen de sus especificaciones técnicas que en el apartado de conectividad utiliza conexiones WiFI 802.11a/b/g/n/ac con capacidad MIMO y también el protocolo AirPlay 2, el cual es el protocolo propietario de la marca.

2.6.4.8. Raspberry Pi

Raspberry Pi es una marca de micro computadores de hardware libre, que comenzó como proyecto para llevar a la educación de escolares ingleses conceptos de la computación, de la programación y de la electrónica, y que luego la comunidad Maker y algunas compañías adoptaron, pues vieron el potencial para crear dispositivos, sensores, actuadores, robots entre otros y para poder llevar a cabo prototipos funcionales de dispositivos y de productos muy rápidamente. Su bajo costo de $35 por la placa más costosa y facilidad de uso, han elevado la adopción de este micro computador por personas en todo el mundo.

Este es un proyecto llevado a cabo con la Raspberry Pi Foundation [158] desde el año 2012 ofreciendo hasta el día de hoy siete modelos distintos de placas, con procesadores ARM, memoria RAM que va desde los 256MB hasta 1GB y cuya memoria es de carácter externo usando tarjetas MicroSD en la mayoría de los modelos y cuyo consumo no es mayor al de 2,5 amperios. Los modelos son:

- Raspberry Pi Modelo A: Fue el primer modelo de Raspberry Pi en salir al mercado, en el año 2012. Basado en un SoC Broadcom BCM28235, cuyo procesador es un ARM11 32 bits a 700MHz, Gráficas Broadcom VideoCore IV, con 256MB de memoria RAM, un puerto USB, una salida HDMI, un conector RCA, una entrada CSI para un módulo de cámara, y sin características de conectividad alguna por defecto. Para el almacenamiento se usan tarjetas SD.

- Raspberry Pi Modelo B/B+: También del año 2012, es una variante del Modelo A, trajo consigo diversas mejoras, como la inclusión del doble de memoria RAM, pasando de 256MB a 512MB. Trajo consigo un puerto USB más y un conector Ethernet (RJ-45) Se mantuvo tanto su tamaño como su coste. No hubo variaciones en el procesador ni en la gráfica. Tiempo después se lanzó el Modelo B+, que incluyó 4 puertos USB y pasó de usar una SD a una MicroSD.

- Raspberry Pi 2 Modelo B: Lanzada en 2014 es el primer modelo que no incluye el mismo procesador usado en los tres anteriores: se sustituye por uno de la misma marca, pero de modelo BCM2836 con lo cual pasa de ser de un núcleo a cuatro, y de 700MHz a 900MHz, no obstante emplea la misma gráfica, la VideoCore IV. Dobla la cantidad de memoria RAM, pasando de 512MB a 1GB de memoria (también compartida con la gráfica). También incluye 40 pines GPIO, y mantiene los cuatro puertos USB. Suprime la conexión RCA.

- Raspberry Pi 3 Modelo B: Sale al mercado en el año 2016, renovando procesador, una vez más de la compañía Broadcom, siendo un procesador de cuatro núcleos

al igual que el modelo anterior, pero pasa de 900MHz a 1.20GHz, manteniendo la RAM en 1GB. Su mayor novedad fue la inclusión de WiFi y Bluetooth (4.1 Low Energy) sin necesidad de adaptadores.

- Raspberry Pi Zero: Fue el primer modelo miniaturizado de las Raspberry Pi, teniendo un tamaño un poco mayor a un Pen Drive. Lanzado en 2015 con un coste de 5 dólares, es una 40 % más potente que el primer modelo de Raspberry. Tiene un CPU Broadcom BCM2835, que funciona a 1GHz con dos núcleos. Posee 512MB de RAM, y comparte la gráfica VideoCore IV. Debido a su tamaño sustituye el puerto HDMI por MiniHDMI, manteniendo así las prestaciones. Tampoco usa USB estándar, sino que tiene dos MicroUSB, uno de alimentación y otro de datos. Posee salida RCA, pero en vez de por clavija son solo dos conectores integrados en la placa. Usa MicroSD como sistema de almacenamiento.
- Raspberry Pi Zero W: Es la sucesora de la Raspberry Pi Zero. La W es por Wireless, ya que la única novedad de esta placa con respecto a su antecesora es la inclusión de WiFi y Bluetooth, por lo que su precio ascendió a 11 dólares.

Existen gran variedad de sistemas operativos que pueden ser usados con este micro computador, la mayoría de ellos, basados en Linux, pero también con la posibilidad de instalar Windows 10 (Iot Core) o Android (Android Things).

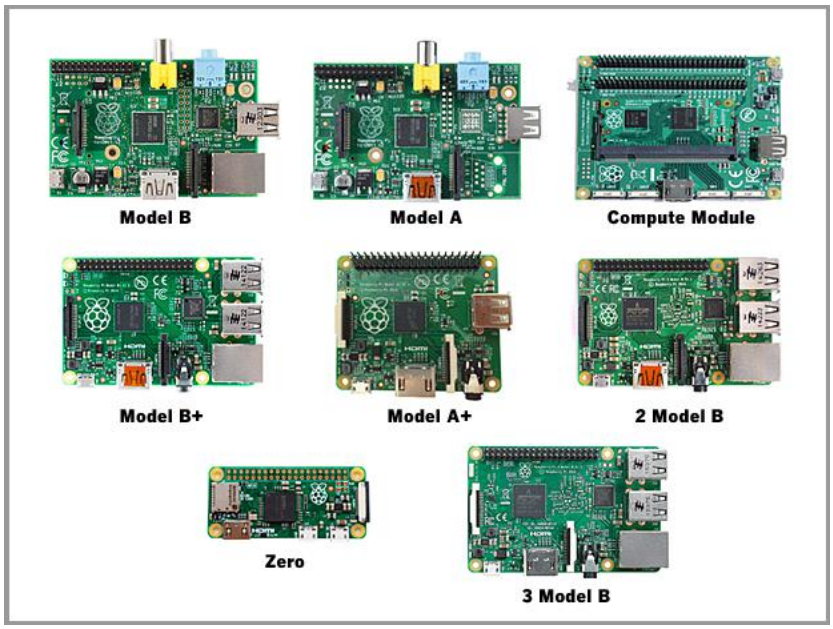

Figura 2.45: Generaciones distintas de Raspberry Pi

Una característica principal de estas placas es la existencia de 40 pines GPIO, los cuales proveen entradas y salidas digitales para el micro computador con lo que se le puede expandir su s funcionalidades con el uso de sensores (HATs o a través de circuitos eléctricos tradicionales), cuya lógica puede ser creada en lenguajes de programación como Python, C++, Java, entre otros.

El proyecto ha ganado una comunidad muy amplia alrededor del mundo, que han probado la versatilidad que posee este micro computador en proyectos de toda índole, dando a las personas la capacidad de crear soluciones basadas en el Internet de las Cosas de una forma divertida y a su vez, con calidad y robustez.

2.6.4.9. Arduino/Genuino

Arduino (Genuino a nivel internacional) es una plataforma electrónica en hardware y software libres, fáciles de usar. Está pensado para cualquier persona que haga proyectos interactivos. [159] Arduino detecta el ambiente recibiendo entradas de muchos sensores, y afecta su entorno controlando luces, motores y otros actuadores. La lógica de micro controlador Arduino de deberá hacer se hace escribiendo código en el lenguaje de programación Arduino (basado en C++) y utilizando el entorno de desarrollo de Arduino.

Arduino nació en el Ivrea Interaction Design Institute como una herramienta fácil para el prototipado rápido, dirigido a estudiantes sin formación en electrónica y programación. Tan pronto como llegó a una comunidad más amplia, la junta de Arduino comenzó a cambiar para adaptarse a las nuevas necesidades y desafíos, diferenciando su oferta de simples placas de 8 bits a productos para aplicaciones IoT, portátiles, impresión 3D y dispositivos embebidos. Todas las placas de Arduino son totalmente de hardware libre, lo que permite a los usuarios construirlas independientemente y eventualmente adaptarlas a sus necesidades particulares. El software, también es de código abierto y está creciendo a través de las contribuciones de los usuarios de todo el mundo.

