UNIVERSIDAD DE CHILE
FACULTAD DE CIENCIAS FÍSICAS Y MATEMÁTICAS
DEPARTAMENTO DE CIENCIAS DE LA COMPUTACIÓN

ANÁLISIS DE ARCHIVOS LOGS SEMI-ESTRUCTURADOS DE AMBIENTES WEB USANDO TECNOLOGÍAS BIG-DATA

TESIS PARA OPTAR AL GRADO DE MAGÍSTER EN TECNOLOGÍAS DE LA INFORMACIÓN

CÉSAR ALEXIS VILLALOBOS LUENGO

PROFESOR GUÍA:
AIDAN HOGAN

MIEMBROS DE LA COMISIÓN:

CLAUDIO GUTIÉRREZ GALLARDO

JORGE PÉREZ ROJAS

CECILIA REYES COVARRUBIAS

SANTIAGO DE CHILE

2016

Resumen

Actualmente el volumen de datos que las empresas generan es mucho más grande del que realmente pueden procesar, por ende existe un gran universo de información que se pierde implícito en estos datos. Este proyecto de tesis logró implementar tecnologías Big Data capaces de extraer información de estos grandes volúmenes de datos existentes en la organización y que no eran utilizados, de tal forma de transformarlos en valor para el negocio.

La empresa elegida para este proyecto se dedicada al pago de cotizaciones previsionales de forma electrónica por internet. Su función es ser el medio por el cual se recaudan las cotizaciones de los trabajadores del país. Cada una de estas cotizaciones es informada, rendida y publicada a las instituciones previsionales correspondientes (Mutuales, Cajas de Compensación, AFPs, etc.). Para realizar su función, la organización ha implementado a lo largo de sus 15 años una gran infraestructura de alto rendimiento orientada a servicios web. Actualmente esta arquitectura de servicios genera una gran cantidad de archivos logs que registran los sucesos de las distintas aplicaciones y portales web. Los archivos logs tienen la característica de poseer un gran tamaño y a la vez no tener una estructura rigurosamente definida. Esto ha causado que la organización no realice un eficiente procesamiento de estos datos, ya que las actuales tecnologías de bases de datos relaciones que posee no lo permiten. Por consiguiente, en este proyecto de tesis se buscó diseñar, desarrollar, implementar y validar métodos que sean capaces de procesar eficientemente estos archivos de logs con el objetivo de responder preguntas de negocio que entreguen valor a la compañía.

La tecnología Big Data utilizada fue Cloudera, la que se encuentra en el marco que la organización exige, como por ejemplo: Que tenga soporte en el país, que esté dentro de presupuesto del año, etc. De igual forma, Cloudera es líder en el mercado de soluciones Big Data de código abierto, lo cual entrega seguridad y confianza de estar trabajando sobre una herramienta de calidad. Los métodos desarrollados dentro de esta tecnología se basan en el framework de procesamiento MapReduce sobre un sistema de archivos distribuido HDFS.

Este proyecto de tesis probó que los métodos implementados tienen la capacidad de escalar horizontalmente a medida que se le agregan nodos de procesamiento a la arquitectura, de forma que la organización tenga la seguridad que en el futuro, cuando los archivos de logs tengan un mayor volumen o una mayor velocidad de generación, la arquitectura seguirá entregando el mismo o mejor rendimiento de procesamiento, todo dependerá del número de nodos que se decidan incorporar.

Agradecimientos

La vida se encuentra llena de retos. Cada reto que uno lo transforma en objetivo es la motivación personal a trabajar lo más duro posible en cumplirlo, por lo mismo, mi principal agradecimiento es a Dios por darme la fuerza y la dedicación para cumplir cada uno de los objetivos que me he propuesto a lo largo de mi vida. Muchas veces esa motivación personal por cumplir los objetivos implica quitar tiempo a la familia, pareja y amigos. Por ello agradezco muy especialmente a mis padres y a mi pareja, quienes supieron comprender y a la vez apoyarme en todo el transcurso del magíster, el cual fue el reto que decidí tomar hace ya tres años.

Un especial agradecimiento es a mi profesor guía Aidan Hogan, quien siempre tuvo el tiempo y la buena voluntad de responder mis preguntas, aceptar las reuniones de trabajo y responder mis e-mails, además de ser un constante apoyo en el desarrollo de mi proyecto de tesis durante el año que duró. De igual forma agradezco a los profesores Sergio Ochoa y Daniel Perovich quienes me guiaron en las distintas fases del proceso de titulación de la universidad.

No puedo dejar de mencionar a mi empresa, en la cual este diciembre del 2015 he cumplido cinco años, ya que me apoyó e incentivó en la realización de este proyecto de innovación, dándome todas facilidades en tiempo y recursos tecnológicos para que cumpliera los objetivos que el proyecto tenía.

Finalmente quiero despedir estos agradecimientos con la misma frase que escribí en mi carta de motivación para entrar al magíster, hace ya tres años, la cual aún identifica mi forma de pensar. La frase pertenece al filósofo Lucio Anneo Séneca y dice:

"*¡Estudia! No para saber una cosa más, sino para saberla mejor*".

Tabla de Contenido

Índice de tablas

Índice de Figuras.

1. Introducción

1.1. Contexto

Este proyecto se enmarca en una empresa del ámbito web, cuya principal función es ser el medio para el pago electrónico de las cotizaciones previsionales, así como entregar múltiples servicios a instituciones previsionales (Mutualidades, Isapres, Cajas de Compensaciones, AFPs, etc.).

La empresa se creó el año 2010 con el fin de mejorar el actual sistema de recaudación y distribución de las cotizaciones previsionales de los trabajadores dependientes de Chile. En ese entonces, el 100% de las cotizaciones se pagaba en planillas manuales de forma presencial en cada una de las instituciones, esto causaba gran cantidad de errores, por ejemplo: Errores de digitación, errores de pérdida de respaldo (debido al daño que sufrían las planillas), así como también pérdidas de dinero.

En estos quince años, la empresa ha tenido un importante ritmo de crecimiento, actualmente posee cerca de 200 trabajadores y atiende más del 90% de mercado del país, siendo líder en el ámbito de recaudación electrónica de cotizaciones previsionales. Además posee más de un millón de usuarios que operan en los diferentes portales administrando cerca de 500 mil empresas chilenas. Todo lo anterior conlleva a generar un volumen de más de tres millones de planillas mensuales, pagando las cotizaciones a cerca de 5 millones de trabajadores.

Finalmente la empresa debe gestionar cerca de 1.500 millones de dólares mensuales en lo que refiere a cotizaciones previsionales, con el fin de que luego, dichos dineros lleguen a las diferentes cuentas que cada trabajador posee en las distintas instituciones. Para lograr entregar el servicio de recaudación electrónica, la organización posee interconexión con todos los bancos e instituciones previsionales del país (por ejemplo, AFP, Isapres, Mutualidades, Cajas de Compensación, APV, etc.).

Debido al crecimiento que ha sufrido la empresa, hubo que restructurar la arquitectura que poseía en sus comienzos (la cual consistía en un par de servidores web), y llevarla hacia una infraestructura más potente y robusta, ya que el peak de pago se produce en un solo día. La actual arquitectura está compuesta de una granja web, de 12 servidores en paralelo que atienden al pool de usuarios. La infraestructura también posee balanceadores de carga potentes (actualmente se usa balanceadores f5[1]) y obviamente grandes bases de datos.

Por ser una empresa del ámbito de recaudación electrónica, ésta se ve afectada por todas las normativas, como por ejemplo: Las superintendencias de AFPs, de Isapres, etc. Lo anterior implica que ante casos legales relacionados a temas previsionales, la información que provee la organización sirve como medio de pruebas en los juzgados laborales.

[1] f5 es un ADC (*Application Delivery Controller*) utilizado para realizar balanceo de carga en sistemas transaccionales.

1.2. Oportunidad de mejora

Como se mencionó anteriormente, la empresa en los últimos años ha crecido fuertemente, pasando de unos pocos servidores, a una granja de servidores balanceados en carga transaccional que funcionan 24/7, los 365 días del año. De igual forma posee una infraestructura que da soporte a los servicios en dos centros de datos, uno principal y el otro de contingencia para asegurar la continuidad operacional.

El crecimiento no sólo se nota en una gran infraestructura que es actualmente requerida, sino también en el volumen de información que la empresa genera. Un tipo de información especial son los registros creados por las diferentes aplicaciones web (logs). Estos son generados a lo largo de toda la organización, y obligatoriamente necesarios para esclarecer dudas, o para ser usados como evidencia en situaciones legales específicas.

Estos archivos de logs, por su volumen y variedad, no son regularmente analizados por las áreas dentro de la empresa. Solamente en ciertos casos, estos logs son puntualmente revisados, como por ejemplo: Para la identificación de errores que son reportados por las áreas de negocio y/o peticiones de juzgados como evidencia ante juicios laborales.

El no analizar la información que existe en los diferentes logs provoca diversos problemas a la empresa:

- **Ser reactivos en la detección de errores:** Muchas veces los logs son revisados tardíamente, ya que primero los usuarios reportan problemas, luego estos reclamos son atendidos por distintas áreas, hasta que finalmente llegan al área de QA quienes solicitan, días después, el log para analizar el error que ocurrió.
- **Desconocimiento de casos borde, no probados por el área QA:** Muchos de los errores que se producen en el sitio se deben a casos de borde no probados por el área de QA. Esto suele ocurrir por falta de tiempo o de recursos para el proyecto. Dichos errores quedan registrados en los diferentes logs, pero debido a que no son analizados oportunamente, no pueden ser incluidos en los futuros proyectos, causando que siempre se cometan los mismos tipos de errores.
- **Comportamiento de usuarios:** En los logs queda el registro del comportamiento de los usuarios en el sitio, pero debido a que estos no son analizados, actualmente no se conoce muy bien el comportamiento de los usuarios.

Todos en la organización tienen conciencia de que existe gran información implícita en los logs que las diferentes aplicaciones generan. Sin embargo, nadie realmente ha podido analizar el gran volumen de información que hay en ellos, en gran medida porque estos logs no poseen una estructura definida que permita fácilmente insertarlos en base de datos relacionales para luego realizar análisis de ellos.

1.3. Objetivos de la tesis

El objetivo principal de este trabajo de tesis radica en diseñar e implementar métodos extensibles de procesamiento de información semi-estructurada, proveniente de la integración de múltiples logs de ambiente web, con el fin de obtener conocimiento relevante para la organización.

Los objetivos específicos que se desprenden del objetivo principal son los siguientes:

1.- Diseñar e implementar la capa de infraestructura que se soportará en las tecnologías Big Data.

2.- Diseñar e implementar los métodos de recolección, de análisis y procesamiento de los archivos de logs semi-estructurados.

3.- Procesar la información de tal forma de que sea posible consultarla en un formato conocido para el área "arquitectura y gestión de datos" (ejemplo: Formato SQL).

4.- Identificar qué 'preguntas de negocio'[2] pueden ser respondidas con los actuales datos existentes en los logs, y al menos responder una pregunta utilizando la infraestructura y los métodos de procesamiento de los datos semi-estructurados creados.

1.4. Propuesta de solución

La solución que se plantea radica en procesar, extraer y analizar la información de los múltiples archivos logs que posee la empresa, empleando para ello el uso de diferentes tecnologías de Big Data.

La elección de estas tecnologías para este caso específico reside en dos factores claves:

1. Los datos no poseen una estructura fija, por lo que cada archivo de logs es distinto entre sí, y no tiene una estructura estrictamente definida. Esto ha hecho que no sea posible subir ni procesar dichos archivos en bases de datos relacionales.
2. El volumen de generación de logs en el transcurso del mes no es estándar, existen un par de días de mucho volumen, así como también existen días de menor volumen. Por lo que es ideal tener una infraestructura fácilmente escalable que permita tanto el incremento como decremento de nodos de procesamiento.

Por lo anterior, el utilizar tecnologías de Big Data permitirá resolver los problemas que plantea el análisis de grandes volúmenes de datos semi-estructurados, que en este caso son archivos de logs. El uso de estas tecnologías entrega la flexibilidad de no preocuparnos tanto de la estructura que poseen los archivos, sino más bien del proceso de análisis. Del mismo modo, dicha tecnología tiene la capacidad de ser muy flexible en lo relacionado con la cantidad de nodos de procesamiento, ya que si se necesita mayor

[2] Preguntas de negocio son por ejemplo: ¿Cantidad de usuarios que no pueden realizar un flujo web? ¿Procesos de negocio que presentan lentitud? ¿Cantidad de errores transaccionales por periodo de tiempo?, etc.

poder de procesamiento, solamente se debe agregar nodos al sistema, y por el contrario, si se quiere disminuir el nivel de procesamiento, basta con quitar nodos, esto hace que la organización tenga siempre la cantidad real de infraestructura que necesita y no desperdicie recursos que no se están ocupando.

2. Marco teórico

2.1. Big Data

2.1.1. Qué es Big Data

Big Data es un concepto que ha nacido para describir la situación que ha venido aconteciendo en los últimos años, donde el volumen de datos que se genera electrónicamente ha ido incrementándose exponencialmente durante cada año, este incremento se debe en gran medida a que el costo de almacenamiento ha bajado y a la gran cantidad de fuentes de generación, como por ejemplo: Dispositivos móviles, logs de software, cámaras, micrófonos, RFID (radares), sensores inalámbricos, tráfico de redes, etc.). Y al mismo tiempo, estos nuevos conjuntos de datos tienden a ser en su gran mayoría datos no estructurados, lo que ha provocado que las clásicas arquitecturas no sean capaces de procesarlos eficientemente.

Para dimensionar el incremento del volumen de la información, la Tabla 1 muestra la base a nivel de bytes.

Tabla 1: Tabla de sistema de información en base byte.

Byte	B	$2^0 = 1$
KiloByte	Kb	$2^{10} = 1.024$
MegaByte	Mb	$2^{20} = 1.048.576$
GigaByte	Gb	$2^{30} = 1.073.741.824$
TeraByte	Tb	$2^{40} = 1.099.511.627.776$
PetaByte	Pb	$2^{50} = 1.125.899.906.842.624$
ExaByte	Eb	$2^{60} = 1.152.921.504.606.846.976$
ZettaByte	Zb	$2^{70} = 1.180.591.620.717.411.303.424$
YottaByte	Yb	$2^{80} = 1.208.925.819.614.629.174.706.176$

La Ilustración 1 presenta un gráfico de proyección realizada por el IDC "*International Data Corporation*" [1], se estima que al año 2020 serán 44 zettabytes los datos creados electrónicamente en la web (universo digital).

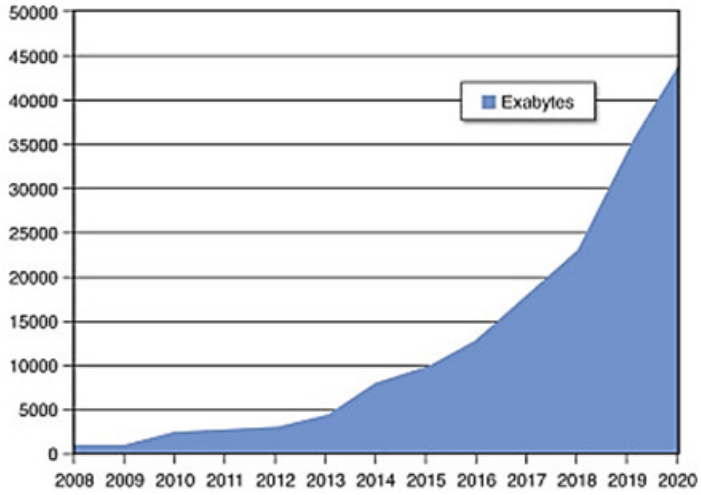


Ilustración 1: Gráfico de crecimiento del universo digital [1].

Según IDC un factor muy relevante se ha dado en los últimos años, donde la gran cantidad de datos que se generan son datos no estructurados proveniente de fuentes como: video, audio, social media, logs, etc.

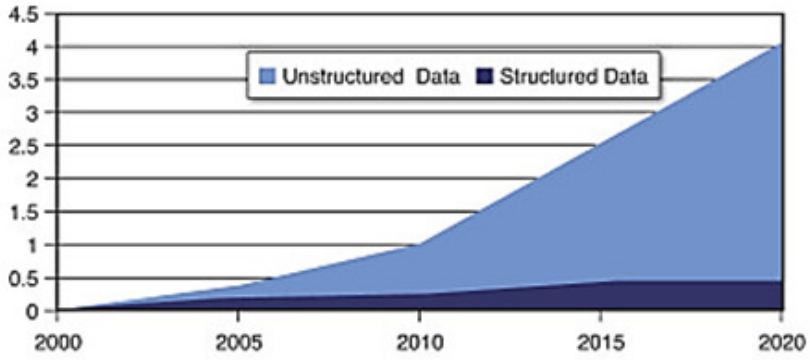


Ilustración 2: Gráfico de tipos de datos [1].

Como fue mencionado al inicio, esta nueva etapa de los datos ha generado la creación de un nuevo concepto “Big Data” que hace referencia a grandes conjuntos de datos que por su velocidad de generación y alta heterogeneidad, ha resultado muy difícil de manejar utilizando las clásicas arquitecturas de procesamiento de información.

Aunque existen muchas y diversas definiciones del concepto “Big Data”, se puede definir en tres características principales que lo componen: Volumen, velocidad, variedad.

- **Volumen**: Muchos factores han llevado a que actualmente se generaren una gran cantidad de datos de forma electrónica, se estima que la cantidad de datos en el mundo se está doblando cada dos años.

- **Velocidad**: La velocidad en la que los datos se crean, almacenan, analizan y visualizan ha pasado a ser un hecho muy relevante, por ende se necesitan formas de procesar dichos datos tan rápido como estos son generados para lograr obtener la información correcta en el momento preciso, ya que a veces dos minutos de procesamiento es demasiado tiempo (detección de fraudes, alerta de sensores, etc.).

- **Variedad**: En estos últimos años se ha pasado de los clásicos tipos de datos estructurados a nuevos y diversos tipos de datos, como por ejemplo: Texto de redes sociales, imágenes, audio, datos obtenidos directamente de sensores, video, correo, etc. La complejidad es realizar un eficiente procesamiento de todos estos tipos de datos de forma de entregar valor a la organización.

Ilustración 3: Las tres V (Volumen, Velocidad, Variedad) de Big Data.

En este último tiempo se ha ido expandiendo el concepto de Big Data agregando nuevas 'Vs' a las ya clásicas "Volumen", "Velocidad" y "Variedad", como por ejemplo "**Veracidad**", la cual hace referencia a qué tan creíble y verdadero es este gran volumen de información, o por ejemplo "**Valor**" hace referencia al valor que se obtiene del análisis. De cualquier modo, al ser una tecnología emergente es muy probable que sigan apareciendo nuevas definiciones en relación a tratar de explicar este nuevo concepto.

Según "*ISO/IEC JTC 1, Information Technology*" [2] esta nueva situación ha dado como resultado la creación de un nuevo paradigma, donde debido al volumen, velocidad y variedad de los datos, obliga a que la gestión de ellos sea un elemento principal en el diseño de las nuevas arquitecturas de sistemas. Fundamentalmente, el paradigma Big Data representa un cambio de los clásicos sistemas monolíticos de crecimiento vertical (procesadores, memoria y discos más rápidos) hacia una arquitectura de crecimiento horizontal. Este nuevo paradigma se basa en métodos masivos de procesamiento en paralelo MPP (*Massively Parallel Processing*), los cuales comenzaron a desarrollarse en los años setenta, pero ahora aplicados a esta nueva situación de grandes conjuntos distribuidos de datos.

"ISO/IEC JTC 1, Information technology" [2] define:

> ***Paradigma Big Data:*** *Consiste de la distribución de los datos a través de recursos independientes y horizontalmente acoplados, con el objetivo de lograr la escalabilidad necesaria para un eficiente procesamiento de grandes conjuntos de datos.*

Este paradigma tiene una idea básica "Mover el procesamiento a los datos, y no los datos al procesamiento", esta expresión viene del inglés: *"Moving the processing to the data, not the data to the processing"*, la cual hace referencia a que es más eficiente llevar el procesamiento a cada nodo donde están alojados los conjuntos de datos, que transportar dichos datos a un nodo central de procesamiento.

Finalmente se puede entregar la definición de Big Data expuesta por la ISO/IEC [2]:

> ***Big Data:*** *Es un conjunto de datos con características (ejemplo: volumen, velocidad, variedad, etc.) que para un particular dominio de problema en un momento dado, no puede ser eficientemente procesado usando las actuales/existentes/establecidas/tradicionales tecnologías y técnicas, con la intención de extraer valor.*

2.1.2. Tecnologías Big Data

Actualmente existe una amplia variedad de técnicas y tecnologías que se han ido desarrollando para almacenar, procesar, y analizar grandes volúmenes de datos. Debido a que Big Data aún es un campo nuevo de investigación, la mayoría de estas tecnologías aún sigue en desarrollo, por lo cual es muy probable en los siguientes años surjan nuevas o se unan algunas de las ya existentes.

Las actuales tecnologías Big Data se pueden categorizar en los siguientes tipos:

- Sistemas de almacenamiento distribuido (*Distributed File System*).

- Tipos de bases de datos:
  - NoSQL Databases.
  - NewSQL Databases.

- Tipos de procesamiento:
  - Distributed Programming.
  - Stream Processing.
  - Graph Processing.
  - SQL Processing.

2.1.2.1. Sistemas de almacenamiento distribuido (*Distributed File System*)

A continuación se presentan los dos ejemplos representativos de sistemas de almacenamiento distribuido.

- **Google File System:** El sistema de archivos de Google "***GFS***" [3] , es un sistema de archivos distribuido desarrollado por Google (propietario), GFS fue presentado por primera vez el año 2003 [3] donde Google exponía su forma de almacenar grandes volúmenes de datos de forma distribuida sobre hardware básico y obteniendo un gran rendimiento. GFS se basa en un nodo maestro "Master" y un conjunto de nodos de datos llamados "chunkservers". Los archivos que se cargan en GFS de dividen en bloques de igual tamaño que se distribuye a través de los "chunkservers".

- **Apache Hadoop Distributed File System (HDFS):** HDFS [4] es un sistema de archivos distribuido escrito en Java para el framework Hadoop, éste se basa en el paper de Google donde explicaba su GFS (Google File System) [3]. HDFS tiene la misma idea que GFS, donde existe un nodo maestro "NameNode" y un conjunto de nodos de datos "DataNodes". Los archivos que se cargan en HDFS son divididos en bloques que son distribuidos a lo largo de los "DataNodes".

  Una mayor descripción es presentada en la sección 2.2.1.

2.1.2.2. Bases de datos NoSQL

Existe una gran variedad de Bases NoSQL, a continuación se mencionan algunos ejemplos más representativos de su tipo.

- **Google BigTable*:*** BigTable [5] es un proyecto propietario de Google basado en un sistema de base de datos NoSQL para almacenar datos estructurados construido sobre el sistema de archivos distribuido de Google GFS. BigTable es presentado de forma detalla en el paper de Google publicado el año 2006 [5].

- **Apache HBase:** Apache HBase [6] es un proyecto de código abierto basado en un sistema de base de datos NoSQL orientado de modo "Columnar". Este sistema fue diseñado para estar sobre el sistema de archivos distribuido de Hadoop "HDFS". Apache HBase se basa en el paper publicado por Google el año 2006 donde exponía Google BigTable [5]. Ese mismo año HBase comenzó a desarrollarse, el año 2008 HBase se convirtió en un sub-proyecto de Hadoop y ya el año 2010 HBase es un proyecto Apache de primera línea.

- **Amazon DynamoDB:** Amazon DynamoDB [7] es un proyecto propietario de Amazon basado en un sistema de base de datos NoSQL orientado a clave-valor. DynamoDB está diseñado en una arquitectura de nodos multi-maestros y utiliza

replicación sincrónica de datos a través de los múltiples nodos que conforman el cluster de operación.

- **MongoDB**: MongoDB [8] es un proyecto de código abierto basado en un sistema de base de datos NoSQL orientado a documentos. MongoDB es considerado como una de las más populares bases de datos NoSQL. Su arquitectura se basa en el modo Maestro-Esclavo donde la función del rol "Maestro" es realizar las operaciones de lectura y escritura, mientras los "Esclavos" se distribuyen los datos provenientes del maestro con el fin de lectura. Actualmente los nodos "Esclavos" no poseen escrituras sobre los conjuntos de datos, pero pueden sustituir al nodo "Maestro" en caso de que este último falle. MongoDB en vez de almacenar la información en tablas como los sistemas de bases de datos relaciones, las almacenas en documentos usando formato binario del estilo JSON llamado BSON "*Binary JSON*".

- **Apache Cassandra:** Apache Cassandra [9], en sus inicios desarrollado por Facebook, es un proyecto de código abierto basado en un sistema de base de datos NoSQL distribuida orientado al modelo de almacenamiento clave-valor. Actualmente Apache Cassandra posee una arquitectura sin nodo "Maestro", en esta arquitectura todos los nodos del cluster actúan de la misma manera, los datos se distribuyen automáticamente en forma de "anillo". Cassandra entrega una interfaz de comunicación basada en CQL (*Cassandra Query Language*) cuyo lenguaje es muy parecido a SQL con la diferencia de que Cassandra no soporta joins o subqueries.

2.1.2.3. Bases de datos NewSQL

Actualmente existen varios motores NewSQL siendo el más representativo Spanner, descrito a continuación.

- **Google Spanner:** Spanner [10] es un proyecto propietario de Google basado en un sistema de base de datos NewSQL globalmente distribuido y sincrónicamente replicado. Google Spanner se basa en la capacidad de tener consistentes transacciones distribuidas a nivel global. Este proyecto se dio a conocer en el paper publicado por Google el año 2012 [10] .

2.1.2.4. Procesamiento del tipo "*Distributed Programming*"

A continuación se describen proyectos representativos de procesamiento distribuido.

