PROTOTIPO DE LABORATORIO HADOOP PARA ANÁLISIS BIG DATA EN LA INSTITUCIÓN UNIVERSITARIA POLITÉCNICO GRANCOLOMBIANO

PROYECTO DE GRADO PRESENTADO EN CUMPLIMIENTO DE LOS REQUISITOS PARA GRADO DE PREGADO EN INGENIERIA DE SISTEMAS EN LA INSTITUCIÓN UNIVERSITARIA POLITECNICO GRANCOLOMBIANA

LILIANA GALEANO CRUZ, DAVID ALEJANDRO DOMÍNGUEZ RIVERA

DICIEMBRE 2017

PROTOTIPO DE LABORATORIO HADOOP PARA ANÁLISIS BIG DATA EN LA INSTITUCIÓN UNIVERSITARIA POLITÉCNICO GRANCOLOMBIANO

ESTUDIANTES:

LILIANA GALEANO CRUZ

DAVID ALEJANDRO DOMÍNGUEZ RIVERA

DIRECTOR: ROJAS CORDERO ALEXIS

CODIRECTOR: JAIMES FERNANDEZ WILMAR

INSTITUCION UNIVERSITARIA POLITÉCNICO GRANCOLOMBIANO

FACULTAD DE INGENIERÍA Y CIENCIAS BÁSICAS

INGENIERÍA DE SISTEMAS

BOGOTÁ

2017

Certifico que he leído este proyecto de grado y que, en mi sentir, es completamente adecuado en alcance y calidad como proyecto para el grado de ingeniero de sistemas.

_____

Certifico que he leído este proyecto de grado y que, en mi sentir, es completamente adecuado en alcance y calidad como proyecto para el grado de ingeniero de sistemas.

_____

DEDICATORIA

Dedico este proyecto de grado a Dios, a mi familia, a mi pareja y a mis amigas, porque en ellos siempre encuentro apoyo incondicional, comprensión y fuerza para continuar y nunca rendirme.

- *Liliana Galeano Cruz*

Dedico este proyecto de grado a mi familia, mis amigos y mi pajera, porque en ellos siempre encuentro el apoyo que necesito para seguir adelante y demostrar mis capacidades.

También la dedico a mis profesores de universidad, en especial a Alexis Rojas y Wilmar Jaimes por apoyarnos en este proyecto y creer en nosotros.

- *David Alejandro Domínguez Rivera*

AGRADECIMIENTOS

A Dios, por permitirme haber llegado hasta aquí, por darme la fortaleza para continuar adelante a pesar de los duros momentos que me enseñaron a ser una persona más fuerte.

A mi madre y padre por acompañarme durante este camino y por siempre animarme a continuar, por ser mi motor para cada día querer con más fuerzas ser profesional, por hacer de mí una persona llena de valores, principios y principalmente por siempre creer en mí.

A mi pareja y compañero de proyecto de grado, por siempre apoyarme incondicionalmente y animarme para seguir adelante, por compartir mis alegrías, triunfos y fracasos. A quien agradezco profundamente la comprensión, la paciencia y el amor con que logramos caminar juntos estos dos años de preparación profesional, además de los que ya hemos compartido juntos.

A mis hermanos, por la ayuda, compresión y el apoyo brindado.

A mis amigas, por siempre motivarme, escucharme, apoyarme y no dejar que me rindiera, por el abrazo brindado en tiempos difíciles y por el buen consejo dado cuando más lo necesite, mil gracias.

A los profesores Alexis Rojas y Wilmar Jaimes por la orientación y guía para la ejecución de este proyecto.

Liliana Galeano Cruz

A mi pareja y compañera de proyecto de grado, con quien empecé la universidad ya hace 2 años y aunque no ha sido fácil esta etapa de la universidad, siempre me has acompañado, me has tolerado, me has apoyado y me has animado a ser cada vez mejor, porque gracias a ti he tenido el juicio y la constancia necesaria para ser un estudiante sobresaliente, te debo más de lo que imaginas, gracias por estar siempre a mi lado y realizar nuestros sueños juntos, este es tu primer gran paso, el primero de muchos que vendrán y espero los demos juntos.

A mi mamá y mi papá por siempre estar para mí en las buenas y las malas, sin importar que no vivamos juntos siempre estuvieron dispuestos a escucharme, apoyarme y darme ánimo.

A mi familia por apoyare y creer en mí, siempre tuvieron fe en que lo lograría.

A mi mejor amigo Daniel Alejandro Castaño, que muchas veces más que mi amigo parece mi hermano, porque, aunque últimamente no hablamos tanto como me gustaría, siempre me ha brindado su amistad incondicional y me ha apoyado en mis metas y proyectos, ya di yo mi gran paso, ahora es tu turno viejo amigo.

A mi otro mejor amigo Diego Bello Valderrama, que ha estado conmigo en esta etapa de esfuerzo, dedicación y trasnocho.

A mis profesores por formarme y brindarme todos los conocimientos necesarios para llegar a donde estoy, sin ellos no seria la persona que soy.

A Alexis Rojas y Wilmar Jaimes, por brindarnos la oportunidad de demostrar nuestras capacidades en este proyecto.

A Andrea Alejandra Velazco Triana, por siempre confiar en nosotros y nunca dudar que lo lograríamos.

David Alejandro Domínguez Rivera

RESUMEN

Este documento busca que los lectores conozcan Hadoop, un framework diseñado para el almacenamiento y análisis inicial de Big Data, además de las ventajas de implementar esta tecnología, sus principales características y los grandes beneficios que puede brindar Hadoop como framework para lograr una arquitectura escalable. Por otro lado, el desarrollo de este proyecto se enfoca en el montaje de un prototipo de clúster Hadoop, para la institución universitaria Politécnico Grancolombiano, que sirva como herramienta que permita realizar el análisis y almacenamiento de información Big Data, además de servir como guía o referencia para aprender a montar un ambiente Hadoop, además de proveer y explicar un ejemplo básico de programación en MapReduce.

TABLA DE CONTENIDO

LISTA DE TABLAS

LISTA DE FIGURAS

LISTA DE ANEXOS

GLOSARIO

Adaptador puente: adaptador de vitualBox, simula que la tarjeta virtual está conectada al mismo switch que la tarjeta física de la maquina principal.

Backup: es una copia de seguridad, copia y archivo de datos de una computadora.

Data Warehouse: es un repositorio unificado para almacenar todos los datos que recogen diversos sistemas.

Dirección IP: es una etiqueta numérica que identifica de manera lógica y jerárquica a una interfaz dentro de una red.

Eclipse: es una plataforma de programación, desarrollo y compilación de aplicaciones hechas en Java.

Gnome: es un entorno de escritorio e infraestructura para sistemas operativos, es un software libre.

Java: es un lenguaje de programación que se encuentra orientado a objetos y está diseñado para tener pocas dependencias de implementación.

Javascript: es un lenguaje de programación orientado a objetos, basado en prototipos, trabaja en el lado del cliente.

Join: sentencia SQL, encargada de unir o combinar registros de una o varias tablas en una base de datos relacional.

Librería: en informática, es una colección de subprogramas usados para desarrollar software.

Maven: herramienta de software, utilizada para la gestión y su respectiva construcción de proyectos Java.

Metadata: se denomina recurso al grupo de datos que describen el contenido informativo de un objeto.

Nodo: es un punto de unión de varios elementos que se conectan en el mismo lugar.

Red Interna: es una red de computadoras conectados mediante cables, señales, ondas que comparten información.

Virtual Box: es un software de virtualización para arquitecturas, permite realizar la instalación de sistemas operativos con su propio ambiente virtual.

ACRÓNIMOS

SQL: (Structured Query Language) lenguaje de consulta estructurada que da acceso a bases de datos relacionales.

NOSQL: (No Structured Query Language) bases de datos no estructuradas-relacionales.

BI: (Business Intelligence) inteligencia de negocios, uso de datos en una empresa para facilitar la toma de decisiones.

XML: (Extensible Markup Language) lenguaje de marcado extensible.

HTML: (HyperText Markup Language) lenguaje de marcas de hipertexto para elaboración de páginas web.

IBM: (International Business Machines) Negocio internacional de máquinas, reconocida multinacional de tecnología.

MINTIC: (Ministerio de Tecnologías de la Información y Comunicaciones), ministerio de la república de Colombia, busca contribuir al desarrollo económico, social y político de la nación.

JSON: (JavaScript Objeto Notation), formato de texto para el intercambio de datos.

CRUD: (Create, Read, Update, Delete), funciones básicas de bases de datos, crear, leer, modificar y eliminar.

ETL: (Extract, Transform and Load), extraer, transformar y cargar, este proceso permite mover datos desde múltiples fuentes.

INTRODUCCIÓN

Hace unos años la cantidad de información a la que se tenía acceso era muy limitada ya que las fuentes de recolección de información eran pocas y si se trataba de recolección de información en tiempo real entonces esta cantidad era todavía menor. Las empresas debían modelar sus negocios en base a esta limitada y en algunos casos, desactualizada información, este no era el mejor panorama para las mismas, ya que como dijo Sir Francis Bacon a fines del siglo XVI, "El conocimiento es poder". Pero en la actualidad se tiene un panorama totalmente distinto, con la llegada de gran cantidad de dispositivos móviles como son tabletas, teléfonos inteligentes, entre otros, que permiten recoger información en cualquier momento y casi en cualquier lugar [1] el mundo entro a la era en la cual la cantidad de información que dispone es tal que el ser humano debió encontrar nuevas maneras de almacenarla, procesarla y pulirla para sacar los datos que son de mayor interés para las empresas, esto abre un nuevo panorama, en el cual se hace frente no a una carencia de datos, sino a una cantidad tan grande que es fácil perder información la cual en muchos casos puede ser útil o tratar datos que en realidad no aportan mucho a la empresa, por eso el uso de Big Data está creciendo debido al enorme potencial que tiene para analizar datos que antes se consideraban carentes de forma o que no era posible analizarlos, además de poder encontrar nuevas oportunidades y mejorar el entendimiento de la relación producto-consumidor [2].

En la actualidad, el poder de la información de una empresa puede incrementarse por su fiabilidad, volumen, accesibilidad y la capacidad que tiene dicha empresa para darle utilidad en un tiempo razonable, con el objetivo de ayudar en la toma de decisiones inteligentes. Big Data surge del hecho de grandes volúmenes de datos para procesarlos, analizarlos, descubrir patrones y otros aspectos fundamentales para la toma de decisiones. "La empresa que tiene la mejor información, sabe cómo encontrarla y puede utilizarla es la que triunfa más rápido" (Michel Daconta, Leo Obrst y Kevin T. Smith, 2004, The Semantic Web: A Guide to the Future of XML, Web Services, and Knowledge Management).

La gran holeada de información que está llegando actualmente tiene un precio y es la gran dificultad que hay para poder almacenar y analizar dicha información en cantidades de tiempo razonables, además de tener una infraestructura capaz de crecer a la misma velocidad que lo hace la cantidad existente de información; de esta problemática nace Hadoop, un ecosistema de aplicaciones que permiten crear una infraestructura altamente escalable, que a su vez permite no solo almacenar dicha cantidad de información, sino que también es la implementación de un framework (MapReduce) que busca coger toda esa información y analizarla de forma eficiente según lo defina el usuario.

En la actualidad la Institución Universitaria Politécnico Grancolombiano cuenta con una clase de Big Data donde los estudiantes abordan el tema mediante lecturas y conocimientos teóricos brindados por el profesor, al realizar la parte practica el tiempo no es suficiente para implementar todo el temario del curso, por este motivo, se decide realizar un laboratorio Hadoop donde la comunidad universitaria pueda incursionar en esta herramienta y desarrollar sus habilidades en Big

Data implementando el framework MapReduce; sin preocuparse por realizar todo el montaje de la arquitectura Hadoop. El presente proyecto está orientado a la construcción de un prototipo de laboratorio Hadoop para análisis Big Data en la Institución Universitaria Politécnico Grancolombiano, que podría ser implementado en un futuro.

GENERALIDADES

2.1 Antecedentes.

En años pasados la cantidad de datos que se recolectaban se ha incrementado considerablemente y esta cantidad sigue aumentando cada vez más, llegando al punto de hablar de quintillones de bytes generados cada día, esto es una cantidad de información enorme y Big Data va de la mano con este explosivo incremento en la cantidad de información recolectada, el termino Big Data fue propuesto por META Group analyst Doug Laney en el 2001 y se usó para describir el incremento en el volumen, velocidad y variedad de los datos que se venía generando [1] [3], lo que se llegó a conocer como las 3 V's que caracterizan a Big Data.

Con el tiempo la idea de Big Data planteada en ese entonces ha evolucionado, agregando la veracidad y el valor de los datos, lo que nos lleva a hablar ya no de 3 V's, si no de 5 V's. Estas 5 V's comprenden el concepto general que abarca a Big Data, el cual es el almacenamiento de enormes Volúmenes de datos, los cuales tienen una gran Variedad de fuentes y formatos (archivos de texto, videos, audios, datos recogidos por censores, entre otros), estos datos pueden llegar a diferentes Velocidades dependiendo de la fuente y además deben ser Veraces, por último, se evalúa el Valor de los datos, ya que no toda la información recogida es realmente útil [3].

Por lo anterior, Big Data se ha convertido en la apuesta principal para las empresas del sector público y privado y para el gobierno colombiano, quienes junto con el Ministerio de Tecnologías de la Información y las Comunicaciones (MinTIC) y el Departamento Administrativo de Ciencia, Tecnología e Innovación (Colciencias) se unieron con la finalidad de fortalecer las soluciones de análisis de información, creando un centro de excelencia en Big Data y Data Analystics denominado Alianza CAOBA [4].

Esta alianza está integrada por empresas colombianas como lo son: Bancolombia, Nutresa, IBM de Colombia, SAS Institute Colombia, EMC Information Systems Colombia, Cluster CREATIC, Departamento Nacional de Planeación (DNP) y las Universidades ICESI, EAFIT, los Andes y la Pontificia Universidad Javeriana que actúa como ejecutora del proyecto. Alianza CAOBA busca a través de la formación del talento humano, la investigación aplicada y el desarrollo de productos, generar servicios y soluciones innovadoras que le agreguen valor a los sectores estratégicos del país [4].

La compañía colombiana Procalidad S.A. fundada en 1989 por un ingeniero industrial, hoy en día ofrece servicios de consultoría analítica, con principal énfasis en inteligencia de negocios(BI), Big Data y planeación empresarial(PM) para empresas de sectores de seguros, petróleo, energía, bancario y hasta educación [5].

Empresas como Codensa utilizan los servicios de Big Data que ofrece Procalidad S.A. para encontrar las causas y el impacto de la perdida de energía en sus finanzas. Otro ejemplo clave de empresas que usan sus servicios es Claro Colombia, quienes utilizan el poder de los datos

recolectados para saber si sus clientes se sentirán satisfechos con un nuevo plan de datos y minutos próximo a ser lanzado en el mercado [6].

En general los sectores que más utilizan Big Data en Colombia son:

- Sector financiero: Se analiza el comportamiento de los clientes para brindarles nuevos productos y servicios. Por ejemplo, en Medellín-Colombia se utiliza Big Data para identificar patrones sospechosos de movimientos y prevenir lavado de activos o fraudes.

- Sector salud: Se analiza el consumo de medicamentos e identificar el perfil de sus pacientes.

- Sector petrolero: análisis de exploración.

- Sector consumo: Se analiza la información brindada por sus clientes vinculados a las tarjetas de fidelización para de esta manera realizar ofertas a la medida y oportunas [6].

*"En Colombia tenemos que entender que la Data no es un recurso, sino una cultura organizacional; cuando una compañía entiende esto, sus resultados de comunicación, mercadeo y en general del negocio se ven beneficiados considerablemente". (*Camilo Plazas, 2015, Revista Dinero).

Otra de las grandes compañías que ha sacado ventaja de esta tecnología es la empresa comercial estadounidense Netflix, quienes asesorados por Teradata analizaron las reacciones, hábitos y gustos de gran parte de sus clientes y de allí surgió la famosa serie House of Cards la cual se construyó en base a dicho análisis [6].

IBM, junto a cientos de ingenieros que trabajan para su compañía desarrollan un supercomputador capaz de procesar Big Data mediante un sistema cognitivo, mediante el proyecto denominado Watson, cuyo objetivo es facilitar al usuario la toma de decisiones que tienen gran complejidad. El proyecto busca que el ser humano se comunique con Watson y este procese toda la información que tiene y de una respuesta lo más acertada de acuerdo con lo preguntado [7]. *"Watson No tiene una respuesta a nuestras preguntas. Él hace una pesquisa con cada pregunta y, como nunca puede estar seguro de que ha comprendido esa pregunta, hace un estudio de probabilidades. Y regresa con respuestas y ofrece un porcentaje de fiabilidad en cada una. Watson diría que Obama es el presidente de Estados Unidos con un 98% de confianza"*. (Darío Gil, 2015, Revista El País)

Google promueve el servicio de BigQuery una potente plataforma de análisis de Big Data empleada por todo tipo de empresas. Esta plataforma escanea en segundos terabytes y en minutos petabytes de datos [8].

Con el crecimiento de la tecnología Big Data las empresas empezaron a tener grandes retos en cuanto al almacenamiento de datos, así como el procesamiento de los mismos, con esto se hizo necesario la creación de herramientas que facilitaran el uso de esta tecnología, una de las aplicaciones que nació de este hecho es Hadoop. Millones de consultas son realizadas en internet

cada hora, por ejemplo, empresas como Google y Facebook que procesan cantidades enormes de consultas e información, esto produjo que los métodos convencionales de manejo de datos no fueran suficientes para analizar y procesar la inmensa cantidad de datos, en respuesta a esto Google creo el modelo de programación MapReduce; este modelo busca solucionar los problemas generados al procesar dicha cantidad de datos, mediante el manejo de la tolerancia a fallos, el procesamiento paralelo, la distribución de datos, el balance de cargas, la alta escalabilidad y disponibilidad; Hadoop es una implementación de este modelo [9]. Hadoop es un proyecto open-source desarrollado como herramienta para proyectos que requieran computación y almacenamiento distribuido que sea altamente escalable, con el tiempo se convirtió en la implementación del modelo MapReduce más popular debido a sus características, confiabilidad, escalabilidad, computación paralela y distribuida, ya que estas cumplen con el propósito de crear un ecosistema basado en la nube [9] [10].

Hadoop es una tecnología que está cambiando el modo en que las empresas trabajan con Big Data al permitirles el uso de arquitecturas que reducen los costos que normalmente tendría el implementar un sistema Big Data, además de permitir la gran escalabilidad y manejo de infraestructura que se requiere para estos proyectos [11].

A lo largo de este proyecto se han encontrado tesis a nivel nacional e internacional, donde se han abordado estos mismos temas, algunas de ellas son:

- Diseño de un prototipo para la implementación de un sistema Big Data, realizada por los estudiantes Daniel Romero y Carlos Vargas del Politécnico Grancolombiano en el año 2015, donde se expone una investigación realizada sobre el concepto de Big Data y Hadoop y los conceptos tecnológicos que lo rodean, desde el punto de software y hardware. La investigación surge para dar solución a un problema de almacenamiento de información y la velocidad al manipularla en una empresa encargada de almacenar transmisiones generadas por GPS, por ello se realizó un análisis y diseño de un sistema Big Data el cual permitió hacerle frente a este problema [12].

- Diseño y desarrollo de una guía para la implementación de un ambiente Big Data implementado Hadoop, realizada Fabian Andrés Guerrero López y Jorge Eduardo Rodriguez Pinilla de la Universidad Católica de Colombia en el año 2013, donde se buscó la implementación de una arquitectura para crear un ambiente Big Data, teniendo en cuenta aspectos importantes como el software y hardware que se utilizaron para llevar acabo dicho proceso, de igual manera todos los procedimientos que implicaba empezar a utilizar bases de datos no relacionales, para llevar a cabo dicha implementación utilizaron el framework de Hadoop [13].

- Análisis de la viabilidad de la implementación de Big Data en Colombia, realizada por Salinas Hernandez, Hector Javier Reita Reyes y Jorge Eduardo de la Universidad Distrital Francisco José de Caldas en el año 2016, donde se mostró la factibilidad de la implementación de las redes Big Data en Colombia, en un estudio más detallado de manera

enfocada al campo de las comunicaciones, el estudio comprendió análisis de tendencias mundiales, nacionales, tecnologías, costos e innovaciones en el campo de esta tecnología [14].

- Análisis de las posibilidades de uso de Big Data en las organizaciones, realizada por David López Garcia de la Universidad de Cantabria España en el año 2013, donde su objetivo principal consistió en explicar a cabalidad el termino de Big Data y a que hace referencia, seguido de cómo utilizarlo en grandes compañías para sacar ventaja ante sus competidores [15].

- Big Data en sectores asegurador y financiero, realizado por David Ramos Pastor de la Universidad de Barcelona España en el año 2015, donde se explica en que consiste el Big Data, además de las partes por las que está formada una solución Big Data y que se puede mejorar con esta tecnología en las compañías del sector asegurados y financiero [16].

Debido a todo lo mencionado anteriormente se evidencia la necesidad e importante de incrementar las habilidades prácticas en herramientas que permitan el análisis de Big Data como lo es Hadoop.

2.2 Planteamiento del problema.

En la actualidad la institución universitaria Politécnico Grancolombiano brinda a su comunidad estudiantil una clase de Big Data donde se enseña la teoría y manejo de Big Data, en las asesorías los profesores comentan que el tiempo que dura el semestre no es suficiente para desarrollar todas las temáticas, es más, en el último corte del semestre simulan una arquitectura de Hadoop bajo talleres prácticos guiados por él, pero no se alcanza a realizar un taller sobre una arquitectura real de Hadoop para entender mejor su funcionamiento ya que el montaje de la arquitectura es complejo y lleva tiempo realizarlo, además muchos estudiantes no tienen las herramientas en las cuales montar tal arquitectura.

Los estudiantes abordan la temática del curso mediante las lecturas y conocimientos teóricos brindados por el profesor, al realizar el desarrollo de los talleres prácticos el tiempo no es suficiente para implementar lo planteado, ya que, si no se tienen los conocimientos previos en Hadoop, Big Data y el análisis de millones de registros de información, para los estudiantes es supremamente complejo culminar un proyecto al 100% con éxito.

En las asesorías se comenta que sería muy útil tener una herramienta donde se pudiera ejecutar lo planteado en el curso en un ambiente real de Hadoop, de modo que no se tuvieran que preocupar por el montaje de la arquitectura si no solamente por la codificación del análisis de los datos, por este motivo se decide realizar un prototipo de laboratorio Hadoop con Big Data que podría ayudar a la comunidad universitaria a reforzar la comprensión de Big Data mediante el uso de un ambiente real Hadoop que les permita analizar grandes volúmenes de datos utilizando las ETL desarrolladas por ellos mismos y aplicando todos los conocimientos adquiridos en clase.

Con base en lo mencionado anteriormente, se plantea la siguiente pregunta:

¿Podría ser útil implementar un laboratorio Hadoop para análisis Big Data, en la Institución Universitaria Politécnico Grancolombiano?

2.3 Objetivos.

2.3.1 Objetivo General.

Diseñar e implementar un prototipo de laboratorio Hadoop para análisis Big Data, que podría permitir a los estudiantes de la Institución Universitaria Politécnico Grancolombiano aprender a instalar y utilizar un entorno Hadoop para análisis Big Data.

2.3.2 Objetivos Específicos.

1. Diseñar un prototipo de laboratorio Hadoop para análisis Big Data, que permita procesar documentos de texto según el programa MapReduce ejecutado.

2. Instalar un entorno de trabajo Hadoop que cumpla las especificaciones del diseño, para la implementación del prototipo en máquinas virtuales utilizando el sistema operativo CentOS.

3. Desarrollar un ejemplo de programación en MapReduce que le permita al entorno Hadoop analizar un documento de texto de acuerdo con lo especificado en el diseño.

4. Implementar el prototipo elaborado para ejecutar el ejemplo programado en MapReduce y mostrar el resultado del análisis.

2.4 Justificación.

En la actualidad el concepto de Big Data está creciendo rápidamente, dado a los grandes volúmenes de datos generados diariamente, se cree que se pueden generar alrededor de 2.5 quintillones de bytes de datos al día [1]. Big Data nace para solucionar problemas de almacenamiento, análisis e interpretación de grandes volúmenes de datos, dada la cantidad, los tipos y la velocidad con que se deben tratar dichos datos. Debido a la evolución de la tecnología, dispositivos como computadores, tabletas, teléfonos inteligentes, entre otros, que se encargan de generar cantidades enormes de datos, de allí la necesidad de una herramienta que pueda recolectarlos y procesarlos, por ejemplo, empresas como Facebook y Google [9], poseen gran demanda de usuarios que constantemente interactúan en internet para realizar millones de consultas.

Debido al crecimiento de datos para análisis Big Data, nace Hadoop, una tecnología que está cambiando el modo en que las empresas trabajan con Big Data al permitirles el uso de arquitecturas que reducen los costos que normalmente tendría el implementar un sistema Big Data gracias a que

es un software gratuito, además de ser un framework completamente escalable puesto que permite pasar de un servidor a miles de máquinas sin que afecte las que ya están implementadas; tomando de esta manera fuerza en el mercado debido a las grandes ventajas que proporciona su implementación.

Dada la necesidad observada en la Institución Universitaria Politécnico Grancolombiano de tener una herramienta Hadoop que podría reforzar el conocimiento y apoyar el desarrollo de las habilidades prácticas en Big Data, la presente investigación se enfoca en el diseño e instalación de un prototipo de laboratorio Hadoop para análisis Big Data.

Este proyecto le brindara un aporte de investigación a la universidad, además de contribuir con la misión y visón institucional [17] al apoyar el desarrollo de una tecnología que según el Min TIC es de gran importancia para la nación [18]; por ultimo sirve como base a la comunidad académica de la universidad, ya que puede brindar información de conceptos importantes, además de brindar a los futuros ingenieros conocimientos planteados y con muchas más ramas que desarrollar e investigar sobre esta tecnología.

