Big Data Platform deployment in a HPC cluster

Pol Ferrer López

Resumen– El Big Data es un término que esta tomando cada vez más importancia en nuestro sector, ya que conforme pasan los años todas las empresas trabajan con cantidades más grandes, complejas e importantes de datos. Por lo tanto, es importante utilizar software y plataformas que nos ayuden a gestionar, distribuir y analizar estos datos. En este informe se mostrará como ha sido trabajar con una famosa plataforma Big Data y con sus distintos componentes para llevar a cabo una ETL, encargada de procesar datos que más adelante se utilizarán en un modelo predictivo. Tratando los siguientes apartados: Primero la metodología que se ha utilizado, comentando brevemente la plataforma que se utilizará, para conseguir los objetivos propuestos en un inicio. Seguidamente la planificación que se ha llevado a cabo y los cambios en esta. Después aquellas tareas que se han llegado a completar, añadiendo los avances y problemas que han habido durante la realización del proyecto. Y finalmente resultados del trabajo realizado y unas conclusiones.

Palabras clave– Amazon Web Services, Apache Drill, Apache Kafka, Apache Spark, API, Big Data, Cloud, ETL, Hadoop, HDFS, HPC, On-premise, MapR, Microsoft Azure, Python, VPN, VPC.

Abstract– Big Data is a term that is taking on increasing importance in our sector, since as the years go by all companies work with larger, more complex and important amounts of data. It is therefore important to use software and platforms to help us manage, distribute and analyse this data. This report will show how it has been working with a famous Big Data platform and its different components to carry out an ETL, in charge of processing data that will later be used in a predictive model. Dealing with the following sections: First the methodology that has been used, commenting briefly on the platform to be used, to achieve the objectives proposed at the beginning. Then the planning that has been carried out and the changes in this one. Then those tasks that have been completed, adding the advances and problems that have occurred during the realization of the project. And finally the results of the work done and some conclusions.

Keywords– Amazon Web Services, Apache Drill, Apache Kafka, Apache Spark, API, Big Data, Cloud, ETL, Hadoop, HDFS, HPC, On-premise, MapR, Microsoft Azure, Python, VPN, VPC.

1 Introducción

Desde Noviembre de 2019 me encuentro en el departamento de Applied Intelligence de Accenture Barcelona donde actualmente estoy trabajando en el proyecto de Vibration Analytics. Applied Intelligence es una combinación de IA con datos, analítica y automatización para transformar todas las funciones y procesos de cualquier negocio. Cada vez está tomando mayor importancia el cómo se tratan, distribuyen y protegen los datos internos y externos. En este punto es donde entra el término Big Data.

Para la gestión y procesamiento de grandes cantidades de datos el departamento ha utilizado plataformas como Palantir o distribuciones de HDFS (Hadoop) junto con motores como Impala. Este realiza consultas SQL para el procesamiento masivo en paralelo de los datos almacenados en clusters de computadoras corriendo Apache Hadoop. [1] Pero actualmente es necesaria una nueva plataforma Big Data, que cumpla una serie de requisitos previos antes de ser desplegada, para garantizar unos mejores resultados.

• E-mail de contacto: pol.ferrerl@e-campus.uab.cat
• Mención realizada: Tecnologias de la Información
• Trabajo tutorizado por: Rafael Fernández (departament)
• Curso 2019/20

1.1. Motivación

Desde que hice la asignatura de Sistemas Distribuidos he sentido especial interés por el HPC y el Big Data. Aunque no había podido profundizar mucho en este tema, pude hacerlo gracias a mi incorporación en Accenture, dada la escasez de informáticos en mi departamento. Nada más empezar ya recibí formación en Big Data para poder aportar un punto de vista distinto al equipo (el de un ingeniero informático) y gracias la valoración positiva que dieron, decidieron que me encargaría de trasladar todos los datos del departamento a una nueva plataforma Big Data que cumpliera con unos requisitos mínimos a la vez que ofrecía mejores resultados que la tecnología utilizada por aquel entonces.

1.2. Estado del arte

En este apartado se mostrarán algunos proyectos a destacar relacionados con el estado actual del HPC, del Cloud Computing o del BigData:

- SAGE - El proyecto SAGE se encarga de construir una infraestructura centralizada de datos para el manejo extremo de estos en la era Exascale/Exabyte centrándose en una solución orientada al almacenamiento. SAGE da soporte a plataformas de almacenamiento de datos altamente distribuido, entre otras. [2]
- BigStorage - BigStorage cuenta con un gran número de proyectos, todos orientados al BigData. Concretamente hay uno de "Modelos predictivos mediante el uso de Big Data" realizado en la Universidad Politécnica de Madrid. Se centraron en dos puntos: Por la parte de gestión de los datos, buscaban controlar un gran volumen de pequeños objetos de datos de manera eficiente. Por la parte de la aplicación, querían desarrollar algoritmos de predicciones gracias al procesamiento y el análisis Big Data. [3]
- Compat - Proyecto orientado a HPMC (High Performance Multiscale Computing). Su principal objetivo es el desarrollo de algoritmos capaces de correr aplicaciones multiescala con un gran número de requisitos de datos mientras mantienen la escalabilidad, la robustez y la eficiencia energética. [4]

1.3. Objetivos

Dentro del proyecto de Vibration Analytics se pretende desarrollar una solución. Esta ha sido ideada para que recopile los datos de cientos de sensores instalados en una gran cantidad de máquinas de rotación (cojinetes) y así poder hacer un análisis/diagnóstico en función de la información que van emitiendo los sensores. Monitorizando a tiempo real el estado de las máquinas. Si se detecta que alguna de estas máquinas entra en un "modo fallo" se lanza una alerta. Los objetivos de este proyecto serán los siguientes:

- Selección de una plataforma Big Data según una serie de criterios.
- Formación en las tecnologías que se utilizarán.
- Despliegue de la plataforma on-premise (utilizando máquinas virtuales) estableciendo una estructura cliente-servidor y en un futuro realizar la migración a cloud.
- Construcción de un ecosistema de servicios que trabajen conjuntamente con esta.
- Envío de los datos de cliente en streaming e importación de estos en la plataforma.
- Diseño de una ETL (Extract, Transform and Load) para procesar estos datos.
- Realización de dashboards de monitorización del rendimiento de la plataforma.
- Realización de documentación técnica y de usuario para que otros miembros del equipo puedan trabajar con la plataforma.

La orientación del proyecto es dentro de un contexto de negocio. Tenemos que pensar que no tiene que ver con Kaggle, la cual es una empresa que se encarga de organizar concursos para analizar grandes cantidades de datos y así conseguir realizar predicciones. [5]

2 METODOLOGÍA

La realización de este proyecto se ha llevado a cabo con la ayuda de mi tutor de la empresa y además, el apoyo de un compañero que también posee conocimientos en informática y más concretamente en HPC.