Las placas Arduino están disponibles de dos formas: ensambladas o en forma de kits “Hágalo tú mismo”. Los esquemas de diseño del hardware están disponibles bajo licencia Libre, con lo que se permite que cualquier persona pueda crear su propia placa Arduino sin necesidad de comprar una prefabricada. La primera placa Arduino fue introducida en 2005, ofreciendo un bajo costo y facilidad de uso para novatos y profesionales. Buscaba desarrollar proyectos interactivos con su entorno mediante el uso de actuadores y sensores. A partir de octubre de 2012, se incorporaron nuevos modelos de placas de desarrollo que usan micro controladores Cortex M3, ARM de 32 bits, que coexisten con los originales modelos que integran micro controladores AVR de 8 bits.

En la actualidad, existen 23 distintas placas de micro controladores, 7 módulos complementarios, 15 Shields (placas de expansión), con lo cual se posee un ecosistema rico de diversos micro controladores para cada tarea o cada ambiente de desarrollo.

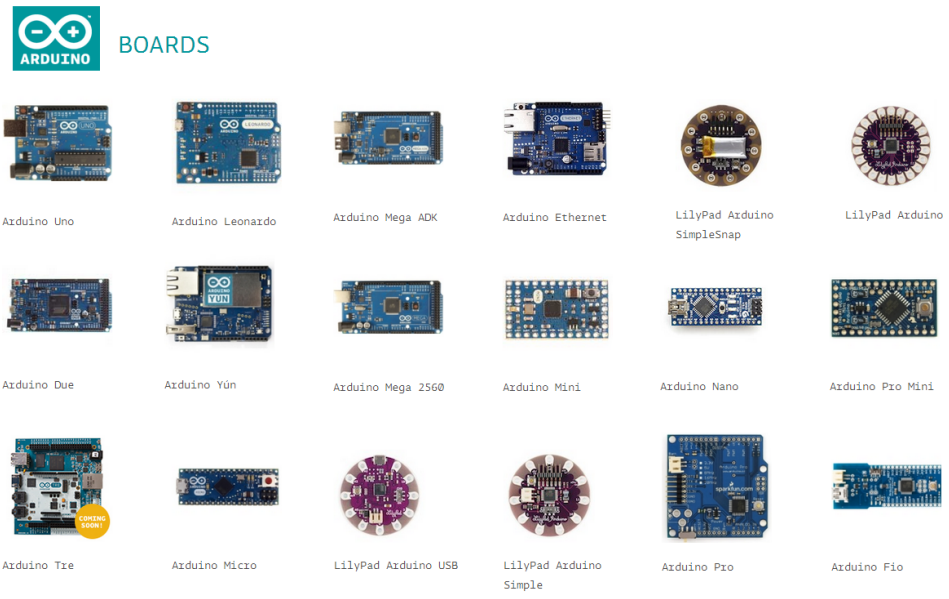

Figura 2.46: Placas Fabricadas por Arduino

Capítulo 3

Marco Metodológico

3.1. Metodología Fundamental para la Ciencia de Datos

La ciencia de datos como campo que busca resolver problemas y dar respuesta a preguntas a través del análisis de datos es una práctica que no es nueva y que debe proveer estrategias eficaces para poder obtener resultados no solamente precisos, sino también acordes con el método científico de una investigación y que permitan a las organizaciones utilizar lo obtenido para tomar acciones que mejoren los resultados futuros.

Con la cada vez mayor cantidad de tecnologías para analizar datos y construir modelos, existe la necesidad de tener un enfoque que ayude a la automatización de muchos de los pasos en su construcción, haciendo que tecnologías de aprendizaje automatizado sean más accesibles a quienes carecen de habilidades estadísticas.

A medida que las capacidades de análisis de datos se vuelven más accesibles, los científicos necesitan una metodología fundacional capaz de proporcionar una estrategia de guía, independientemente de las tecnologías, volúmenes de datos o enfoques involucrados. Para ello IBM ideo una metodología para abordar los problemas de ciencia de datos en 10 pasos, que abarca tanto tecnologías como enfoques "'de arriba hacia abajo" o "de abajo hacia arriba".

Esta metodología tiene algunas similitudes con metodologías utilizadas para la minería de datos, pero hace hincapié en varias de las nuevas prácticas en la ciencia de datos, tales como el uso de grandes volúmenes de datos, la incorporación de análisis de texto en los modelos de predicción y la automatización de algunos procesos. [160] La metodología consta de 10 etapas que forman un proceso iterativo para el uso de datos para descubrir ideas:

1. Comprensión del Negocio: Cada proyecto comienza con la comprensión del negocio. Los patrocinantes de los negocios que necesitan la solución analítica desempeñan el papel más crítico en esta etapa al definir el problema, los objetivos del proyecto y los requisitos de la solución desde una perspectiva empresarial. Esta primera etapa establece las bases para una resolución exitosa del problema. Para

ayudar a garantizar el éxito del proyecto, los patrocinadores deben estar involucrados en todo el proyecto para proporcionar experiencia en el dominio, revisar los resultados intermedios y garantizar que el trabajo permanezca en el buen camino para generar la solución deseada.

2. Enfoque analítico: Una vez el problema del negocio sea claramente especificado, el científico de datos puede definir el enfoque analítico para el problema. Esta etapa implica expresar el problema en el contexto de las técnicas estadísticas y de aprendizaje automatizado, para que la organización pueda identificar las más adecuadas para obtener el resultado deseado.

3. Requerimientos en los datos: El enfoque analítico elegido determina los requisitos en los datos, es decir, luego de definidos los métodos analíticos a utilizar, establecer que se requiere de los datos, formatos y representaciones, guiados por el conocimiento del dominio.

4. Recolección de datos: En la etapa inicial de recopilación de datos, los científicos de datos identifican y recopilan los recursos de datos disponibles, sin importar su tipo (estructurados, no estructurados y semi-estructurados) relevantes para el dominio del problema. Normalmente, se debe elegir si realizan inversiones adicionales para obtener elementos de datos menos accesibles. Si hay vacíos en la recopilación de datos, el científico de datos puede tener que revisar los requisitos de datos en consecuencia y recopilar nuevos datos y/o en mayores cantidades. Mediante la incorporación de más datos, los modelos predictivos pueden ser más capaces de representar eventos raros como la incidencia de la enfermedad o el fracaso del sistema.

5. Entender los datos: Después de la recopilación de datos original, el utilizar técnicas de estadística descriptiva y de visualización para comprender el contenido de los datos, evaluar la calidad de los mismos y descubrir conocimientos iniciales sobre los ellos. La recolección de datos adicionales puede ser necesaria para llenar los vacíos.

6. Preparación de los datos: Esta etapa abarca todas las actividades para construir los datos que se utilizarán en la siguiente etapa de modelado. Las actividades de preparación de datos incluyen la limpieza de datos (tratar valores perdidos o no válidos, eliminar duplicados, formatear adecuadamente), combinar datos de múltiples fuentes (archivos, tablas, plataformas) y transformar datos en variables más útiles. En un proceso denominado ingeniería de características, los científicos de datos pueden crear variables explicativas adicionales, también denominadas predictivas o características, a través de una combinación de conocimientos de dominio y las variables estructuradas existentes. La preparación de los datos suele ser el paso más importante y largo de un proyecto de ciencias de datos. En muchos dominios, algunos pasos de preparación de datos son comunes en diferentes problemas. Automatizando ciertas etapas de preparación de datos de antemano pueden acelerar el proceso minimizando el tiempo de preparación ad-hoc. Con los actuales sistemas de alto rendimiento y la funcionalidad analítica en los que

se almacenan los datos, los científicos de datos pueden preparar datos con más facilidad y rapidez utilizando conjuntos de datos muy grandes.

7. Modelado: A partir de la primera versión del conjunto de datos preparado, la etapa de modelado se centra en el desarrollo de modelos predictivos o descriptivos de acuerdo con el enfoque analítico previamente definido. Con modelos predictivos, los científicos utilizan un conjunto de formación (datos históricos en los que se conoce el resultado del interés) para construir el modelo. El proceso de modelado es típicamente interactivo ya que las organizaciones obtienen ideas intermedias, lo que lleva a refinamientos en la preparación de los datos y la especificación del modelo. Para una técnica dada, los científicos de datos pueden probar múltiples algoritmos con sus respectivos parámetros para encontrar el mejor modelo para las variables disponibles.