- **MapReduce:** MapReduce [11] es un modelo de programación distribuida y paralela diseñado para procesar grandes conjuntos de datos. MapReduce consta de dos simples funciones, La función "Map" que procesa valores del tipo clave-

valor produciendo un conjunto de valores intermedios donde todos poseen la misma clave, luego se aplica la función “Reduce” teniendo como entrada este conjunto de valores de la forma de clave-<valores>. MapReduce fue introducido en el paper publicado por Google el año 2004 [11], de este paper se adaptó el modelo MapReduce al proyecto Hadoop [12].
Una mayor descripción es presentada en la sección 2.2.2

- **Apache Pig:** Apache Pig [13] es una plataforma para analizar grandes conjuntos de datos. Esta plataforma consiste en un lenguaje ‘scripting’ de alto nivel basado en expresiones, con el objetivo de describir operaciones como por ejemplo: Lectura, filtros, transformaciones, uniones, etc. Una vez compilados estos programas generan procesos del tipo MapReduce, los cuales son ejecutados sobre el conjunto de datos.

- **Apache Spark:** Apache Spark [14] es un proyecto de código abierto desarrollado sobre lenguaje Scala, se basa en un framework de procesamiento distribuido, originalmente fue desarrollado en la Universidad de California (Berkeley) en el año 2009. Apache Spark se diferencia de otros framework de procesamiento distribuido como MapReduce por tener múltiples etapas de procesamiento principalmente en memoria. Spark necesita tener un nodo maestro y un sistema de almacenamiento distribuido. Actualmente puede operar sobre diversos almacenamientos como por ejemplo Cassandra, OpenStack Swift, Amazon S3, Kud.

2.1.2.5. Procesamiento del tipo “*Stream Processing*”

A continuación se describen Storm y MillWheel, los dos ejemplos más representativos de este tipo de procesamiento.

- **Apache Storm:** Apache Storm [15] es un proyecto de código abierto basado en un framework de procesamiento distribuido para streams (flujos de datos). Apache Storm tiene el objetivo de recuperar en tiempo real y de múltiples fuentes streams de datos, por ejemplo: datos provenientes de sensores, de redes sociales, etc. Apache Storm se compone principalmente de dos partes: “Spout” que está encargada de recoger estos flujos de datos de entrada, y de “bolt” encargada de procesar y transformar los datos.

- **Google MillWheel:** MillWheel [16] es un proyecto propietario de Google. Esta tecnología fue presentada en el paper publicado por Google el año 2013 donde se exponía un framework de procesamiento de streams (flujos de datos) a una escala de internet. MillWheel a grandes rasgos se basa en un grafo de transformación definido a nivel de usuario sobre un conjunto de datos de entrada produciendo datos de salida.

2.1.2.6. Procesamiento del tipo "*Graph Processing*"

A continuación se presentan Pregel y Giraph, dos tecnologías representanticas de "*Graph Processing".*

- **Google Pregel:** Google Pregel [17] es un proyecto propietario de Google basado en el procesamiento de grafos a un nivel de datos de internet. Esta tecnología fue publicada por Google en su paper del año 2010 [17]. Básicamente Google Pregel muestra una ejecución de trabajo en la que la entrada es un grafo, donde cada grafo posee un id único, un conjunto de aristas salientes y una función asociada, finalmente se obtiene un grafo con la solución.

- **Apache Giraph:** Apache Giraph [18] es un proyecto de código abierto basado en procesamiento de grafos diseñado para una alta escalabilidad. Actualmente esta tecnología es usada por Facebook para analizar el grafo formado por los usuarios y sus conexiones. Giraph se basa en el paper de Google del año 2010 donde se presentaba Pregel [17].

2.1.2.7. Procesamiento del tipo "SQL Processing"

A continuación se presentan Hive e Impala, dos tecnologías del tipo "*SQL processing*".

- **Apache Hive:** Apache Hive [19] es una tecnología desarrollada por Facebook que facilita las consultas y permite gestionar grandes conjuntos de datos que residen en un sistema de almacenamiento distribuido. Hive provee un mecanismo para estructurar los datos y a la vez una forma tipo-SQL para consultar dichos datos, este lenguaje es llamado HiveQL. La ejecución de las consultas se basan, al igual que Apache Pig en la creación de diferentes procesos del tipo MapReduce que son aplicados sobre los conjuntos de datos.

- **Cloudera Impala:** Cloudera Impala [20] es un proyecto propietario de Cloudera Inc., se basa en un motor de consultas SQL para el procesamiento masivo en paralelo de grandes volúmenes de datos que están almacenados en el sistema de archivos distribuidos de Hadoop (HDFS). Impala se integra con Apache Hive utilizando su base de datos de metadatos, permitiendo una integración con todo el ecosistema de Hadoop, de esta forma se puede utilizar las funcionalidades de Impala con el lenguaje usado por clientes Hive (HiveQL).

2.1.3. Tipos de datos analizados en Big Data

Según Dataversity "*Data Education for Business and IT Professionals*" en su estudio [21], entre los tipos de datos que se suelen analizar en Big Data están.

1. **Web y Redes sociales:** Corresponde a la información que es obtenida de la web y redes sociales como Facebook, Instagram, Twitter, etc.

2. **Machine-to-machine (m2m):** Este tipo se refiere a tecnologías que permiten conectarse a otros tipos de dispositivos, M2M utiliza dispositivos como sensores, medidores que capturan algún tipo de evento, como por ejemplo velocidad, temperatura, presión, etc., los cuales los transforman en datos que son almacenados electrónicamente.

3. **Big Transaction Data:** Este tipo corresponde a registros de telecomunicaciones CDR (registros detallados de llamados). Estos tipos de datos están tanto en formato estructurados como semi-estructurados.

4. **Biometrics:** Información de carácter biométrico, como por ejemplo: huellas digitales, escaneo de retinas, etc.

5. **Human Generated:** Información proveniente de la interacción humana, como por ejemplo: Llamadas de un Call Center, documentos electrónicos, estudios médicos, etc.

2.2. Hadoop

Apache Hadoop es un proyecto creado por Doug Cutting[3], y se basa en el proyecto de Google File System (GFS) presentado por Google el año 2003 [3]. De igual forma se basa en el paradigma de programación MapReduce introducido por Google del año 2004 [11].

Hadoop nació con un supuesto muy simple [22]:

"Moving Computation is Cheaper than Moving Data"

El supuesto anterior hace referencia a que en esta era de grandes conjuntos de datos, es mucho más eficiente para las aplicaciones realizar el procesamiento cerca de donde están los datos que operan sobre ella, ya que esto minimiza la congestión de la red y aumenta el rendimiento del sistema.

Hadoop está divido en tres componentes principales:

- Hadoop Distributed File System (HDFS)
- Hadoop MapReduce
- Hadoop Common

[3] El nombre Hadoop no es un acrónimo, sino que era el nombre que el hijo de Doug Cutting le dio a su elefante amarillo de peluche.

2.2.1. Hadoop Distributed File System (HDFS)

El sistema de archivos distribuido de Hadoop HDFS (*'The Hadoop Distributed File System')* [4], ha sido diseñado para ejecutarse con hardware básico (*Commodity Hardware*).

Las principales características de HDFS son:

- **Tolerancia a fallas (*Hardware Failure*):** La falla de hardware se debe tomar como la norma y no como una excepción, por ende las situaciones donde partes del hardware fallan dentro de una arquitectura de sistema debe ser tratado como un comportamiento esperado en ambientes distribuidos.
- **Acceso a datos de forma continua (*Streaming Data Access*):** Para HDFS es necesario y prioritario tener acceso vía Streaming (flujo de datos continuo) a los archivos dentro del sistema distribuido.
- **Grandes conjuntos de datos (*Large Data Sets*):** HDFS fue creado para almacenar datos provenientes de la web, por ende es fundamental que maneje grandes volúmenes de datos de forma eficiente.
- **Modelo de coherencia simple (*Simple Coherency Model*):** El modelo buscado por HDFS es de una coherencia simple, es decir, las aplicaciones pueden escribir una sola vez, pero leer muchas veces sobre el conjunto de datos.

HDFS posee una arquitectura de Maestro/Esclavo. Cada cluster HDFS consiste en un único NameNode que administra el espacio de datos (Namespace) y regula el acceso a los archivos para los clientes. En el mismo modo, existen varios DataNode, generalmente uno DataNode por nodo, su función es administrar el almacenamiento que existe en ese específico nodo (lectura, escritura, espacio, etc.). La Ilustración 4 muestra la arquitectura de HDFS [23].

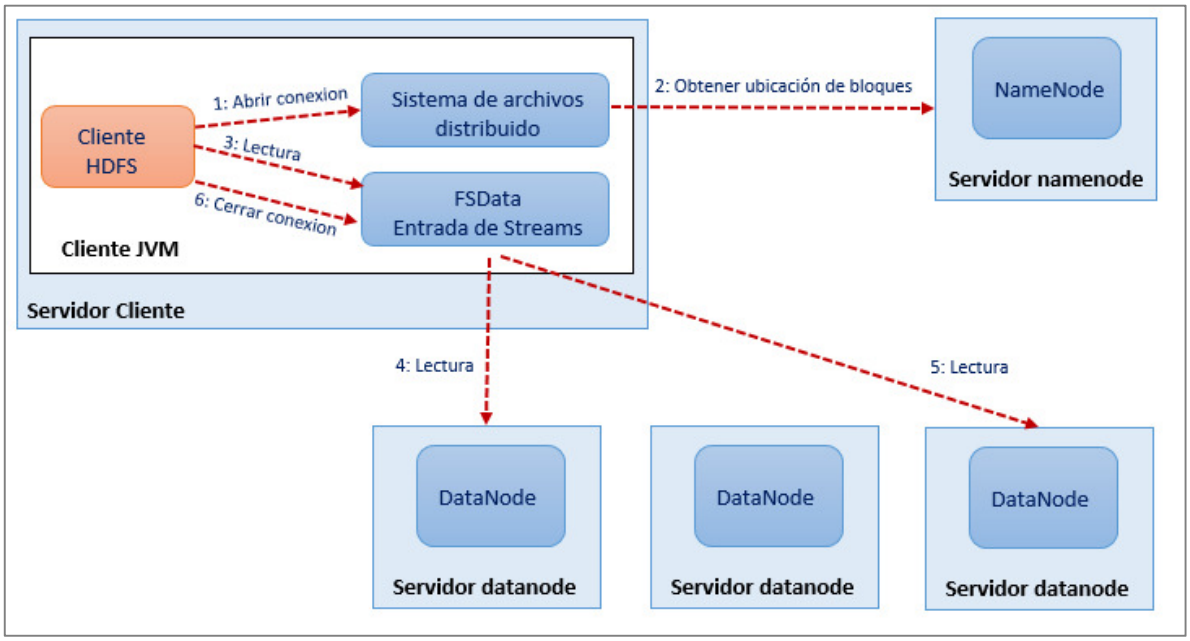


Ilustración 4: Diagrama de la arquitectura de HDFS.

Cuando un archivo es cargado al cluster HDFS es dividido en bloques de 64Mb de tamaño, estos bloques son distribuidos a través de los nodos del cluster. De igual forma mediante la distribución se realiza la replicación. El factor de réplica es configurable a nivel de usuario, pero generalmente se estable en tres, permitiendo que HDFS sea tolerante a fallas. La Ilustración 5 describe la forma de carga de un archivo a HDFS [23].

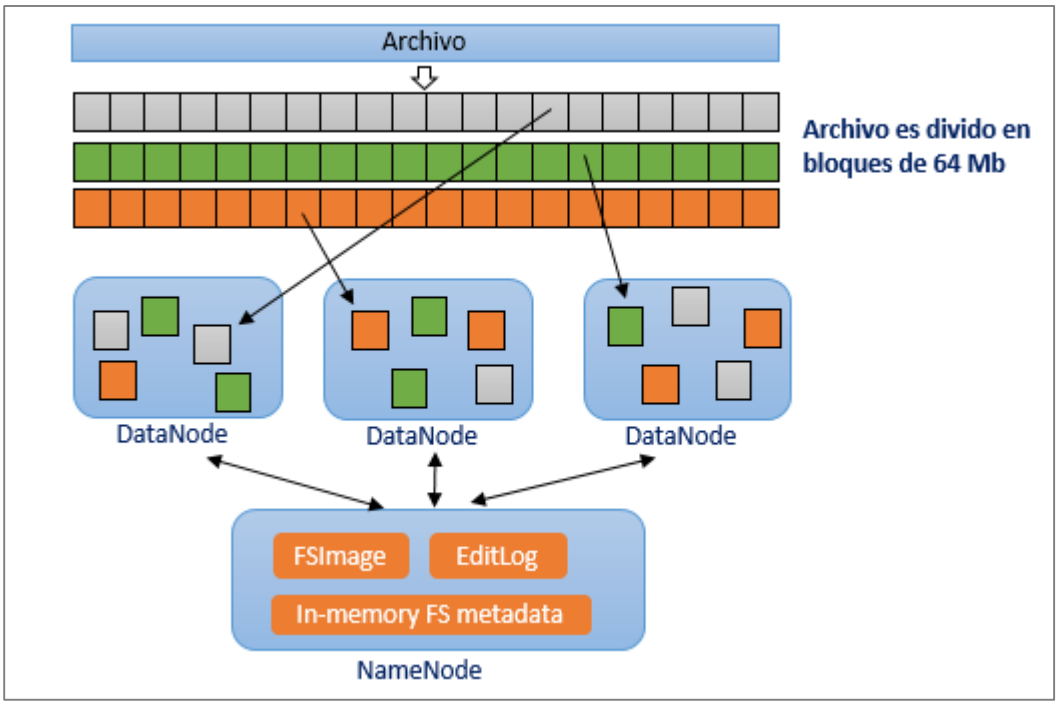


Ilustración 5: Diagrama de carga de archivo a Hadoop HDFS.

2.2.2. Hadoop MapReduce

MapReduce es un framework para procesar grandes volúmenes de datos en forma paralela sobre ambientes distribuidos. Fue introducido por primera vez en una publicación de Google en el año 2004 [11], donde se presentó cómo Google realiza la partición, procesamiento y agregación de los grandes conjunto de datos de una forma óptima, sin la necesidad de utilizar súper-computadores o servidores dedicados. De igual forma Google planteaba que este modelo era fácilmente escalable utilizando Hardware básico.

Tiempo después de la liberación de MapReduce por parte de Google, Doug Cutting creó su propia versión de este modelo, la cual incorporó al proyecto de código abierto Hadoop.

El modelo de MapReduce simplifica a gran nivel el procesar paralelamente datos de forma distribuida, ya que sólo hay que definir dos simples funciones: “Map” y “Reduce”, las cuales son aplicadas al conjunto de datos de entrada. Por otro lado, es el framework quien se encarga de las tareas de particionar los datos de entrada, la intercomunicación y sincronización entre los subprocesos, tratamiento de caídas, unión de los conjuntos de datos, etc.

A continuación se describen las principales funciones del modelo de MapReduce.

Función Map: Se aplica en paralelo sobre cada dupla clave valor (k1,v1) generando una lista intermedia del tipo:

(k1,v1) → List (k2, v2)

Función Shuffle and Sort: Se juntan todas las claves intermedias ordenándolas según su clave, para luego distribuir cada conjunto de clave y sus valores por el cluster de nodos de procesamiento.

Función Reduce: Se aplica en paralelo sobre cada grupo de duplas con la misma clave intermedia (k2), tomando los valores (v2) para luego generar una salida de valores finales.

(k2, list(v2)) → list(v3)

La Ilustración 6 describe el flujo de procesamiento de MapReduce [24].

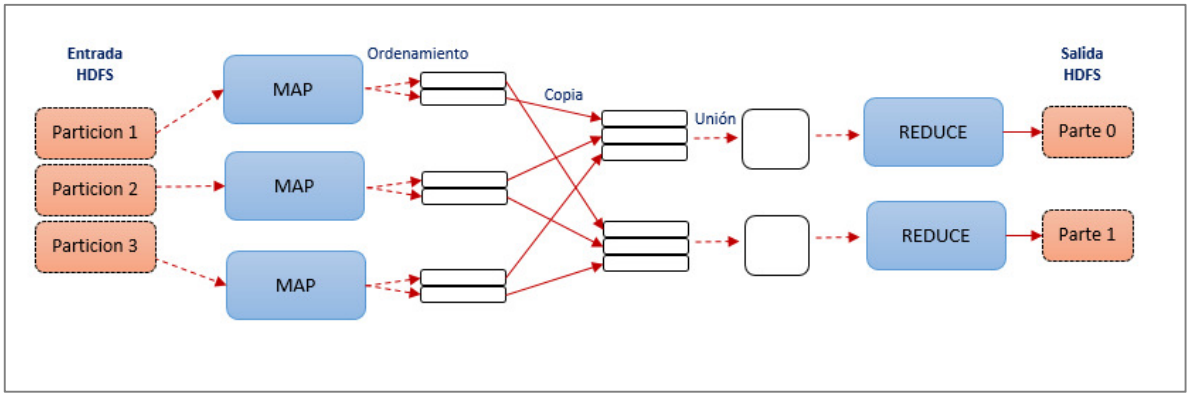


Ilustración 6: Diagrama del flujo de procesamiento de MapReduce.

2.2.3. Hadoop Common

"Hadoop Common Components" es un conjunto de componentes de librerías que soportan los diversos módulos de Hadoop, así como un conjunto de sub-proyectos propios de Apache Hadoop.

3. Esquema de Trabajo

Por ser una tecnología emergente, aún no está definido un ciclo de vida o una metodología definitiva para un proyecto Big Data, sino que cada fabricante expone su propia metodología.

Para este proyecto en particular, se utilizó el esquema de trabajo presentado en la Ilustración 7 [4], el que representa las fases realizadas para cumplir con el objetivo final del proyecto.

Ilustración 7: Diagrama del esquema de trabajo.

I.- Etapa de Exploración: Se basa en entender el entorno de la organización y ver qué datos dentro de los existentes son útiles para obtener información de valor.

- **Escenario actual:** Representa el contexto actual de información dentro de la organización, como por ejemplo: La arquitectura de servicios y los datos que fluyen entre ellos.
- **Oportunidades:** Dentro del contexto actual, revisar la existencia de oportunidades de utilizar tecnologías Big Data, de tal forma de entregar valor a la organización.

II.- Etapa de Definición y Alcance: Una vez analizada la situación actual y encontrada las oportunidades donde la aplicación de tecnologías Big Data puede resultar relevante para la organización, se deben definir y especificar los casos de uso que se quieren tratar de responder.

[4] Este esquema de trabajo es particular de este específico proyecto, basado en las fases que hubo que realizar para completarlo.

- **Definir y especificar casos de uso:** Para que el proceso de Big Data tenga un objetivo claro, se deben determinar qué casos de uso se responderán aplicando tecnologías Big Data.

III.- Etapa de Solución Técnica: Una vez definidos los casos de uso, se debe dar paso a la elección de qué tecnología Big Data es la más adecuada para el problema específico de la organización, así como la arquitectura de solución.

- **Tecnología Big Data:** Definir qué tipo de tecnología Big Data es la más adecuada para la organización.
- **Arquitectura de solución:** Se debe diseñar cuál es la arquitectura de la solución que se implementará dentro de la compañía y cómo ésta dará solución al problema plateado.
- **Métodos Desarrollados:** Se realiza la especificación de cuáles fueron los métodos desarrollados para realizar el procesamiento de información.

IV.- Etapa de Evaluación: Se realiza una evaluación y validación de la solución entregada, y cómo ésta da respuesta a los requisitos iniciales presentados en los casos de uso.

- **Evaluación de los resultados*:*** Se debe revisar si los resultados obtenidos por la solución dan cumplimiento al conjunto de casos de usos expuestos al inicio.

- **Evaluación de solución**: Este punto representa la escalabilidad de la solución, es decir, cómo se comporta dicha solución a medida que aumentan el tamaño de los archivos o el número de nodos dentro del sistema distribuido de procesamiento.

4. Etapa de Exploración

El propósito de esta etapa es realizar el análisis del actual estado que posee la organización y ver las oportunidades donde el uso de tecnologías Big Data pueda ser un elemento de valor dentro de la organización.

4.1. Escenario Actual

La organización se basa en una arquitectura compuesta por un conjunto de servicios web que están instalados en una granja de servidores expuestos a internet mediante una capa de seguridad. Existen servidores dedicados a atender público general y otros a ejecutivos internos de la empresa, quienes ingresan mediante la intranet corporativa.

Todo lo que se realiza en el sitio web se graba en distintos archivos de logs, cada archivo de log permanece en el propio servidor local donde reside el servicio web, una vez al día todos los archivos logs se envían a un servidor concentrador (CDL), el cual es un servidor dedicado para almacenar logs de todas las aplicaciones de la organización y con ello facilitar el acceso y búsqueda para las personas de la compañía.

Un esquema de la arquitectura de servicio se muestra en la Ilustración 8.

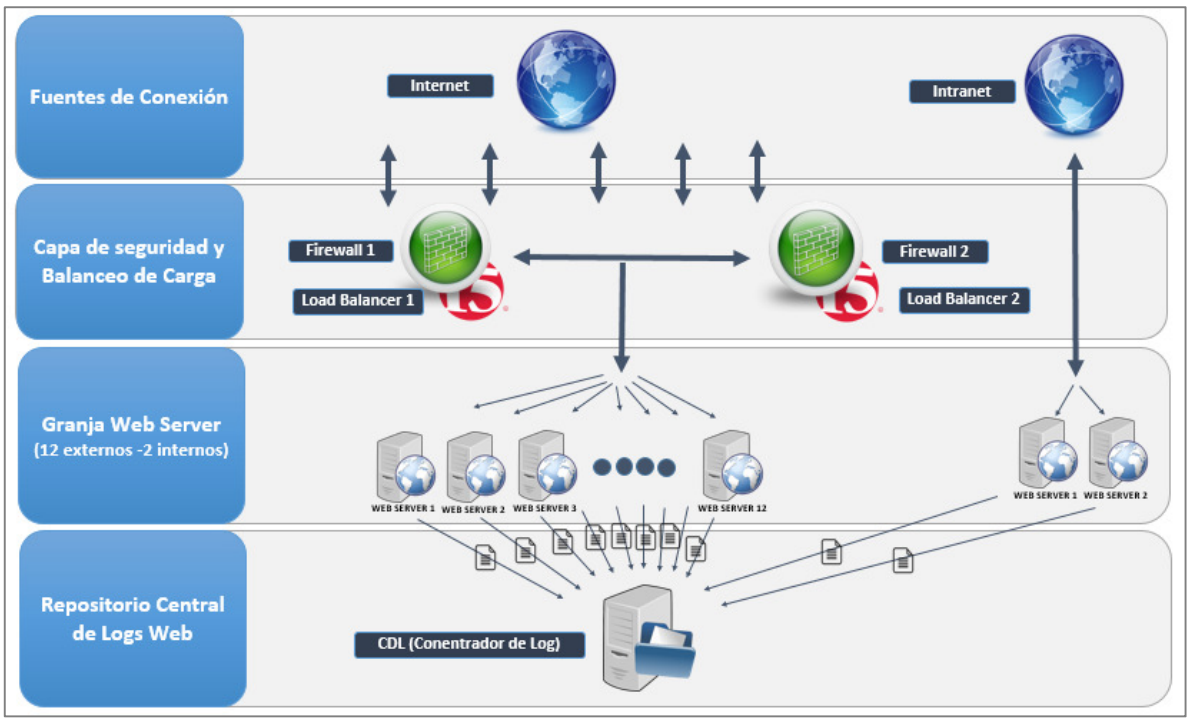


Ilustración 8: Diagrama de la arquitectura actual de la organización.

Existen cerca de treinta logs diferentes en esta granja de servidores web, cada log almacena información referente a servicios web específicos, y debido a la alta transaccionalidad que posee el sitio, ha hecho que la cantidad de líneas de logs que se almacenan mensualmente lleguen a los 1500 millones en promedio.

El gráfico de la Ilustración 9 muestra una visión global del volumen de datos que mensualmente se genera entre todos los archivos logs que posee la organización.

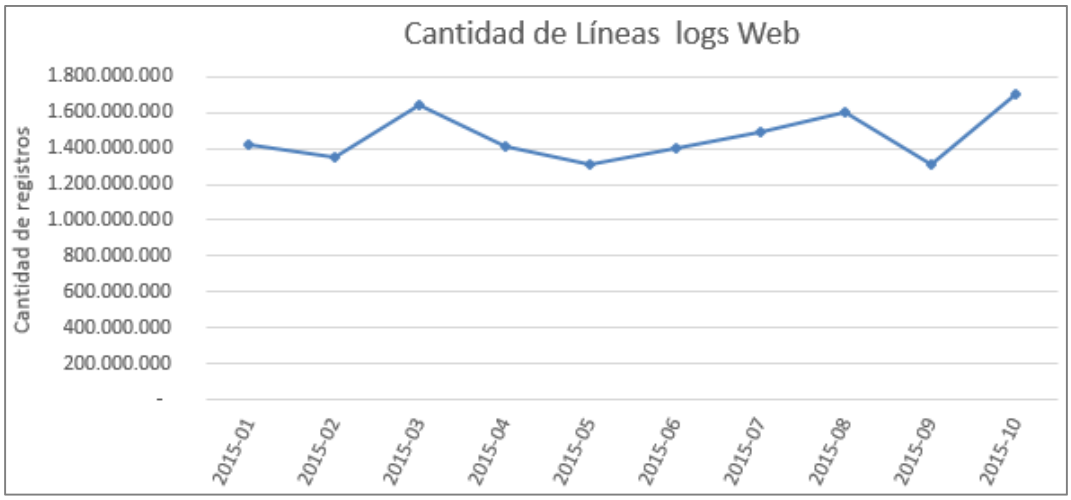


Ilustración 9: Gráfico de líneas de log de ambiente web de la organización.

Dentro del universo de logs que posee la organización en lo relacionado a servicios web, se puede encontrar los siguientes tipos:

- **Logs de "Servicios web":** Corresponden a los distintos servicios web expuestos por la compañía, entre estos se encuentran: Log del portal web de empresas (wEmpresasLog), log del portal web de personas (wPersonasLog), etc.