Al ser un proyecto que no se encuentra sujeto a ningún cliente, en principio no se desarrolla siguiendo ningún tipo de metodología, que suele estar acordada con este a la hora de establecer un contrato. Por lo tanto, de cara a la empresa tan solo será necesaria una reunión semanal (la mayoría de las veces se realizará mediante videoconferencia, donde yo expondré una presentación mostrando estado actual, problemas y temas pendientes) entre las personas mencionadas para comentar los avances y próximos aspectos a trabajar. Aun así, de cara a la realización de este trabajo, y para conseguir mis objetivos con una mayor organización, he utilizado la metodología Scrum. En Scrum se realizan entregas parciales y regulares del producto final. Este me ha ayudado para obtener resultados rápidamente en un entorno donde los requisitos han variado constantemente durante el desarrollo del proyecto. [6]

Los primeros sprints tenían una duración inicial de una semana dado que se centraron en la formación y la parte más teórica del proyecto, por lo tanto se iban cumpliendo objetivos con mayor facilidad y al cerrar cada sprint habíamos realizado avances. A partir del quinto sprint decidimos duplicar su duración dado que desde el momento que comenzamos con el despliegue de la plataforma y el desarrollo de código se avanzó más lentamente. Además al terminar con la reunión semanal (Sprint meeting) guardabamos el feedback recibido para más adelante poder realizar una documentación completa.

Aunque en un principio se tenía pensado utilizar Git para control de versiones del proyecto, el desarrollo de código no es el apartado de mayor importancia y por lo tanto se almacenará una copia de seguridad tanto de la documentación como del código a una carpeta de OneDrive dedicada al proyecto cuyo acceso solo está permitido a miembros del equipo.

2.1. Plataforma

La plataforma Big Data que se utilizará durante la realización de este trabajo es MapR Data Platform. Con esta podremos almacenar, administrar, procesar y analizar todos los datos, incluidos archivos, tablas y streams de distintas fuentes de datos (ya sean operacionales, históricos o en tiempo real) de manera eficiente. MapR ofrece un conjunto de servicios de datos para garantizar una escala exabyte y un alto rendimiento. También brinda protección de datos, recuperación ante desastres, seguridad y servicios de gestión de la plataforma.

Cuenta con un gran número de open APIs, además de soporte para containers. Por lo tanto, asegura un amplio acceso distribuido a la aplicación y una portabilidad perfecta de la aplicación. MapR Data Platform se ejecuta en hardware básico en implementaciones locales, en la nube y perimetrales. [7]

TABLA 1: REQUISITOS DE INSTALACIÓN DE MAPR DATA PLATFORM

Element	Requirements
CPU	64-bit
OS	RedHat, Oracle Linux, CentOS, SUSE or Ubuntu.
Memory	16 GB minimum, 32 GB recommended.
Disk	Raw, unformated drived and no partitions.
DNS	Hostname, reaches all other nodes.
Users	Common users across all nodes; passwordless ssh (optional)
Java	Must run Java 1.8
Other	NTP, Syslog, PAM

3 PLANIFICACIÓN

La planificación inicial estaba organizada de tal manera que el primer y segundo sprint sirvieran para la selección de la plataforma a utilizar. Durante el tercer y cuarto sprint nos encargaríamos del despliegue y la configuración de esta on-premise para dedicar los próximos al diseño la ETL. Los últimos sprints en un principio estaban pensados para el despliegue de la plataforma en un conjunto de nodos en el cloud para comparar su rendimiento respecto al single-node cluster on-premise con el que se está trabajando actualmente, pero han habido una serie de problemas. MapR es una plataforma orientada a empresas, esto hace que los recursos necesarios en cada nodo para su instalación sean realmente altos (como podemos observar en la Tabla 1).

Esto hace que en condiciones normales no se pueda desplegar esta en un cluster on-premise de más de un nodo (de ahí la decisión de trabajar con un solo nodo de primeras), ya que la mayoría de personas no disponen de ordenadores de más de 32GB de memoria. Aspectos como replicación, distribución del cómputo o tolerancia de fallos podrían ser analizados en el cloud, pero Accenture actualmente no puede hacerse cargo de los gastos del despliegue en cloud dadas las circunstancias. Mediante esta plataforma se buscaba suplir las capacidades de Microsoft Azure o Amazon Web Services, y dada mi dotación presupuestaria no puedo encargarme de pagar por mi cuenta la configuración prevista:

- 5 nodos m4.2xlarge (8 CPUs virtuales, 32GB de RAM y alto rendimiento de red) de Amazon Web Services. 1 como master, los demás como workers. [8]
- MapR Enterprise Edition desplegado en los cinco nodos mediante una VPC (Virtual Private Cloud)
- VPN (Virtual Private Network) para el acceso al clúster.

Por lo tanto, los últimos sprints se han dedicado a mejoras de código y análisis de rendimiento de la plataforma mediante el conocido software Grafana, que nos permitirá elaborar dashboards con las métricas que nos interesen.

4 DESARROLLO

En esta sección dividiremos el desarrollo del proyecto por sprints:

4.1. Primer sprint

Tras la primera reunión con mi tutor de la empresa (y jefe de proyecto), decidimos llevar a cabo una pequeña formación en Big Data utilizando material propio del portal de la empresa. Debido a que el principal problema era la falta de conocimientos en este tema. Para suplir esta carencia, el portal nos brindó una gran variedad de documentación, generada durante el desarrollo de otros proyectos de Accenture.

Una vez finalizada la formación, me presentaron las 53 plataformas analíticas de Big Data más importantes en la actualidad. Antes de poder escoger una de ellas, para ser desplegada, era necesario analizar cada una siguiendo una serie de pasos:

- Comentario breve de cada una de las plataformas.
- Primer filtrado eliminando aquellas que no ayuden a cumplir los objetivos de este proyecto, es decir aquellas que no se centren en procesos analíticos.
- Estudio en profundidad de las plataformas en formato presentación para los miembros de mi equipo. Se mostraba información respecto: La arquitectura software, funcionalidades clave, lenguajes, tamaño de los grupos de usuarios a los que iba dirigida, flujo de datos y distinción de sus diferentes componentes. Tras haber preparado esta documentación, realicé una primera presentación para recibir feedback por parte del equipo y ver qué plataformas podrían encajar mejor en el proyecto.

4.2. Segundo sprint

Tras realizar el análisis principal de todas las plataformas, mi tutor me comentó una serie de criterios a considerar a la

hora de seleccionar la plataforma a desplegar, necesarios para que realmente tuviera sentido el hecho de buscar una nueva tecnología. Se realizó un filtrado de todas las plataformas analizadas para seleccionar un pequeño grupo que cumplieran estos aspectos:

- Compatibilidad con Microsoft Azure y Amazon Web Services. Estas dos tecnologías son las que utilizan la mayoría de clientes de mi departamento. Es de vital importancia que en un futuro se puedan trasladar los datos, con suma facilidad, de Azure o AWS a la plataforma escogida. E incluso tomar éstas como fuentes de datos para realizar la ETL directamente.
- Orientada a procesos analíticos. El código a desarrollar busca analizar grandes cantidades de datos, para su posterior interpretación, por lo tanto interesa que la plataforma esté preparada para ello.
- Grupos de usuarios pequeños. Los equipos asignados a cada proyecto no suelen superar los 10 miembros, además de que la mitad de ellos no trabajaran con la plataforma de manera directa. Por lo tanto esta debe ser capaz de gestionarse correctamente con un grupo reducido de usuarios.
- Nivel de conocimientos bajo. La formación previa debe ser mínima para los usuarios. Queremos que el hecho de migrar nuestros datos no suponga una pérdida de tiempo para los miembros del equipo al tener que realizar formaciones en distintas tecnologías, la mayoría desconocidas para ellos.
- Lenguaje Python o R. Todo el código desarrollado en el departamento se encuentra escrito en estos dos lenguajes. El código a desarrollar, una vez desplegada la plataforma, tendrá que ser escrito en Python para aprovechar funciones hechas por otros miembros del equipo para trabajar con Azure o AWS.
- Procesamiento de datos de manera distribuida. Debemos poder desplegar esta plataforma en la nube y/o on-premise en un grupo numeroso de nodos y que estos se coordinen para que se cumpla la tolerancia de fallos, replicación, distribución de cómputo,...