8. Evaluación: Durante el desarrollo del modelo y antes del despliegue, el científico de datos evalúa el modelo para entender su calidad y asegurarse de que aborda adecuadamente y completamente el problema del negocio. La evaluación del modelo implica el cálculo de diversas medidas de diagnóstico y otros resultados tales como tablas y gráficos, lo que permite al científico de datos interpretar la calidad del modelo y su eficacia para resolver el problema. Para un modelo predictivo, los científicos de datos utilizan un conjunto de pruebas, que es independiente del conjunto de entrenamiento, pero sigue la misma distribución de probabilidad y tiene un resultado conocido. El conjunto de pruebas se utiliza para evaluar el modelo para que pueda refinarse según sea necesario. A veces el modelo final se aplica también a un conjunto de validación para una evaluación final. Además, los científicos de datos pueden asignar pruebas de significación estadística al modelo como prueba adicional de su calidad.

9. Despliegue: Una vez que se ha desarrollado un modelo satisfactorio y es aprobado por los patrocinadores de negocios, se despliega en el entorno de producción o un entorno de prueba comparable. Por lo general, se despliega de forma limitada hasta que se ha evaluado completamente su rendimiento. El despliegue puede ser tan simple como generar un informe con recomendaciones o involucrarse como en un flujo de trabajo complejo y un proceso de puntuación gestionado por una aplicación personalizada. El despliegue de un modelo en un proceso empresarial operacional suele implicar grupos, habilidades y tecnologías adicionales dentro de la empresa.

10. Feedback: Mediante la recopilación de los resultados del modelo implementado, la organización obtiene retroalimentación sobre el rendimiento del modelo y su impacto en el entorno en el que se implementó. El análisis de esta retroalimentación permite a los científicos de datos refinar el modelo para mejorar su precisión y utilidad. Pueden automatizar parte o la totalidad de los pasos de recopilación de información y evaluación de modelos, refinamiento y redistribución para acelerar el proceso de actualización de modelos para obtener mejores resultados.

El flujo de la metodología ilustra la naturaleza iterativa del proceso de resolución de problemas. A medida que los científicos de los datos aprenden más sobre los datos y el

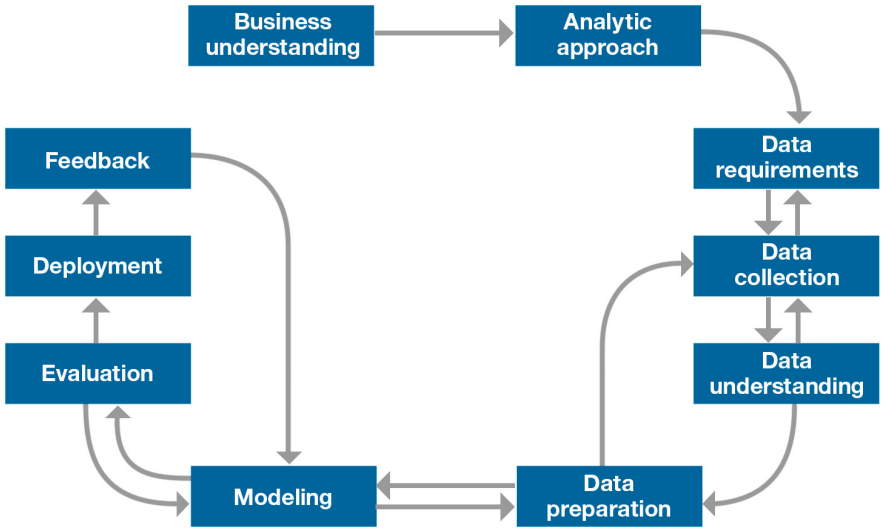

Figura 3.1: Flujo de Actividades en la Metodología Fundamental para la Ciencia de Datos

modelado, frecuentemente regresan a una etapa previa para hacer ajustes. Los modelos no se crean una sola vez, se despliegan y se dejan en su sitio tal cual, ni se mantienen estáticos en el tiempo. En cambio, a través de retroalimentación, refinamiento y redistribución, los modelos se mejoran continuamente y se adaptan a las condiciones en evolución. De esta manera, tanto el modelo como el trabajo detrás de él pueden proporcionar un valor continuo a la organización mientras se necesite la solución [160].

3.2. Cross Industry Standard Process for Data Mining (CRISP-DM)

Cross Industry Standard Process for Data Mining abreviado como CRISP-DM, es una metodología para el proceso de la minería de datos que describe los enfoques comunes para orientar los proyectos, trabajos o investigaciones que se lleva a cabo una organización. Como metodología, incluye descripciones de las fases normales de un proyecto, las tareas necesarias en cada paso y una explicación de las relaciones entre las tareas. También se suele hablar de CRISP-DM como un modelo de proceso, pues ofrece un resumen del ciclo de vida de la minería de datos.

El ciclo de vida de CRISP-DM consta de seis fases que indican las dependencias más importantes y frecuentes entre ellas [161] y son las siguientes:

1. Comprensión del negocio: Esta fase inicial se centra en la determinación y la comprensión de los objetivos y requisitos del proyecto desde una perspectiva empresarial, y luego convertir este conocimiento en una definición del problema de minería de datos, y un plan preliminar diseñado para alcanzar los objetivos.

2. Comprensión de datos: La fase de comprensión de datos de CRISP-DM implica estudiar más de cerca los datos disponibles para la minería. Este paso es esencial para evitar problemas inesperados durante la siguiente fase (preparación de datos)

que suele ser la fase más larga de un proyecto. Esta fase comienza con una colección inicial de datos y con el objetivo de familiarizarse con los datos, los tipos de ellos, sus esquemas de codificación, describiendo métricas de calidad y precisión de los mismos, para descubrir las primeras señales dentro de los datos y detectar temas interesantes para poder formular hipótesis de información oculta.

3. Preparación de datos: La preparación de datos es uno de los aspectos más importantes y con frecuencia que más tiempo exigen en la minería de datos. De hecho, se estima que la preparación de datos suele llevar el 50 % al 70 % del tiempo y esfuerzo de un proyecto. Consiste en cubrir todas las actividades para construir el conjunto de datos para la minería [161]. Estas tareas son ejecutadas en múltiples oportunidades y sin un orden especifico. Las tareas incluyen selección y transformación de tablas, registros y atributos y limpieza de datos para las herramientas de modelado, derivación de nuevos atributos.

4. Modelado: En esta fase se seleccionan y aplican varias técnicas de modelado y se calibran los parámetros para obtener óptimos resultados. Hay varias técnicas que tienen requerimientos específicos para la forma de los datos, por lo que normalmente, los analistas de datos ejecutan varios modelos utilizando los parámetros por defecto y ajustan los parámetros o vuelven a la fase de preparación de datos para las manipulaciones necesarias por su modelo. Es extraño que las cuestiones relativas a la minería datos de una empresa se solucionen satisfactoriamente con un modelo y ejecución únicos [161].

5. Evaluación: En esta etapa en el proyecto ha construido un modelo (o modelos) se evalúa si los resultados obtenidos son acordes con el objetivo de la minería. En este punto, habrá completado la mayor parte de su proyecto de minería de datos.

6. Despliegue: Es el proceso que consiste en utilizar sus nuevos conocimientos para implementar las mejoras en su organización. Esta fase depende de los requerimientos, pudiendo ser simple como la generación de un reporte o compleja como la implementación de un proceso de explotación de información que atraviese a toda la organización.

Aunque no es un paso formal dentro de CRISP-DM, es común en la mayoría de los casos se haga una revisión final del proyecto lo que le ofrece una oportunidad de formular sus impresiones finales e incorporar los conocimientos adquiridos durante el proceso de minería de datos. También es importante recordar que a pesar que estas etapas están claramente definidas y ordenadas, es muy común que durante el proceso se retroceda a una etapa anterior con el fin de mejorar alguna característica deseada durante la investigación.