- **Logs de "Apache":** Corresponden a aquellos producidos por el servicio httpd[5] dentro de los servidores, entre estos se encuentran: Log de errores de Apache (ErrorLog), Log de peticiones de Apache (AccessLog), etc.

- **Logs de "Application Server":** Corresponden a los logs del servidor de aplicaciones Jboss EAP, entre estos se encuentran: Log de eventos de Jboss EAP 6.2 (serverlog), log del framework de Java JSF (JSFLog), etc.

- **Logs de "Botón/Canal de pago":** Corresponden a los desarrollados para revisar y registrar las transacciones de los botones de pagos que la organización posee con los distintos bancos y tarjetas de comercio, entre estos se encuentran: Log de del banco BCI (log botonPagoBCI), Log del banco Santander Santiago (log botonBcoStgo), etc.

- **Logs de "Monitoreo y configuración":** Corresponden a logs que revisan la disponibilidad de los servicios web, así como logs de configuración, entre estos se encuentran: Logs de caché de base de datos asociado a servicios de usuarios (log PrivUsuariosPreviredConCacheDAO), logs de eventos del monitoreo de aplicación (log MonitoreoAplicativo), etc.

[5] httpd es un servidor http de código abierto de Apache, siendo el más popular desde abril 1996.

- **Logs de "Aplicativos Específicos":** Corresponden a aquellos que son diseñados para registrar eventos específicos de ciertas aplicaciones, entre estos se encuentran: Log de cambio de password (log getChangePassword), log de validación de nóminas (log validaNominas), etc.

El gráfico de la Ilustración 10 muestra la distribución de líneas de logs según tipo.

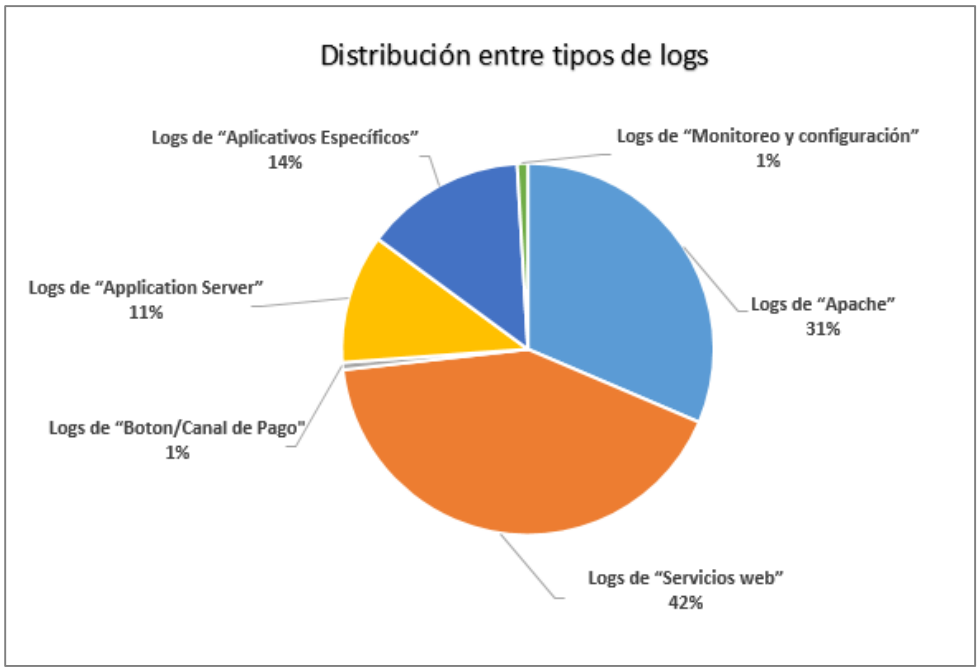


Ilustración 10: Gráfico de distribución entre los tipos de log de ambiente web.

4.2. Oportunidad

En la organización no hay forma de procesar la gran cantidad de archivos logs, ya que los actuales métodos existentes (bases de datos relacionales) no dan la capacidad para realizar eficientemente el procesamiento de este gran volumen de datos, los que tienden a ser en su mayoría datos semi-estructurados. Por ello, una buena alternativa es utilizar nuevas tecnologías capaces de lidiar con dichos tipos de datos, como Big Data.

Sería de gran ayuda para la organización procesar los archivos logs que se generan en el sitio web mediante dichas tecnologías, ya que permitirá extraer información que actualmente está implícita, mejorando con ello los procesos de calidad con una temprana detección de errores y poseer un mayor conocimiento de los usuarios mediante el análisis de su comportamiento.

5. Etapa de Definición y Alcance

El objetivo de esta etapa es especificar claramente el alcance que tendrá el proyecto Big Data dentro de la organización. Lo anterior hace referencia a definir los casos de uso en los que se trabajará, así como identificar qué datos (archivos de logs) serán los que se utilizarán para extraer la información.

5.1. Definición de Casos de Usos

Se especificaron dos casos de uso, uno para la trazabilidad de usuarios dentro del sitio web de la organización, y otro para detectar los errores.

5.1.1. Trazabilidad de Usuarios

Este caso de uso tiene el objetivo de generar el detalle de la trazabilidad de todos los usuarios que operan el sitio web de la organización.

Entre la información que se necita obtener se puede destacar la siguiente:

- **IP*:*** IP del cliente del usuario por donde se conectó al sitio web.
- **Fecha y hora*:*** Fecha y hora del registro de la acción.
- **Navegador web:** Navegador web (Chrome, IE, Mozilla) que estaba usando cuando ingresó al sitio, así como la versión de éste.
- **Identidad de usuario:** Identidad del usuario que estaba utilizando en el sitio web. Por identidad se hace referencia al Rut de la persona.
- **Dispositivo*:*** Qué tipo de dispositivo (computador personal, móvil, tablet) utilizaba el usuario cuando operaba en el sitio web de la organización.
- **Acción*:*** Acciones que el usuario realizó dentro del sitio web (Login al sitio, pagar planillas, crear nóminas, etc.)
- **Servidor:** Servidor web de la granja que atendió al usuario.

Las preguntas que se desea responder son las siguientes:

- Dado un Rut de un usuario, tener el detalle de la trazabilidad de sus acciones dentro del sitio web de la organización.
- Detalle de los usuarios que ingresan al sitio web por una determinada versión del navegador web.
- Identificar la carga de usuarios en la granja de servidores que posee la organización.
- Revisar flujos dentro del sitio web que puedan presentar dificultad para los usuarios.

5.1.2. Detección de Errores

Este caso de uso tiene como objetivo tener una visión preventiva de los errores que suceden en el sitio web, así como el detalle de los usuarios a quienes estos errores pudiesen afectar.

Entre la información que se necesita obtener se puede destacar la siguiente:

- **IP:** IP cliente donde se detectó el error (dato opcional, no siempre en errores está presente este dato).
- **Fecha y hora:** Fecha y hora del registro del error.
- **Aplicación web:** Identificar la aplicación web que registró el error.
- **Identidad de usuario:** Identidad del usuario que estaba utilizando el sitio web al momento que sucedió el error (dato opcional, no siempre en errores está presente este dato). Por identidad se hace referencia al Rut de la persona.
- **Servidor:** Servidor donde se produjo el error.
- **Detalle de error:** Detalle del error que se encuentra en el archivo de log.

Las preguntas que se desea responder son las siguientes:

- Cantidad de errores más presente en el sitio web de la organización.
- Revisar un error específico en la granja de servidores, entregando el comportamiento de este error en el tiempo.
- Detalle de personas con mayor número de errores.

5.2. Definición de logs a utilizar

El sitio web de la organización se divide en roles, existe el rol de Personas y Empresas. Las Personas son quienes realizan el pago de las cotizaciones previsionales de forma independiente: Trabajadores a honorarios, trabajadores de casa particular (empleadas domésticas, nanas, jardineros, etc.), o personas que quieran pagar ahorro directos a su sistema de previsión (aporte voluntario APV). Por otro lado, las Empresas son quienes pagan las cotizaciones previsionales de sus trabajadores mediante el sitio web (cotizaciones de AFP, ISAPRE, Mutual, etc.). Para cada tipo de usuario (Empresa o Persona) se registra toda la actividad que realiza dentro del sitio web de la organización.

A continuación se especifican los logs que se utilizarán para el procesamiento, de igual forma se muestra un extracto[6] de cada log para un mejor entendimiento.

wPortalLog: Log que registra cuándo un usuario está trabajando en el portal genérico del sitio, donde aún no ha elegido cuál rol usará (rol Empresa, rol Persona).

```
2015-10-15 09:58:01,887 INFO  [com.previred.wportal] http-8180-9  SIN SESION  SIN USUARIO IP: 10.0.1.106 - Ejecutando programa prglogin
2015-10-15 09:58:01,888 INFO  [com.previred.wportal] http-8180-9  SIN SESION  SIN USUARIO Invocando servicio srvlogin
2015-10-15 09:58:01,891 INFO  [com.previred.wportal] http-8180-9  SIN SESION  SIN USUARIO Usando clase de negocios: com.previred.privilegios.BOLogin
2015-10-15 09:58:01,968 INFO  [com.previred.wportal] http-8180-9  SIN SESION  SIN USUARIO Invocando servicio srvsesionloginonUsuario
2015-10-15 09:58:01,968 INFO  [com.previred.wportal] http-8180-9  SIN SESION  SIN USUARIO Usando clase de negocios: com.previred.privilegios.sesion.BOSesi
2015-10-15 09:58:01,999 INFO  [com.previred.wportal] http-8180-9  SIN SESION  SIN USUARIO Invocando servicio srvfhui
2015-10-15 09:58:01,999 INFO  [com.previred.wportal] http-8180-9  Ejecutando la consulta: {call spLST_Parametros_Todos} en el pool sqlserveratosUsuario
2015-10-15 09:58:02,000 INFO  [com.previred.wportal] http-8180-9  SIN SESION  SIN USUARIO Usando clase de negocios: com.previred.wPortal.login.BOCheckUpdD
2015-10-15 09:58:02,003 INFO  [com.previred.wportal] http-8180-9  SIN SESION  SIN USUARIO Invocando servicio srvfhuireal
2015-10-15 09:58:02,097 INFO  [com.previred.wportal] http-8180-9  SIN SESION  SIN USUARIO Invocando servicio srvchkparametrosmetro
2015-10-15 09:58:02,097 INFO  [com.previred.wportal] http-8180-9  SIN SESION  SIN USUARIO Usando clase de negocios: com.previred.wPortal.login.BOCheckPara
2015-10-15 09:58:02,097 INFO  [com.previred.wportal] http-8180-9  SIN SESION  SIN USUARIO Invocando servicio srvfinal
2015-10-15 09:58:02,097 INFO  [com.previred.wportal] http-8180-9  SIN SESION  SIN USUARIO Usando clase de negocios: com.previred.privilegios.BOLoginChange2015
2015-10-15 09:58:02,097 INFO  [com.previred.wportal] http-8180-9  9E7DA5670B109D840AA3E9B9C7B77F7F.node1  16275643  Invocando servicio srvpaso
2015-10-15 09:58:02,097 INFO  [com.previred.wportal] http-8180-9  9E7DA5670B109D840AA3E9B9C7B77F7F.node1  16275643  Usando clase de negocios: com.previred.wPo
2015-10-15 09:58:02,209 INFO  [com.previred.wportal] http-8180-9  9E7DA5670B109D840AA3E9B9C7B77F7F.node1  16275643  Tiempo programa [prgroles] 112
2015-10-15 09:58:02,209 INFO  [com.previred.wportal] http-8180-9  9E7DA5670B109D840AA3E9B9C7B77F7F.node1  16275643  Tiempo programa [prglogin] 322
```

Ilustración 11: Extracto de log wPortalLog.

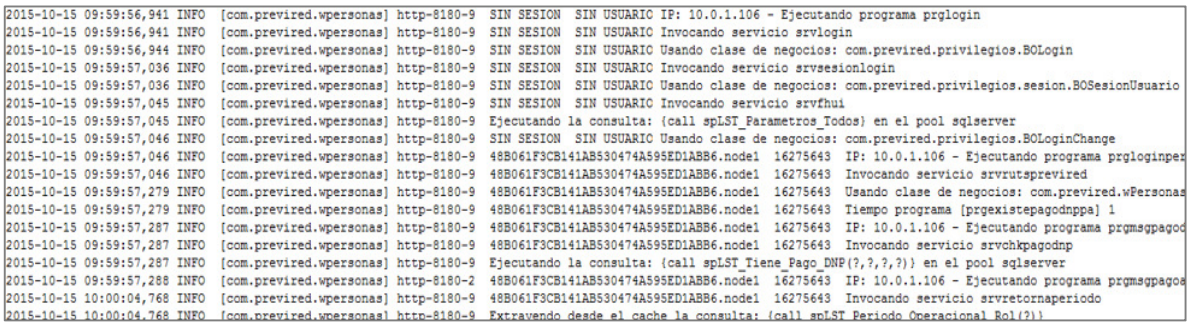
[6] Los extractos de logs es en base a información ficticia de pruebas, ya que por confidencialidad de información no se muestran datos personales de los usuarios del sitio.

wEmpresasLog: Log que registra cuándo un usuario ha elegido trabajar dentro del rol Empresa.

```
2015-10-15 10:00:19,269 INFO  [com.previred.wempresas] http-8180-9  SIN SESION  SIN USUARIO IP: 10.0.1.106 - Ejecutando programa prglogin
2015-10-15 10:00:19,277 INFO  [com.previred.wempresas] http-8180-9  SIN SESION  SIN USUARIO Invocando servicio srvlogin
2015-10-15 10:00:19,280 INFO  [com.previred.wempresas] http-8180-9  SIN SESION  SIN USUARIO Usando clase de negocios: com.previred.privilegios.BOLogin
2015-10-15 10:00:19,285 INFO  [com.previred.wempresas] http-8180-9  SIN SESION  SIN USUARIO Invocando servicio srvsesionlogin
2015-10-15 10:00:19,285 INFO  [com.previred.wempresas] http-8180-9  SIN SESION  SIN USUARIO Usando clase de negocios: com.previred.privilegios.sesion.BOSesionUsuario
2015-10-15 10:00:19,286 INFO  [com.previred.wempresas] http-8180-9  SIN SESION  SIN USUARIO Invocando servicio srvfhui
2015-10-15 10:00:19,286 INFO  [com.previred.wempresas] http-8180-9  Ejecutando la consulta: {call spLST_Parametros_Todos} en el pool sqlserver
2015-10-15 10:00:19,287 INFO  [com.previred.wempresas] http-8180-9  SIN SESION  SIN USUARIO Usando clase de negocios: com.previred.privilegios.BOLoginChange
2015-10-15 10:00:19,288 INFO  [com.previred.wempresas] http-8180-9  A4E763760BF0F6561E2BA02DED97FF5D.node1  16275643  IP: 10.0.1.106 - Ejecutando programa prgloginempresas
2015-10-15 10:00:19,288 INFO  [com.previred.wempresas] http-8180-9  A4E763760BF0F6561E2BA02DED97FF5D.node1  16275643  Invocando servicio srvrutsprevired
2015-10-15 10:00:19,288 INFO  [com.previred.wempresas] http-8180-9  Ejecutando la consulta: {call spLST_Priv_Usuarios_Previred} en el pool sqlserver_priv
2015-10-15 10:00:19,288 INFO  [com.previred.wempresas] http-8180-9  A4E763760BF0F6561E2BA02DED97FF5D.node1  16275643  Usando clase de negocios: com.previred.wEmpresas.usua
2015-10-15 10:00:19,288 INFO  [com.previred.wempresas] http-8180-9  A4E763760BF0F6561E2BA02DED97FF5D.node1  16275643  Invocando servicio srvretornaperiodo
2015-10-15 10:00:19,295 INFO  [com.previred.wempresas] http-8180-9  Extrayendo desde el cache la consulta: {call spLST_Parametros_Todos}
2015-10-15 10:00:19,295 INFO  [com.previred.wempresas] http-8180-9  A4E763760BF0F6561E2BA02DED97FF5D.node1  16275643  Invocando servicio srvelegirmaletin
2015-10-15 10:00:19,295 INFO  [com.previred.wempresas] http-8180-9  Ejecutando la consulta: {call spLST_pagadores_usuario(?)} en el pool sqlserver
2015-10-15 10:00:19,296 INFO  [com.previred.wempresas] http-8180-9  A4E763760BF0F6561E2BA02DED97FF5D.node1  16275643  Usando clase de negocios: com.previred.wEmpresas.male
2015-10-15 10:00:19,297 INFO  [com.previred.wempresas] http-8180-9  A4E763760BF0F6561E2BA02DED97FF5D.node1  16275643  IP: 10.0.1.106 - Ejecutando programa prglistadoempres
2015-10-15 10:00:22,311 INFO  [com.previred.wempresas] http-8180-9  A4E763760BF0F6561E2BA02DED97FF5D.node1  16275643  Invocando servicio srvpaso
```

Ilustración 12: Extracto de log wEmpresasLog.

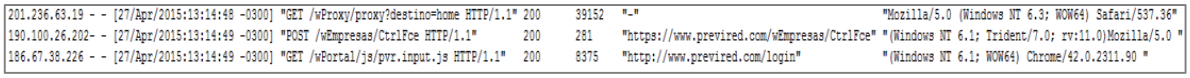
wPersonasLog: Log que registra cuándo un usuario ha elegido trabajar dentro del rol Persona.



```
2015-10-15 09:59:56,941 INFO  [com.previred.wpersonas] http-8180-9  SIN SESION  SIN USUARIO IP: 10.0.1.106 - Ejecutando programa prglogin
2015-10-15 09:59:56,941 INFO  [com.previred.wpersonas] http-8180-9  SIN SESION  SIN USUARIO Invocando servicio srvlogin
2015-10-15 09:59:56,944 INFO  [com.previred.wpersonas] http-8180-9  SIN SESION  SIN USUARIO Usando clase de negocios: com.previred.privilegios.BOLogin
2015-10-15 09:59:57,036 INFO  [com.previred.wpersonas] http-8180-9  SIN SESION  SIN USUARIO Invocando servicio srvsesionlogin
2015-10-15 09:59:57,036 INFO  [com.previred.wpersonas] http-8180-9  SIN SESION  SIN USUARIO Usando clase de negocios: com.previred.privilegios.sesion.BOSesionUsuario
2015-10-15 09:59:57,045 INFO  [com.previred.wpersonas] http-8180-9  SIN SESION  SIN USUARIO Invocando servicio srvfhui
2015-10-15 09:59:57,045 INFO  [com.previred.wpersonas] http-8180-9  Ejecutando la consulta: {call spLST_Parametros_Todos} en el pool sqlserver
2015-10-15 09:59:57,046 INFO  [com.previred.wpersonas] http-8180-9  SIN SESION  SIN USUARIO Usando clase de negocios: com.previred.privilegios.BOLoginChange
2015-10-15 09:59:57,046 INFO  [com.previred.wpersonas] http-8180-9  48B061F3CB141AB530474A595ED1ABB6.node1  16275643  IP: 10.0.1.106 - Ejecutando programa prgloginpe
2015-10-15 09:59:57,046 INFO  [com.previred.wpersonas] http-8180-9  48B061F3CB141AB530474A595ED1ABB6.node1  16275643  Invocando servicio srvrutsprevired
2015-10-15 09:59:57,279 INFO  [com.previred.wpersonas] http-8180-9  48B061F3CB141AB530474A595ED1ABB6.node1  16275643  Usando clase de negocios: com.previred.wPersonas
2015-10-15 09:59:57,279 INFO  [com.previred.wpersonas] http-8180-9  48B061F3CB141AB530474A595ED1ABB6.node1  16275643  Tiempo programa [prgexistepagodnppa] 1
2015-10-15 09:59:57,287 INFO  [com.previred.wpersonas] http-8180-9  48B061F3CB141AB530474A595ED1ABB6.node1  16275643  IP: 10.0.1.106 - Ejecutando programa prgmsgpagod
2015-10-15 09:59:57,287 INFO  [com.previred.wpersonas] http-8180-9  48B061F3CB141AB530474A595ED1ABB6.node1  16275643  Invocando servicio srvchkpagodnp
2015-10-15 09:59:57,287 INFO  [com.previred.wpersonas] http-8180-9  Ejecutando la consulta: {call spLST_Tiene_Pago_DNP(?,?,?,?)} en el pool sqlserver
2015-10-15 09:59:57,288 INFO  [com.previred.wpersonas] http-8180-2  48B061F3CB141AB530474A595ED1ABB6.node1  16275643  IP: 10.0.1.106 - Ejecutando programa prgmsgpagoe
2015-10-15 10:00:04,768 INFO  [com.previred.wpersonas] http-8180-9  48B061F3CB141AB530474A595ED1ABB6.node1  16275643  Invocando servicio srvretornaperiodo
2015-10-15 10:00:04,768 INFO  [com.previred.wpersonas] http-8180-9  Extrayendo desde el cache la consulta: {call spLST_Periodo_Operacional_Rol(?)}
```

Ilustración 13: Extracto de log wPersonasLog.

Accesslog: Log sistémico del servicio Apache (HTTP Server), éste se encarga de registrar todas las consultas web que se realizan al servidor, entre sus características se encuentra que almacena información acerca del cliente (IP, agente de usuario, etc.)



```
201.236.63.19 - - [27/Apr/2015:13:14:48 -0300] "GET /wProxy/proxy?destino=home HTTP/1.1" 200     39152   "-"                                          "Mozilla/5.0 (Windows NT 6.3; WOW64) Safari/537.36"
190.100.26.202- - [27/Apr/2015:13:14:49 -0300] "POST /wEmpresas/CtrlFce HTTP/1.1"       200     281     "https://www.previred.com/wEmpresas/CtrlFce" "(Windows NT 6.1; Trident/7.0; rv:11.0)Mozilla/5.0 "
186.67.38.226 - - [27/Apr/2015:13:14:49 -0300] "GET /wPortal/js/pvr.input.js HTTP/1.1"  200     8375    "http://www.previred.com/login"              "(Windows NT 6.1; WOW64) Chrome/42.0.2311.90 "
```

Ilustración 14: Extracto de log AccessLog.

Para entender la relación entre el sitio web y los logs descritos anteriormente, la Ilustración 15 muestra gráficamente la ubicación de ellos dentro del portal web de la organización.

Ilustración 15: Diagrama de ubicación de log dentro del sitio web.

El registro de información en los archivos de logs dentro del sitio web es temporal, siempre se escribe en un solo log, ya que el usuario sólo puede estar en un rol a la vez (rol Empresa o rol Persona).

La Ilustración 16 muestra el ejemplo de un usuario que ingresa al sitio web realizando las siguientes acciones.

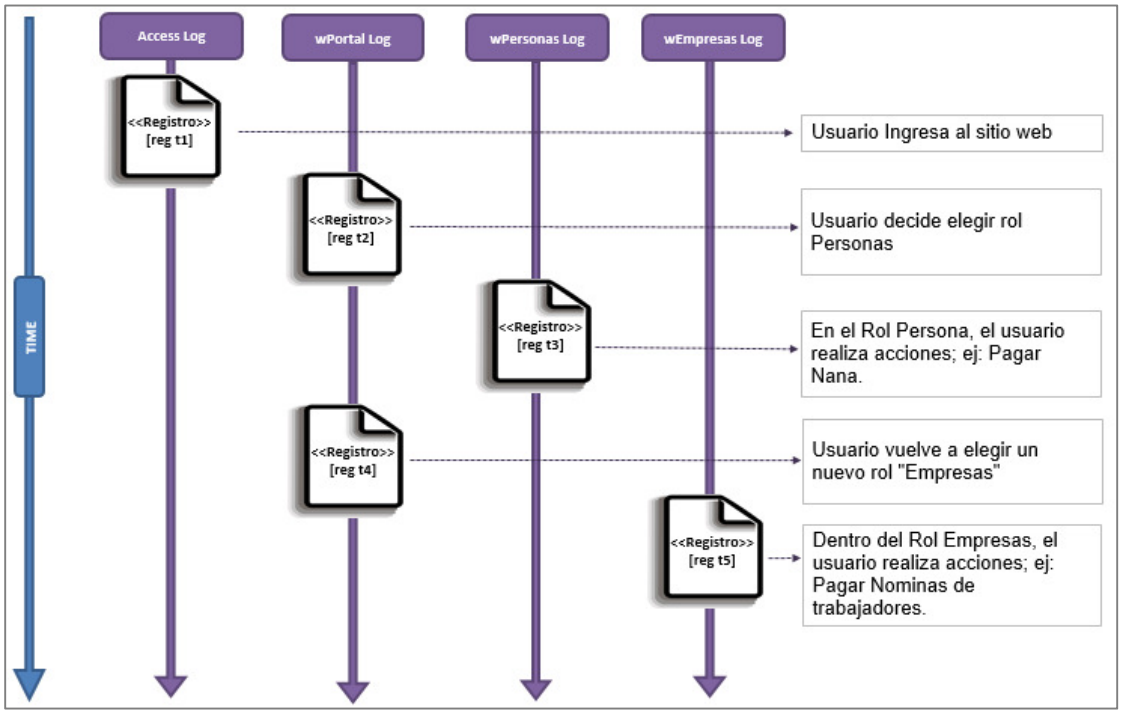


Ilustración 16: Temporalidad de registro en archivos de log del sitio web.

6. Etapa de Solución Técnica

El objetivo de esta etapa es presentar la arquitectura y los métodos de la solución Big Data implementados, donde se especifica claramente qué arquitectura Big Data se utilizó y el porqué de la decisión. Junto con ello se muestra las distintas fases que la componen, desde la captura, procesamiento y salida de los datos, así como el detalle de los métodos MapReduce desarrollados.

6.1. Definir Tecnología de Big Data

Actualmente existe una variada gama de productos Big Data que los proveedores de tecnologías están promocionando. Cada uno posee características propias, pero todos comparten la base de Hadoop en su implementación.