2.5 Delimitación.

2.5.1 Tiempo.

Para el control de tareas y tiempos de entrega se realizó el siguiente cronograma de trabajo *(Ver Tabla 1)*:

Actividad Semanal	**Mes Agosto**			**Mes Septiembre**				**Mes Octubre**				**Mes Noviembre**				**Mes Diciembre**	
	S2	**S3**	**S4**	**S1**	**S2**	**S3**	**S4**	**S1**	**S2**	**S3**	**S4**	**S1**	**S2**	**S3**	**S4**	**S1**	**S2**
Investigación Big Data/ Hadoop	**X**	**X**	**X**														
Elaborar Documento Anteproyecto		**X**	**X**														
Diseñar prototipo de laboratorio Hadoop				**X**	**X**	**X**	**X**										
Instalar un entorno de trabajo Hadoop								**X**	**X**	**X**	**X**						

Desarrollar el ejemplo de programación en MapReduce									X	X	X	X					
Implementar el prototipo elaborado											X	X					
Realizar los instructivos para el desarrollo y uso de la herramienta										X	X	X	X	X	X	X	
Elaborar documento tesis						X	X	X	X	X	X	X	X	X	X	X	
Elaborar y preparar sustentación												X	X	X	X	X	X

Tabla 1 Cronograma Actividades (Fuente: Galeano Cruz y Domínguez Rivera, 2017)

2.5.2 Alcance.

Este proyecto está orientado a la construcción de un prototipo de un laboratorio Hadoop para análisis Big Data en la Institución Universitaria Politécnico Grancolombiano, que comprende el siguiente alcance:

- El diseño abarca toda la estructura del cluster, lo que se requiere para el ambiente de desarrollo y la conexión de estas dos partes.
- La instalación del cluster de acuerdo con lo definido en el diseño, que se compone por un nodo maestro, dos nodos esclavos y una máquina que contiene el ambiente de desarrollo, en esta parte se instala Hadoop y el manejador de base de datos Hbase.
- Desarrollar un ejemplo básico de programación MapReduce donde se analiza un archivo de texto con poca información, dado a que es un prototipo y no se cuenta con los requerimientos suficientes para hacerlo con una gran cantidad de datos.
- En la implementación se incluye pasar el ejemplo de programación en MapReduce al cluster junto con los datos, además de ejecutar Hadoop para que realice el análisis de datos para finalmente mostrar un resultado.

2.6 Metodología

Dada la magnitud y complejidad del proyecto, este se elaborará únicamente a nivel de prototipo con base en un modelo desarrollado en máquinas virtuales para tal fin, estas máquinas utilizaran como sistema operativo a CentOS, el framework Hadoop para el análisis BigData y HBase para la base de datos NoSQL dados los grandes beneficios que brindan estas herramientas; como son:

- Hadoop es un framework gratuito y de código abierto. Es altamente escalable y flexible, ya que permite pasar de un servidor a miles de servidores sin afectar el funcionamiento de

los otros, almacena grandes cantidades de datos y los procesa rápidamente. Posee gran tolerancia a fallos ya que por el volumen de datos que maneja, es necesario tener la información segura y que sea fácil de recuperar.

Se utilizará una metodología de desarrollo basada en el modelo cascada manejado para desarrollo de software definida por etapas, en la cual el desarrollo será dividido en fases que irán en función del cumplimiento de los objetivos del proyecto para al final llegar al cumplimiento del objetivo principal que es *"Diseñar e implementar un prototipo de laboratorio Hadoop para análisis Big Data, que podría permitir a los estudiantes de la Institución Universitaria Politécnico Grancolombiano aprender a instalar y utilizar un entorno Hadoop para análisis Big Data."*

Las fases definidas para el proyecto son las siguientes (*Ver Figura 1*):

1. Fase de levantamiento de información e investigación: Durante esta fase se realizará el levantamiento de información de las características que debe tener el clúster de Hadoop según la necesidad planteada, además, se recopilará el conjunto te conocimientos necesarios para llevar a cabo este proyecto.

2. Fase de diseño: Durante esta fase se realizará el diseño del prototipo de laboratorio Hadoop para análisis Big Data, que permitirá procesar documentos de texto según el programa MapReduce ejecutado, aplicando los conocimientos adquiridos en la fase de Levantamiento de información e investigación.

3. Fase de instalación y montaje: Durante esta fase se instalará el entorno de trabajo Hadoop, para la implementación del prototipo según él diseño realizado durante la fase de diseño.

4. Fase de desarrollo: Durante esta fase se desarrollará un ejemplo de programación en MapReduce que le permita al entorno Hadoop analizar un documento de texto de acuerdo con lo especificado en la fase de diseño.

5. Fase de implementación: Durante esta fase se implementará el prototipo elaborado en la fase de instalación para ejecutar el ejemplo programado en MapReduce realizado en la fase de desarrollo y se mostrará e resultado del análisis.

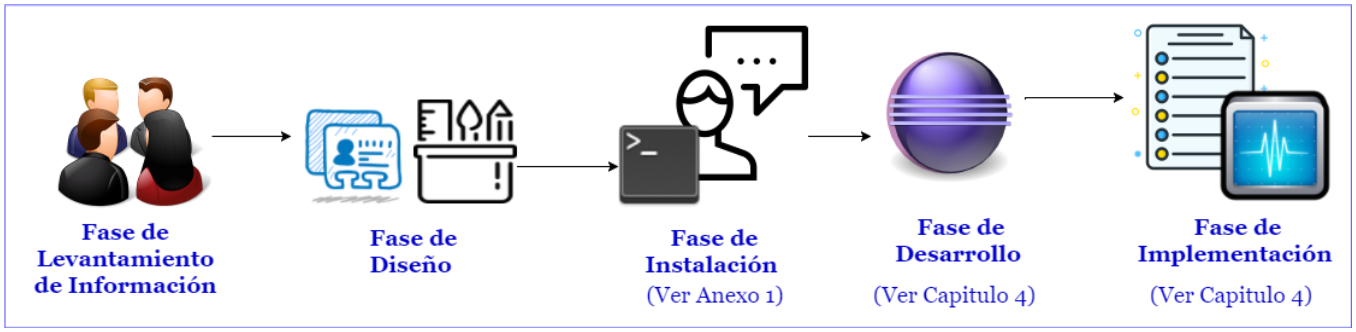

Figura 1 Fases del Proyecto (Fuente: Galeano Cruz y Domínguez Rivera, 2017)

MARCO TEORICO

Con el paso de los años el termino de Big Data ha código fuerza, gracias a un sinfín de tendencias tecnológicas que venían madurando desde la primera década del siglo XXI y que se han consolidado en los últimos años, como lo son la interacción de los usuarios con las redes sociales, el aumento de banda ancha, la conexión a internet a un bajo costo, internet de las cosas, computación en la nube, entre otros. Debido a estas tendencias, el volumen de los datos ha crecido significativamente y por tal motivo, la necesidad de almacenar la información para analizarla, procesarla e interpretarla para dar soluciones a cosas que ni siquiera se creían posibles o eran irrelevantes. Por ende, esta nueva tecnología crece con gran rapidez y se cree que utilizándola adecuadamente puede traer grandes ventajas para las empresas y compañías del mundo [19].

Por lo tanto, en este capítulo se explicará a los lectores el concepto de Big Data, las ventajas que tiene implementarlo, algunas de sus características, los tipos de datos que maneja, qué relación tiene con el framework Hadoop, qué ventajas tiene dicha herramienta, entre otros términos relacionados con esta tecnología que llevaran a una mejor claridad para el desarrollo de este proyecto.

3.1 ¿Que es Big Data?

Cuando se habla de Big Data, se habla de grandes volúmenes de datos y de la necesidad de capturarlos, almacenarlos y analizarlos para conseguir grandes ventajas para las empresas que deciden adoptar esta tecnología [19].

Según la consultora Gartner se define Big Data como activos de información de alto volumen, alta velocidad y variedad, que demanda formas rentables e innovadoras de procesamiento de información que permiten un mejor conocimiento, toma de decisiones y atomización de procesos [20].

Para IBM, Big Data se resume a toda la información que no puede ser procesada o analizadam utilizando herramientas comunes o tradicionales. Sin embargo, el termino como tal no se refiere a una cantidad específica de información, debido a que cuando se habla de Big Data se habla en términos de petabytes y exabytes de datos (Ver Tabla 2). IBM explica que además del gran volumen de información, también existe una gran variedad de datos que a su vez requieren una gran velocidad de respuesta al recibir la información, para obtener la correcta en el momento preciso [21].

Nombre	Símbolo	Equivale
byte	B	8 bits
kilobyte	KB	1 000 B
megabyte	MG	1 000 000 B
gigabyte	GB	1 000 000 000 B
terabyte	TB	1 000 000 000 000 B

petabyte	PB	1 000 000 000 000 000 B
exabyte	EB	1 000 000 000 000 000 000 B
zettabyte	ZB	1 000 000 000 000 000 000 000 B
yottabyte	YB	1 000 000 000 000 000 000 000 000 B
saganbyte	SB	1 000 000 000 000 000 000 000 000 000 B
jotabyte	JB	1 000 000 000 000 000 000 000 000 000 000 B

Tabla 2 Unidades de Medida (Fuente: Galeano Cruz y Domínguez Rivera, 2017) [22]

En resumen, Big Data hace referencia a una cantidad de datos que supera la capacidad del software actual, donde sus principales características son: velocidad, veracidad, variedad y valor, las cuales contribuyen para poder capturar los datos, almacenarlos, procesarlos y analizarlos en un tiempo razonable, pero que mediante una información bien analizada se puede sacar gran provecho para las grandes empresas y compañías que deseen asumir el reto con esta tecnología.

3.2 Las 7 Vs del Big Data

Las características más importantes del Big Data son volumen de información, velocidad, variedad y veracidad de los datos, conocidas como las cuatro V del Big data, además del valor de los datos, aunque en la actualidad, en los últimos artículos se empieza hablar de las siete V del Big Data, agregando a las mencionadas, viabilidad y visualización de los datos [23].

3.2.1 Volumen de la información

Hace referencia a la cantidad de los datos. Es la característica principal de Big Data, debido a la cantidad masiva de datos que intentan analizar las empresas para la toma de decisiones y la gran ventaja que pueden sacar de esto. Según encuesta realizada por IBM se habla de Big Data cuando se trata de petabytes y zetabytes de datos [24].

3.2.2 Velocidad de los datos

Hace referencia a los datos en movimiento, la rapidez en la que se crea se almacena y se procesa la información en tiempo real. Debido a la cantidad de datos que se generan en la actualidad por los usuarios es difícil que un software tradicional logre capturar toda la información necesaria para analizarla, por ello es fundamental que se tenga una buena velocidad sobre todo para compañías que manejan procesos en tiempo real, como los bancos para la detención de fraude en las transacciones bancarias [24].

3.2.3 Variedad de los datos

Se refiere a las formas, tipos y fuentes de datos. Hace referencia a gestionar la complejidad de diversos tipos de datos, esto debido a que las empresas requieren analizar información que tiene todo tipo de formas, por ejemplo, las redes sociales, páginas webs, correos electrónicos, documentos de texto, blogs, imágenes, videos, audios, inclusive el análisis de clics que hace un

usuario para realizar un proceso, todos estos tipos de datos hacen parte de los estructurados, semiestructurados y no estructurados [24].

3.2.4 Veracidad de los datos

Trata de la incertidumbre de los datos. En otras palabras, es la fiabilidad de la información recolectada, es decir, se necesitan datos de calidad que sean confiables, siendo este uno de los grandes y más importantes retos para Big Data [24].

3.2.5 Viabilidad

Busca la viabilidad de la implementación de Big Data y el éxito empresarial que esto puede traer consigo. El termino va muy de la mano con la inteligencia empresarial que busca mirar que tan viable es realizar la implementación de esta tecnología y de que datos son pertinentes almacenar y analizar, es decir se debe filtrar y seleccionar la información, para de allí sacar frutos que le interesen a la empresa, para el desarrollo y la toma de decisiones de la misma [23].

3.2.6 Visualización de los datos

Consiste en el modo en el que los datos son presentados. Una vez son procesados los datos, la visualización se enfoca en representar visualmente los datos de manera legible y accesible, de modo que se pueda interpretar la información y encontrar patrones con facilidad y que lleven a un fin acerca tema que se está analizando [23].

3.2.7 Valor de los datos

El valor se obtiene de los datos que se transforman en información valiosa y útil, que luego se convierte en conocimiento y genera una acción que lleva consigo la toma de una decisión. En conclusión, que las empresas puedan tomar las mejores decisiones con base en lo encontrado, por ejemplo, una red social puede tomar los datos expresados por los usuarios en su lenguaje natural y puede analizar mediante esta información, sentimientos ya sean positivos o negativos [23].

3.3 Tipos de Datos

Los tipos de datos que puede manejar Big Data se dividen en tres grandes categorías, estructurados, semiestructurados y no estructurados, los cuales se explican a continuación:

3.3.1 Datos Estructurados

Son datos con un formato o esquema que posee campos fijos, la mayoría de los datos en la actualidad se manejan de manera estructurada, ya que es la más tradicional, puesto que en estas

fuentes los datos se encuentran bien definidos debido a que poseen un formato fijo que especifica en su gran mayoría todos los detalles, por ejemplo las bases de datos relacionales, generan hojas de cálculo y archivos planos con la información, en estos formatos se definen los campos, el orden que deben llevar y el tipo de dato, ya que estas especificaciones facilitan el trabajo con dichos datos; se llega a tal punto de especificar que una fecha es en formato (DD/MM/AAAA), que un documento de identidad debe estar compuesto por 10 dígitos o un teléfono fijo debe llevar 7 dígitos, entre otros [19]. Los datos pueden ser:

- Creados: generados por el sistema de una manera definida, por ejemplo, registros de tablas o ficheros XML.

- Provocados: datos creados de manera indirecta a partir de una acción generada previamente, evaluar una película, un buen restaurante.

- Por transacciones: datos generados al finalizar correctamente una acción previa, por ejemplo, un recibo de un cajero automático.

- Compilados: resúmenes de datos de una empresa, por ejemplo, los datos recopilados en un censo nacional realizado.

- Experimentales: datos generados como pruebas o simulaciones para verificar la viabilidad de un negocio. [25]

3.3.2 Datos Semiestructurados

Son datos que no tienen un formato fijo que pueden tenerlo, pero no son de fácil compresión para el usuario, contienen etiquetas y marcadores que separan los elementos dato, es decir los funcionales. Se requieren usos de reglas complejas para la lectura de cada bloque de información que determinan como procedes después de realizar dicha lectura. Por ejemplo, el texto de etiquetas HTML y XML [19].

3.3.3 Datos No Estructurados

Son datos sin formato ni tipo definido, se almacenan sin una estructura uniforme y se tiene muy poco o más bien nada de control sobre estos. Por los general son datos provenientes de textos, videos, imágenes, audios, documentos impresos, mensajes de correo electrónico, libros, mensajería instantánea como WhatsApp o Messenger, artículos, libros, etc. Se conoce como datos no estructurados, a la información de una empresa que no se encuentra almacenada en una base de datos relacional, sino que se encuentra dispersa en los usuarios que trabajan en ella [19]. Los datos pueden ser:

- Capturados: hacen referencia a la información capturada para un futuro beneficio de un usuario, por ejemplo, cuando una persona ingresa a Google a realizar una consulta.

- Generados por el usuario: están compuestos por todos los datos que se generan diariamente por todos los usuarios en internet, desde Facebook, Twitter, videos en YouTube, etc. [25]

Gracias a los datos no estructurados nace MapReduce, Hadoop, las bases de datos NoSQL, que buscan analizar dichos datos de una manera más fácil.

3.4 Tipos de Datos por Origen

3.4.1 Web y Redes Sociales

Implica la información de la web, como las búsquedas en Google y las publicaciones de redes sociales como Facebook, Twitter, inclusive la información de los clicks dados en una página para realizar alguna acción.

3.4.2 Comunicación entre Maquinas

Es la información generada entre dispositivos o máquinas, allí se integra el uso de sensores, medidores que capturan un evento que es transferido a otra máquina, por ejemplo, las señales GPS.

3.4.3 Transacciones

Incluye registro de llamas, mensajería, pagos con tarjeta o de forma online. Estos datos pueden semiestructurados como no estructurados.

3.4.4 Biométrica

Datos como huellas digitales, reconocimiento facial, escaneo de retina, ADN.

3.4.5 Generados por personas

La mayoría de los datos generados actualmente son por los humanos quienes contribuimos en gran medida desde diferentes fuentes como, por ejemplo, una llamada, una nota de voz, un correo electrónico, un mensaje de texto. [25]

3.5 Usos de Big Data

	Descubrimiento Científico	Nueva Tecnología	Manufactura y Transporte	Servicios Personales Campañas	Medio Ambiente Infraestructura	Cuidado de la Salud
Ciencia	+ + + + +	+ + + +	+	-	+ +	+ + +
Telecomunicaciones	+	+ + + +	+ +	+	+ + + +	+
Industria	+ +	+ + + +	+ + + + +	-	-	+ +
Negocio	+	+ + +	+ +	-	+	+ +
Vida Medio Ambiente	+ +	+ +	+ +	+ +	+ + + + +	+
Social Media	+	+ +	-	+ + + +	+ +	-
Cuidado de la Salud	+ + +	+ +	-	-	+ +	+ + + + +

Tabla 3 Usos de Big Data (Fuente: Defining the Big Data Architecture Framework) [26]

La tabla de usos de Big Data *(Ver Tabla 3)* se realiza una comparación entre el origen de los datos de Big Data (Columnas) y el objetivo de uso de los datos de Big Data (Filas), el símbolo más (+) representa el uso que se le ha dado al dato de acuerdo con su origen; el símbolo menos (-) representa que no tiene uso el dato.

3.5.1 Beneficios del Big Data

Con la tecnología de Big Data, las empresas pueden mejorar sus estrategias para posicionarse muy bien en el mercado, debido al gran volumen de datos o información que se maneja a diario por los procesos realizados en la empresa o hasta por los mismos empleados, Big Data puede analizar estos datos y puede contribuir en la empresa para ofrecer mejores productos o hasta para tener buena relación con sus clientes. Para que una empresa puede implementar Big Data, debe tener en cuenta los siguientes pasos [27]:

- Entender el negocio y los datos, se debe tener un análisis de los empleados de la empresa y de los procesos que manejan.
- Determinar los problemas y como los datos pueden ayudar.
- Definir metas alcanzables.
- Trabajar en paralelo con el sistema actual y con el proyecto que se desea implementar de Big Data.
- Ser flexible con la metodología y las herramientas que ofrece Big Data.
- El principal objetivo siempre debe ser Big Data, ya que el proceso es tedioso y muy pesado de manejar.

3.6 Arquitectura de Big Data

3.6.1 Ciclo de Vida

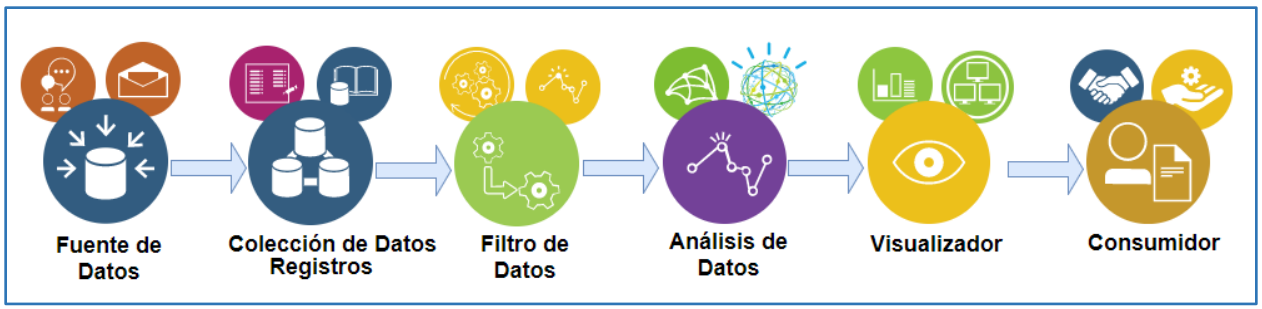

Figura 2 Ciclo de Vida Big Data (Fuente: Defining the Big Data Architecture Framework) [26]

En la figura *(Ver Figura 2)* se muestra el ciclo de vida de Big Data, el cual parte desde una fuente de datos, de allí se sacan los datos que se necesitan y se crea una colección de registros, con dicha colección se realiza un filtro de datos para luego analizarlos con diferentes algoritmos y obtener un resultado del análisis, esta información es procesada de manera que pueda ser visualizada de una forma entendible para el usuario, quien es finalmente el que consume el servicio y saca ventaja de ello.

3.6.2 Infraestructura y Herramientas Analíticas

Como se muestra en la figura *(Ver Figura 3)*, el proceso inicia con el origen de los datos (Big Data Source), allí se pueden encontrar datos registrados por sensores, por experimentos, por redes sociales, entre otras fuentes de datos. Dependiendo de del uso que se va a dar a la información, se puede almacenar de forma general, ya sea almacenamiento para propósitos generales (Storage General Purpose) o almacenamiento de computación para propósitos generales (Compute General Purpose); o también se puede almacenar de forma especializada como, por ejemplo, computación de altas prestaciones (High Performance Computer Clusters) o almacenamientos especializados de archivos (Storage Specialised Databased Archives). Toda esta información almacenada depende completamente del consumidor que es el Big Data Target, que es la que va a usar la información dependiendo de su objetivo, ya sea para usuarios, para objetos etc. Luego pasa a una infraestructura de acceso a la información, de allí pasa por las herramientas analíticas de Big Data (Big Data Analytic/Tools) donde se puede analizar la información por medio de refinamiento, unión, fusión, también se puede analizar en tiempo real y de forma interactiva entro otros.

En el Data Management se muestra cómo se va a manejar la información, según las categorías de datos que pueden ser de: metadata, estructurados, no estructurados, identificables, no identificables.

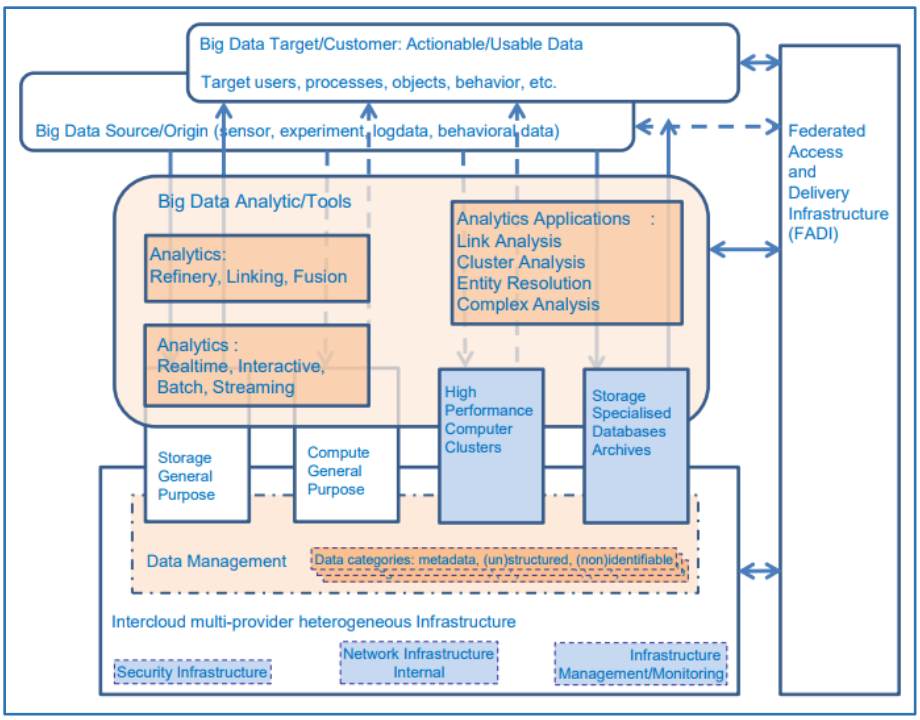

Figura 3 Infraestructura y Herramientas Analíticas Big Data (Fuente: Defining the Big Data Architecture Framework) [26]

3.7 Bases de Datos Relacionales

Una base de datos relacional es una colección de elementos de datos con unas relaciones determinadas entre ellos, esta colección de elementos se organiza en un conjunto de tablas que contienen filas (registros) y columnas (campos). Los registros representan los objetos descritos en una tabla y los campos son los atributos de los objetos. Las bases de datos relacionales comparten campos entre tablas, de este modo pueden establecer relaciones para realizar consultas complejas [28].

3.7.1 Aspectos Importantes de las Bases de Datos Relacionales

3.7.1.1 SQL

Sus siglas traducen lenguaje de consulta estructurada, es la interfaz principal para gestionar bases de datos relacionales, permite realizar diferentes tipos de operaciones, así como registrar, modificar, eliminar y consultar datos, además del procesamiento de transacciones y la administración de dicha base de datos, entre otros [28].

Algunos motores de bases de datos relacionales son:

- Oracle
- MySQL

- Microsoft SQL Server
- PostgreSQL

3.7.1.2 Integridad de Datos

Hace referencia a la totalidad, precisión y coherencia general de los datos; las bases de datos relacionales contienen un conjunto de restricciones para aplicar integridad a los datos, estas ayudan a generar unas reglas de negocio para garantizar la fiabilidad y precisión de los datos [28].

3.7.1.3 Transacciones

Es una o más sentencias SQL que se ejecutan y forman una unidad lógica única de trabajo, si una transacción es ejecutada con éxito todas las modificaciones que incluida son tenidas en cuenta de la base de datos, de lo contrario, si no tiene éxito debe cancelarse y no debe realizar ninguna modificación [28].

3.8 Base de Datos NoSQL

Son aquellas bases de datos que no requieren un modelo clásico de bases de datos relacionales, con el fin de tener facilidad de desarrollo, un desempeño más escalable, alta disponibilidad al trabajar con grandes volúmenes de información. A diferencia de las bases de datos relacionales, las NoSQL no requieren tablas fijas, generalmente no soportan sentencias join para garantizar un mejor desempeño, tienen dificultad para con el manejo de transacciones, debido a la cantidad de información que manejan, no tienen una estandarización por lo tanto cada base de datos maneja su propio lenguaje [29]. Algunas de las características que tiene una base de datos NoSQL son [13]:

- **Distribuido:** normalmente un sistema NoSQL funciona en un sistema distribuido donde varias máquinas están conectadas entre sí para trabajar en conjunto, generalmente la información se replica en varias máquinas para mejorar la redundancia y tener alta disponibilidad.

- **Escalabilidad horizontal:** hace referencia a unir nodos de forma dinámica, es decir que al agregar uno de ellos no afecta a los demás.

- **Construido para grandes volúmenes:** las bases de datos NoSQL están diseñadas para almacenar y procesar cantidades de datos muy grandes de forma fácil y rápida.

- **Modelos de datos no relacionales:** permiten estructuras más complejas y no son tan estrictas como el modelo de datos relacional.

- **No hay definiciones de esquema:** No existen esquemas ni estructuras fijas para almacenar los dados, ya que es va de acuerdo con lo que el cliente desee y a la necesidad que presente, es decir la estructura de los datos no está predefinida.

A continuación, se muestran algunas de las categorías de bases de datos NoSQL más usadas [30]:

3.8.1 Almacenes Key-Value

Estas bases de datos almacenan valores identificados por una clave, de tal manera que no importa el tipo de contenido que tenga la base de datos lo único importante es la clave y el valor que tiene asociado dicha clave, este tipo de almacén de datos es muy eficiente para realizar lecturas y escritas, además de garantizar la escalabilidad puesto que los valores de los datos se pueden particionar de acuerdo con sus claves para una fácil utilidad.

3.8.2 Bases de Datos Columnares

Son bases de datos que guardan sus datos en columnas en lugar de reglones que es la estructura tradicional que manejan las bases de datos relacionales, esta categoría de base de datos por lo general funciona para aplicaciones donde se realizan muchas lecturas, porque para realizar escrituras no es muy eficiente.