Seguidamente se realizó documentación en forma de presentación de las plataformas filtradas para reunir información de estas. Esta presentación servirá más adelante para decidir conjuntamente con el equipo que tecnología se utilizará de manera definitiva. Las plataformas seleccionadas fueron las siguientes:

- SPLUNK
- Google BigQuery
- CSC Big Data Platform
- MapR
- Mu Sigma Big Data
- Amdocs Insight
- HPCC Systems Big Data
- BigObject
- SAP HANA

Por los motivos que se expondrán a continuación, se han considerado estas tres plataformas como las idóneas para nuestro proyecto:

MapR: Permite el acceso seguro a todos los datos importados desde plataformas como Google Cloud Platform, Amazon Web Services o Microsoft Azure (esta última es de suma importancia dado que la mayoría de nuestros clientes optan por ella a la hora de trabajar). Mediante MapR XD podemos almacenar datos distribuidos globalmente mediante un sistema altamente escalable y con tolerancia a fallos. Su base de datos NoSQL da soporte a distintos tipos de modelos de datos como wide-column, key-value* o time-series [10]. MapR Data Science Refinery permite el computo distribuido. En el aspecto del desarrollo de código se trabaja con Python, R, Node.js, Java y Scala. Además de trabajar con software como Apache Hadoop, Apache Spark o Apache Drill. Además, cuenta con una gran facilidad de despliegue y de gestión gracias a herramientas propias de la plataforma.

HPCC: El hecho de juntar HPC (High Performance Computing) con Data Science permite a las organizaciones utilizar analíticas predictivas para realizar mejores decisiones y generar estrategias innovadoras. Permite diferentes tipos de modelos de datos, importados desde otras plataformas como Azure (La de mayor importancia para nosotros), provenientes en streaming, en tiempo real o en batch. Destaca respecto a otras plataformas al contar con dos clusters con funciones específicas: Thor y Roxie. HPCC trabaja con una gran variedad de bases de datos dotando al usuario de flexibilidad a la hora de trabajar. Utiliza APIs de Machine Learning y un lenguaje específico de la plataforma llamado ECL, que mediante la inserción de pocas líneas de código puede permitir el trabajo con Python.

SAP HANA: Su infraestructura permite a los analistas trabajar con un gran número de algoritmos que pueden ser lanzados en la propia HANA o en otras plataformas como Azure. Cuenta con la funcionalidad de Analytical Intelligence para procesos analíticos.

Tras comentar con todo el equipo mis conclusiones respecto a las plataformas, decidimos utilizar MapR ya que dispone de una Community Edition totalmente gratuita con los servicios necesarios para llevar a cabo la ETL y por lo tanto era más fácil trabajar con ella en un principio.

4.3. Tercer sprint

Una vez seleccionada la plataforma a desplegar era necesario conocer en profundidad su funcionamiento. Gracias a la cantidad de información disponible online de MapR pudimos aprender cómo funcionaban todos los componentes de esta y como se relacionaban entre ellos. Esta plataforma cuenta con tres grandes elementos:

- MapR Filesystem: MapR utiliza de primeras un disco sin montar para construir su sistema de ficheros. Este puede exponer el almacenamiento a través de varias API y tienen la capacidad de asegurar el sistema de archivos mejor que los derivados de Apache, que almacenan los bloques de archivos en el sistema de archivos de Linux.

- MapR Data: MapR Datastorage es un sistema de gestión de bases de datos NoSQL integrado en MapR Data Plaftorm. Puede reunir operaciones y análisis, así como cargas de trabajo de streaming de datos y bases de datos en tiempo real para permitir un gran conjunto de aplicaciones que utilicen datos en las empresas. Tienen dos tipos de tablas: Binarias y JSON. Nosotros trabajaremos con las segundas.

- MapR Streaming: Útil para enviar datos de cliente a servidor y cuenta con dos tipos de aplicación. El Producer permite publicar mensajes en canales llamados topics, colecciones lógicas de mensajes, que son administrados por MapR Streams. Y los Consumers pueden leer esos mensajes a su propio ritmo. Todos los mensajes publicados en MapR Streams son persistentes, lo que permite a los futuros consumidores ”ponerse al díaçon el procesamiento y las aplicaciones de análisis para procesar datos históricos.

Tras esta pequeña formación, comenzaron a haber problemas. Según la planificación, la plataforma sería desplegada primero on-premise en un solo nodo para hacer todas las pruebas necesarias antes de su despliegue en cloud. MapR Data Platform no está preparada para ser desplegada en un solo nodo con pocos recursos, como era el ordenador del que disponíamos al principio de todo. Para ello existe una versión de prueba que ellos mismos ofrecen con un número de servicios y tecnologías bastante limitados, MapR Sandbox. Se utilizó de primeras para hacer pequeñas pruebas con Apache Spark y Apache Kafka. Este último se utilizará para el streaming de los datos.

4.4. Cuarto sprint

Selección de las características de la VM II: Gracias a mi tutor de la empresa, conseguimos un nuevo ordenador con más recursos que el anterior. Por lo tanto ya podíamos desplegar MapR Data Platform con todos los servicios que nos interesaban. Escogimos el mismo SO que se utiliza en MapR Sandbox porque nos dió buenos resultados, CentOS 7 y utilizamos un procesador de cuatro núcleos. Además a la VM le dimos 32 GB de memoria (la plataforma necesitaba mínimo 16) y 130 GB de disco duro. Con estas características, estaba casi preparada para la instalación de la plataforma ya que antes era necesaria una configuración previa para que no fallara el despliegue. Fueron necesarios un gran número de ajustes de la VM para poder instalar la plataforma. Empezando por algo tan sencillo como que el locale tenía que ser “en-US”. Deberíamos darle un hostname apropiado para que otros ordenadores “clientes” puedan conectarse a este y enviarle trabajos (jobs), para ello era necesario poner la VM en modo Bridge comportándose como una máquina física. Seguidamente se trataron temas de rendimiento, seguridad y software necesario, como Java. Para terminar, era necesaria la partición del disco de cada nodo en dos: La primera parte dedicada a MapR Core (donde se encontraría el software instalado al desplegar la plataforma) y la segunda para MapR Filesystem. Ambas formateadas con ext4.

4.5. Quinto sprint

Antes de empezar con la instalación de la plataforma necesitábamos tener un listado con los hostname de cada nodo (en este caso como solo trabajamos con uno, le dimos de nombre maprnode. Además era necesario un listado con los discos donde se iban a instalar los componentes, utilizamos las particiones que se generaron al finalizar el sprint anterior.