El proveedor de tecnología que se eligió fue Cloudera CDH (*Cloudera Distribution Hadoop)* por las siguientes razones:

- **Soporte a nivel Nacional**: La empresa tiene como regla que cualquier tecnología que se incorpore debe tener soporte a nivel nacional, esto es con la intención de que ante algún problema exista un ente que pueda responder. Cloudera posee una alianza estratégica con Red Hat, socio tecnológico de la compañía en lo relacionado a plataforma Linux, por lo que ante cualquier situación es Red Hat quien entrega soporte a nivel nacional de Cloudera.

- **Tecnologías Open Source**: Cloudera CDH es una suite basada en la distribución open-source de Apache Hadoop y un conjunto de proyectos relacionados. CDH entrega los elementos centrales de Apache Hadoop (*Core Elemensts*), junto con una forma fácil y amigable de administración de grandes cluster de procesamiento. Actualmente Cloudera está en la versión 5.

- **Tecnología líder en el mercado:** Cloudera es uno de los principales propulsores de las nuevas tecnologías abiertas en lo que respecta a Big Data. Un dato importante es que Doug Cutting, creador de Hadoop, es el Arquitecto en jefe desde el año 2009.

- **Documentación:** Cloudera posee un vasto material en lo que se refiere a documentación para aprendizaje, incluyendo manuales explicativos para cada uno de los diferentes módulos que posee la suite, cursos online, cursos presenciales, material descargable interactivo, etc.

La Ilustración 17 muestra los componentes de la suite Cloudera CDH.

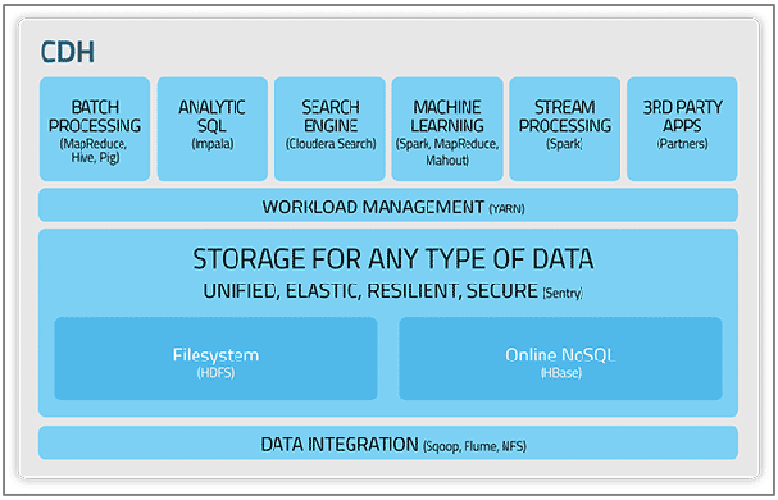


Ilustración 17: Esquema de servicios de Cloudera CDH [25].

Dentro de este producto tecnológico (Cloudera CDH) se decidió utilizar la solución batch para procesamiento de grandes volúmenes de datos, basada en la utilización de MapReduce sobre el sistema de almacenamiento distribuido de HDFS.

Se eligió la opción de procesamiento batch de Big Data debido a las siguientes razones:

- **Procesamiento mensual de información:** El procesamiento de los archivos logs será mensual en su inicio, lo que implica tener gran cantidad de archivos que procesar. Estos archivos corresponde a los datos de logs del sitio web generados en el mes anterior.
- **No es necesario procesamiento en tiempo real:** El procesamiento no es necesario que esté en línea, sino más bien es un procesamiento de grandes cantidades de datos.

Los servicios que se decidió utilizar dentro de la suite de Cloudera CDH son básicamente:

- Cloudera CDH: Batch Processing (MapReduce)
- Cloudera CDH: Filesystem HDFS

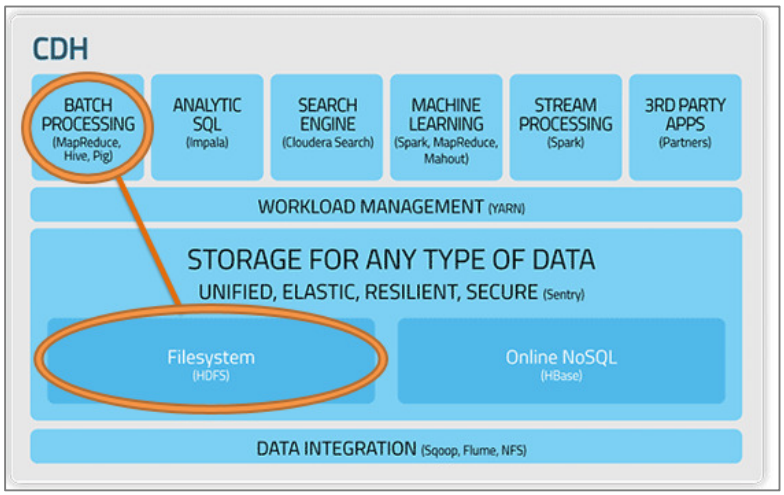


Ilustración 18: Servicios elegidos de Cloudera CDH [25].

6.2. Arquitectura de la solución

La arquitectura de solución que se implementó es la que se describe en la Ilustración 19. Cada una de sus secciones será descrita en profundidad en los capítulos siguientes.

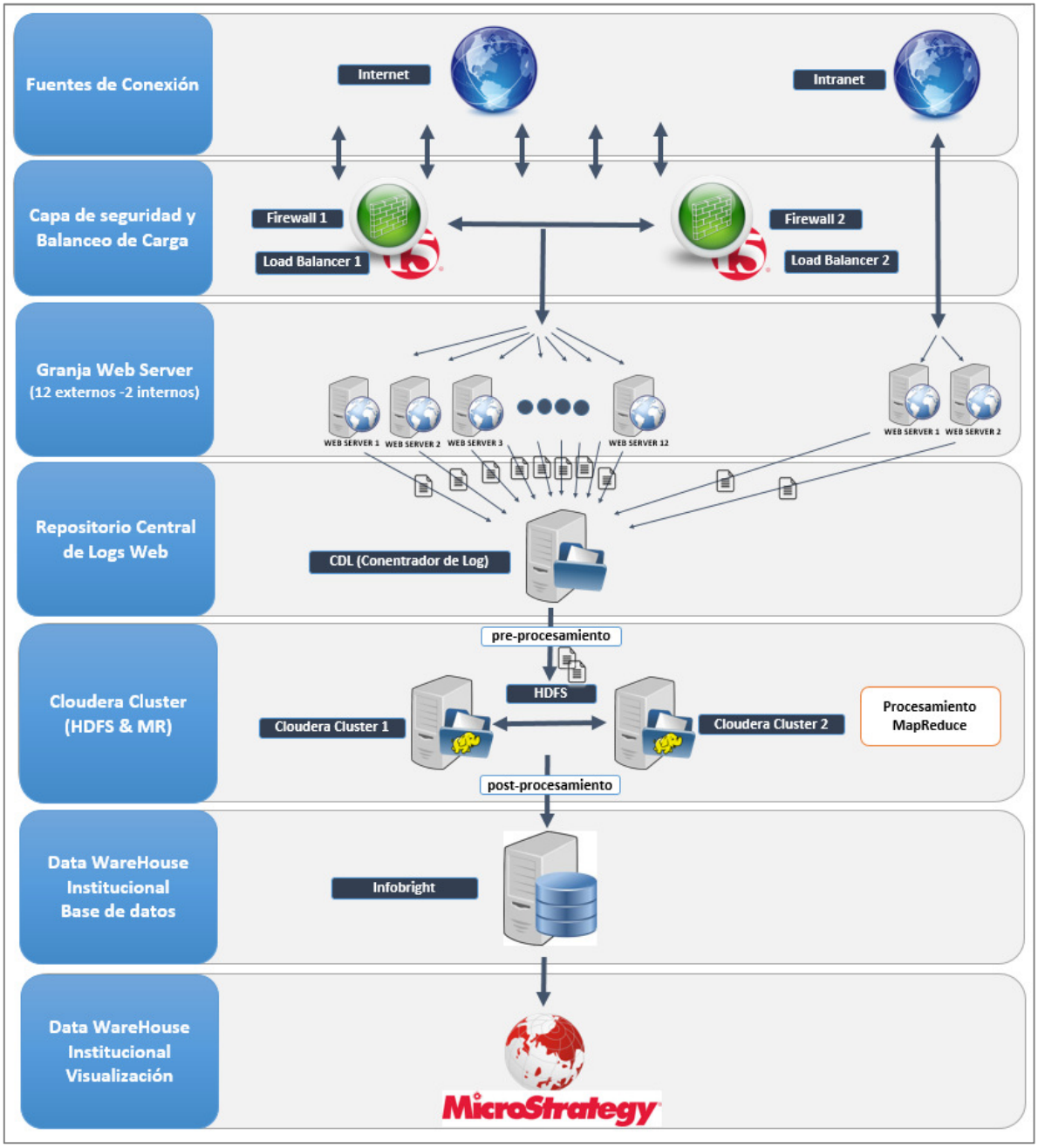


Ilustración 19: Esquema de la arquitectura de la solución.

6.2.1. Fuentes de Conexión

Las conexiones hacia los sitios web provienen de dos fuentes: La externa que corresponde al público general que accede a los servicios a través de internet y la conexiones internas que provienen de los ejecutivos de la empresa que operan de forma especial en los sitios web de la compañía y poseen servidores dedicados.

6.2.2. Capa de seguridad y balanceo de carga

Existe una capa de seguridad que está en alta disponibilidad (en caso de que falle un balanceador existe otro que seguirá operando). Esta capa provee tanto un firewall como un balanceador de carga de las transacciones que provienen de internet hacia la granja de servidores, su función es tener una equitativa cantidad de transacciones por servidor y a la vez entregar seguridad de que estas transacciones son confiables.

6.2.3. Granja de servidores web

Existe una granja de servidores web distribuidos en dos grupos, 12 servidores son para atender al pool de transacciones que provienen de internet, y dos servidores dedicados para atender a los ejecutivos de la organización, estos servidores dedicados son exclusivos de uso interno con el objetivo que dichos usuarios internos no compitan en recursos con el pool de usuarios de internet y proveer así una operatividad con mejor rendimiento.

Cada uno de estos servidores registra todas las peticiones, acciones y errores en distintos archivos logs, todos estos archivos logs, al terminar el día, son llevados a un concentrador de logs, el cual es un servidor encargado de poseer la totalidad de logs de la empresa.

6.2.4. Repositorio central de logs web

El repositorio central de logs es un servidor que es conocido como CDL (Concentrador de logs) que posee la totalidad de logs de la empresa. Cada servidor web es el encargado de dejar sus propios archivos de logs que se generaron en el día en este CDL. Los archivos de logs son comprimidos en formato bzip2, formato estándar en la organización, ya que entrega una tasa de compresión del orden del 3%-4% del tamaño original del archivo, esto permite almacenar gran cantidad de archivos sin requerir demasiado espacio en disco.

6.2.5. Cloudera Cluster (MR-HDFS)

El producto elegido fue Cloudera CDH en su última versión v.5, se realizó la implementación de un cluster de dos nodos de iguales características que se describen en la Tabla 2.

Tabla 2: Características de máquina cluster Cloudera.

Server Cloudera	
Tipo	Server Virtual
RAM	12 Gb
CPU	4 Cores
Disco	- 50 Gb para Sistema Operativo - 120 Gb para HDFS
Sistema Operativo	RHEL (Red Hat Enterprise Linux) 6.5

En este cluster Cloudera CDH 5 son cargados los logs comprimidos en formato bipz2 (pre-procesamiento de datos), los archivos logs que se cargan a HDFS son los especificados en la sección 5.2 (wEmpresasLog, wPortalLog, wPersonasLog, AccessLog). Una vez que estos archivos están cargados, procede la etapa de ejecución de los métodos MapReduce (procesamiento MapReduce sobre HDFS) que tienen la tarea de procesar todos los archivos de logs. Finalmente el resultado obtenido del procesamiento es exportado en formato texto para ser cargado en Infobright, la base de datos NoSQL de la organización (post-procesamiento de datos).

Como se comentó, existen tres procesos claves en esta etapa: Pre-procesamiento de datos, el procesamiento MapReduce, y finalmente el post-procesamiento de datos.

6.2.5.1. Pre-procesamiento de datos

Los archivos dentro del CDL están comprimidos en formato bzip2. Un punto importante es que HDFS está preparado para el procesamiento de archivos comprimidos en los formatos que se exponen en la Tabla 3, esto permite procesar archivos de logs sin necesidad de expandir su volumen real.

Tabla 3: Tabla de compresión Cloudera Hadoop.

Formato de compresión	Herramienta	Algoritmo	Extensión de archivo	Múltiples archivos	Divisible
Deflate	N/A	Deflate	.deflate	No	No
Gzip	gzip	Deflate	.gz	No	No
Bzip2	bzip2	bzip2	.bz2	No	Si
LZO	lzop	LZO	.lzo	No	No

La etapa de pre-procesamiento, presentada en Ilustración 20, tiene la tarea de:

- Descargar los archivos desde el servidor CDL.
- Revisar que los archivos estén correctamente comprimidos en bzip2.
- Generar una nomenclatura de nombres de los archivos de logs.
- Cargar los archivos en HDFS del cluster Cloudera.

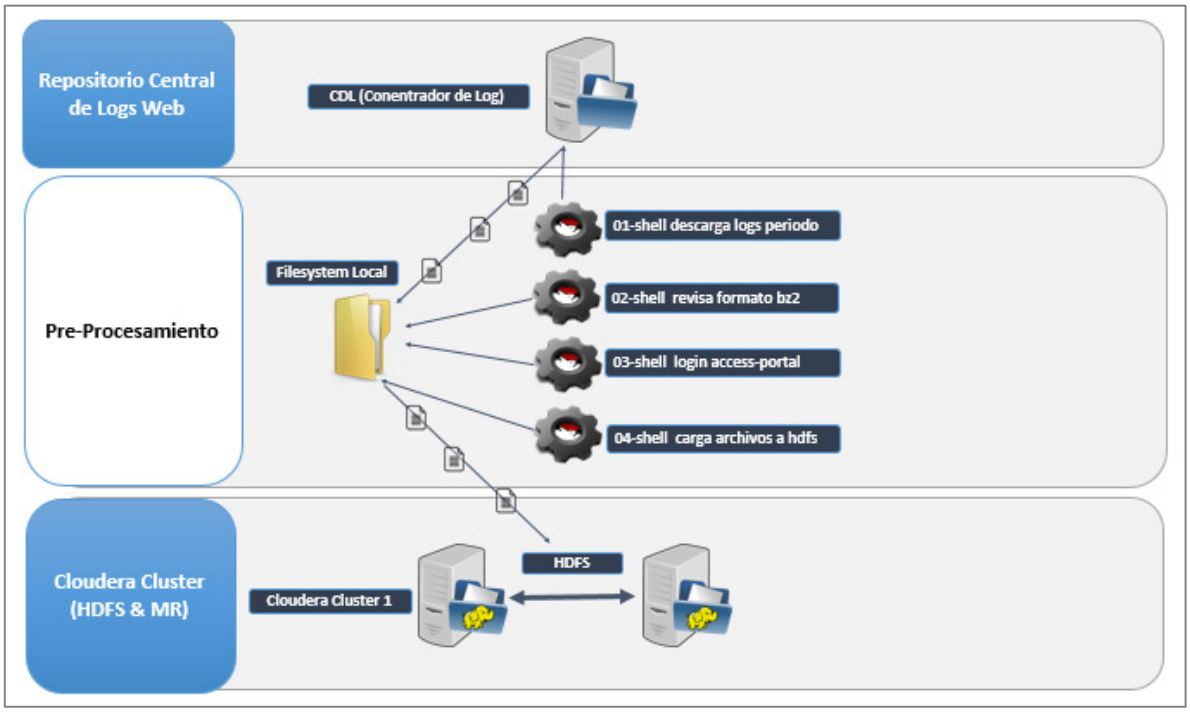


Ilustración 20: Esquema de pre-procesamiento de datos.

6.2.5.2. Procesamiento mediante MapReduce

Como antes se mencionó, el método que se utilizó para realizar el procesamiento de este gran volumen de datos fue el framework MapReduce, y la forma de utilizarlo se basó en el procesamiento batch de los archivos de logs.

Los métodos MapReduce desarrollados fueron tres, cada uno realiza una tarea específica de procesamiento.

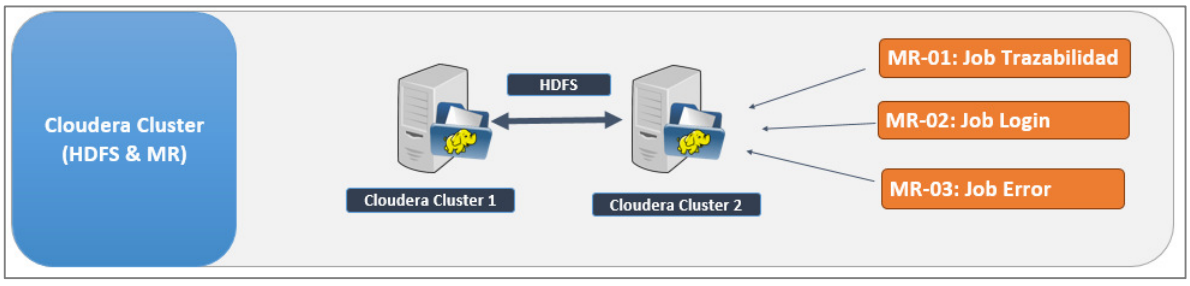


Ilustración 21: Esquema de procesamiento mediante MapReduce.

MapReduce "Job Trazabilidad": Este Job MapReduce realiza el procesamiento de todos los archivos logs de ambiente web (wPortalLog, wEmpresasLog, wPersonasLog), obteniendo la trazabilidad de los usuarios (se excluye de esta trazabilidad el ingreso 'login' al sitio web).

- La función *"Map"* realiza la detección de las líneas de los archivos logs semi-estructurados usando patrones en forma de expresiones regulares. Las líneas que esta función identifica son las que describen las acciones que los usuarios realizaron en el sitio web. La clave que emite cada tarea *"Map"* es el id de la sesión web.
- La función "*Reduce*" toma todas las acciones de un mismo id de sesión web y realiza el procesamiento de dichas acciones, entregando un detalle ordenado y estructurado de la trazabilidad de usuario.

MapReduce "Job Login": Este es un Job MapReduce específico para obtener una sola acción, el ingreso al sitio web del usuario. Esta acción es especial porque se deben unir dos archivos logs: "wPortalLog" (entrega la información de la acción de ingreso al sitio), con "accessLog" (entrega información del agente de usuario que el cliente estaba utilizando: Navegador web, Sistema Operativo y dispositivo).

- La función "*Map*" realiza la lectura de las líneas de los archivos: AccessLog y wPortalLog entregando como clave la IP-fecha referente a las líneas de cada archivo.
- La función "*Reduce"* toma todas las líneas de la misma clave (IP–fecha) y se encarga de realizar la unión del login con la información de agente de usuario que estaba usando en ese minuto.

MapReduce "Job Errores": Este Job se encarga de revisar todos los archivos logs del sitio web buscando el patrón de error que está definido mediante expresiones regulares.

- La función "Map" se encarga de leer las líneas de los archivos de logs semi-estructurados (wPortalLog, wPersonasLog, wEmpresasLog) y revisa si esa línea coincide con el patrón de error definido en las expresiones regulares.
- En este caso no fue necesario utilizar función Reduce.

6.2.5.3. Post-procesamiento de datos

Una vez que los archivos de logs son procesados por los métodos MapReduce descritos anteriormente, el resultado queda alojado dentro de HDFS, por lo cual existe una etapa de post-procesamiento que realiza la descarga de los resultados desde HDFS al disco local, esto para luego realizar la carga de dichos resultados en la base de datos institucional Infobright, la cual es una base de datos NoSQL orientada a modelo Columnar.

La etapa de post-procesamiento, es presentada en la Ilustración 22, tiene básicamente las siguientes sub-etapas:

- **Descarga de resultados desde HDFS**: Descarga desde HDFS, en formato de texto, el resultado obtenido por el procesamiento de los diferentes Jobs MapReduce.
- **Carga de datos a BD NoSQL:** Realiza la carga de los datos de resultados a Infobright, la cual es la base de datos institucional del data warehouse.

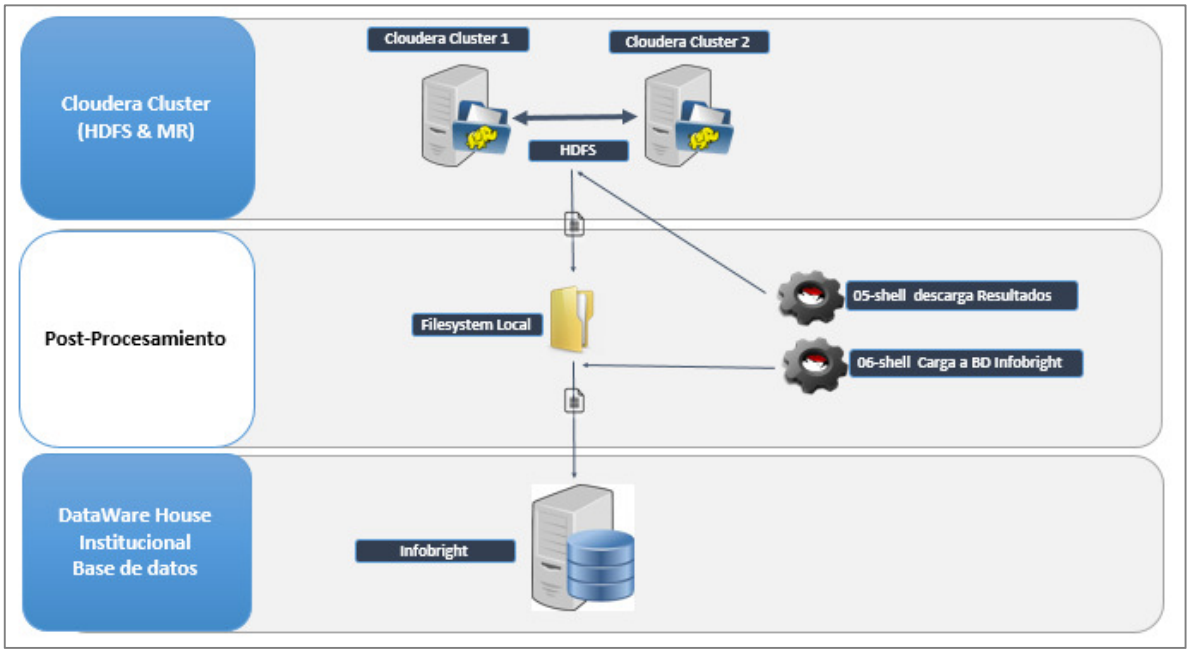


Ilustración 22: Esquema de post-procesamiento de datos.

6.2.6. Data WareHouse Institucional (Base de datos)

El resultado obtenido en etapas anteriores es cargado en una base de datos analítica que pertenece al Data warehouse institucional, esta base de datos "Infobright" tiene la característica de ser una base de datos NoSQL orientada al modelo Columnar. Entre las características del motor de datos "Infobright" están:

- Gran compresión de los datos entregando un 10% del tamaño normal de los datos.
- Rápidas consultas en tablas desnormalizadas.
- No necesita creación de índices, ni mantenimiento.
- Posee una estructura "*Knowledge Grid*" que almacena información compactada del contenido de las tablas, reemplazando el concepto clásico de índices.

El archivo de trazabilidad resultante del procesamiento de los métodos MapReduce "Job Trazabilidad" y "Job Login", es cargado en un pequeño esquema dentro de la base de datos de "Infobright", este esquema se presenta en la Ilustración 23.

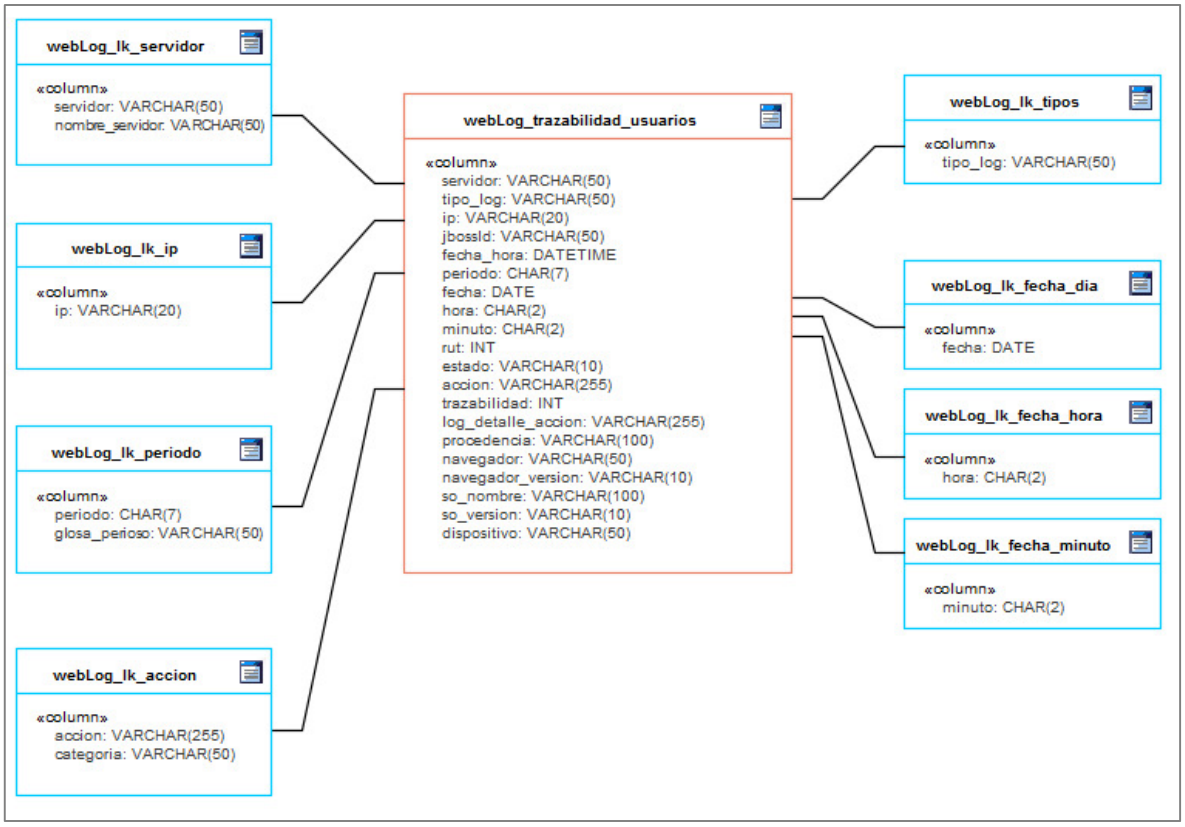


Ilustración 23: Modelo de datos de trazabilidad de Infobright.