3.8.3 Bases de Datos Orientadas a Documentos

Esta base de datos funciona igual que los almacenes Key-Value, la única diferencia es que los valores no se guardan en binario, sino en un formato definido que puede ser JSON, XML o cualquier otro, de esta manera la base de datos puede leer la información y así permiten realizan consultas muy avanzadas, inclusive permite realizar relaciones entre los datos.

3.8.4 Bases de Datos Orientadas a Grafos

Son bases de datos que almacenan los datos en forma de grafos, de tal forma que se le da importante a los datos y también a las relaciones que existen entre ellos, dichas relaciones pueden tener atributos y se pueden consultar directamente. Se habla de que este tipo de base de datos es más eficiente debido a que es más fácil navegar entre relaciones de los grafos que en una base de datos relacional.

3.8.5 Bases de Datos Orientados a Objetos

Son bases de datos en donde la información que se guarda, no son registros ni documentos, sino que se guardan objetos como los definidos en el paradigma de programación orientado a objetos, dado el paradigma de programación se incorporan conceptos importantes como herencia, polimorfismo, encapsulamiento entre otros.

3.9 Ejemplos de Bases de Datos NoSQL

3.9.1 HBase

Es una base de datos distribuida no relacional de código abierto, almacena datos de forma Key-Value y permite tipos de datos estructurados, semiestructurados y no estructurados; esta base de datos almacena y recupera sus datos de forma aleatoria. Funciona únicamente para cluster de ordenadores, es decir no puede ejecutarse en un solo ordenador, además de no permitir consultas SQL. Posee una escalabilidad horizontal, debido a que si se le desean agregar más servidores al cluster o remplazar alguno no presentara ningún inconveniente o falla [27].

3.9.2 DynamoDB

Esta base de datos desarrollada, probada y administrada por Amazon, este servicio es económico y permite guardar gran cantidad de información. Los datos se almacenan en unidades de estado sólido SSD, ya que permiten mayor velocidad al querer consultar información [27].

3.9.3 MongoDB

Es una base de datos no relacional de código abierto, está orientada a documentos; tiene una gran potencia y es de fácil manejo, además de su capacidad para manejar pocos o grandes volúmenes de datos. MongoDB permite las operaciones de CRUD, para almacenar y recuperar los datos utiliza JSON, para ocupar menos espacio al almacenar los datos utiliza BSON, que es una forma binaria de JSON. Una de sus características principales es que puede realizar consultas dinámicas, es decir sin demasiada planificación [27].

3.10 Conceptos Relacionados a Big data

3.10.1 Inteligencia de Negocios

BI (Business Intelligence) en español inteligencia de negocios, se define como el conjunto de herramientas, procesos, tecnología, datos, estrategias, que tiene como objetivo convertir datos almacenados en información valiosa, esta información se convierte en conocimiento, que lleva al diseño de una estrategia para la empresa o facilita la toma de decisiones.

3.10.1.1 Arquitectura de Inteligencia de Negocios

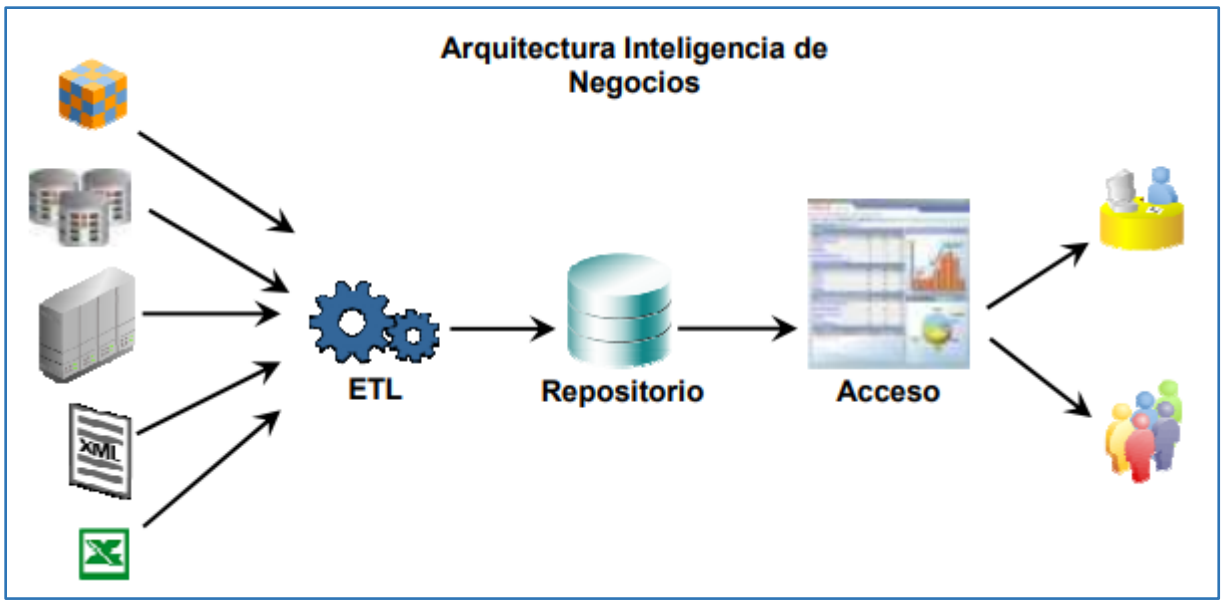

Figura 4 Arquitectura Inteligencia de Negocios (Fuente: Oracle) [31]

En la figura *(Ver Figura 4)* se muestra la arquitectura de inteligencia de negocio, la cual se divide en:

- **Datos**: inicia con una fuente de datos tal como se muestra pueden ser archivos planos, XML, hojas de Excel, bases de datos relacionales, etc.

- **ETL:** se denomina ETL, al proceso de mover datos desde muchas fuentes, transformarlos y limpiarlos, para luego cargarlos en un repositorio; a este proceso se le denomina mapeo ya que allí se define que campos se van a utilizar.

- **Repositorio:** es donde se encuentran cargados los datos definidos por el proceso de ETL, allí se centraliza la información, los datos transformados son mostrados visualmente en tablas de datos.

- **Motor BI:** permite administrar consultas, habilitar componente, realizar cálculos y monitorearlos, este proceso es intermedio entre el repositorio y el acceso a usuarios.

- **Acceso a Usuarios:** es una interface que permite al usuario visualizar los datos de forma entendible de acuerdo con los resultados de las consultas, generalmente se muestran graficas representar mejor la información [31].

3.10.2 Minería de Datos

Data Mining en español minería de datos, se define como el proceso de identificar la información que se puede procesar de los conjuntos de grandes volúmenes de datos. Su objetivo principal consiste en extraer información mediante análisis matemáticos para encontrar patrones

y tendencias que posee la información. Debido a la cantidad de datos no es fácil encontrar los patrones de manera tradicional, por este motivo se utiliza la minería de datos [32].

3.10.2.1 Pasos de la Minería de Datos

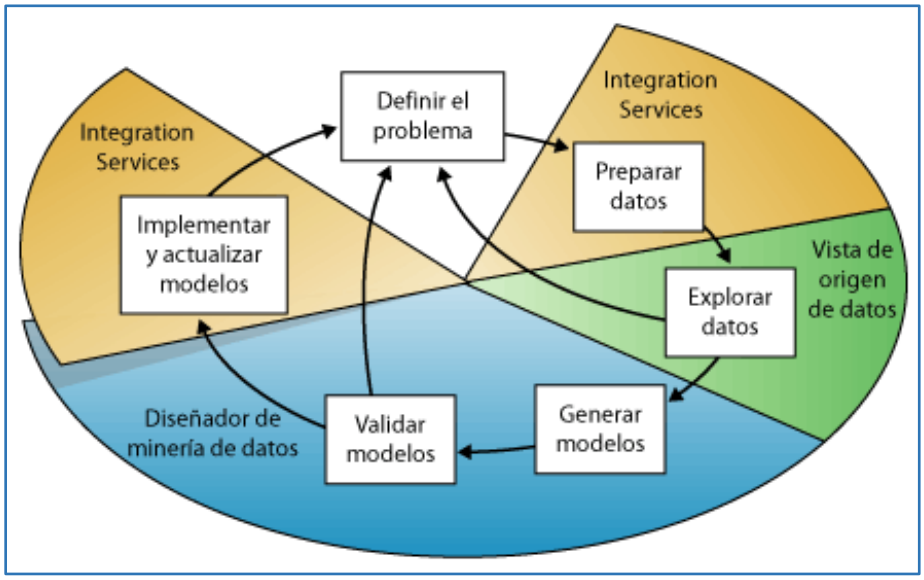

Figura 5 Pasos de la Minería de Datos (Fuente: Microsoft) [32]

La figura *(Ver Figura 5)* muestra un diagrama cíclico que define seis pasos básicos para la minería de datos:

- **Definir el Problema:** consiste en definir el problema principal de manera clara y mirar que datos de los que se tienen pueden servir para llegar a dar una solución a dicho problema.

- **Preparar Datos:** hace referencia a limpiar los datos, es decir a preparar los datos que me son útiles, esto implica quitar datos no validos o interpolar los vacíos, esto se debe a que en una empresa se manejan datos de todo tipo almacenados en formatos diferentes.

- **Explorar Datos:** este proceso se realiza con el fin de conocer los datos para crear modelos de minería de datos adecuados que funcionen para identificar lo que se desea, esta exploración se realiza con el objetivo de determinar si los datos son confiables.

- **Generar Modelos:** aquí se debe generar el modelo de minería de datos de acuerdo con lo obtenido al explorar los datos.

- **Explorar y Validar los Modelos:** en este punto se explora el modelo o los modelos generados en el paso anterior y se valida que funcionen correctamente, por lo general se crean varios modelos para mirar cual es el más acertado a la solución de un problema.

- **Implementar y Actualizar los Modelos:** finalmente se debe implementar el modelo o los modelos de minería de datos que mejor funcionen en producción [32].

3.11 Hadoop

Apache Hadoop es un framework que permite el almacenamiento y procesamiento de cantidades enormes de datos en cluster, para almacenar la información utiliza HDFS que es el sistema de archivos distribuidos de Hadoop y los algoritmos para realizar cálculos mediante MapReduce. Este framework está diseñado para ampliar el sistema, ya que permite pasar de un nodo a miles de nodos de manera muy ágil, además tiene la capacidad de detectar las fallas.

Hadoop permite que las tareas que se van a analizar se dividan en fragmentos de trabajo y se distribuyan en miles de nodos, ofreciendo de este modo un tiempo de análisis menor y un sistema de almacenamiento distribuido para enormes cantidades de datos. En cuando a la programación, ofrece un enfoque muy sencillo, nada complejo comparado a las implementaciones distribuidas anteriormente [33].

Hadoop implementa un mecanismo potente para el análisis de datos, sus características principales son:

3.11.1 Gran Capacidad de Almacenamiento

Hadoop puede utilizar un cluster con millones de nodos para ofrecer un almacenamiento enorme de datos y una gran potencia de procesamiento a un precio asequible para las empresas.

3.11.2 Procesamiento Distribuido con Acceso a Datos Rápido

Hadoop además de almacenar de manera eficiente gran cantidad de datos, tiene la capacidad de proporcionar un acceso rápido a dichos datos. Anteriormente las aplicaciones de computación paralela presentaban problemas para distribuir las tareas en los nodos disponibles del clister, hoy en día Hadoop desplaza la ejecución hacia los datos, de forma tal que traslada las aplicaciones a los datos logrando así un alto rendimiento. Hadoop no procesa los datos aleatoriamente sino secuencialmente de esta forma disminuyen en gran parte la carga de entradas y salidas.

3.11.3 Fiabilidad, Tolerancia a Fallos y Capacidad de Ampliación

Hadoop genera una gran fiabilidad, pues debido a como fue diseñado e implementado detecta fallos y vuelve a realizar la ejecución utilizando nodos diferentes, además soporta gran capacidad de ampliación ya que permite añadir varios servidores al cluster y utilizarlos para el almacenamiento y procesamiento de los datos [33].

3.12 Arquitectura Principal de Hadoop

3.12.1 HDFS

Hadoop distributed File System en español sistema de archivos distribuido Hadoop, es el sistema de almacenamiento de ficheros distribuidos, fue creado por Google File System (GFS). Está diseñado para reducir las entradas y salidas en la red, además de su optimización para trabajar con ficheros grandes de lecturas y escrituras. Tiene gran disponibilidad y escalabilidad debido a la recolección de datos y a la tolerancia a fallos [33].

- **NameNode:** Se trata de la maquina maestra del cluster, se encarga de regular el acceso a los ficheros por parte del cliente, controla el sistema de ficheros que tiene cada nodo y los bloques de cada uno de estos y los mantiene en memoria *(Ver Figura 6)* [34].

- **DataNode:** Son los nodos conectados a la maquina maestra del cluster, son los encargados de leer y escribir los requerimientos de los clientes. En cada no se encuentran replicados los ficheros que están compuestos por bloques *(Ver Figura 6)* [34].

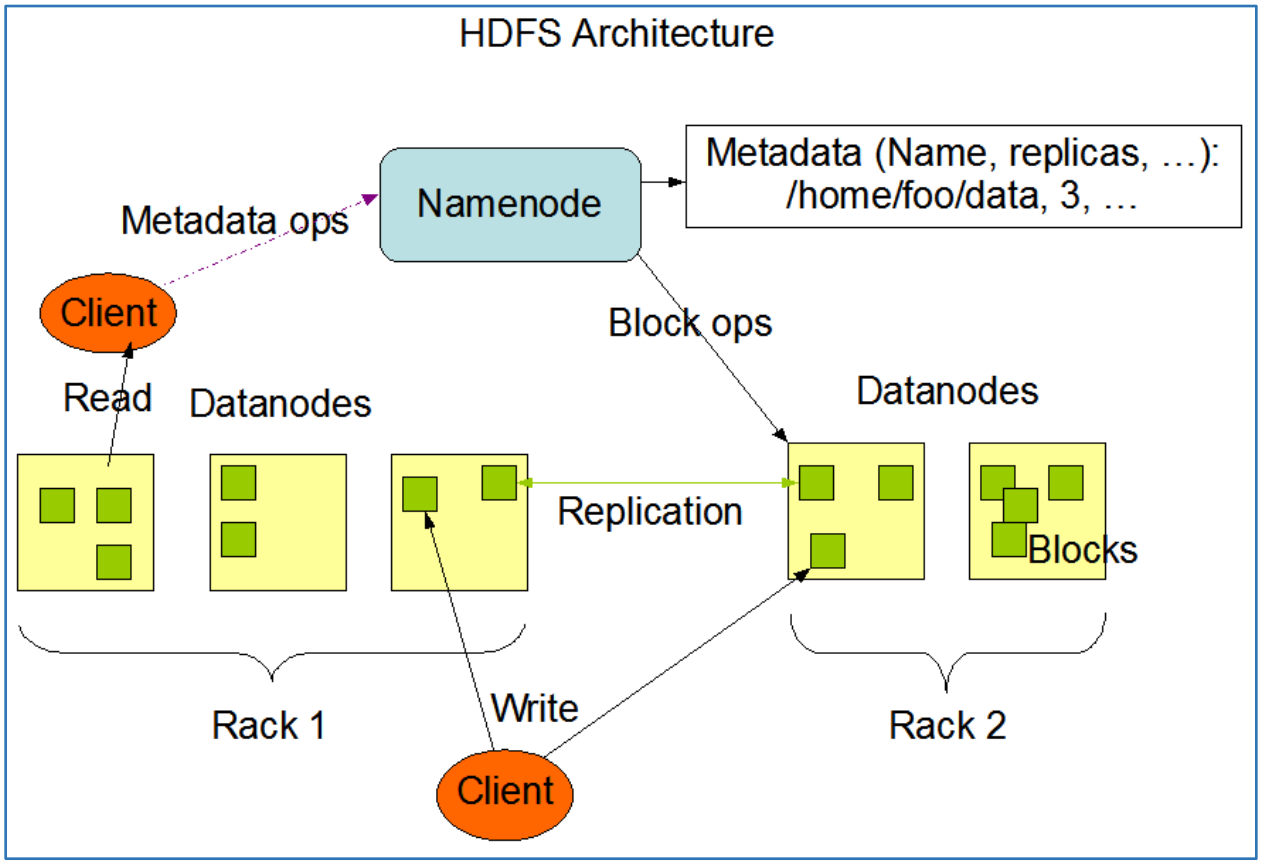

Figura 6 Arquitectura HDFS (Fuete: Apache-Hadoop) [34]

3.12.2 MapReduce

Es un modelo de programación para trabajos distribuidos, se encarga de procesar datos en paralelo, para luego realizar un proceso de mapeo (map) y otro de reducción (reduce). La idea principal es que los desarrolladores escriban el código, programando las tareas que debe realizar el MapReduce para que Hadoop lo ejecute utilizando HDFS para acceder rápidamente a los datos [35].

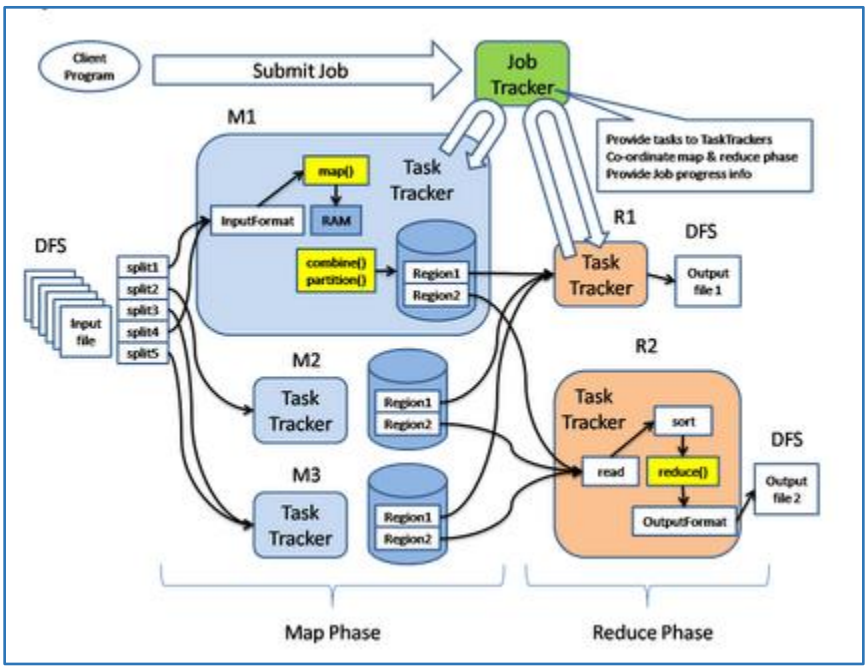

Figura 7 Funcionamiento MapReduce (Fuete: Apache-Hadoop) [35]

En la figura *(Ver Figura 7)* se muestra cómo funciona un programa en MapReduce, también denominado Job, inicialmente la ejecución del programa arranca cuando el cliente envía la configuración al JobTracker, este servicio de Hadoop se encarga de distribuir las tareas de MapReduce a nodos específicos del cluster. Esta configuración especifica las funciones Map, Shuttle (combina) y Reduce, además de las E/S de los datos [35].

Figura 8 Ejemplo Flujo Lógico de Procesos MapReduce (Fuente: Revista Cubana de Ciencias Informáticas) [36]

- **Función Map:** trabaja con grandes volúmenes de datos, se encarga de dividirlos en varias partes, cada una de ellas con colecciones de registros, luego la función map se ejecuta por cada colección de datos, para finalmente calcular un conjunto de valores intermedios basados en el procesamiento de cada registro *(Ver Figura 8)*.

- **Función Reduce:** se ejecuta por cada elemento del conjunto de valores intermedios obtenido, lo que hace es reducir el conjunto de los valores que comparten una clave para obtener un conjunto más pequeño *(Ver Figura 8)* [36].

DESARROLLO DEL PROYECTO

4.1 Fase de levantamiento de información e investigación

En la actualidad la institución universitaria Politécnico Grancolombiano brinda a su comunidad estudiantil una clase dirigida por él, donde se enseña la teoría y manejo de Big Data, pero se expone que el tiempo no es suficiente para desarrollar todo el contenido del curso; en el último corte se realiza la simulación de una arquitectura Hadoop, pero nunca se alcanza a realizar el montaje de una arquitectura real, ya que es muy complejo y toma tiempo. Por otro lado, se indica que la metodología de enseñanza para este curso es mediante lecturas, talleres guiados por el profesor y sus conocimientos teóricos.

De acuerdo con lo anterior, se evidencia que sería muy útil tener una herramienta donde se pudiera ejecutar lo planteado en el curso en un ambiente real de Hadoop, que permita a los estudiantes aplicar las ETL desarrolladas para obtener el análisis de datos, utilizando lo enseñado en clase. Por este motivo, se planteó realizar un clúster con Hadoop que contiene un nodo maestro y dos esclavos, donde la idea principal es que los estudiantes utilicen la herramienta para realizar mediante el desarrollo programado en MapReduce el análisis de datos.

4.2 Fase de Diseño

Se plantea un prototipo donde los estudiantes puedan montar el ambiente de desarrollo en sus propias maquinas utilizando el IDE de desarrollo que mejor se ajuste a sus necesidades para desarrollar el programa MapReduce que se encargara de analizar y procesar los datos que ellos hayan elegido, además de almacenarlos en una base de datos NoSQL, para luego desarrollar el programa que se encargara de recoger esos datos ya procesados y mostrar el resultado final del análisis de una forma gráfica entendible y que sea útil para un usuario final.

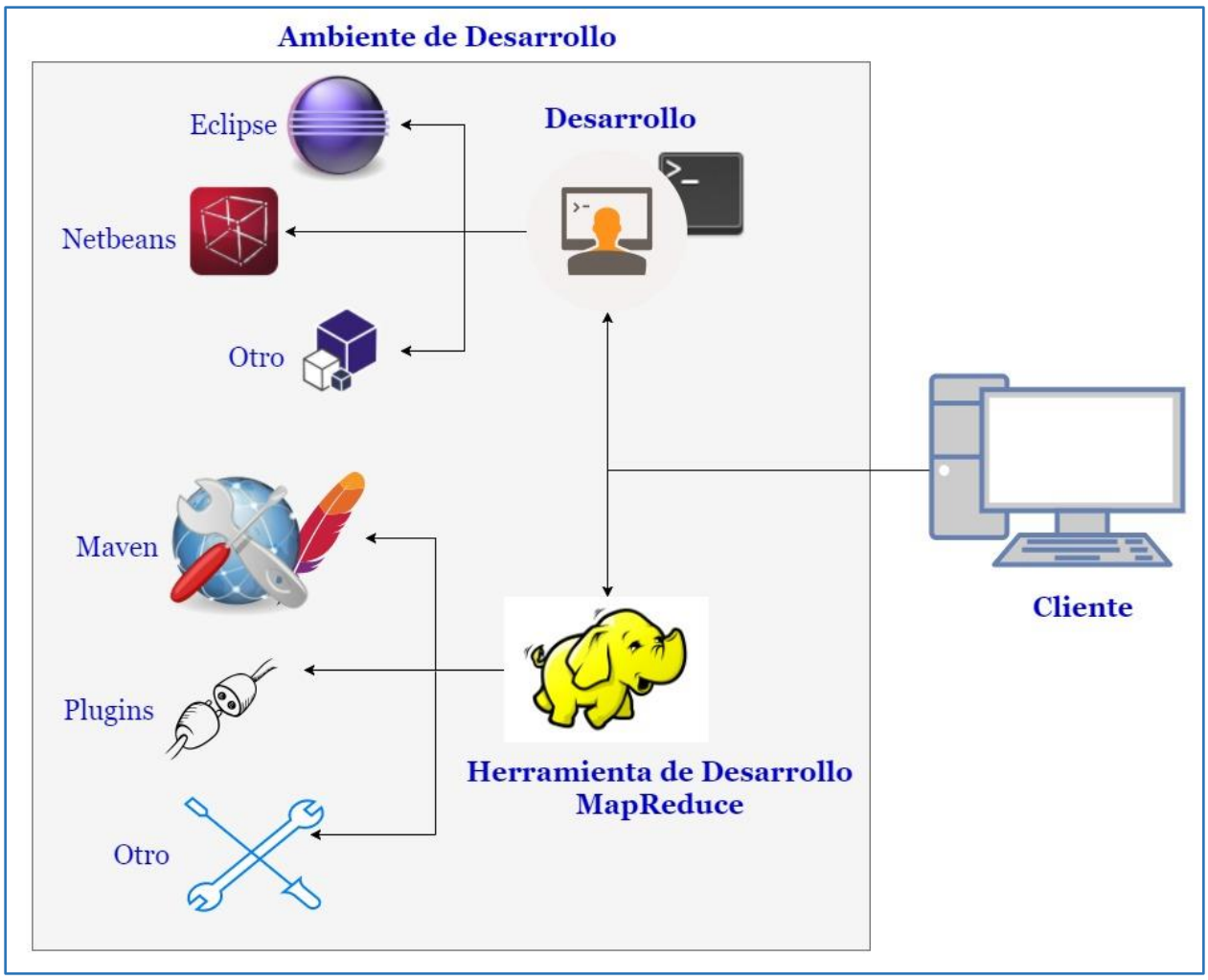

Figura 9 Ambiente de Desarrollo (Fuente: Galeano Cruz y Domínguez Rivera, 2017)

En la *figura 9* se muestra el diseño de lo que requeriría la maquina cliente, que en este caso serían el ambientes de trabajo de los estudiantes; la maquina cliente es la que debe contener el ambiente de desarrollo en el cual trabajaran los estudiantes, esta debe contar con IDE de desarrollo y una herramienta que permita escribir el código MapReduce, se recomienda el uso de Eclipse y Maven, pero su uso no es restrictivo (se proporcionaran manuales para la instalación de Eclipse y Maven, además de un ejemplo de cómo programar en MapReduce), una vez se construye el programa MapReduce se compila en un archivo.jar. Una vez desarrollado el programa para el análisis, procesamiento y almacenamiento de los datos se debe construir el programa que realice la extracción de esos datos y la modele en algún tipo de grafico utilizando d3.js que permita visualizar los datos de manera entendible y que aporten al que los vea, una vez se construya el programa se compila en un archivo.jar.

Cuando se tienen los dos programas realizados se envían al clúster de Hadoop donde se procesarán. La arquitectura del clúster se ilustra en la *figura 10*, esta se compone por:

- Un nodo maestro, en donde se alojará el NameNode y el Secondary NameNode.

- Dos nodos esclavos, los cuales serán los DataNodes.