Existe también un script instalador que automatiza el proceso y te redirige a una URL donde configurar incluso los servicios extra. Pero estaba orientado a otro tipo de usuarios y no nos daba un número muy grande de opciones de configuración. En un principio, tuvimos una serie de problemas dado que desconociamos el formato correcto de las particiones y por lo tanto no se ejecutaba bién este instalador. Por lo tanto decidimos hacer la instalación desde cero. Se llevaron a cabo los siguientes pasos:

- Instalar MapR Package Key. Necesario ya que los paquetes de MapR están firmados criptograficamente.

- Instalar MapR core packages. Aquellos que dan las funcionalidades básicas al cluster, como el propio sistema de ficheros. Para ello utilizamos un repositorio que se encontraba ya en Internet. En el host solo será necesario instalar el paquete de cliente para poder acceder a la plataforma sin formar parte del cluster.

- Verificación de la instalación. Comprobar, en el directorio donde se haya instalado el núcleo de la plataforma, si se encuentran disponibles los servicios base. También se definen variables de entorno, como la de ”JAVA HOME”para que los servicios que instalemos en un futuro puedan utilizar Java.

Se trabajaría de la siguiente manera:

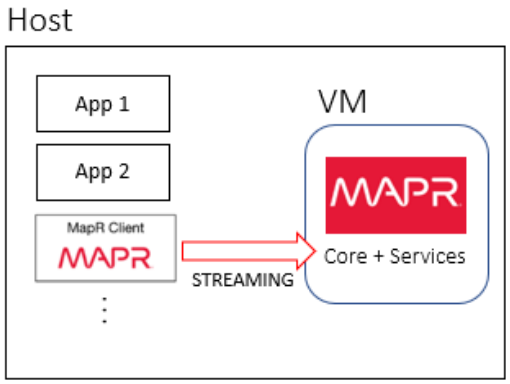

Fig. 1: Estructura de trabajo cliente + single-node cluster.

Una vez desplegada la plataforma, era necesario configurarla para poder empezar construir el “ecosistema” de componentes. Para ello hemos de conocer los elementos que conforman MapR Core y se encargan de la gestión y el monitoreo de este:

- Zookeeper: Servicio de coordinación para aplicaciones distribuidas.

- Container Location Database (CLDB): Rastrea toda la información relacionada con cada container que se en-

cuentra en el filesystem. También rastrea la actividad de los nodos.

- MapR Control System (MCS): Encargado de la gestión de la propia plataforma, una vez instalada, disponible a través de una URL.
- MapR Warden: Es una aplicación Java ligera que se ejecuta en todos los nodos de un clúster y coordina los servicios del clúster.

Como trabajaremos con un solo nodo, tanto Zookeeper como CDLB se ejecutarán en este para que cuente con todas las funciones disponibles. Antes de llevar a cabo la configuración, la propia empresa te brinda el script configure.sh y de esta manera se hace mucho más sencilla llevar a cabo esta tarea. Este nos permite realizar tareas como dar un nombre al cluster, comprobar espacio o establecer un servicio de logging de manera automática. [9]

Tras la configuración, antes de poder elegir que servicios nos gustaría integrar en esta y así construir nuestro "ecosistema". Era necesario iniciar la plataforma, corriendo los servicios de Zookeeper y Warden y aceptar la licencia.

A partir de este momeno ya podemos instalar los servicios de MapR Ecosystem Pack (MEP). Escogimos los siguientes:

- Apache Drill.
- HttpFS.
- MapR Event Store.
- Apache Kafka.
- Apache Spark.
- MapR Data Access Gateway.

Para empezar a trabajar con los servicios que hemos instalado probamos con datos que no fueran de la empresa, utilizando la Yelp Open Dataset. Esta es un conjunto de datos de negocio, de reviews y de usuarios para uso personal o académico que se encuentra disponible en formato JSON. [10] Realizamos streaming del contenido de este fichero JSON hasta la base de datos de MapR, utilizando MapR Event Store, el cual se integra con Spark Streaming (gracias a la API de Kafka).

Para ello generamos dos MapR Event Store topics (hemos hablado de ellos anteriormente). En este caso tendremos un Producer, que será el propio host actuando como cliente, y un Consumer, el nodo del cluster (VM). Generamos dos topics, uno para datos de prueba y otro para datos de cliente (los que utilizaremos más adelante).

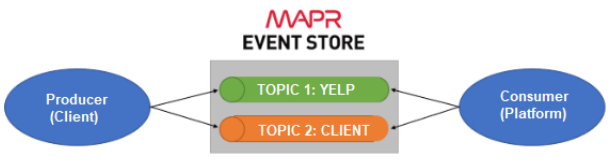

Fig. 2: Organización de los topics.

Dentro de los diferentes JSON que ofrece YELP, utilizamos el de user debido a que para poder almacenar este tipo de ficheros en la base de datos necesitamos definir un idField (un campo identificador de cada registro del JSON), el cual será la clave de cada fila y el índice primario para la base de datos de MapR (Mapr DB). Por lo tanto, el campo id nos ahorraría trabajo.

El código desarrollado para el producer se encuentra escrito en Java y el del Consumer en Scala, dado que trabajaremos con Spark. Crearemos el input stream para un topic en concreto mediante un DirectStream (de kafkaUtils), donde cada mensaje contiene un key-value. El fichero se enviará en un conjunto de RDDs (principal abstracción de datos que tiene Apache Spark) [11]

El fichero JSON original ha sido particionado, por lo tanto para almacenar el contenido de estos RDDs, que van llegando al cluster, en la base de datos utilizaremos una función propia de MapR DB Spark Connector. Una vez se encuentre el contenido en la DB lo trasladaremos a un dataset de Spark para empezar a utilizarlo.

Dado que ya se ha realizado el streaming de datos de prueba correctamente, pudimos empezar a trabajar con los ficheros de cliente que se encontraban en el OneDrive de la empresa. Estos podían encontrarse en tres tipos distintos de formato: CSV, JSON y Parquet. Además, era de vital importancia poder realizar dentro de la plataforma la conversión de JSON a Parquet y viceversa con suma facilidad.

Se descargaban en local en el host y para poder llevar a cabo el streaming de estos, antes era necesario volver a definir un idField. Por suerte como cada registro (fila) era un mensaje que llegaba de los distintos sensores de la maquinaria, se utilizó el propio campo MessageId.

4.6. Sexto sprint

Diseño de la ETL: Una vez habíamos encontrado la manera de enviar nuestros datos al cluster empezamos a diseñar la ETL; Extraer (extract), transformar (transform) y cargar (load). En la fase de extracción se obtienen los datos de sistemas de origen, se analizan, se interpretan para ver si cumplen la pauta o estructura esperada y si es necesario se convierten estos en un formato preparado para iniciar el proceso de transformación. En la fase de transformación se aplican una serie funciones sobre los datos para convertirlos en datos listos para ser cargados. Y finalmente en el proceso de carga los datos transformados son almacenados en el sistema destino (que en nuestro caso será el mismo nodo). [12] Esta seria la estructura de nuestra ETL:

1. Se obtienen los datos, ya sea en formato JSON, Parquet o CSV (aunque por ahora solo hemos trabajado con los dos primeros).
2. Se envían los datos de cliente al single-node cluster mediante Spark Streaming (como se ha explicado anteriormente).
3. Una vez enviados los datos se utiliza la tabla que se encuentra en MapR DB para generar una Spark Dataset. Este servirá para trabajar con los datos: Primero se filtraran para solo utilizar aquellos con los valores que nos interesan y seguidamente se procesaran.
4. Una vez llevado a cabo el procesamiento de estos, se volverán a almacenar en la base de datos para su posterior análisis con PySpark (para generación de gráficos) o Apache Drill para realizar queries con estos.