3.3. Scrum

Scrum es una metodología ágil y flexible para gestionar el desarrollo de software, cuyo principal objetivo es maximizar el retorno de la inversión para su empresa (ROI). Se basa en construir primero la funcionalidad de mayor valor para el cliente y en los principios de inspección continua, adaptación, auto-gestión e innovación. [162] En Scrum se

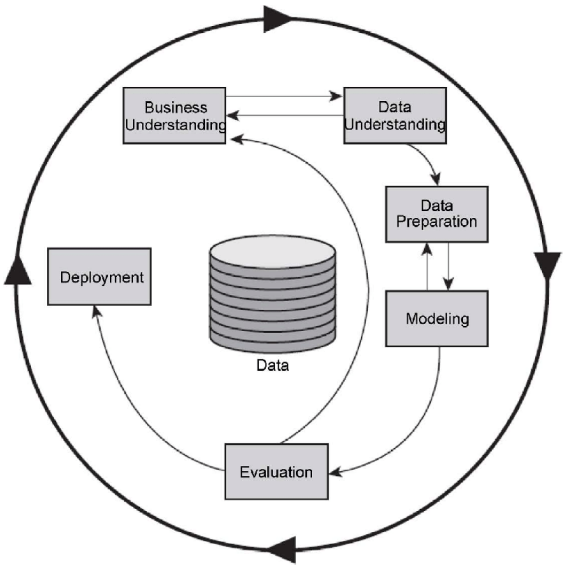

Figura 3.2: Flujo de Actividades en la Metodología CRISP-DM

aplican de manera regular un conjunto de buenas prácticas para trabajar colaborativamente, en equipo, y obtener el mejor resultado posible de un proyecto. Estas prácticas se apoyan unas a otras y su selección tiene origen en un estudio de la manera de trabajar de equipos altamente productivos [163].

Es parte de la filosofía de Scrum el poder realizar entregas parciales y regulares del producto final, priorizadas por el beneficio que aportan al receptor del proyecto. Por ello, Scrum está especialmente indicado para proyectos en entornos complejos, donde se necesita obtener resultados pronto, donde los requisitos son cambiantes o poco definidos, donde la innovación, la competitividad, la flexibilidad y la productividad son fundamentales.

En Scrum un proyecto se ejecuta en bloques temporales cortos y fijos (iteraciones que normalmente son de 2 semanas, aunque en algunos equipos son de 3 y hasta 4 semanas, límite máximo de feedback y reflexión). Cada iteración tiene que proporcionar un resultado completo, un incremento de producto final que sea susceptible de ser entregado con el mínimo esfuerzo al cliente cuando lo solicite. El proceso parte de la lista de objetivos/requisitos priorizada del producto, que actúa como plan del proyecto. En esta lista el cliente prioriza los objetivos balanceando el valor que le aportan respecto a su coste y quedan repartidos en iteraciones y entregas.

Las actividades que se llevan a cabo bajo la metodología de Scrum son las siguientes:

1. Planificación de la iteración: En el primer día de cada iteración, el equipo realza una reunión de planificación de la iteración. Esta etapa consiste en una primera fase de selección de requisitos, en la cual el cliente presenta al equipo la lista de requisitos priorizada del proyecto y se responden a las dudas que surgen sobre el dominio de las tareas a realizar y una segunda fase de planificación de la iteración, donde el equipo elabora la listas de tareas necesarias para desarrollar lo que se es requerido para la iteración, además de la estimación de esfuerzos por parte de los

miembros del equipo.

2. Ejecución de la iteración: Cada día se realiza una reunión con todo el equipo de 15 minutos como máximo para inspeccionar el avance en la realización de las tareas, dependencias u actividades bloqueantes ye insumos pendientes para su culminación. Durante la iteración el Scrum Master (facilitador) se encarga de que el equipo pueda articularse y cumplir los compromisos adquiridos por el equipo.

3. Inspección y adaptación: El último día de la iteración se realiza la reunión de revisión de la iteración. Esta se conforma de dos partes, la demostración en donde el equipo presenta al cliente el/los entregable(s) con los requisitos mínimos acordados y la retrospectiva en la que el equipo analiza si su manera de trabajar ha sido la adecuada, los problemas que han surgido, su solución, de forma que se pueda mejorar de manera continua la productividad.

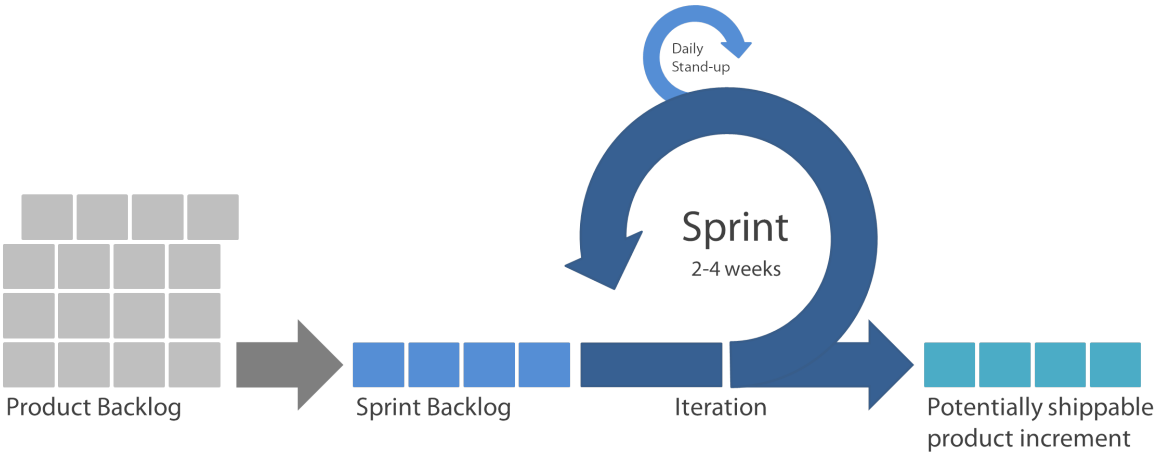

Figura 3.3: Flujo de las Actividades en la Metodología Scrum

Capítulo 4

Problema de Investigación

4.1. Planteamiento del Problema

Cada día es más común observar como dispositivos y artefactos que consideramos normales, adquieren capacidades de conexión, de captura de datos y ser capaces de ejercer acciones sobre las tareas que deben realizar. Estos dispositivos son parte de una evolución del área tecnológica al cual lleva por nombre Internet de las Cosas y su importancia radica en como cambiarán la manera en que podremos interactuar con los ambientes en los que hacemos vida, como nuestros trabajos, hogares y ciudades.

El Internet de las Cosas abre la puerta a un grupo importante de innovaciones a todo nivel y un lugar que se beneficiará especialmente de estos avances serán los hogares. La idea de tener artefactos, dispositivos y electrodomésticos más autónomos no es nueva, pero la domótica ya esta teniendo una evolución en cuanto a la manera en que estos no solo adquieren capacidad de comunicarse entre ellos y con otros sistemas sino como interactuamos nosotros con nuestros hogares, con el fin de automatizar aquellas labores tediosas y difíciles, además de proveer de mayores niveles de seguridad y accesibilidad a los recursos que tenemos, a nuestros bienes y a nuestros seres queridos.

Es así como los procesos del hogar se van automatizando gracias a estos artefactos inteligentes y que son capaces de medir variables de interés, como el consumo de los recursos, alertas de sensores de vigilancia o aquellos datos generados por experiencias de usuario personalizadas. Es así como nace la necesidad de establecer arquitecturas, sistemas y plataformas que puedan explotar no solo el potencial que ofrecen los dispositivos para poder realizar múltiples tareas con la menor intervención humana posible.

Entre las cualidades importantes que deben poseer estas plataformas ante la creciente cantidad de dispositivos y artefactos conectados, están las de poder mostrar una instantánea del estado actual de los sensores, actuadores, dispositivos y gateways IoT, así como también aprovechar el poder de cómputo y almacenamiento cada vez mayor de los mismos para poder guardar, preprocesar y filtrar la gran cantidad de datos (muchos de ellos semi-estructurados o no estructurados) que se producen de manera continua, sin la necesidad de recurrir totalmente a sistemas en la nube para encontrar información que pueda ser utilizada para la toma de decisiones tanto automatizada como no auto-

matizada. Este esquema de llevar dichas capacidades hacia la base de la jerarquía de los sistemas distribuidos es llamada cómputo en la niebla (cuando el preprocesamiento es llevado a cabo en uno o más nodos intermedios de la red local) o cómputo en el borde (cuando el preprocesamiento es llevado a cabo en los dispositivos).