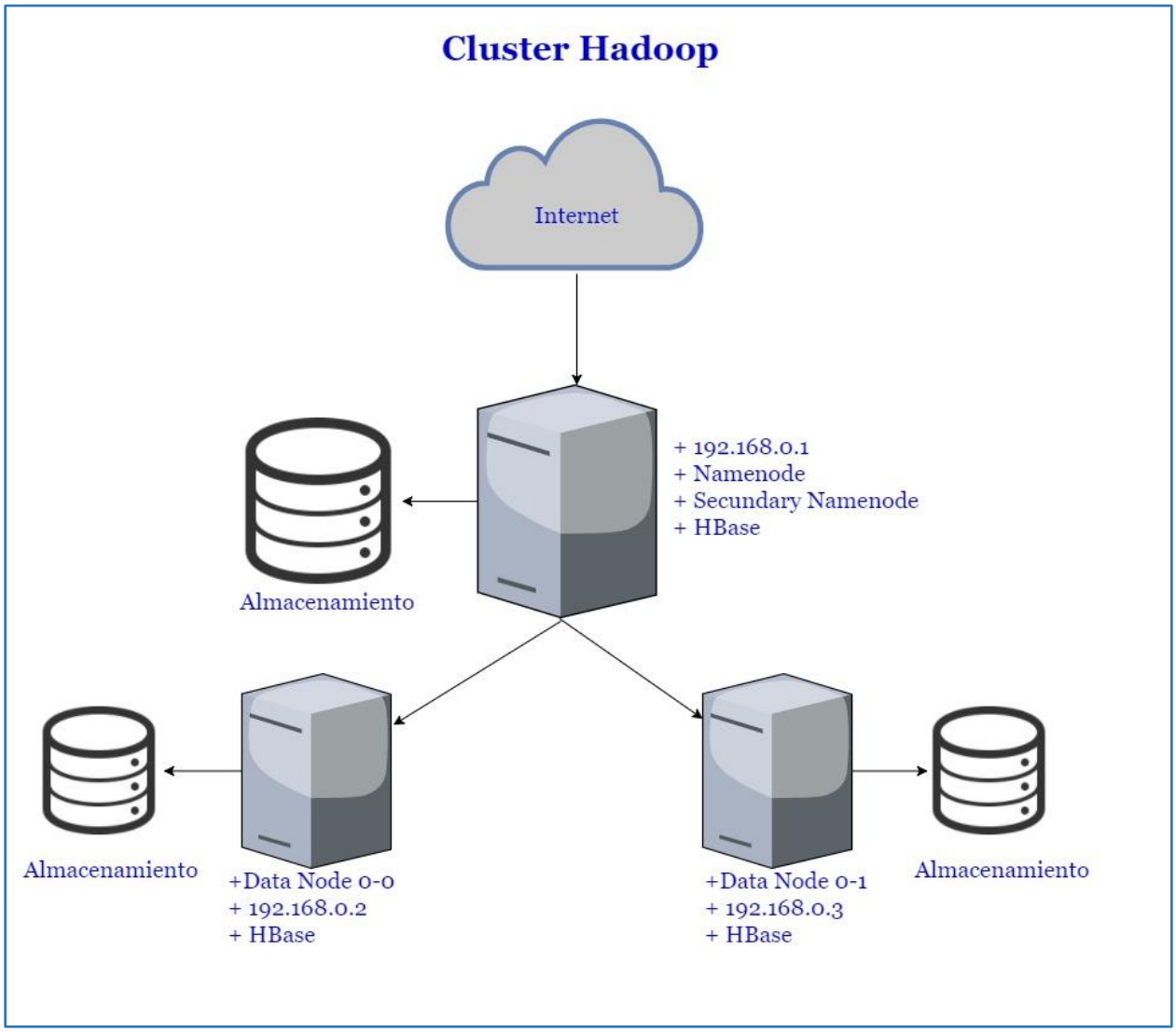

Figura 10 Clúster Hadoop (Fuente: Galeano Cruz y Domínguez Rivera, 2017)

Las maquinas estarán conectadas por una red interna con las direcciones 192.168.0.1 para el NameNode y 192.168.0.2-3 para los DataNodes, además el NameNode tendrá una segunda conexión de red a internet, de este modo el clúster tendrá acceso a internet para descargar los recursos necesarios además de poder comunicarse con las maquinas clientes. El NameNode y los DataNodes hacen parte de la arquitectura básica de Hadoop, además de esto se utilizará YARN como administrador de las máquinas, y HBase como base de datos NoSQL para almacenar los datos. Cada máquina del clúster tendrá su propio repositorio de archivos esto con el fin de que no se deba pelear por el acceso a los recursos ni los repositorios de información, la arquitectura Hadoop proporciona una forma de almacenar los datos de tal manera que quedan distribuidos en todo el clúster, Hadoop parte un archivo en varios archivos de hasta 64 MB y los distribuye en el clúster, como se muestra en la *figura 11*, además genera copias de seguridad de los archivos distribuidas por los nodos a modo de backup.

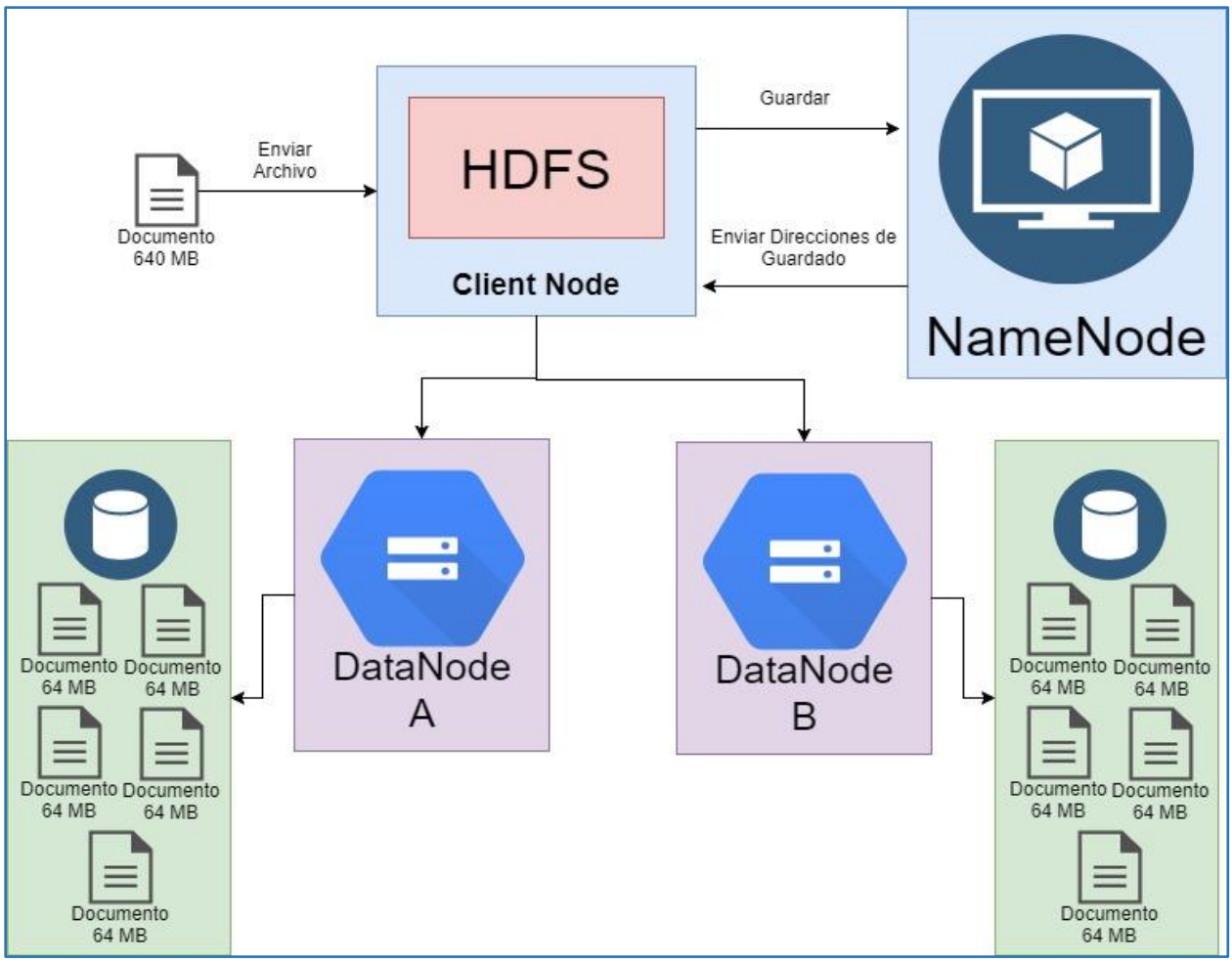

Figura 11 Almacenamiento de Datos en HDFS (Fuente: Galeano Cruz y Domínguez Rivera, 2017)

En resumen, el funcionamiento del prototipo se basa en que los estudiantes construyan los programas y los envíen al clúster para que este realice el análisis de los datos y cree el archivo de visualización como se ilustra en la *figura 12*.

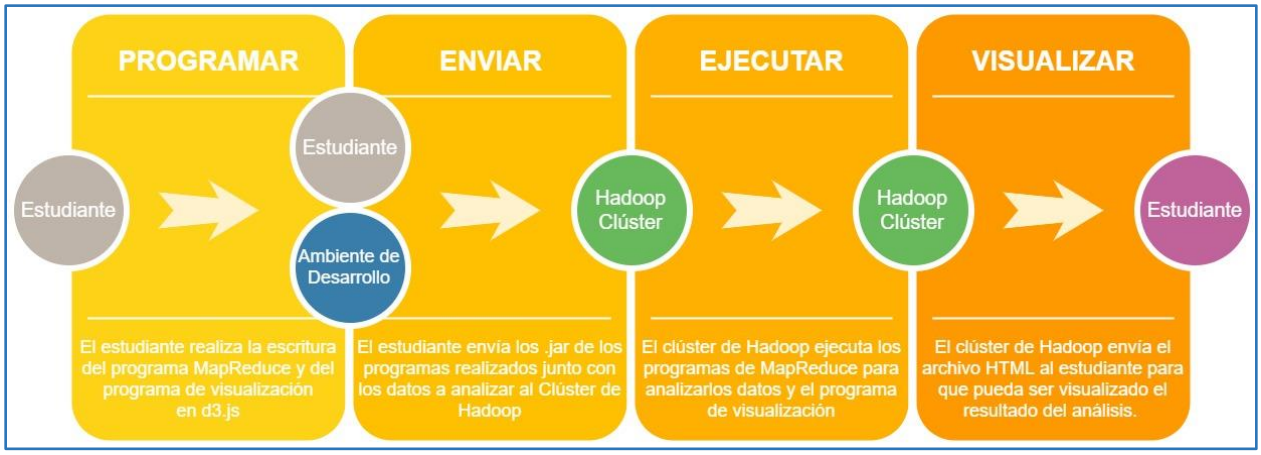

Figura 12 Funcionamiento de Prototipo de Laboratorio (Fuente: Galeano Cruz y Domínguez Rivera, 2017)

El diseño completo del prototipo se muestra en la *figura 13*, en donde se pueden ver el ambiente de desarrollo del cliente y el clúster de Hadoop.

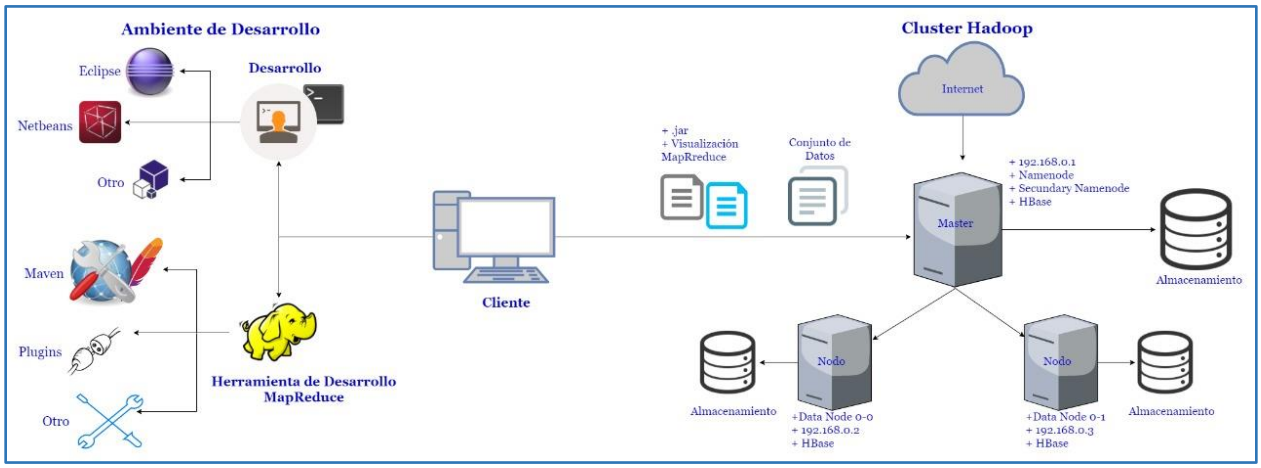

Figura 13 Diseño Completo Prototipo Laboratorio Big Data (Fuente: Galeano Cruz y Domínguez Rivera, 2017)

4.3 Fase de instalación y montaje

4.3.1 Instalación del Clúster Hadoop

Para instalar Hadoop hay que realizar los pasos ilustrados en la *figura 14*:

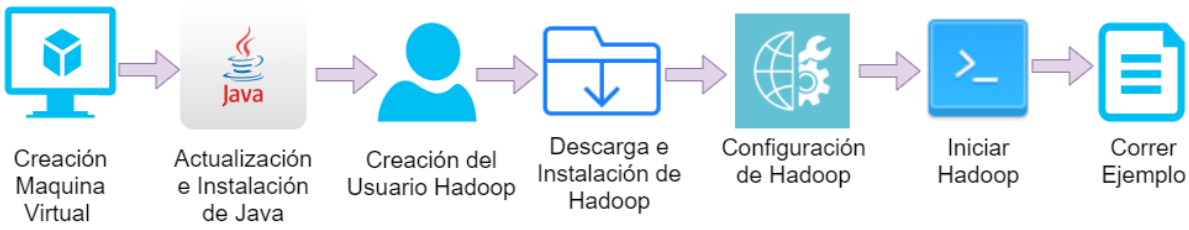

Figura 14 Pasos para instalar el clúster Hadoop (Fuente: Galeano Cruz y Domínguez Rivera, 2017)

La instalación del Clúster Hadoop se realizó en máquinas virtuales utilizando VirtualBox, se realiza la creación de 4 máquinas virtuales, una para el maestro, 2 para los esclavos y una para el cliente, el proceso de instalación para el maestro fue el siguiente:

Se creó la máquina virtual como Linux Red Hat de 64 bits, se le asignaron 4 GB de memoria RAM y se creó un disco duro virtual con 50 GB para el almacenamiento de la información, para las configuraciones de red se habilitaron 2 tarjetas de red, la primera es una red interna llamada HadoopLAN, esta red interna es la que se utilizara para conectar los equipos del clúster, por lo cual todas las maquinas del clúster deben tener configurada esta red. La otra es un adaptador de puente, el cual sirve para 2 cosas, primero permite que la máquina virtual esté conectada a

internet, segundo sirve para que la máquina virtual tenga una conexión a internet directa, es decir, la máquina virtual puede ser accedida desde afuera del sistema de virtualización.

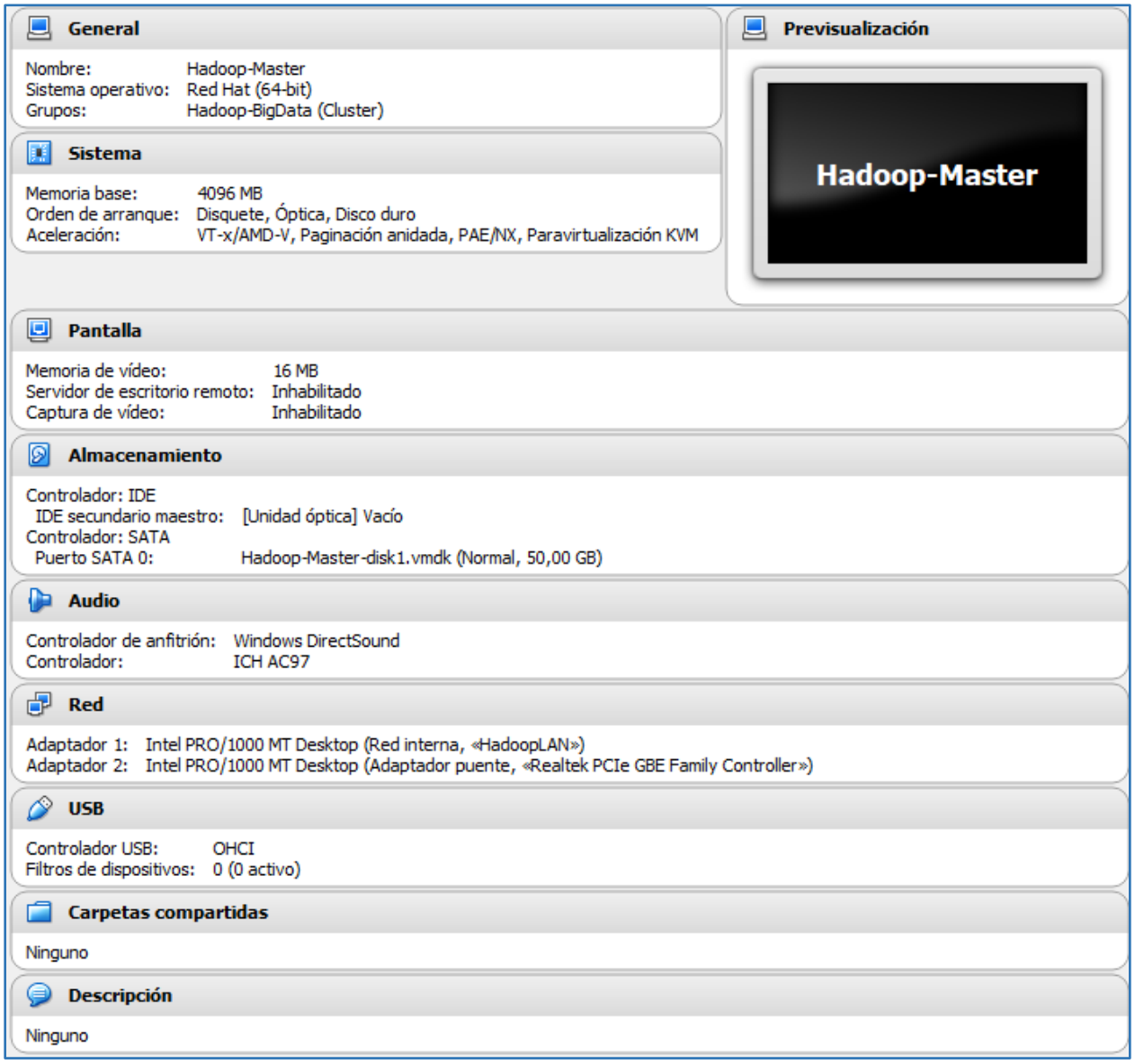

Figura 15 Configuración de máquina virtual Hadoop-Master (Fuente: Galeano Cruz y Domínguez Rivera, 2017)

Una vez creada la máquina virtual procedemos a instalar el sistema operativo, en este caso escogimos CentOS 7 como sistema operativo para todas las maquinas del clúster. En el proceso de instalación se realizó la configuración del SO, se escogió idioma español (Colombia), se configuro la zona horaria y se seleccionó en el entorno base del software a instalar un escritorio Gnome y como complementos seleccionamos herramientas de desarrollo, la razón por la cual se escogió escritorio Gnome es porque hay cosas que resulta más sencillo realizarlas en un entorno gráfico, mas no es necesario instalar un entorno gráfico, con la instalación minia (selección de software predeterminada donde se instala solo el SO sin entorno gráfico, solo por

consola). Como destino de la instalación se escogió el disco duro virtual creado junto con la máquina virtual.

Para la configuración de la red interna se estableció una IP fija, la cual se definió como 192.168.0.1 como se muestra en el diseño y se deshabilito las IPv6 ya que Hadoop no es compatible con este tipo de IP.

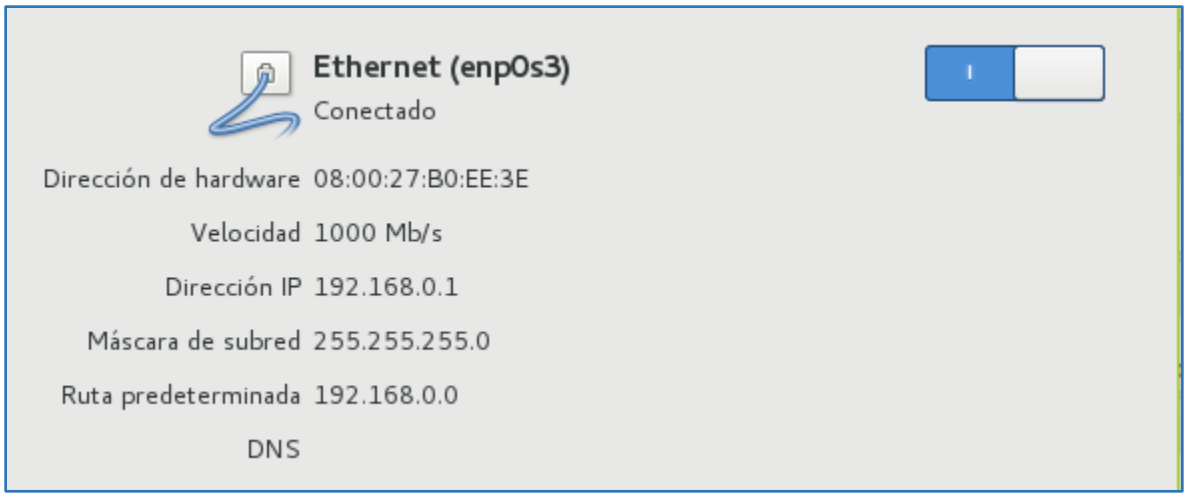

Figura 16 Configuración de red interna MV Hadoop-Master (Fuente: Galeano Cruz y Domínguez Rivera, 2017)

Para la configuración del adaptador de puente se inactivo las IPv6 y se dejó la configuración de DHCP para la IPv4.

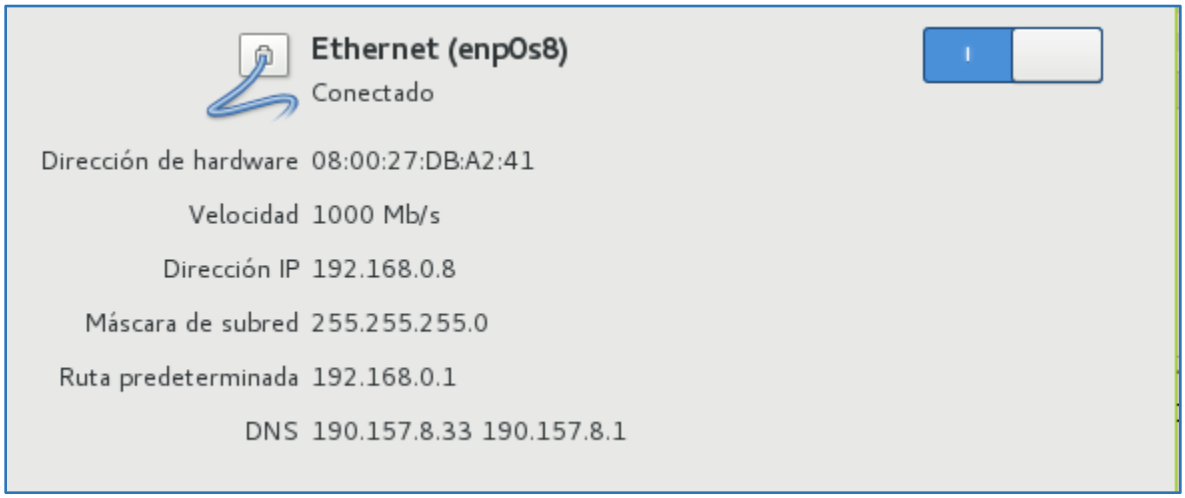

Figura 17 Configuración de adaptador puente MV Hadoop-Master (Fuente: Galeano Cruz y Domínguez Rivera, 2017)

Por último, se configuraron los usuarios de la máquina virtual, primero definimos una contraseña para el usuario root y luego creamos un usuario normal por el cual vamos a acceder, este usuario lo llamamos BigData, pero se puede nombrar de cualquier forma.

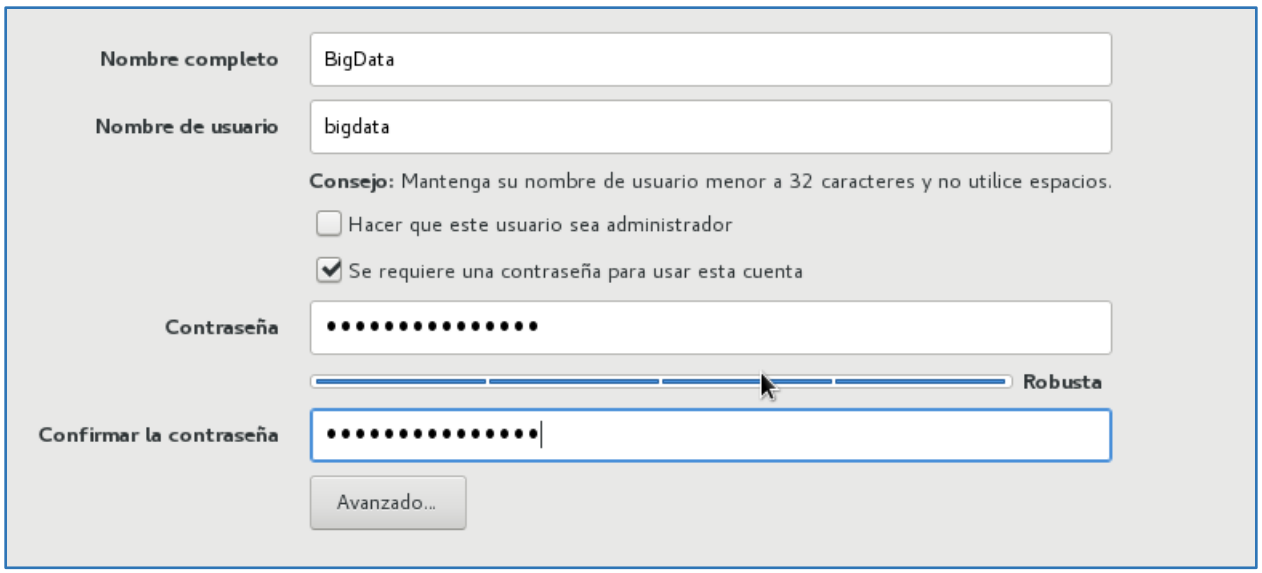

Figura 18 Configuración usuario principal MV Hadoop-Master (Fuente: Galeano Cruz y Domínguez Rivera, 2017)

El proceso de instalación para los esclavos fue el siguiente (para los 2 esclavos se siguió el mismo proceso):

Se crearon dos máquinas virtuales como Linux Red Hat de 64 bits, se le asignaron 2 GB de memoria RAM y se creó un disco duro virtual con 50 GB (por cada una) para el almacenamiento de la información, para las configuraciones de red se habilito una tarjeta de red como red interna y se le configuro la red llamada HadoopLAN (la misma que la definida para el maestro), esta red interna es la que se utilizara para conectar los equipos del clúster, por lo cual todas las maquinas del clúster deben tener configurada esta red.