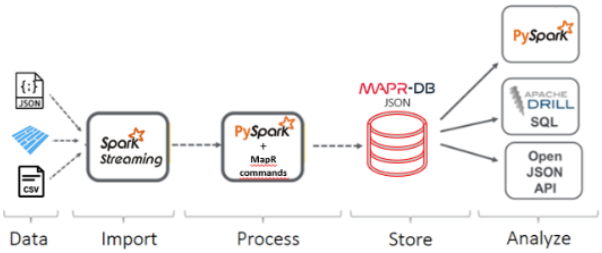

Fig. 3: Flujo de datos de la ETL.

Desarrollo de codigo de procesado: Este ha sido desarrollado en Python 3.7, debido a que se han reutilizado algunas funciones ya implementadas por miembros del equipo. Este código era el encargado de:

1. Crear un Spark Dataset de los datos almacenados en la DB.

2. Filtrar el contenido de este para solo obtener unos datos en concreto. El dataset cuenta con distintos tipos de registro en cuanto al sensor que se está utilizando en la maquinaria: Temperatura, velocidad, amplitud,... Nosotros por ahora solo utilizaremos los de amplitud.

3. Mediante estos datos es posible generar una waveform de las vibraciones de la maquinaria en base al tiempo. Nos interesa aplicar una transformación a esta onda para generar un espectro en base a las frecuencia. Para ello se utilizará la FFT o Transformada Rápida de Fourier: La Transformada Rápida de Fourier es un algoritmo que permite calcular eficientemente la Transformada de Fourier Discreta y su inversa. La Transformada Rápida de Fourier es de suma importancia en el análisis, diseño y realización de algoritmos y sistemas de procesamiento de señales dado que brinda mayor eficiencia tanto en tiempo como en recursos. [13]

4.7. Séptimo sprint

Una vez hemos obtenido los datos necesarios para generar el espectro, crearemos un JSON que será enviado al modelo predictivo de fallos que están desarrollando mis compañeros. Este JSON no contará con los datos de un sólo mensaje, sino con dos que se hayan enviado en un periodo de tiempo no mayor de una hora desde distintos sensores de la misma pieza de la maquinaria. Además contaremos con datos adicionales (umbrales de velocidades de la maquinaria) que ya se encontraran cargados en la plataforma dado su pequeño tamaño en comparación a los datasets con los que trabajamos.

Al haber finalizado correctamente la ETL, creímos que era importante generar documentación para futuros usuarios, dado que esta plataforma se utilizará en un futuro en la empresa. Se ha preparado en formato presentación para que sea lo más visual posible. Se ha dividido en dos tipos:

- Documentación de usuario: Se encarga de explicar paso a paso el desarrollo de una ETL y las diferentes librerías que implica utilizando de ejemplo nuestro caso, Vibration Analytics. Además da información sobre MapR y sus tecnologías implicadas en este proyecto.
- Documentación técnica: Muestra cómo desplegar la plataforma. Empieza con el inicio de la VM y la configuración previa para poder instalar la plataforma. En este caso hemos querido explicar el despliegue de MapR Data Platform utilizando el instalador propio de la empresa para facilitar esta tarea a posibles empleados no tan experimentados con sistemas Linux.

4.8. Último sprint

Para finalizar nuestro proyecto nos centramos en dos tareas. Primero de todo mejoramos el código eliminando partes innecesarias o simplificándolas y utilizamos distintas librerías para generar las ondas de vibración y los espectros, permitiéndonos interactuar con estas. Por ejemplo, para localizar a que frecuencias pertenecían los picos dentro del espectro. Después, queríamos controlar el tiempo para llevar a cabo el procesamiento de los datos. Por lo tanto, generamos varias métricas: Tiempo promedio de procesado de un mensaje, tiempo total de procesado de un dataset completo y tiempo completo de la ETL (desde que se envian los datos de cliente hasta que se han almacenado los JSON finales)

Finalmente, para monitorización de la plataforma generamos dos tipos de dashboards mediante Grafana (uno de los servicios que ya incluía la propia plataforma): CLDB dashboard (control de los nodos) y Node dashboard (uso de CPU, memoria y espacio utilizados,...).

5 RESULTADOS

En primer lugar se mostrarán diversos aspectos de la plataforma para comprobar su correcto despliegue. Seguidamente, se añadirán las gráficas de las ondas generadas a partir de los datos recopilados, mediante los sensores de vibración anclados a la maquinaria de rotación, y la transformación de estos, mediante la FFT, para lograr obtener las frecuencias predominantes. Estas serán las que nos permitirán predecir que un aspecto en concreto de la máquina está cerca de tener un fallo. Finalmente, se mostrarán los dashboards generados con Grafana para la monitorización de la plataforma.

Al trabajar con un solo nodo, todos los servicios que utiliza MapR, y que suelen repartirse por el cluster, se encontraran en el mismo nodo.

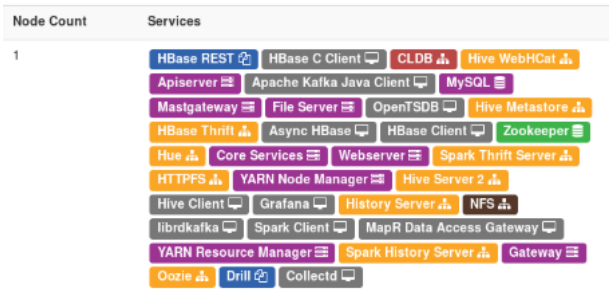

Fig. 4: Servicios desplegados en el single-node cluster.

Una vez desplegada la plataforma en el nodo podremos gestionarla o acceder a sus diferentes servicios mediante el servicio MapR Installer. Este nos permitirá extender, gestionar, apagar o desinstalar el cluster. Además

muestra un listado de los servicios instalados y la URL para acceder a su UI. Si Warden y ZooKeeper no están ejecutándose, no se puede trabajar con la plataforma dado que no podremos acceder a los servicios que queramos utilizar. Por lo tanto, realizamos la comprobación del correcto funcionamiento de estos, accedemos al Installer y probamos con Apache Drill.

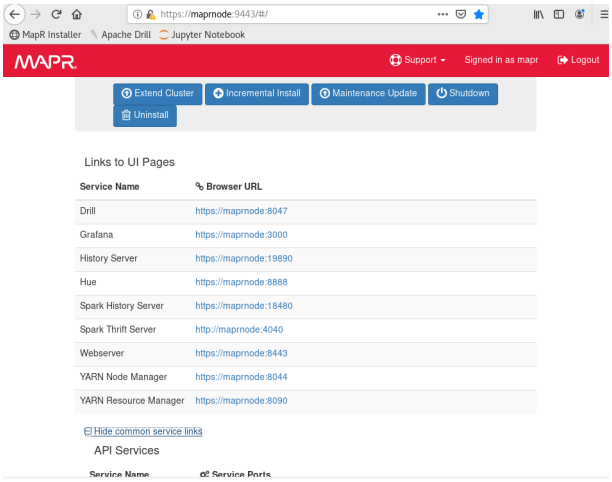

Fig. 5: MapR Installer y los servicios accesibles.