Sin embargo, en la actualidad existen pocos sistemas y plataformas (tanto libres como propietarias) que permiten conocer el estado de los dispositivos, además de proveer de formas para controlarlos usando recetas de comportamiento preestablecidas y/o programables. Dadas las características y complejidad que implica este problema, surgen las preguntas de ¿cuál es la mejor arquitectura para un sistema domótico que aproveche los datos que se generan a partir de los procesos automatizados?, ¿qué interfaz debe existir entre los sensores, actuadores y dispositivos en general para que puedan interactuar con servicios y plataformas de cómputo para la obtención de conocimiento?, ¿qué tipo de conocimiento es posible obtener de los datos recabados de los dispositivos y sus preprocesamiento? y finalmente ¿es posible que al tener ese conocimiento se pueda realizar una toma de decisiones automatizadas que tengan en cuenta inteligentemente las condiciones actuales del ambiente, así como también los gustos de las personas, de forma de ajustar el funcionamiento de los dispositivos?

4.1.1. Justificación

Los datos generados por el uso y consumo de recursos, por las estadísticas y operaciones de sistemas domóticos en la actualidad tienen un carácter comercial poco explorado. Muchos de los recursos utilizados en los hogares no cuentan con inspección o solo se obtienen métricas de forma global a través de la facturación de los proveedores de los servicios. Esta tendencia está cambiando poco a poco pero abre la oportunidad de ofrecer tanto en hogares construidos o como en los diseños de los nuevos hogares inteligentes una forma de aportar características de eficiencia energética, de consumo y reposición sustentables de recursos vitales, de mayores niveles de seguridad, accesibilidad y confort para los habitantes.

4.1.2. Alcance

La respuesta a las diversas preguntas planteadas en la investigación pasa por la creación de una interfaz (mejor conocido como gateway IoT) diseñada para interactuar, controlar y asegurar múltiples dispositivos bajo una misma plataforma, además de generar y filtrar datos, información o conocimiento accionable para la toma de decisiones de forma automatizada o no automatizada, en ese o en posteriores eslabones de la cadena de procesamiento de un ambiente distribuido. Sea plantea que esta plataforma sea tanto hardware como software libre y capaz de integrarse con soluciones existentes de almacenamiento y procesamiento en la nube, con la cual se puedan obtener modelos que revelen patrones útiles para ser desplegados en el gateway IoT y mejorar los procesos automatizados.

4.1.3. Objetivos

4.1.3.1. Objetivos Generales

El objetivo de esta investigación es el poder determinar la mejor estrategia para afrontar el desarrollo de un sistema de domótico inteligente y adaptado al Internet de las Cosas, con la capacidad de analizar los datos de los dispositivos desplegados en un hogar para optimizar los procesos automatizados, haciendo uso de técnicas de la computación distribuida como el cómputo en el borde y Big Data.

4.1.3.2. Objetivos Específicos

Para alcanzar el objetivo general estipulado se plantean los siguientes objetivos específicos:

- Utilizar la metodología Scrum durante el diseño y creación de las herramientas durante todo el proceso.
- Establecer la arquitectura adecuada para dar soporte a un sistema domótico y el software de análisis de los datos de dispositivos IoT.
- Construir prototipos funcionales de uno o más dispositivos IoT para la captura de datos de las variables sobre los procesos involucrados.
- Construir una interfaz para la gestión de dispositivos IoT que permita preprocesar datos e información relevante, tanto para toma de decisiones automatizadas como no automatizadas y para posterior envío, almacenamiento, procesamiento y análisis en la nube.
- Realizar la captura de datos y la construcción de modelos bajo la metodología KDD y la metodología fundamental para la ciencia de datos para el análisis de los mismos y generar conocimiento útil para los usuarios.
- Establecer comportamientos dinámicos en base al conocimiento obtenido en los prototipos funcionales.
- Crear herramientas de visualización de datos adecuadas para informar los resultados obtenidos y realizar sugerencias.
- Crear herramientas de control y/o programación de comportamiento sobre dispositivos IoT.
- Proveer de métodos de seguridad en el acceso y uso de los datos y dispositivos involucrados.
- Utilización de herramientas, protocolos y plataformas agnósticas, construidas bajo sotfware y hardware libres.

Bibliografía

[1] Jeffrey Stanton. *An introduction to Data Science*. Syracuse University, 2012.

[2] Real Academia Española. Dato. `http://dle.rae.es/?id=Bskzsq5|BsnXzV1`, Junio 2017. [Accedido: 08-06-2017].

[3] Jhonstone High School. Characteristics of information. `http://www.jhigh.co.uk/Intermediate2/Using%20Information/12_charact_of_info.html`, Junio 2017. [Accedido: 08-06-2017].

[4] Real Academia Española. Conocimiento. `http://dle.rae.es/srv/search?m=30&w=conocimiento`, Junio 2017. [Accedido: 08-06-2017].

[5] Drew Conway. Vitae. `http://drewconway.com/vitae/`, 2015. [Accedido: 09-06-2017].

[6] Drew Conway. The data science venn diagram. `http://drewconway.com/zia/2013/3/26/the-data-science-venn-diagram`, Septiembre 2010. [Accedido: 09-06-2017].

[7] Rachel Schutt & Cathy O'Neil. *Doing Data Science*. O'Really Media Inc., Octubre 2013.

[8] Michel Minelli & Michele Chambers & Ambiga Dhiraj. *Big Data, Big Analytics: Emerging Business Inteligence and Analytics Trends for Today's Businesses*. John Wiley & Sons, Inc, 2013.

[9] Ricardo Barranco Fragoso. ¿Qué es Big Data? `http://www.ibm.com/developerworks/ssa/local/im/que-es-big-data/index.html`, Junio 2012. [Accedido: 09-06-2017].

[10] Doug Laney. 3D Data Management: Controlling Data Volume, Velocity and Variety. In *Application Delivery Strategies*, Febrero 2001.

[11] Ashley DeVan. The 7 V's of Big Data. `http://www.impactradius.com/blog/7-vs-big-data`, Abril 2016. [Accedido: 09-06-2017].

[12] Bernard Marr. Why only one of the 5 Vs of big data really matters. `http://www.ibmbigdatahub.com/blog/why-only-one-5-vs-big-data-really-matters`, Marzo 2015. [Accedido: 09-06-2017].

[13] Mark van Rijmenam. Four Ways Big Data Will Revolutionize Education. https://datafloq.com/read/big-data-will-revolutionize-learning/206, Abril 2015. [Accedido: 10-06-2017].

[14] SAS Institute Inc. Big Data: What is and why matters. https://www.sas.com/en_us/Insights/big_data/what_is_big_data.html#dmusers. [Accedido: 10-06-2017].

[15] Álex Rayón. Big data en la inudstria 4.0: Un marco de oportunidades. https://blog.deusto.es/big-data/en-la-industria-4-0-un-marco-de-oportunidades/, Ocutbre 2016. [Accedido: 10-06-2017].

[16] Cristina Juan. Cómo el Big Data ha revolucionado la Logística. https://comunidad.iebschool.com/iebs/software-de-gestion/big-data-en-logisitica/, Dciciembre 2013. [Accedido: 10-06-2017].

[17] Lois-James Davis. How big data analytics in sports supports the experts. http://www.itportal.com/features/how-big-data-analytics-in-sport-supports-the-experts/, March 2017. [Accedido: 10-06-2017].

[18] EMC Education Service. *Data Science & Big Data Analytics: Discovering, Analyzing, Visualizing and Presenting Data*. John Wiley & Sons, Inc., 2015.

[19] Abraham Solberschatz. *Fundamentos de Bases de Datos*. McGraw Hill, 2002.

[20] IBM Knowledge Center. Bases de datos relacionales. https://www.ibm.com/support/knowledgecenter/es/SSEPGG_8.2.0/com.ibm.db2.udb.doc/admin/c0004099.htm. [Accedido: 10-06-2017].