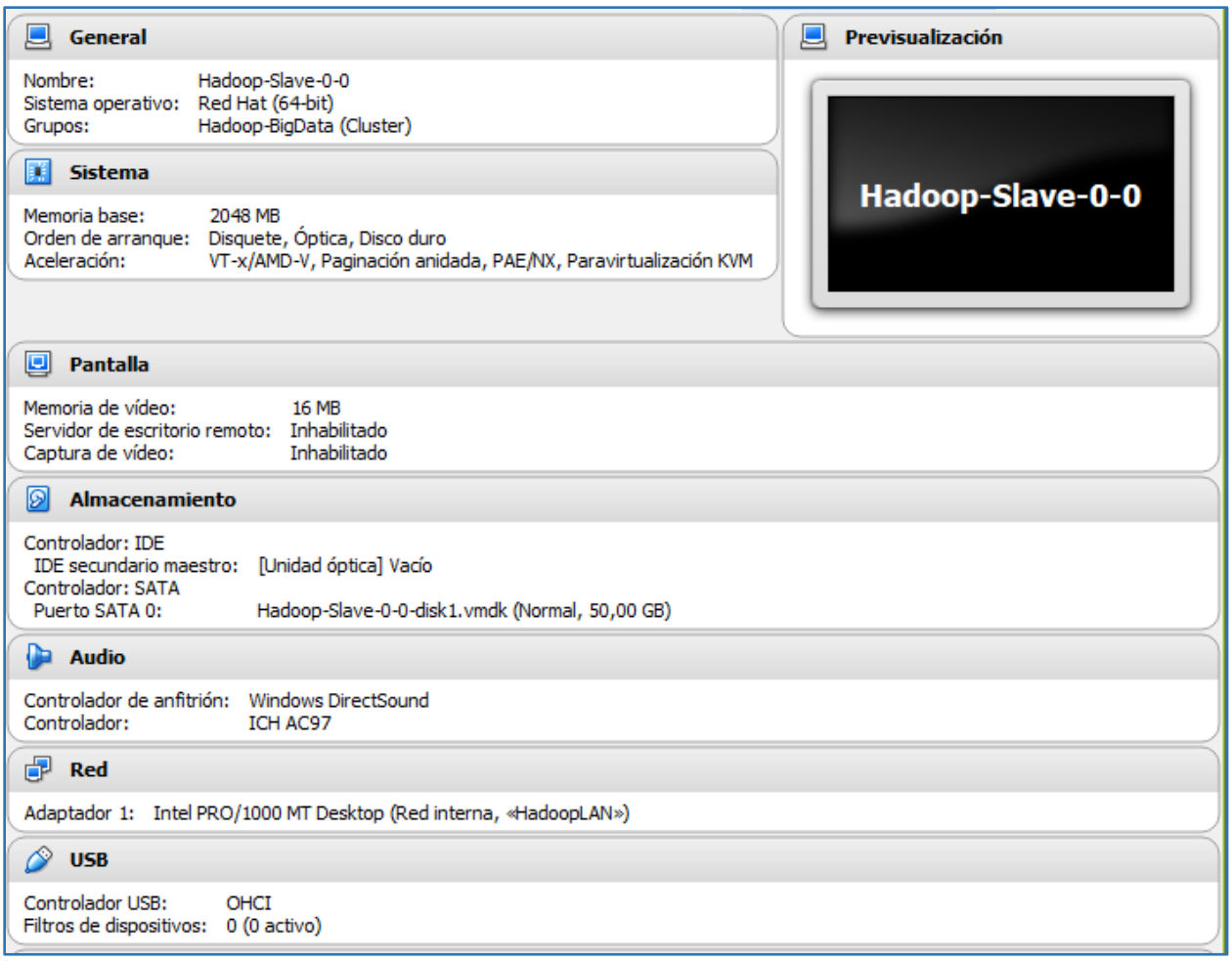

Figura 19 Configuración de máquina virtual Hadoop-Slave-0-0 y 0-1 (Fuente: Galeano Cruz y Domínguez Rivera, 2017)

Una vez creadas las máquinas virtuales procedemos a instalar el sistema operativo, como se aclaró anterior mente se escogió CentOS 7 como sistema operativo para todas las maquinas del clúster. En el proceso de instalación se realizó la configuración del SO, se escogió idioma español (Colombia), se configuro la zona horaria y se seleccionó en el entorno base del software a instalar un escritorio Gnome y como complementos seleccionamos herramientas de desarrollo, la razón por la cual se escogió escritorio Gnome es porque hay cosas que resulta más sencillo realizarlas en un entorno gráfico, mas no es necesario instalar un entorno gráfico, con la instalación minia (selección de software predeterminada donde se instala solo el SO sin entorno gráfico, solo por consola). Como destino de la instalación se escogió el disco duro virtual creado junto con la máquina virtual.

Para la configuración de la red interna se estableció una IP fija, la cual se definió como 192.168.0.2 para el esclavo 0-0 y 192.168.0.3 para el esclavo 0-1 como se muestra en el diseño y se deshabilito las IPv6.

Figura 20 Configuración de red interna MV Hadoop-Slave-0-0 (Fuente: Galeano Cruz y Domínguez Rivera, 2017)

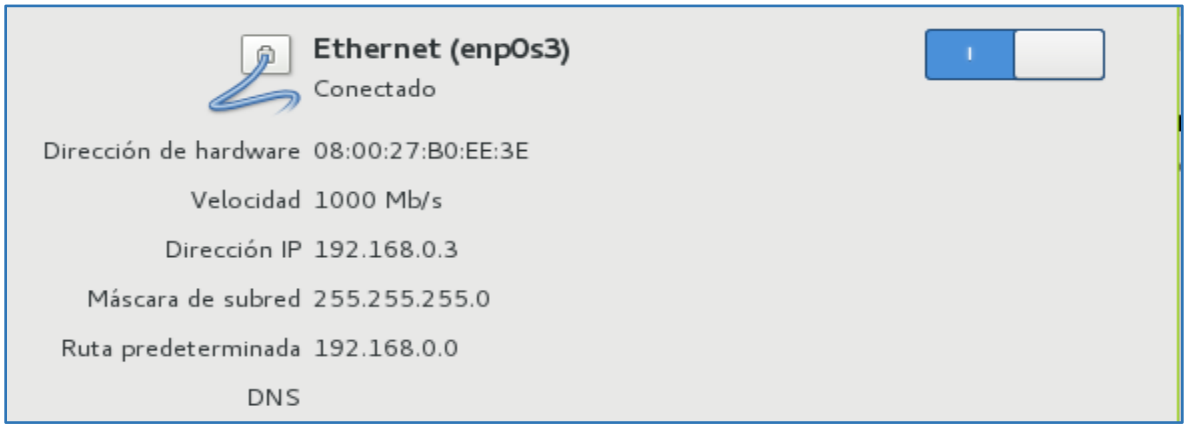

Figura 21 Configuración de red interna MV Hadoop-Slave-0-1 (Fuente: Galeano Cruz y Domínguez Rivera, 2017)

Por último, se configuraron los usuarios de las máquinas virtuales, primero definimos una contraseña para el usuario root y luego creamos un usuario normal por el cual vamos a acceder, este usuario lo llamamos BigData, pero se puede nombrar de cualquier forma (se utilizó la misma configuración de usuarios que en el maestro).

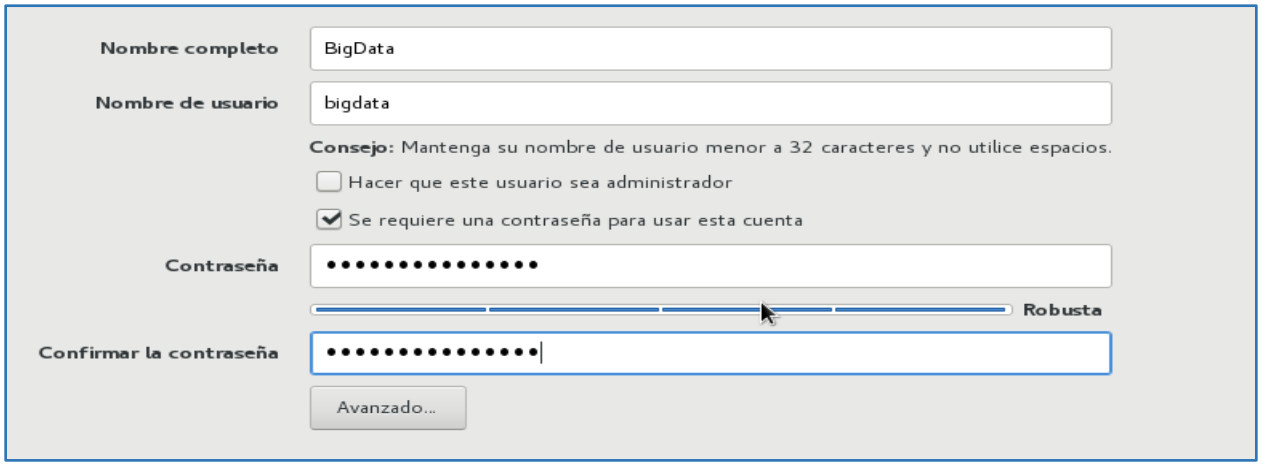

Figura 22 Configuración usuario principal MV Hadoop-Slave-0-0 y 0-1 (Fuente: Galeano Cruz y Domínguez Rivera, 2017)

En este punto se sacaron copias de seguridad de las maquinas como backup, para que en caso de que se dañe una de las maquinas se pueda sacar otra de respaldo.

Una vez creadas las máquinas virtuales se procede a realizar la instalación de Hadoop, para esto se inician las 3 máquinas virtuales.

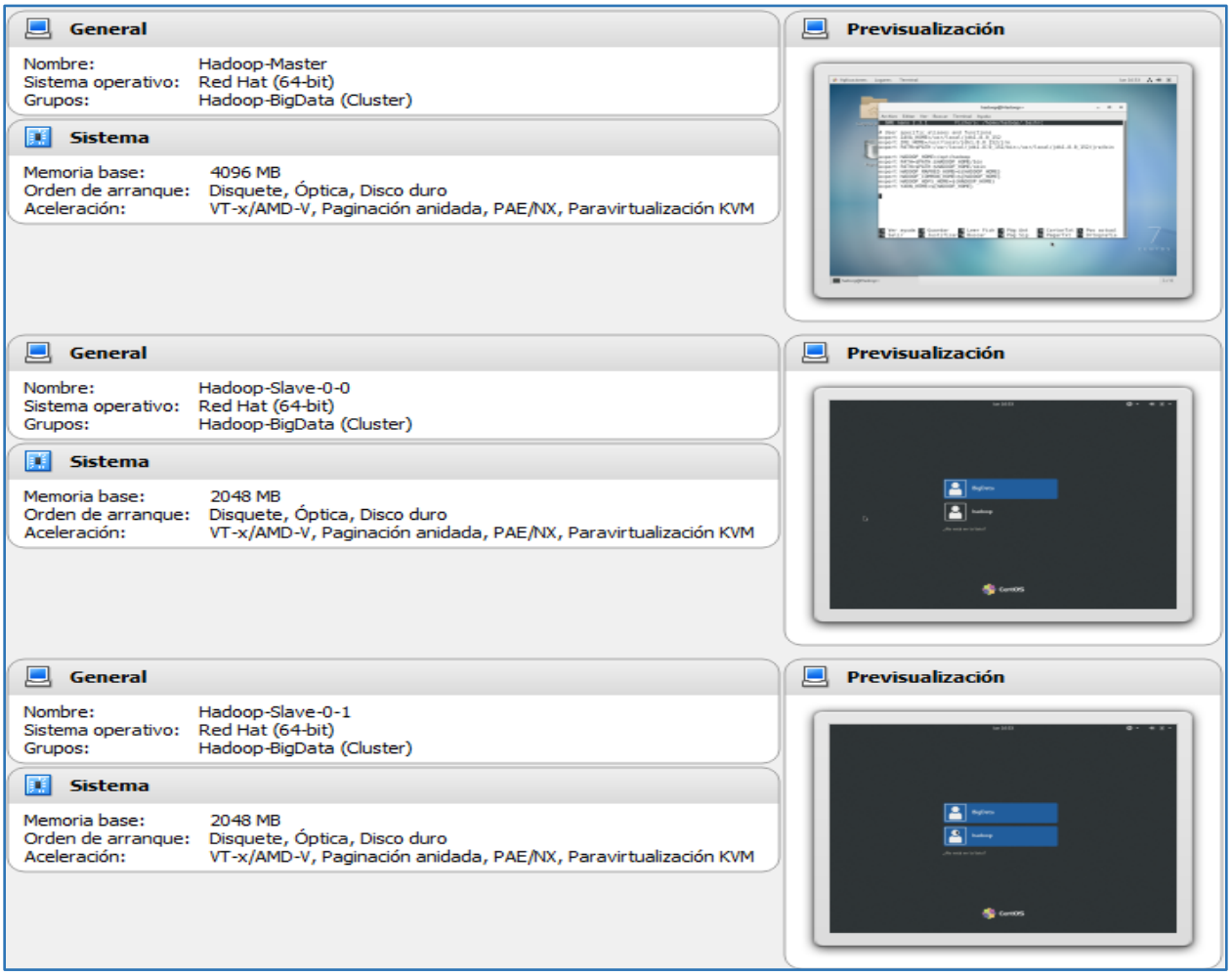

Figura 23 Resultado de instalación de las máquinas virtuales (Fuente: Galeano Cruz y Domínguez Rivera, 2017)

Lo primero es instalar y/o actualizar Java en todas las máquinas virtuales, la versión de Java se instalo fue la 1.8.0_152.

```
java version "1.8.0_152"
Java(TM) SE Runtime Environment (build 1.8.0_152-b16)
Java HotSpot(TM) 64-Bit Server VM (build 25.152-b16, mixed mode)
```

Figura 24 Resultado de instalación y/o actualización de Java (Fuente: Galeano Cruz y Domínguez Rivera, 2017)

Después de instalar y/o actualizar Java se procedió a realizar la creación del usuario donde se alojarán todos los servicios de Hadoop (siempre es una buena práctica crear un usuario aparte

para el manejo de Hadoop), este usuario se debe crear en todas las máquinas. El usuario se llama hadoop (se puede utilizar cualquier nombre) y se accedió con este usuario en todas las máquinas.

```
[bigdata@Hadoop ~]$ su - hadoop
Contraseña:
Último inicio de sesión:dom nov 26 20:57:32 -05 2017en pts/0
```

Figura 25 Resultado de creación de usuario hadoop (Fuente: Galeano Cruz y Domínguez Rivera, 2017)

Se modifica el archivo de hosts en donde se ponen los nombres de las maquinas del clúster y sus direcciones IP, esto se realiza para todas las maquinas del clúster, para le nombre se usa la estructura Hadoop-(nombre de la maquina).

```
192.168.1.1 hadoop-master
192.168.1.2 hadoop-slave-0-0
192.168.1.3 hadoop-slave-0-1
```

Figura 26 Resultado de modificación de archivo hosts (Fuente: Galeano Cruz y Domínguez Rivera, 2017)

Una vez creado el usuario Hadoop se procede a realizar la configuración del sistema ssh para comunicarse entre las máquinas virtuales, se define un archivo de clave y se traslada a las claves autorizadas, después se pasan las claves a todas las maquinas del clúster por último se procede a dar los permisos necesarios a el usuario hadoop sobre el archivo de claves autorizadas, esta configuración se realiza en todas las máquinas.

```
[hadoop@Hadoop ~]$ ssh hadoop-master
Enter passphrase for key '/home/hadoop/.ssh/id_rsa':
Last login: Mon Nov 27 16:12:32 2017
[hadoop@Hadoop ~]$ ssh hadoop-slave-0-0
Enter passphrase for key '/home/hadoop/.ssh/id_rsa':
Last login: Sun Nov 26 16:37:31 2017
[hadoop@Hadoop ~]$ ssh hadoop-slave-0-1
Enter passphrase for key '/home/hadoop/.ssh/id_rsa':
Last login: Sun Nov 26 16:38:20 2017
[hadoop@Hadoop ~]$
```

Figura 27 Resultado de configuración de servicio ssh y prueba de conexión (Fuente: Galeano Cruz y Domínguez Rivera, 2017)

Una vez se comprueba que las maquinas puedan conectarse entre sí se procede a instalar Hadoop, para esto primero se crea el directorio Hadoop en el sitio donde quedara alojado el sistema en todas las máquinas, en nuestro caso la carpeta opt, luego descargamos la versión de Hadoop a instalar, esto se realiza desde la página de Apache, en este caso de descargo la versión 2.7.0, la instalación de Hadoop como tal se realiza únicamente en el maestro y luego se replican

los archivos en todos los esclavos, con esto se evita tener que configurar Hadoop una vez por cada máquina.

Una vez descargado Hadoop 2.7.0 en el maestro procedemos a descomprimir el archivo tar.gz descargado, una vez se descomprime se cambia el nombre de la carpeta con el comando mv para que quede hadoop y movemos la carpeta hadoop al directorio que acabamos de crear (/opt/hadoop).

Después procedemos a configurar las variables de entorno que requiere Hadoop para funcionar correctamente, estas variables permiten que otros programas o incluso el mismo Hadoop encuentre las carpetas donde se guardan archivos importantes para el funcionamiento y/o ejecución de tareas de Hadoop.

Figura 28 Resultado de configuración variables de entorno (Fuente: Galeano Cruz y Domínguez Rivera, 2017)

Con las variables de entorno ya configuradas ahora si procedemos a configurar Hadoop como tal, para esta tarea se deben modificar los archivos core-site.xml, hdfs-site.xml, mapred-site.xml, yarn-site.xml y hadoop-env.sh, estos son los archivos que le indican a Hadoop todas las propiedades de su funcionamiento y en esta versión de Hadoop se encuentran en el directorio /etc/hadoop/, en el directorio conf como en versiones anteriores.

Primero configuramos, en el archivo core-site.xml, el nombre y puerto por defecto de la aplicación Hadoop, así como el directorio en el cual se guardarán los archivos temporales (este directorio es importante tenerlo en cuenta al momento de que ejecutar Hadoop, ya que muchos de los errores de ejecución pueden ser producidor por archivos temporales que haya que borrar).

```
<property>
    <name>fs.default.name</name>
    <value>hdfs://hadoop-master:9000</value>
</property>
<property>
    <name>hadoop.tmp.dir</name>
    <value>/opt/hadoop/tmp</value>
</property>
```

Figura 29 Configuración archivo core-site.xml (Fuente: Galeano Cruz y Domínguez Rivera, 2017)

Segundo configuramos, en el archivo hdfs-site.xml, la ubicación donde se guardarán los archivos de HDFS, así como el número de réplicas a crear de los archivos guardados en HDFS.

```
<property>
    <name>dfs.namenode.name.dir</name>
    <value>file:/opt/hadoop/workspace/dfs/name</value>
</property>
<property>
    <name>dfs.datanode.data.dir</name>
    <value>file:/opt/hadoop/workspace/dfs/data</value>
</property>
<property>
    <name>dfs.replication</name>
    <value>2</value>
</property>
```

Figura 30 Configuración archivo hdfs-site.xml (Fuente: Galeano Cruz y Domínguez Rivera, 2017)

Tercero configuramos el archivo mapred-site.xml, este archivo no se encuentra creado como tal, en su lugar está el archivo mapred-site.xml.template, se debe coger este archivo, copiarlo y cambiarle el nombre a mapred-site.xml y en este se configuran el Job Tracker, la dirección del directorio local para uso de MapReduce, la cantidad de map task y de reduce task, las map task se calculan multiplicando el número de nodos existentes en el clúster por 10 y reduce task se calcula tomando el número de esclavos multiplicado por el número reduce slots por esclavo multiplicado por 0.99, si los esclavos son pocos se debe utilizar el número de esclavos menos 1.

```
<property>
    <name>mapred.job.tracker</name>
    <value>hadoop-master:9001</value>
</property>
<property>
    <name>mapred.local.dir</name>
    <value>/opt/hadoop/mapred-local</value>
</property>
<property>
    <name>mapred.map.tasks</name>
    <value>20</value>
</property>
<property>
    <name>mapred.reduce.tasks</name>
    <value>10</value>
</property>
```

Figura 31 Configuración archivo mapred-site.xml (Fuente: Galeano Cruz y Domínguez Rivera, 2017)

Cuarto configuramos, en el archivo yarn-site.xml, los puertos que serán utilizados por los resource manager.

```
<property>
    <name>yarn.resourcemanager.resource-tracker.address</name>
    <value>hadoop-master:8025</value>
</property>
<property>
    <name>yarn.resourcemanager.scheduler.address</name>
    <value>hadoop-master:8035</value>
</property>
<property>
    <name>yarn.resourcemanager.address</name>
    <value>hadoop-master:8050</value>
</property>
```

Figura 32 Configuración archivo yarn-site.xml (Fuente: Galeano Cruz y Domínguez Rivera, 2017)

Quinto configuramos, en el archivo hadoop-env.sh, la implementación de Java a utilizar, esto se hace definiendo la ruta donde se encuentra instalado el JDK de Java.

```
# The java implementation to use.
export JAVA_HOME=/usr/local/jdk1.8.0_152
```

Figura 33 Configuración archivo hadoop-env.sh (Fuente: Galeano Cruz y Domínguez Rivera, 2017)

Una vez configurados los archivos de Hadoop en el maestro, procedemos a replicarlos en los esclavos, para esto se utiliza el comando scp, una vez copiados los archivos de Hadoop en todas las maquinas procedemos hacer una última configuración en el maestro, la cual es crear los archivos de masters y slaves.

En el archivo de masters van los nombres de las máquinas que van a ser maestros.

Figura 34 Configuración archivo masters MV Hadoop-Master (Fuente: Galeano Cruz y Domínguez Rivera, 2017)

En el archivo slaves van los nombres de las máquinas que serán los esclavos.

Figura 35 Configuración archivo slaves MV Hadoop-Master (Fuente: Galeano Cruz y Domínguez Rivera, 2017)

Con esto se finaliza la configuración de archivos de Hadoop, ahora podemos proceder a inicializar el servicio de Hadoop.

Antes de inicializar el servicio de Hadoop debemos formatear el sistema de archivos HDFS, esto se realiza únicamente antes de la primera inicialización ya que al formatear se eliminar todos los archivos que se encuentren en HDFS, si no se formatea HDFS Hadoop arrojara errores al momento de utilizar el HDFS ya que, en el proceso de formateo, Hadoop crea el archivo VERSION que tiene una clave aleatoria única la cual se genera cada que se formatea el sistema de archivos y esta clave la utilizan los nodos para poder comunicarse en el sistema HDFS.

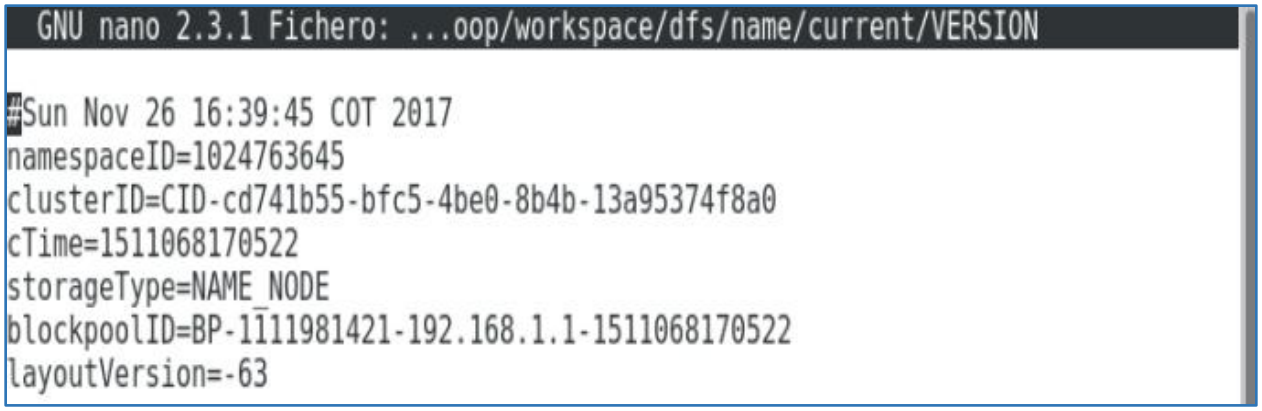

Figura 36 Archivo VERSION MV Hadoop-Master (Fuente: Galeano Cruz y Domínguez Rivera, 2017)

Una vez formateado el HDFS podemos proceder a inicializar los nodos con los comandos start-dfs.sh y start-yarn.sh, los comandos se deben ejecutar en ese orden, ya que primero se inicializa el HDFS y luego el YARN, si no realiza de esta forma podría generar errores.

Cuando se ejecuten los comandos de inicio de Hadoop se pueden encontrar los procesos de NameNode, SecondaryNameNode y ResourceManager en el maestro, se verifica esto con el comando jps.

Figura 37 Resultado de inicializar Hadoop MV Hadoop-Master (Fuente: Galeano Cruz y Domínguez Rivera, 2017)

En los nodos esclavos se deben visualizar únicamente los procesos DataNode y NodeManager.

Figura 38 Resultado de inicializar Hadoop MV Hadoop-Slave-0-0 y 0-1 (Fuente: Galeano Cruz y Domínguez Rivera, 2017)

Con esto finaliza la instalación de Hadoop como tal, una vez instalado se puede trabajar con el HDFS utilizando el comando hadoop fs -acción a realizar. HDFS nos proporciona un amplio rango de acciones, muy parecidos al manejo de archivos que se les da a los directorios de Linux, como lo son el comando mkdir para crear directorios, el comando rm para remover directorios, el comando ls para listar el contenido de un directorio, etc.

Por último, para comprobar que todo haya salido bien se puede entrar al navegador e ingresar a las direcciones locales de los manejadores de Hadoop (hadoop-master:8088) y HDFS(hadoop-master:50070), estos sitios son locales por lo que hadoop-master es el equivalente a localhost.