El archivo de errores resultante del método MapReduce “Job Errores” es cargado en un pequeño esquema dentro de la base de datos de “Infobright”, presentado en la Ilustración 24.

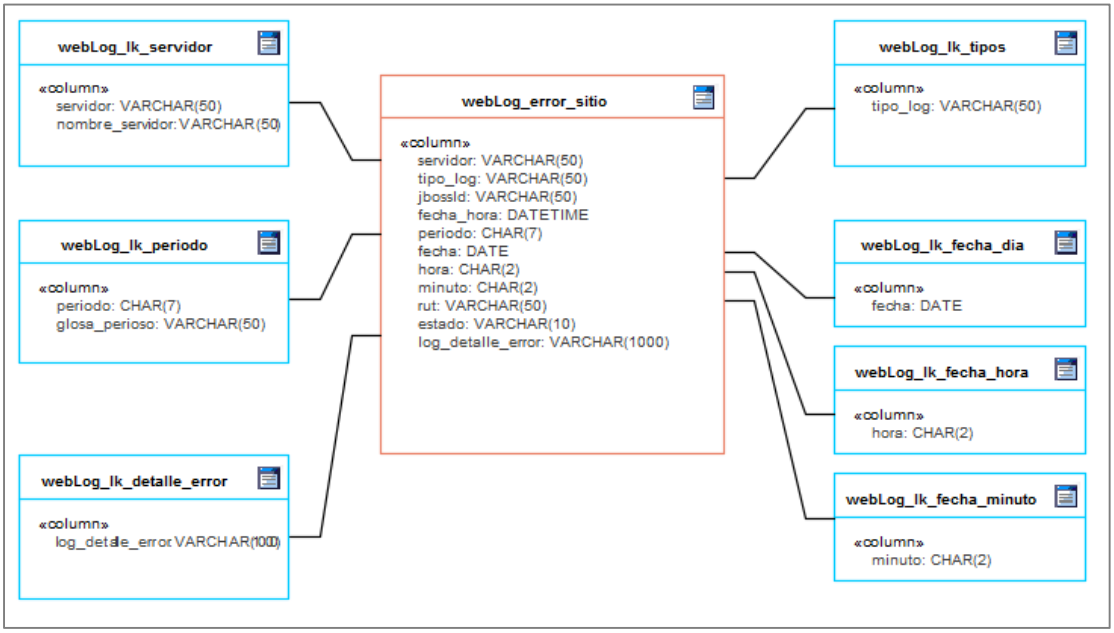


Ilustración 24: Modelo de datos de errores de Infobright.

6.2.7. Data WareHouse Institucional (Visualización)

Los datos cargados en la base de datos Infobright son vistos a través de la herramienta MicroStrategy, la cual es la forma de visualización que posee la compañía.

La visualización es un proyecto de forma paralela a éste y es desarrollado por el equipo BI de la compañía.

6.3. Métodos Desarrollados

Para procesar los archivos de logs de datos semi-estructurados se decidió desarrollar métodos basados en el framework de procesamiento distribuido MapReduce. Estos métodos se implementaron en lenguaje Java (generando ejecutables Jar), aunque la lógica pudo haber sido desarrollada en cualquier otro lenguaje: C++, Python, etc., ya que todos están actualmente soportados por Cloudera para definir procesos MapReduce.

Como se mostró en el capítulo anterior, los métodos MapReduce desarrollados fueron tres, los que serán descritos en profundidad en los capítulos siguientes:

- Job MapReduce Trazabilidad
- Job MapReduce Login
- Job MapReduce Errores

Cada uno de estos métodos se basa en procesar de forma distribuida un conjunto de archivos de logs con datos semi-estructurado provenientes de sistemas web. Estos archivos de logs semi-estructurados residen en un sistema de archivos distribuidos HDFS desde donde son procesados.

La forma de procesar estos datos semi-estructurados, donde las líneas de logs no poseen una estructura fija ni un formato estrictamente definido, es mediante expresiones regulares (Regex) que poseen la capacidad de detectar líneas con un cierto patrón de búsqueda flexible a cambios en su formato.

La Ilustración 25 muestra una expresión regular para leer líneas de trazabilidad del log *wPortalLog*. Se define la expresión de forma de identificar grupos que posteriormente serán procesados individualmente. En este caso, las líneas identificadas por la expresión regular son aquellas que comienzan con la fecha y hora de formato ‘2015-05-03 00:01:53,167’, para ello se utiliza la expresión *“^([\\d-]+\\s[\\d:,]+)*” que representa:

- “**^[\\d-]+”***:* El inicio de la línea debe comenzar con dato numérico y presencia del carácter ’-‘. El signo ‘+’ implica la ocurrencia de más de una vez.
- **“\\s”**: Debe seguir un espacio.
- **“[\\d:,]+”**: Finalmente debe terminar con dato numérico con la presencia del carácter ‘:’ y ‘,’. El signo ‘+’ implica la ocurrencia de más de una vez.

Todo este conjunto, al estar encerrado entre paréntesis “()” queda almacenado en el grupo 1, esto después permite a la expresión regular saber que la fecha y hora está almacenada en el primer grupo de definición.

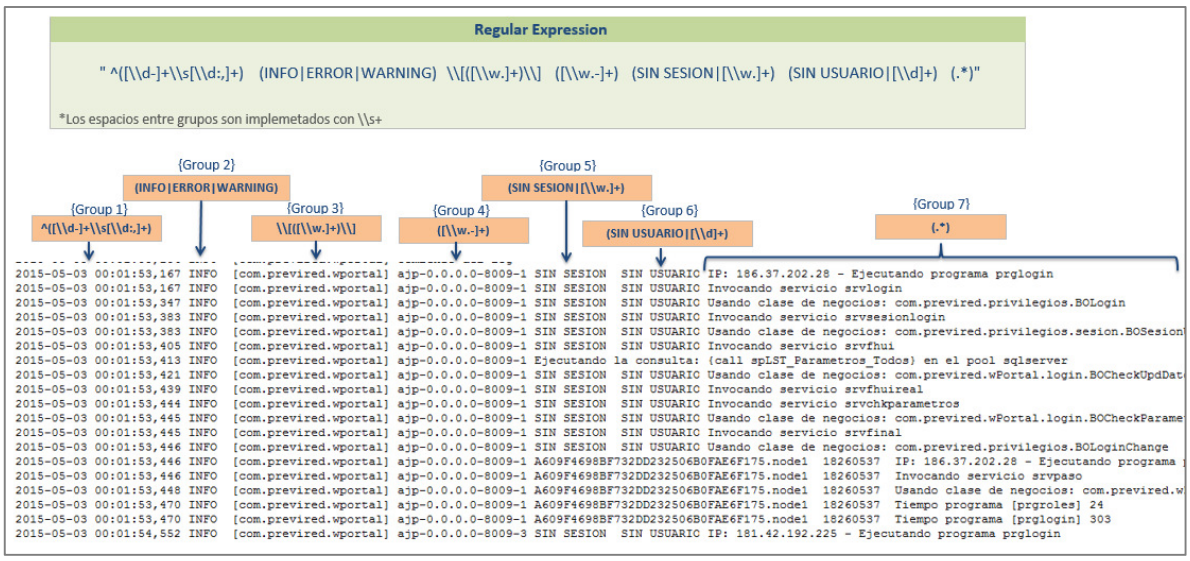


Ilustración 25: Modelo de "Expresión Regular".

6.3.1. MapReduce 01 "Job Trazabilidad"

El "Job de Trazabilidad" es un método basado en el framework de procesamiento distribuido MapReduce. Este método se encarga de procesar los archivos de logs generados en el sitio web dando como resultado el detalle de la trazabilidad de los usuarios.

A continuación, en la Ilustración 26 se describe gráficamente el Job MapReduce de trazabilidad.

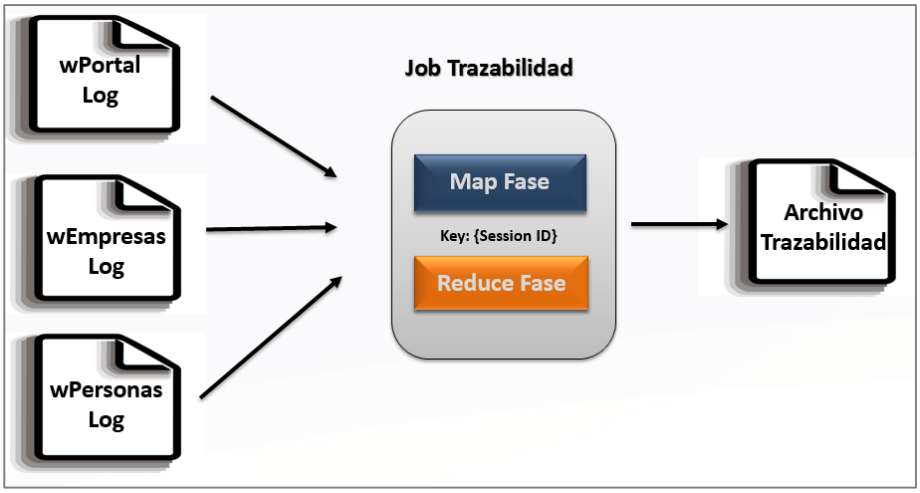


Ilustración 26: Diagrama de MapReduce de Trazabilidad.

El "Job de Trazabilidad" procesa tres tipos de archivos distintos (descripción más detallada de estos logs son entregados en el Capítulo 5.2).

- **wPortalLog:** Registros de logs de forma semi-estructurada del portal web (portal de inicio).
- **wEmpresasLog**: Registros de logs de forma semi-estructurada del portal web de Empresas (rol de Empresas dentro del portal).
- **wPersonasLog**: Registros de logs de forma semi-estructurada del portal web de Personas (rol de Personas dentro del portal).

Este método al estar basado en el framework de procesamiento distribuido MapReduce posee dos fases principales, la fase de Map y la fase Reduce, dichas fases son las que a nivel de usuario son implementadas.

Fase Map (“*Map Phase”*): Esta fase se encarga de realizar el mapeo de los datos que luego serán enviados a la fase de Reduce donde finalmente serán procesados para obtener la trazabilidad de los usuarios.

Un aspecto importante de los archivos logs (wPortalLog, wEmpresasLog, wPersonasLog) es que en todos existe un dato que es la sesión web del usuario, esta sesión web es única, por ende se decidió utilizarla como clave (key) en esta fase, permitiendo que la fase Reduce procese todas las líneas correspondientes a la misma sesión web.

El pseudocódigo de la fase de Map del “Job de Trazabilidad” es mostrado en la Ilustración 27, el cual básicamente se puede resumir en:

- Se identifica el archivo que se está procesando, esto es útil debido a que el nombre del servidor está dentro del nombre del archivo, con esta información se asocia todas las acciones dentro de este archivo al servidor web de la granja.
- Lee las líneas del archivo de logs (wPortalLog, wEmpresasLog, wPersonasLog) para compararlas con la expresión regular que está definida para líneas de trazabilidad.
- En caso de que la línea leída concuerde con la expresión regular definida para líneas de trazabilidad, se toma de ella la sesión web como clave, y el resto de la línea es formateada para pasar a la fase Reduce.
- Finalmente se emite la clave y el valor.

```
Class Mapper
    filename ⟵ ∅                                  // Es usado para obtener el servidor

    Method setup (Context context)
        filename: context.getPath()

    Method map (key Offset , Value line)
        regexUser: Patrón para expresión Regular linea trazabilidad
        if line match regexUser then
            outkey : getSessionID(line)
            outvalue: parseLineUserTable(Line,filename)
            Emit(outkey,outvalue)
        end
```

Ilustración 27: Función "Map" de método MapReduce Trazabilidad.

Fase Reduce (“*Reduce Phase*”): Esta fase se encarga de realizar el procesamiento de todas las líneas que poseen la misma clave, para este caso la clave que se definió es el id de la sesión web, eso quiere decir que esta fase procesa todas las líneas asociadas a la misma sesión web.

El pseudocódigo de la fase de Reduce del “Job de Trazabilidad” es mostrado en la Ilustración 28, el cual básicamente se puede resumir en:

- Se crea una lista del tipo HashMap[7] para almacenar la descripción de los códigos de programas, ejemplo: (“prgsubNominas”-“Dentro del sitio de empresas acción de subir Nóminas”), esta lista de paridad es con el fin de tener una descripción amigable al momento de listar las acciones que los usuarios realizaron en el sitio.
- Se almacena en una lista todas las líneas asociadas de trazabilidad de usuarios de la misma sesión web.
- Se identifica el usuario asociado a la sesión, esto con el objetivo de que en caso de que alguna línea de esta sesión no haya guardado el usuario, se le colocará el usuario de la sesión.
- Finalmente se recorre la lista de acciones, se busca la descripción amigable del programa que se ejecutaba y se exporta dicha línea formateada.

```
Class Reducer
    listValues ⟵ {∅}
    Map < Str, Str > funcDescHashMap ⟵ {∅, ∅}
    UserSession ⟵ ∅

    Method Reduce (key SessionID , Values records[record1 record2 ... recordN] )
        Clear(listValues)
        funcDescHashMap: Load HashMap(nombrePrograma,Funcionalidad)
        for record ∈ records do
            listValues.Add(record)
            if getUser(record) ≠'Sin Usuario' & UserSession='' then
                userSession: getUser(record)
            end
        end
        for record ∈ listValues do
            recordParts[]: record.split()
            programName: records[ix]
            programFuncdesc: funcDescHashMap.get(programName)
            outkey : null
            outvalue:parseLineUserTable(recordParts[],userSession,programFuncdesc)
            Emit(outkey,outvalue)
        end


        Method parseLineUserTable Entrega una linea final formateada
        Map funcDescHashMap es un hashmap que posee la descripción de los programas
        Method getSessionId entrega la sessionId de la linea de log
        Method getUser entrega el usuario de la linea de log
```

Ilustración 28: Función "Reduce" de método MapReduce Trazabilidad.

[7] HashMap es una estructura que permite almacenar información del tipo clave-valor, permitiendo una eficiente búsqueda para pequeños conjuntos de datos.

6.3.2. MapReduce 02 “Job Login”

El “Job de Login” al igual que el anterior es un método basado en el framework de procesamiento distribuido MapReduce. Este método se encarga de procesar los archivos de logs generados en el sitio web junto con los registros de logs sistémicos para obtener la información del ingreso del usuario al portal web de la compañía.

A continuación, en la Ilustración 29 se describe gráficamente el “Job MapReduce de Login”.

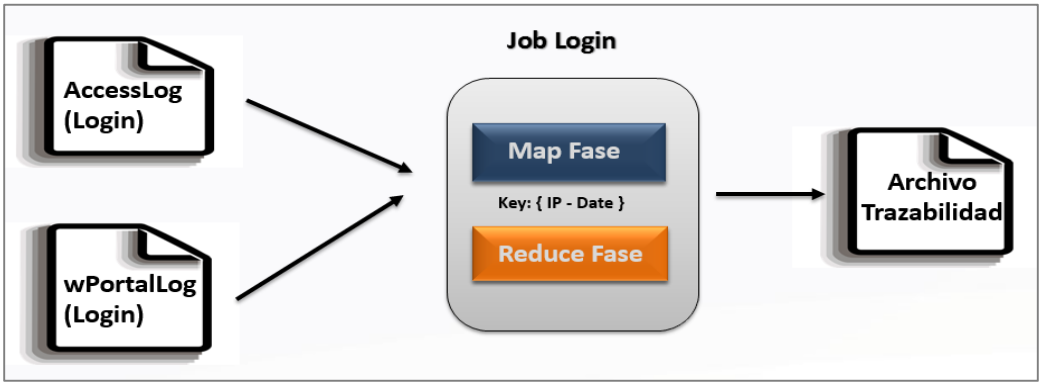


Ilustración 29: Diagrama de MapReduce de Login.

El “Job de Login” procesa dos tipos de archivos distintos (descripción más detallada de estos logs son entregados en el Capítulo 5.2).

- **wPortalLog:** Registros de logs de forma semi-estructurada del portal web (portal de inicio).
- **AccessLog**: Registros de logs sistémicos, este log registra información de cliente (IP, agente de usuario (navegador, Sistema Operativo, dispositivo), URL de donde proviene, etc.).

Este Job tiene la tarea de unir dos logs distintos entre sí para obtener la información de cuándo un usuario ingresó al sitio web, y a la vez determinar qué tipo de navegador, dispositivo y sistema operativo estaba usando en ese momento.

Este método al ser basado en el framework de procesamiento distribuido MapReduce, posee dos fases principales, la fase de Map y la fase Reduce.

Fase Map (“*Map Phase*”): Esta fase se compone de dos funciones que realizan la lectura de dos tipos de archivos: wPortalLog y AccessLog. Ambos archivos tienen tipos de estructura distintas, por ende se decidió optar por dos funciones Map.

El pseudocódigo de la fase de Map del “Job de Login” es mostrado en la Ilustración 30 e Ilustración 31, de las cuales básicamente se puede resumir:

- Se identifica el archivo que se está procesando, esto es útil debido a que el nombre del servidor está dentro del nombre del archivo, con esta información se asocia todas las acciones dentro de este archivo al servidor web de la granja.

- Se lee los archivos (wPortalLog, AccessLog) línea por línea, cada función Map lee un tipo de archivo distinto.
- Cada línea es revisada contra una expresión regular previamente creada.
- De las líneas seleccionadas se obtiene la clave ('IP'-'fecha') y el valor (línea formateada) para pasarlo a la fase Reduce.
- Finalmente se emite la clave y el valor.

```
Class Mapper1: AccessLog
    filename ⟵ ∅                                  // Es usado para obtener el servidor

        Method setup (Context context)
            filename: context.getPath()

        Method map (key Offset , Value line)
            regexAccessLogin: Patrón para expresión Regular login Access Log
            if line match regexAccessLogin then
                outkey : getIP(line) + getDate(line)
                outvalue: parseLineUserTable(Line,filename)
                Emit(outkey,outvalue)
            end
```

Ilustración 30: Función primer "Map" de método MapReduce Login.

```
Class Mapper2: WportalLog
    filename ⟵ ∅                                  // Es usado para obtener el servidor

    Method setup (Context context)
        filename: context.getPath()

    Method map (key Offset , Value line)
        regexWportalLogin: Patrón para expresión Regular login wPortal Log
        if line match regexUser then
            outkey : getIP(line) + getDate(line)
            outvalue: parseLineUserTable(Line,filename)
            Emit(outkey,outvalue)
        end
```

Ilustración 31: Función segundo "Map" de método MapReduce Login.

Fase Reduce ("*Reduce Phase*"): Esta fase se encarga de realizar el procesamiento de todas las líneas que poseen la misma clave, para este caso la clave que se definió es la IP y la fecha, esto quiere decir que se realiza el procesamiento de todas las líneas que representen un ingreso al sitio web que provengan de la misma IP y hayan ingresado en la misma fecha.

El pseudocódigo de la fase de Reduce del “Job de Login” es mostrado en la Ilustración 32, el que básicamente se puede resumir en:

- Se crean dos listas, una para almacenar los registros correspondientes a AccessLog, y otra para almacenar los provenientes de wPortalLog.
- Se ordenan los registros de la lista de wPortalLog.
- Se recorre los registros de la lista de wPortalLog que representan los logins de usuarios al sitio web.
- Por cada registro de la lista de wPortalLog se busca dentro de la lista AccessLog el registro más cercano.
- Finalmente una vez encontrado el registro se formatea la línea con la unión de ambos datos.

```
Class Reducer
    listAccess ⟵ {∅}
    listWportal ⟵ {∅}

    Method Reduce (key IP-Date , Values [record1 record2 ... recordN] )
        Clear(listAccess)
        Clear(listWportal)

        for record ∈ records do
            if line match Accesslog then
                listAccess.Add(record)
            else
                if line match wPortalLog then
                    listWportal.Add(record)
                end
            end
        end

        if !Empty(listWportal) then
            Sort(listWportal)
            for record ∈ listWportal do
                AccessLogin=listAccess.Find(record.getDateTime())
                outkey : null
                outvalue: parseLineUserTable(AccessLogin,record)
                Emit(outkey,outvalue)
            end
        end


        Method parseLineUserTable Entrega una linea final formateada
        Method getIp y getDate Entregan la IP y la fecha de la linea de log
        Method Sort(listWportal) ordena la lista de wPortal Log en base a la fechaHora
        Method listAccess.Find entrega la linea de Accesslog mas cercana a esa fecha-hora
```

Ilustración 32: Función "Reduce" de método MapReduce Login.

6.3.3. MapReduce 03 "Job Errores"

El "Job de Errores" al igual que los anteriores es un método basado en el framework de procesamiento distribuido MapReduce. Este método se encarga de procesar los archivos de logs generados en el sitio web para detectar todos los errores que pudiesen existir dentro de ellos.

A continuación, en la Ilustración 33 se describe gráficamente el Job MapReduce de Errores.

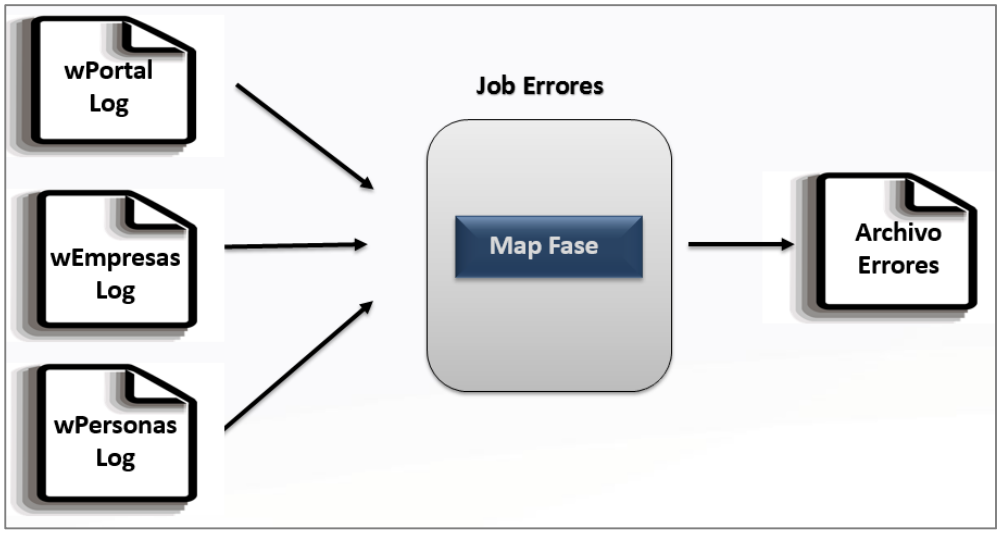


Ilustración 33: Diagrama de MapReduce de Errores.

El Job de Errores procesa tres tipos de archivos distintos (descripción más detallada de estos logs son entregados en el Capítulo 5.2).

- **wPortalLog:** Registros de logs de forma semi-estructurada del portal web (portal de inicio).
- **wEmpresasLog**: Registros de logs de forma semi-estructurada del portal web de Empresas (rol de Empresas dentro del portal).
- **wPersonasLog**: Registros de logs de forma semi-estructurada del portal web de Personas (rol de Personas dentro del portal).

Este método se definió con una sola fase Map que realiza la búsqueda por todas las líneas de los archivos logs detectando patrones de errores. La fase de Reduce no fue implementada, debido a que el procesamiento completo se realiza en la primera fase.

El pseudocódigo de la fase de Map del Job de Errores es mostrado en la Ilustración 34, de la que se resume:

- Se identifica el archivo que se está procesando, esto es útil debido a que el nombre del servidor está dentro del nombre del archivo, con esta información se asocia todas las acciones dentro de este archivo al servidor web de la granja.
- Se lee los archivos (wPortalLog, AccessLog) línea por línea.
- Se definen dos expresiones regulares, la diferencia entre ellas es que la primera identifica errores con mayor detalle (usuario a quién le sucedió el error, la IP de donde se detectó el error, etc.) y la segunda es información más reducida pero más segura de detectar. Se trata de coincidir la línea con la primera expresión regular y en caso de no coincidir, se trata de coincidir con la segunda expresión regular.
- Finalmente para las líneas que coincidan, tanto con la primera como segunda expresión regular, se formatea su salida de forma estructurada.

```
Class Mapper
    filename ⟵ ∅
    Method setup (Context context)
        filename: context.getPath()

    Method map (key Offset , Value line)
        regexError1: Patrón para expresión Regular tipo Error 1
        regexError2: Patrón para expresión Regular tipo Error 2

        if line match regexError1 then
            Emit(null,parseLineErrorTable(line,filename))
        else
            if line match regexError2 then
                Emit(null,parseLineErrorTable(line,filename))
            end
        end


        Method parseLineErrorTable Entrega una linea formateada
```

Ilustración 34: Función "Map" de método MapReduce Errores.

7. Etapa de Evaluación

La evaluación del proyecto se hizo en base a los resultados obtenidos, así como de la escalabilidad de ésta.