Empezamos a trabajar con los datos que llegaban al nodo. Utilizando estos pudimos generar una onda de 10 segundos de duración respecto a las vibraciones que recibe un sensor en una posición concreta de la maquinaria de rotación.

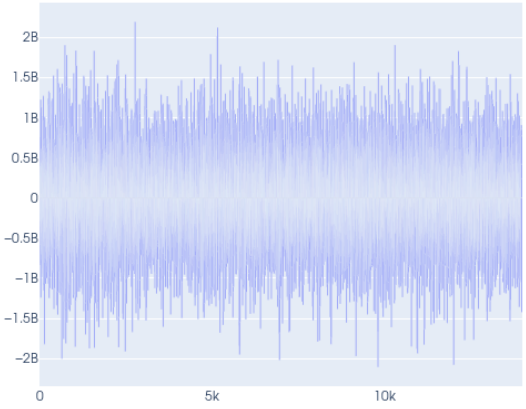

Fig. 6: Onda generada a partir de los datos sin procesar.

Tras realizar el procesado de los datos, aplicando la FFT, generamos unos outputs para poder visualizar un espectro. Este nos permitirá ver que frecuencias predominan. Los picos del espectro están asociados a diferentes fallos de la maquinaria.

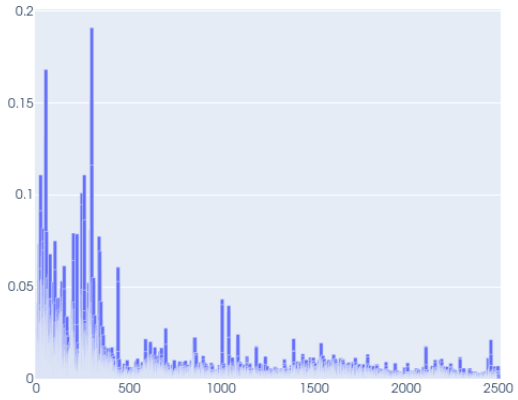

Fig. 7: Espectro generado tras procesar los datos.

Finalmente, para observar como se comportaba la plataforma durante la carga y el procesado de los datos, generamos los siguientes dashboards mediante Grafana.

Fig. 8: Dashboard de maprnode con algunos aspectos de la plataforma.

6 CONCLUSIONES

El proyecto comenzó algo más tarde de lo esperado ya que no sabía muy bien cómo enfocarlo y de primeras no iba a realizarlo con la empresa. Pero quise aprovechar los conocimientos de expertos en la materia de mi departamento y toda la información disponible en el portal de la empresa para sacar adelante un proyecto completo tanto a nivel teórico como práctico. Se ha desplegado una plataforma analítica Big Data en un cluster de un solo nodo on-premise para poder dar soporte al proyecto de Vibration Analytics de Accenture, que se encarga de predicción de fallos en maquinaria de rotación, y acabar sustituyendo a la tecnología que se utiliza actualmente, Microsoft Azure.

La primera parte fue la más teórica, debido a que era necesaria una formación previa en Big Data. Seguidamente se llevó a cabo un análisis de las plataformas y un filtrado según unos criterios para poder escoger qué plataforma se iba a utilizar. La escogida fue MapR Data Platform. Después nos centramos en preparar y configurar una máquina

virtual donde más adelante se instalaría MapR para que esta actuara como un single-node cluster y se accediera desde el propio host (como si fuera una estructura cliente-servidor). Una vez corriendo todos los servicios y componentes necesarios se llevó a cabo la ETL. Tras procesar los datos mediante esta ETL, se llevaría a cabo el análisis de rendimiento y se generarían dashboards. Para finalizar, se redacta documentación para futuros usuarios.

El proyecto ha finalizado con muy buenos resultados pese a haber tareas que no se pudieron llevar a cabo. Como por ejemplo, el despliegue de la plataforma contratando servicios Cloud dada la dotación presupuestaria. Pero todas las demás tareas se han completado en un tiempo menor del esperado y se ha recibido un feedback positivo por parte de los miembros de la empresa. A finales de Mayo fue presentado el trabajo realizado durante estos meses delante de todo el departamento de Analytics para dar a conocer los resultados, y tuvo una muy buen recepción por parte de demás analistas y becarios.

AGRADECIMIENTOS

Primero de todo darle las gracias a mi tutor de la empresa, Jesús Gabaldón, por su gran apoyo desde mi incorporación en Accenture. Me ha hecho mejorar como profesional al darme una gran implicación en el desarrollo de este proyecto. Además de Alejandro Abol, a quién considero un experto dentro de este sector y que estoy deseando trabajar con el en futuros proyectos. Finalmente agradecerle a Rafael Fernández, mi tutor de la universidad, por el esfuerzo constante que ha tenido para mejorar mi trabajo desde un punto de vista académico.

REFERENCIAS

[1] “Cloudera Impala” Wikipedia. La enciclopedia libre. [Online]. Disponible: https://es.wikipedia.org/wiki/Cloudera_Impala [Acceso: 03-Marzo-2020].

[2] “Sage Flyer” Sage2. [Online]. Disponible: https://sagestorage.eu/sites/default/files/sc15_flyer.pdf [Acceso: 07-Marzo-2020]

[3] “BigStorage: Job Applications” BigStorage. [Online]. Disponible: http://bigstorage-project.eu/index.php/job-applications [Acceso: 07-Marzo-2020]

[4] “The COMPAT Project” CompatProject. [Online]. Disponible: https://www.compatproject.eu [Acceso: 07-Marzo-2020]

[5] “Kaggle” Wikipedia. La enciclopedia libre. [Online]. Disponible: https://ca.wikipedia.org/wiki/Kaggle [Acceso: 05-Marzo-2020].

[6] “¿Qué es Scrum?” ProyectosAgiles.org. [Online]. Disponible: https://proyectosagiles.org/que-es-scrum/ [Acceso: 06-Marzo-2020]

[7] “THE MAPR DATA PLATFORM” Mapr.com. [Online]. Disponible: https://mapr.com/products/ [Acceso: 07-Abril-2020]

[8] “Tipos de instancias de Amazon Web Services”. AWS. [Disponible]; https://aws.amazon.com/es/ec2/instance-types/ [Acceso: 08-Abril-2020]

[9] “configure.sh” MaprDocs. [Online]. Disponible: https://mapr.com/docs/61/ReferenceGuide/configure.sh.html [Acceso: 17-Mayo-2020]

[10] “Yelp Open Dataset”. Yelp.com [Online]. Disponible: https://www.yelp.com/dataset [Acceso: 17-Mayo-2020]

[11] “Claves principales de los Resilient Distributed Datasets”(RDD) de Apache Spark”. El Blog de Mikel Niño [Online]. Disponible: http://www.mikelnino.com/2016/03/claves-principales-resilient-distributed-datasets-rdd-apache-spark.html [Acceso: 17-Mayo-2020]

[12] “El valor de la gestión de los datos. ¿Qué son los procesos ETL?” PowerData. [Online]. Disponible: https://blog.powerdata.es/el-valor-de-la-gestion-de-datos/qu-son-los-procesos-etl [Acceso:17-Mayo-2020]

[13] “FFT. Transformada Rápida de Fourier” Ana Lucía Schmidt. [Online]. Disponible: http://lcr.uns.edu.ar/fvc/NotasDeAplicacion/FVC-Schmidt %20Ana %20Luc %C3 %ADa.pdf [Acceso: 17-Mayo-2020]

APÉNDICE

En esta sección adicional aprovecharemos para mostrar contenido de especial interés. La mayoría de las imágenes son capturas de documentación propia del proyecto.