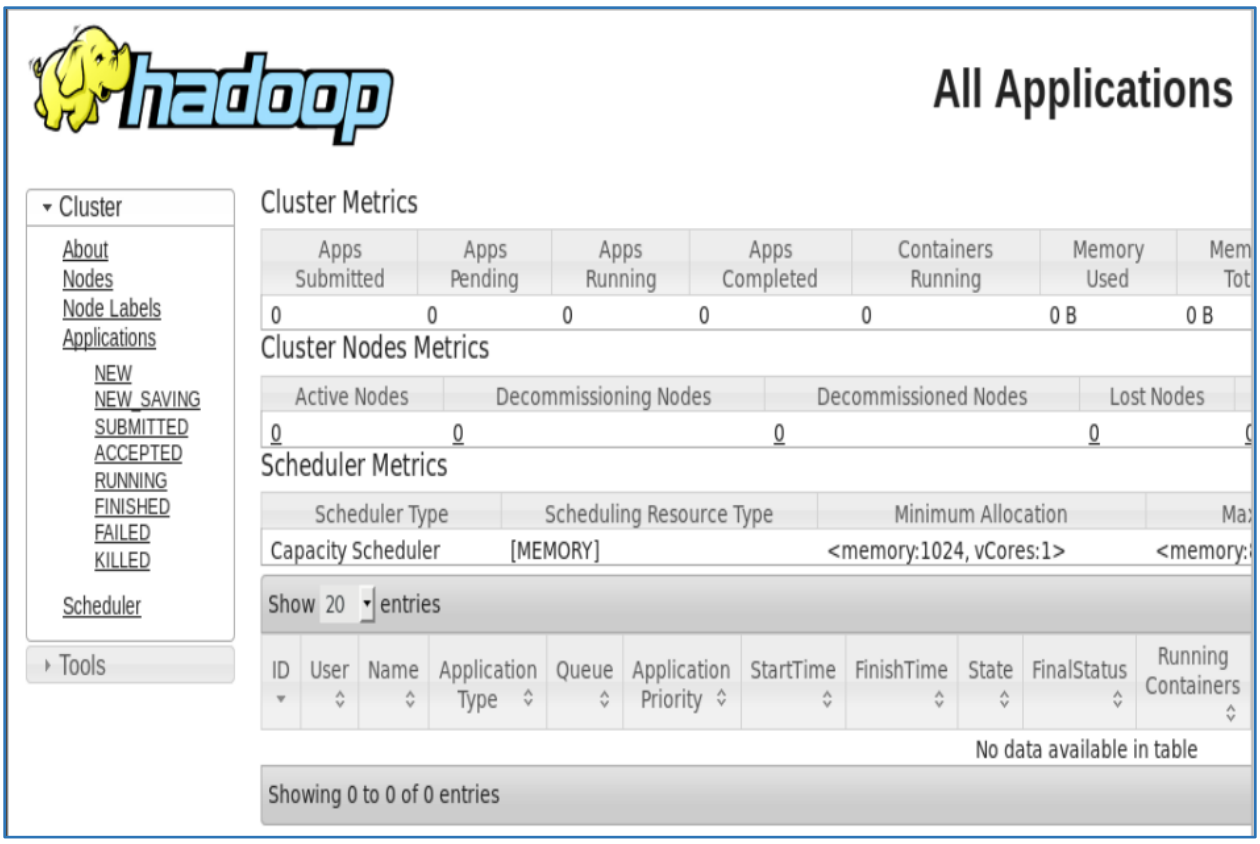

Figura 39 Resultado instalación de Hadoop ingresando por hadoop-master:8088 (Fuente: Galeano Cruz y Domínguez Rivera, 2017)

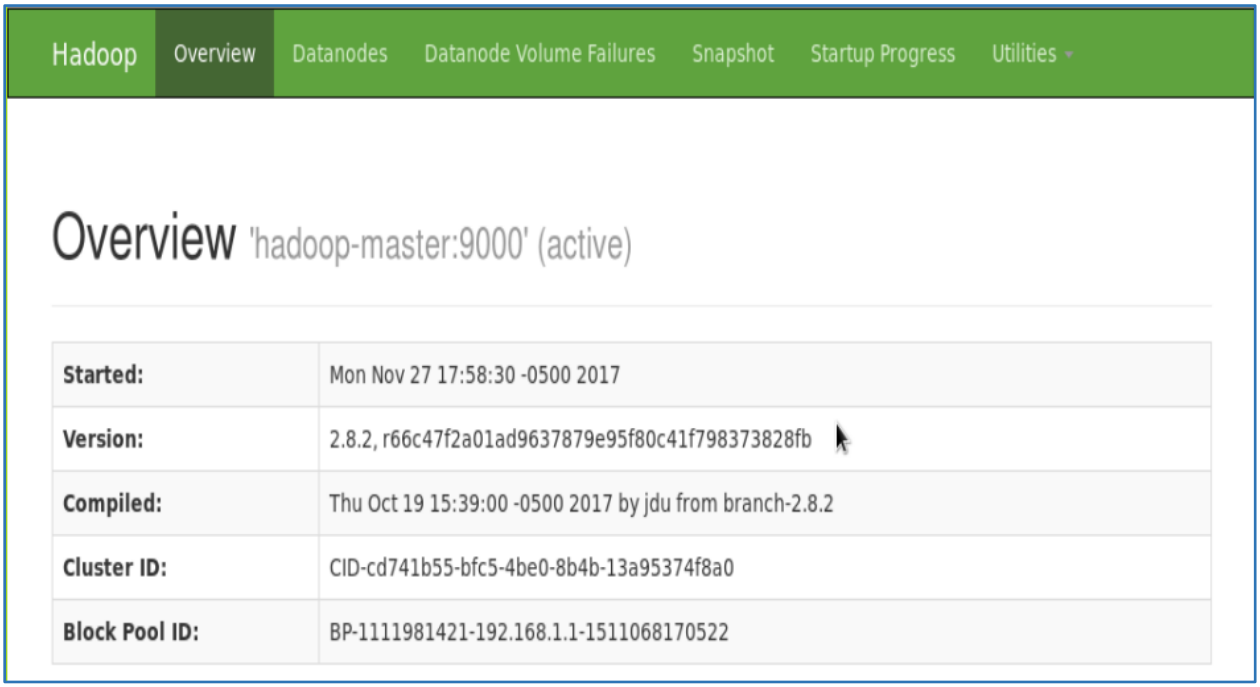

Figura 40 Resultado instalación de Hadoop ingresando por hadoop-master:8088 (Fuente: Galeano Cruz y Domínguez Rivera, 2017)

Durante la instalación de Hadoop salieron varios errores que hicieron que el proceso fuera más lento, pero que en definitiva ayudaron a comprender mucho más a fondo como funciona, en este momento compartiremos un pequeño resumen de los errores que salieron y como ayudaron a comprender mejor el funcionamiento de Hadoop.

Al principio al iniciar Hadoop no iniciaba todos los servicios en el maestro, esto se debía a que los archivos de configuración core-site.xml y yarn-site.xml no se encontraban bien configurados, cuando no inicien todos los procesos se debe revisar el log de Hadoop, el cual se encuentra en el directorio /logs del directorio raíz de Hadoop, todos los errores y problemas que tenga Hadoop en su funcionamiento se verán reflejados en dichos logs como WARN o ERROR, ojo, no todos los WARN son errores, algunos son solo advertencias, pero siempre es bueno revisarlos por si acaso.

Otro de los grandes problemas fue que en uno de los esclavos no iniciaban todos los procesos, pero en el otro si, esto fue muy extraño y se tardó mucho tiempo buscando una solución, al final la solución fue borrar todo el contenido de Hadoop y volver a replicarlo desde el maestro (borrando los archivos masters y slaves), esto dejo como enseñanza que muchas veces errores de este tipo es mejor solucionarlos replicando los archivos o simplemente formateando el nodo y volviéndolo a configurar desde cero, muchos errores son muy complicados de encontrar como solucionarlos y lo más probable es que se produzcan debido a errores cometidos durante la instalación, pero si después de volver a reinstalar todo te sigue saliendo el error, es mejor que se revise si es un problema con la versión de JAVA o de pronto es un problema directamente con el sistema operativo.

El ultimo error y el que más tiempo costo encontrar fue tal vez uno de los más fáciles de solucionar, al momento de correr los ejemplos de MapReduce que trae Hadoop, salía un error de replicación, el cual decía que no había nodos para replicar, pero los procesos estaban ejecutándose en todos los nodos, se revisó el log del maestro y no se encontró nada, pero en el log de los esclavos se encontró un error que decía que no había comunicación con el host, esto se puede producir por varias razones, configuraciones mal hechas en los archivos masters o slaves, archivo hosts mal configurado, etc. al final en nuestro caso el error se producía por 2 motivos, la primera, se formateo más de una vez el HDFS, esto produjo que se creara un nuevo archivo VERSION en HDFS que, por obvias razones, contenía una identificación diferente a la de los nodos esclavos, la solución es, siempre que se vaya a formatear el HDFS se debe borrar el contenido del directorio donde se guarden los archivos de HDFS del maestro y los esclavos, este se define en el archivo hdfs-sites.xml, después de borrar el contenido se debe volver a formatear el HDFS. La segunda causa fue el firewall que estaba bloqueando la comunicación entre los esclavos y el maestro, nosotros lo solucionamos desactivando el firewall (ya que en nuestro caso la seguridad no es lo primordial), pero lo correcto sería crear excepciones para que los procesos de Hadoop pudieran pasar a través del firewall.

4.3.2 Instalación HBase

Se instaló HBase como manejador de base de datos para HDFS, HBase es un base de datos NoSQL orientada a columnas, para la instalación de HBase se requiere que se encuentre Hadoop instalado en el clúster, también se requiere tener instalado Java el cual ya debería estar instalado (para el correcto funcionamiento de Hadoop se instaló Java), pero primero se debe revisar que versión de Java es compatible con la versión de HBase que se va a utilizar.

Lo primero es descargar HBase del repositorio de Apache, en nuestro caso descargaremos la versión 1.2.6, una vez se descargue HBase se procede a descomprimir el archivo tar.gz, a la carpeta resultante le cambiamos el nombre con el comando mv y la movemos al directorio de Hadoop /opt/hadoop.

El primer paso para instalar HBase es la configuración del archivo regionservers, el cual se encuentra en la carpeta conf de HBase, en este archivo se ponen los nombres de los regionservers, los cuales hacen las funciones de esclavos, en nuestro caso son hadoop-slave-0-0 y hadoop-slave-0-1.

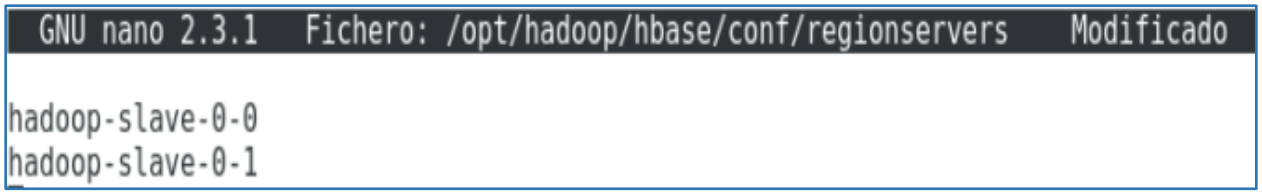

Figura 41 Configuración archivo regionservers MV Hadoop-Master (Fuente: Galeano Cruz y Domínguez Rivera, 2017)

El segundo paso es crear un archivo nuevo en el directorio conf, el cual se llamará backup-masters, en este archivo se configurarán los nodos que servirán como backup para el master, es decir, si el master, por algún motivo, llega a caerse, perder conexión, apagarse, etc. el nodo que este registrado como backup automáticamente toma su lugar para que el clúster siga funcionando, en nuestro caso escogeremos el nodo hadoop-slave-0-0 como backup del master.

Figura 42 Configuración archivo backup-masters MV Hadoop-Master (Fuente: Galeano Cruz y Domínguez Rivera, 2017)

El tercer paso es la configuración del archivo hbase-site.xml, en el cual configuraremos los nodos en los cuales correrá ZooKeeper, en este caso agregaremos todos los nodos ya que ZooKeeper debe correr en todos los nodos, también configuramos el directorio root para hbase,

el cual es una dirección en HDFS, en este caso es hadoop-master y definimos el puerto como el 9000 y definimos el directorio HBase, por último, se configura el clúster en modo distribuido.

```
<property>
        <name>hbase.rootdir</name>
        <value>hdfs://hadoop-master:9000/hbase</value>
</property>
<property>
        <name>hbase.cluster.distributed</name>
        <value>true</value>
</property>
<property>
        <name>hbase.zookeeper.quorum</name>
        <value>hadoop-master,hadoop-slave-0-0,hadoop-slave-0-1</value>
</property>
```

Figura 43 Configuración archivo hbase-site.xml MV Hadoop-Master (Fuente: Galeano Cruz y Domínguez Rivera, 2017)

El cuarto paso es configurar el archivo hbase-env.xml, para que los pids no se guarden en el directorio /tmp, el cual es el directorio por defecto.

```
# The directory where pid files are stored. /tmp by default.
export HBASE_PID_DIR=/opt/hadoop/hbase/pids
```

Figura 44 Configuración archivo hbase-env.xml MV Hadoop-Master (Fuente: Galeano Cruz y Domínguez Rivera, 2017)

Por último, configuramos las variables de entorno que requiere HBase para su correcto funcionamiento agregándolas al archivo bashrc.

```
export PATH=$PATH:/opt/hadoop/hbase/bin
export CLASSPATH=$CLASSPATH:/opt/hadoop/hbase/lib/*
```

Figura 45 Resultado de configuración variables de entorno HBase (Fuente: Galeano Cruz y Domínguez Rivera, 2017)

Una vez configurado todo procedemos a replicar la información en los demás nodos del clúster utilizando el comando scp, por último, ejecutamos el comando start-hbase.sh, se debe tener en cuenta que para poder iniciar HBase primero debe haberse iniciado HDFS, aunque no es necesario haber iniciado YARN.

Cuando se ha iniciado HBase se podrán encontrar en el maestro, adicional a los procesos de Hadoop, los procesos HQuorumPeer y HMaster.

Figura 46 Resultado de inicializar HBase MV Hadoop-Master (Fuente: Galeano Cruz y Domínguez Rivera, 2017)

Cuando se ha iniciado HBase se podrán encontrar en el esclavo 0-0 (el que fue escogido para ser el backup del master), adicional a los procesos de Hadoop, los procesos HRegionServer, HQuorumPeer y HMaster.

Figura 47 Resultado de inicializar HBase MV Hadoop-Slave-0-0 (Fuente: Galeano Cruz y Domínguez Rivera, 2017)

Cuando se ha iniciado HBase se podrán encontrar en el esclavo 0-1, adicional a los procesos de Hadoop, los procesos HQuorumPeer y HRegionServer.

Figura 48 Resultado de inicializar HBase MV Hadoop-Slave-0-1 (Fuente: Galeano Cruz y Domínguez Rivera, 2017)

Por último, para comprobar que todo haya salido bien se puede entrar al navegador e ingresar a la dirección local de los manejadores de HBase (hadoop-master:16010), estos sitios son locales por lo que hadoop-master es el equivalente a localhost, la dirección del sitio fue configurada en el archivo hbase-site.xml.

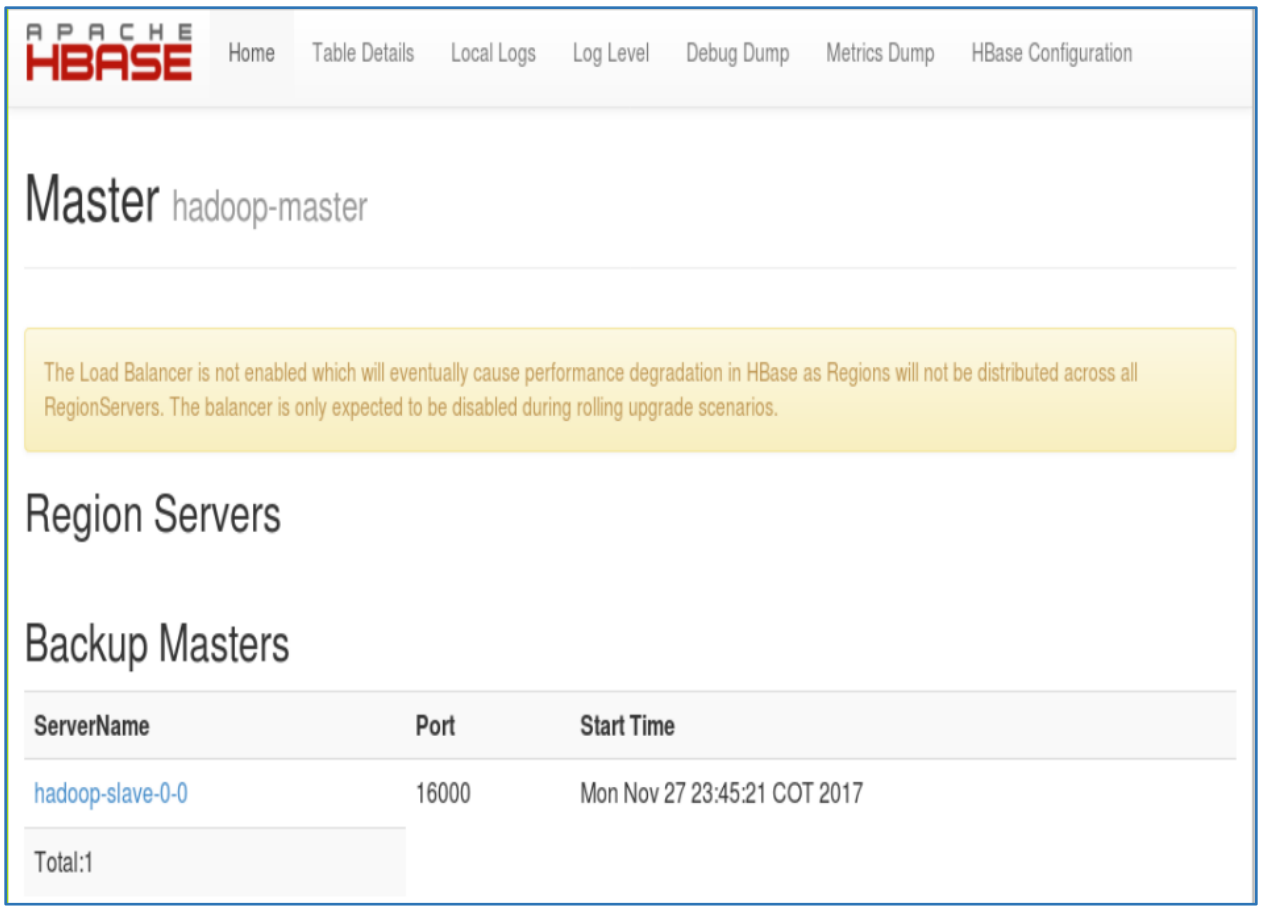

Figura 49 Resultado instalación de HBase ingresando por hadoop-master:16010 (Fuente: Galeano Cruz y Domínguez Rivera, 2017)

El proceso completo de instalación de HBase paso a paso se encuentra en el anexo manual de instalación de HBase.

Durante la instalación de HBase salieron algunos errores que hicieron que el proceso fuera más lento, pero que en definitiva ayudaron a comprender mucho más a fondo como funciona, en este momento compartiremos un pequeño resumen de los errores que salieron y como ayudaron a comprender mejor el funcionamiento de Hadoop.

Al principio al iniciar HBase no estaba iniciando el proceso HMaster en el maestro, esto se debía a que el archivo de configuración hbase-site.xml no se encontraba bien configurados, cuando no inicie el proceso HMaster en el nodo maestro se debe revisar la configuración del archivo hbase.site.xml y verificar que la dirección dada en hbase.rootdir sea correcta, se debe tener en cuenta que en este campo se pone la misma dirección que en el archivo core-site.xml de Hadoop, solo que se le agrega al final la línea /hbase.

Otro de los grandes problemas fue que en todos los nodos iniciaban todos los proceso, además de esto, si se intentaba parar el HBase con el comando stop-hbase.sh, este se quedaba estancado

y nunca apagaba, esto fue algo difícil ya que no se podían ni siquiera parar los procesos, el error se produce nuevamente por una mala configuración de los archivos de configuración, en el archivo backup-master se encontraban las direcciones de todos los esclavos y en el archivo regionservers se encontraba incluida la dirección del master, aparte de la de los esclavos, para solucionarlo solo había que borrar el nombre de uno de los nodos del archivo backup-masters y borrar el nombre del maestro del archivo regionservers, pero aun había un problema, los procesos no se podían parar, por lo que no podía reiniciar HBase, cuando esto suceda solo se tiene que matar directamente los procesos, para esto debes usar el comando kill con el PID del proceso y el PID te lo da el comando jps, se debe realizar esta acción en todos los nodos matando todos los procesos de HBase, esto ayudo a comprender como configurar los maestros de backup y los región servers, además el motivo por el cual HBase no se dejaba finalizar es porque los pids se estaban guardando en el directorio /tmp, en algunos casos esto produce que el HBase nunca pare.

4.3.3 Instalación de Entorno de Desarrollo

Para el entorno de desarrollo se creó una máquina virtual como Linux Red Hat de 64 bits, se le asignaron 4 GB de memoria RAM y se creó un disco duro virtual con 50 GB para el almacenamiento de la información, para las configuraciones de red se habilito una tarjeta de red, adaptador de puente, la cual sirve para 2 cosas, primero permite que la máquina virtual esté conectada a internet, segundo sirve para que la máquina virtual tenga una conexión a internet directa, es decir, la máquina virtual puede ser accedida desde afuera del sistema de virtualización.

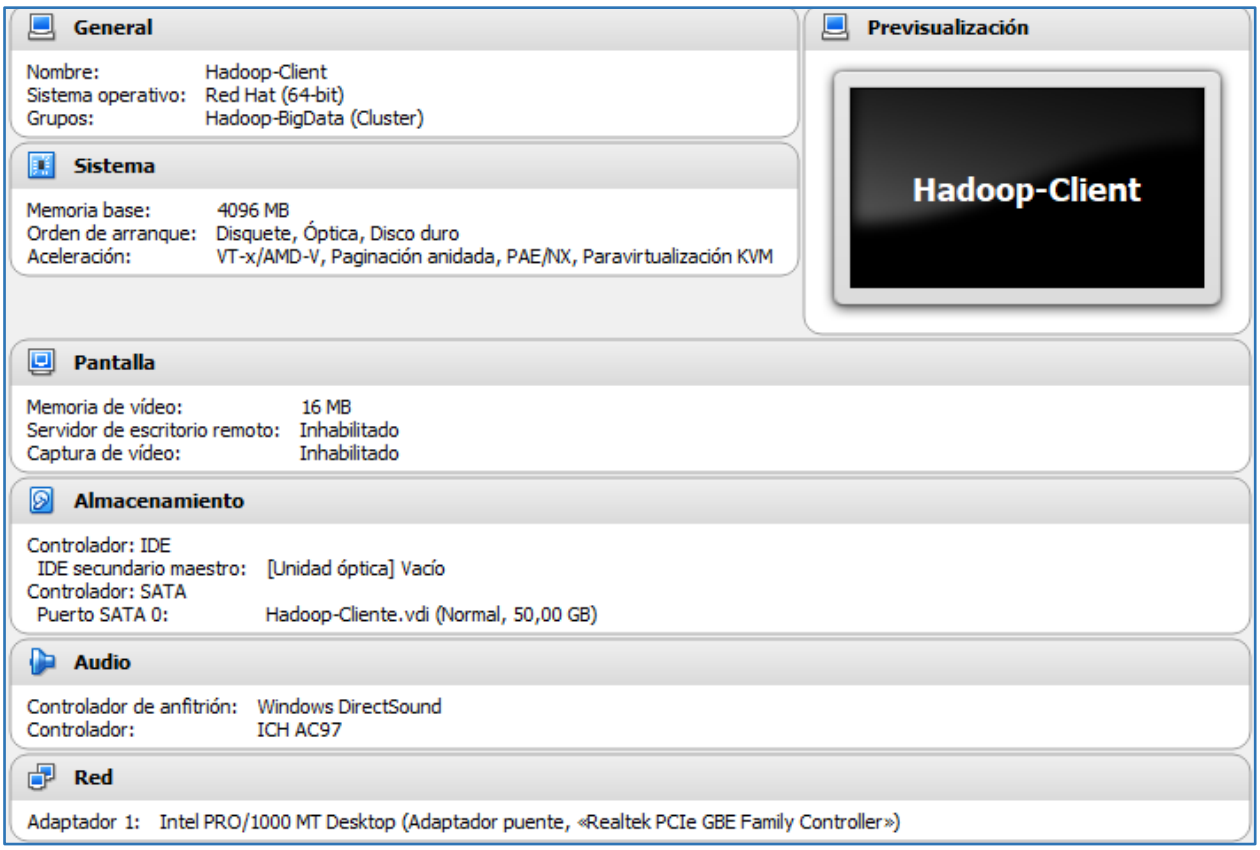

Figura 50 Configuración de máquina virtual Hadoop-Client (Fuente: Elaboración Propia)

Una vez creada la máquina virtual procedemos a instalar el sistema operativo, en este caso escogimos CentOS 7 como sistema operativo para todas las maquinas del clúster. En el proceso de instalación se realizó la configuración del SO, se escogió idioma español (Colombia), se configuro la zona horaria y se seleccionó en el entorno base del software a instalar un escritorio Gnome y como complementos seleccionamos herramientas de desarrollo, la razón por la cual se escogió escritorio Gnome es porque hay cosas que resulta más sencillo realizarlas en un entorno gráfico, mas no es necesario instalar un entorno gráfico, con la instalación minia (selección de software predeterminada donde se instala solo el SO sin entorno gráfico, solo por consola). Como destino de la instalación se escogió el disco duro virtual creado junto con la máquina virtual.

Para la configuración del adaptador de puente se inactivo las IPv6 y se dejó la configuración de DHCP para la IPv4.

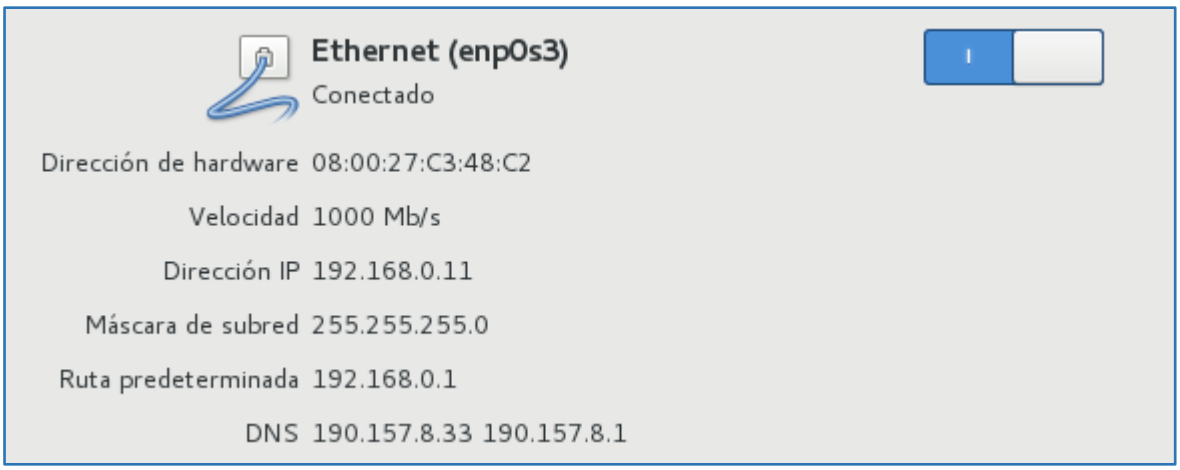

Figura 51 Configuración de adaptador puente MV Hadoop-Client (Fuente: Elaboración Propia)

Por último, se configuraron los usuarios de la máquina virtual, primero definimos una contraseña para el usuario root y luego creamos un usuario normal por el cual vamos a acceder, este usuario lo llamamos BigData, pero se puede nombrar de cualquier forma.