[21] Amazon Web Services. Qué es una base de datos relacional? https://aws.amazon.com/es/realational-database/. [Accedido: 10-06-2017].

[22] Mercy Ospina. Bases de datos nosql: Conceptos básicos. https://es.slideshare.net/mercyo/bd-no-sql-conceptos-basicos, Junio 2016. [Accedido: 10-06-2017].

[23] Oracle Help Center. What is Data Mining? https://docs.oracle.com/cd/B28359_01/datamine.111/b28129/process.htm. [Accedido: 11-06-2017].

[24] Universidad de Buenos Aires. ¿qué es data mining? http://triton.exp.dc.uba.ar/datamining/index.php/que-es-data-mining. [Accedido: 11-06-2017].

[25] SAS Institute Inc. Data Mining: What it is and why matters. https://www.sas.com/en_us/insights/analytics/data-mining.html. [Accedido: 11-06-2017].

[26] WebMining Consultores. KDD: Proceso de Extracción de conocimiento. http://www.webmining.cl/2011/01/proceso-de-extraccion-de-conocimiento/, Enero 2011. [Accedido: 11-06-2017].

[27] Owen Duncan & Craig Guyer. Data Mining Algorithms. https://docs.microsoft.com/en-us/sql/analysis-services/data-minig/data-mining-algorithms-analysis-services-data-mining, MArzo 2016. [Accedido: 11-06-2017].

[28] J. Han and M. Kamber. *Data Mining: Concepts and Techniques*. The Morgan Kaufmann series in data management systems. Elsevier, 2006.

[29] IBM Knowledge Center. About Text Minig. https://www.ibm.com/support/knowledgecenter/en/SS3RA7_17.1.0/ta_guide_ddita/textmining/shared_entities/tm_intro_tm_defined.html. [Accedido: 11-06-2017].

[30] Juan D. Velázques & Lorena Donoso. Aplicación de Técnicas de Web Mining Sobre los Datos Originados por Usuarios de Páginas Web. Visión Crítica Desde La Libertad, La Privacidad y el Honor de las Personas. Technical report, Universidad de Chile, Junio 2010.

[31] Guido Van Rossum. What is Python, Executive Summary. https://www.python.org/doc/essays/blurb/, 1997. [Accedido: 11-06-2017].

[32] Guido Van Rossum. Foreword for 'Programing Python' (1st ed.). https://www.python.org/doc/essays/foreword/, Mayo 1996. [Accedido: 11-06-2017].

[33] A. M. Kuchling. Python Advanced Librery. https://www.python.org/dev/pep/pep-0206/. [Accedido: 11-06-2017].

[34] Meet Django. https://www.djangoproject.com. [Accedido 11-06-2017].

[35] Flask web development, one drop at a time. https://flask.pocoo.org. [Accedido 11-06-2017].

[36] Pyramid the Start Small, Finish Big Stay Finished Framework. https://trypyramid. [Accedido 11-06-2017].

[37] pandas: Python Data Analysis Library. http://pandas.pydata.org/index.html. [Accedido 11-06-2017].

[38] scikit-learn Machine Learning in Python. http://. [Accedido 11-06-2017].

[39] NumPy. http://www.numpy.org. [Accedido 11-06-2017].

[40] What is the Jupyter Notebook. http://jupiter-notebook-beginner-guide.readthedocs.io/en/latest/what_is_jupyter.html. [Accedido 11-06-2017].

[41] SciPy. https://www.scipy.org. [Accedido 11-06-2017].

[42] Ben Croston. RPi.GPIO. http://sourceforge.net/projects/raspberry-gpio-python, Octubre 2016. [Accedido 11-06-2017].

[43] Ben Nuttall & DAve Jones. gpiozero. https://gpiozero.readthedocs.io/en/stable, 2015. [Accedido 11-06-2017].

[44] The R Fundation. What is R? https://www.r-project.org/about.html. [Accedido 11-06-2017].

[45] Geethika Bhavya Peddibhotla. Top 20 R Machine Learning and Data Science packages . http://www.kdnuggets.com/2015/06/top-20-r-machine-learning-packages.html, Junio 2016. [Accedido 11-06-2017].

[46] Oracle. ¿Qué es la tecnología Java y para qué la necesito? https://www.java.com/es/download/faq/whatisjava.xml. [Accedido 11-06-2017].

[47] https://spring.io/. [Accedido 11-06-2017].

[48] Grails: A powerful Groovy-based web application framework for the JVM built on top of Spring Boot. https://grails.org/. [Accedido 11-06-2017].

[49] Oracle. JavaServer Faces Technology . http://www.oracle.com/technetwork/java/javaee/javaserver 139869.html. [Accedido 11-06-2017].

[50] OpenJDK. Device I/O Project. http://openjdk.java.net/projects/dio/. [Accedido 11-06-2017].

[51] Robert Savage & Daniel Sendula. Welcome to Pi4J. http://pi4j.com/index.html, Julio 2016. [Accedido 11-06-2017].

[52] Martin Odersky & CO. Scala language specification. https://www.scala-lang.org/files/archive/spec/2.12/. [Accedido 11-06-2017].

[53] Scalaz: An extension to the core Scala library for functional programming. https://github.com/scalaz/scalaz. [Accedido 11-06-2017].

[54] Bill Venners. ScalaTest: Simply productive. http://www.scalatest.org/. [Accedido 11-06-2017].

[55] Gordon Henderson. Wiring Pi: GPIO Interface library for the Raspberry Pi. http://wiringpi.com/. [Accedido 11-06-2017].

[56] Language reference. https://www.arduino.cc/en/Reference/HomePage. [Accedido 11-06-2017].

[57] Chukl Lam. *Hadoop in Action*. Manning Publications, 2010.

[58] Tom White. *Hadoop The Definitive Guide*. O'Reilly Media Inc., 2015.

[59] Apache Software Foundation. Project Description. https://wiki.apache.org/hadoop/ProjectDescription, Enero 2014. [Accedido: 12-06-2017].

[60] Hortonworks Inc. Apache Hadoop YARN. https://es.hortonworks.com/apache/yarn/. [Accedido: 12-06-2017].

[61] Dhruba Borthakur. HDFS Architecture Guide. https://hadoop.apache.org/docs/r1.2.1/hdfs_design.html#Introduction, Abril 2013. [Accedido: 12-06-2017].

[62] Kevin Sitto & Marshall Presser. *Field Guide to Hadoop*. O'Reilly Media, Inc, Marzo 2015.

[63] Apache Software Foundation. Apache Hadoop YARN. https://hadoop.apache.org/docs/r2.7.2/hadoop-yarn/hadoop-yarn-site/YARN.html, Enero 2016. [Accedido: 12-06-2017].

[64] Hadoop architecture. https://hadooptutorial.wikispace.com/Hadoop+architecture. [Accedido: 12-06-2017].

[65] Power Data. El asombrosamente rico ecosistema de Hadoop. http://blog.powerdata.es/el-valor-de-la-gestion-de-datos/bid/390089/El-asombrosamente-rico-ecosistema-de-Hadoop, Julio 2014. [Accedido: 13-06-2017].

[66] The Apache Software Foundation. Apache Accumulo. https://accumulo.apache.org/. [Accedido: 13-06-2017].

[67] Hortonworks Inc. Apache Accumulo. https://es.hortonworks.com/apache/accumulo/. [Accedido: 13-06-2017].

[68] Hortonworks Inc. Apache HBase. https://es.hortonworks.com/apache/hbase/. [Accedido: 13-06-2017].

[69] The Apache Software Fundation. Welcome to Apache HBase. https://hbase.apache.org. [Accedido: 13-06-2017].

[70] The Apache Software Fundation. Apache Hive. https://hive.apache.org/. [Accedido: 13-06-2017].

[71] The Apache Software Fundation. Apache Hive. https://cwiki.apache.org/confluence/display/Hive/Home. [Accedido: 13-06-2017].

[72] The Apache Software Fundation. Welcome to Apache Pig! https://pig.apache.org/. [Accedido: 13-06-2017].

[73] Ricardo Barranco Fragoso. Análisis de Big Data con Apache Pig, Abril 2017.