	Support for Microsoft Azure or AWS?	Analytical process-oriented?	Small user groups?	Low level of knowledge?	Developing with Python or R?	Distributed data processing?
SPLUNK	Yes.	Yes. Its strong point is real-time processing	Oriented for all types of groups..	Requires training.	Yes, with Python.	No information has been found regarding this. It also has a very high runtime.
BigQuery	Yes.	No, main function as datawarehouse.	Oriented for all types of groups.	Yes. Allows training through Google Trainings.	Python, R, Nodejs,...	Yes. MapReduce with Hadoop.
CSC	Only AWS.	Yes. Very scalable and versatile thanks to the use of many different DBs.	Oriented for all types of groups.	Yes. Using so many different DBs is very user accessible.	Yes, although it depends on the DB.	Yes, with Hadoop Processing.
MapR	Yes. And even also with Google Cloud Platform.	Yes. It works with a large number of APIs and allows you to work with structured or unstructured data.	Oriented for all types of groups.	Yes, working with so many different APIs makes it easy to use.	Con Python, R, Java, Scala, Node.js.	Yes. Data classification. Organization of these in different layers.
Mu Sigma	No.	Yes. muFlow, muStream and muHPC are some of the components in charge of this.	Oriented for all types of groups.	No. It is a complex platform given its large number of related components when it comes to working.	With Python, using ML libraries.	Yes. Thanks to the muHPC component.

Fig. 9: Comparativa final de plataformas I

	Support for Microsoft Azure or AWS?	Analytical process-oriented?	Small user groups?	Low level of knowledge?	Developing with Python or R?	Distributed data processing?
Amdocs	There is no information on this.	No. Communications and Reporting Oriented.	Oriented for all types of groups.	*Yes. Allows self-service ad-hoc reporting for commercial users.*	No.	Yes. It also allows this with great security thanks to the use of a single Gateway that serves as a filter.
HPCC	Yes.	Yes. Uses two different cluster types, Roxie (High-Performance Query Oriented) and Thor (Batch Data Oriented)	Oriented for all types of groups.	No. But it has a lot of information regarding its use and its infrastructure to know how it works.	Works with ECL. But it allows the use of Python by inserting few lines of ECL code.	Sí.
BigObject	There's no information on it.	Yes. Very efficient thanks to the use of In-Data Computing.	Oriented for all types of groups.	Yes. Simple query language and allows you to work with more than one DB model.	No.	It is not necessary thanks to the use of In-Data Computing that allows you to have an "infinite" space for your data.
SAP	Yes.	Yes. It consists of Analytical Intelligence functionality aimed at improving data analysis.	Oriented for all types of groups.	Yes.	Allows work with R.	Yes. Highly scalable processing engine that works with many sources.