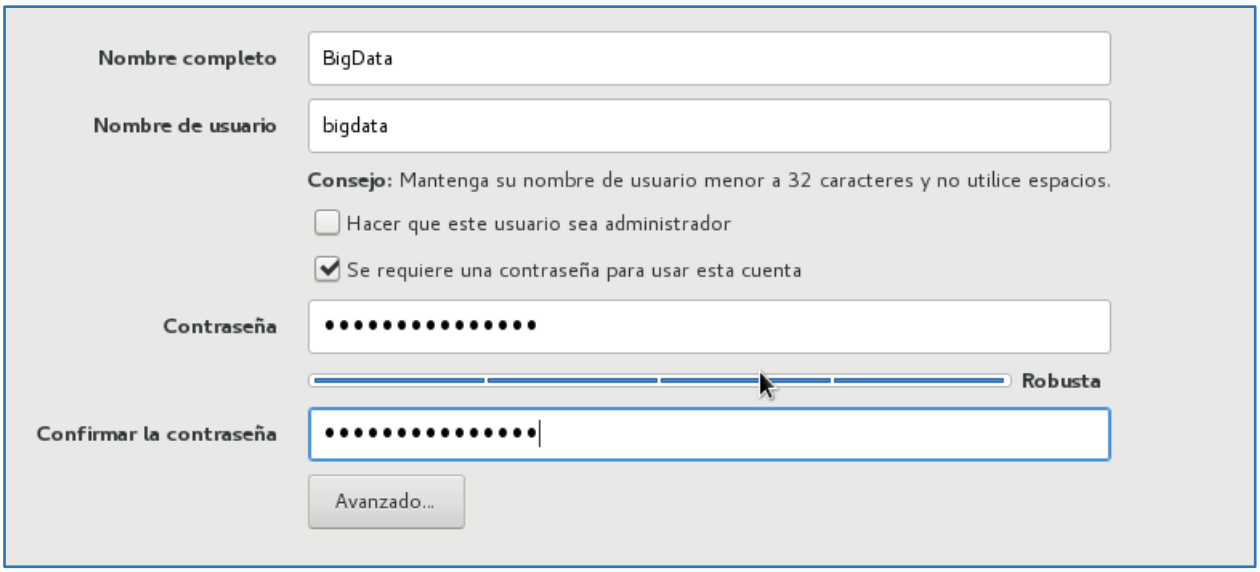

Figura 52 Configuración usuario principal MV Hadoop-Client (Fuente: Elaboración Propia)

Lo primero es instalar y/o actualizar Java en todas las máquinas virtuales, la versión de Java se instalo fue la 1.8.0_152.

```
java version "1.8.0_152"
Java(TM) SE Runtime Environment (build 1.8.0_152-b16)
Java HotSpot(TM) 64-Bit Server VM (build 25.152-b16, mixed mode)
```

Figura 53 Resultado de instalación y/o actualización de Java (Fuente: Elaboración Propia)

Una vez instalado Java se procede a instalar el IDE de desarrollo junto con las herramientas de desarrollo para MapReduce, en este caso de utilizar Eclipse como IDE y Maven como herramienta para programar MapReduce, pero puede usarse cualquier otro IDE que posea las herramientas necesarias para programar MapReduce.

Primero descargamos Eclipse de la página oficial de Eclipse, nosotros descargamos la última versión que hay actualmente Oxigen, una vez descargada la versión de Eclipse a utilizar descomprimimos el archivo tar.gz y movemos la carpeta resultante al directorio final donde se guardara eclipse, en este caso /usr/local. Luego de mover la carpeta de Eclipse procedemos a ubicarnos en su directorio raíz y ejecutamos el archivo eclipse-inst.

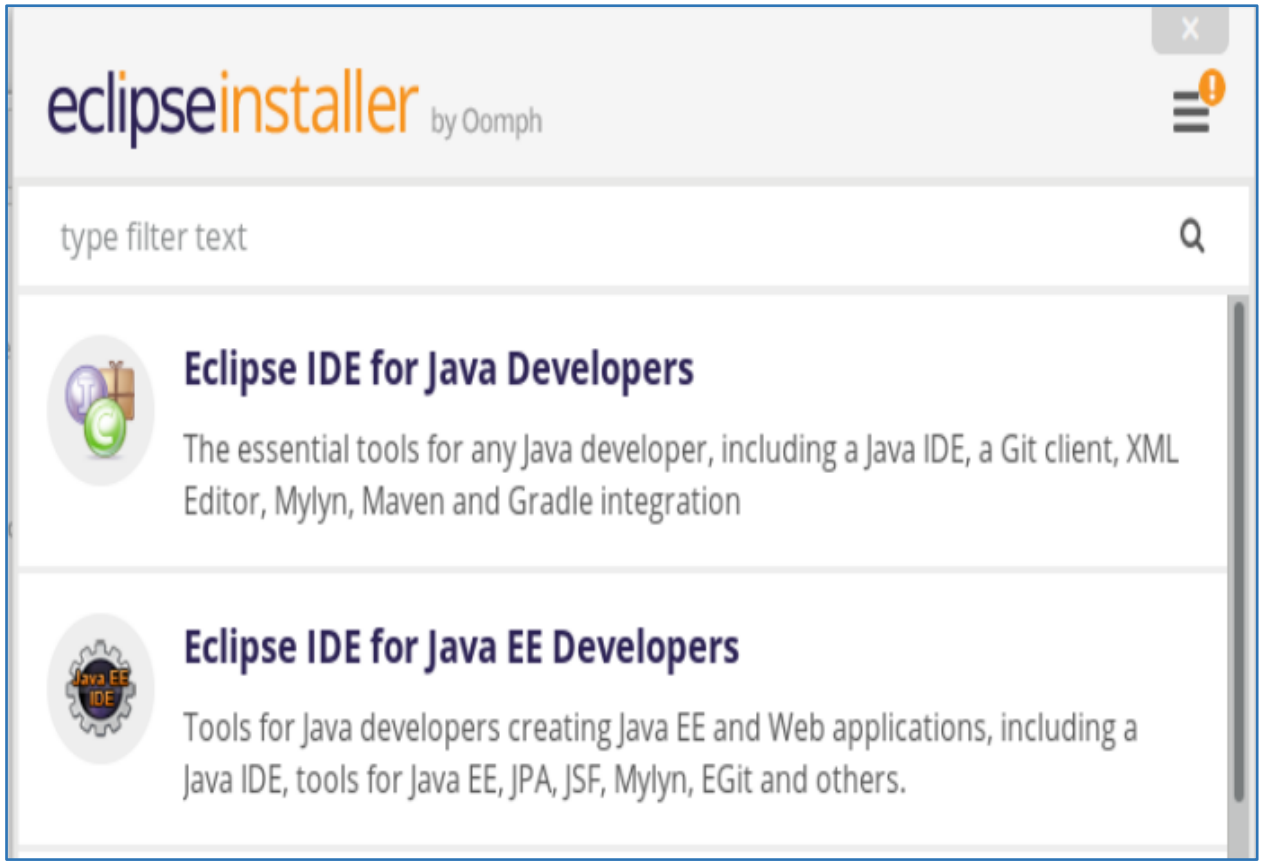

Figura 54 Resultado de ejecución de instalador de Eclipse (Fuente: Elaboración Propia)

El segundo paso es descargar Maven, que es la herramienta que nos proporcionara los complementos necesarios para desarrollar el programa MapReduce.

Una vez instaladas las herramientas procedemos a realizar el código con el cual analizaremos los datos, pero primero observamos los datos para saber que se va a analizar, en este caso la información a analizar es un archivo .csv que contiene información sobre estados y ciudades que producen energía CHP en estados unidos y cuanta energía producen.

```
State;City;Organization Name;Facility Name;Application;SIC4;NAICS;Op Year;Last Installation Year;Capacity (kW);Prime Mover;Fue
AK;Healy;Golden Valley Electric Association;Healy Unit 2 Power Plant;Misc. Manufacturing;3900;339999;2016;;50.000;Boiler/Steam
AK;New Stuyahok;Alaska Village Electric Cooperative;New Stuyahok;Schools;8211;61111;2016;;1.331;Reciprocating Engine;OIL - Oil
AK;Noorvik;Noorvik;Noorvik;Utilities;4931;221112;2016;;1.614;Reciprocating Engine;OIL - Oil;;CRLF
AK;Wasilla;Knikatnu, Inc.;TransAlaska Building;Office Building;6512;53112;2016;;35;Microturbine;NG - Propane;;CRLF
AK;Yakutat;Yak-Tat Kwaan, Inc.;Yakutat Community Health Center- Kwaan Plaza;Hospitals/Healthcare;8011;621111;2016;;10;Microtur
AK;Saint Michael;Alaska Village Electric Cooperative;Saint Michael;Laundries;7215;81231;2015;;700;Reciprocating Engine;OIL - D
AK;Tatitlek;Native Village of Tatitlek;Tatitlek;Utilities;4931;221112;2015;;315;Reciprocating Engine;OIL - Oil;;CRLF
AK;;Hospital;Hospital;Hospitals/Healthcare;8062;62211;2014;;10;Reciprocating Engine;NG - NG;;XCRLF
AK;Saint George;Saint George Municipal Electric Utility;Saint George;General Gov't.;9199;92119;2014;;850;Reciprocating Engine;
AK;Stebbins;Alaska Village Electric Cooperative;Stebbins;Utilities;4939;22131;2014;;2.000;Reciprocating Engine;OIL - Oil;;CRL
AK;Unalaska;City of Unalaska Utility;Unalaska Utility;Utilities;4931;221112;2014;;200;Organic Rankine Cycle;WAST - Waste Heat;
AK;Akiak;Akiak City Council;Akiak;Utilities;4931;22112;2013;;860;Reciprocating Engine;OIL - Distillate Fuel Oil;;CRLF
AK;Anchorage;H2Oasis Indoor Water Park;H2Oasis Indoor Water Park;Amusement/Recreation;7991;713940;2013;;245;Microturbine;NG -
AK;Buckland;City of Buckland;Buckland;Utilities;4931;221112;2013;;1.125;Reciprocating Engine;OIL - Distillate Fuel Oil;;CRLF
AK;Fairbanks;Capstone Turbine Corporation;Fairbanks Food Bank;Community Services;8322;82421;2013;;70;Microturbine;NG - NG;;CRL
AK;Mekoryuk;Alaska Village Electric Cooperative;Mekoryuk;Utilities;4931;221112;2013;;849;Reciprocating Engine;OIL - Oil;;CRLF
AK;Point Lay;North Slope Borough Power & Light;Point Lay;Laundries;7215;81231;2013;;700;Reciprocating Engine;OIL - Oil;;CRLF
```

Figura 55 Archivo a analizar con el ejemplo de programación MapReduce (Fuente: Galeano Cruz y Domínguez Rivera, 2017)

4.4 Fase de desarrollo

Después de conocer los datos que se van a analizar procedemos configurar las dependencias de Maven, en el archivo pom.xml, necesarias para programar MapReduce.

```xml
<project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd">
<modelVersion>4.0.0</modelVersion>

    <groupId>com.laboratory.hadoop</groupId>
    <artifactId>MapReduce</artifactId>
    <version>0.0.1-SNAPSHOT</version>
    <packaging>jar</packaging>

    <name>MapReduce</name>
    <url>http://maven.apache.org</url>

    <properties>
        <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>
    </properties>

    <repositories>
        <repository>
            <id>java.net</id>
            <url>http://download.java.net/maven/2/</url>
        </repository>
        <repository>
            <id>cloudera-releases</id>
            <url>https://repository.cloudera.com/artifactory/cloudera-repos</url>
            <releases>
                <enabled>true</enabled>
            </releases>
            <snapshots>
                <enabled>false</enabled>
            </snapshots>
        </repository>
    </repositories>
```

Figura 56 Fragmento de archivo pom.xml Maven (Fuente: Galeano Cruz y Domínguez Rivera, 2017)

A continuación, en la *figura 57* tenemos el diagrama de clases que ilustra la estructura de las clases que vamos a utilizar para elaborar este ejemplo.

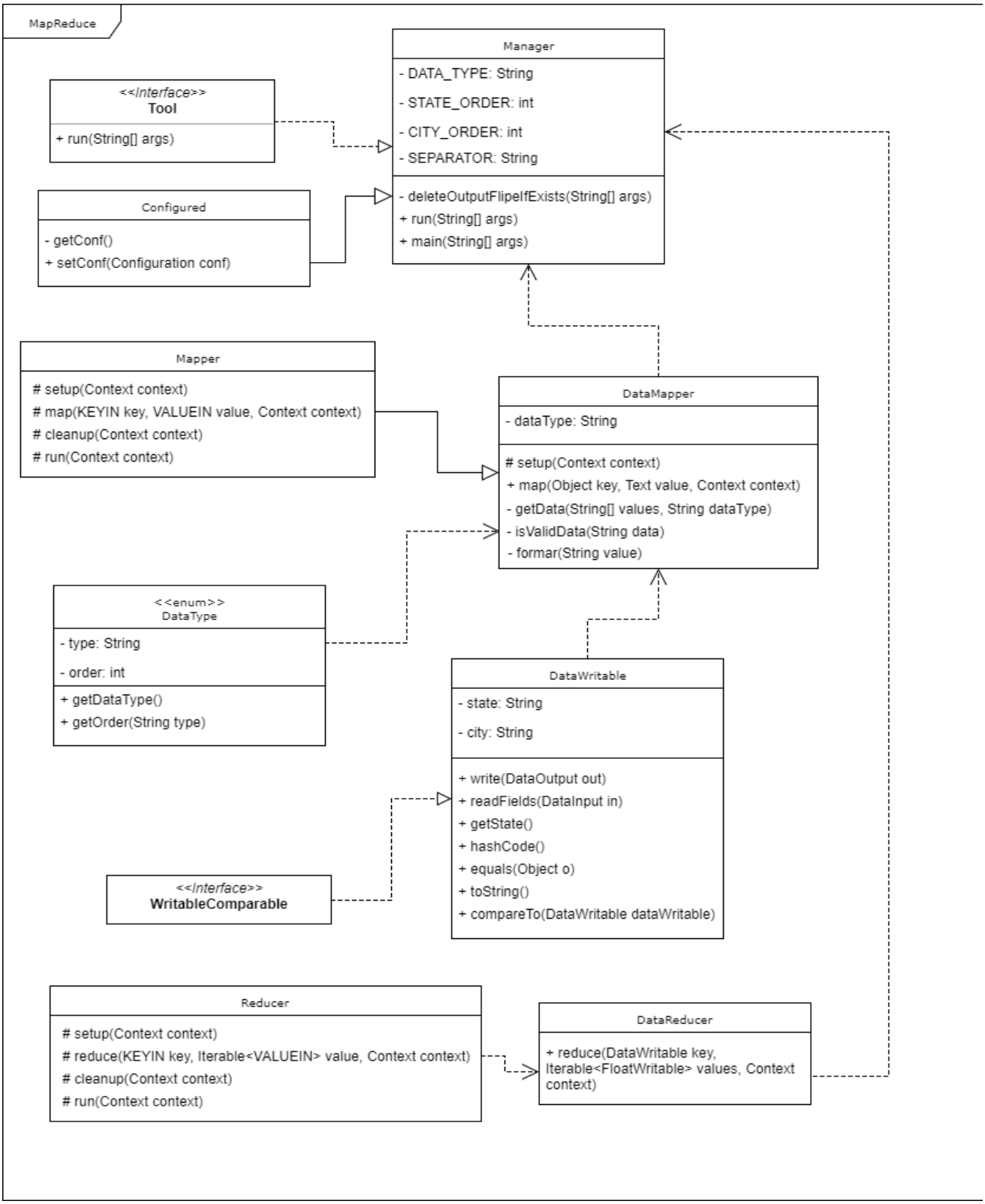

Figura 57 Diagrama de clases del ejemplo programación MapReduce (Fuente: Galeano Cruz y Domínguez Rivera, 2017)

En el diagrama de clases se puede observar como clase principal la clase Manager, que contiene las clases DataMapper y DataReducer, además instancia la clase DataWriteable y utiliza el enum DataType.

Ya con todo listo podemos empezar a la realización del ejemplo de programación MapReduce, en este caso se utilizarán la implementación del diagrama de clases de la *figura 57*:

La primera es la case Manager, en donde se programa la ejecución del programa, además en esta clase se encuentran los métodos Mapper, que se encarga de implementar la función map que hace el mapeo o barrido de los datos, y la clase Reducer, que contiene la función reduce encargada de realizar el procesamiento de los datos mapeados (de ahí el nombre MapReduce).

```
public class Manager extends Configured implements Tool {

    private static final String DATA_TYPE = "dataType";

    private static final int STATE_ORDER = 0;

    private static final int CITY_ORDER = 1;

    private static final String SEPARATOR = ";";

    public static class DataMapper extends Mapper<Object, Text, DataWritable, FloatWritable> {

        private String dataType;

        @Override
        protected void setup(Context context) throws IOException, InterruptedException {
            this.dataType = context.getConfiguration().get(DATA_TYPE);
        }

        public void map(Object key, Text value, Context context) throws IOException, InterruptedException {
            final String[] values = value.toString().split(SEPARATOR);
            if (values.length >= 9 && !values[CITY_ORDER].isEmpty()) {
                final DataWritable data = getData(values, dataType);
                final String dataValue = format(values[DataType.getOrder(dataType)]);

                if (data != null && NumberUtils.isNumber(dataValue)) {
                    context.write(data, new FloatWritable(Float.valueOf(dataValue)));
                }
            }
        }

        private DataWritable getData(String[] values, String dataType) {
            DataWritable dataWritable = null;

            final String state = format(values[STATE_ORDER]);

            if (isValidData(state)) {
                final String city = format(values[CITY_ORDER]);

                dataWritable = new DataWritable(state, city);
            }

            return dataWritable;
        }

        private boolean isValidData(final String data) {
            return !data.isEmpty();
        }

        private String format(String value) {
            return value.trim();
        }
    }
```

Figura 58 Fragmento de archivo Manajer.java Maven (Fuente: Galeano Cruz y Domínguez Rivera, 2017)

La segunda clase es la clase es DataWritable, la cual es la que define un nuevo objeto Writable que se usa en la clase Mapper y Reducer, en este caso la clase Writable hace referencia a un objeto que almacena el estado y la ciudad de los datos.

```
public class DataWritable implements WritableComparable<DataWritable> {

    private String state;
    private String city;

    public DataWritable() {

    }

    public DataWritable(String state, String city) {
        this.state = state;
        this.city = city;
    }

    @Override
    public void write(DataOutput out) throws IOException {
        Text.writeString(out, state);
        Text.writeString(out, city);
    }

    @Override
    public void readFields(DataInput in) throws IOException {
        state = Text.readString(in);
        city = Text.readString(in);
    }

    public String getState() {
        return this.state;
    }

    @Override
    public int hashCode() {
        return new HashCodeBuilder().append(state).append(city).toHashCode();
    }

    @Override
    public boolean equals(Object o) {
        if (!(o instanceof DataWritable)) {
            return false;
        }

        final DataWritable other = (DataWritable) o;
        return new EqualsBuilder().append(state, other.state).append(city, other.city).isEquals();
    }

    @Override
    public String toString() {
        return "(" + state + ") - " + city;
    }

    @Override
    public int compareTo(DataWritable dataWritable) {
        return new CompareToBuilder().append(this, dataWritable).toComparison();
    }
```

Figura 59 Fragmento de archivo DataWritable.java Maven (Fuente: Galeano Cruz y Domínguez Rivera, 2017)

La tercera clase es la clase DataType, en la cual se estructura y define el objeto que se utilizara, el cual que recibe como parámetro el nombre de los datos y arroja como respuesta la el numero de la columna del archivo .csv a extraer, esta clase se utiliza para definir que columna del archivo .csv se va a trabajar.

```java
package com.laboratory.hadoop.MapReduce;

public enum DataType {

    ON("capacity",2),
    FN("capacity",3),
    AP("capacity",4),
    SI("capacity",5),
    NA("capacity",6),
    OY("capacity",7),
    LY("capacity",8),
    CAPACITY("capacity",9);

    private final String type;
    private final int order;

    private DataType(String value, int order) {
        this.type = value;
        this.order = order;
    }

    public String getType() {
        return type;
    }

    public static int getOrder(String type) {

        for (DataType dataType : DataType.values()) {
            if (dataType.getType().equals(type)) {
                return dataType.order;
            }
        }

        // Value by default
        return DataType.CAPACITY.order;
    }
}
```

Figura 60 Archivo DataWritable.java Maven (Fuente: Galeano Cruz y Domínguez Rivera, 2017)

Por último, después de haber escrito el programa MapReduce procedemos a compilarlo con el comando mvn clean install de Maven, el cual nos genera el archivo .jar que se procederá a ejecutar con Hadoop.

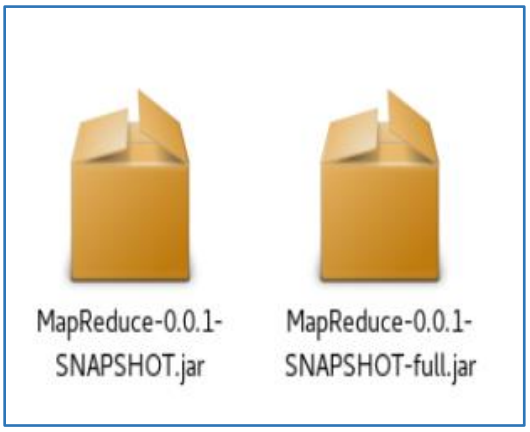

Figura 61 Programa MapReduce compilado con Maven (Fuente: Galeano Cruz y Domínguez Rivera, 2017)

4.5 Fase de implementación

La última fase del desarrollo se realiza la ejecución del ejemplo de programación MapReduce, para esto ingresamos los datos al HDFS y ejecutamos el programa en MapReduce.

```
[hadoop@Hadoop ~]$ hadoop jar /opt/hadoop/MapReduce-0.0.1-SNAPSHOT.jar com.labor
atory.hadoop.MapReduce.Manager input output DataType
```

Figura 62 Ejecución de programa MapReduce (Fuente: Galeano Cruz y Domínguez Rivera, 2017)

Por último, entramos al HDFS de Hadoop y con el comando cat procedemos a ver el resultado del análisis de los datos.

```
(MI) - Marshall 800.0
(MN) - Le Sueur 900.0
(MO) - Macon    10.0
(MS) - Monticello       60.0
(MT) - Kalispell        1.6
(NC) - Winston-Salem    740.0
(ND) - Hillsboro        13.3
(NH) - Middleton        600.0
(NJ) - Lakewood 620.0
(NY) - Jackson Heights  820.0
(OH) - Wooster  750.0
(OR) - Junction City    14.0
(PA) - Swedeland        400.0
(RI) - Providence       480.0
(SC) - Hartsville       22.0
(TN) - Calhoun  66.0
(TX) - Texas City       935.0
(UT) - Moab     60.0
(VA) - Petersburg       7.68
(VT) - Weybridge        600.0
(WA) - Seattle  20.0
(WI) - Middleton        763.0
(WY) - Afton    7.0
[hadoop@Hadoop ~]$
```

Figura 63 Fragmento del resulta del análisis de datos con MapReduce (Fuente: Galeano Cruz y Domínguez Rivera, 2017)

RESULTADOS

5.1 Resultados del Desarrollo e implementación

Como resultado del desarrollo se realizará y documentará un ejercicio completo en el clúster de Hadoop.

Primero abrimos el archivo de Excel con la información, el archivo consta de un documento informativo donde se muestran las entidades que generan energía CHP en estados unidos, así como la ciudad a la que pertenece, el estado, el tipo de combustible, etc.