- **Evaluación de Resultados:** Se contrastó los resultados obtenidos con los casos de usos definidos al inicio del proyecto.
- **Evaluación de Escalabilidad:** Se creó una infraestructura en ambiente Cloud (Amazon) de 9 nodos (8 de procesamiento, 1 de administración) y se probaron los métodos de forma creciente, empezando con 1 nodo, seguido de 2, 4 y 8 nodos respectivamente.

7.1. Evaluación de Resultados

La evaluación de resultados se realizó en base a los dos casos de uso expuestos al inicio del proyecto, así como del conjunto de preguntas de negocio de cada caso de uso.

La Tabla 4 entrega un resumen de los casos de uso descritos al inicio de proyecto, así como un detalle de las preguntas de negocio asociada a cada uno.

Tabla 4: Tabla de preguntas de negocio para evaluación de resultados.

#	Caso de Uso	Preguntas de Negocio (PN)
1	Trazabilidad de usuarios	**PN-1.** Dado un Rut de un usuario, tener el detalle de la trazabilidad de sus acciones dentro del sitio web de la organización. **PN-2.** Tener el detalle de los usuarios que ingresan al sitio web por una determinada versión de navegador web. **PN-3.** Identificar carga de usuarios en la granja de servidores que posee la organización. **PN-4.** Revisar flujos dentro del sitio web que puedan presentar dificultad para los usuarios.
2	Detección de Errores	**PN-1.** Cantidad de errores más presente en el sitio web de la organización. **PN-2.** Revisar un error específico en la granja de servidores, entregando el comportamiento de este error en el tiempo. **PN-3.** Detalle de personas con mayor número de errores.

7.1.1. Evaluación de resultados caso de uso "Trazabilidad de usuarios"

Este caso de uso se basaba en tener la información relacionada a la trazabilidad de los usuarios dentro de los distintos portales web de la compañía.

7.1.1.1. PN-1 Trazabilidad de un usuario específico

Esta pregunta de negocio se basa en determinar la trazabilidad de las acciones de un usuario específico dentro del sitio web de la organización. Poseer esta información es de carácter muy relevante, ya que permite saber un conjunto de información del usuario específico.

- Qué servidor atendió al dicho usuario.
- Qué páginas del sitio web estaba usando en su sesión web.
- Desde qué dispositivo, versión de Sistema Operativo y navegador web ingresó al portal.
- Desde qué IP estaba trabajando.
- A qué hora estaba usando el sitio web.

La Ilustración 35 entrega el resultado de esta información para un usuario específico[8], donde se muestra la información de trazabilidad que da respuesta a la pregunta de negocio N° 1.

Rut	Fecha Hora	IP	Servidor	Tipo Log	Accion realizada	Dispositivo	SO	navegador
11234567	26-10-2015 13:16:55	10.0.1.150	pv1rprweb042	com.previred.wportal	Login al sitio web	Personal computer	Windows 7	Firefox 41
11234567	26-10-2015 13:16:55	10.0.1.150	pv1rprweb042	com.previred.wportal	Página de selección de Rol (TI - TCP - TE - Pago Directo)			
11234567	26-10-2015 13:16:59	10.0.1.150	pv1rprweb042	com.previred.wpersonas	Selección de "Pago Directo"			
11234567	26-10-2015 13:17:00	10.0.1.150	pv1rprweb042	com.previred.wpersonas	--implícito-- Login en "Rol Persona (wpersonas)"			
11234567	26-10-2015 13:17:00	10.0.1.150	pv1rprweb042	com.previred.wpersonas	Muestra maletín de "Pago Directo - Apv-Cuenta2"			
11234567	26-10-2015 13:17:47	10.0.1.150	pv1rprweb042	com.previred.wpersonas	Flujo de Ingreso de "Depósito Directo - APV Cuenta2)			
11234567	26-10-2015 13:18:42	10.0.1.150	pv1rprweb042	com.previred.wpersonas	Flujo de Modificación de Pago Directo - Cuenta 2 -APV			
11234567	26-10-2015 13:18:42	10.0.1.150	pv1rprweb042	com.previred.wpersonas	Inserta Nómina de trabajador (Empleador Persona)			
11234567	26-10-2015 13:18:42	10.0.1.150	pv1rprweb042	com.previred.wpersonas	Muestra maletín de "Pago Directo - Apv-Cuenta2"			
11234567	26-10-2015 13:18:55	10.0.1.150	pv1rprweb042	com.previred.wpersonas	Cambia Forma de Pago (Empleador Persona)			
11234567	26-10-2015 13:18:55	10.0.1.150	pv1rprweb042	com.previred.wpersonas	Inserta/Actualiza información de folios Persona			
11234567	26-10-2015 13:18:58	10.0.1.150	pv1rprweb042	com.previred.wpersonas	Paso de ir a mi banco "Redirección al banco o generación cupón"			
11234567	26-10-2015 13:20:50	10.0.1.150	pv1rprweb042	com.previred.wpersonas	Muestra maletín de "Pago Directo - Apv-Cuenta2"			
11234567	26-10-2015 13:20:55	10.0.1.150	pv1rprweb042	com.previred.wpersonas	Ver/Imprimir Planillas Pagadas (Empleador Persona)			

Ilustración 35: Resultado de trazabilidad de un usuario específico.

La forma de obtener los resultados expuestos en la Ilustración 35, no tiene mayor complejidad que realizar una consulta a la base de datos por un determinado usuario en un determinado día, como se presenta en la Ilustración 36.

[8] Los datos personales mostrados en la Ilustración 35 e Ilustración 36 como Rut e IP son datos ficticios, los originales fueron borrados por confidencialidad de la información.

```
SELECT  rut
        ,fecha_hora
        ,ip
        ,servidor
        ,tipo_log
        ,accion
        ,dispositivo
        ,CONCAT(so_nombre,' ',so_version) AS so
        ,CONCAT(navegador,' ',navegador_version) AS navegador
FROM    webLog_trazabilidad_usuarios
where   rut='11234567'
        AND fecha='2015-10-26'
ORDER BY fecha_hora
```

Ilustración 36: Consulta SQL de trazabilidad de un usuario específico.

7.1.1.2. PN-2 Identificación de usuarios de un determinado navegador web

Esta pregunta de negocio hace referencia a identificar el detalle de todos los usuarios que ingresan al sitio web de la organización por una determinada versión de navegador web. Este tipo de información es muy relevante para el área comercial, ya que muchos productos que se incorporan están soportados desde cierta versión de navegadores en adelante, por ende, se debe determinar quiénes de los usuarios que ingresan al sitio web están usando navegadores no soportados, ya que dichos usuarios deben ser contactados por el área comercial para informarles de los cambios y ayudarlos a usar una versión más nueva de su navegador.

La Ilustración 37 muestra el detalle de los usuarios[9] que han ingresado al sitio por una determinada versión de navegador web.

Rut	Dv	Nombres	Ap paterno	E-Mail	Servidor	Fecha y Hora	Acción	Dispositivo	SO	Versión	Navegador	Versión
12248208	1	ANIBAL ARTURO	CONTRERAS	anibal@mail.com	pv1rprweb030	01-10-2015 0:04:07	Login al sitio web	Personal computer	Linux		Chrome	11
8775593	2	CARLOS MAURICIO	ROJAS	carlos@mail.com	pv1rprweb039	01-10-2015 0:10:25	Login al sitio web	Personal computer	Windows XP		Chrome	14
10496257	2	MARCELA KATHERINE	ROGEL	marcela@mail.com	pv1rprweb035	01-10-2015 8:02:21	Login al sitio web	Personal computer	Windows 7		Chrome	18
16961403	2	VICKY ROSE ELIZ	BUSTOS	vicky.bustos@mail.com	pv1rprweb035	01-10-2015 8:13:54	Login al sitio web	Personal computer	Windows XP		Chrome	22
12877397	5	KARINA PATRICIA	PAINEFILO	karina@mail.com	pv1rprweb035	01-10-2015 8:17:38	Login al sitio web	Personal computer	Windows 7		Chrome	18
13204780	4	CARELIAN MARGARITA	CONTRERAS	carelian@mail.com	pv1rprweb031	01-10-2015 8:24:25	Login al sitio web	Personal computer	Mac OS X	10	Chrome	23
11367714	7	JESUS ISRAEL	VALDES	jesus@mail.com	pv1rprweb030	01-10-2015 8:26:53	Login al sitio web	Personal computer	Linux		Chrome	24
18190692	8	RODRIGO ALFREDO	GARCIA	rodrigo@mail.com	pv1rprweb040	01-10-2015 8:27:02	Login al sitio web	Personal computer	Windows 7		Chrome	12
8804181	K	RUTH NOEMI	ESCOBAR	ruth@mail.com	pv1rprweb040	01-10-2015 8:37:57	Login al sitio web	Personal computer	Windows 7		Chrome	29
17076132	4	MICHAEL WILLIAM	TRONCOSO	michael@mail.com	pv1rprweb038	01-10-2015 8:46:01	Login al sitio web	Personal computer	Windows 7		Chrome	26
13371755	2	CARLOS EDUARDO	CANIU	carlos@mail.com	pv1rprweb033	01-10-2015 8:46:29	Login al sitio web	Personal computer	Windows XP		Chrome	22
15559089	0	CRISTIAN ANDRES	MUNOZ	cristian@mail.com	pv1rprweb033	01-10-2015 8:46:29	Login al sitio web	Personal computer	Windows XP		Chrome	22
16180312	K	MARIANA MARLEN	GASEP	mariana@mail.com	pv1rprweb033	01-10-2015 9:20:19	Login al sitio web	Personal computer	Windows XP		Chrome	29
12756861	8	MARIANELA OLGA	TORO	marianela@mail.com	pv1rprweb033	01-10-2015 9:21:02	Login al sitio web	Personal computer	Windows XP		Chrome	13
13266769	1	MAURICIO RODRIGO	MARIPAN	mauricio@mail.com	pv1rprweb033	01-10-2015 9:43:29	Login al sitio web	Personal computer	Windows XP		Chrome	25

Ilustración 37: Resultado de usuarios de un determinado navegador web.

[9] Los datos personales mostrados en la Ilustración 37 como Rut, dv, nombres, apellidos y e-mail son datos ficticios, los originales fueron borrados por confidencialidad de la información.

La forma de obtener los resultados que se muestran en la Ilustración 37, es sólo una consulta SQL sobre los datos de trazabilidad (ya cargados previamente) junto con los datos de contactabilidad de usuarios. La consulta es presentada en la Ilustración 38.

```
SELECT t.rut
      ,u.dv
      ,u.nombres
      ,u.ap_paterno
      ,u.email
      ,t.servidor
      ,t.fecha_hora
      ,t.accion
      ,t.dispositivo
      ,t.so_nombre
      ,t.so_version
      ,t.navegador
      ,t.navegador_version
FROM webLog_trazabilidad_usuarios t
      JOIN usuarios u on t.rut=u.rut
WHERE t.accion='Login al sitio web'
    AND t.navegador LIKE 'Chrome'
    AND t.navegador_version<=30
    AND t.dispositivo='Personal computer'
ORDER BY t.fecha_hora
```

Ilustración 38: Consulta SQL de usuarios de un determinado navegador web.

7.1.1.3. PN-3 Carga de usuarios en la granja de servidores web

Esta pregunta de negocio hace referencia a la carga que posee la granja de servidores web de la organización, y por carga hace referencia al número de usuarios que está operando concurrentemente en los distintos servicios web. Esta información es relevante ya que permite dimensionar la cantidad de usuarios que usan el sitio web y a la vez entender el comportamiento de carga asociado a ellos. Saber y entender cómo es el comportamiento de la concurrencia de usuarios permite realizar pruebas de estrés más cercanas a la realidad para los nuevos servicios web de la compañía. Actualmente existe un pool de servidores que conforman la granja web, el entender la cantidad real de usuarios que operan en el sitio web ayuda a mejorar la estimación de la real cantidad de servidores que se necesitan y no caer en sobre-estimación o sub-estimación de la infraestructura.

El gráfico presentado en la Ilustración 39 muestra la carga de usuarios concurrentes operando en el sitio web en un determinado punto en el tiempo.

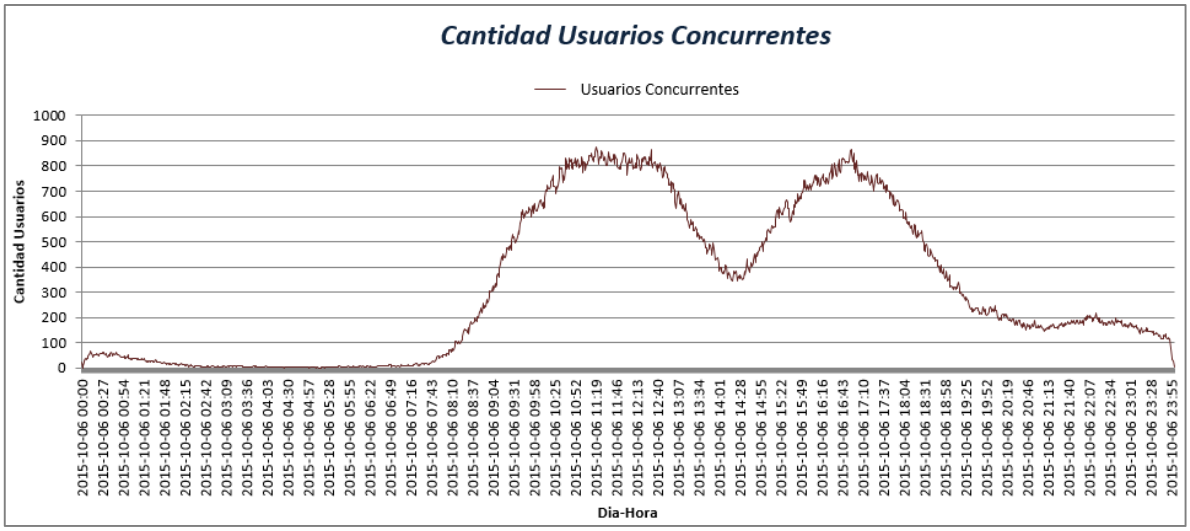


Ilustración 39: Gráfico de carga de usuarios concurrentes.

De la Ilustración 39 se puede ver que el día 06 de octubre del 2015 hubo un peak de 900 usuarios concurrentes trabajando en el sitio.

Para obtener la información anterior, sólo es necesario una consulta bastante simple y sin más complejidad que la mostrada en la Ilustración 40.

```
SELECT fecha,hora,minuto,COUNT(DISTINCT rut) as cantidad_usuarios
FROM   webLog_trazabilidad_usuarios
WHERE  periodo='2015-10'
GROUP BY fecha,hora,minuto
ORDER BY fecha,hora,minuto
```

Ilustración 40: Consulta SQL de carga de usuarios concurrentes.

7.1.1.4. PN-4 Flujos del sitio web con dificultad

Esta pregunta de negocio hace referencia a entender qué flujos del sitio web presentan dificultad para los usuarios. Actualmente existen muchos flujos de navegación dentro de los distintos portales web de la organización, de los cuales existe un desconocimiento de cuán fácil o difícil son estos para los usuarios, por esta razón, el tener una visión de los flujos que presentan mayor complejidad, ya sea porque son confusos de seguir o tienen algún error en su navegación, sería de mucha importancia a la hora la creación de nuevos flujos o mejorar los ya existentes.

En la Ilustración 41 se muestra el flujo de recuperación de clave del sitio web de la organización, siempre se ha sabido que este flujo presenta problemas, pero jamás se han identificados casos específicos que permitan realizar un rastreo del porqué dicho flujo presenta dificultad.

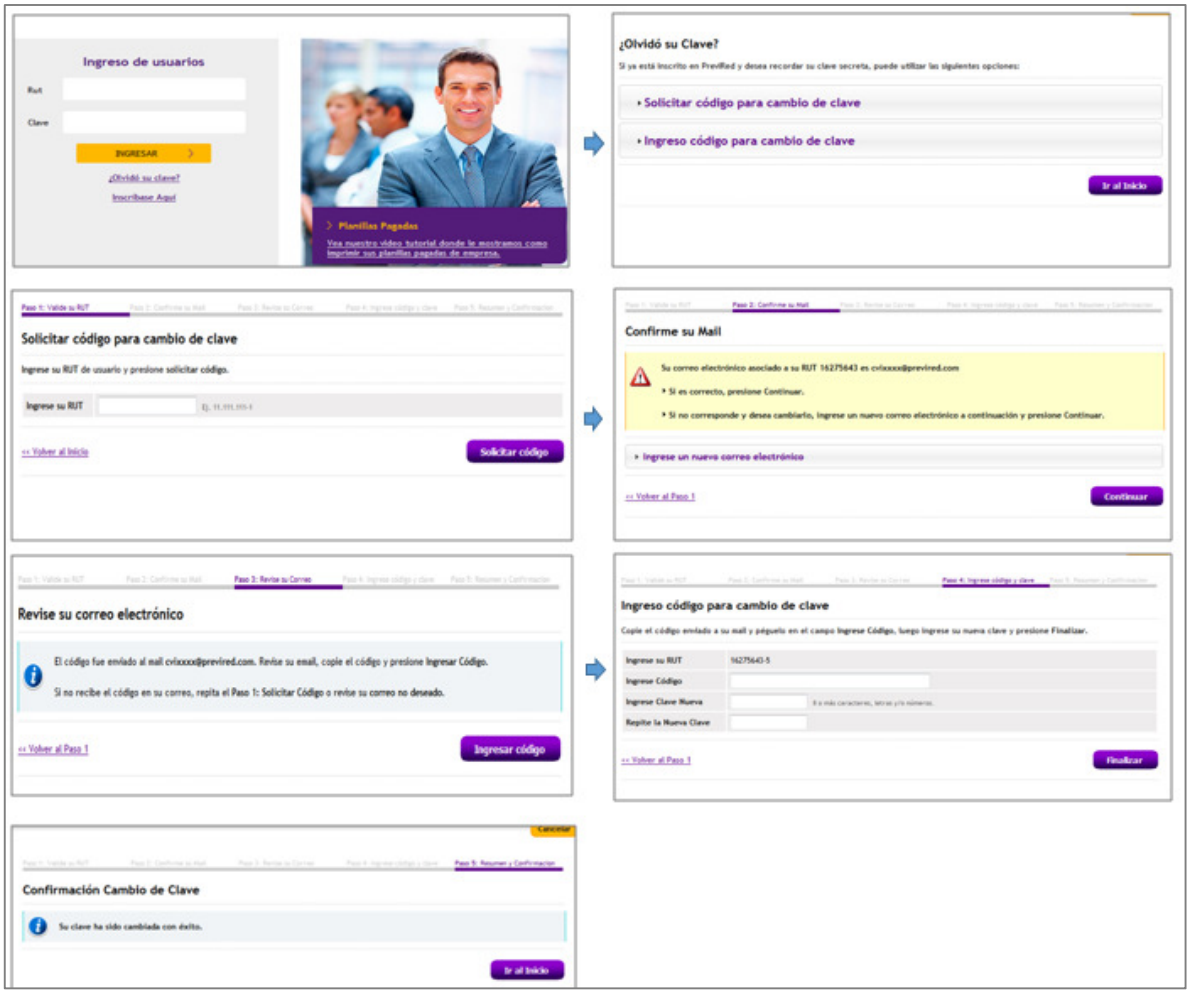


Ilustración 41: Flujo de navegación de recuperación de clave.

Lo primero que se debe revisar para analizar los casos del flujo es identificar lo que se registra en el archivo de log del portal web (wPortalLog).

Tipo Log : com.previred.wportal

Log detalle	Accion
- Ejecutando programa prgsolicitacodigo	Dentro del flujo "Olvido su clave" Solicita código "Paso 1: Valide su RUT"
- Ejecutando programa prgsolicitacodigo2	Dentro del flujo "Olvido su clave" Solicita código "Paso 2: Confirme su Mail"
- Ejecutando programa prgsolicitacodigo3	Dentro del flujo "Olvido su clave" Solicita código "Paso 3: Revise su Correo"
- Ejecutando programa prgsolicitacodigo4	Dentro del flujo "Olvido su clave" Solicita código "Paso 4: Ingrese código y clave"
- Ejecutando programa prgsolicitacodigo5	Dentro del flujo "Olvido su clave" Solicita código "Paso 5: Resumen y Confirmación"

Ilustración 42: Registro de acciones del flujo de recuperación de clave.

Finalmente se pueden revisar dos casos donde cada usuario realiza el flujo dos veces, el primero logrando exitosamente terminar el proceso de recuperación de su clave y el segundo teniendo error en ambos intentos.

Primer Caso Usuario: 8567234

	Servidor	IP	rut	fecha_hora	Diff Time	accion
1 Intento	pv1rprweb035	10.0.1.150	8567234	13-10-2015 12:03:49		Dentro del flujo "Olvido su clave" Solicita código "Paso 1: Valide su RUT"
	pv1rprweb035	10.0.1.150	8567234	13-10-2015 12:03:59		Dentro del flujo "Olvido su clave" Solicita código "Paso 2: Confirme su Mail"
	pv1rprweb035	10.0.1.150	8567234	13-10-2015 12:04:09		Dentro del flujo "Olvido su clave" Solicita código "Paso 3: Revise su Correo"
2 Intento	pv1rprweb035	10.0.1.150	8567234	13-10-2015 12:05:23	1 min 14 seg	Dentro del flujo "Olvido su clave" Solicita código "Paso 1: Valide su RUT"
	pv1rprweb035	10.0.1.150	8567234	13-10-2015 12:05:34		Dentro del flujo "Olvido su clave" Solicita código "Paso 2: Confirme su Mail"
	pv1rprweb035	10.0.1.150	8567234	13-10-2015 12:05:41		Dentro del flujo "Olvido su clave" Solicita código "Paso 3: Revise su Correo"
	pv1rprweb035	10.0.1.150	8567234	13-10-2015 12:05:43		Dentro del flujo "Olvido su clave" Solicita código "Paso 4: Ingrese código y clave"
	pv1rprweb035	10.0.1.150	8567234	13-10-2015 12:06:06		Dentro del flujo "Olvido su clave" Solicita código "Paso 5: Resumen y Confirmación"

Segundo Caso Usuario: 12543876

	Servidor	IP	rut	fecha_hora	Diff Time	accion
1 Intento	pv1rprweb040	10.0.1.170	12543876	24-10-2015 12:28:38		Dentro del flujo "Olvido su clave" Solicita código "Paso 1: Valide su RUT"
	pv1rprweb040	10.0.1.170	12543876	24-10-2015 12:28:52		Dentro del flujo "Olvido su clave" Solicita código "Paso 2: Confirme su Mail"
	pv1rprweb040	10.0.1.170	12543876	24-10-2015 12:29:12		Dentro del flujo "Olvido su clave" Solicita código "Paso 3: Revise su Correo"
2 Intento	pv1rprweb040	10.0.1.170	12543876	24-10-2015 12:29:24	22 seg	Dentro del flujo "Olvido su clave" Solicita código "Paso 1: Valide su RUT"
	pv1rprweb040	10.0.1.170	12543876	24-10-2015 12:29:38		Dentro del flujo "Olvido su clave" Solicita código "Paso 2: Confirme su Mail"
	pv1rprweb040	10.0.1.170	12543876	24-10-2015 12:30:43		Dentro del flujo "Olvido su clave" Solicita código "Paso 3: Revise su Correo"
	pv1rprweb040	10.0.1.170	12543876	24-10-2015 12:32:15	2 min 32 seg	Dentro del flujo "Olvido su clave" Solicita código "Paso 4: Ingrese código y clave"

Ilustración 43: Resultado de casos de flujo de navegación de recuperación de clave.

De la Ilustración 43 se puede observar que existe un paso donde el usuario[10] queda esperando un tiempo considerable, es el paso cuando se envía el código de autorización a su correo, por lo que se puede concluir que en este flujo existe un problema relacionado al tiempo de envío del correo desde los servidores al cliente, provocando que no pueda terminar el flujo, o tenga que repetirlo muchas veces para completarlo.

Para obtener los usuarios que lograron terminar el flujo y aquellos que no lograron, se realiza una consulta SQL a la base de datos como la que se presenta en Ilustración 44.

```
-- USUARIOS QUE NO LOGRARON TERMINAR EL FLUJO EN LA MISMA SESSION WEB
SELECT t.rut,t.fecha_hora
FROM   webLog_trazabilidad_usuarios t
WHERE     t.periodo='2015-10'
      AND t.log_detalle_accion = '- Ejecutando programa prgverificacodigo'
      AND NOT EXISTS (
                  select 1
                  from webLog_trazabilidad_usuarios t2
                  WHERE     t2.rut=t.rut
                        AND t2.jbossId = t.jbossId
                        AND t2.fecha = t.fecha
                        AND t2.log_detalle_accion = '- Ejecutando programa prgverificacodigo5'
                  )

-- USUARIOS QUE SI LOGRARON TERMINAR EL FLUJO
SELECT t.rut,t.fecha_hora
FROM   webLog_trazabilidad_usuarios t
WHERE     t.periodo='2015-10'
      AND t.log_detalle_accion = '- Ejecutando programa prgverificacodigo5'
```

Ilustración 44: Consulta SQL de flujo de navegación de recuperación de clave.

[10] Los datos personales mostrados en la Ilustración 43 como Rut e IP son datos ficticios, los originales fueron borrados por confidencialidad de la información.

7.1.2. Evaluación de resultados caso de uso “Detección de Errores”

Este caso de uso hace referencia a detectar los errores que suceden en los diferentes portales web de la compañía con el objetivo de tener una temprana detección de ellos y a la vez un registro del impacto que cada uno de ellos pudo haber tenido.

7.1.2.1. PN 1 Cantidad de errores más presente en el sitio web

Esta pregunta de negocio hace referencia a detectar aquellos errores que más se presentan en los usuarios que utilizan los portales web de la compañía. El tener esta información es de gran relevancia dentro de la organización, ya que da un marco de trabajo con una prioridad, es decir, debido a que en la mayoría de las organizaciones el tiempo para trabajar en mejorar de los sitios web es escaso, se debe priorizar qué errores o problemas serán corregidos en los proyectos de mantención, el poseer los errores que más afectan entrega una clara priorización de trabajo.