Fig. 10: Comparativa final de plataformas II

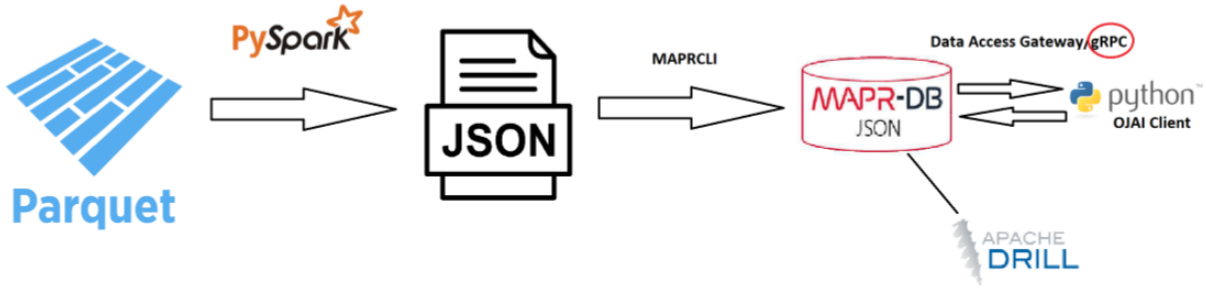

Fig. 11: Estructura inicial del flujo de datos de la ETL

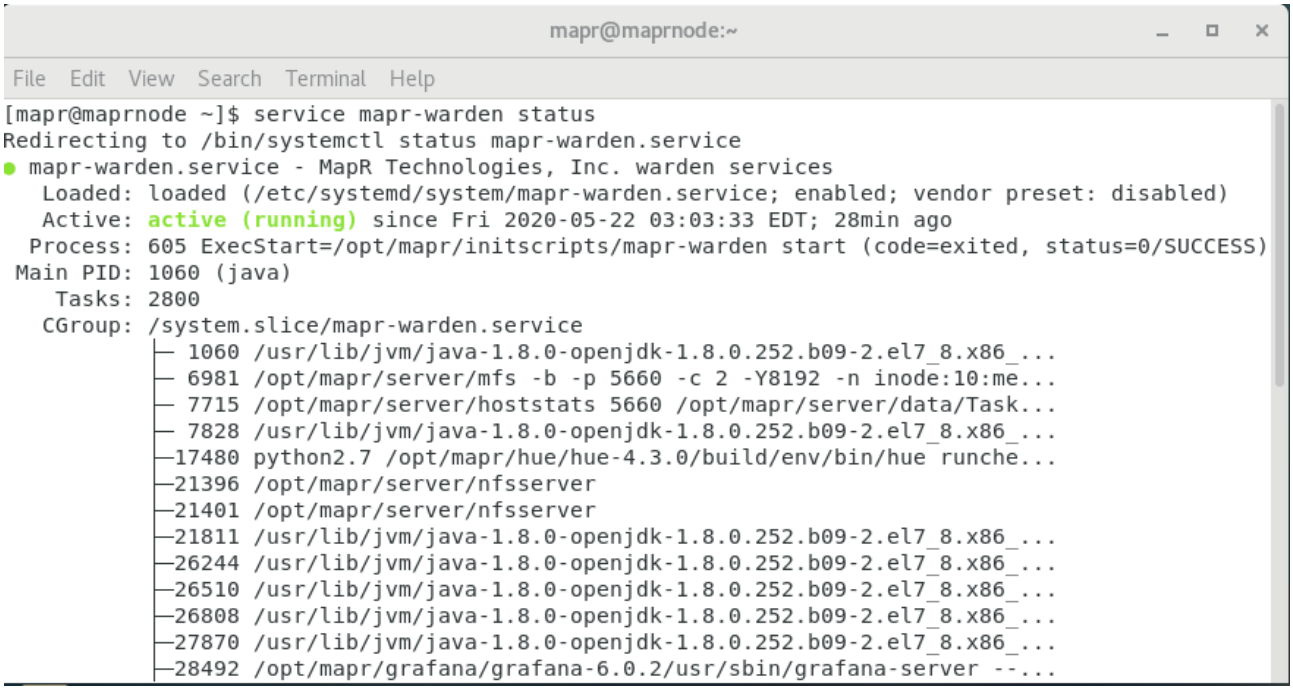

Fig. 12: Comprobación del correcto funcionamiento de MapR Warden

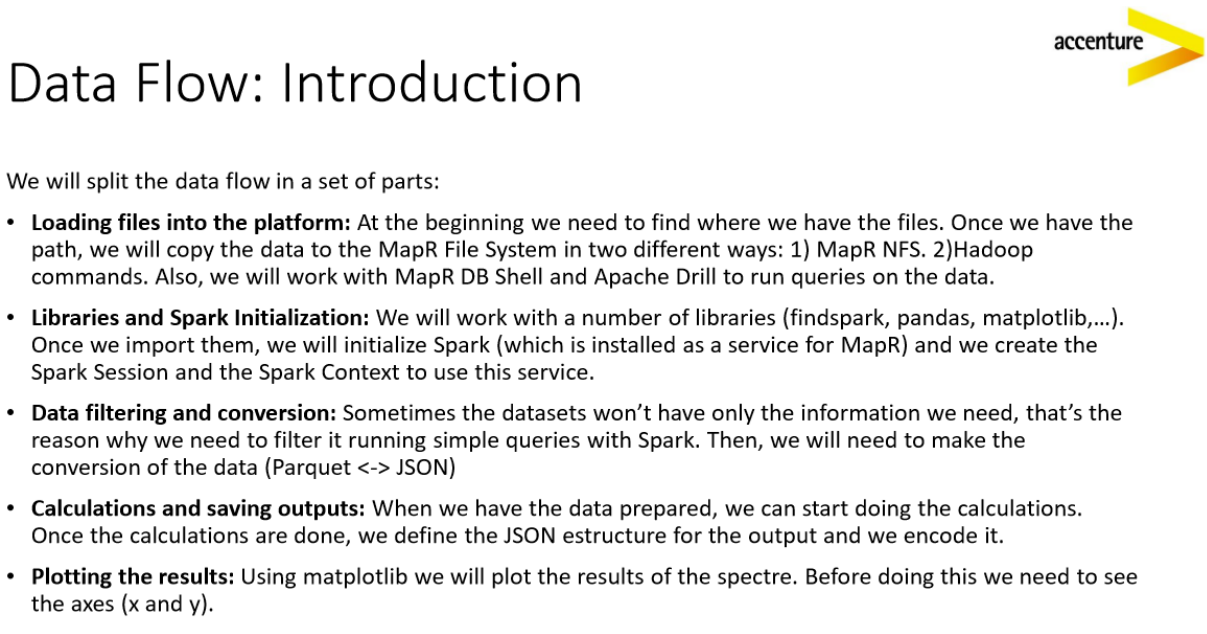

Fig. 13: Documentación para los futuros usuarios de la plataforma

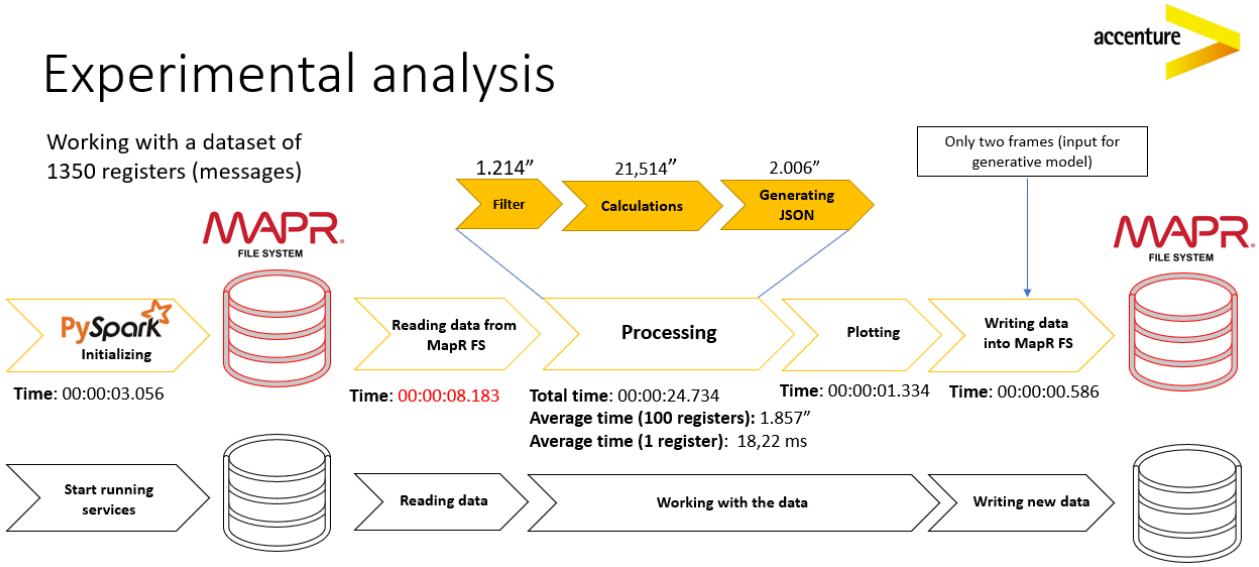

Fig. 14: Análisis de tiempos durante el tratamiento de los datos una vez se encuentran en la plataforma I

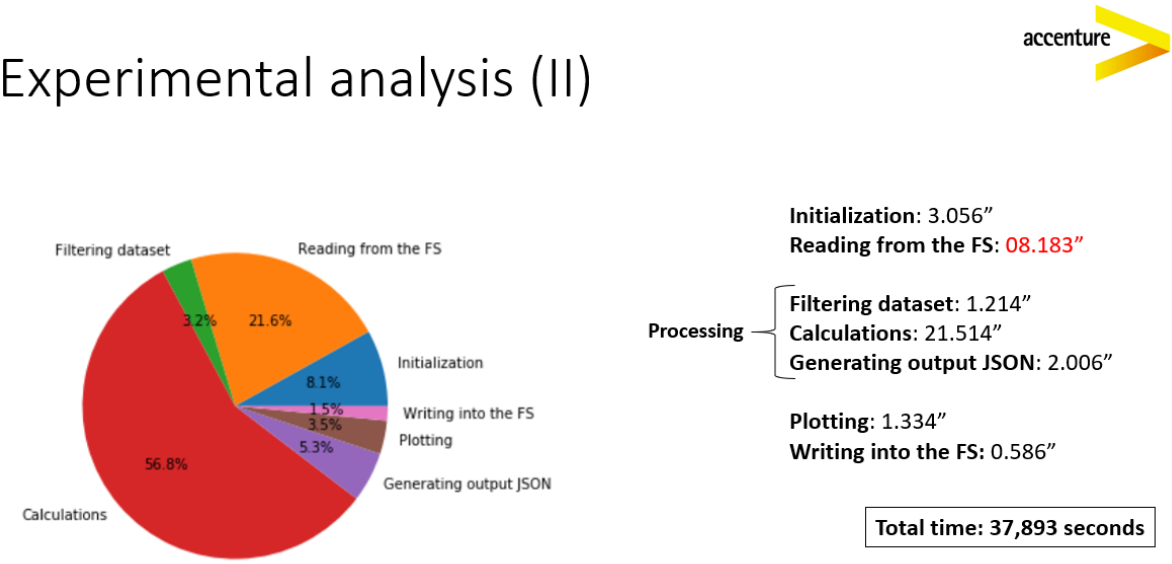

Fig. 15: Análisis de tiempos durante el tratamiento de los datos una vez se encuentran en la plataforma II