State	City	Organization Name	Facility Name	Application	SIC	NAIC	Op Yea	Last Installation Yea	Capacity (kW	Prime Mover	Fuel Class - Primary Fuel	Project Profi	ritical Infrastruc
AK	Healy	Golden Valley Electric Association	Healy Unit 2 Power Plant	Misc. Manufacturing	3900	339999	2016		50.000	Boiler/Steam Turbine	COAL - Coal		
AK	New Stuyahok	Alaska Village Electric Cooperative	New Stuyahok	Schools	8211	61111	2016		1.331	Reciprocating Engine	OIL - Oil		X
AK	Noorvik	Noorvik	Noorvik	Utilities	4931	221112	2016		1.614	Reciprocating Engine	OIL - Oil		
AK	Wasilla	Knikatnu, Inc.	TransAlaska Building	Office Building	6512	53112	2016		35	Microturbine	NG - Propane		
AK	Yakutat	Yak-Tat Kwaan, Inc.	Yakutat Community Health Center	Hospitals/Healthcare	8011	621111	2016		10	Microturbine	NG - Propane		X
AK	Saint Michael	Alaska Village Electric Cooperative	Saint Michael	Laundries	7215	81231	2015		700	Reciprocating Engine	OIL - Distillate Fuel Oil		
AK	Tatitlek	Native Village of Tatitlek	Tatitlek	Utilities	4931	221112	2015		315	Reciprocating Engine	OIL - Oil		
AK		Hospital	Hospital	Hospitals/Healthcare	8062	62211	2014		10	Reciprocating Engine	NG - NG		X
AK	Saint George	Saint George Municipal Electric Util	Saint George	General Gov't.	9199	92119	2014		850	Reciprocating Engine	OIL - Oil		
AK	Stebbins	Alaska Village Electric Cooperative	Stebbins	Utilities	4939	22131	2014		2.000	Reciprocating Engine	OIL - Oil		
AK	Unalaska	City of Unalaska Utility	Unalaska Utility	Utilities	4931	221112	2014		200	Organic Rankine Cycle	WAST - Waste Heat		
AK	Akiak	Akiak City Council	Akiak	Utilities	4931	22112	2013		860	Reciprocating Engine	OIL - Distillate Fuel Oil		
AK	Anchorage	H2Oasis Indoor Water Park	H2Oasis Indoor Water Park	Amusement/Recreation	7991	713940	2013		245	Microturbine	NG - NG		
AK	Buckland	City of Buckland	Buckland	Utilities	4931	221112	2013		1.125	Reciprocating Engine	OIL - Distillate Fuel Oil		
AK	Fairbanks	Capstone Turbine Corporation	Fairbanks Food Bank	Community Services	8322	82421	2013		70	Microturbine	NG - NG		
AK	Mekoryuk	Alaska Village Electric Cooperative	Mekoryuk	Utilities	4931	221112	2013		849	Reciprocating Engine	OIL - Oil		
AK	Point Lay	North Slope Borough Power & Light	Point Lay	Laundries	7215	81231	2013		700	Reciprocating Engine	OIL - Oil		
AK	Anchorage	Municipality of Anchorage/Doyon U	Anchorage Landfill Gas to Energy	Solid Waste Facilities	4953	562212	2012		5.600	Reciprocating Engine	BIOMASS - LFG		
AK	Cordova	Cordova Electric Cooperative	Cordova	Utilities	4931	221112	2012		4.050	Reciprocating Engine	OIL - Distillate Fuel Oil		
AK	Hoonah	Tlingit-Haida Regional Electrical Au	Hoonah Power Plant	Utilities	4939	221112	2012		3.000	Reciprocating Engine	OIL - Oil		
AK	King Cove	City of King Cove Utility	King Cove	Construction	1629	23713	2012		2.600	Reciprocating Engine	OIL - Distillate Fuel Oil		
AK	Tok	Gateway School District	Gateway School District	Schools	8211	61111	2012		120	Other	BIOMASS - Biomass	http://www.nort	X
AK	Chitina	Chitina	Chitina	Utilities	4931	22112	2011		310	Reciprocating Engine	OIL - Distillate Fuel Oil		
AK	Igiugig	Rural Power System Upgrade - Villag	Igiugig	Utilities	4931	221112	2011		235	Reciprocating Engine	OIL - Distillate Fuel Oil		
AK	Kwethluk	Kwethluk	Kwethluk	Utilities	4931	22112	2010		1.050	Reciprocating Engine	OIL - Distillate Fuel Oil		
AK	Nelson Lagoon	Nelson Lagoon Electric Cooperative	Nelson Lagoon	Utilities	4931	221112	2010		290	Reciprocating Engine	OIL - Distillate Fuel Oil		
AK	Ouzinkie	Ouzinkie Electric	Ouzinkie	Utilities	4931	22112	2010		350	Boiler/Steam Turbine	OIL - Distillate Fuel Oil		
AK	Tanana	Tanana Power Company	Tanana	Utilities	4931	221112	2010		1.260	Reciprocating Engine	OIL - Distillate Fuel Oil		
AK	Unalakleet	Unalakleet Valley Electric Cooperati	Unalakleet	Wastewater Treatment	4941	22131	2010		2.010	Reciprocating Engine	OIL - Oil		X
AK	Akiachak	Akiachak Native Community	Akiachak	Utilities	4931	22112	2009		1.500	Reciprocating Engine	OIL - Distillate Fuel Oil		
AK	Bethel	Bethel	Bethel	Utilities	4931	22112	2009		12.600	Reciprocating Engine	OIL - Distillate Fuel Oil		
AK	Karluk	Karluk	Karluk	Utilities	4931	22112	2009		108	Reciprocating Engine	OIL - Distillate Fuel Oil		
AK	Kipnuk	Kipnuk Light Plant	Kipnuk	General Gov't.	9131	92114	2009		1.200	Reciprocating Engine	OIL - Oil		
AK	Levelock	Levelock Electric Cooperative, Inc.	Levelock	Schools	8211	61111	2009		234	Reciprocating Engine	OIL - Distillate Fuel Oil		X
AK	Ruby	Ruby Electric Utility	Ruby	Utilities	4931	22112	2009		600	Reciprocating Engine	OIL - Distillate Fuel Oil		
AK	Beaver	Beaver Joint Utilities	Beaver	Utilities	4931	22112	2008		250	Reciprocating Engine	OIL - Distillate Fuel Oil		
AK	Diomede	Diomede	Diomede	General Gov't.	9199	92119	2008		460	Reciprocating Engine	OIL - Distillate Fuel Oil		
AK	Pilot Point	Pilot Point Electric Utility	Pilot Point	Schools	8211	61111	2008		235	Reciprocating Engine	OIL - Oil		X
AK	Savoonga	Alaska Village Electric Cooperative	Savoonga	Utilities	4931	221112	2008		1.523	Reciprocating Engine	OIL - Oil		
AK	Dillingham	Dillingham	Dillingham	Wastewater Treatment	4952	221132	2007		4.400	Reciprocating Engine	OIL - Distillate Fuel Oil		X
AK	Elfin Cove	Elfin Cove Electric Utility	Elfin Cove	Community Services	8322	62419	2007		547	Reciprocating Engine	OIL - Distillate Fuel Oil		
AK	Kotlik	Kotlik Electric Service	Kotlik	Utilities	4931	221112	2007		1.400	Reciprocating Engine	OIL - Oil		

Figura 64 Fragmento de archivo Exel (Fuente: Galeano Cruz y Domínguez Rivera, 2017)

El archivo tiene un total de 4396 registros, lo convertimos a csv para que sea un archivo de texto.

Figura 65 Fragmento de archivo CSV (Fuente: Galeano Cruz y Domínguez Rivera, 2017)

Lo segundo a realizar es compilar el programa MapReduce, esto se realiza con el comando mvn clear install de Maven, el comando se ejecuta por consola, pero para ejecutarlo debemos encontrarnos en el directorio donde se encuentra el archivo pom.xml del ambiente de desarrollo.

```
[root@Hadoop ~]# cd /home/bigdata/mapreduce/jobs/MapReduce/
[root@Hadoop MapReduce]# ls
pom.xml  src  target
[root@Hadoop MapReduce]# mvn clean install
[INFO] Scanning for projects...
[INFO]
[INFO] ------------------------------------------------------------------------
[INFO] Building MapReduce 0.0.1-SNAPSHOT
[INFO] ------------------------------------------------------------------------
```

Figura 66 Proceso de compilación del programa MapReduce (Fuente: Galeano Cruz y Domínguez Rivera, 2017)

```
[INFO] ------------------------------------------------------------------------
[INFO] BUILD SUCCESS
[INFO] ------------------------------------------------------------------------
[INFO] Total time: 01:04 min
[INFO] Finished at: 2017-11-29T18:56:42-05:00
[INFO] Final Memory: 37M/152M
[INFO] ------------------------------------------------------------------------
[root@Hadoop MapReduce]# ls target/
archive-tmp            MapReduce-0.0.1-SNAPSHOT-full.jar  surefire-reports
classes                MapReduce-0.0.1-SNAPSHOT.jar       test-classes
generated-sources      maven-archiver
generated-test-sources maven-status
[root@Hadoop MapReduce]#
```

Figura 67 Proceso de compilación del programa MapReduce (Fuente: Galeano Cruz y Domínguez Rivera, 2017)

Una vez compilado el programa MapReduce procedemos a pasarlo de la maquina Cliente al Maestro por medio de ssh.

```
[bigdata@Hadoop ~]$ scp /home/bigdata/mapreduce/MapReduce-0.0.1-SNAPSHOT.jar 192
.168.0.6:/home/bigdata
bigdata@192.168.0.6's password:
MapReduce-0.0.1-SNAPSHOT.jar                  100%   10KB   1.0MB/s   00:00
[bigdata@Hadoop ~]$ scp /home/bigdata/ 192.168.0.6:/home/bigdata
apache-maven-3.5.2/ eclipse/            Música/
.bash_history       .eclipse/           .p2/
.bash_logout        eclipse-installer/  Plantillas/
.bash_profile       Escritorio/         Público/
.bashrc             .esd_auth           .ssh/
.cache/             .ICEauthority       .swt/
CHPDB_database.csv  Imágenes/           .tooling/
.config/            .local/             Vídeos/
Descargas/          mapreduce/
Documentos/         .mozilla/
[bigdata@Hadoop ~]$ scp /home/bigdata/mapreduce/ 192.168.0.6:/home/bigdata
MapReduce-0.0.1-SNAPSHOT.jar  .metadata/
[bigdata@Hadoop ~]$ scp /home/bigdata/CHPDB_database.csv 192.168.0.6:/home/bigda
ta
bigdata@192.168.0.6's password:
CHPDB_database.csv                            100%  574KB   4.5MB/s   00:00
[bigdata@Hadoop ~]$
```

Figura 68 Proceso de transferencia de archivos del cliente al maestro (Fuente: Galeano Cruz y Domínguez Rivera, 2017)

Al finalizar de pasar los archivos comprobamos que estos se encuentren en el maestro.

```
[bigdata@Hadoop ~]$ ls /home/bigdata/
CHPDB_database.csv  eclipse     mapreduce                      Plantillas
Descargas           Escritorio  MapReduce-0.0.1-SNAPSHOT.jar  Público
Documentos          Imágenes    Música                         Vídeos
[bigdata@Hadoop ~]$
```

Figura 69 Comprobación de la transferencia de archivos ssh del cliente al maestro (Fuente: Galeano Cruz y Domínguez Rivera, 2017)

Luego de tener los archivos en el maestro ingresamos con el usuario root para mover los archivos al directorio input que creamos dentro de hadoop, esto con el fin de que el usuario hadoop tenga acceso a los archivos que asamos ya que estos originalmente pasaron al directorio bigdata, al cual no tiene acceso el usuario hadoop.

```
[root@Hadoop ~]# mv /home/bigdata/CHPDB_database.csv /opt/hadoop/input/
[root@Hadoop ~]# mv /home/bigdata/MapReduce-0.0.1-SNAPSHOT.jar /opt/hadoop/input/
[root@Hadoop ~]# ls /opt/hadoop/input/
CHPDB_database.csv  MapReduce-0.0.1-SNAPSHOT.jar
[root@Hadoop ~]#
```

Figura 70 Proceso de transferencia de archivos a directorio input de hadoop (Fuente: Galeano Cruz y Domínguez Rivera, 2017)

Como penúltimo paso de preparación nos aseguramos de que este corriendo correctamente hadoop con el código jps, ya que sin este ejecutándose no podemos utilizar HDFS para el almacenamiento de los archivos ni el análisis de los mismos.

```
[hadoop@Hadoop ~]$ jps
6374 Jps
3831 NameNode
6183 HMaster
4298 SecondaryNameNode
5052 HQuorumPeer
4525 ResourceManager
[hadoop@Hadoop ~]$
```

Figura 71 Comprobación de ejecución de hadoop (Fuente: Galeano Cruz y Domínguez Rivera, 2017)

Por último, se guardan los archivos a analizar en HDFS de donde serán tomados por Hadoop para el análisis, para esto creamos una nueva carpeta en HDFS que se llame input e introducimos en esta los archivos.

```
[hadoop@Hadoop ~]$ hadoop fs -ls
[hadoop@Hadoop ~]$ hadoop fs -mkdir input
[hadoop@Hadoop ~]$ hadoop fs -copyFromLocal /opt/hadoop/input/CHPDB_database.csv  input
[hadoop@Hadoop ~]$
```

Figura 72 Creación de directorio input y paso de archivos en DHFS (Fuente: Galeano Cruz y Domínguez Rivera, 2017)

Ahora si ejecutamos el análisis de datos con el comando hadoop jar [/ubicación/archivo] [package.clase] [directorio de datos] [directorio de resultado] [otros parámetros definidos].

```
[hadoop@Hadoop ~]$ hadoop jar /opt/hadoop/input/MapReduce-0.0.1-SNAPSHOT.jar com.labora
tory.hadoop.MapReduce.Manager input output default
17/12/02 08:28:29 INFO Configuration.deprecation: session.id is deprecated. Instead, us
e dfs.metrics.session-id
17/12/02 08:28:29 INFO jvm.JvmMetrics: Initializing JVM Metrics with processName=JobTra
cker, sessionId=
```

Figura 73 Hadoop en todo su esplendor (Fuente: Galeano Cruz y Domínguez Rivera, 2017)

Ahora observamos el resultado del análisis del documento y comparamos con el documento original.

```
(NJ) - Lakewood 620.0
(NY) - Jackson Heights  820.0
(OH) - Wooster  750.0
(OR) - Junction City    14.0
(PA) - Swedeland        400.0
(RI) - Providence       480.0
(SC) - Hartsville       22.0
(TN) - Calhoun  66.0
(TX) - Texas City       935.0
(UT) - Moab     60.0
(VA) - Petersburg       7.68
(VT) - Weybridge        600.0
(WA) - Seattle  20.0
(WI) - Middleton        763.0
(WY) - Afton    7.0
[hadoop@Hadoop ~]$
```

Figura 74 Resultado de ejecución de Hadoop y análisis MapReduce (Fuente: Galeano Cruz y Domínguez Rivera, 2017)

En el archivo original teníamos un listado de registros sin información exacta el cual no era sencillo de analizar ahora tenemos un archivo más estructurado donde se entiende más claramente los datos que se querían analizar, encontramos que como resultado del mapeo y reducción de los datos sale una lista de ciudades con sus respectivos estados y la cantidad de energía CHP que producen, este información se puede pasar por una herramienta de minería de datos para sacar gráficas y un análisis predictivo.

CONCLUSIONES

6.1 Objetivos

1. Diseñar un prototipo de laboratorio Hadoop para análisis Big Data, que permita procesar documentos de texto según el programa MapReduce ejecutado.

   Se realizó el diseño del prototipo utilizando una arquitectura compuesta por 3 nodos, 2 esclavos y un maestro, además se configuro un ambiente de desarrollo que funciona como cliente y se comunica con el maestro por medio de ssh para la transferencia de archivos y ejecución de Hadoop.

2. Instalar un entorno de trabajo Hadoop que cumpla las especificaciones del diseño, para la implementación del prototipo en máquinas virtuales utilizando el sistema operativo CentOS.

   Se crearon y configuraron 4 máquinas virtuales, 3 de ellas hacen parte del clúster Hadoop, estas máquinas están conectadas por una red interna y la maquina maestro tiene un adaptador puente para salir a internet, en estas máquinas se instaló Java 1.8.0_152, Hadoop 2.7.0 y HBase 1.2.6; la última maquina es el cliente, esta máquina también está conectada a internet por medio de un adaptador puente y además tiene instalado el ambiente de desarrollo para MapReduce, para este ambiente de desarrollo se utilizó Eclipse Oxigen y Maven 3.5.2

3. Desarrollar un ejemplo de programación en MapReduce que le permita al entorno Hadoop analizar un documento de texto de acuerdo con lo especificado en el diseño.

   Primero se buscó un repositorio de datos de donde sacar información para el análisis, se escogió U.S. DOE Combined Heat and Power Installation Database [37], luego se procedió a escribir el código MapReduce en Java utilizando Maven para el uso de los complementos necesarios, el programa desarrollado permite que MapReduce analice la información y saque un registro en el cual aparta todos los estados y sus ciudades en una lista para luego sumar todas las cantidades de energía producida por cada ciudad.

4. Implementar el prototipo elaborado para ejecutar el ejemplo programado en MapReduce y mostrar el resultado del análisis.

   Para la implementación de prototipo se pasaron los archivos del cliente al maestro del clúster Hadoop utilizando ssh, una vez se transfieren los archivos se procede a conectarse al clúster por medio de ssh e ingresar con el usuario hadoop, desde el cual se transferirán los archivos al HDFS y se ejecutó el programa creado anterior mente, como resultado arrojo

un conjunto de archivos que contienen una lista de estados y ciudades y en frente de cada ciudad se ve la cantidad de energía producida por dicha ciudad por medio de CHP.

5. Diseñar e implementar un prototipo de laboratorio Hadoop para análisis Big Data, que podría permitir a los estudiantes de la Institución Universitaria Politécnico Grancolombiano aprender a instalar y utilizar un entorno Hadoop para análisis Big Data.

   Resumiendo, en cumplimiento al objetivo general se realizó el diseño, instalación e implementación del prototipo para luego probarlo con una fuente de datos y un programa MapReduce que se encarga de realizar un análisis especifico de los datos.

6.2 Pregunta de Investigación

1. ¿Podría ser útil implementar un laboratorio Hadoop para análisis Big Data, en la Institución Universitaria Politécnico Grancolombiano?

   Según lo especificado en las asesorías la universidad requiere una herramienta que le permita a sus estudiantes llevar sus conocimientos teóricos a la práctica sin tener que preocuparse por el tiempo que lleva la instalación del ambiente Hadoop para el análisis Big Data, el laboratorio le permitiría a los estudiantes enfocarse en poner en práctica la habilidades adquiridas en clase con la creación de un programa MapReduce que les permita ejecutar el análisis de los datos según las ETL diseñadas en clase y utilizar el resultado de ese análisis para montarlo en otra herramienta que arroje el análisis predictivo de información.

6.3 Conclusiones Generales

1. Una de las conclusiones más importantes obtenidas en el desarrollo del proyecto fue el aprendizaje adquirido, utilizar Big Data ayudo a alcanzar una curva de aprendizaje que en pocas partes de Colombia se ha logrado e implementado, el hecho de verificar, investigar, desarrollarlo, fue un gran logro y requirió de gran esfuerzo.

2. De acuerdo con la investigación realizada se puede concluir que Hadoop es una herramienta muy útil para realizar el almacenamiento y procesamiento de grandes volúmenes de información, además de ser un framework gratuito, ya que permite recolectar información ya sea en una empresa grande o pequeña y analizarla sin que sea muy costoso, además de suministrar una herramienta completamente escalable, pues si la empresa crece y por ende la información crece, basta con agregar un servidor a hadoop y no afectar el análisis y almacenamiento de la información que ya se encuentra alojada en los otros servidores. Otro aspecto muy importante es que en Colombia esta tecnología es muy nueva ya que muy pocas empresas la implementan.

3. Existen 3 tipos de instalaciones: único nodo en local (single node) hace referencia a la instalación y ejecución en un único nodo, clúster pseudo-distribuido hace referencia a la instalación en una misma máquina simulando un cluster y totalmente distribuido, hace referencia a un clúster entre distintas máquinas (multi node). De acuerdo a lo investigado en Colombia se ha realizado la implementación en un único nodo o semi distribuida, por este motivo se decidió realizarlo en un cluster distribuido multi nodo, dejando la arquitectura de un prototipo para que sea implementado en la universidad, de modo que, en caso de necesitar agregar más nodos se realice de forma rápida y eficiente.

4. Se deja el prototipo y los manuales de instalación e implementación para uso de los estudiantes del Politécnico Grancolombiano que quieran realizar la implementación de un ambiente Hadoop y decidan desarrollar un ejemplo en MapReduce que analice datos.

TRABAJO FUTURO

Como trabajo futuro, en cuanto a la herramienta planteada en este documento, se propone mejorar el sistema en cuanto a seguridad y el desarrollo de una plataforma web que permita el acceso a esta herramienta sin tener que conectarse directamente al clúster por medio de ssh. Además, se pueden añadir más funcionalidades de Hadoop, Hadoop provee un amplio ecosistema de aplicaciones que tienen diversos usos y pueden ser de utilidad para profundizar en esta herramienta, se puede reemplazar parte de la programación en MapReduce con Pig o Hive, que son herramientas útiles para facilitar el trabajo, también se puede integrar la herramienta con un programa de minería de datos que permita hacer el análisis predictivo de datos y saque las gráficas necesarias.

REFERENCIAS

[1] S. X. Q. Z. Y. G. P. Y. Y. L. Ting Zhu, «Emergent Technologies in Big Data Sensing: A Survey,» *International Journal of Distributed Sensor Networks,* vol. 2015, 2015.

[2] A. Rajpurohit, «Rajpurohit, A. (2013). Big data for business managers - Bridging the gap between potential and value. Big data for business managers - Bridging the gap between potential and value,» *2013 IEEE International Conference on Big Data*, p. 3, 2013.

[3] J. B. Pedro Caldeira Neves, «Big Data Issues,» *Proceeding IDEAS '15 Proceedings of the 19th International Database Engineering & Applications Symposium*, pp. 200-201, 2015.

[4] A. CAOBA, «Alianza CAOBA,» Centro de Excelencia y apropiación en Big Data y Data Analytics, [En línea]. Available: http://alianzacaoba.co/que-es-caoba/. [Último acceso: 12 11 2017].

[5] Dinero, «Dinero,» 19 09 2017. [En línea]. Available: http://www.dinero.com/empresas/articulo/big-data-y-analitica-en-las-empresas-de-colombia/246643. [Último acceso: 11 11 2017].

[6] Dinero, «Dinero,» 08 07 2015. [En línea]. Available: http://www.dinero.com/edicion-impresa/tecnologia/articulo/el-poder-economico-del-big-data-su-desarrollo-colombia/210853. [Último acceso: 11 11 2017].

[7] E. País, «El País,» 04 07 2015. [En línea]. Available: https://elpais.com/elpais/2015/07/02/eps/1435845247_202110.html. [Último acceso: 12 11 2017].

[8] G. C. Plataforma, «Google Cloud Plataforma,» [En línea]. Available: https://cloud.google.com/bigquery/?hl=es. [Último acceso: 12 11 2017].

[9] D. G. Mohd Rehan Ghazi, «Hadoop, mapreduce and HDFS: A developers perspective,» *Procedia Computer Science,* vol. 48, pp. 45-50, 2015.

[10] J. Q. J. Y. B. D. X. L. Y. L. Feng Wang, «Hadoop high availability through metadata replication,» *Proceeding of the first international workshop on Cloud data management - CloudDB '09,* p. 37, 2009.

[11] I. B. M. S. Septiembre, «Integración Big Data y Hadoop,» 2014.

[12] C. V. Daniel Romero, *Tesis Diseño de un prototipo para la implementación de un sistema Big Data,* Bogotá - Colombia: Repositorio Institucional Alejandría, 2015.

[13] J. E. R. P. Fabian Andrés Guerrero López, *Tesis Diseño y desarrollo de una guía para la implementación de un ambiente Big Data,* Bogotá-Colombia: Repositorio Universidad Católica, 2013.

[14] H. J. R. R. J. E. Salinas Hernandez, *Tesis Análisis de la viabilidad de la implementación de Big Data en Colombia,* Bogotá-Colombia: Repositorio Universidad Distrital Francisco José de Caldas, 2016.

[15] D. L. Garcia, *Análisis de las posibilidades de uso de Big Data en las organizaciones,* España: Repositorio Universidad de Cantabria, 2013.

[16] D. R. Pastor, *Big Data en sectores asegurador y financiero,* Barcelona-España: Repositorio Universidad de Barcelona, 2015.

[17] P. Grancolombiano, «Politécnico Grancolombiano,» 2017. [En línea]. Available: https://www.poli.edu.co/content/quienes-somos - 2017. [Último acceso: 25 09 2017].

[18] M. d. T. d. l. I. y. Comunicaciones, «MinTIC,» [En línea]. Available: http://www.mintic.gov.co/portal/604/w3-article-6163.html. [Último acceso: 25 09 2017].

[19] L. J. Aguilar, Big Data - Análisis De Grandes Volúmenes De Datos En Organizaciones, México: Alfaomega Grupo Editor, 2013.

[20] Gartner, «Gartner,» 2013. [En línea]. Available: https://www.gartner.com/it-glossary/big-data. [Último acceso: 23 11 2017].

[21] R. B. Fragoso, «IBM,» 18 06 2012. [En línea]. Available: https://www.ibm.com/developerworks/ssa/local/im/que-es-big-data/index.html. [Último acceso: 23 11 2017].

[22] R. Informático-Tecnológica, «Revista Informático-Tecnológica,» [En línea]. Available: https://revista.jovenclub.cu/unidades-de-medidas-de-informacion-1-kilobyte-no-es-1000-bytes/. [Último acceso: 25 11 2017].

[23] I. d. i. d. conocimiento, «Instituto de ingeniería del conocimiento,» 28 06 2016. [En línea]. Available: http://www.iic.uam.es/innovacion/big-data-caracteristicas-mas-importantes-7-v/. [Último acceso: 23 11 2017].

[24] R. S. J. S. D. R.-M. T. Michael Schroeck, «IBM,» 2012. [En línea]. Available: https://www-05.ibm.com/services/es/gbs/consulting/pdf/El_uso_de_Big_Data_en_el_mundo_real.pdf. [Último acceso: 23 11 2017].

[25] Á. Rayo, «bit - Computer Training,» 17 05 2016. [En línea]. Available: https://www.bit.es/knowledge-center/tipos-de-datos-en-big-data/. [Último acceso: 24 11 2017].

[26] Y. Demchenko, «Defining the Big Data Architecture,» 17 07 2013. [En línea]. Available: https://bigdatawg.nist.gov/_uploadfiles/M0055_v1_7606723276.pdf. [Último acceso: 25 11 2017].

[27] J. F. C. O. L. J. A. Juan José Camargo Vega, «SciELO,» 01 10 2014. [En línea]. Available: http://www.scielo.org.co/scielo.php?script=sci_arttext&pid=S0121-11292015000100006. [Último acceso: 26 11 2017].

[28] Amazon, «Amazon Web Services,» [En línea]. Available: https://aws.amazon.com/es/relational-database/. [Último acceso: 26 11 2017].

[29] A. A. Rivas, «IBM,» 30 09 2013. [En línea]. Available: https://www.ibm.com/developerworks/ssa/library/bd-almacenamiento-datos/index.html. [Último acceso: 26 11 2017].

[30] E. Camacho, «SG Buzz,» [En línea]. Available: https://sg.com.mx/revista/42/nosql-la-evolucion-las-bases-datos#.WhraFUriaHt. [Último acceso: 26 11 2017].

[31] Oracle, «Oracle,» [En línea]. Available: http://www.oracle.com/ocom/groups/public/@otn/documents/webcontent/317529_esa.pdf. [Último acceso: 26 11 2017].

[32] Microsoft, «Microsoft,» 14 03 2017. [En línea]. Available: https://docs.microsoft.com/es-es/sql/analysis-services/data-mining/data-mining-concepts. [Último acceso: 26 11 2017].

[33] K. S. A. Y. Boris Lublinsky, Hadoop - Soluciones Big Data, ANAYA MULTIMEDIA, 2013.

[34] D. Borthakur, «Apache Hadoop,» 2013. [En línea]. Available: https://hadoop.apache.org/docs/r1.2.1/hdfs_design.html. [Último acceso: 27 11 2017].

[35] Ticout, «Ticout,» 02 04 2013. [En línea]. Available: http://www.ticout.com/blog/2013/04/02/introduccion-a-hadoop-y-su-ecosistema/. [Último acceso: 27 11 2017].

[36] A. H. Y. Antonio Hernández Dominguez, «Revista Cubana de Ciencias Informáticas,» 01 06 2015. [En línea]. Available: http://scielo.sld.cu/pdf/rcci/v9n3/rcci04315.pdf. [Último acceso: 27 11 2017].

[37] U. D. o. Energy, «U.S. DOE Combined Heat and Power,» 31 12 2016. [En línea]. Available: https://doe.icfwebservices.com/chpdb/. [Último acceso: 03 11 2017].

ANEXOS

9.1 Manual de instalación Hadoop en CentOS

Para realizar la instalación de Hadoop en el sistema operativo CentOS, se puede seguir el siguiente manual de procedimiento:

Anexo 1 Manual Instalación Hadoop en CentOS

9.2 Instructivo de Programación de Ejemplo MapReduce

Para realizar la programación del ejemplo MapReduce, se puede utilizar el siguiente instructivo:

Anexo 2 Instructivo de Programación de Ejemplo MapReduce

9.3 Ejemplo Programa MapReduce

Para realizar el ejemplo de programa MapReduce, se utilizaron los siguientes códigos programados en Java utilizando Maven:

Anexo 3 Ejemplo Programa MapReduce