[74] Grant Ingersoll. Introducing Apache Mahout. https://www.ibm.com/developerworks/java/library/j-mahout/, Septiembre 2008. [Accedido: 13-06-2017].

[75] The Apache Software Fundation. What is Apache Mahout? https://mahout.apache.org/. [Accedido: 13-06-2017].

[76] Grant Ingersoll. Apache Mahout: Aprendizaje escalable con máquina para todos. https://www.ibm.com/developerworks/ssa/library/j-mahout-scaling/index.html, Febrero 2012. [Accedido: 13-06-2017].

[77] The Apache Software Fundation. Welcome to Apache Flume. https://flume.apache.org/.

[78] IBM Analytics. Flume. https://www.ibm.com/analytics/us/en/technology/hadoop/flume/. [Accedido: 13-06-2017].

[79] Hortonworks Inc. Apache Flume. https://es.hortonworks.com/apache/flume/. [Accedido: 13-06-2017].

[80] The Apache Software Fundation. Spark Overview. https://spark.apache.org/docs/latest/. [Accedido: 14-06-2017].

[81] Hortonworks Inc. Apache Spark. https://es.hortonworks.com/apache/spark/. [Accedido: 14-06-2017].

[82] Gwen Shapira & Jeff Holoman. Apache Kafka for Beginners. http://blog.cloudera.com/blog/2014/09/apache-kafka-for-beginners/, Septiembre 2014. [Accedido: 14-06-2017].

[83] The Apache Software Fundation. Apache Kafka Introduction. https://kafka.apache.org/intro. [Accedido: 14-06-2017].

[84] The Apache Software Fundation. Apache Storm. https://stprm.apache.org/index.html. [Accedido: 14-06-2017].

[85] Adam Muise. INTRODUCTION TO APACHE FALCON: DATA GOVERNANCE FOR HADOOP. https://es.hortonworks.com/blog/introduction-apache-falcon-hadoop/, Marzo 2015. [Accedido: 15-06-2017].

[86] The Apache Software Fundation. Apache Sqoop. https://sqoop.apache.org/, Marzo 2017. [Accedido: 15-06-2017].

[87] Hortonworks Inc. Apache Sqoop. https://es.hortonworks.com/apache/storm/. [Accedido: 15-06-2017].

[88] Arvind Prabhakar. Apache Sqoop - Overview. http://blog.cloudera.com/blog/2011/10/apache-sqoop-overview/, Octrubre 2011. [Accedido: 15-06-2017].

[89] IBM Knowledge Center. Descripción general de la pasarela de Apache Knox. https://www.ibm.com/support/knowledgecenter/es/SSPT3X_4.1.0/com.ibm.swg.im.infosphere.biginsights.admin.doc/doc/knox_overview.html. [Accedido: 15-06-2017].

[90] The Apache Software Fundation. REST API and Application Gateway for the Apache Hadoop Ecosystem. https://knox.apache.org/, Junio 2017. [Accedido: 15-06-2017].

[91] The Apache Software Fundation. Aopache Ranger. https://ranger.apache.org/, Junio 2017. [Accedido: 15-06-2017].

[92] Hortonworks Inc. Apache Ranger. https://es.hortonworks.com/apache/ranger/. [Accedido: 15-06-2017].

[93] The Apache Software Fundation. Introduction. https://ambari.apache.org/, Junio 2017. [Accedido: 15-06-2017].

[94] Hortonworks Inc. Apache Ambari. https://es.hortonworks.com/apache/ambari/. [Accedido: 15-06-2017].

[95] Hortonworks Inc. Apache Oozie. https://es.hortonworks.com/apache/oozie/. [Accedido: 15-06-2017].

[96] The Apache Software Fundation. Welcome to Apache ZooKeeper. https://zookeeper.apache.org/. [Accedido: 16-06-2017].

[97] Benjamin Reed. Project Description. https:https://cwiki.apache.org/confluence/display/ZOOKEEPER/ProjectDescription. [Accedido: 16-06-2017].

[98] The Apache Software Fundation. Products that include Apache Hadoop or derivative works and Commercial Support. https://wiki.apache.org/hadoop/Distributions%20and%20Commercial%20Support, Diciembre 2014. [Accedido: 16-06-2017].

[99] Kare Rose & Scott Eldridge & Lyman Chapin. The Ïnternet of Things: An Overview - Understanding the issues and Challenges of a More Connected World. Technical report, The Internet Society, Octubre 2015.

[100] Sector de Normalización de las Telecomunicaciones de la UIT. Descripción general de Internet de los objetos. Technical report, Unión Internacional de Telecomunicaciones, Junio 2012.

[101] H. Tschofenig & others. Architectural Considerations in Smart Object Networking. Technical report, Internet Architecture Board, Marzo 2015.

[102] James Manyika & otros. The internet of Things: Mapping the Value Beyond the Hype. http://www.mckinsey.com/insights/business_technology/the_internet_of_things_the_value_of_digitizing_the_physical_world, Junio 2015. [Accedido: 17-06-2017].

[103] Rahul. IoT applications spanning across industries. https://www.ibm.com/blogs/internet-of-things/iot-applications-industries/, Abril 2017. [Accedido: 17-06-2017].

[104] Tamara Kulesa. 8 sensors to help you create a smart home. https://www.ibm.com/blogs/internet-of-things/sensors-smart-home/, Deciembre 2016. [Accedido: 17-06-2017].

[105] Bill McCabe. Creating smart cities with IoT. https://www.ibm.com/blogs/internet-of-things/creating-smart-cities-iot/, Junio 2016. [Accedido: 17-06-2017].

[106] Kevin Patel. 6 Benefits of IoT for Hospitals and Healthcare. https://www.ibm.com/blogs/internet-of-things/6-benefits-of-iot-for-healthcare/, Enero 2017. [Accedido: 17-06-2017].

[107] Yash Mehta. Agricultural Internet of Things technology applications. https://www.ibm.com/blogs/internet-of-things/iot-agricultural-applications/, Mayo 2017. [Accedido: 17-06-2017].

[108] Matt Bellias. How IoT is innovating transport to make cities more liveable. https://www.ibm.com/blogs/internet-of-things/transport/, Noviembre 2016. [Accedido: 17-06-2017].

[109] Lynne Slowey. From automotive to mobility, the future of IoT for the car industry. https://www.ibm.com/blogs/internet-of-things/automotive-future-iot/, Febrero 2017. [Accedido: 17-06-2017].

[110] Jen Clark. What is industry 4.0? https://www.ibm.com/blogs/internet-of-things/industry-4-0/, Octubre 2016. [Accedido: 18-06-2017].

[111] Ari Keränen & Carsten Bormann. Internet of Things: Standards and Guidance from the IETF. http://www.internetsociety.org/publications/ietf-journal-april-2016/internet-things-standards-and-guidance-ietf, Abril 2016. [Accedido: 18-06-2017].

[112] Stan Schneider. Understanding The Protocols Behind The Internet Of Things. http://www.electronicdesign.com/iot/understanding-protocols-behind-internet-things, Octubre 2013. [Accedido: 18-06-2017].

[113] Sergio Collado & otros. HTTP. https://developer.mozilla.org/es/docs/Web/HTTP, 2017 Mayo. [Accedido: 18-06-2017].

[114] Z. Shelby & otros. The Constrained Application Protocol (CoAP). Technical report, Internet Engineering Task Force (IETF), Junio 2014.

[115] James Stansberry. The IoT communication protocols. https://www.linkedin.com/pulse/iot-communication-protocols-james-stansberry, Octubre 2015. [Accedido: 18-06-2017].

[116] Roger A Light. Mosquitto: server and client implementation of the MQTT protocol. *The Journal of Open Source Software*, 2(13), Mayo 2017.

[117] Joachim Lindborg. About. http://www.xmpp-iot.org/about/, Noviembre 2014. [Accedido: 19-06-2017].

[118] Afshar Ganjali. XMPP: Swiss Army Knife for Internet of Things (IoT). https://blog.securitycompass.com/xmpp-swiss-army-knife-for-internet-of-things-iot-9eff783c44ba, Mayo 2016. [Accedido: 19-06-2017].