La Ilustración 45 entrega un detalle de los 15 errores que más se presentan dentro del sitio web.

Detalle de Error	Cantidad
Warning: Sin sesion	294.372
Warning: Sin estado	189.321
com.previred.fce.generales.WSException: No se encontr? el programa [null]	126.730
com.previred.fce.generales.WSException: Usuario no existe o consulta sin resultados	21.751
com.previred.fce.generales.WSException: No se encontr? el programa []	14.339
com.previred.fce.generales.WSException: Invalid column index 21.	10.239
com.previred.fce.generales.WSException: Invalid column name acepta_PA_TCP.	8.677
java.lang.NullPointerException	5.252
com.previred.fce.generales.WSException	1.411
com.previred.fce.generales.WSException: Unable to get managed connection for jdbc/dsSqlServer; - nested throwable: (javax.resource.ResourceException:	975
com.previred.fce.generales.WSException: En este momento no lo podemos atender	730
java.lang.IllegalStateException: setAttribute: La Sesi?n ya ha sido invalidada	716
org.apache.jasper.JasperException: Ha sucedido una excepci?n al procesar la p?gina JSP /mandato/mandato_paso_4.jsp en l?nea 57	502
com.previred.fce.generales.WSException: No se encontr? el programa [prglogintcp]	397
com.previred.fce.generales.WSException: Servicio no existe - MSG: Servicio srvsesionlogin no existe en XML para el programa prgreenviocodigo	386
com.previred.fce.generales.WSException: Usuario bloqueado	314
com.previred.fce.generales.WSException: For input string: "null"	274

Ilustración 45: Resultado de errores más presentes en sitio web.

La información de errores, al igual que la de trazabilidad, está almacenada en base de datos, el obtener la mayor cantidad errores es básicamente una simple consulta como la que se presenta en la Ilustración 46.

```
SELECT log_detalle_error, COUNT(1) as Cantidad
FROM   webLog_error_sitio
WHERE  periodo='2015-10'
GROUP BY log_detalle_error
ORDER BY COUNT(1) DESC
```

Ilustración 46: Consulta SQL de errores más presentes en sitio web.

7.1.2.2. PN 2 Comportamiento de un error específico en la granja web

Esta pregunta de negocio hace referencia a que una vez identificado el error, se desea ver cómo es el comportamiento de ese error en la granja de servidores web, ya que esto permite saber:

- En cuántos servidores web se presenta el error específico.
- El tiempo que el error ocurrió, es decir desde y hasta qué fecha y hora sucedió.
- Saber si el error fue en transversal en el sitio, o sólo afectó a ciertos servidores.

La Ilustración 47 muestra el detalle de los 15 errores más comunes, pero se destaca uno en especial que hace referencia a un proyecto que recién pasó a producción.

Detalle de Error	Cantidad
Warning: Sin sesion	294.372
Warning: Sin estado	189.321
com.previred.fce.generales.WSException: No se encontr? el programa [null]	126.730
com.previred.fce.generales.WSException: Usuario no existe o consulta sin resultados	21.751
com.previred.fce.generales.WSException: No se encontr? el programa []	14.339
com.previred.fce.generales.WSException: Invalid column index 21.	10.239
com.previred.fce.generales.WSException: Invalid column name acepta_PA_TCP.	8.677
java.lang.NullPointerException	5.252
com.previred.fce.generales.WSException	1.411
com.previred.fce.generales.WSException: Unable to get managed connection for jdbc/dsSqlServer; - nested throwable: (javax.resource.ResourceException:	975
com.previred.fce.generales.WSException: En este momento no lo podemos atender	730
java.lang.IllegalStateException: setAttribute: La Sesi?n ya ha sido invalidada	716
org.apache.jasper.JasperException: Ha sucedido una excepci?n al procesar la p?gina JSP /mandato/mandato_paso_4.jsp en l?nea 57	502
com.previred.fce.generales.WSException: No se encontr? el programa [prglogintcp]	397
com.previred.fce.generales.WSException: Servicio no existe - MSG: Servicio srvsesionlogin no existe en XML para el programa prgreenviocodigo	386
com.previred.fce.generales.WSException: Usuario bloqueado	314
com.previred.fce.generales.WSException: For input string: "null"	274

Ilustración 47: Selección de error específico dentro de los errores más comunes.

El gráfico presentado en la Ilustración 48 muestra cómo ese error se comportó en la granja de servidores web. Se puede concluir no fue un error transversal en el sitio, ya que sólo afectó a dos servidores durante la mañana del día 15 de octubre, justo el día del paso a producción del proyecto "Nuevo Pago Atrasado para Rol TCP". Finalmente se entiende que el control de cambio asociado al paso a producción no fue aplicado en dos servidores, en los cuales se presentó el error.

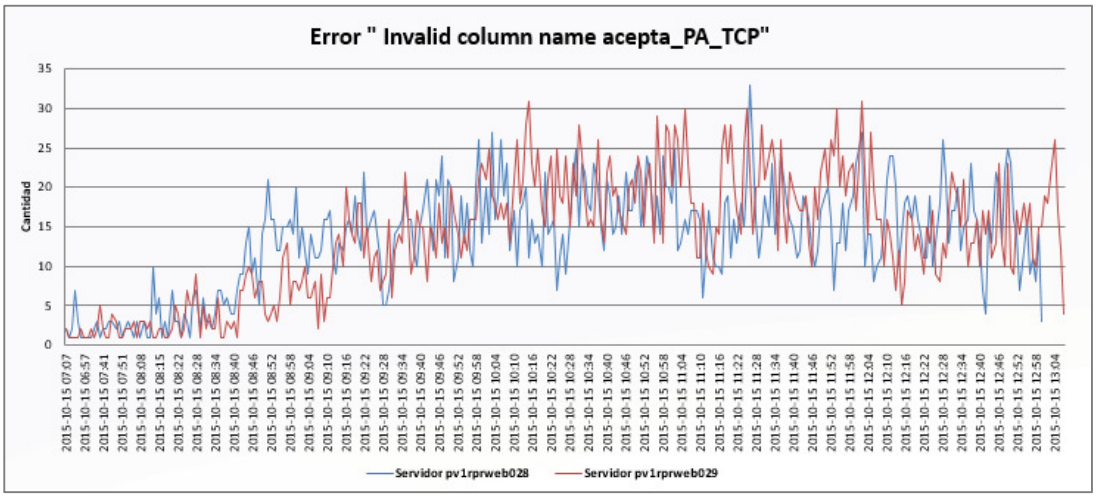


Ilustración 48: Gráfico de comportamiento de un error específico en la granja web.

Para la obtención de información anterior, sólo es necesario una consulta simple a la base de datos donde es cargada la información de errores. La consulta es presentada en la Ilustración 49.

```
SELECT servidor,fecha,hora,minuto,COUNT(1) as cantidad
FROM   webLog_error_sitio
WHERE  periodo='2015-10'
       AND log_detalle_error='com.previred.fce.generales.WSException: Invalid column name acepta_PA_TCP.'
GROUP BY fecha,hora,minuto
ORDER BY fecha,hora,minuto
```

Ilustración 49: Consulta SQL de comportamiento de un error específico.

7.1.2.3. PN-3 Detalle de personas con mayor número de errores

Esta pregunta de negocio hace referencia a las personas que presentan mayor cantidad de errores en su operatoria en el sitio web, permitiendo a la empresa dar atención personalizada a dichos usuarios, entendiendo que tener muchos errores en su operatoria se debe a que desconocen cómo funciona el sitio y necesitan ayuda.

La Ilustración 50 muestra el detalle de los usuarios[11] que presentan el mayor número de errores dentro del sitio web de la organización.

Rut	dv	nombres	ap_paterno	email	Cantidad
17735458	9	VICTOR MIGUEL	IBANEZ	victor@mail.com	97
10478876	9	VILMA FELISA	CATRIFILO	vilma@mail.com	77
13251357	0	ANDRES ALFONSO	MARTIN	andres@mail.com	75
19260213	0	INA ANDREA	MARQUINA	ina@mail.com	69
14214312	7	GREGORY ALEXIS	BUSCH	gregory@mail.com	61
18207229	K	ALFREDO ALEJAN	GALLARDO	alfredo@mail.com	57
10292159	3	SAMUEL ANTONIO	CANAS	samuel@mail.com	56
16662146	1	EUGENIA DEL PILAR	CARES	eugenia@mail.com	54
16069998	1	JOSE MERCEDES	ROJAS	jose@mail.com	53
15683772	5	HECTOR HORACIO	NAVARRO	hector@mail.com	52
13594453	K	RICALDI	VALDEBENITO	ricaldi@mail.com	52
8310949	1	PATRICIO HERNAN	OYARZO	patricio@mail.com	52
10668721	8	ALEX EDUARDO	EBNER	alex@mail.com	49
13728475	8	MARIELA IVY	CABRERA	mariela@mail.com	48
11080866	6	CAROLINA ALEJAN	ECHEVERRIA	carolina@mail.com	47
15144923	9	LUZMENIA NATALI	BERNAL	luzmenia@mail.com	47
19570422	8	MERY ANDREA	QUISPE	mery@mail.com	46
18630251	6	JACQUELINE NICOLE	SARRALDE	jacqueline@mail.com	46

Ilustración 50: Resultado de las personas con mayor número de errores.

[11] Los datos personales mostrados en la Ilustración 50 como Rut, dv, nombres, apellidos, e-mail son datos ficticios, los originales fueron borrados por confidencialidad de la información.

De igual forma, para obtener esta información sólo basta ejecutar una simple consulta sobre la tabla de errores, como la mostrada en la Ilustración 51.

```
SELECT e.rut,u.dv,u.nombres,u.ap_paterno,u.ap_materno,u.email, COUNT(1) as Cantidad
FROM   webLog_error_sitio e
       join usuarios u on e.rut=u.rut
WHERE  periodo='2015-10'
GROUP BY e.rut,u.dv,u.nombres,u.ap_paterno,u.ap_materno,u.email
ORDER BY COUNT(1) DESC
```

Ilustración 51: Consulta SQL de las personas con mayor número de errores.

7.2. Evaluación de escalabilidad

La evaluación de escalabilidad se realizó en base a una plataforma Cloud de Amazon Web Services (AWS), donde la infraestructura constaba de nueve nodos, ocho nodos de procesamiento y un nodo de administración. En esta plataforma se instaló Cloudera CDH versión 5 con los componentes de Hadoop MapReduce junto con HDFS.

Las pruebas se realizaron de forma incremental, comenzando con la ejecución en 1 nodo, 2, 4 y finalizando con 8 nodos. Estas pruebas fueron de dos tipos:

- **Tipo de prueba 1:** Pruebas de escalabilidad sobre archivos comprimidos en formato bzip2 en 1, 2, 4, y 8 nodos de procesamiento.
- **Tipo de prueba 2:** Pruebas de rendimiento sobre compresión de archivos (bzip2, gzip, texto), esta prueba se realizó sobre 8 nodos.

A continuación se muestra la Tabla 5 que representa un resumen del conjunto de pruebas realizado en ambiente Cloud AWS.

Tabla 5: Tabla de pruebas de escalabilidad.

Tipo Prueba	N° Prueba	N° Nodos	MapReduce	Tipo de Archivos
Tipo 1	Prueba-1	1 Nodo	Job Trazabilidad Job de Login Job Errores	Comprimidos bzip2
Tipo 1	Prueba-2	2 Nodos	Job Trazabilidad Job de Login Job Errores	Comprimidos bzip2
Tipo 1	Prueba-3	4 Nodos	Job Trazabilidad Job de Login Job Errores	Comprimidos bzip2
Tipo 1	Prueba-4	8 Nodos	Job Trazabilidad Job de Login Job Errores	Comprimidos bzip2
Tipo 2	Prueba-5	8 Nodos	Job Trazabilidad	Comprimidos bzip2
Tipo 2	Prueba-6	8 Nodos	Job Trazabilidad	Comprimidos gzip
Tipo 2	Prueba-7	8 Nodos	Job Trazabilidad	Normales Texto

7.2.1. Infraestructura de pruebas

La infraestructura que se usó para las pruebas fue Cloud AWS (Amazon Web Services), donde se utilizó un total de nueve nodos, ocho de ellos para procesamiento y uno para administración. La herramienta Big Data utilizada fue Cloudera CDH v5.

- **Ocho Nodos de procesamiento:** Cada nodo de procesamiento tenía los elementos de Hadoop MapReduce y Hadoop HDFS.
- **Un Nodo de Administración:** El nodo de procesamiento tenía el elemento de Cloudera Manager, encargado de revisar el estado y las configuración del cluster de nodos.

La arquitectura es presentada en la Ilustración 52, donde se puede ver las siguientes características.

- **VPC (Virtual Private Cloud):** La red creada estaba en base a la subnet 10.0.0.0/24. Los nodos dentro de esta VPC partían en la IP 10.0.0.10 (nodo de administración), y continuaban en el segmento 10.0.0.11 – 10.0.0.18 (nodos de procesamiento).
- **Características nodos:** Los nodos se basan en el tipo de servidor de Amazon ec2.m3.large, con características de: 7.5 GB RAM, 2 CPU, 25 GB para Sistema Operativo, 30 GB para HDFS. El Sistema Operativo instalado fue RHEL 6.5 (Red Hat Enterprise Linux).

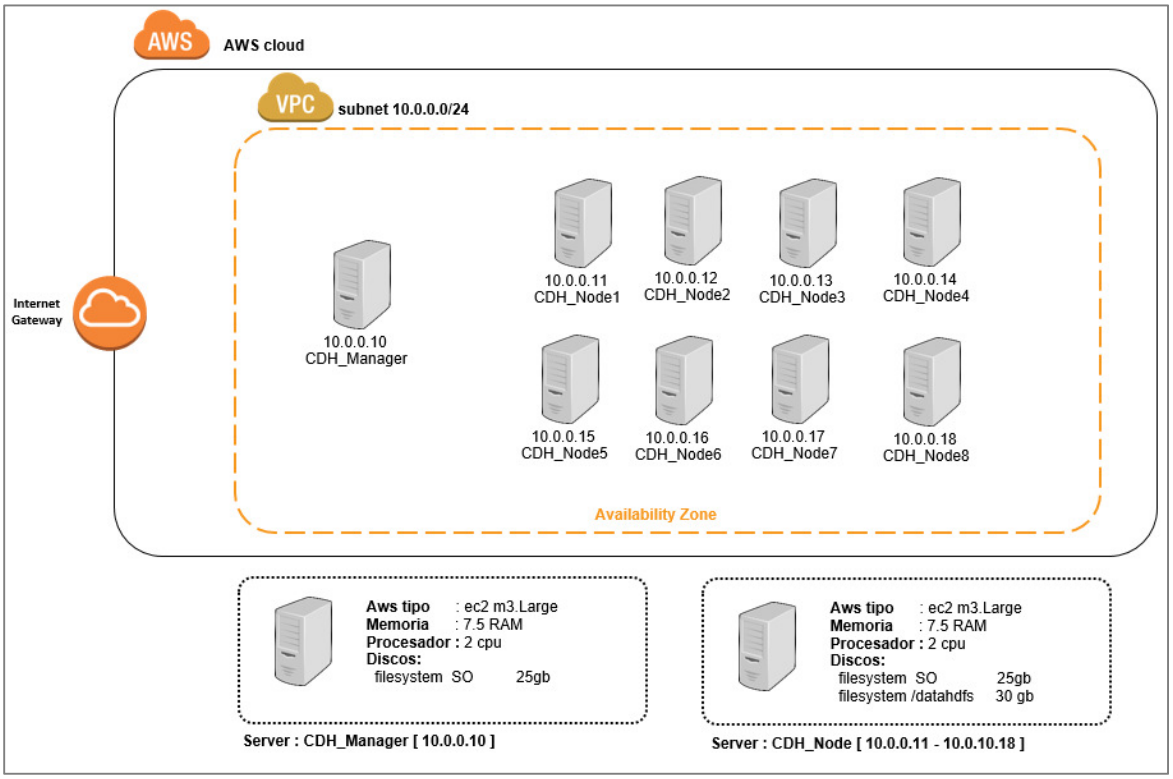


Ilustración 52: Diagrama de arquitectura Cloud AWS de pruebas.

La Ilustración 53 muestra la vista dentro de AWS del cluster de nodos ec2.m3.large.

Name	Instance ID	Instance Type	Availability Zone	Instance State
CDH_Manager	i-63ac1381	m3.large	sa-east-1a	running
CDH_Node1	i-bdaf105f	m3.large	sa-east-1a	running
CDH_Node2	i-ca7bc528	m3.large	sa-east-1a	running
CDH_Node3	i-f7d96015	m3.large	sa-east-1a	running
CDH_Node4	i-edd9600f	m3.large	sa-east-1a	running
CDH_Node5	i-174af3f5	m3.large	sa-east-1a	running
CDH_Node6	i-fc45fc1e	m3.large	sa-east-1a	running
CDH_Node7	i-2447fec6	m3.large	sa-east-1a	running
CDH_Node8	i-373980d5	m3.large	sa-east-1a	running

Ilustración 53: Vista de la arquitectura dentro de AWS.

7.2.2. Evaluación de Escalabilidad Archivos bzip2 en 1, 2, 4 y 8 Nodos

Las pruebas de escalabilidad se realizaron en base a la ejecución de los tres métodos MapReduce: “Job de Trazabilidad”, “Job de Login” y el “Job de Errores”, cada uno de ellos se ejecutó en 1, 2, 4 y 8 nodos de procesamiento. A continuación se muestran los resultados obtenidos por cada Job MapReduce.

7.2.2.1. Estadísticas de escalabilidad MapReduce Job Trazabilidad

En la Tabla 6 se muestra los resultados del “Job de Trazabilidad” ejecutado sobre 1, 2, 4 y 8 nodos de procesamiento.

Tabla 6: Tabla de resultados de escalabilidad del Job Trazabilidad.

Nombre	webLogTrazabilidadJob.jar
Descripción	trazabilidad proceso completo
Cantidad de archivos	886
formato	bzip2
Input records	288.895.061
Output records	37.274.896

Nodos	tiempo (hh:mm:ss)	Tiempo(seg)	Velocidad (Líneas/Min)
1	02:29:57	8.997	1.926.609
2	01:20:03	4.803	3.608.933
4	00:37:41	2.261	7.666.388
8	00:20:26	1.226	14.138.421

De la Tabla 6 se observa que el proceso tiene una entrada de 290 millones de registros, de los cuales post-procesamiento se obtienen 37 millones, estos representan la trazabilidad de los usuarios.

En el gráfico de duración del proceso, presentado en la Ilustración 54, se observa que el tiempo se reduce a la mitad cada vez que se dobla la cantidad de nodos de procesamiento. El escalamiento en la cantidad de nodos es mostrado en el eje X del gráfico de la Ilustración 54, así como también en el resto de los gráficos.

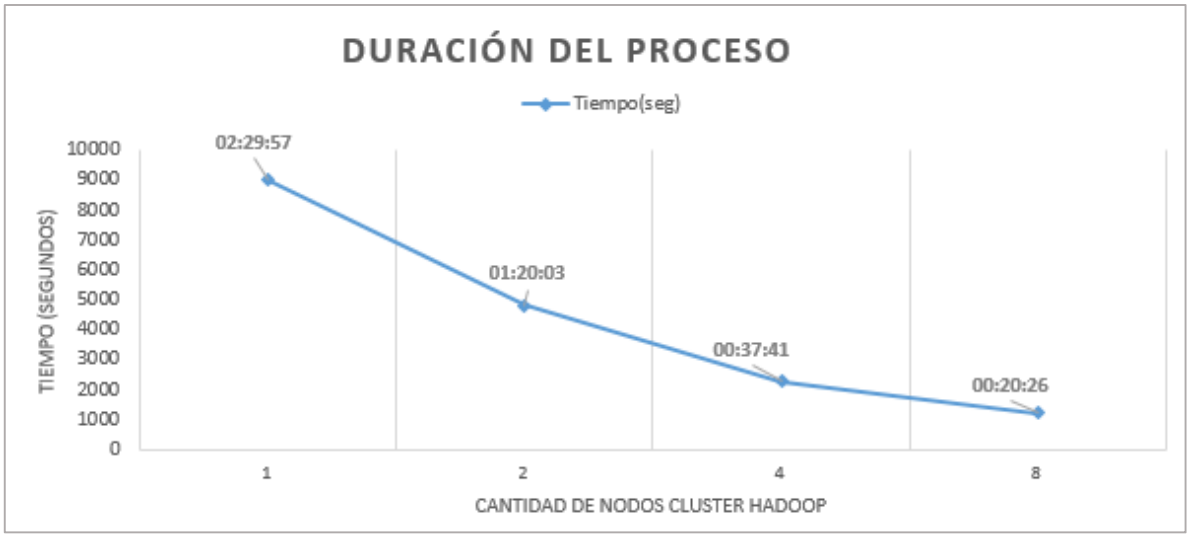


Ilustración 54: Gráfico de duración del Job Trazabilidad.

De igual forma la velocidad de procesamiento (cantidad de líneas procesadas por minuto), presentado en la Ilustración 55, aumenta casi linealmente cada vez que aumenta la cantidad de nodos de procesamiento, permitiendo suponer una escalabilidad lineal del proceso MapReduce.

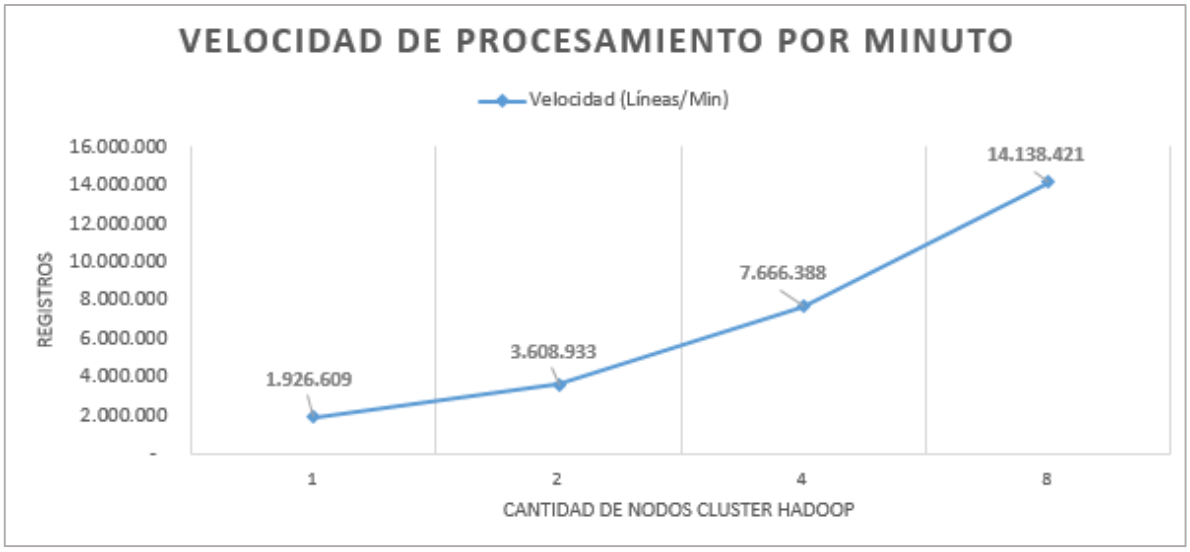


Ilustración 55: Gráfico de velocidad de procesamiento del Job Trazabilidad.

7.2.2.2. Estadísticas de escalabilidad MapReduce Job Login

En la Tabla 7 se presentan las estadísticas de “Job Login”, este proceso se ejecutó sobre 1, 2, 4 y 8 nodos de procesamiento.

Tabla 7: Tabla de resultados de escalabilidad del Job Login.

Nombre	webLogPreviredJobLogin.jar
Descripción	Login proceso completo
Cantidad de archivos	13
formato	bzip2
Input records	6.256.099
Output records	1.541.902

Nodos	tiempo (hh:mm:ss)	Tiempo(seg)	Velocidad (Líneas/Min)
1	00:04:02	242	1.551.099
2	00:02:02	122	3.076.770
4	00:01:51	111	3.381.675
8	00:01:20	80	4.692.074

Este método tiene un procesamiento de 6 millones de registros de entrada, de los cuales después del procesamiento se emiten 1.5 millones de registros. Estos registros tienen las características de estar pre-procesados, por ende el proceso MapReduce es muy rápido en su ejecución debido a que la cantidad de registros de entrada no es tan grande.

A continuación se muestran los gráficos de escalabilidad de este Job en los nodos de procesamiento, donde se puede ver en la Ilustración 56 que el tiempo de duración del proceso decrece a medida que se incorporan nodos al cluster de procesamiento.

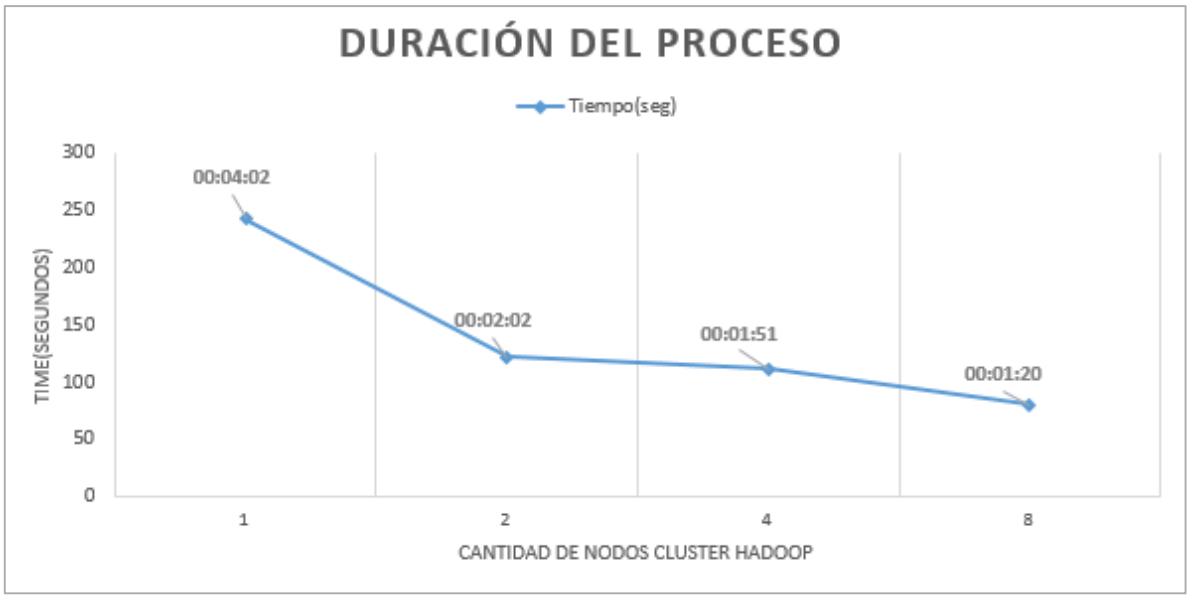


Ilustración 56: Gráfico de duración del Job Login.

Al ser la cantidad de datos no tan significativa ni en volumen ni complejidad, se observa que la escalabilidad no se ve reflejada de forma tan significativa en los tiempos de procesamiento. Aunque disminuye el tiempo de procesamiento a medida que se van aumentando los nodos, la disminución no es linealmente decreciente, así como tampoco la velocidad de procesamiento (cantidad de líneas por minuto) como se muestra en la Ilustración 57.

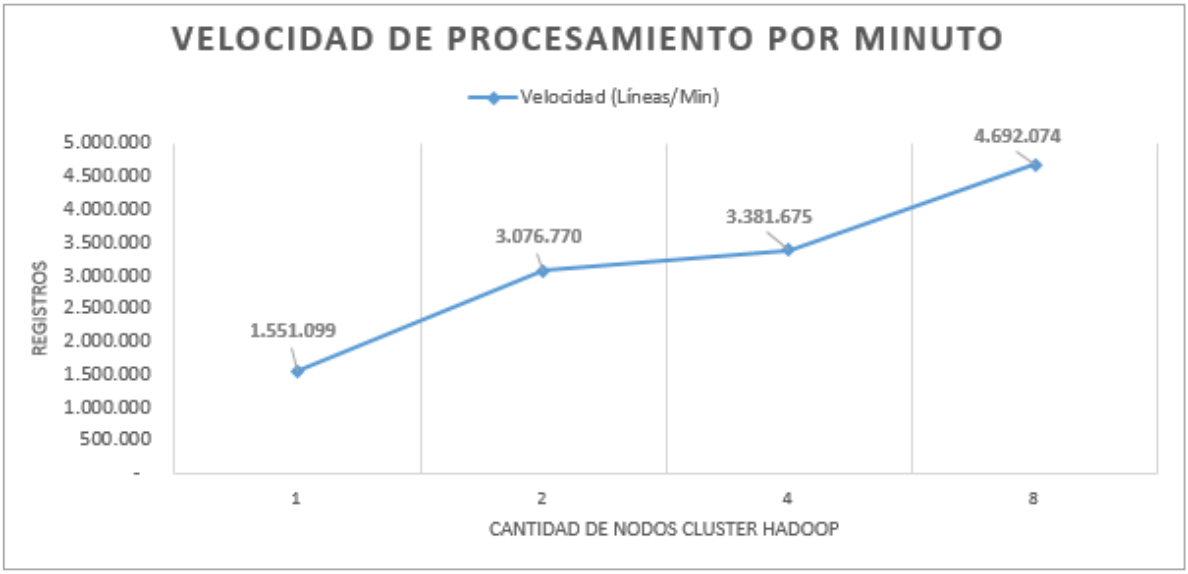


Ilustración 57: Gráfico de velocidad de procesamiento del Job Login.

7.2.2.3. Estadísticas de escalabilidad MapReduce Job Errores

En la Tabla 8 se presentan las estadísticas de "Job Errores", este proceso se ejecutó sobre 1, 2, 4 y 8 nodos de procesamiento.

Tabla 8: Tabla de resultados de escalabilidad del Job Errores.

Nombre	webLogPreviredJobError.jar
Descripcion	Errror proceso completo
Cantidad de archivos	886
formato	bz2
Input records	288.895.061
Output records	666.094

Nodos	tiempo (hh:mm:ss)	Tiempo(seg)	Velocidad (Líneas/Min)
1	02:19:00	8340	2.078.382
2	01:02:07	3727	4.650.846
4	00:31:32	1892	9.161.577
8	00:16:43	1003	17.281.858

Este Job procesa 290 Millones de registros de entrada que están distribuidos en 886 archivos a lo largo del cluster, finalmente después del procesamiento se obtiene 700 mil registros que representan la cantidad de errores encontrados en dichos archivos de logs de los servicios web de la organización.

Los gráficos del resultado de la ejecución de este Job en 1, 2, 4 y 8 nodos de procesamiento se muestran a continuación, se puede observar en la Ilustración 58 que el tiempo de duración del proceso decrece linealmente con el número de nodos que posee el cluster Hadoop.

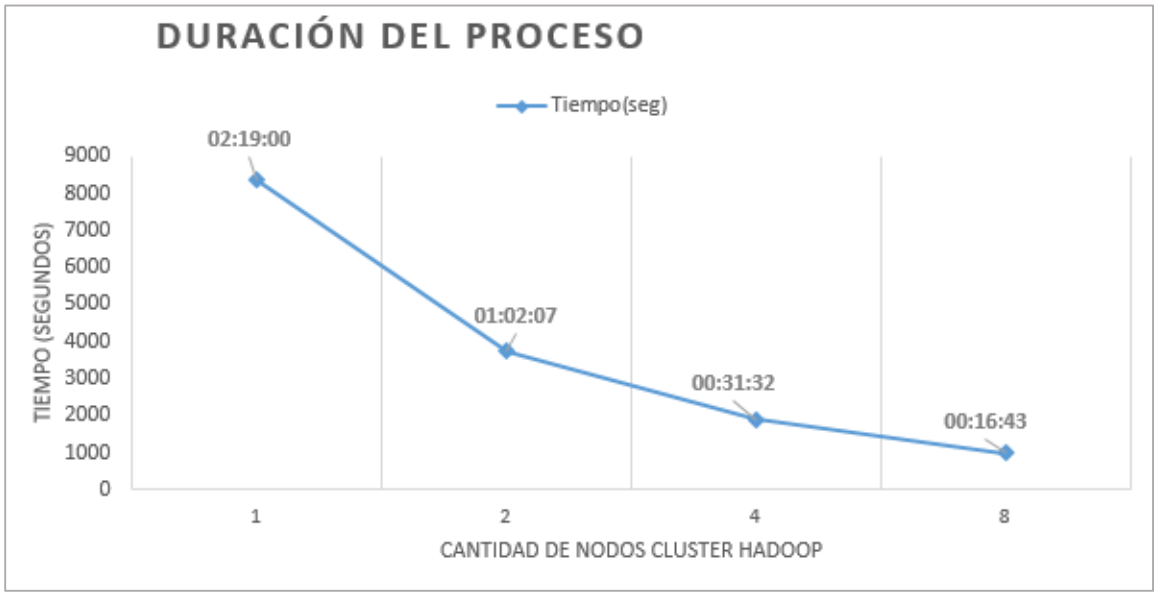


Ilustración 58: Gráfico de duración del Job Errores.

La velocidad de procesamiento (cantidad de líneas procesadas por minuto) que se muestra en la Ilustración 59, se observa que se incrementa linealmente a medida que el número de nodos aumenta.

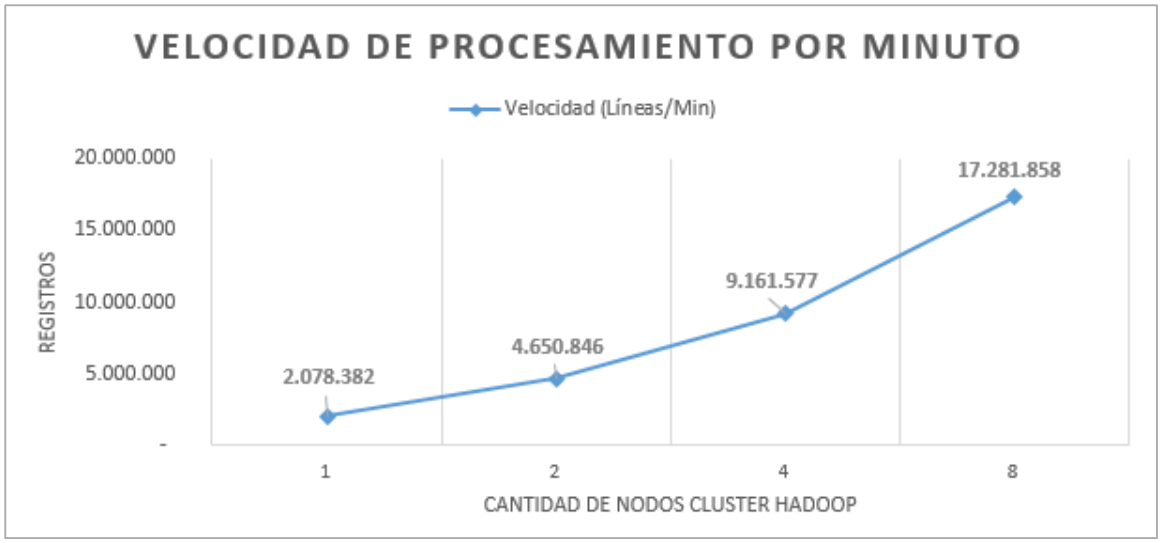


Ilustración 59: Gráfico de velocidad de procesamiento del Job Errores.

7.2.3. Evaluación de Compresión Archivos bzip2, gzip y texto en 8 nodos

La evaluación de compresión se basa en revisar cuál es la mejor forma de procesar los archivos en Hadoop. Actualmente los archivos están comprimidos en formato bzip2, ya que es el formato por defecto que posee la organización para almacenar información. Esto debido a que bzip2 posee la mejor tasa de compresión entre los formatos que existen en ambiente Linux, pero nadie asegura que sea el mejor formato para procesar información en Hadoop.

La evaluación se basa en ejecutar el Job de trazabilidad sobre el mismo conjunto de archivos, pero en tres formatos distintos: Archivos comprimidos en formato bzip2, archivos comprimidos en formato gzip y archivos normales en formato texto.

La Tabla 9 muestra los resultados de la ejecución del "Job de Trazabilidad" en el cluster de procesamiento de ocho nodos con el conjunto de archivos en los tres formatos: Bzip2, gzip y texto.

Tabla 9: Tabla de resumen de evaluación de compresión.

Nombre del job MapReduce	Cantidad de archivos	Formato Archivos	Tiempo (hh:mm:ss)	tamaño	Cantidad Filas Entrada	Cantidad Filas Salida	Tiempo CPU	velocidad
webLogTrazabilidadJob	886	bzip2	00:20:26	1.6 Gb	288.895.061	37.274.896	10.799.370	14.138.421
		gzip	00:17:23	2.7 Gb	288.895.061	37.274.896	8.095.420	16.619.083
		texto	00:19:25	44 Gb	288.895.061	37.274.896	8.533.540	14.878.716

Se observa en la Ilustración 60 que la primera gran y obvia diferencia es el volumen de procesamiento, siendo el formato texto quien posee el mayor volumen llegando a procesar 44 Gb de datos, y el menor, el formato bzip2 con 1,6 Gb de datos.

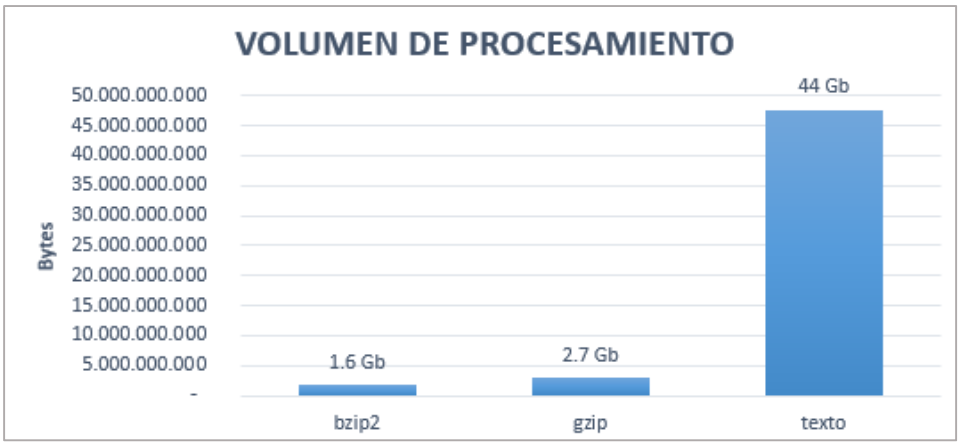


Ilustración 60: Gráfico de volumen de procesamiento de la evaluación de compresión.

Aunque el volumen de datos procesado es muy distinto entre los formatos, el tiempo de procesamiento es similar, como se muestra en la Ilustración 61, esto en gran medida se debe a que: Aunque el descomprimir los datos ocupa recursos (tiempo de CPU) el tener un volumen de datos muy grande (formato texto) consume otros tipos de recursos (ancho de banda de red, accesos a discos I/O, etc.).

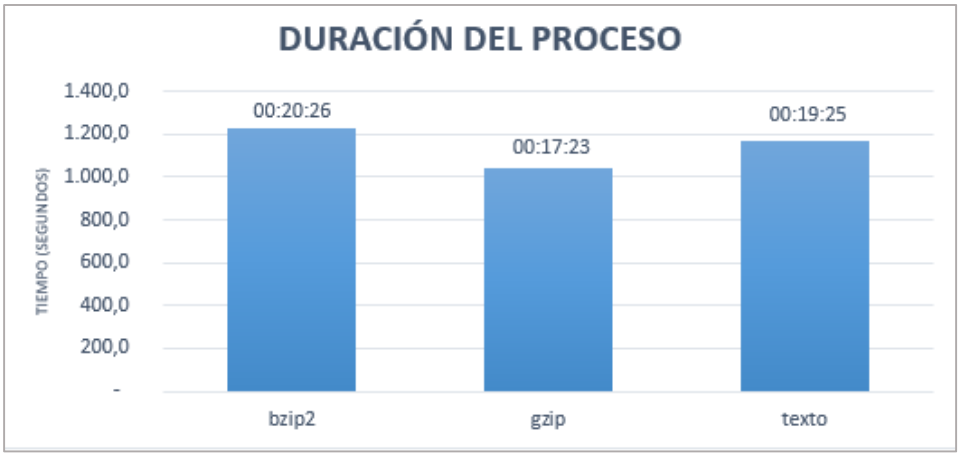


Ilustración 61: Gráfico de duración de procesamiento de la evaluación de compresión.

Se puede ver en la Ilustración 62 que la velocidad de procesamiento presenta un mejor rendimiento para archivos comprimidos con formato gzip, un formato que utiliza el algoritmo de deflación para realizar la compresión de los archivos. El tipo de archivos gzip tiene la particularidad de que sólo puede ser leído por una instancia de la fase Map de los procesos MapReduce, lo que en este caso pudiese haber sido la causa del mejor rendimiento, ya que los archivos en general eran de muy poco tamaño una vez comprimidos.

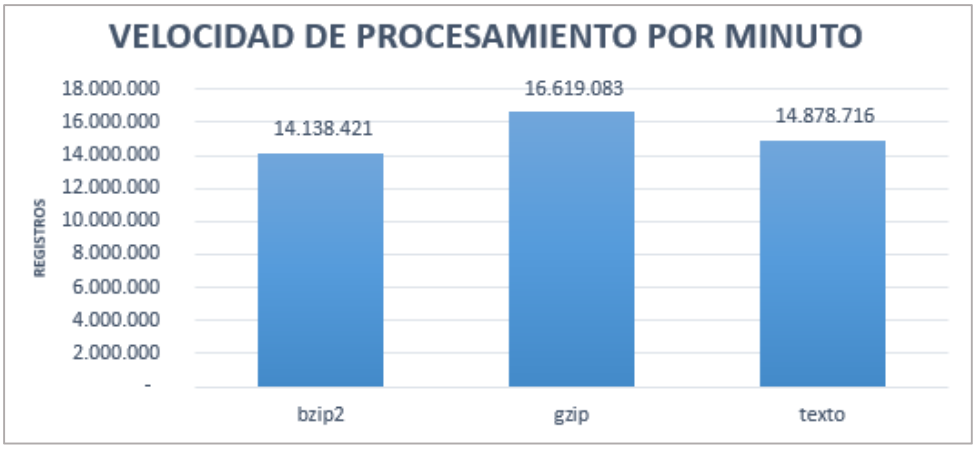


Ilustración 62: Gráfico de velocidad de procesamiento de la evaluación de compresión.

8. Conclusión

Este trabajo presentó una implementación de tecnologías Big Data aplicada en un entorno empresarial, donde se revisó las condiciones existentes en la organización y cómo el uso de esta nueva tecnología puede ayudar a entregar información de valor para el negocio de una organización. La tecnología utilizada fue Cloudera, principalmente por ser líder en el mercado, poseer soporte a nivel nacional y finalmente estar dentro del presupuesto de la organización.

Un punto importante tras la finalización de este trabajo es contrastar los objetivos específicos planteados al inicio del proyecto con los resultados obtenidos, la Tabla 10 entrega un resumen del nivel de cumplimiento de los objetivos específicos.

Tabla 10: Tabla de cumplimiento de objetivos específicos.

Objetivo Específico		Cumplimiento
1	Diseñar e implementar la capa de infraestructura que se soportará en las tecnologías Big Data.	Se realizó una implementación de Cloudera CDH v.5, tecnología de Big Data líder en el mercado.
2	Diseñar e implementar los métodos de recolección, de análisis y procesamiento de los archivos de logs semi-estructurados.	Se implementaron tres métodos MapReduce de procesamiento (Job Trazabilidad, Job Login, Job Errores), junto con ello también los métodos de pre-procesamiento y post-procesamiento de información.
3	Procesar la información de tal forma de que sea posible consultarla en un formato conocido para el área “arquitectura y gestión de datos” (ejemplo: Formato SQL).	La información una vez procesada es cargada en Infobright, la base de datos columnar de la organización. De este modo quedó accesible tanto para la herramienta de visualización, como también para consultas en formato SQL.
4	Identificar qué preguntas de negocio pueden ser respondidas con los actuales datos existentes en los logs, y al menos responder una pregunta utilizando la infraestructura y los métodos de procesamiento de los datos semi-estructurados creados.	Se identificaron dos casos de uso, el de trazabilidad y el caso de uso de errores, cada uno de ellos con un set de preguntas de negocio que fueron respondidas utilizando los métodos desarrollados en base a la tecnología Big Data implementada. Las preguntas de negocio respondidas fueron por ejemplo: - Trazabilidad de un usuario específico. - Usuarios que acceden al sitio por un navegador antiguo. - Errores más comunes en el sitio web.

De la tabla anterior se puede resumir que este proyecto logró un completo cumplimiento de cada uno de los objetivos específicos planteados al inicio. De igual forma se puede entender que se cumplió con el objetivo principal, el cual consistía en diseñar e implementar métodos extensibles de procesamiento de información semi-estructurada,

proveniente de la integración de múltiples logs de ambiente web, con el fin de obtener conocimiento relevante para la organización.

Otro punto importante dentro de este proyecto fue el conjunto de pruebas de escalabilidad, ya que con ellas se entregó la certeza de saber que la solución implementada es linealmente escalable. Esto quiere decir que el rendimiento que entregan los métodos desarrollados son directamente proporcionales al número de nodos que se incorporan al cluster de procesamiento, si el cluster posee 4 nodos y se quiere obtener el doble de rendimiento, basta con agregar 4 nodos extras a los ya existentes.

Si bien se pudiese pensar que es mejor procesar archivos en formato de texto plano en vez de archivos comprimidos, por el ahorro en el tiempo de descompresión, las pruebas entregaron una visión completamente distinta. Es más recomendable ante escenarios de procesamiento de grandes conjuntos de datos en ambientes distribuidos, hacerlo en formato comprimido, ya que aunque existe un pequeño costo en la tarea de descompresión (tiempo de CPU principalmente), se obtiene un conjunto de beneficios:

- Se reduce el consumo de ancho de banda de red, ya que los datos que se transmiten entre los nodos de procesamiento, al estar comprimidos, son de mucho menor tamaño.
- Se optimiza el acceso de escritura y lectura a disco (I/O), el cual es el recurso más caro en computación, por ende al estar los datos comprimidos el tamaño que se debe escribir o leer desde los discos físicos es mucho menor.
- Se reduce el uso de Disco (actualmente disco SSD) en lo relacionado a tamaño de almacenamiento, ya que archivos comprimidos (bzip2) utilizan el 3%-5% del tamaño real de los archivos.

A nivel macro, el cluster Cloudera se utilizó sólo como motor de procesamiento Hadoop, la información que finalmente se obtuvo de todo el procesamiento se cargó a la base de datos Infobright, el cual es data warehouse institucional. Esta base de datos entrega la característica ser Columnar y poseer una gran tasa de compresión de los datos cargados. La información dentro del data warehouse está unida a todos los modelos de gestión de la organización; Modelos de recaudación de pagos, Modelo de indicadores, Modelo de privilegios, modelo de Empresas, etc., con lo cual, a nivel de herramientas de visualización, se puede unir la información del procesamiento hadoop con toda la información de gestión de la organización.

Este proyecto tuvo impacto en los nuevos desarrollos tecnológicos de la compañía, el dar importancia a la información que se almacenará en los archivos logs, y cómo éstos luego serán procesados masivamente. Junto con ello nacieron nuevas preguntas e ideas de negocio para realizar en etapas futuras, como por ejemplo: Tener un monitoreo en línea de la cantidad de usuarios por flujo de sitio web, tener un control de errores en línea, tener una interfaz web para el área de Atención al Cliente con los usuarios que han presentado dificultades en los flujos de pagos del sitio web.

9. Bibliografía

[1] John Gantz, David Reinse «The Digital Universe in 2020: Big Data, Bigger Digital Shadows, and Biggest Growth in the Far East», 2012. URL: www.emc.com/leadership/digital-universe/index.htm (Última visita: Diciembre 2015).

[2] ISO/IEC JTC 1, Information technology, «ISO/IEC Big data Preliminary Report» 2014. URL: http://www.iso.org/iso/big_data_report-jtc1.pdf (Última visita: Diciembre 2015).

[3] Sanjay Ghemawat, Howard Gobioff, y Shun-Tak Leung, «The Google File System», 19th ACM Symposium on Operating Systems Principles, Lake George NY,USA, 2003.

[4] Apache HDFS, URL: https://hadoop.apache.org/docs/r1.2.1/hdfs_design.html (Última visita: Diciembre 2015).

[5] Fay Chang, Jeffrey Dean, Sanjay Ghemawat, Wilson C. Hsieh, Deborah A. Wallach, «Bigtable: A Distributed Storage System for Structured Data», Seventh Symposium on Operating System Design and Implementation, Seattle, WA, USA, 2006.

[6] Apache HBase, URL: http://hbase.apache.org/book.html#architecture (Última visita: Diciembre 2015).

[7] Amazon DynamoDB, URL: http://aws.amazon.com/dynamodb/ (Última visita: Diciembre 2015).

[8] MongoDB Inc, URL: https://www.mongodb.com/ (Última visita: Diciembre 2015).

[9] Apache Cassandra, URL: http://cassandra.apache.org/ (Última visita: Diciembre 2015).

[10] James C. Corbett, Jeffrey Dean, Michael Epstein, Andrew Fikes, y otros, «Spanner: Google's Globally-Distributed Database», Tenth Symposium on Operating System Design and Implementation, Hollywood, CA, USA, 2012.

[11] Ghemawat, Jeffrey Dean y Sanjay, «MapReduce: Simplified Data Processing on Large Clusters», Sixth Symposium on Operating System Design and Implementation, San Francisco, CA, USA, 2004.

[12] Apache Hadoop MapReduce, URL: https://hadoop.apache.org/docs/r1.2.1/mapred tutorial.html (Última visita: Diciembre 2015).

[13] Apache Pig, URL: https://pig.apache.org/ (Última visita: Diciembre 2015).

[14] Apache Spark, URL: http://spark.apache.org/ (Última visita: Diciembre 2015).

[15] Apache Storm, URL: http://storm.apache.org/ (Última visita: Diciembre 2015).

[16] Tyler Akidau, Alex Balikov, Kaya Bekiroglu y otros, «MillWheel: Fault-Tolerant Stream Processing at Internet Scale», 39th International Conference on Very Large Data Bases, Trento, Italia, 2013.

[17] Grzegorz Malewicz, Matthew H. Austern, Aart J. C. Bik y otros, «Pregel: A System for Large-Scale Graph Processing», International Conference on Management of Data, New York, NY, USA, 2010.

[18] Apache Giraph, URL: http://giraph.apache.org/ (Última visita: Diciembre 2015).

[19] Apache Hive, URL: https://hive.apache.org/ (Última visita: Diciembre 2015).

[20] Cloudera Impala., URL: http://impala.io/ (Última visita: Diciembre 2015).

[21] Dataversity, URL: http://www.dataversity.net/not-your-type-big-data-matchmaker-on-five-data-types-you-need-to-explore-today (Última visita: Diciembre 2015).

[22] Apache Hadoop, URL: https://hadoop.apache.org/docs/r1.2.1/hdfs_design.html (Última visita: Diciembre 2015).

[23] White, Tom, « Hadoop: The Definitive Guide», O'Reilly, 4th Edition, 2015, Capítulo 1.3: The Hadoop Distributed Filesystem.

[24] White, Tom, « Hadoop: The Definitive Guide», O'Reilly, 4th Edition, 2015, Capítulo 1.2: MapReduce.

[25] Cloudera, URL: http://www.cloudera.com/documentation (Última visita: Diciembre 2015).