[119] Seth Manhein & John Taubensee. AMQP 1.0 support in Service Bus. https://docs.microsoft.com/en-us/azure/service-bus-messaging/service-bus-amqp-overview, Abril 2017. [Accedido: 19-06-2017].

[120] Universidad Técnica Particular de Loja. Servicio de Distribución de Datos (Data Distribution Service), DDS . http://www.utpl.edu.ec/proyectomiddleware/?q=tutorial-dds. [Accedido: 19-06-2017].

[121] Information Sciences Institute of University of Southern California. INTERNET PROTOCOL. Technical report, Defense Advanced Research Projects Agency, Septiembre 1981.

[122] Stephen Deering & Robert Hinden. Internet Protocol, Version 6 (IPv6). Technical report, The Internet Society, Diciembre 1998.

[123] The Institute of Electrical and Inc. Electronics Engineers. Ieee 802.15 wpan task group 1 (tg1). http://www.ieee802.org/15/pub/TG1.html. [Accedido: 22-06-2017].

[124] The Institute of Electrical and Inc. Electronics Engineers. Ieee 802.15 wpan™ task group 4 (tg4). http://www.ieee802.org/15/pub/TG4.html. [Accedido: 22-06-2017].

[125] RS Components. 11 internet of things (iot) protocols you need to know about. https://www.rs-online.com/designspark/eleven-internet-of-things-iot-protocols-you-need-to-know-about, Abril 2015. [Accedido: 22-06-2017].

[126] El androide libre. Todo sobre zigbee, la tecnología ultrabarata para comunicación inalámbrica. https://elandroidelibre.elespanol.com/2015/08/todo-sobre-zigbee-la-tecnologia-ultrabarata-para-comunicacion-inalambrica.html, Agosto 2015. [Accedido: 22-06-2017].

[127] Pentadom. LA DOMÓTICA INALÁMBRICA CON Z-WAVE, UNA REALIDAD PARA VIVIENDAS EXISTENTES. http://pentadom.com/2014/03/la-domotica-inalambrica-con-z-wave-una-realidad-para-viviendas-ya-construidas/, Marzo 2014. [Accedido: 22-06-2017].

[128] Gabriel Montenegro & Otros. Transmission of IPv6 Packets over IEEE 802.15.4 Networks. Technical report, Network Working Group, Septiembre 2007.

[129] Jonathan Hui & Pascal Thubert. Compression Format for IPv6 Datagrams over IEEE 802.15.4-Based Networks. Technical report, Internet Engineering Task Force (IETF), Septiembre 2011.

[130] Zach Shelby & Otros. Neighbor Discovery Optimization for IPv6 over Low-Power Wireless Personal Area Networks (6LoWPANs). Technical report, Internet Engineering Task Force (IETF), Noviembre 2012.

[131] The Institute of Electrical and Inc. Electronics Engineers. IEEE 802.11 WIRELESS LOCAL AREA NETWORKS. http://www.ieee802.org/11/. [Accedido: 22-06-2017].

[132] International Telecommunication Union. About mobile technology and IMT-2000. https://www.itu.int/osg/spu/imt-2000/technology.html. [Accedido: 22-06-2017].

[133] UMTSWorld.com. 3G and UMTS Frequently Asked Questions. http://www.umtsworld.com/umts/faq.htm#f1. [Accedido: 22-06-2017].

[134] 3GPP Initiative. W-CDMA. http://www.3gpp.org/technologies/keywords-acronyms/104-w-cdma. [Accedido: 22-06-2017].

[135] 3GPP Initiative. HSPA. http://www.3gpp.org/technologies/keywords-acronyms/99-hspa. [Accedido: 22-06-2017].

[136] 3GPP Initiative. LTE. http://www.3gpp.org/technologies/keywords-acronyms/98-lte. [Accedido: 22-06-2017].

[137] The Institute of Electrical and Inc. Electronics Engineers. IEEE 802.16e Task Group (Mobile WirelessMAN). http://www.ieee802.org/16/tge/. [Accedido: 22-06-2017].

[138] Organización Internacional de Estandarización. ISO/IEC 18000, Part 3, Information Technology AIDC Techniques - RFID for Item Management - Air Interface, Part 3 - Parameters for Air Interface Communications at 13.56 MHz. Technical report, Organización Internacional de Estandarización, Mayo 2002.

[139] Enoche Andrade. IoT (Internt of Things) Security Considerations. public.tableau.com/profile/enoche.andrade#!/vizhome/iOTSecurity/iOTSecurity. [Accedido: 23-06-2017].

[140] Carlos Robato. Confirmado: el ataque DDoS que tumbó la red en EEUU surgió de una botnet en el Internet de las Cosas. http://es.gizmodo.com/confirmado-el-ataque-ddos-que-tumbo-la-red-en-eeuu-sur-1788100823, Octubre 2016. [Accedido: 24-06-2017].

[141] U.S. Department of Homeland Security. STRATEGIC PRINCIPLES FOR SECURING THE INTERNET OF THINGS (IoT). Technical report, U.S. Department of Homeland Security, Noviembre 2015.

[142] Atos S.A. Smart Homes and Buildings. Technical report, Atos S.A., Diciembre 2011.

[143] José Luis Molina Marticorena. ¿Qué es la domótica? http://www.profesormolina.com.ar/tecnologia/domotica/que_esdomo.htm. [Accedido: 25-06-2017].

[144] Jen Clark. What is home automation. https://www.ibm.com/blogs/internet-of-things/home-automation/, Noviembre 2016. [Accedido: 25-06-2017].

[145] IECOR. ESTÁNDARES INTERNACIONALES DE DOMÓTICA. https://www.iecor.com/estandares-internacionales-de-domotica/. [Accedido: 08-07-2017].

[146] SERCONINT. Sistemas y Protocolos. http://www.serconint.com/sistemas_protocolos.php. [Accedido: 08-07-2017].

[147] José Sirgo & otros. BUSES Y PROTOCOLOS EN DOMÓTICA E INMÓTICA. Technical report, Universidad de Oviedo, Diciembre 2008.

[148] The openHAB Foundation. Welcome to openHAB - a vendor and technology agnostic open source automation software for your home. https://www.openhab.org. [Accedido: 08-07-2017].

[149] The openHAB Foundation. Welcome to the openHAB 2 Documentation! http://docs.openhab.org. [Accedido: 08-07-2017].

[150] The openHAB Foundation. Introduction. https://www.openhab.org/introduction.html. [Accedido: 08-07-2017].

[151] Home Assistant. Home Assistant: Awaken your home. https://home-assistant.io. [Accedido: 08-07-2017].

[152] Home Assistant. Components. https://home-assistant.io/components/. [Accedido: 08-07-2017].

[153] Domoticz. Domoticz Wiki Manual. http://www.domoticz.com/wiki/Domoticz_Wiki_Manual. [Accedido: 08-07-2017].

[154] Domoticz. Domoticz — Control at yor fingers. http://www.domoticz.com. [Accedido: 08-07-2017].

[155] Google Inc. Android Things. https://developer.android.com/things/index.html. [Accedido: 08-07-2017].

[156] Inc. Amazon.com. Amazon Echo - Black. https://www.amazon.com/Amazon-Echo-Bluetooth-Speaker-with-WiFi-Alexa/dp/B00X4WHP5E. [Accedido: 08-07-2017].

[157] Google Inc. Google Home. Voice-activated speaker. https://madeby.google.com/home/. [Accedido: 08-07-2017].

[158] Raspberry Pi Foundation. Raspberry Pi. https://www.raspberrypi.org. [Accedido: 08-07-2017].

[159] The Arduino Company. WHAT IS ARDUINO? https://www.arduino.cc. [Accedido: 08-07-2017].

[160] John B. Rollins. Foundational Methodology for Data Science. Technical report, IBM Corporation, Junio 2015.

[161] IBM Software Group. Manual CRISP-DM de IBM SPSS Modeler. Technical report, IBM Corporation, Diciembre 2011.

[162] SOFTENG. Metodología Scrum. https://www.softeng.es/es-es/empresa/metodologias-de-trabajo/metodologia-scrum.html. [Accedido: 17-07-2017].

[163] proyectosagiles.org. Qué es SCRUM. https://proyectosagiles.org/que-es-scrum/. [Accedido: 17-07-2017].